SOFTWARE-BASED GRADIENT NONLINEARITY DISTORTION CORRECTION

A Thesis

Presented to the

Faculty of

California State University,

San Bernardino

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

in

Computer Science

---

by

Thomas Seward Lee

December 2006

SOFTWARE-BASED GRADIENT NONLINEARITY DISTORTION CORRECTION

A Thesis

Presented to the

Faculty of

California State University,

San Bernardino

by

Thomas Seward Lee

December 2006

Approved by:

Keith Evan Schubert, Advisor	Date
Ernesto Gomez	
Arturo Concepcion	
Reinhard W. Schulte	

ABSTRACT

A new system for functional proton radiosurgery has been proposed, and is currently under development at Loma Linda University Medical Center, Loma Linda, CA. The goal of the system is to target specific brain areas with high doses of proton beams with submillimeter accuracy. High-energy proton beams have exquisitely sharp lateral penumbra and are, therefore, ideal for functional radiosurgery. Localizing the anatomical target with an MRI-based fiducial system requires correction of gradient nonlinearity distortions inherent in the scanner images. Modern MR scanners are particularly prone to such distortions due to wider bores and stronger gradient fields. The gradient nonlinearity correction described in this work is based on a high-resolution 3D MR scan of a cube phantom. Using a least-squares fitting procedure correction parameters are found that convert the geometrically warped planes of the cube into the ideal planes. The accuracy of the correction method is then tested and verified using a second phantom containing targeting markers. The locations of the markers reported by the software are compared to their actual measured locations, and the difference between the measurements quantify the accuracy of the distortion correction method.

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank the individuals who played a crucial role in my college education.

First and foremost, I give my most sincere thanks and appreciation to my Lord, Jesus Christ. Without His knowledge, strength, wisdom, and patience, this thesis—and my graduate education—would not have been possible. Philippians 4:13 states that *I can to all things in Christ, which strengtheneth me.* This thesis is a testament to this fact. Through many trials and tribulations, He has always been by my side, and I know He will continue to be there for me.

I would like to express my gratitude for Dr. Reinhard Schulte, for the hours upon hours of dedication in guiding me through my graduate research. I am very thankful for his sacrifice and faithfulness. This thesis would not be possible without his commitment. The time that I have spent with Dr. Schulte has enabled me to greatly increase my knowledge in mathematics, physics, and radiation medicine.

I would also like to express my wholehearted appreciation for my advisor and mentor, Dr. Keith Schubert, for providing me with the necessary tools and knowledge to succeed both as a student and as a computer scientist. Working alongside Dr. Schubert has opened my eyes to the wonderful world of science and mathematics, and has increased my appreciation and desire for scientific knowledge. I am very grateful for his encouragement, advice, and assistance throughout my time at Cal State San Bernardino.

My thanks also go to the computer science faculty at Cal State San Bernardino, for the knowledge, experience, and wisdom I possess today. Most specifically, I would

like to thank Dr. Arturo Concepcion for all his advice and assistance throughout my experience as a student at Cal State San Bernardino. My thanks also go to Dr. Ernesto Gomez, for his constructive criticism and encouragement during my graduate research.

Lastly, but certainly not least, I express endearing thanks for my family. I am deeply thankful for my mother, and her undying love for me. She has brought me through thick and thin, and always encouraged me through the most difficult times in my life. My accomplishments are all thanks to her faith and commitment. I am extremely grateful for my father, who gave me the wisdom and strength to succeed in life. The inspiration and vision he provides is so valuable to me. And finally, I am grateful for Christie. Her unwaivering love, support, and encouragement has enabled me to accomplish so much in so little time.

DEDICATION

To Christie, my love.

TABLE OF CONTENTS

Abstract iii

Acknowledgements iv

List of Tables xiv

List of Figures xvii

1. Introduction 1

1.1 Thesis Overview 1

1.2 Protons for Therapeutic Purposes 4

1.2.1 Wilson's Pioneering Work 5

1.2.2 Unique Properties of Protons 5

1.2.3 Limitations of Protons 7

1.2.4 Conclusions from Wilson's Work 9

1.2.5 Further Research in Protons 10

1.2.6 Problems Still Remain 11

1.3 Loma Linda University Medical Center: Pioneering Proton Therapy . 12

1.3.1 The (Current) System 17

1.3.2 Proposal for a New Proton Radiosurgery System 18

1.4 Computed Tomography 21

1.4.1 History and Development 22

1.4.2 Principles of Operation 25

1.4.3 Characteristics of CT 30
1.5 Magnetic Resonance Imaging 35
1.5.1 History and Development 35
1.5.2 Principles of Operation 36
1.5.3 Characteristics of MRI 44
1.6 Inherent Distortions in MR Images 50
1.6.1 Sources of MRI Distortions 50
1.7 The Enemy: Gradient Nonlinearity Distortion 54
1.7.1 Sources of Gradient Nonlinearity Distortion 54
1.7.2 Main Magnetic Field 55
1.7.3 X and Y Gradients 60
1.7.4 Shim Coils 61
1.7.5 Total Magnetic Field 63
1.8 Previous Works 65
1.8.1 Scanner Correction 65
1.8.2 Fusing CT and MRI 67
1.8.3 Using Pre-Defined Specifications and Parameters 68
1.8.4 Calculating Distortion Correction Indirectly 70

2. Software-Based Gradient Nonlinearity Distortion Correction 73
2.1 Approaching the Problem 75
2.2 Experimental Setup 76
2.3 Method 78
2.3.1 Data Quality Assurance (Preprocessing) 80
2.3.2 Plane Calculation 89
2.3.3 Calculate Distortion Correction Parameters 96
2.4 Testing and Verification of Methods 100

2.5 Summary of the Software System 102

3. Step 1: Data Quality Assurance (Preprocessing) 105

3.1 Purpose 105

3.2 Application 106

3.3 Algorithm Analysis 108

3.3.1 Loading the Data 109

3.3.2 Invoking the "Process" Function 113

3.3.3 Determining Useful Slices 115

3.3.4 Performing Edge Detection 120

3.3.5 Removing Drain Plug 126

3.3.6 Verifying Edges 134

3.3.7 Removing Bubbles 140

3.3.8 Verifying Corners 151

3.3.9 Repairing Holes 159

3.3.10 Verify Points 166

3.3.11 Calculating Rotational Tilt 170

3.3.12 Calculating Center Offset 173

3.3.13 Auxiliary: Get Corners 177

3.3.14 Auxiliary: Follow Edges 184

4. Step 2: Plane Calculation 188

4.1 Midplane Calculation 188

4.1.1 Purpose 188

4.1.2 Application 188

4.2 Ideal Plane Calculation 192

4.2.1 Purpose 192

4.2.2 Application 193

4.3 Algorithms . . . 200
4.3.1 Invoking the "Calculate Planes" Algorithms . . . 201
4.3.2 Reading Edge Images . . . 203
4.3.3 Calculating Midpoints and Preparing Midplane Data . . . 206
4.3.4 Remove Solitary Points . . . 214
4.3.5 Pack Edge Points . . . 218
4.3.6 Construct Data Sets . . . 223
4.3.7 Concatenate Data Sets . . . 225
4.3.8 Fit Midplanes . . . 228
4.3.9 Shift Midplanes / Create Ideal Planes . . . 234

5. Step 3: Distortion Estimation and Correction . . . 238
5.1 Distortion Estimation and Correction . . . 238
5.1.1 Purpose . . . 238
5.1.2 Application . . . 239
5.2 Algorithms . . . 245
5.2.1 Invoking the "Calculate Distortion" Algorithms . . . 246
5.2.2 Pack Edge Points . . . 248
5.2.3 Create Edge Data Sets . . . 250
5.2.4 Create Face Data Sets . . . 252
5.2.5 Calculate Theoretical Undistorted Data . . . 256
5.2.6 Calculate Distortion Correction Model . . . 261

6. Results . . . 267
6.1 Introduction . . . 267
6.2 Distortion Correction Numerical Analysis . . . 271
6.2.1 7-17-05 Study . . . 273
6.2.2 10-2-05 Study . . . 282

6.2.3 11-6-05 Study 291
6.2.4 4-30-06 Study 300
6.2.5 6-19-06 Study 311
6.2.6 Calculated Distortion Correction Parameters 322
6.3 Distortion Correction Image Analysis 326
6.3.1 7-17-05 Study 326
6.3.2 10-2-05 Study 330
6.3.3 11-6-05 Study 335
6.3.4 4-30-06 Study 339
6.3.5 6-19-06 Study 343
6.4 Data Simulation 348
6.4.1 Producing a Simulated Data Set 349
6.4.2 Data Simulations 352
6.4.3 Conclusions 362
6.5 Filtered vs. Unfiltered MR Images 365
6.5.1 Testing Filtered and Unfiltered Images 366
6.6 Performance Benchmarks 399
6.6.1 Impetus 400
6.6.2 Test Setup 401
6.6.3 Benchmark Results 404
6.6.4 Conclusions 408
6.7 Accuracy of Target Localization 415
6.7.1 Localizing Targeting Markers 415
6.7.2 Scan Details 417
6.7.3 Specifications of Lucy Phantom 419
6.7.4 Numerical Results 420
6.7.5 Error Analysis 422

6.7.6 Conclusion 424

7. Summary, Conclusions, and Recommendations 431

7.1 Primary Sources of Error, and Solutions 433

7.1.1 Phantom Centering 437

7.1.2 Effects of Magnetic Shimming 444

7.1.3 Stereotactic Coordinate Frame 453

7.1.4 Metallic Objects 455

7.1.5 A Metallic Stereotactic Coordinate Frame? 457

7.1.6 Conclusion 460

7.2 Success or Failure? 460

7.2.1 Meeting LLUMC Requirements 461

7.2.2 Accuracy of Distortion Correction 462

7.2.3 Accuracy of Target Localization 464

7.2.4 Software Performance 466

7.2.5 Hardware Compatibility 469

7.2.6 Scanner Filtering 470

7.3 Comparison of Other Solutions 473

7.3.1 Previous Works 474

7.4 Suggestions for Improvement 484

7.4.1 Phantom Centering Techniques 485

7.4.2 MRI Scanner Configuration 488

7.4.3 Image Quality 490

7.4.4 Programming Languages/Environments 491

7.4.5 Time 493

7.5 Future Work 495

7.5.1 Increasing User-Friendliness 496

7.5.2 Porting to Different Languages 499
7.5.3 Supporting Unix-Based Operating Systems 501
7.5.4 The Calibration Phantom 502
7.5.5 The Metallic Stereotactic Coordinate Frame 505
7.5.6 Further Testing and Verification 507
7.6 Final Remarks 508

Appendix 511

A. Error Analysis 512

B. Least Squares 514
B.1 Introduction 514
B.2 Characteristics of Linear Least Squares 515
B.3 Characteristics of Distortion Correction Problem 517
B.3.1 Overdetermined Systems 517
B.3.2 Data Fitting 519
B.3.3 Linear System 520
B.4 Process of Least Squares 520
B.4.1 Orthogonal Transformation 522
B.4.2 Implementing Orthogonal Transformation 524
B.5 Computer Algorithm 529
B.6 Conclusion 532

C. Stereotactic Transformation 534
C.1 Stereotactic Frame 534
C.2 Stereotactic Transformation: Background 536
C.3 Mathematical Basis 537

C.3.1 Coordinate Translation 538
C.3.2 Coordinate Rotation 540
C.3.3 The Final Expression 545
C.3.4 Verifying Correctness 545

D. Edge Detection 548
D.1 Theory 548
D.2 Application 550
D.2.1 Sobel 551
D.2.2 Prewitt 556
D.2.3 Roberts 558
D.2.4 Laplacian of Gaussian (LOG) 560
D.2.5 Zero-Cross 567
D.2.6 Canny 568
D.3 Conclusion 572

LIST OF TABLES

1.1 Radiodensities of common media. [1] 29
1.2 Spin numbers of common nuclei. [2] 37

6.1 Plane coefficients of midplane fitting in 7-17-05 study. 274
6.2 Plane coefficients of ideal planes fitting in 7-17-05 study. 275
6.3 Errors calculated from plane fitting results from 7-17-05 study. 276
6.4 Errors calculated from distortion correction results from 7-17-05 study. 278
6.5 Plane coefficients of midplane fitting in 10-2-05 study. 283
6.6 Plane coefficients of ideal planes fitting in 10-2-05 study. 283
6.7 Errors calculated from plane fitting results from 10-2-05 study. 284
6.8 Errors calculated from distortion correction results from 10-2-05 study. 288
6.9 Plane coefficients of midplane fitting in 11-6-05 study. 293
6.10 Plane coefficients of ideal planes fitting in 11-6-05 study. 293
6.11 Errors calculated from plane fitting results from 11-6-05 study. 294
6.12 Errors calculated from distortion correction results from 11-6-05 study. 297
6.13 Plane coefficients of midplane fitting in 4-30-06 study. 302
6.14 Plane coefficients of ideal planes fitting in 4-30-06 study. 302
6.15 Errors calculated from plane fitting results from 4-30-06 study. 303
6.16 Errors calculated from distortion correction results from 4-30-06 study. 307
6.17 Plane coefficients of midplane fitting in 6-19-06 study. 312
6.18 Plane coefficients of ideal planes fitting in 6-19-06 study. 312
6.19 Errors calculated from plane fitting results from 6-19-06 study. 313

6.20 Errors calculated from distortion correction results from 6-19-06 study. 318
6.21 Distortion correction parameters for all studies, part 1. 322
6.22 Distortion correction parameters for all studies, part 2. Also includes Langlois' parameters. 323
6.23 Coefficients of distortion model used for all trials. 352
6.24 Plane coefficients of midplanes fitting in Trial #1. 353
6.25 Plane coefficients of ideal planes fitting in Trial #1. 353
6.26 Errors calculated from plane fitting results from Trial #1. 354
6.27 Errors calculated from distortion correction results from Trial #1. . . 368
6.28 Plane coefficients of midplanes fitting in Trial #2. 369
6.29 Plane coefficients of ideal planes fitting in Trial #2. 369
6.30 Errors calculated from plane fitting results from Trial #2. 369
6.31 Errors calculated from distortion correction results from Trial #2. . . 370
6.32 Plane coefficients of midplanes fitting in Trial #3. 371
6.33 Plane coefficients of ideal planes fitting in Trial #3. 371
6.34 Errors calculated from plane fitting results from Trial #3. 371
6.35 Errors calculated from distortion correction results from Trial #3. . . 372
6.36 Plane coefficients of ideal planes fitting in 4-30-06 study, with scanner correction. 375
6.37 Midplane fitting results from 4-30-06 study, with scanner correction. . 376
6.38 Distortion correction results from 4-30-06 study, with scanner correction. 379
6.39 Plane coefficients of ideal planes fitting in 6-19-06 study, with scanner correction. 387
6.40 Plane fitting results from 6-19-06 study, with scanner correction. . . . 388
6.41 Distortion correction results from 6-19-06 study, with scanner correction. 391
6.42 RMS (standard deviations) (in mm) of distortion correction results for 4-30-06 and 6-19-06 studies, filtered and unfiltered. 395

6.43 Distortion correction parameters for 4-30-06 and 6-19-06 studies, filtered and unfiltered. 396
6.44 Configurations of test machines. 402
6.45 Errors calculated from localizing fiducials in uncorrected and corrected MR images. 421
6.46 Actual and measured locations of the Lucy markers in uncorrected (U) and corrected (C) MR images, in mm. 428
6.47 Actual and measured locations of the Lucy markers in CT images, in mm. 429
6.48 Errors calculated from localizing Lucy markers in CT and MR images, with and without distortion correction. 430

7.1 Calculation error of midplane fitting in 11-6-05 study. 445
7.2 Plane coefficients of midplane fitting in 11-6-05 study. 445
7.3 Accuracy of theoretical undistorted points in 11-6-05 study. 446
7.4 Results of distortion correction in 11-6-05 study. 447

LIST OF FIGURES

1.1 Dosage peaks of 250 MeV proton and 6 MV photon beams. 7

1.2 Coordinate system for tomographic reconstruction. 26

1.3 Loss in image resolution due to image reconstruction. 33

1.4 Distortion inherent in MR images caused by gradient field nonlinearity [3]. 55

1.5 Magnetic fields produced by Maxwell (a) and Helmholtz (b) coils. . . 57

1.6 Helmholtz coils configuration. 59

1.7 Golay coil for producing linear field gradients in B_z along the x or y axes. l = 3.5a, d = 0.775a and Φ = 120 degrees. [2]. 61

2.1 Phantoms in the Siemens Sonata MRI scanner. 76

2.2 Examples of useful slices of oil-filled phantom. 78

2.3 Example scans of Lucy phantom, demonstrating MRI markers (center) and fiducial points (edges). 79

2.4 Slice exhibiting oil leakage. 84

2.5 Slices containing a large air bubble. 85

2.6 Demonstration of bubble correction on large corner bubble. 88

2.7 The upper and lower edge points of the image are averaged to form two midlines, shown as the horizontal and vertical lines intersecting in the middle of the image. A plane is then fit to all the slices' midlines. 90

2.8 Distorted faces produced by MR scanner, and the ideal planes describing the phantom that produced them. The axis dimensions are pixels, and note the curvature is exaggerated so that it is visible. 93
2.9 Example scans of Lucy phantom, demonstrating MRI markers (center) and fiducial points (edges). 101

3.1 The DICOM coordinate system. 112
3.2 Examples of useful slices of oil-filled phantom. 115
3.3 Examples of slices that do not contain useful data. (b) and (c) contain only a partial view of the phantom, while (a) does not contain the phantom at all. 116
3.4 Differences in threshold strengths for Canny edge detection. 122
3.5 Edge image of phantom, with large air bubble 124
3.6 Typical slices with images of the phantom's drain plug. 127
3.7 Result of circular convolution on various slices. 129
3.8 Detected edges on a slice with the drain plug. The plug appears at the center of the top and bottom edges. 130
3.9 Edge image after plug correction. 132
3.10 The adverse effects of phantom oil leakage. 136
3.11 Slices containing a large air bubble. 137
3.12 Small and large air bubbles, as they appear in a slice. 141
3.13 Pixel shifts across a 5-point span in 2 edges. Note the peaks of (a) are one pixel, while the peaks in (b) are four pixels. 143
3.14 Pixel shifts across a 5-point span in an edge with a large bubble. The positive peak is at 13 pixels, and the negative peak is at 8 pixels. . . 145
3.15 Pixel shifts of slice with large air bubble from Figure 3.14, with the edge mirrored horizontally. 146
3.16 Demonstration of bubble correction on small edge bubble. 149

3.17 Demonstration of bubble correction on large corner bubble. 151
3.18 Edge image with noisy corner. 153
3.19 Corner correction on edge image with corrupted corner. 158
3.20 Edge with a 1-pixel hole. 160
3.21 Edge image with 2 large holes, caused by bubble removal. 160
3.22 Demonstration of hole repair on a severely perforated image. 165
3.23 Extraneous point below an edge. 167
3.24 Edge image with corrupted corner. 171
3.25 Example of phantom offcentering in MRI scanner 174
3.26 Edge image with corrupted corner. 175
3.27 Typical slices from various MRI scans. 178
3.28 Looks can be deceiving! An air bubble that can be mistaken for a corner of the phantom. 182
3.29 Typical corners in edge images. 182

4.1 The upper and lower edge points of the image are averaged to form two midlines, shown as the horizontal and vertical lines intersecting in the middle of the image. A plane is then fit to all the slices' midlines. 189
4.2 DICOM coordinate system. 219
4.3 Midplane fitting results. 232

5.1 Distorted faces produced by MR scanner (black), and the theoretical undistorted data sets calculated by create_undist_data() (grey). The axis dimensions are pixels, and note the curvature in the distorted faces is exaggerated so that the difference between the two faces is visible. . 259

6.1 Distorted data points (black) from the 7-17-05 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm. 277

6.2 Corrected data points (black) from the 7-17-05 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm. 279
6.3 Distorted data points (black) from the 10-2-05 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm. 287
6.4 Corrected data points (black) from the 10-2-05 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm. 289
6.5 Distorted data points (black) from the 11-6-05 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm. 296
6.6 Corrected data points (black) from the 11-6-05 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm. 298
6.7 Distorted data points (black) from the 4-30-06 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm. 306
6.8 Corrected data points (black) from the 4-30-06 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm. 308
6.9 Distorted data points (black) from the 6-19-06 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm. 317
6.10 Corrected data points (black) from the 6-19-06 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm. 319

6.11 Example images from the 7-17-05 study. 326
6.12 Axial and coronal images from the 7-17-05 study, before and after distortion correction. 328
6.13 Example images from the 10-2-05 study. 330
6.14 Axial, coronal, and sagittal images from the 10-2-05 study, before and after distortion correction. 331
6.15 Example images from the 11-6-05 study. 335
6.16 Axial, coronal, and sagittal images from the 11-6-05 study, before and after distortion correction. 336
6.17 Example images from the 4-30-06 study. 339
6.18 Axial, coronal, and sagittal images from the 4-30-06 study, before and after distortion correction. 340
6.19 Example images from the 6-19-06 study. 343
6.20 Axial, coronal, and sagittal images from the 6-19-06 study, before and after distortion correction. 344
6.21 Axial, coronal, and sagittal images from the 4-30-06 study, with and without scanner filtering. 367
6.22 Axial, coronal, and sagittal images from the 4-30-06 study with scanner filtering, before and after distortion correction. 373
6.23 Distorted data points (black) from the filtered 4-30-06 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm. 378
6.24 Corrected data points (black) from the filtered 4-30-06 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm. 380
6.25 Axial, coronal, and sagittal images from the 6-19-06 study, with and without scanner filtering. 384

6.26 Axial, coronal, and sagittal images from the 6-19-06 study with scanner filtering, before and after distortion correction. 385
6.27 Distorted data points (black) from the filtered 6-19-06 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm. 390
6.28 Corrected data points (black) from the filtered 6-19-06 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm. 392
6.29 Execution times of Super PI on the various test machines. 405
6.30 Execution times of distortion correction software with two (a) and three (b) studies on the various test machines. 407
6.31 Lucy phantom affixed to stereotactic frame, resting in head coil in MRI scanner. 416
6.32 Example scans of Lucy phantom ((a) and (b)). Observe the three dots produced by the stereotactic frame, on the outer edges of the images. The third dot indicates the position of the slice, relative to the frame. 417
6.33 Localizing target regions in the Odyssey® Radiation Treatment Planning Software. 418
6.34 Detailed schematic drawing of Lucy phantom, demonstrating provisions for targeting markers. 420

7.1 Axial slice from 11-6-05 study, with a box drawn around the image of the phantom. Observe how the lower left corner is not collinear with upper left corner, indicating a problem. 449
7.2 Axial studies identifying source of discrepancy in lower left corners of both images. 451

7.3 Example images produced with calibration of the shim coils (b) and without (a). A box is drawn around the image of the phantom. Notice the location of the lower left corner in each image. 452
7.4 Leksell® Coordinate Frame Model G. 454
7.5 Coronal scan of Lucy phantom and Leksell® MR Indicator Frame, after gradient nonlinearity distortion correction. Observe the slight curvature at the bottom of the fiducial rods, causing the distance of the rods to be inconsistent. 458

B.1 Graphical representation of Householder transformations, demonstrating the transformation as reflections [4]. 526

C.1 Leksell® MR Indicator frame. 535
C.2 Leksell® Coordinate Frame Model G. 536
C.3 Example scans of Lucy phantom ((a) and (b)). Observe the three dots produced by the stereotactic frame, on the outer edges of the images. The third dot indicates the position of the slice, relative to the frame. 537
C.4 Lucy phantom affixed to stereotactic frame, resting in head coil in MRI scanner. 538
C.5 Two marker sets in global and local reference systems. 539
C.6 Vector products to derive M_A. 542
C.7 Vector products to derive M_B. 544

D.1 Example signal. 549
D.2 First Derivative of Example Signal. 549
D.3 Second Derivative of Example Signal. 550
D.4 Sample MRI Slice. 551
D.5 Results of Sobel with Default Threshold Value = 2.0554×10^{-4}. . . . 554
D.6 Results of Sobel with Threshold Value = 3.0830×10^{-4}. 555

D.7 Results of Prewitt with Default Threshold Value = 1.9985×10^{-4}. . . 557
D.8 Results of Prewitt with Threshold Value = 3.0830×10^{-4}. 558
D.9 Results of Roberts with Default Threshold Value = 2.8711×10^{-4}. . . 561
D.10 Results of Roberts with Threshold Value = 4.5937×10^{-4}. 562
D.11 Results of LOG with Default Threshold Value = 5.9078×10^{-6}. . . . 564
D.12 Results of LOG with increased threshold values. 565
D.13 Results of LOG with Threshold = 3.0130×10^{-5}. 566
D.14 Results of Canny with Default Threshold Values = 0.0188 and 0.0469. 570
D.15 Results of Canny with increasing threshold values.. 571
D.16 Results of Canny with Thresholds = 0.0544 and 0.1359. 572

1. INTRODUCTION

1.1 Thesis Overview

The primary purpose of this thesis is to discuss the use of Magnetic Resonance Imaging in functional proton radiosurgery. While this subject is a familiar topic in the medical community, this thesis will serve as a conclusive, consolidated reference for MRI and its use in functional proton radiosurgery.

The methods presented in this thesis were specifically designed to correct *gradient nonlinearity distortion,* the single greatest hurdle that limits the deployment of MRI-based functional proton radiosurgery systems. Although there exist current systems that successfully treat patients, this new system for proton radiosurgery will fully utilize MRI to provide localization of anatomical targets with submillimeter accuracy. The work completed in this thesis was performed under the direct supervision of Dr. Reinhard W. M. Schulte, M.D., at Loma Linda University Medical Center (LLUMC), Loma Linda, CA.

This thesis provides analysis and solutions to the problems related to gradient nonlinearity distortion. The following questions are addressed:

1. To what extent do gradient nonlinearity distortion affect MR images?

2. How does the proposed method differ from current implementations, and where

have improvements been made in this work?

3. What accuracy can be obtained from corrected images? Does this meet or surpass the established goals of the project?

To fully understand the impetus of this work, a thorough introductory analysis is provided, to fully encompass the ideas presented. First, the characteristics of proton radiosurgery are introduced, in addition to a discussion of its advantages over other current methods of radiation oncology. A historical background for proton radiosurgery is also presented, along with a description of its implementation at LLUMC. For the past two decades, LLUMC has overseen the development of the latest generation of proton radiosurgery systems. The work in this thesis enables LLUMC to move closer to the completion of its latest proton radiosurgery system, and achieve the goal outlined in its motto: *"To Make Man Whole."*

LLUMC has chosen to use MRI in its latest proton radiosurgery system. To provide further understanding of the system, and how it differs from current systems, a discussion covering CT and MRI is presented. This also includes an in-depth analysis of the imaging techniques utilized by each. These analyses present theoretical and mathematical foundations concerning the operation of each system. Finally, the sources of gradient nonlinearity distortion in MRI are analyzed, to provide insight as to the purpose of this software package. Analyses of current distortion correction methods is also presented.

Following the discussion of the intricacies of MRI and gradient nonlinearity distortion, the distortion correction method is presented. The individual steps in the method are described in great detail, to provide comprehensive analysis of the prob-

lem and the methods used to solve it. An algorithm analysis is provided for each function used in the correction process. This analysis strays from a conventional algorithm analysis, in that it does not involve computational complexities; therefore, the analysis will make no mention of algorithmic notation (i.e., big-oh, big-theta, big-omega). Despite this characteristic, the analysis will provide the following:

- Overview of the method employed to solve each particular problem.
- Purpose and theory behind the design and implementation of the algorithms.
- Organization and manipulation of various data types in numerous data structures.
- Step-by-step description of the computations performed.
- Results produced by the algorithms.

The structure and content of the analysis is provided to aid future work in this field, such that these methods can be easily reproduced and repeated to achieve similar results. Additionally, the design serves as a blueprint for future endeavors, simplifying the task of porting the software system to various languages and platforms.

After the presentation of the methods employed in the correction process, the results of the correction are presented. As part of the data analysis, results obtained from a simulated ideal data set are also presented. This simulation aids in the identification of sources of systematic errors, in addition to providing a strong metric with which to compare real results. Sources of systematic error are discussed, and the methods to account for and eliminate these errors are also presented.

An outline of the conclusions of this thesis follows, as well as applications for the contributions of this project to the research field. A comparison between this method and other published methods also follows, as well as a theoretical analysis of obtained results. Avenues for future work and research are discussed, to aid in the continuation of this work. Since MRI and functional proton radiosurgery are ever-evolving fields, research and development will definitely continue in both of these areas. The conclusion will touch on these subjects, to direct future research in this area.

1.2 Protons for Therapeutic Purposes

Historically, surgery has been characterized as an invasive practice to treat diseases or injuries in the human body. Over the past several decades, medical research has concentrated on the investigation of noninvasive surgery techniques, eliminating the need to perform incisions or remove biological tissue. The benefits of noninvasive surgery are numerous, among these include:

- Reduced patient recovery period.
- Reduction of post-surgical complications.
- Safer treatment of diseases and disorders.
- Increased accuracy and confidence of treatment.
- Provides treatment for previously untreatable or inoperable disorders and diseases.

1.2.1 Wilson's Pioneering Work

Noninvasive surgery is made possible by the use of high-energy particles. The idea of utilizing high-energy particles for medical purposes has been the topic of medical research for nearly 60 years. The pioneering work in this area was published in 1946 by Robert R. Wilson. His work, *Radiological Use of Fast Protons* [5], details how protons and other high-energy particles above 125 MeV (million electron volts) can easily penetrate human tissue. The ideas and research presented in Wilson's work serve as the cornerstone for nearly all research conducted in noninvasive stereotactic radiosurgery.

1.2.2 Unique Properties of Protons

The most useful property of protons is the ability to penetrate human tissue. The depth of tissue penetration is directly proportional to the energy of the protons generated by the accelerator. Therefore, delivering a dose of protons to any specific anatomical region requires a certain energy level. The range of a proton in air is given by the following equation:

$$R = (E/9.29)^{1.8} \tag{1.1}$$

where E is energy expressed in MeV. The range of a proton in tissue is 1.11×10^{-3} times that of its range in air [5]. For example, a proton energized to 125 MeV has the capability to penetrate 12 cm of tissue. Increasing the energy level to 200 MeV will allow the proton to penetrate 27 cm of tissue. Therefore, properly energized protons have the ability to penetrate any region in the human body, allowing for treatment

at any specific location.

A proton *ionizes* human tissue and deposits a dose along its path. The ionization of tissue occurs as a result of energy loss from the proton. Protons stop after they have lost all their energy. The distance traveled by the protons is proportional to the ionization per centimeter of path, also known as the *specific ionization*. Additionally, the dosage required to ionize a certain region of tissue is inversely proportional to the energy of the proton. Ionization of tissue results in the formation of ions and free radicals, produced usually from water or other biological materials present in the tissue. The production of ions and free radicals causes severe damage to cells, leading to cell death. The cells that receive the effects of ionization are only those that lie in the target location, that is, in the location of specific ionization. Surrounding cells remain unharmed.

A proton's energy decreases as it ionizes tissue. During this process, the interaction cross section of the proton increases, meaning that the likelihood of the proton's interaction with surrounding particles becomes higher. As the proton's energy and velocity approach zero, a peak in the dosage occurs. This phenomenon is known as the *Bragg Peak*. Wilson directly observed this through his experiments, stating that the dosage delivered by protons is many times less at the surface than it is in the final portion of the proton's path. Figure 1.1 illustrates the peak dosages delivered by proton and photon beams. Photon beams deliver strong dosages over a large area very near the surface, and are unable to provide high dosages to deep tissue. Protons, on the other hand, deliver small dosages near the surface, minimizing effects on healthy tissue. At the end of the proton's path, deep within tissue, the proton delivers its full

dosage over a small area.

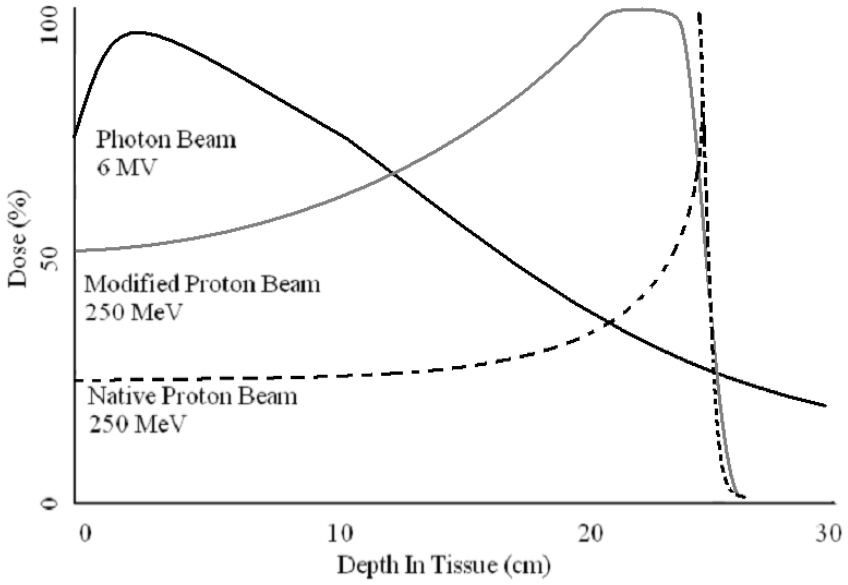


Fig. 1.1: Dosage peaks of 250 MeV proton and 6 MV photon beams.

Wilson's research concluded that the path of a proton is *"very nearly a straight line"* once it has penetrated tissue [5]. This unique property is very advantageous for noninvasive surgery, since a proton can be guided to a localized anatomical target with high confidence, without the need to worry about particle deflection. Utilization of this property would give the physician one very powerful tool: control. As long as the physician could properly identify and localize a target region, delivering the dosage to that region could be accomplished, while leaving surrounding regions intact.

1.2.3 Limitations of Protons

Even though Wilson concluded that the path of a proton is nearly straight, protons will always vary slightly from straight-line paths while traveling through tissue; however, Wilson did prove that the path is straight enough to hit a target location with

very high accuracy. Additionally, Wilson observed that the stopping location of protons is not exactly the same, given a set of protons energized to the exact same level. The factors that contribute to these phenomena are the results of the characteristics of the path protons follow through tissue.

The first effect is ionization. Recall that protons ionize surrounding tissue as they travel. The ionization of tissue is not constant for all protons, but rather a statistical relationship. Therefore, Wilson found that the stopping locations of protons is also not constant, but can be described by a statistical distribution, given by:

$$P(x)dx = \frac{R}{\alpha\sqrt{\pi}} e^{-\frac{(R-x)^2}{R^2\alpha^2}} \tag{1.2}$$

where x is the depth of tissue (distance below the surface), and

$$\alpha = \frac{7.1}{E_0^{\frac{1}{2}}} \left(\frac{NZz^2R}{E_0} \right)^{-0.055} \tag{1.3}$$

where N is the number of atoms per cubic centimeter, Z is the atomic number, z is the ion charge number, E_0 is the rest energy of the ion in MeV, and R is the range in centimeters. To sum up this relation in simple terms, the horizontal spread of the resting locations of protons is about 1% of the initial range.

The second effect is due to particle collision. As protons penetrate tissue, small angle scatterings occur between the protons and nuclei of the atoms in the tissue. A collision results in a slight change in the proton's direction of travel, but does not alter its energy level. This effect is known as multiple scattering, and its effect is a spreading out of the beam. Wilson summarized this effect with the following:

$$P(y)dy = \frac{R}{\beta\sqrt{\pi}} e^{-\frac{-y^2}{R^2\beta^2}} dy \tag{1.4}$$

where y is the distance from the average end of the range measured normal to the beam direction, and β is given by:

$$\beta = 12 \left(\frac{Z}{E_0}\right)^{\frac{1}{2}} \left(\frac{NRZz^2}{E_0}\right)^{-0.055} \tag{1.5}$$

1.2.4 Conclusions from Wilson's Work

Through these considerations, Wilson concluded that the aforementioned effects do not greatly alter the paths of protons, although these effects do indeed influence the travel of protons through tissue. With this bearing in mind, Wilson theorized that a location as small as 1.0 cm^3 could be treated using this method. Unfortunately, the technology was not available for Wilson to test his hypothesis, but his numerical analyses provide sufficient confidence. Additionally, of particular interest, as it pertains to the work described in this thesis, is the following statement:

"It will be simple to collimate proton beams to less than 1.0 mm. diameter or to expand them to cover any area uniformly." [5].

Wilson's research and observations confirm that it is very feasible to deliver protons to a localized anatomical target as small as 1.0 cm^3 with submillimeter accuracy. Although Wilson was not able to fully test his theories, his work provided insight to the specifications necessary for machinery to achieve this level of accuracy. In fact,

at the time of Wilson's writing, machinery capable of producing these results were under construction.

Even though Wilson stated that his research could be applied to any high-energy particle, he concluded that protons would be best suited for therapeutic practices. This is because of the inherent properties of protons, in terms of ionization. When compared to electrons, protons exhibit a much higher specific ionization, thereby resulting in the ability to deliver stronger damage to target cells and destroying them more quickly and efficiently. Secondly, protons are not greatly influenced by scattering and spreading, allowing a dosage to be concentrated on a much smaller target area. Electrons, on the other hand, are much more susceptible to scattering and spreading. This greatly limits the concentration of ionization, and increases the minimum sized target that can be treated.

1.2.5 Further Research in Protons

Once Wilson had published his work, other physicists began experimenting with the idea of using protons for medical purposes. Their findings agreed with those outlined by Wilson, providing evidence that confirmed the feasibility of using protons for therapeutic practices. As soon as the suitable technologies became available (which was shortly after Wilson's work was published), radiobiological studies [6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17] proved that it was indeed viable to:

- Produce protons and other high-energy particles that could penetrate tissue, to reach any target area.
- Concentrate a dosage on very minute target areas, with submillimeter accuracy.

- Treat tissue regions deep within the human body.

Over time, physicians and biologists began to realize that in order to properly cure patients with tumors, it was vital that the entire tumor was removed or destroyed. Manual removal of a tumor will not yield total destruction of the cancerous cells. Therefore, there is a very high probability of the tumor growing back, thereby afflicting the patient once again. By utilizing protons for oncology, it was recognized that protons provided substantial potential in removing tumors. With this procedure, surgeons could have the ability to remove the entire tumor, to destroy all the cancerous cells in a target region, and minimize the removal of healthy tissue.

1.2.6 Problems Still Remain

However, the research conducted only showed the potential of such systems. These results merely showed that a high-energy beam has the potential to succeed in these applications. Most notably, Wilson showed that small lesions could be created using high-energy beams, such as protons. Once a target location is identified, there is little doubt that a beam can be guided to that target volume with submillimeter accuracy. But one daunting challenge still remains:

How is a target region identified?

A non-invasive imaging technique would be necessary in order to fully utilize protons for therapeutic uses. The applications and uses of treatments using high-energy beams were extremely limited, simply due to the lack of noninvasive imaging techniques. Until the imaging techniques became available, treatments with high-energy

beams was simply a concept, and could not become reality. With the advent of noninvasive imaging techniques, noninvasive surgery using high-energy beams finally became possible. The technology used to produce the high-energy beams was essentially ahead of its time, and the noninvasive imaging technologies had to "play catch-up" in order for the procedures to finally be put into practice.

1.3 Loma Linda University Medical Center: Pioneering Proton Therapy

Treatment using heavy charged particles, such as protons, was born in a physics research facility. Robert Wilson, perhaps regarded as the founding father of this research, was heavily involved with the design of the Harvard Cyclotron Laboratory (HCL). When others began to follow Wilson's research, they also performed their research in physics laboratories. The very first treatments using proton treatment were, as a result, performed in facilities dedicated to physics research. There were no hospitals or medical centers that possessed this type of technology. In 1962, Massachusetts General Hospital (MGH) entered into a partnership with HCL to provide proton treatment to patients. But the treatments were still not performed on-site at MGH; patients needed to travel to the HCL. The situation was similar in other areas around the world, especially in Sweden and at Berkeley. The problem was the fact that the only facilities that possessed the necessary equipment—particle accelerators—were physics research facilities. The technology was simply too expensive and too unwieldy to be deployed in a hospital environment.

After refinements in technology and an increased knowledge base, hospitals began investigating the means necessary to build on-site proton treatment centers. The

first hospital-based proton treatment center in the United States was built in 1990 by Loma Linda University Medical Center, Loma Linda, CA. LLUMC's research and development in proton therapy is continuous. The research program began in the 1970s, when LLUMC generated interest in particle therapy. At that time, engineers conducted an investigation to find an "ideal" beam to use in the newly proposed facility. In 1975, collaboration between LLUMC and researchers at University of California, Berkeley was established to examine the use of helium and heavy-ion irradiation. Later, LLUMC collaborated with Los Alamos Meson Physics Facility (LAMPF), to better understand the use of pions. Unfortunately, the program with LAMPF was cancelled in 1982 before a full understanding of pions was gained.

After further research, LLUMC decided to use protons in the new facility. Work performed by Wilson and others [5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17] greatly influenced LLUMC's decision to use protons. Through their analysis, LLUMC found that the cost to build a center around protons was lower, compared to other heavy charged particles. LLUMC researchers also wanted to deploy a system that minimized damage to healthy tissue as much as possible, as well as provide a high level of control over doses. Protons provided all of these benefits.

Several factors facilitated the development of LLUMC's proton therapy center. The primary factor was simply time. Technology related to particle accelerators and medical imaging progressed greatly during the 1970s and 1980s. Prior to this time, the technology was either not available, or was not practical for use outside a research laboratory environment. In the case of obtaining the particle accelerators necessary to generate the proton beams, the extremely high cost of such machines

severely limited their use and operation to only the most sophisticated of research centers. The cost of these machines was simply outside of the price range hospitals were willing to pay for one of their own. After widespread refinements in technology, particle accelerators became more affordable than in previous decades; however the cost was still very high. In addition to this, particle accelerators of the 1940s and 1950s were very large, complex machines. Each required large experienced crews of engineers to operate them. This resulted in high operational costs, both monetarily and in terms of man-hours. With the introduction of a newer generation of particle accelerators, new opportunities came about that greatly benefited the deployment of proton therapy, including:

- Improved technology that further increased the ability to exploit protons.
- Reduction in operational costs of particle accelerators.
- Decrease in the number of technicians required to operate the machines.

Finally, the greatest limiting factor behind proton treatments, was the lack of noninvasive imaging techniques. Without proper and accurate target localization techniques, physicians were unable to utilize the full potential of protons. Protons can be guided to a target region with very high accuracy, as confirmed by many researchers in their published works, see Section 1.2. The only limiting factor is identifying and localizing a target with which to deliver the protons.

LLUMC researched many avenues before making a final decision as how to build and deploy their new treatment center. During this time period, LLUMC heavily researched previous efforts in this field. Initial research endeavors into utilizing pro-

tons and other heavy charged particles for noninvasive surgery did indeed produce conclusive results, even in the absence of advanced noninvasive imaging techniques. However, the concept was, in a sense, ahead its time. In the 1940s-1960s, with no feasible and reliable method to guide a proton beam to a specific target, proton therapy was not a feasible therapeutic method. Other endeavors did provide successful noninvasive surgical techniques. Most notably, Dr. Lars Leksell, at the Karolinska Institute of Stockholm, Sweden, developed what became known as the *Gamma Knife* in 1968. In this system, a stereotactic frame is affixed to a patient's skull, which allows the physician to map out various cranial locations. From this physical reference, radioactive sources of Cobalt-60 are placed on the helmet with central channels of irradiation, using gamma rays. Although the stereotactic reference allowed physicians to perform noninvasive surgery, the usefulness of such a stereotactic reference has a limit to its accuracy. Indeed, intracranial structures lie in general regions in a patient's skull; however, this method does not ensure *accurate localization* of a *particular* region in a *particular* patient's skull. Additionally, the physician has little idea as to the size, shape, or precise location of a target volume. Since no medical imaging techniques existed at the time, the only confident method with which to locate and visualize a particular intracranial region was to physically open the patient's skull and look inside, thereby defeating the entire purpose of noninvasive surgery. The use of the stereotactic frame did provide a solution to the localization problem, but cannot be reliably used by itself for accurate localization of small target volumes. All of this changed in the 1980s with the introduction and wide acceptance of noninvasive imaging modalities, the most notable of which were Computed Tomography (CT) and

Magnetic Resonance Imaging (MRI). These allowed physicians to image intracranial regions precisely, to thereby provide accurate target localization for proton therapy.

The timing for LLUMC could not have been more perfect. In the 1980s, accurate, robust noninvasive imaging was finally available in the form of CT, thereby allowing physicians to accurately target specific anatomical regions for treatment. On the other side of the system, the technology needed to produce the proton beams had experienced great advancements. This led to decreased complexity, more compact size, and lower operating costs. All that was needed was the funding in order to finance the entire project. After turning to many private sources, and not finding the necessary monetary support, Dr. James Slater of LLUMC finally made a plea to his Congressman, Representative Jerry Lewis. In what seemed like an amazing act, Representative Lewis was able to secure more than $127 million to fund the construction and deployment of LLUMC's on-site proton therapy center. With all of these factors in their favor, LLUMC researchers were finally able to construct and deploy the center in 1990. Since its debut, the proton treatment center at LLUMC has treated more than 11,000 cancer patients. The proton treatment center has truly proven its usefulness time and time again, and has achieved the initial goals of the engineering team:

- Provide a (relatively) cost-effective treatment for cancerous and non-cancerous ailments.
- Implement a system that efficiently delivers doses to a target region, while minimizing damage to healthy tissues.
- Reduce the side effects experienced by patients compared to other treatments,

both invasive and noninvasive.

1.3.1 The (Current) System

The system LLUMC developed has the ability to treat patients with tumors or vascular malformations in the brain as small as 1 to 3 cm. Target localization is performed using CT and projection angiography, although MRI has also been used to outline brain tumors and locate adjacent brain structures. The accuracy of target localization is currently 1 to 2 mm, meaning that the system has the ability to place doses within 1 to 2 mm of the actual target volume. Prior to treatment, the patient goes through a treatment planning process, which allows the physician and radiation oncologist to simulate beam arrangement and visualize how the dose will be delivered. These are projected onto CT images, and a specific plan of treatment is determined. As the patient undergoes treatment, an immobilization device is used to properly secure the patient, and a stereotactic frame system is used to achieve consistent accuracy in head positioning. Both of these systems work together to provide proper registration of the patient with imaging and treatment equipment. Patient alignment is checked and verified before treatment begins. This is accomplished by imaging the patient in the treatment position using radiographic films (X-ray images). Additionally, a computerized monitoring system prevents incorrect dose delivery, should the patient move out of position prior to or during treatment. The system allows for treatment to proceed only if the patient has not moved after alignment verification by radiographs.

To overcome the basic problems that existed in the origins of proton treatment, LLUMC developed unique solutions to properly implement noninvasive proton radio-

surgery. The reduced cost and decreased complexity of present-day particle accelerators allowed LLUMC to obtain the equipment necessary to build the new system. Now, LLUMC had a dedicated particle accelerator for therapeutic usage, rather than having to share one at an off-site physics research laboratory. Another limitation that needed to be overcome was target localization. To meet this requirement, researchers created systems that made use of CT to provide stereotactic localization of anatomical targets. This solved the fundamental problem of guiding a proton beam to a desired target region. A third problem to resolve was proper beam delivery: in order to deliver doses to various anatomical regions, the system must have the ability to direct the beam from multiple directions. In order to satisfy this requirement, LLUMC researchers built a rotating gantry system—the only one of its kind in the world—allowing the proton beam to be dispensed from any location around the patient.

1.3.2 Proposal for a New Proton Radiosurgery System

The proton radiosurgery system implemented at LLUMC proved its effectiveness in treating patients with both cancerous and non-cancerous ailments. The success of the system allowed proton beam treatment to become an accepted therapeutic modality. To provide more therapeutic solutions to the medical community, LLUMC announced plans for a new proton radiosurgery system in 1996. With funding provided by the Henry L. Guenther Foundation (which also provided the funds for this work), LLUMC began development of an improved proton radiosurgery system. While the original system is sufficient for treatment of target volumes 1 to 3 cm in size, the newly

proposed system will be used to treat even smaller targets, most specifically, cranial targets. To achieve this goal, a more accurate system is required to properly and safely deliver doses to patients. The system will be used primarily to destroy brain tumurs, but can also be extended to cure such ailments and disorders as:

- trigeminal neuralgia
- Parkinson's disease
- epilepsy
- intractible pain
- arteriovenous malformations
- acoustic neuromas
- pituitary adenomas

The system will be ideal for those patients who are not candidates for traditional invasive surgery. Additionally, treating tumors of the smallest size (less than 1 cm volume) is impossible with traditional surgical methods. In both of these cases, noninvasive proton neurosurgery is the only option. Additionally, more traditional forms of noninvasive treatment can increase the chances of the patient developing radiation-induced cancers, thereby creating more problems.

Increasing the accuracy of dose delivery requires improvement in two primary areas: (1) patient alignment and alignment verification, and (2) stereotactic target localization. The issues pertaining to patient alignment and alignment verification can be solved using more accurate and complex fiducial systems, to attain proper

beam alignment and correct dose delivery. This topic is currently being handled by other projects, and the concepts pertaining to them are published. For more details, see [18].

The scope of this work will pertain to the second issue: stereotactic target localization. To successfully deploy LLUMC's new system for functional proton radiosurgery, targets must be identified to a higher degree of accuracy, to provide the system with detailed information during the administering of doses. This means obtaining accurate information from tomographic scans. Recall that the current imaging modality used for stereotactic target localization is CT. Using this system, target localization can be achieved with an accuracy of 1 to 2 mm. Although this measure is quite small, the new system must achieve an accuracy level even higher than this. Several of the aforementioned disorders require lesions to be placed directly on or between nerves. With a margin of error of 1 to 2 mm, there is a high risk of delivering a dose to the incorrect location, which could result in serious complications for the patient. Working in such close proximities to critical brain features, in addition to operating on such minute structures, provides very little margin for error. Being even 1 -2 mm away from the target volume can cause irreparable damage to healthy brain structures that were not intended to receive treatment. Therefore, a new method for stereotactic target localization must be developed prior to deployment of the new system.

During development of the original proton radiosurgery system, LLUMC researchers identified the usefulness of MRI as an imaging modality, but chose to use CT in the original system. This decision was mainly due to CT being the more mature of the two modalities at the time. Originally, MRI was only used to visualize and outline

brain tumors, and to locate neighboring brain structures. CT, on the other hand, served in the primary role in stereotactic localization. With advances in MRI technologies, a greater amount of information can now be obtained from MRI images versus CT images. This provides the physician with more information concerning the patient, to better diagnose and provide a solution. LLUMC researchers also realized additional benefits that MRI provided over CT, making MRI the preferred imaging modality for use in the new proton radiosurgery system.

Comparing CT and MRI

The advantages of MRI over the old system based on CT are numerous. MRI has the capability to allow LLUMC to achieve its goals outlined for the new system. To fully understand LLUMC's decision to use MRI over CT in their new system, an analysis of each system must first be presented.

1.4 Computed Tomography

Computed Tomography[1] (CT) represents the ongoing evolution of the applications of X-ray technologies for medical use. The concept of using X-rays for medical imaging has undergone several evolutionary iterations. Today, X-rays are basis for the following imaging modalities:

- Plain X-ray images
- Angiograms, used to image blood vessels

[1] Parts of this discussion are adapted from [19], see References.

- Digital Subtraction Angiography (DSA), a more advanced angiographic technique
- Computed Tomography (CT)

1.4.1 History and Development

X-rays were originally discovered by William K. Roentgen in 1895, earning him the first Nobel Prize in Physics in 1901. Since, then X-rays have been widely utilized in the medical community for noninvasive imaging. The advent of CT came about from research involving X-rays at THORN EMI Central Research Laboratories in Hayes, England. There, Godfrey Newbold Hounsfield invented the first CT system in 1967, and announced the invention to the public five years later. Independent of Hounsfield's work, Allan McLeod Cormack of Tufts University invented a similar process of imaging. For their work, Hounsfield and Cormack both shared a Nobel Prize in medicine in 1979. Since then, CT techniques have continuously evolved, thanks to technological advances.

Through the years, CT has also been referred to as Computed Axial Tomography (or CAT scan) and Body Section Roentenography, in honor of William K. Roentgen's discovery of X-rays. Whatever name is used to describe the technique, the method is still the same: use X-rays to produce a series of two-dimensional images of the internals of an object around a single axis of rotation. *Tomography* specifies an imaging technique in which the images are taken at discrete locations, or in sections. In essence, the images provide a representation of an object as if the object was cut into small slices. Hence, the images obtained from a CT study are usually referred to

as *slices*.

All CT scanners make use of X-rays to create images. Refinements to this technique have greatly improved the imaging capabilities of CT scanners. Additional advancements have significantly increased image resolution and sensitivity, while simultaneously decreasing the time needed to acquire and process images. The first generation of CT scanners in the 1970s required a great amount of time to acquire and process images. A typical scan would consist of 160 parallel images through a 180° arc. Each scan took approximately five minutes to complete, and then the images required 2.5 hours of processing on a computer, to algebraically reconstruct the images. Because of the extremely slow scan time, the use of CT was strictly limited to brain regions or limbs. Other organs, such as the lungs and heart, could not be imaged due to their movement. The resulting images would appear blurred, just as a photograph would when using a camera with a low shutter speed.

Over the next decade, the image acquisition and processing times were greatly reduced. These developments were very crucial to CT, allowing it to become a feasible medical practice. The primary setback to using CT in the early days was the great amount of time required to obtain and process the images. The widespread usage and acceptance of CT was a direct result of the reduction in time required to perform the procedure. Additionally, the quality and resolution of the images were greatly improved, further increasing the applications of CT in medicine. The first generation of CT scanners were restricted to an image resolution of only 80×80 pixels, and required the use of a water-filled Plexiglas tank to surround the scan region on the patient. The tank was necessary for proper operation of the detectors inside

the scanner, to reduce the dynamic range of the X-rays that the detectors received. Over time, refinements to the X-ray emission and detection systems allowed for huge improvements in image quality and processing time, as well as eliminating the need for the Plexiglas tank.

Enhancements in scan times and imaging quality were accomplished with more robust and stable detector arrays. Early CT scanners used photomultiplier tubes, which were fragile, expensive, and clumsy. The first breakthroughs were achieved using xenon gas ionization chambers in the detector arrays. Modern CT scanners now utilize detector arrays containing solid-state photodiodes, which are smaller and more stable than previous detector designs. These detectors are coated with a fluorescent rare earth phosphor, enhancing sensitivity and stability. In fact, these detectors are so stable that calibration for each image is no longer required.

With the advent of more capable computer technology, reconstruction of CT images could now be achieved with greater speed and accuracy. Refinement of reconstruction algorithms, as well as more powerful computers, allowed for faster and more accurate image reconstruction. Early scanners required several minutes to reconstruct one image. In contrast, modern scanners can reconstruct large studies with several hundred images in a matter of seconds. Enhancing the robustness of the reconstruction algorithms also reduced the number of artifacts present in each image, making the images more useful for visual examination by physicians.

1.4.2 Principles of Operation

As mentioned previously, CT scanners make use of X-rays to produce images. The principle of imaging is the same as a normal X-ray machine, just taken to the next level. The X-ray source in a CT scanner rotates around the patient, as the patient moves through the gantry. The detectors are placed on the opposite side of the circle. A series of scans are taken, creating the slices of the scan region. Once the scan is complete, a computer performs tomographic reconstruction, which yields the cross-sectional slices [20].

Tomographic reconstruction is based on a technique developed in 1910 by Johann Radon, a mathematician at the University of Vienna. Radon's work, which led to the Radon transform [20, 21, 22, 23], is based on integrating over hyperplanes. This concept can be applied to any tomographic imaging modality, including MRI.

When a patient is scanned, the projection of the patient at an arbitrary angle, q, is composed of a set of line integrals. These line integrals represent the attenuation of the X-ray beams (or other source) as they travel in a straight line through the scan region on the patient. The resulting image, $m(x, y)$, is a 2-dimensional representation of the attenuation coefficient. The simplest implementation of this method is to consider a system of parallel projections, which is precisely what the early CT scanners utilized to construct images.

Consider the coordinate system shown in Figure 1.2. The data that is collected and used to construct the images is simply a series of parallel rays across a projection at angle θ. Data is collected at various angles, and at several positions. It is the *attenuation* of the tissue that becomes visible in the images. When the X-rays travel

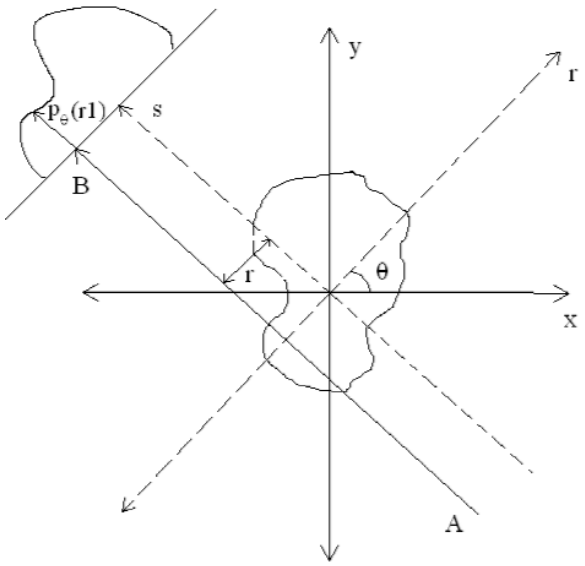


Fig. 1.2: Coordinate system for tomographic reconstruction.

through tissue, there is a reduction in the strength of the signal that reaches the detector arrays. The attenuation I of tissue is exponential, as shown in the following equation:

$$I = I_0 e^{(-\int \mu(r) ds)} \tag{1.6}$$

where $\mu(r)$ is the *attenuation coefficient* at point r along the path of the ray, which relates to the electron density of atoms. Given this relationship, the total attenuation of a ray on the project at angle θ and at point r is given by:

$$p(r,\theta) = ln(I_0/I) \tag{1.7}$$

Substituting the previous equation and simplifying yields:

$$p(r,\theta) = \int \mu(x,y)ds \tag{1.8}$$

At angle θ, the value of r onto which the point (x,y) is projected is described by:

$$r = x\,cos\,\theta + y\,sin\,\theta \tag{1.9}$$

Therefore, substitution of this result yields the final equation, known as the *Radon transform:*

$$p(r,\theta) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\,\delta(x\,cos\,\theta + y\,sin\,\theta - r)\,dx\,dy \tag{1.10}$$

where $f(x, y)$ is representative of $\mu(x, y)$. This function is also known as the *sinogram* of the 2D object, since the function is characteristic of a sine wave. Because of this, the graphed result appears as a series of blurred sine waves, with various amplitudes and phases. It is interesting to note that the Radon transform is very similar to the *Fourier transform*.

Reconstruction of the original object can be accomplished by using the *projection-slice theorem*. Essentially, the theorem states that the original object can be perfectly reconstructed if there exist an infinite number of one-dimensional projections of that object, taken at an infinite number of angles. Unfortunately, inverting the Radon transform directly makes the function become quite unstable, especially when used with noisy data. To further complicate matters, inverting the Radon transform is clumsy, and the process results in a reduction in the accuracy of the final result. In order to overcome these limitations, a more robust representation of the inverse Radon transform is used. This function is known as the *filtered back projection algorithm*. By using this function, the original image can be recovered from the "sinogram" data (produced by the Radon transform) by applying a ramp filter and then backprojecting. Application of the ramp filter can be performed efficiently by using well-known digital signal processing techniques (Fourier transforms, for example). Backprojecting simply creates an accumulation of values in each pixel in the image, which is a relatively trivial task. Combining the two techniques provides a very simple, efficient representation of the inverse Radon transform.

The images obtained from a CT scanner indicate the relative *radiodensities* of the scanned objects. The radiodensity of an object is the relative transparency of that

object to the passage of X-rays. Radiolucent objects are those that are transparent to X-rays, while radiodense objects are those that exhibit great opacity to X-rays. Radiodensity is quantified using the *Hounsfield scale* [24, 25]. The scale is based on the radiodensity of distilled water at standard temperature and pressure (STP). Zero Hounsfield units (HU) represents the radiodensity of distilled water at STP. Radiodense objects exhibit higher values in the scale than radiolucent objects. Table 1.1 describes the radiodensities of various media, in HU.

Substance	**Hounsfield Units (HU)**
Air	-1000
Fat	-120
Water	0
Muscle	+40
Bone	+1000
Cranial Bone	+2000

Tab. 1.1: Radiodensities of common media. [1]

Pixels in CT images are representative of the mean attenuation of the objects scanned, and represent radiodensities on the Hounsfield scale between -1024 and +3071. Based on this information, it is trivial to see that CT provides great detail for bone structures, especially cranial bone.

To overcome the great differences in radiodensities of bone and muscle, windowing techniques are used to enhance the images and maximize the amount of data obtainable. Windowing basically provides a mapping from Hounsfield units (-1024 to +3071) to grayscale pixel values (0 to 255). Depending on the application, the window can be very broad (to include a wide range of radiodensities), or can be made

very narrow (to enhance only a small range of radiodensities). This is very useful especially when imaging the brain. The radiodensity of cranial bone is nearly 100 times greater than that of brain tissue. Without limiting the range of radiodensities displayed in the image, a CT scan of a brain will only display the cranial bone structures. The brain tissue will be nearly invisible next to the cranial bone, since the radiodensity of the brain tissue is so small compared to that of the cranial bone. If images of brain tissue are desired, a narrow window would be used to best enhance the tissue structures. With a narrow window, only a small range of radiodensities would be mapped to the 256 gray levels, allowing greater contrast amongst the brain tissue structures. This also allows the image to easily display small variations in the physical characteristics of the tissue, and ignore the larger radiodensities associated with bone. Any radiodensity below the window would be black (pixel intensity of 0), while any radiodensity above the window would be white (pixel intensity 255). With proper windowing, it is possible to distinguish between tissues that differ in physical density by less than 1%.

1.4.3 Characteristics of CT

Because of the flexibility of CT, it has become the *gold standard* in diagnosing several types of diseases [26]. The gold standard (also known as the *criterion standard*) refers to the modality that is unanimously regarded as the definitive method in diagnosing diseases. When analyzing a modality and how it measures up to a gold standard, a sensitivity rating is given to that test. A perfect gold standard test would have a sensitivity of 100%, meaning that it is capable of correctly identifying a disease in

100% of afflicted individuals. This means that the test never gives any false-negative results, in that it will not report negative on the test for a patient that actually has the disease. In addition, the perfect gold standard test would also possess a specificity of 100%. This is the converse of the sensitivity, in that the test will not return a result of positive for a patient that does not have a specific disease. Obviously, the perfect gold standard test is, and always will be, a dream. Therefore, every gold standard test that exists today (and in the future) will, at some point, either return false negative or false positive results. At these times, it is up to the physician to determine the correct result for the patient.

At the moment, CT is the gold standard for the majority of imaging techniques. CT offers many advantages, including:

- High geometric accuracy
- Images are not subject to distortion
- Decent image resolution
- Windowing technique allows flexibility in displaying images

Even though CT is indeed the gold standard for many imaging techniques, it is far from perfect, and utilizing CT does come at a price. The primary reason that CT became the gold standard was because it was the only tomographic non-invasive imaging modality in existence. Therefore, at the time, there was no other suitable alternative to challenge CT. Among CT's shortcomings include:

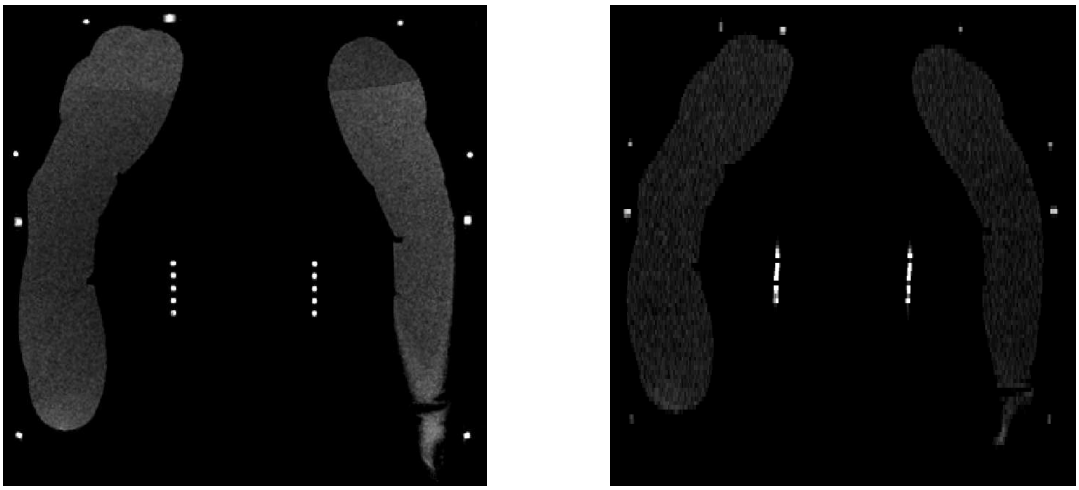
- Limited sensitivity and imaging capabilities for tissue regions
- Requires moderate to high doses of ionizing radiation

- Only scans on one plane (axial/transverse, or XY)

Taking into consideration the requirements for LLUMC's new system for functional proton radiosurgery, it is easy to see that CT does not provide sufficient support for submillimeter accuracy in target localization. As stated earlier, CT excels at imaging bone structures, but has limited capabilities for imaging tissue regions. Even though it is possible to enhance tissue regions and localize tumors in CT using windowing techniques (this practice is very common today), high accuracy cannot be obtained from CT images. The sensitivity that CT does provide in imaging tissue regions may be sufficient for visual reference, as well as the current radiosurgery system. However, it is by no means accurate enough for target localization in the new functional proton radiosurgery system.

Recently, improvements in the imaging capabilities of CT were largely due to increases in radiation dose. In brief: in order to obtain higher image resolution and perform more complex CT scans, the solution was to increase the strength of the X-rays. While the results are promising, the increased doses of ionizing radiation are not healthy to the patient. Recall back to Section 1.2, which identifies that ionization is the principle by which protons and other high-energy beams are used to destroy tissue. Introducing this principal on a large scale in CT can pose health risks. Exposure to high doses of radiation, or extended exposure, can lead to malignancies, hereditary effects, radiation sickness, or (in extreme cases) death.

Arguably the biggest shortcoming of CT in terms of obtaining submillimeter accuracy in target localization is the fact that CT only provides scanning on one plane: axial/transverse. Indeed, it is possible to reconstruct images in the coronal (xz) or



(a) Native MRI coronal scan. (b) Reconstructed MRI coronal scan.

Fig. 1.3: Loss in image resolution due to image reconstruction.

sagittal (yz) plane by using the axial scan: xy is represented in the individual slices, while z is obtained from slice to slice. However, this can also cause problems. It is quite common that the resolution within each image (pixel size) and between slices (slice thickness) are not equal. A typical scan features a slice thickness 4 - 5 times greater than the pixel size. With this configuration, the resolution in the reconstructed coronal and sagittal images becomes anisotropic. Often, the slice thickness in a CT scan are relatively thick compared to the pixel size. The purpose of this is two-fold: (1) to reduce the scan time, since fewer slices are created over the scan region, and (2) larger slice thicknesses usually improve signal-to-noise ratio, increasing the quality of the images. Because of this configuration, the reconstructed coronal and sagittal images simply do not provide the same level of detail and information as the native axial scans, see Figure 1.3. Here, the pixel size was 0.527 mm, and the

slice thickness was 2.0 mm. Of particular interest in these scans are the great loss in detail in the phantom's markers and fiducial points, as well as the blurring in the gray regions.

Indeed, it is possible to configure a scan to have equal resolution for pixel size and slice thickness, to promote isotropy in reconstructed images. Unfortunately, employing this method can lead to problems down the road. The issue with maintaining a suitable signal-to-noise ratio is of the utmost importance. Isotropic resolution can be obtained, but at the cost of signal-to-noise ratio, and therefore image quality. Secondly, if there is noise or erroneous data in the axial scan, those errors will be propagated into the reconstructed coronal and sagittal images as well, thereby degrading the quality of *all* the images. Thirdly, the quality of the reconstructed images is also dependent on the quality of the reconstruction algorithm. A poorly devised or inefficient algorithm will degrade the quality of the images, and therefore limit the usefulness of the images. If the coronal and sagittal images were generated independently, rather than calculated from an axial image set, such occurrences would not occur, and the accuracy of coronal and sagittal images would be preserved. Independent, accurate acquisition of axial, coronal, *and* sagittal images is required to achieve submillimeter accuracy in target localization.

It is now apparent that CT is not a suitable solution for LLUMC to achieve submillimeter accuracy in target localization in its new system for functional proton radiosurgery. Thankfully, there is an appropriate alternative imaging modality: *Magnetic Resonance Imaging.*

1.5 Magnetic Resonance Imaging

Prior to the introduction of *Magnetic Resonance Imaging (MRI)*[2], CT was the only advanced noninvasive imaging modality in existence. CT's widespread acceptance made it slightly difficult for MRI to take root in the medical field. However, MRI quickly proved its effectiveness. Today, MRI is a quickly growing modality, now challenging CT as the imaging technique of choice for major applications.

1.5.1 History and Development

In 1945, Felix Bloch of Stanford University and Edward Purcell of MIT (both of whom contributed research to the development of RADAR in the World War II era) independently discovered the phenomenon of nuclear magnetic resonance (NMR). They both shared the Nobel prize for physics in 1952 in commemoration of their research. However, the idea of utilizing NMR for imaging purposes did not come about for another 20 years. It was not until 1973 that the first published works described how NMR could be used as an imaging technique. Two independent groups—Lauterbur at State University of New York at Stonybrook, and Mansfield and Grannell at Nottingham University—suggested the technique. By this time, the name was changed to simply *magnetic resonance* (MR), to label the technique with a more politically correct name. Omitting "nuclear" from the name was deemed a wise idea due to the negative connotations to the word at this time in history, with the tension caused by the Cold War. Additionally, it was agreed that using the word "nuclear" would provide the false impression to physicians and patients alike that the technique is

[2] Parts of this discussion are adapted from [27], see References.

based on harmful radiation exposure, which is not the case. Either way, *NMR* and *MR* both refer to the same method, and both names are often heard today.

Recall that at this time in history (the early 1970s), CT scanners were now becoming commercially available. While CT scanner technology was still in its infancy, advanced noninvasive imaging was now possible. MR, on the other hand, was still a concept on the drawing board. Indeed, the concept was known, and published works described its applications. But it would not be until 1976 that MR was first demonstrated for the purposes of imaging. It would take another 10 years before the idea began to catch on with the medical community. But once it started to catch on, the use of MRI quickly grew, especially in neurological applications, where the usefulness of CT is relatively limited. Although CT was (and still is) the gold standard for most applications, physicians started using MRI to supplement CT scans for their imaging needs. However, the situation is changing today, as the usefulness of MRI has been recognized in other areas as well. Thanks to increased knowledge and big strides in technology, MRI is quickly challenging CT as the gold standard for many applications.

1.5.2 Principles of Operation

What makes MRI so different than CT? Why did MRI make such big waves in the medical community? The basic principles behind the imaging techniques in each modality are vastly different. While CT utilizes X-rays to provide images, MRI uses magnetism. Most of the quantum mechanics and physics involved in NMR are beyond the scope of this work. But at least a brief overview of the basic underlying concepts

of NMR is necessary to understand the impetus of this work.

NMR makes use of the properties of magnetism pertaining to an atom's nucleus, most specifically, the atom's intrinsic magnetic moment [28, 29, 30, 31]. NMR utilizes a strong, uniform magnetic field (commonly above 1 tesla) to align the spins of hydrogen nuclei and other atomic nuclei with non-zero spin numbers (i.e., unpaired protons and neutrons). The spin number determines the spin angular momentum of the nucleus. Examples of spin numbers for various atoms are given in Table 1.2.

Nucleus	**Spin Number**
^{1}H	$\frac{1}{2}$
^{2}H	1
^{12}C	0
^{13}C	$\frac{1}{2}$
^{19}F	$\frac{1}{2}$
^{31}P	$\frac{1}{2}$

Tab. 1.2: Spin numbers of common nuclei. [2]

In terms of utilizing NMR for medical purposes, the hydrogen (^{1}H) atom is of most interest because of its high abundance in the human body.

In the presence of the magnetic field, the nuclei are arranged in one of two directions in relation to the field: parallel, or antiparallel (geometrically parallel, but opposite in direction). It is the relaxation of the excited nuclei that makes imaging with NMR possible. MRI scanners then perturb the alignments of the nuclei with an electromagnetic pulse, to determine the specific density of hydrogen atoms at various locations in the scanned object. The field also reduces spatial encoding to two dimensions, and creates a signal response that can be translated into an image.

The scanner consists of a set of gradient coils and a large magnet. The job of the gradient coils is to generate a linear gradient field that allows indexing along an arbitrary plane. The large magnet, which creates the main magnetic field, is used to create a uniform magnetic field in the patient's body. The main magnetic field is oriented along the z-axis (which runs head to foot on the patient). When a patient is placed in the scanner, the magnetic field produced by the large magnet aligns the hydrogen nuclei in the patient's body. At this initial stage, there are an equal number of nuclei aligned parallel and antiparallel to the magnetic field. Therefore, the sum of the magnetic moments is initially zero. During scanning, when the gradient coils are activated, redistribution of the nuclei occurs, causing a certain number of the antiparallel nuclei to be realigned parallel to the field. This results in a change in the net magnetization of the field. The magnetic dipole moments of the nuclei undergo *Larmor precession*, in which there is a change in the direction of the spin axis around the induced magnetic field. The precession experienced by the nuclei is torque-induced. The torque that the magnetic field applies to the nuclei produces a gyroscopic motion. The angular velocity of the precession is described by:

$$\omega_0 = -\gamma\beta_0 \tag{1.11}$$

where β_0 is the external magnetic field, and γ is the gyromagnetic ratio, a value that relates the magnetic dipole moment to the angular momentum (*Larmor frequency*)

of a nucleus or elementary particle. This relationship is called the *Larmor equation*. The Larmor frequency of the nuclei is proportional to the magnetic field strength, commonly between 1.5 and 2 tesla (although the magnetic field strength can reach as high as 20 tesla in research models).

Immediately following this process, the electromagnetic pulse is introduced perpendicular to the magnetic field. The pulse is in the radio frequency (RF) range. The RF pulse perturbs the hydrogen nuclei, causing some of them to temporarily move into a non-aligned high-energy state. The individual spins of the nuclei, as well as the net magnetization vector, precess in-phase as a result of the RF pulse. The pulse then causes the net magnetization to flip towards the transverse plane. The angle of the pulse with respect to the main magnetic field determines the flip of the net magnetization. Therefore, the 90-degree RF pulse will flip the net magnetization by 90°. The location and method in which the RF pulse is introduced determines the orientation and type of image that is produced. This unique ability allows the physician to image across any orientation, thereby enabling imaging of an entire object in x, y, and z. Generating *voxels*—the three-dimensional version of a pixel, or *pic*ture *el*ement—is accomplished during activation of the gradient coils, which creates orthogonal magnetic gradients. These encode position within the phase of the signal (phase encoding, also known as the *spin-warp method*). To obtain x, y, and z scanning, the RF pulse is administered in discrete bandwidths in each orientation. It is this process that makes MRI a tomographic imaging technique, creating images in discrete locations along an arbitrary imaging plane. The process of spatial encoding occurs as a result of the magnetic signals produced by the gradient coils. Water and fat molecules in the body

respond to the RF pulse at a particular frequency by returning the same spin-echo signal, which is composed of multiple frequencies. Each frequency corresponds to different positions along the magnetic field gradient. Slices are defined based on the different spatial frequencies within the scanned object. To create the actual images, the *discrete Fourier Transform* is used to translate the signal data. The intensities represented in the images correspond to the hydrogen density at that location.

The signal data that is obtained relates to the method in which the nuclei relax and establish equilibrium after the RF pulse is discontinued. The various tissues in the human body contribute to different relaxation rates. The emission of energy and the Larmor frequencies of the nuclei compose the signal data, which is detected and recorded by the scanner. The RF pulse essentially knocks the nuclei out of alignment with respect to the magnetic field, and produces the aforementioned signal response. The process by which the nuclei realign themselves to the magnetic field is known as *longitudinal relaxation*. The time that it takes for a certain number of the nuclei to complete longitudinal relaxation is called "Time 1" or *T1*. The T1 time constant is typically about one second for tissues. The magnetization along the z-axis (the main magnetic field) increases exponentially with the T1 time constant. A second time constant, termed "Time 2" or *T2*, refers to the time it takes for the nuclei to undergo *transverse relaxation*, or the dephasing that occurs after introducing the transverse RF pulse. This causes the magnetization perpendicular to the z-axis to decay rapidly. Typically, T2 is about 100 ms for tissue. A slight modification to the T2 technique is called *T2**. Here, a *spin echo* pulse sequence is used, in which two RF pulses are introduced: one at 90°, followed by a second at 180°. Utilizing the two pulses causes

the spins to refocus, which allows compensation for magnetic field inhomogeneities.

Taking advantage of these properties allows great flexibility in NMR imaging techniques. Adjusting the image contrast is simply a matter of weighing the signals by T1, T2, T2*, or no relaxation time. The desired weighting is accomplished by configuring several basic image acquisition parameters, most notably *echo time* (TE) and *repetition time* (TR). Depending on the composition of the scanned object, the technician can adjust the signal sampling technique to achieve desired effects. For example, it is possible to highlight the various regions in the brain using different weighting of the signals. Weighing the signals by T1 will cause nerves to appear white, collections of neurons to appear gray, and cerebrospinal fluid to be dark. Weighing by T2 provides the opposite effect. The use of T2* weighting increases the relative contributions of other dephasing terms, such as proton diffusion, to increase image contrast. No weighing, also known as proton-weighted imaging, provides little contrast in normal subjects, due to the diffusion of protons in cerebral fluid. However, proton-weighted imaging is very useful in identifying regions in the brain affected by a stroke (infarcted regions), since the loss in contrast is attributed to the reduction in water diffusion in infarcted regions.

Even though T1, T2, T2*, and proton-weighting are sufficient for most applications, there are certain situations in which these techniques will not properly display the features of the scanned object. In this case, *contrast agents* are introduced to enhance image contrast. These agents can range from substances as simple as water, or as lavish as iron oxide. Other agents such as barium sulfate and gadolinium compounds have also been used. The choice of contrast agent is determined by the

specific magnetic properties of each agent, which then determines the best weighting to use. The properties of the nuclei that constitute the contrast agent, combined with the particular characteristics of the scanned object, determine the attributes of the signal response from interaction with the magnetic field and RF pulse.

A major breakthrough in formalizing the process of encoding images from signal data occurred as the work of Ljunggren and Tweig in 1983 [32]. Both scientists independently developed what is known today as the *k-space formalism*. The introduction of this technique was a huge breakthrough for MRI, in that it provided a universal method to interpret the signal data obtained from different MR imaging techniques. This expression effectively consolidates all the complicated concepts of imaging using NMR, and makes consistent imaging reconstruction possible, regardless of what technique was used to generate signal data.

In essence, the k-space formalism states that the demodulated MR signal freely precessing in the presence of a linear magnetic field gradient equals the Fourier transform of the effective spin density [32]. Expressing this mathematically:

$$S(t) = \tilde{\rho}_{\text{effective}}(\vec{k}(t)) \equiv \int d^3x \ \rho(\vec{x}) \cdot e^{2\pi\imath \ \vec{k}(t)\cdot\vec{x}} \tag{1.12}$$

where $S(t)$ is the demodulated MR signal, $\tilde{\rho}_{\text{effective}}$ is the effective spin density (defined as the true spin density $\rho(\vec{x})$ corrected for the effects of T1 preparation, T2 decay, magnetic field inhomogeneities, and other effects), and $k(t)$ is defined as:

$$\vec{k}(t) \equiv \int_0^t \vec{G}(t')dt' \tag{1.13}$$

where G signifies the linear magnetic field gradient.

From the k-space formalism, reconstruction of the image is rather trivial. To obtain the image $I(\vec{x})$, simply take the inverse Fourier transform of the sampled data:

$$I(\vec{x}) = \int d^3k \; S(\vec{k}(t)) \cdot e^{-2\pi\imath \; \vec{k}(t)\cdot\vec{x}} \tag{1.14}$$

In terms of the Fourier transform, $\vec{x}$ and $\vec{k}$ are both conjugates. With some further manipulation using the *Nyquist-Shannon sampling theorem*, it can be shown that the field of view of the image can be determined by the step in k-space; additionally, the resolution is determined by the maximum sampled value of k. Expressing these relations mathematically:

$$FOV \propto \frac{1}{\triangle k} \tag{1.15}$$

$$resolution \propto |k_max x| \tag{1.16}$$

Keep in mind that the k-space formalism expresses k independently for x, y, and z, therefore the field-of-view (FOV) and resolution obtainable in x, y, and z are independent of each other as well. The obtainable FOV and resolution will consistently improve over time, as gradient fields become stronger and bores become wider.

1.5.3 Characteristics of MRI

The introduction of MRI proved to be a very significant event. Today, MRI makes possible many procedures and imaging techniques not possible with CT or other modalities. Although CT exists today as the gold standard for most imaging techniques, MRI is quickly becoming the preferred choice for various techniques. This is due in part to refinements in MRI technologies, as well as increased research efforts to improve the capabilities of the system.

Presently, MRI and CT are the two leading imaging modalities in the medical community. In many applications, MRI and CT are used side-by-side, supplementing each other, to provide the physician with a more complete picture (literally) of a patient. As knowledge increases and imaging techniques mature, MRI could one day supercede CT as the gold standard.

What makes MRI such a robust modality? MRI possesses many advantages over CT, among these include:

- Unlike CT, MRI does not make use of radiation for imaging. Instead, MRI utilizes magnetic fields and radiofrequencies to obtain images. Presently, there are no known dangers to humans from being exposed to the magnetic fields or radiofrequencies generated by MRI scanners. The same cannot be said about the radiation utilized by CT.

- MRI offers high flexibility in imaging options. MRI scans can be configured using a plethora of options, to easily obtain desired results. Most importantly, MRI allows imaging on three different planes: axial (xy), coronal (xz), and sagittal (yz). CT, on the other hand, only provides imaging on one plane, the axial plane. To provide data from the other planes using CT, either the images must be constructed from axial images, or the patient must be moved accordingly (as is the case with traditional X-ray). Since MRI allows for imaging in all three planes, images are not constructed from other studies, and the patient never requires repositioning.

- MRI provides higher resolution and greater tissue contrast compared to CT. The latest generation of MRI scanners possesses larger bores and utilizes stronger gradient fields to achieve high-quality images. Additionally, MRI offers imaging in all three planes, preserving isotropic image resolution.

- MRI offers comparable spatial resolution to CT, but provides vastly superior contrast, especially for tissue. To improve, enhance, or otherwise configure contrasts in images, MRI has the ability to use signal weighting, as well as external contrast agents. Although CT can make use of windowing to improve contrast, MRI is much more robust and flexible.

All of these attributes of MRI have made it a very useful tool in oncology and neurological studies. The features MRI possesses allows for very effective localization of tumors and malignancies. First and foremost, MRI offers scanning across all three planes, which allows for isotropic resolution. Although a CT scan can reconstruct images in all three planes, and can be configured to scan using isotropic resolution, the images are not as high quality as MR images (see Figure 1.3). There is a certain amount of degradation in quality during image reconstruction, since complex calculations are required to create the images. Truncation and rounding errors that occur during reconstruction, although small, do affect the final output image. The quality of a native coronal or sagittal image—one that was created during the scanning process, rather than being calculated from an axial image—will always be greater, regardless of the quality of the axial scan, or how accurate or efficient the reconstruction algorithm. The only method that will allow CT to natively image along all three planes would be to move the patient accordingly. Obviously this will cause problems for accurate localization of structures, since moving the patient compromises proper image registration. In other words, locations in the images in each plane will not correctly correspond to each other due to the slight differences in patient positioning in each scan. Indeed, this method would be sufficient, if the images were to be used simply for visual purposes. But performing any type of accurate geometric localization would not be recommended, due to the inconsistent patient placement in each scan.

Secondly, MRI's manipulation of magnetic properties provides images of tissue regions with exceptional contrast and detail. Recall that contrasts in CT images are

based upon the radiodensities of scanned objects. Bone is very radiodense, while tissues are relatively radiolucent, resulting in high image contrast for bone and much lower contrast for tissue. On the other hand, MRI operates on the magnetic moments of atomic nuclei, which helps create greater image contrast for tissue regions. Additionally, thanks to signal weighting techniques, image contrasts can be further adjusted, to highlight various anatomical regions and provide even more information.

The benefits that MRI provides for stereotactic target localization are numerous. But it goes without saying that no imaging modality is without its faults, and MRI certainly falls under that classification. Even though MRI provides great advantages over CT, there are still some problems that first must be overcome before MRI is to be used in successful and accurate stereotactic localization for proton radiosurgery. Since MRI is an ever-evolving field, many of the original flaws inherent in previous generation scanners have been addressed, and current work continues to improve MRI imaging techniques. However, several primary problems still exist:

- Because of the utilization of strong magnetic fields, metallic objects within or around the scanner pose a very serious problem. These can include tooth fillings, clothing accessories (i.e., belt buckles, buttons, etc.), cardiac pace makers, and other ferromagnetic objects. There have been several cases where these objects have been attracted to the center of the magnet at very high velocity, resulting in injury or death. Even when the machine is not scanning, the magnetic field is still active, and always poses a potential hazard.

- Deploying and implementing MRI technologies is very expensive. The high cost of installing and maintaining MRI equipment greatly limits the accessibility of

medical centers to use the technology. Modern machines cost in excess of $1 million, and require hundreds of thousands of dollars in annual upkeep.

- The sounds generated by rapidly switching the magnetic gradient coils exceed the threshold of pain. In some scanners, the sounds can reach as high as 130 dB, which is equivalent to that of a jet engine at take-off. Special precautions must be taken in order to prevent hearing loss.

- MR images are susceptible to distortions. The latest generation of MRI scanners provides very high-resolution images, through the use of the larger bores and stronger gradient fields. However, these characteristics have also increased the susceptibility of distortions in images.

The shortcomings of MRI, although potentially significant, are far outshined by its advantages. By no means is MRI the perfect imaging modality. The aforementioned disadvantages do not detract from the effectiveness of the technique. Therefore, the unique characteristics of MRI make it the clear choice for use in functional proton radiosurgery:

- MRI provides exceptional image contrast of tissue regions.

- Scanning on three planes does not require the patient to be moved. This promotes accuracy, consistency, and isotropic resolution of scanned regions across all three planes.

- The higher resolution of MRI compared to CT is suitable to obtain the high accuracy needed for target localization.

It is because of these characteristics that LLUMC chose to base its new system for functional proton radiosurgery on MRI.

The choice of building a system for functional proton radiosurgery based on MRI may seem simple. MRI clearly satisfies the requirements necessary for identifying tumors and malignancies, as well as localizing target regions. However, MRI is not a perfect system. Neither is CT, or any other system, for that matter. No system will ever be perfect. But, with some careful planning and techniques, the full potential of MRI can be utilized to accomplish the task at hand.

Before functional proton radiosurgery based on MRI technologies is possible, it is necessary to first address the problems posed by MRI. Failure to do so will severely limit the effectiveness of MRI for use in functional proton radiosurgery.

Properly handling the issue of metallic objects on patients or in scanning rooms requires special precautions. Simple steps can be taken in order to minimize danger, such as removing as many metallic objects as possible before scanning the patient. Strict safety measures are imposed, to restrict the number of ferromagnetic objects in and around the scanner. However, metallic implants, tooth fillings, and cardiac pace makers cannot be removed. The problems and implications posed here are beyond the scope of this work, and will not be discussed as a result. However, they are important to note and consider due to the potential severity of the issue.

The high cost of MRI equipment may prevent some medical centers from being able to utilize the technology. There are a limited number of centers that are willing (and able) to provide the monetary support for such technology. Comparable CT units, although quite expensive themselves, cost substantially less. Through new

research and breakthroughs, MRI technology is becoming more affordable, making MRI a more widespread modality. As time progresses, the technology will continue to improve, thereby decreasing overall costs and allowing more medical centers to obtain and deploy MRI in its medical practices. It may be the case in the future that MRI scanners become nearly as common as X-ray machines; only time will tell.

Protecting patients and personnel from the loud noises generated by the switching of the gradient coils requires nothing more than proper ear protection. Patients are outfitted with earmuffs, to shield their ears from the sounds while undergoing scanning. The scanning room is heavily insulated, mainly to shield exterior rooms from the magnetic field and electromagnetic pulses. This also is effective in quelling the sounds of the scanner outside the room.

The disadvantages posed by MRI are not severe. With costs aside, the detriments are rather minor. However, there is one great challenge that still remains: properly addressing the various distortions inherent in MR images.

1.6 Inherent Distortions in MR Images

1.6.1 Sources of MRI Distortions

The primary disadvantage posed by MRI is the presence of distortions in the images. There are six generally accepted sources of MRI distortions: *chemical shift, shimming, eddy currents, linear scale errors, magnetic susceptibility variations, and gradient field nonlinearities* [3]. Fortunately, four of these six sources are either very minor (in terms of the scope of this project), or are not an issue with properly maintained scanners. However, it is still useful to understand the dynamics behind each source.

Chemical Shifts

The presence of *chemical shifts* in MR images is simply due to the difference in composition of material in the human body. Most particularly, the hydrogen atoms in water resonate at a slightly different frequency than those found in fat. The difference is on average 205 Hz in a 1.5 tesla magnetic field [3]. This is due to the chemical composition of water and fat. In fat, the hydrogen atoms are attached to long hydrocarbon chains, while the hydrogen atoms in water are attached to oxygen atoms. This causes the electrons within the bonds to react differently to the main magnetic field. Fortunately, considering that this project focuses on neurological studies, chemical shifts are negligible because the human brain is composed of little fat.

Shimming

During scanning, the primary assumption is that the main magnetic field is homogeneous. However, within the scanned volume, this is not the case. To compensate for this difference, a process called *shimming* is performed. Extra coils are used in the scanner to create a homogeneous magnetic field. Modern scanners have shim coils, which are calibrated automatically by the scanner. Since the scanner handles this process, this source of distortion does not become a major issue. However, failure to complete this process will introduce extra distortions in the images. Refer to Section 7.1.2 for more information on this topic.

Eddy Currents

The switching of the magnetic fields during scanning causes *eddy currents,* which introduces distortion in the images. The eddy currents produce a time-varying magnetic field that opposes the main magnetic field in the scanner. This inhomogeneity in the magnetic field causes the hydrogen atoms to react differently to the field than originally assumed. Thankfully, the effects of eddy currents can be suppressed by utilizing a shielded gradient system. Frequent screening of the gradient coil serves to further reduce the effects of eddy currents, to the point where the problem becomes negligible.

Linear Scale Errors

Linear scale errors result from improper calibration of the scanner. These occurrences are not very likely, since the software controlling the scanners greatly simplifies the calibration process. Additionally, MRI scanners typically undergo regular maintenance, to ensure proper operation. Since it is unlikely that careless or improper handling of the MRI scanner will occur, these errors are nearly nonexistent.

Four Down, Two Remaining

The two remaining sources of distortion are, unfortunately, quite significant. Therefore, the geometric warping present in MR images are primarily caused by magnetic susceptibility variations and gradient field nonlinearities.

Magnetic Susceptibility Variations

Magnetic susceptibility variations are created because of the varying compositions of tissues. The various tissues in the human body exhibit different magnetic susceptibilities, and therefore cause small changes in the main magnetic field. However, during scanning and image reconstruction, it is assumed that the main magnetic field is homogeneous. The inhomogeneities in the magnetic field produce off-resonant frequencies at each pixel, which displaces affected tissue regions to incorrect locations in the images. The errors produced in the images are a direct result of the incorrect mathematical assumption of the magnetic field. The distortions caused by magnetic susceptibility variations are beyond the scope of this project, and will therefore not be discussed in this work. Since the phantoms utilized in this project do not mimic the composition of the brain (i.e., the phantoms are not composed of various materials, but, rather, one homogeneous substance), the calculations and results given in this work are not vulnerable to magnetic susceptibility variations. However, these distortions will definitely need to be considered for correction of cranial or other anatomical images.

Gradient Field Nonlinearities

Even though the first four sources of distortion are quite minor, and magnetic susceptibility variations do not apply to the testing phantoms utilized in this project, there still remains one source of distortion: *gradient field nonlinearities.* This source of distortion is independent of the composition of the scanned volume, therefore *ALL* MR images are susceptible to gradient field nonlinearities. The entire focus of this

work will center around handling gradient field nonlinearities, since these distortions are the most apparent in all MR images.

1.7 The Enemy: Gradient Nonlinearity Distortion

The greatest limitation of MRI that must be overcome before implementing functional proton radiosurgery is the *gradient nonlinearity distortion* inherent in MR images. The distortions are not great enough to distort images beyond recognition; MR images are excellent for visual observation and study. However, the gradient nonlinearity distortion are strong enough to prevent accurate localization of anatomical regions within scanned images. The distortion can shift pixels in images up to several millimeters. Therefore, these distortions must be corrected as a first step to provide accurate localization to less than one millimeter.

1.7.1 Sources of Gradient Nonlinearity Distortion

Fringe effects in the magnetic field lead to gradient nonlinearity distortion. Since it is assumed that the main magnetic field is homogeneous, the distortions become apparent during image reconstruction. The fringe effects cause the magnetic field to spread out and become less linear. The farther away from the center of the magnet, the less linear the field becomes, causing a consistent geometric warping in each image, as demonstrated in Figure 1.4.

The shape of the distortion can be described with respect to the cylindrical geometry of the magnets used in MRI scanners [33]. Recall that the MRI scanner produces a uniform homogeneous magnetic field, to effectively utilize the concepts of NMR

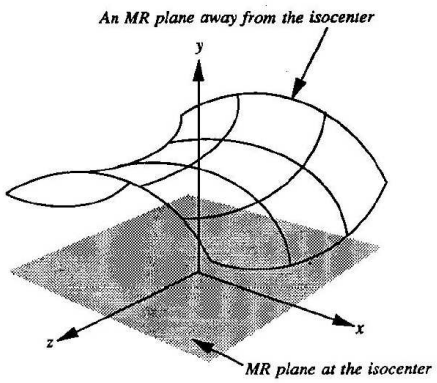


Fig. 1.4: Distortion inherent in MR images caused by gradient field nonlinearity [3].

to produce images. In order to create the uniform magnetic field, various coils are used. There are two types of coils used to produce the main field: *Maxwell coils* and *Helmholtz coils* [2]. The other gradient coils, used to encode in x and y, are produced by a *Golay coil* arrangement.

1.7.2 Main Magnetic Field

The magnet is the most costly part in the MRI scanner. This is because the magnet is not a run-of-the-mill magnet. Permanent magnets are only capable of producing magnetic fields around 0.3 T, which is not sufficient for NMR imaging [2]. Additionally, the weight of such magnets becomes too large at this stage to be of practical use. Therefore, MRI scanners make use of superconducting magnets, allowing magnetic fields of several tesla to be produced. The magnets are composed of circular coils of NbTi alloys, which exhibit low resistance at very low temperatures (about 9 K) [2]. Once current is first introduced into the coils, the current flows nearly indefinitely, as long as the proper temperature is maintained. To maintain proper operating temperatures and to keep the current flowing, liquid helium surrounds the coils. The

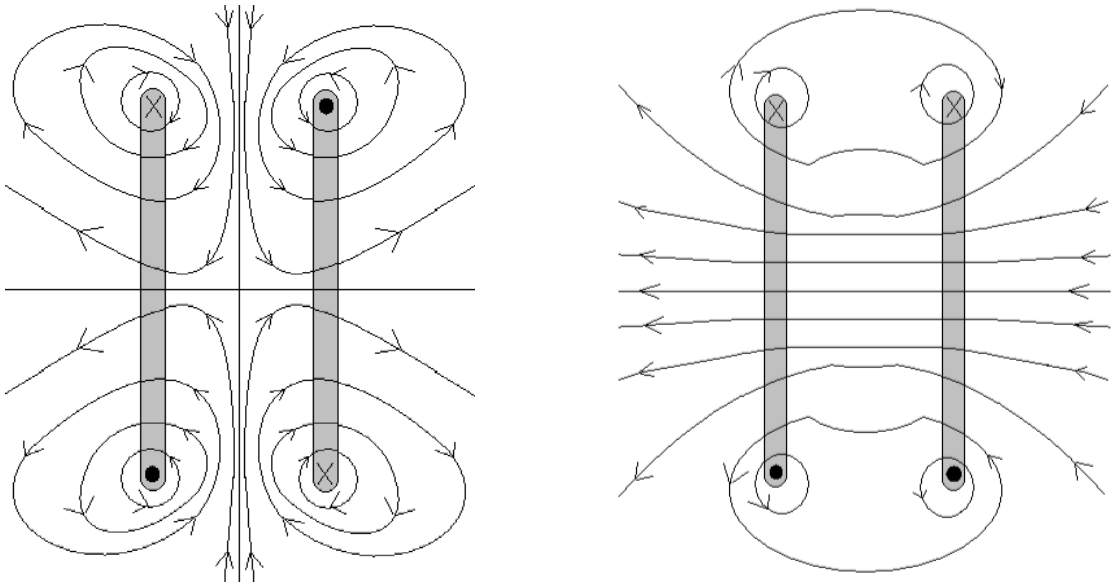
resulting configuration produces a very stable magnetic field, essential for consistent NMR imaging.

In order to generate a uniform field, two identical coils are used. In Maxwell coils, current is passed in opposite directions. Helmholtz coils (also known as Helmholtz pairs), on the other hand, feature current that flows in the same direction. The distance that the coils are separated produces the magnetic field region with spatial uniformity. To accomplish this using Maxwell coils, the coils should be separated by 1.73 times their radius [2]. Helmholtz coils, on the other hand, should be separated by a distance equal to the radius of the coils.

The magnetic field generated by the Maxwell coils is called a quadrupole magnetic field, and appears as shown in Figure 1.5(a). Likewise, the magnetic field produced by Helmholtz coils appears as shown in Figure 1.5(b). Both coil systems are commonly used in modern MRI scanners.

Observe the shapes of the magnetic fields created by each system of coils in Figures 1.5(a) and (b). The centermost regions are nearly uniform, while those regions farther away from the center exhibit curvature. Additionally, notice the symmetry of the curvature about the center line of the field. Extending these models to three dimensions will yield the *exact* shape shown in Figure 1.4.

Calculating the magnetic field produced by the coils is straightforward. Using the *Biot-Savart law* [34], the magnetic field that is created by the steady current density in the coils can be described by the following:



(a) Maxwell coils. (b) Helmholtz coils.

Fig. 1.5: Magnetic fields produced by Maxwell (a) and Helmholtz (b) coils.

$$dB = \frac{\mu_0 I}{4\pi} \int \frac{ds \times \hat{r}}{r^2} \tag{1.17}$$

where dB is the field created by the current through the conductor, I is the current flowing through the coils (in amperes), ds represents the length that the current flows, $\hat{r}$ is the displacement vector from the current element to an arbitrary point, r is the distance from the current element to an arbitrary point, and μ_0 is the permeability of free space, defined as $4\pi \times 10^{-7} T \cdot m/A$ [34].

This expression can be adapted to describe the Maxwell or Helmholtz coils. First,

consider the coils individually, rather than as a pair. Modifying the Biot-Savart law to describe each coil yields the following expression:

$$B_x = \oint \frac{\mu_0 I R^2}{2(x^2 + R^2)^{\frac{3}{2}}} \tag{1.18}$$

With the pairs of coils working together to produce one magnetic field, the total field produced by the paired coils is therefore:

$$B_{total} = B_1 + B_2 \tag{1.19}$$

$$B_{total} = \frac{\mu_0 I R^2}{2} + \left[\frac{1}{(x^2 + R^2)^{\frac{3}{2}}} + \frac{1}{((R - x)^2 + R^2)^{\frac{3}{2}}} \right] \tag{1.20}$$

$$B_{total} = \frac{\mu_0 I R^2}{2} + \left[\frac{1}{(x^2 + R^2)^{\frac{3}{2}}} + \frac{1}{(2R^2 + x^2 - 2xR)^{\frac{3}{2}}} \right] \tag{1.21}$$

where I and R are described in the Figure 1.6, demonstrating a Helmholtz pair.

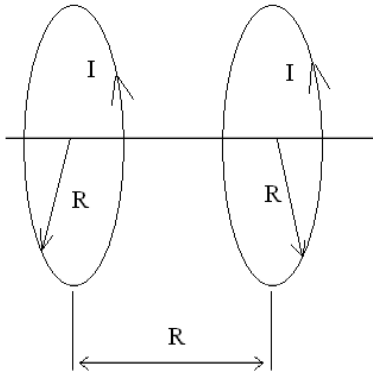


Fig. 1.6: Helmholtz coils configuration.

Since the main magnetic field in the MRI scanner is in z, the expression can be rewritten as:

$$B_z = \frac{\mu_0 I R^2}{2} + \left[\frac{1}{(z^2 + R^2)^{\frac{3}{2}}} + \frac{1}{(2R^2 + z^2 - 2xR)^{\frac{3}{2}}} \right] \tag{1.22}$$

Keeping in mind the symmetry of the coils, the magnetic field produced at every point inside the cylinder between the coils can be described by:

$$B_{(x,y,z)} = \sum_{i=0}^{\infty} A'_i (x^2 + y^2)^{2i} \sum_{j=0}^{\infty} B'_j z^{2j+1} \tag{1.23}$$

where A'_i and B'_i are constants, such that the ideal gradient G_z produced during

the reconstruction process is $G_z = A_0' B_0'$ [33]. Based on this information, the general expression of $B_z(z)$ then becomes:

$$B_z(z) = G_z(z) \left(1 + \sum_{i=0}^{\infty} A_i (x^2 + y^2)^i \sum_{j=0}^{\infty} B_j z^{2(j-i)} \right) \tag{1.24}$$

Equation 1.24 provides the basis for the sum of spherical harmonics functions described below.

1.7.3 X and Y Gradients

The other gradients, which encode x and y, are produced by *Golay coils*. These coils are saddle-shaped, and run along the bore of the magnet. Their odd shape produces a linear variation in the main magnetic field (B_z) along either the x or y axis. The axis that is chosen is dependent on the axial orientation. Utilizing this configuration, it is possible to create a very linear magnetic field at the central plane. However, this uniform field quickly loses linearity at locations away from the center. The coils are configured appropriately to lessen the effects of the loss in linearity. Since the coils are independent of one another, the magnetic field can be adjusted in x, y, and z, simply by sending currents in the appropriate proportions to the G_x, G_y, and G_z coils. Figure 1.7 demonstrates the Golay coils in a typical MRI scanner.

Considering the magnetic field produced in the xz plane, which encodes the x direction, the Biot-Savart law can be extended to this situation, to yield the following

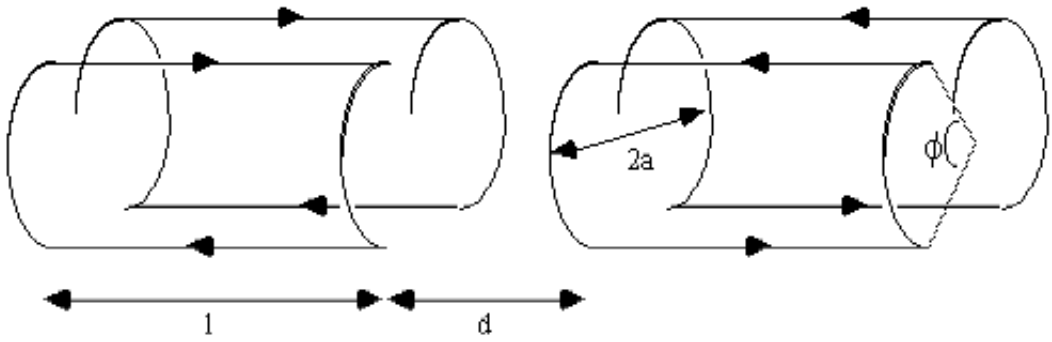


Fig. 1.7: Golay coil for producing linear field gradients in B_z along the x or y axes. l = 3.5a, d = 0.775a and $\Phi = 120$ degrees. [2].

expression:

$$B_x(x,z) = \frac{\mu_0 I}{4\pi(x+a)}\left(\frac{b/2-z}{\sqrt{(b/2-z)^2+(x+a)^2}} + \frac{b/2+z}{\sqrt{(b/2+z)^2+(x+a)^2}}\right)$$

$$+\frac{\mu_0 I}{4\pi(x-a)}\left(\frac{b/2-z}{\sqrt{(b/2-z)^2+(x-a)^2}} + \frac{b/2+z}{\sqrt{(b/2+z)^2+(x-a)^2}}\right) \text{[33]} \qquad (1.25)$$

The same case would also follow for the magnetic field produced in the yz plane, encoding the y direction.

1.7.4 Shim Coils

There is an additional set of accessory coils that contribute to the characteristics of the main field. These coils are called *shim coils,* and are used to improve the homogeneity

of the main field. Indeed, the saddle coils improve the main field homogeneity, but are only effective to a certain extent. To further increase the homogeneity of the main field, and come closer to the assumption of sampling a homogeneous main field, a set of resistive shim coils are used. As mentioned previously, the assumption made during signal processing and image acquisition/construction is that the main field is homogeneous. The harsh reality is that the field is in fact inhomogeneous. This incorrect assumption can lead to image artifacts and peculiarities [2], the severity of these depends solely on the extent of the gradient field inhomogeneity. To help compensate for these effects, the MRI scanner samples the magnetic field, and uses an algorithm to adjust the positions of the shim coils inside the bore. The shim coils apply shaped fields to the main magnetic field, which vary with a particular function of position [2]. When properly configured, the shim coils are very effective in improving the intrinsic homogeneity of the main field, and are able to reduce the field effects due to susceptibility differences in the scanned objects[3]. The benefits of shimming, and the results of scanning without shimming, are discussed in Section 7.1.2.

The process of shimming, and the implications involved in shimming, are beyond the scope of this project. The concept is, however, very important, and does play a large role in obtaining quality images. Section 7.1.2 outlines problems that occurred as a result of failure to apply shimming. For detailed information concerning the details behind and the theoretical basis for shimming, refer to [35].

[3] Keep in mind that the coils *improve* the main field homogeneity; it does not create a totally homogeneous field, therefore gradient nonlinearity distortion is still a problem.

1.7.5 Total Magnetic Field

All the coils work in concert to produce one large, generally uniform magnetic field. In order to describe the entire field, it is necessary to consider all the coils together, rather than separately. Combining the previous equations, and once again taking into consideration the symmetry of the system and the ideal values of the gradients, will yield a general expression of the entire magnetic field.

Considering the x direction, the general expression for the magnetic field produced in x is:

$$B_x(x,y,z) = G_x(x)\left(1 + \sum_{i=0}^{\infty} A_i(x^2+y^2)^i \sum_{j=0}^{\infty} B_j z^{2(j-i)}\right) [33] \tag{1.26}$$

This case can be extended to describe the same relationship in y.

From these equations, it can be now concluded that the general function for the entire magnetic field exists as:

$$B_{(x,y,z)}(x,y,z) = (x,y,z)\left(1 + \sum_{i=0}^{\infty} A_i(x^2+y^2)^i \sum_{j=0}^{\infty} B_j z^{2(j-i)}\right) [33] \tag{1.27}$$

Expressing this relationship as a function will yield the final relationship that de-

scribes the nonlinearity functions:

$$f_\alpha(x,y,z) = \alpha\left(1 + \sum_{i=0}^{\infty} A_i (x^2+y^2)^i \sum_{j=0}^{\infty} B_j z^{2(j-i)}\right),$$

$$\alpha = (x,y,z) \tag{1.28}$$

where x, y, z follow the coil axes.

This series of infinite sums, described in full by Equation 1.28, is known as the *sum of spherical harmonics*. The expression is the generally accepted expression used to describe the characteristics of the nonlinearity of the magnetic gradients [33, 36, 37, 38, 39]. As a result, this function serves as the foundation for nearly all published gradient nonlinearity distortion correction techniques. The algorithms developed in this work, as in other works, use a truncated version of this function (see Equations 2.8 - 2.10), since it is impossible to express the entire series of infinite sums on a computer. This is suitable for correction, since the higher order terms become increasingly minute, and do not contribute greatly to the overall result. Therefore truncation will not introduce significant errors into the final result [33]. The two largest manufacturers of MRI scanners, GE and Siemens, also use truncated versions of this expression to describe gradient nonlinearities: GE expresses this series using 5 terms, while Siemens uses 11 terms [36].

1.8 Previous Works

The topic of implementing a software correction method to perform gradient nonlinearity distortion correction has been mentioned in many works [3, 33, 36, 37, 38, 40, 41], confirming that developing and deploying such a system is indeed feasible. This work continues where these works stopped, and builds upon the techniques first presented and discussed. Each discussion presents a different method to tackle the gradient nonlinearity distortion problem, and provide accurate images for target localization. While the work presented in this thesis discusses the same topic, the implementation is very different.

The numerical analyses of Chapter 6 demonstrate that the gradient nonlinearity distortion correction possesses an accuracy of 0.39 mm to 0.78 mm. As a result, the markers in the Lucy phantom were localized with an accuracy of between 0.48 mm and 0.99 mm. These results show that the distortion correction does indeed provide submillimeter accuracy. To fully grasp the meaning of these results, as well as the motivation and impetus behind the work described in this thesis, it is best to understand the characteristics and results of previous endeavors in this research area.

1.8.1 Scanner Correction

The two most prevalent manufacturers of MRI scanners, General Electric (GE) and Siemens, both provide distortion correction methods, which can be activated within the scanner software. The correction takes into account the properties of the magnetic field in each scanner, and implements a number of correction filters.

Several works have pointed out the limitation of such filters [36]. These filters

are merely two-dimensional corrections. The correction is performed based on the location of the slice relative to the gradient coil. Each image is corrected in turn. Unfortunately, this method is not a true "correction." It is simply a method that filters the image, to *reduce* the amount of visible distortion in the images. Another limitation with this technique is the fact that the method does not consider the data in three dimensions. Therefore, the pixels in each image possess only a limited degree of geometric accuracy. Localizing target regions in the filtered images will not provide much improvement in geometric accuracy as would localization on unfiltered/uncorrected images. In fact, a true three-dimensional distortion correction method will improve upon the scanner correction, to create true geometric accuracy in all three planes.

Indeed, manufacturers are currently implementing three-dimensional correction techniques, but none are available currently on the market. However, this distortion correction will only work on the particular scanner it was developed for. This severely limits the deployment of a three-dimensional distortion correction to only those scanners that feature the new software. Not all scanners will be able to benefit from this technology, further limiting the scale of the deployment of a distortion correction method. To further complicate matters, the distortion correction will require integration into the scanner software, requiring medical facilities to perform software upgrades on the scanner system. Therefore, a correction method must be developed independent of the manufacturers.

To complicate matters, most physicians use all the available imaging filters offered by the scanner. This is done to improve the visual quality of the images, for use in visual referencing. Since MRI images will commonly feature image filtering performed

by the scanner (in addition to ones that feature no filtering at all), the distortion correction method must also be able to improve upon the scanner's filtering, to yield a true "undistorted" image.

1.8.2 Fusing CT and MRI

In his work, Sumanaweera [3] presents a method of fusing CT and MR images together, to provide geometric accuracy. This seems logical, and is actually common practice today. Since CT is the gold standard, MR has been primarily used to supplement CT, to provide more complete patient information. This implementation is slightly backwards, so to speak, in that CT is used to supplement MRI. Historically, MRI was secondary to CT. Using CT and MRI together provides successful results since CT is not subject to distortions as MRI. Combining the two together provides a method that utilizes the best properties of each modality.

Considering this method in terms of implementing functional proton radiosurgery reveals the need for something more. Indeed, the method is useful and successful in accounting for gradient nonlinearity distortion. However, the concepts are quite old, especially when considering the speed at which medical and computer technologies have progressed. Of most significance is the fact that MRI technologies were still fairly young at the time of publishing. Some of the issues needing to be dealt with at that time, as well as some of the shortcomings of MRI, are nearly nonexistent today. Advances in technology have successfully eliminated most of the initial problems with MRI. Technological advances have also improved the quality of MR images. At that time, the obtainable resolution on MRI was half of that obtainable on CT (256 x

256 pixels, vs 512 x 512 pixels), limiting the amount of information obtainable from each image. Today, MR image resolution is on par with CT. Additionally, performing scan sequences on previous generation MRI scanners was very slow, requiring a great deal of time. Modern scanners, which may still be slower than modern CT scanners, are much faster than their 15-year-old predecessors. Finally, the greatest change introduced with MRI advancements is an increase in severity of gradient nonlinearity distortion. Since today's scanners utilize wider bores and stronger gradient fields than scanners of old, the problem is even greater now than in the past. Therefore, the correction method must be strong enough as well to handle the increasing severity of gradient nonlinearity distortion.

Indeed, the concepts presented and discussed are useful. But the work is simply too out-dated, given that technology in all aspects has progressed so quickly over time. This does not by any means invalidate the work. But a newer, more up-to-date method is needed to properly handle the changing demands of MRI.

1.8.3 Using Pre-Defined Specifications and Parameters

Nearly simultaneously with the development of this work, researchers at the NMR facility at Massachusetts General Hospital, Harvard (MGH) proposed a new method of distortion correction [36]. This correction is based on knowledge of the characteristics of the magnetic gradient coils in the scanner. Their premise was that if the properties of the gradient coils are known, then the distortion caused by the coils could be calculated directly. This is known as one of the two universally accepted correction methods [36, 33, 37, 38, 39].

The method used to calculate the distortion correction is very straightforward and direct, not requiring a great deal of complex calculations. Successful implementation of the procedure is accomplished by obtaining the manufacturer's specifications of the magnetic gradient coils. Included with these specifications is an expansion of the spherical harmonics model (see Section 1.7 and Equation 1.28). This information is then used to geometrically transform each image to remove the effects of the distortion. The results presented do indeed reflect the feasibility of such an implementation.

However, the entire correction method is based solely on one source: the manufacturer specifications. This proves to be a very significant shortcoming in a distortion correction method, due to the following reasons:

- The spherical harmonics expansion provided by the manufacturer is based on calculations performed on one machine. Each machine will vary slightly from the machine tested by the manufacturer. Therefore, each scanner produced will differ from the manufacturer's specifications.

- Performing regular maintenance on the scanner will slightly change the characteristics of the gradient coils, thereby increasing the inaccuracy of the factory specifications. Indeed, the magnetic field is very stable over time, but every change in the system will produce small variations in the field characteristics.

- The correction requires the information to be acquired from the manufacturer, increasing the complexity and difficulty of performing such a method.

Regardless of the amount of work required to obtain the factory specifications (which may not be much of an issue), the fact that the correction method is based on

static values opens the door to calculation errors. The characteristics of the gradient coils simply do not remain perfectly constant over time. This is the sole reason for the very expensive annual maintenance costs required to properly deploy MRI: to ensure that the scanner is *as close* to factory specifications as possible. No machine will ever match the manufacturer's specifications *exactly*. The specifications may be loose enough such that the errors introduced in the correction method do not allow for submillimeter accuracy in target localization.

Even though the method is successful in its implementation, there is not enough confidence in terms of being able to achieve high geometric accuracy in localizing targets across various MRI scanners. The methods employ here seem eerily similar to those employed by the scanner filtering techniques, in that the correction is based on static, predefined measurements. A more robust, flexible method is needed to achieve the goals outlined by LLUMC.

1.8.4 Calculating Distortion Correction Indirectly

Perhaps the work that provided the greatest influence in this research topic area was that of Langlois, et al [33]. The method proposed by Langlois was the first work that discussed a more direct approach to correcting gradient nonlinearity distortion. Today, this work is widely recognized as the primary source for this subject, and is referenced by other works [36].

To perform the correction, Langlois followed the lead of several other researchers, using a truncated spherical harmonics model to mathematically model the distortion. Calculating the coefficients of the model in this method provides a representation of

the distortion inherent in the images. The correction method is then accomplished by inverting this model, to obtain an "undistortion" model, or a mathematical expression used to remove distortion from the images.

Although this method seems simple and straightforward, it is not without its limitations. There are many dangers associated with inverting large polynomial functions; the spherical harmonics model is definitely one of them. Even though the model used is only a truncated version (the full mathematical expression is an infinite sequence of terms), the complexity of the expression is still quite high. Inverting any polynomial function, no matter how complex or large, will always result in calculation errors. This is not so much a product of the algorithm used (although the algorithm design does in fact contribute to the size of the errors produced); it is a simple inherent trait of computer design. In mathematical theory, a true inversion of numbers, matrices, or expressions is always perfect, in that the inverse provides a true opposite of the original input. However, computer science would say otherwise. In order to perform a full inverse, an infinite amount of storage space is required. Unfortunately, today's computers do not have an infinite amount of storage space at their disposal. This limitation leads to one often-disregarded fact: numbers cannot be stored with infinite precision, because the machine is not infinite. Therefore, every number that is stored must be truncated or rounded, in order to fit into a specified storage space. This finite precision is the inherent problem with all modern computers. ALL calculations—botm simple and complex—are subject to rounding and truncation errors. Therefore, the goal in algorithm design is not just to implement an algorithm, but to provide the *most accurate* method of implementation, one that minimizes the errors

produced.

In terms of the correction method proposed by Langlois, the inversion of the spherical harmonics model is subject to significant rounding and truncation errors. A general rule of thumb to follow is: the more calculations performed, the more the errors grow, since more values are being passed, stored, and truncated. Inverting a complex polynomial function involves many, *many* calculations. Therefore, the inversion is also prone to many, *many* rounding and truncation errors. Indeed, the individual errors themselves are quite small. However, over time, as more calculations are performed, the size of the total error can grow quite quickly. The *error propagation* can get out of control, leading to an inaccurate result, and therefore an improper distortion correction.

By no means is any distortion correction method free from rounding and truncation errors. For that matter, no correction method will ever be. But steps can be taken to *reduce* the amount of error produced during the calculation process. Even though Langlois' method proves to be successful, the accuracy achieved (> 1 mm) is still not sufficient for LLUMC's requirements for target localization. A reduction in the number of calculations performed in the distortion correction method will greatly enhance the accuracy of the final result. The easiest method in which to achieve this is to calculate the distortion correction *directly*, rather than inverting the spherical harmonics expansion. Not only will this preserve the accuracy of the final result, but will also increase the efficiency of the algorithm.

2. SOFTWARE-BASED GRADIENT NONLINEARITY DISTORTION CORRECTION

The work presented in this thesis describes the implementation of a software-based method to correct gradient nonlinearity distortion, to provide submillimeter accuracy in localization of anatomical regions[1]. This thesis continues where previous works left off, as well as build upon the techniques first presented in earlier works [42, 43, 44, 37, 38, 33]. Analysis of other published works reveals the need for an entirely new approach to the issue of handling gradient nonlinearity distortion. Unfortunately, current methods available presently do not satisfy the requirements LLUMC set forth for their MRI-based functional proton radiosurgery system. Although the methods described previously are successful in handling gradient nonlinearity distortion, none of them provide the high level of geometric accuracy required to perform noninvasive proton neurosurgery.

The filtering methods provided by the MRI scanners (Section 1.8.1) are not sufficient in correcting gradient nonlinearity distortion. A three-dimensional correction is needed to achieve submillimeter accuracy in all three planes. Since these methods

[1] To simplify the nomenclature used in this work, the method implemented here will be referred to simply as a "distortion correction" method from this point forward. Keep in mind that the correction only handles gradient nonlinearity distortion; it does not correct for all forms of distortion. For the sake of simplicity, the method will not be called a "gradient nonlinearity distortion correction" method.

only provide correction in two dimensions, accurate target localization is simply not possible with these images. Furthermore, the methods only *reduce* the amount of visible distortion, to enhance the appearance of the images; the filtering methods are not true corrections. Therefore a dedicated three-dimensional technique is needed.

The fusion of CT and MRI images (Section 1.8.2) is not a suitable method. The geometric accuracy achievable through such methods does not meet the guidelines set forth by LLUMC. Therefore, a more robust method must be implemented, in order to successfully achieve submillimeter accuracy.

A system that is based on static, pre-calculated values (Section 1.8.3) will not be as robust, flexible, or accurate as a system that calculates the distortion dynamically. Discrepancies between the manufacturer specifications and the actual distortion inherent in each particular scanner will introduce errors into the final correction result. Additionally, such a method is specific to each manufacturer, requiring the need for close support from the manufacturer. Therefore, the correction method must be able to calculate the distortion dynamically, independent of the scanner make and model.

Implementing a distortion correction method that involves inversion of complex polynomials (Section 1.8.4) is not only inefficient, but is also prone to calculation errors due to rounding and truncation. Error propagation as a result of performing calculations on computers is indeed a great challenge to surmount. By no means is the correction method presented in this work free from rounding or truncations errors. For that matter, no correction method will ever be! Therefore, the goal of this work is to provide not just an effective distortion correction method, but also one that minimizes error propagation. The simplest way to suppress error propagation is

to minimize the number of calculations performed.

2.1 Approaching the Problem

Outlining the shortcomings of current methods provides a clear list of requirements for a robust, accurate software-based gradient nonlinearity distortion correction:

1. A three-dimensional correction method must be implemented, that goes beyond the scanner filtering techniques. Additionally, the correction method must be able to handle all images that a scanner produces, filtered and unfiltered.

2. The correction method must be specifically designed for MRI to calculate a distortion correction, without fusing CT image data.

3. Dynamic calculation of the gradient field nonlinearity is required to achieve submillimeter accuracy, rather than relying on static specifications and values provided by the scanner manufacturer.

4. Calculation of the gradient field nonlinearity needs to be performed directly, rather than inverting the spherical harmonics expansion.

5. Efficient algorithm design must be the foremost concern in implementing a distortion correction method, to reduce the number of calculations necessary to compute the result. Recall that the size and number of rounding and truncation errors is directly proportional to the number of calculations performed.

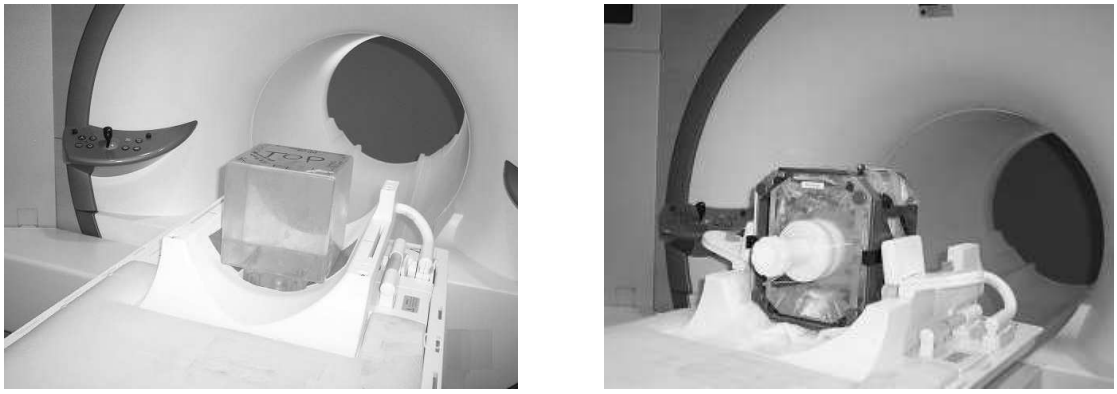
6. The correction must be accurate to less than one millimeter, in order to have the ability of promoting submillimeter accuracy in target localization.

7. The software should produce valid results in a timely fashion.

8. The software should have the flexibility of running on a wide range of computer systems, without requiring special hardware.

The distortion correction method described in this thesis was designed with all these features in mind. Additionally, the method incorporates some of the highlights of previous works, to create a robust, flexible, accurate system, capable of achieving submillimeter accuracy in target localization.

2.2 Experimental Setup



(a) Oil-filled phantom.

(b) Lucy phantom.

Fig. 2.1: Phantoms in the Siemens Sonata MRI scanner.

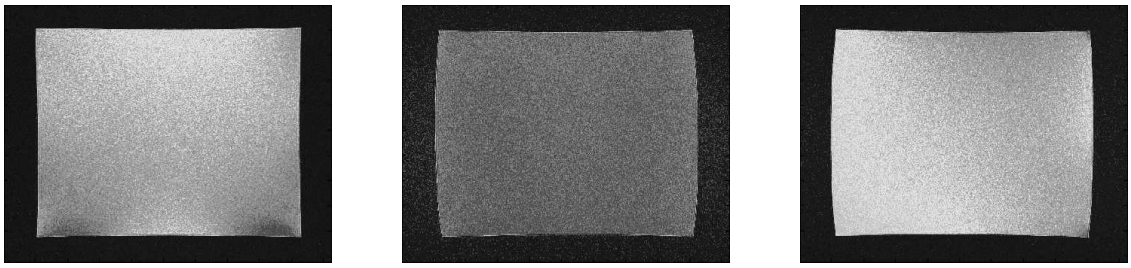
The gradient nonlinearity distortion correction method created for this thesis was developed in a fashion similar to previous works [33, 36]. As with these works, a calibration unit, known as a *phantom*, was used to obtain the scan images. Two phantoms were used in this work: one for development of the software, and one for

testing and verification. The phantom utilized for software development is a Plexiglas cube, shown in Figure 2.1(a). Because of the properties of Plexiglas, very little signal response is produced as a result of scanning. Therefore, the images obtained represent the signal response of the oil, which corresponds to the interior dimensions of the phantom (159.50 mm x 159.70 mm x 158.11 mm). The second phantom, called the Lucy phantom, is a plastic sphere with a cylindrical base, see Figure 2.1(b). Inside the Lucy phantom are locations with which to place MRI markers containing oil. To enhance the appearance of the markers in the images, water was used as a contrast agent.

All the MRI images were acquired on a 1.5 T whole body scanner (Magnetom VISION, Siemens, Germany) with a standard head coil. The phantoms were centered on the gradient isocenter of the scanner to within 1 mm. To maximize the amount of available data for correction, axial, coronal, and sagittal sequences were acquired. For the verification stage, only axial and coronal sequences were acquired for the Lucy phantom.

For sequences performed using the oil-filled phantom, a matrix of 512 x 512 pixels, 0.391 mm pixel size, and a field of view of 200 mm were chosen. One slab of 104 slices of 2.0 mm thickness was acquired for each sequence. Thus, the voxel size was 0.391 mm x 0.391 mm x 2.000 mm. Figure 2.2 shows representative axial, coronal, and sagittal images of the phantom, with distortion clearly visible in the outer regions.

For sequences performed with the Lucy phantom, a matrix of 512 x 512 pixels, 0.586 mm pixel size, and a field of view of 300 mm were chosen. One slab of 112 slices of 2.0 mm thickness was acquired for each sequence. Thus, the voxel size was



(a) Axial scan. (b) Coronal scan. (c) Sagittal scan.

Fig. 2.2: Examples of useful slices of oil-filled phantom.

0.586 mm x 0.586 mm x 2.000 mm. Figure 2.3 shows representative axial and coronal images of the Lucy phantom.

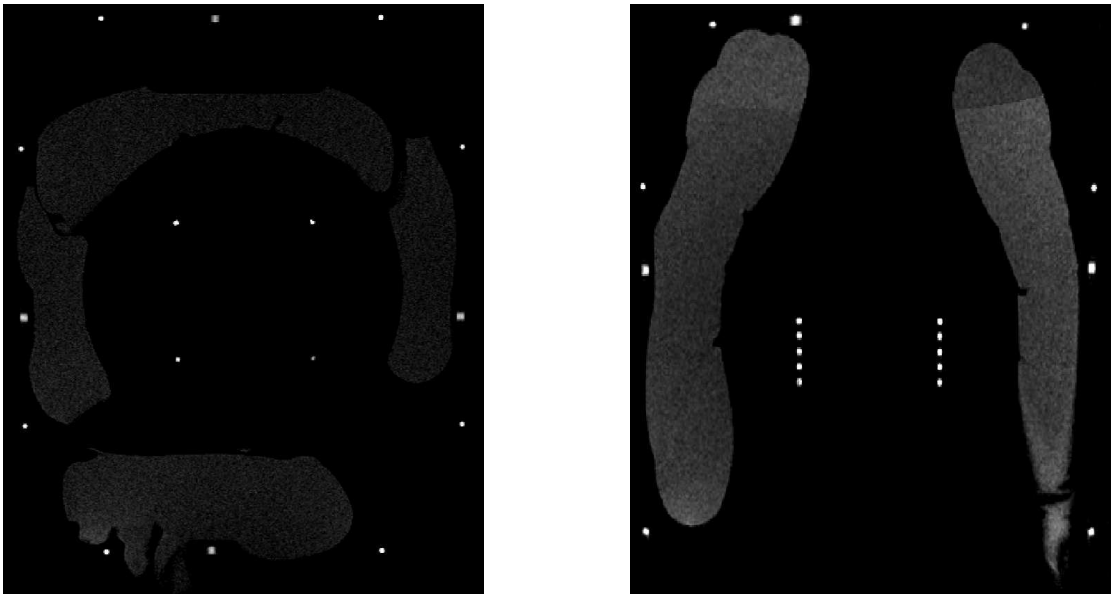
The software package performing the gradient nonlinearity distortion correction was developed with Matlab version 7.0.4.365 Service Pack 2. Matlab was chosen because of its robust support for the Digital Imaging and Communications in Medicine (DICOM) standard, in which all MRI images are encoded. Additionally, Matlab provides an extensive imaging processing library, which this project makes use of in certain sections. Finally, the Matlab environment is simple and straightforward, allowing for easy software maintenance and simple porting to various platforms.

2.3 Method

To perform the distortion correction, the following basic steps are performed:

1. Scan the oil-filled phantom. This provides the system with images of a structure with known measurements.

2. Perform quality assurance on the data.



(a) Axial scan.

(b) Coronal scan.

Fig. 2.3: Example scans of Lucy phantom, demonstrating MRI markers (center) and fiducial points (edges).

3. Calculate the distortion correction model, using the spherical harmonics expression given in Section 1.7.

4. Test and verify correction either by providing corrected three-dimensional coordinates of target regions (in Lucy phantom), or correct entire MRI images.

Since there are *many* specific steps involved in this implementation of gradient nonlinearity distortion correction, it is best to provide a general overview of the entire process prior to delving deep into the detailed analysis of each portion of the method.

2.3.1 Data Quality Assurance (Preprocessing)

The first step in performing gradient nonlinearity distortion correction is to first obtain the data that will be used to calculate the correction. Scanning the oil-filled phantom will yield images similar to those shown in Figure 2.2. As previously stated, the signal response obtained in the images represents the interior dimensions of the phantom. Prior to calculating the parameters of the distortion correction model, the data undergoes a series of image processing and quality checks described in detail in Chapter 3. The quality checks have three main goals:

1. Eliminate as much image noise as possible.

2. Remove extraneous and unwanted features in each MR image.

3. Ensure the data is as accurate as possible for future calculations.

The ultimate accuracy of the correction will depend on the quality of the data produced after these data processing steps. Each quality check is executed as an

individual module during the initial data processing stage. What follows is a basic overview of the primary data quality assurance algorithms. All algorithms are presented in Chapter 3.

Selection of Useful Slices

The distortion correction should only be based on slices that contain a full view of the phantom, as shown in Figure 2.2. To select useful slices, the algorithm analyzes the pixel intensities of each slice and compares them to a reference slice near the center of the slab, which will always contain a full view of the phantom. Refer to Section 3.3.3 for more details. The central slice is not chosen as a reference because it contains the phantom's drain plug, which gives an inaccurate edge representation and intensity distribution (see Section 3.3.5). As the first step in the selection of useful slices, edge detection (see Section 3.3.4 and Appendix D) is performed on the reference slice, identifying the region that contains the phantom. Next, the maximum pixel intensity of the "empty" region surrounding the phantom (low-intensity value) and the minimum pixel intensity in the central 20% area of the reference slice (high-intensity value) are found. Two criteria are used to define useful slices:

1. The number of pixels with equal or larger than the high-intensity value has to be within 1% from the reference count, effectively eliminating slices without the phantom.

2. The relative number of pixels with intensities between the low- and high-intensity values has to be less than 40%, thereby excluding slices with a partial view of the phantom.

The indexes of the first and last useful slices are reported, and slices outside this range are excluded from further analysis.

Edge Detection

Edge detection is performed to provide the representation of the distorted physical edges of the phantom for the gradient nonlinearity distortion correction. The Canny edge detector (ED) provided by Matlab was chosen because it was least sensitive to noise, and the edge images produced were sharper and more accurate, thus delivering the most consistent results from study to study, when compared to other methods implemented in Matlab (see Appendix D). When invoking the Canny ED, Matlab will automatically calculate a threshold for edge detection. However, this threshold considers every single intensity shift within the image, and in doing so, finds fictitious edges within the phantom (see Section 3.3.4 and Chapter D). In order to find the optimum threshold value for this application, a threshold detection algorithm is utilized. This algorithm converges on the optimal value through an iterative series of edge detection trials. For the threshold algorithm, the reference image already selected for the definition of useful images is used. The threshold algorithm automatically configures its settings (threshold increment and stop condition) based on the strength of the image gradient across the edges. In the baseline iteration, Canny ED is performed using Matlab's automatically calculated threshold. Then, the threshold algorithm increments the threshold in small successive steps (0.5-0.75). As the threshold is increased, the number of fictitious edges decreases, leaving only the strong image features behind. The optimal threshold is assumed when the number of edge

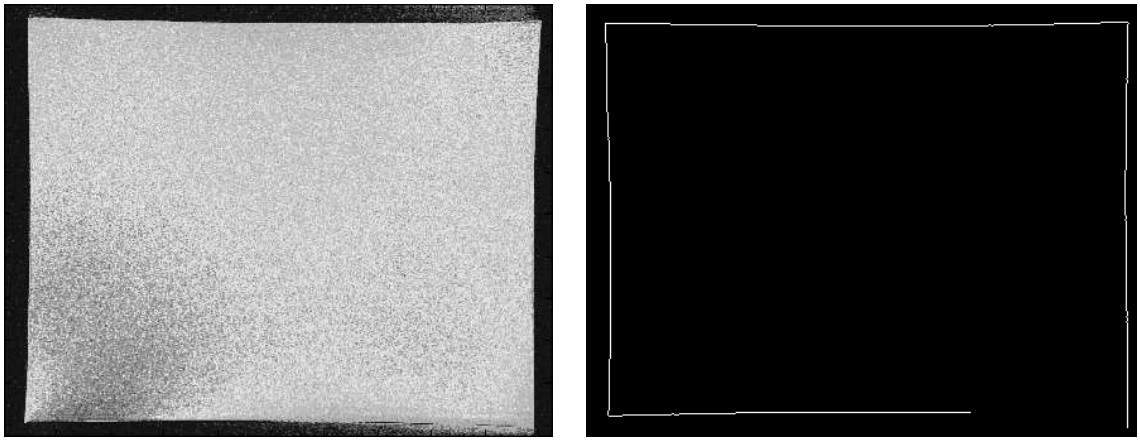
points does not change by more than 1% over several successive iterations (5-10, depending on the stop condition). This optimized threshold is then used with Matlab's Canny ED on the entire set of useful slices. In a series of tests, it was found that this will mostly retain the physical edges of the phantom in edge images. The remaining unwanted image features have to be dealt with separately. These steps are covered in following sections.

Drain Plug Removal

One image feature in the scan of the oil-filled phantom that is not removed during edge detection is the phantom's drain plug (see Section 3.3.5). The drain plug is simply a screw that is located underneath the center of the posterior face. The threads of the screw are present in edge images acquired from axial and sagittal sequences. Tests showed that failure to remove the image of the plug results in improper calculations in all subsequent data processing steps. Therefore, a plug removal algorithm is necessary to correct this condition. In the first step of the drain-plug removal process, edge images are oriented such that the plug appears on the bottom face of the image. Removal of the plug is then performed by placing a window that has a width of 25% of the phantom width about the center column of each image. The plug removal algorithm removes all image data within this window, leaving behind a hole in the top and bottom phantom edges. Both holes are located by the algorithm and filled by a straight line. Since the deleted areas are located in the center of these faces, the distortion is small, meaning that the edges of the phantom are nearly straight. At worst, the corrected portion of the edges is shifted by one pixel to either side of the

edge. It was found that in all phantom scans, the plug removal algorithm properly connected the edge with a straight line, without introducing any gaps.

Edge Verification and Repair



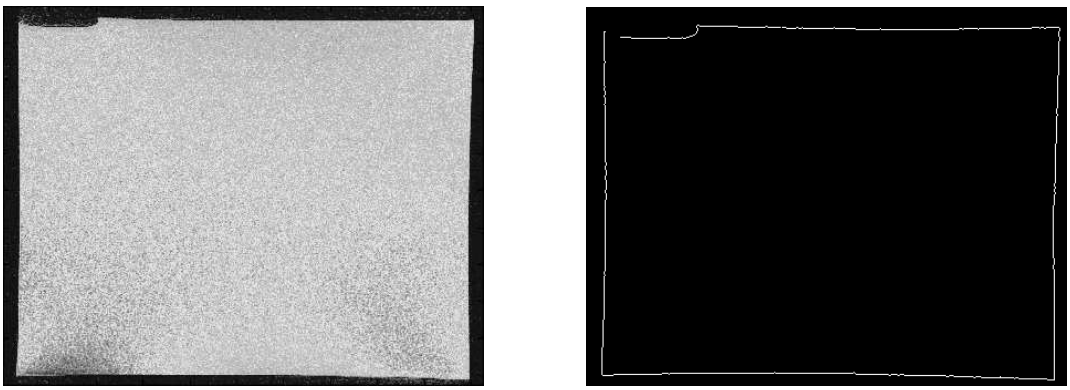
(a) Slice with oil leakage. (b) Result of edge detection.

Fig. 2.4: Slice exhibiting oil leakage.

The next step involves the detection of incomplete edge images, i.e., those with missing edge data (see Section 3.3.6). For example, in Figure 2.4, the phantom had leaked oil along the left posterior surface. The edge detection resulted in a large defect in the bottom right corner of several edge images. In the edge-verification step, all edge images in an image sequence are checked for completeness. First, the location and size of the phantom is calculated by the edge verification algorithm. Based on this information, the phantom edge image is split into four equal quadrants, and the total number of horizontal and vertical edge points in each quadrant is counted. In a complete edge image, the phantom will contain nearly the same number of edge points

in each quadrant. On the other hand, in an incomplete edge image, the number of edge points in one quadrant will be different. These images, and the quadrant that appears anomalous, are flagged. Based on user preferences, problematic images can either be discarded from further consideration, or the images can be repaired. Image repair is accomplished by copying and grafting an intact quadrant from the opposite side in place of the flagged quadrant. It was found that the resulting image is very similar to that of a "natural" edge image, and is sufficient for use in further calculations. This repair worked successfully for all slices affected by the leaking oil, shown in Figure 2.4.

Detection and Removal of Air Bubbles



(a) Slice with large air bubble. (b) Result of edge detection.

Fig. 2.5: Slices containing a large air bubble.

Due to the use of oil as a medium for the phantom, it is virtually impossible to remove all air bubbles. Small air bubbles that float within the confines of the

phantom are not problematic, but those attached to the phantom surfaces can distort the phantom edges. Since the signal response seen in the MR images is that of the oil, air bubbles along the edges appear as concave defects of varying size. Air bubbles attached to the phantom corners give the false impression of a rounded corner. The study performed on the leaking phantom mentioned above resulted in a large air bubble in the right anterior corner of the phantom, as shown in Figure 2.5(a).

Air bubbles are detected in several steps (see Section 3.3.7). First, the locations of the four corners in each image are determined using a specifically-designed corner detection algorithm, see Section 3.3.13. The algorithm defines a corner as the point shared by a horizontal and vertical edge (i.e., where those two edges meet). The corners are located by scanning up and down the vertical edges of the phantom (depending on which corner is to be located) until the start of the adjacent horizontal edge is found. When the horizontal edge is found, and there are no additional edge points found along the vertical edge, the true corner is considered located. This extra step is taken to avoid the algorithm getting confused by an air bubble along that edge, as air bubbles may resemble corners. By scanning beyond a possible corner location, the algorithm will disregard air bubbles along an edge as possible corners, and will only report the location of the true corner.

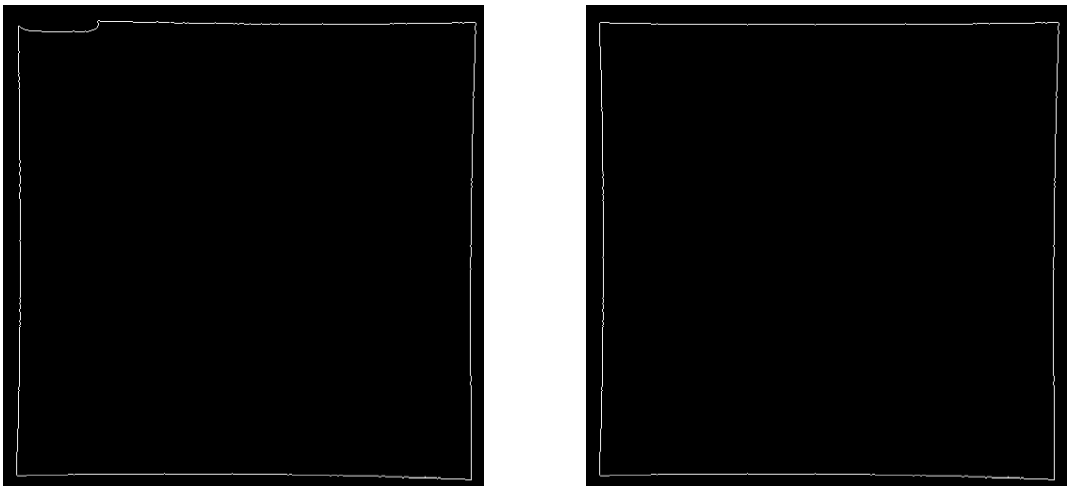
The corner data is then used by an edge-tracking algorithm (see Section 3.3.14), which was implemented to detect air bubbles, as well as to accomplish other secondary tasks. Starting from one corner, the algorithm moves along the edge, pixel by pixel, recording the location of each edge point. Horizontal edges are scanned from left to right, and vertical edges are scanned from top to bottom. The edge tracker will stop

once it has reached the opposite corner.

In the next step, the edge slope (see Section 3.3.7) is calculated for edge intervals of 5 points. Along a "normal" edge, the edge slope does not exceed one pixel over this range. On the other hand, an air bubble will cause a sharp positive (>1 pixel) rise in slope, which is followed by a similar drop on the opposite side of the bubble. The range between the rise and drop of the slope is considered to be the limits of the bubble. If the bubble is located on the edge rather than in a corner, it is erased, creating a hole in the edge, and the image is processed by a hole repair algorithm, introduced in the next section and described in-depth in Section 3.3.9. On the other hand, air bubbles located in a corner are treated by replacing the entire corner with a copy of the opposite bottom corner, to avoid inserting another bubble in place of the original. The likelihood of having air bubbles in both bottom corners of one image was considered as very low. The bottom corners are tested for air bubbles first, and are corrected by the mirrored opposite bottom corner, if necessary. Next the top corners are tested for bubbles and replaced by the corresponding bottom corner, if air bubbles are detected. Figure 2.6 demonstrates the results of removing the large air bubble present in the study with the leaking phantom.

Hole Repair

Holes can occur in an edge either as a result of noise, or some previous action taken by another correction algorithm (i.e., bubble repair). A hole is simply a discontinuity in an edge. Holes that occur as a result of noise are quite small, usually spanning only one or two pixels. On the other hand, holes that result from previous correction



(a) Before correction. (b) After correction.

Fig. 2.6: Demonstration of bubble correction on large corner bubble.

procedures may be many pixels wide. A hole-repair algorithm is utilized to correct for these conditions, see Section 3.3.9. When invoked, the algorithm draws a straight-line approximation of the edge to fill the hole. First, holes are located by using the edge tracking algorithm. The error condition of the edge tracker occurs when a hole is reached. Here, the edge tracker returns the list of edge points, up to the start location of the hole, and quits. The hole-repair algorithm will then locate the end location of the hole by scanning beyond the hole start location in the direction of the edge until the opposite side of the hole is found. Next, the algorithm calculates the horizontal and vertical distance between the edge points bordering the hole. A straight line that connects the border points of the hole is then calculated and used to fill the hole defect.

2.3.2 Plane Calculation

With the quality checks and repair algorithms in place, the parameters of the distortion correction model can be estimated. The data is ensured to be free of nearly all defects and blemishes, which ensures the success and accuracy of the distortion correction. Linear least squares fitting methods are used to create the ideal phantom planes and to estimate the parameters of the distortion correction model, which is based on the sum of spherical harmonics, presented in Section 1.7. Once the model is known it can be applied to all studies performed on the particular scanner.

Midplane Calculation

In the second stage, the processed distorted data is used to model the undistorted surfaces of the phantom, see Section 4.1. The final gradient nonlinear distortion correction method is based on a set of calculated "ideal" planes. These planes represent the appearance of the faces of the phantom, without subjection to distortion. Although these planes are estimations of the actual faces, they can be calculated to a high degree of accuracy. The errors that result from these calculations are quite minute, and do not adversely affect the final calculations. Each pair of opposite edge points in the edge images is used to calculate a midpoint. The result of midpoint calculation along an entire edge produces a midline, see Figure 2.7. The algorithm excludes those points in the lists that do not have opposite neighbors, to ensure that the midpoint calculations generate valid results. The edge lengths can vary depending on the appearance of the distortion in the image, so it is acceptable to have opposite edges of slightly different lengths; the algorithm removes unpaired edge points to

obtain edges of equal length.

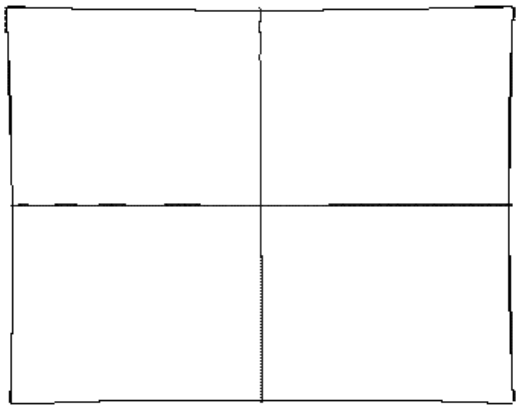


Fig. 2.7: The upper and lower edge points of the image are averaged to form two midlines, shown as the horizontal and vertical lines intersecting in the middle of the image. A plane is then fit to all the slices' midlines.

Two midlines are calculated for each edge image: one horizontal, and one vertical. Due to the symmetry of gradient nonlinearity distortion, these midlines are nearly straight, and provide a suitable representation of the orientation of the undistorted edges in each image, disregarding the positioning of the edges. The midpoints from the entire set of images serve as the data that models the ideal midplanes with typical studies having around 30,000 midpoints. This midplane represents the ideal shape, size, and orientation of the particular surface on the phantom. A plane equation is calculated that best fits the midplane data set in the least squares sense. The equation of a plane is of the form:

$$N^T x = d \tag{2.1}$$

where N is the normal vector to the plane, d is distance of the plane from the

origin, and x is the set of vectors satisfying the plane equation.

For example, fitting a plane to the top and bottom edges of the images in an axial study results in a plane oriented in xz. Therefore, the plane can be expressed by:

$$y = -\frac{N_x}{N_y}x - \frac{N_z}{N_y}z + \frac{d}{N_y}$$

$$= C_x x + C_z z + C_c$$

$$= \begin{bmatrix} x & z & 1 \end{bmatrix} \begin{bmatrix} C_x \\ C_z \\ C_c \end{bmatrix} \tag{2.2}$$

Thus based on the determined orientation, the algorithm organizes the data into one large matrix of dimension $n \times 3$, and one large array of dimension n, where n is the number of midpoints. The $n \times 3$ matrix is the *design matrix*, denoted by A, and the n-element array is the *observation vector*, denoted by b. The resulting problem is $A\chi = b$, where χ is a vector of the plane coefficients. This equation can then be solved for the midplane coefficients using a technique like QR or SVD.

Ideal Plane Calculation

The midplanes describe the correct orientation of the ideal phantom planes, but they are not in the proper location. The basic idea of this section is to shift the midplanes along its normal vector by one half the physical dimensions of the phantom to their correct location in 3-dimensional space (see Section 4.3.9). The three phantom dimensions were derived from dimensional inspection and stored prior to algorithm execution. The result of this step is a set of six ideal planes, one for each of the six square faces of the phantom. These ideal planes describe how the faces of the phantom would theoretically appear in the absence of gradient nonlinearity distortion, see Figure 2.8. Note that these ideal planes are close but not identical to fitting planes to the original sides, as the averaging forces the opposing faces to be parallel and the shifting forces the faces to be at the right distance from each other and the midplane.

To shift the midplane in terms of the coefficients of the midplane, only the offset term is changed as the other coefficients must remain the same in order to maintain the midplane's original orientation. To calculate the shift, consider the plane equation, Equation 2.1, in terms of the coefficients defined in Equation 2.2 of the example in the last section:

$$N^T x = C_c$$

$$N = \begin{bmatrix} -C_x \\ 1 \\ -C_z \end{bmatrix} \tag{2.3}$$

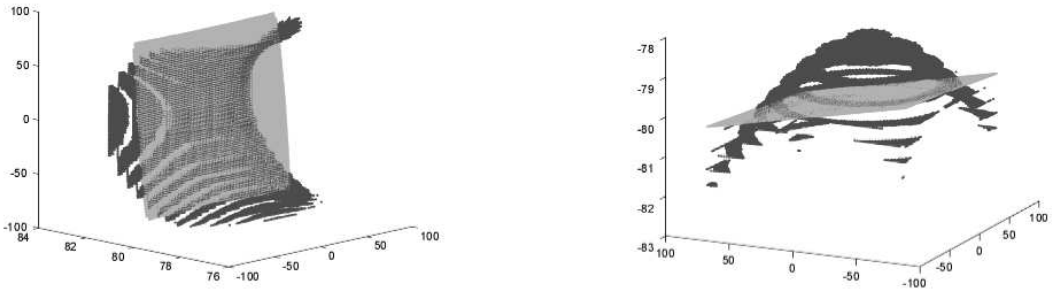


Fig. 2.8: Distorted faces produced by MR scanner, and the ideal planes describing the phantom that produced them. The axis dimensions are pixels, and note the curvature is exaggerated so that it is visible.

The normal vector must be normalized in order to prevent any unnecessary weighting in the final result. Normalization of N produces the unit normal vector, $\bar{N}$:

$$\bar{N} = \frac{N}{\| N \|}$$

$$\begin{bmatrix} = \frac{-C_x}{\sqrt{C_x^2+1+C_z^2}} \\ \frac{1}{\sqrt{C_x^2+1+C_z^2}} \\ \frac{-C_z}{\sqrt{C_x^2+1+C_z^2}} \end{bmatrix} \tag{2.4}$$

Consider a point,

$$p = \begin{bmatrix} 0 \\ C_c \\ 0 \end{bmatrix} \tag{2.5}$$

that lies on the midplane in our example. It is now possible to determine the position of point p on the shifted plane. The shifted point is:

$$p_2 = \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix}$$

$$= p \pm \delta \bar{N}$$

$$= \begin{bmatrix} \frac{\pm -C_x \delta}{\sqrt{C_x^2+1+C_z^2}} \\ C_c \pm \frac{\delta}{\sqrt{C_x^2+1+C_z^2}} \\ \pm \frac{-C_z \delta}{\sqrt{C_x^2+1+C_z^2}} \end{bmatrix} \tag{2.6}$$

where δ is one half the phantom dimension, and the $\pm$ determines positive or negative shift needed to specify both ideal planes resulting from the midplane. Inputting p_2 into the original plane expression, Equation 2.1, will make it possible to solve for the offset term for the shifted plane.

$$D_{shifted} = N^T p_2$$

$$= \begin{bmatrix} -C_x \\ 1 \\ -C_z \end{bmatrix}^T \begin{bmatrix} \frac{\pm -C_x\delta}{\sqrt{C_x^2+1+C_z^2}} \\ C_c \pm \frac{\delta}{\sqrt{C_x^2+1+C_z^2}} \\ \pm\frac{-C_z\delta}{\sqrt{C_x^2+1+C_z^2}} \end{bmatrix}$$

$$= \frac{\pm - C_x^2\delta}{\sqrt{C_x^2+1+C_z^2}} + C_c \pm \frac{\delta}{\sqrt{C_x^2+1+C_z^2}} \pm \frac{-C_z^2\delta}{\sqrt{C_x^2+1+C_z^2}}$$

$$= C_c \pm \delta\sqrt{C_x^2+1+C_z^2} \tag{2.7}$$

Therefore, we see that the amount to shift the midplane is equal to $\delta\sqrt{C_x^2 + 1 + C_z^2}$. The same method follows as well for the other planes.

2.3.3 Calculate Distortion Correction Parameters

The third and final stage involves performing the actual gradient nonlinearity distortion correction. The distortion correction model is based on the sum of spherical harmonics. Using the ideal planes obtained from the previous stage, theoretical "undistorted" data points will be calculated using the ideal plane equations and acquired data points. The undistorted data points are then used to calculate the coefficients of the distortion correction model. Three sets of coefficients are necessary, one for each dimension. Once the model is known it can be applied to all studies performed on the particular scanner. In particular, let x_i, y_i, and z_i be the distorted coordinates of some point i measured by an MR scanner. Further let $\bar{x}_i$, $\bar{y}_i$, and $\bar{z}_i$ be the undistorted coordinates that corresponds to x_i, y_i, and z_i that is $(\bar{x}_i, \bar{y}_i, \bar{z}_i)$ is the point on the ideal plane that produced the point (x_i, y_i, z_i) in the MR image. Taking the first spherical harmonics, derived from Equation 1.28, the correction model takes the form:

$$\bar{x}_i = x_i(1 + K_{x_0}(x_i^2 + y_i^2) + K_{x_1}z_i^2 + K_{x_2}z_i^2(x_i^2 + y_i^2) + K_{x_3}(x_i^2 + y_i^2)^2 + K_{x_4}z_i^4) \quad (2.8)$$

$$\bar{y}_i = y_i(1 + K_{y_0}(x_i^2 + y_i^2) + K_{y_1}z_i^2 + K_{y_2}z_i^2(x_i^2 + y_i^2) + K_{y_3}(x_i^2 + y_i^2)^2 + K_{y_4}z_i^4) \quad (2.9)$$

$$\bar{z_i} = z_i(1 + K_{z_0}(x_i^2 + y_i^2) + K_{z_1} z_i^2 + K_{z_2} z_i^2 (x_i^2 + y_i^2) + K_{z_3}(x_i^2 + y_i^2)^2 + K_{z_4} z_i^4) \quad (2.10)$$

Note that all the terms in these equations are known except the K_{x_j}, K_{y_j}, and K_{z_j} terms, which are the desired correction parameters. Solving for the K_{x_j} is accomplished by using the ideal planes whose normal is in the x direction, i.e. the yz planes, and similarly for the K_{y_j} and K_{z_j} terms. Since the original example presented was for the top and bottom edges of the images in an axial study results, i.e. a plane oriented in xz, the solution for this example will follow that case. The other cases follow directly.

Begin by noting that $\bar{y_i}$ can be calculated using the ideal plane. Since the distortion for this face is essentially all in the y direction, the ideal x and z coordinates ($\bar{x_i}$ and $\bar{z_i}$) are assumed to be the same as the distorted (x_i and z_i).

$$C_c \pm \frac{\delta}{\sqrt{C_x^2 + 1 + C_z^2}} = N^T p_{ideal}$$

$$= \begin{bmatrix} -C_x \\ 1 \\ -C_z \end{bmatrix}^T \begin{bmatrix} x_i \\ \bar{y_i} \\ z_i \end{bmatrix}$$

$$C_c \pm \frac{\delta}{\sqrt{C_x^2 + 1 + C_z^2}} + C_x x_i + C_z z_i = \bar{y}_i \tag{2.11}$$

Knowing the ideal point $(x_i, \bar{y}_i, z_i)$ and the distorted points (x_i, y_i, z_i), substitute into Equation 2.9 and note that it is linear in the K terms. Using all n points of both planes oriented in the xz direction yields the following:

$$S = TK_y$$

$$S = \begin{bmatrix} \bar{S}_0 \\ \bar{S}_1 \\ \vdots \\ \bar{S}_{2n-1} \end{bmatrix}$$

$$S_i = \bar{y}_i - y_i$$

$$T = \begin{bmatrix} T_{0,0} & T_{0,1} & T_{0,2} & T_{0,3} & T_{0,4} \\ T_{1,0} & T_{1,1} & T_{1,2} & T_{1,3} & T_{1,4} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ T_{2n-1,0} & T_{2n-1,1} & T_{2n-1,2} & T_{2n-1,3} & T_{2n-1,4} \end{bmatrix}$$

$$\begin{aligned} T_{i,0} &= y_i(x_i^2 + y_i^2) \\ T_{i,1} &= y_i(z_i^2) \\ T_{i,2} &= y_i(x_i^2 + y_i^2)z_i^2 \\ T_{i,3} &= y_i(x_i^2 + y_i^2)^2 \\ T_{i,4} &= y_i(z_i^4) \end{aligned} \tag{2.12}$$

$$K_y = \begin{bmatrix} K_{y0} \\ K_{y1} \\ K_{y2} \\ K_{y3} \\ K_{y4} \end{bmatrix}$$

Using a least squares solution technique, such as QR or SVD will solve for K_y. This process is repeated for the other correction parameters K_x and K_z. Once the correction parameters are obtained, any MR image from the same scanner can be corrected using Equations 2.8 - 2.10 to solve for the undistorted points $(\bar{x}_i, \bar{y}_i, \bar{z}_i)$, given the distorted image points (x_i, y_i, z_i).

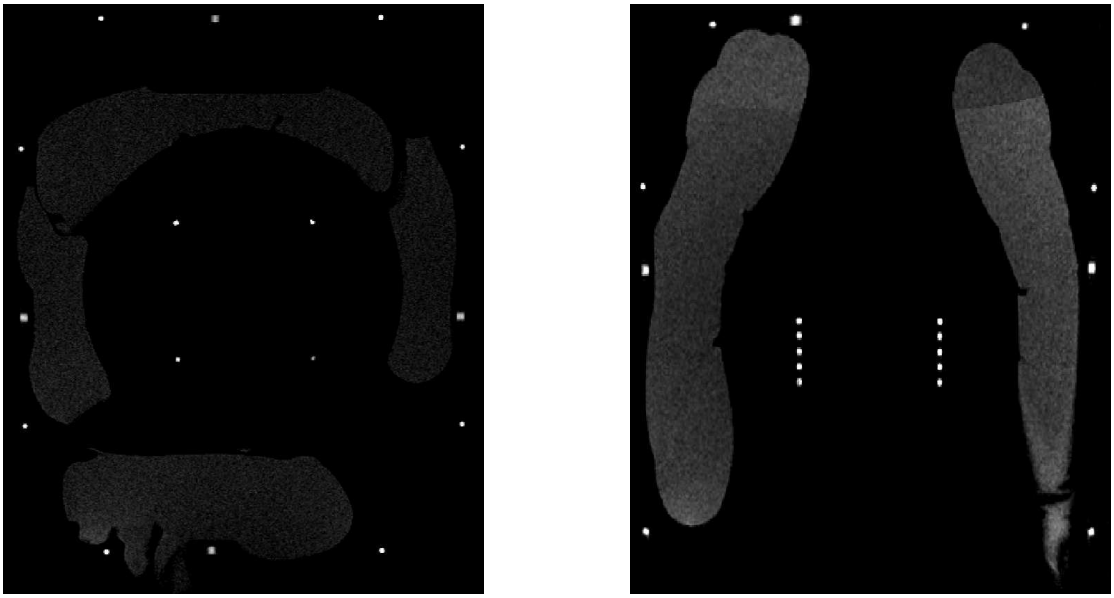
2.4 Testing and Verification of Methods

After performing the distortion correction, the results are tested and verified by using both the oil-filled phantom and the Lucy phantom. Verification of the accuracy of the correction method using the oil-filled phantom will include:

- Ensuring that the edges of the phantom appear straight in the images.
- Verifying that the phantom's dimensions are preserved in the images.
- Confirming that the distortion correction is valid for all images in each slab.

A true verification of the accuracy of the system cannot be established without independent testing and verification. This is where the Lucy phantom comes into play. As previously mentioned, the Lucy phantom contains a series of MRI markers. These appear in the images as bright dots in the center of the images, see Figure 2.9.

The locations of the markers are known with high precision. To test the effectiveness of the distortion correction, the corrected locations of the markers are compared to the actual physical locations. This step requires performing a *stereotactic transformation*, to convert the coordinates in the images to a stereotactic coordinate system set up by the fiducial points on the attached stereotactic frame, see Appendix C.



(a) Axial scan.

(b) Coronal scan.

Fig. 2.9: Example scans of Lucy phantom, demonstrating MRI markers (center) and fiducial points (edges).

The method used here will be the same method physicians will use to define the coordinates of target regions during radiosurgery. If the locations of the markers in the images are within one millimeter to their actual locations on the Lucy phantom, then the correction is verified to successfully meet LLUMC's goal of obtaining submillimeter accuracy in target localization.

2.5 Summary of the Software System

The software method implemented in this work is executed as a series of modules. As described in Section 2.3, there are three primary modules:

1. Data quality assurance.
2. Plane calculation.
3. Distortion correction modeling.

Within each module is a series of smaller modules, each designed to perform one specific task. The advantages to having a modular, hierarchal system such as this are numerous, including:

1. Simplified maintenance, since each step in the distortion correction process is handled by one function.
2. Ease of implementation of the concepts of distortion correction.
3. Flexible architectural design, which promotes easy removal of old functions or addition of new functions.

At each stage in the correction process, an in-depth analysis will be provided. This is to demonstrate the theoretical and mathematical background behind each algorithm. To further understand the implementation of the software, detailed algorithm analyses will also be provided. As mentioned previously, this analysis strays from a conventional algorithm analysis, in that it does not involve computational complexities (i.e., big-oh, big-theta, big-omega). Despite this characteristic, the analysis does provide information necessary to explain both the scientific foundation of the correction method, as well as the programming techniques used to implement the theory.

For each algorithm, a detailed overview of the method employed to solve each particular problem is provided. This overview provides the motivation behind each step in the correction method, and explains its purpose in the entire scope of the project. Relating this overview with the theoretical and mathematical background of the distortion correction process will reveal the methods used in software implementation. A deep understanding of the code that implements the distortion correction is not possible without an overview explaining the theory behind its implementation.

Once the foundational concepts have been presented, the actual algorithm implementation is discussed. This is given as a step-by-step description, to demonstrate exactly what procedures each algorithm performs. The structure and content of the analysis is provided to aid future work in this field, such that these methods can be easily reproduced and repeated to achieve similar results. Additionally, the design serves as a blueprint for future endeavors, simplifying the task of porting the software system to various languages and platforms.

Finally, the results of each individual algorithm is presented, if applicable. This provides insight as how each algorithm contributes to the overall distortion correction process. Additionally, this will serve as a useful tool in future software development, to facilitate debugging and shed light on possible improvements to the algorithms.

The next several chapters will cover the entire software development process. These discussions will also reveal the theory and implementation of this distortion correction process. The challenges encountered during software development are also presented, as well as the unique solutions implemented to overcome them.

3. STEP 1: DATA QUALITY ASSURANCE (PREPROCESSING)

3.1 Purpose

The first step in performing the gradient nonlinearity distortion correction is to prepare the data for correction. Inputting the raw MRI scans into the parameter estimation and correction steps is not only impractical, but extremely foolish. There are many features in the scans that need to be removed before proceeding forward. These include air bubbles, slices that do not contain the phantom, effects of the phantom's drain plug, and random image noise. Extraneous or incorrect data will have adverse effects on the correction steps, and will provide incorrect and utterly useless results.

The process of correcting the gradient nonlinearity distortion can be thought of as being similar to the process of painting a car. The car cannot be simply placed into the paint shop, and have the new paint applied right away. Although this will achieve the proper end result, the results will not be suitable to the customer's needs or expectations. The first step in painting the car is to prep the car's body panels for paint. This would include removing the old paint, repairing any dents or body damage, and removing any rust or corrosion. Performing this initial step is absolutely crucial in order to achieve a high-quality paint job at the end of the process. Failing to do so will not only result in poor results, but will also increase the likelihood of requiring another paint job in the future. To simply put it: the paint job is practically

useless and a waste of time without performing that initial preparation stage.

Such is the case with the MRI distortion correction. The data must be properly prepared in order to achieve high-quality results. The extraneous data features must be removed prior to performing calculations on that data. However, the consequences of failing to do so in this case are much greater than in the case of the vehicle paint job scenario. The incorrect application–and future use–of the distortion correction can have the potential to kill the patient. In the car scenario, the car can simply receive a coat of paint many times over, and the end result may be close to what is expected. With the distortion correction, the results will **never** be close to any expectation, no matter how many times the correction is performed. One simple saying sums this concept up extremely well:

Garbage in, garbage out.

The importance of this first step cannot be stressed enough. This concept will only become more apparent once an analysis of the individual procedures is performed.

3.2 Application

There are several steps that are performed in this preprocessing stage in the distortion correction. Each function will be described in the utmost detail. The following is a list of the procedures performed here:

1. Load the DICOM images
2. Determine useful slices

3. Perform edge detection
4. Perform plug correction
5. Verify edges
6. Remove bubbles
7. Verify corners
8. Repair holes in edges
9. Calculate the phantom's rotational tilt in the scanner
10. Calculate the phantom's position, in reference to the center of the scanner

With each step in this process, the data undergoes a battery of tests to determine if there are any extraneous features that need to be addressed. The end result is a set of images that are suitable for use in the subsequent correction steps. The absence of the extraneous features will ensure the following:

1. Accuracy in calculations (and, ultimately, the correction results)
2. Prevention of data errors
3. Prevention of improper algorithm behavior

Even in these processing steps, the presence of some extraneous features can break the algorithms. For example, improper edge detection techniques will complicate the functionality of every single subsequent algorithm. Additionally, failing to ensure the corners of the phantom are correct in the images will not provide accurate tilt or center offset calculations, and the operator has no way of knowing whether or not the phantom is properly aligned within the scanner.

3.3 Algorithm Analysis

Now that a basis has been laid for the purpose of preprocessing the data prior to performing the gradient nonlinearity distortion correction, an analysis of each step involved in the preprocessing step can follow.

3.3.1 Loading the Data

- **Function Name:** load_studies
- **Input:** Indexed array of directories containing DICOM images
- **Output:**
  - Indexed array containing DICOM images
  - Indexed array containing study info
- **Syntax:** load_studies
- **Purpose:** Load DICOM image data, and store study characteristics for future reference.

Explanation

The very first step in preprocessing may sound mind-numbingly obvious. However, there is much more involved than what originally meets the eye. Indeed, the first step to loading the data is to actually read in the data from a set of DICOM (Digital Imaging and Communications in Medicine) images, and store them in memory. But there are several challenges that must be overcome next.

The goal of this software is to be as user-friendly as possible. This means that it is necessary to eliminate as many opportunities for user error as possible. The only way to accomplish this goal is to make the software as automated as possible. The less user input necessary to operate the software, the less possibility the user has of making errors and adversely affecting the results of the correction.

With automation in mind, it is therefore necessary for the software to determine the proper settings to use for the correction procedures to follow. Even before determining these settings, the software must know the characteristics of the data that it is about to work with. Important characteristics that need to be considered include:

- Scan size (in pixels)
- Voxel size
- Slice thickness
- Scan orientation

Determining these pieces of data is quite trivial, thanks to the information stored within the DICOM header. Various attributes are stored in the DICOM header, describing the details of the study for future analysis. Matlab can extract many of these tags and make them available in the workspace. In the future, if it is deemed necessary that more information is needed about the scan in order to modify this software, it is trivial to retrieve that information from the DICOM header at any time.

The importance of each piece of data extracted from the DICOM header will become more obvious as they are used in later functions. The *scan size* determines the size of the images that the software will be dealing with. The *voxel size* identifies to the software the conversion from image pixels to actual physical coordinates in millimeters. The *slice thickness* is a measure of the distance between adjacent images, in millimeters.

And finally, the *image orientation* identifies the orientation of the study in three-dimensional space. Since the software will be working mainly with two-dimensional images of a three-dimensional object, it is crucial that the software knows which dimensions it is currently working in. For each study, a set of *direction cosines* is specified. This is a 6-element vector that specifies the encoding of the rows and columns in the images. The first three elements identify the dimension represented by the columns in the image (x, y, or z), and the last three identify the dimension represented by the rows in the image (x, y, or z).

Direction cosines, as stored in the DICOM header, might look like this:

```
[1  0  0  0  1   0]

[1  0  0  0  0  -1]

[0  1  0  0  0  -1]
```

The first example shows that the columns represent the x dimension, and the rows represent the y dimension. This study is therefore an *axial (transverse)* study. The second and third examples are a little more complicated. The rows in these two studies represent the negative z direction. A negative value for a direction cosine means that moving in the positive direction down the rows or columns in the image will result in moving in the negative direction across that axis. Here, since the rows represent negative z, traversing the rows in the positive direction will result in moving across the z axis in the negative direction. For the second example, since the columns represent x and the rows represent z, this study is a *coronal* study. The third example encodes y in its columns, and negative z in its rows. Therefore, this study is a *sagittal* study.

x, y, and z are defined in the *DICOM coordinate system*, specified as such in Figure 3.1.

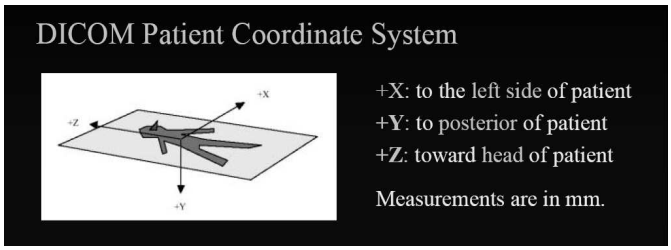


Fig. 3.1: The DICOM coordinate system.

Application

Once the images data is read from the DICOM files, and the appropriate data is extracted from the DICOM headers, the information is stored into separate arrays, for easy access later on. In Matlab, each data type is held in a separate cell array. The size of the cell array is determined by how many studies are loaded. This allows simple and easy access later on to each of the data pieces. Each set of data can be indexed within the array, and the proper data can be used accordingly.

For added convenience, the study parameters are printed out to the console, to help the operator verify that the proper studies are being loaded, and that the proper parameters are being used. This feature can be toggled on and off at the user's desire.

3.3.2 Invoking the "Process" Function

- **Script Name:** process
- **Input:**
  - Array containing DICOM images
  - Arrays containing study information (from DICOM headers)
- **Output:** Arrays of processed images
- **Syntax:** process()
- **Purpose:** Master script that invokes all preprocessing steps automatically and sequentially.

Explanation

The core of the data processing step can begin right after the DICOM data is loaded into memory. Since there are so many steps involved in the actual processing, a master function was used to promote neatness and user-friendliness. This is nothing more than a script that properly calls each function in turn, and gives each function the proper data it needs to perform its duties.

As stated earlier, the processing step is highly sequential. The output of one function will feed directly as input for the next function. At each stage, the data is either prepared for further processing, or is tested for quality issues.

During each stage, messages are printed to the console, to alert the user of the current progress of the processing. Due to the sheer number of calculations and processing occurring at each stage, watching the console and waiting for each stage

to complete is almost akin to watching water boil in a pot. The processing stage takes quite a while to complete, depending on the number of corrections deemed necessary for that data set. Additionally, the system that is running this software will ultimately decide how quickly these tasks will complete. For the record, the machine used to develop this software has the following specifications:

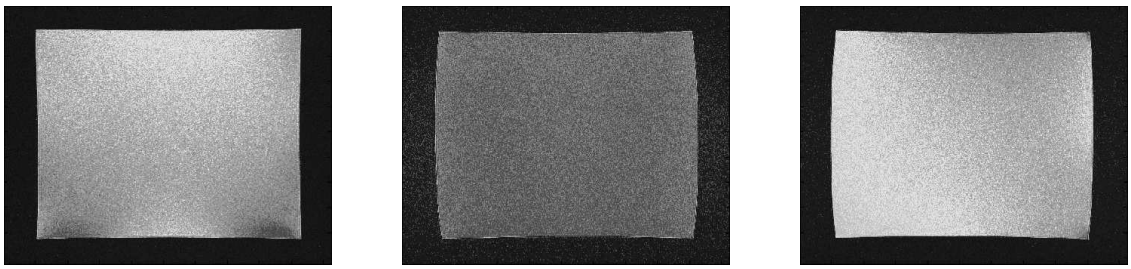
- Intel Pentium M 1.6 GHZ CPU (2 MB L2 Cache)
- 2 GB DDR2-533 MHZ RAM
- 80 GB 5400 RPM Hard Drive

When Matlab is processing the data, the CPU remains pegged at 100%. Matlab memory usage can vary anywhere between 300 MB (when only one study has been loaded, prior to processing), to over 1.5 GB (after processing has completed for 3 studies). After a series of tests, it was found that if a copy of each image set was saved after each stage of processing for debugging and analysis (with 3 studies), Matlab will eventually crash with memory errors. This is indicative of how memory-hungry these operations are. This problem can most likely be solved with more than 2 GB of RAM. Memory at the present time is quite cheap, so this solution is definitely not out of the question.

3.3.3 Determining Useful Slices

- **Function Name:** use_slices
- **Input:**
  - Array containing DICOM images
  - Flag to turn on/off automatic threshold determination
- **Output:** 2 element vector containing slice indexes
- **Syntax:** use_slices(study, auto_detect)
- **Purpose:** Determine the range of slices that actually contain the phantom, and therefore usable for all subsequent processing.

Theory



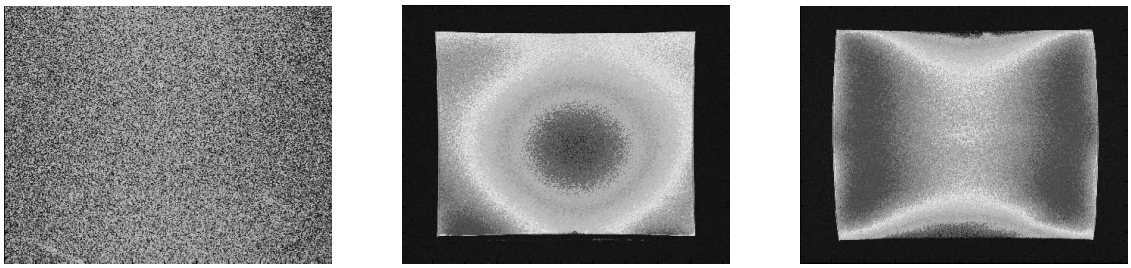
(a) Axial scan. (b) Coronal scan. (c) Sagittal scan.

Fig. 3.2: Examples of useful slices of oil-filled phantom.

The name of this function is fairly self-explanatory. This function will analyze images in a given study, and determine which images are useful for processing. The field of view and scanning range of the MRI scanner is greater than the size of the phantom.

This means that the first and last several slices do not contain the phantom at all. It is vital that these slices are not used for processing; there is no useful data in these slices. Figure 3.3(a) demonstrates this.

Additionally, it is possible that a slice may contain an image that partially captures the phantom. These are also not useful because the phantom appears to be curved, rather than straight. Figures 3.3(b) and (c) demonstrates this effect.



(a) Empty slice. (b) Partial axial slice. (c) Partial coronal slice.

Fig. 3.3: Examples of slices that do not contain useful data. (b) and (c) contain only a partial view of the phantom, while (a) does not contain the phantom at all.

Slices that contain only part of the phantom will complicate matters when edge detection is performed. Even though the human brain may be able to easily identify the edges of the phantom in these images, the edge detection algorithm will not be as successful. The varying grey levels that are present in the images are pixel intensity strengths. Darker pixels represent lower intensity values, while lighter pixels represent higher intensity values.

Compare the results from Figure 3.3(b) and 3.3(c) to those of good images, demonstrated in Figure 3.2. The difference is quite apparent. The good images are very uniform in intensity. The range of intensities within the phantom do not vary greatly,

and are much higher than the empty space surrounding the phantom. In the partial images, the range of intensities within the phantom is much greater. Some of the lower intensities are much closer to the intensities of the empty space. Or, in extreme cases, the empty space might appear to be consuming a good portion of the phantom!

In terms of intensity gradients, the good images have much higher intensity contrast between the phantom and the empty space. On the other hand, the difference in intensity between empty space and the phantom is not as great. The strength of the intensity gradient is what aids in proper and effective edge detection. Therefore, it is **absolutely vital** that all "bad" images are not considered for processing.

Application

As previously stated, the goal of this function is to determine which range of slices contain the phantom, and are therefore suitable for processing. Because the FOV specified during scanning exceeds the boundaries of the phantom, the phantom will lie within a specific set of slices in each study. It is only necessary to determine the first and last slices that are useful. All the slices in between will be good images, in that they will always contain a full image of the phantom (not considering any quality issues yet).

The technique in determining useful slices is based exactly on the observations made previously concerning bad and good slices:

- Bad slices contain only empty space.
- Bad slices contain a partial image of the phantom.
- Good slices contain an entire image of the phantom.

Putting this in terms of intensity values:

- Bad slices containing only empty space have a low maximum intensity.
- Bad slices containing a partial image of the phantom have a small number of high intensity pixels, and a large range of higher intensity values.
- Good slices contain an entire image of the phantom have a large number of high intensity pixels, and a small range of higher intensity values.

These observations and properties are key for the algorithm to determine which slices are good, and which are bad.

The following steps are taken to make these determinations:

1. Use a slice near middle slice in the study as a reference. The middle slice cannot be used because it may contain the drain plug, so use the slice that is x slices away, $x = 10$ % of the number of slices in the study.
2. Determine the intensity of the empty space around the phantom. This value will be the **cutoff intensity**.
3. Sample a window of points in the middle of that slice, to determine the approximate range of intensities within the area that the phantom exists (or where it *should* exist, in the case of a bad slice).
4. Simultaneously scan through slices, going from front to back and back to front, to find the first slices that have a similar number of bright intensities as the test slice. Partial phantom images will contain a high number of points between the cutoff intensity (the approximate max intensity of the empty space around the phantom) and the minimum intensity of the phantom.

5. Save and return the determined range (first usable slice, last usable slice).

In short, this algorithm looks for images that are similar in appearance to an image near the middle of the study, since this image will *always* contain a full, complete image of the phantom, without interference from the phantom's drain plug.

The two-dimensional vector that is returned specifies which slice the phantom starts, and which slice the phantom ends. Every other slice in the middle of that range contains a whole, entire image of the phantom. Slices outside that range either contain only empty space, or contain only a partial image of the phantom. Subsequent algorithms will only consider data within this specified range.

Providing this functionality once again reinforces the ideal of automation. Rather than having a technician scanning each image and determining useful slices by hand, this algorithm does this automatically. Additionally, the definition of a "useful slice" can be a topic subject to controversy and debate, depending on who is reviewing the images. By automating this feature, the data that is used is ensured to be consistent each time, thereby maintaining consistency in the end results.

3.3.4 Performing Edge Detection

- **Function Name:** detect_edges
- **Input:**
  - Array containing DICOM images
  - Vector containing range of useful slices
  - Flag to turn on/off automatic detection of threshold
- **Output:** Array of edge images
- **Syntax:** detect_edges(volume, range, auto_detect)
- **Purpose:** Detect the physical edges of the phantom, and create edge images, for use in the actual gradient nonlinearity distortion correction.

Explanation

The edge detection step of the processing is the most important step of all. The edge images produced here will provide the software a two-dimensional representation of the phantom's edges. This data will ultimately be used in the distortion estimation and correction steps.

The specification of the range of useful slices will ensure that the edge detector will not consider bad images. The goal here is quite obvious: produce the best quality edge images possible. Matlab supports six different edge detection methods. After a great deal of testing, it was determined that the **Canny edge detector** yields the highest quality edge images. For a full account of this research, please refer to Appendix D.

However, there is much more to this process than just running an edge detector. The following questions must be answered:

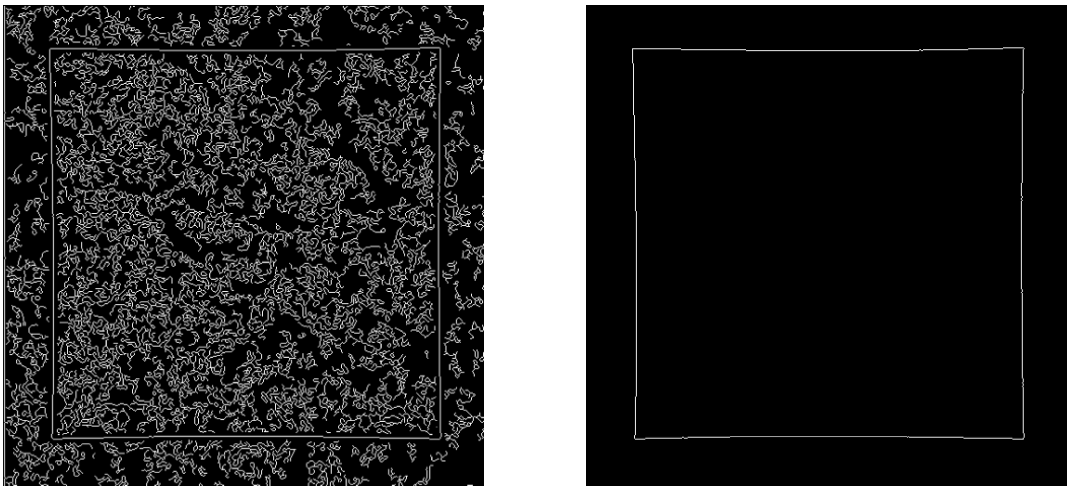
1. What does a suitable edge image look like?
2. What strength is the proper threshold strength for the edge detector?
3. Do all slices contain a full set of edges?

The best way to answer these questions is, of course, to perform these necessary steps by hand. However, this is not practical. Testing varying edge detection threshold values can take quite a long time to perform manually. Additionally, a proper threshold for one slice may not work for all slices in that study. Therefore, it is best for an algorithm to systematically determine the proper settings for edge detection, and to configure the edge detector with global settings.

Yes, it is possible to let the built-in Matlab edge detector to determine an "optimal" threshold for the images. But the output images that the Matlab edge detector considers suitable differ greatly than the output images suitable for this project. Figure 3.4(a) is an example image that Matlab produces, when allowing Matlab to automatically determine a threshold value.

Keep in mind that the Canny edge detector will find **every** intensity gradient in the image. Additionally, the Canny edge detector performs edge tracking, resulting in the curly S-shaped lines covering the entire image. The edges of the phantom are quite apparent in the image, definitely promising for this application. But it is clear that the rest of the detected points need to be discarded. A threshold that is

suitable for the purposes of this project will yield images similar to that shown in Figure 3.4(b).



(a) Threshold determined by Matlab. (b) Threshold determined by algorithm.

Fig. 3.4: Differences in threshold strengths for Canny edge detection.

The only features left in the images are the physical edges of the phantom. The data values in this image are very simple: 1, if the pixel refers to an edge, 0 otherwise. This simplifies matters greatly in other processing steps. The goal of proper edge detection becomes quite apparent.

Application

The purpose of this algorithm is to use the built-in Matlab Canny edge detector, but use it in such a way that the images produced are suitable for the requirements of the project. This means that the threshold for the edge detector needs to be detected based on the study provided.

Each time the edge detector is run, the following steps are performed:

1. If auto_detect is set to 0, a predefined strong threshold will be used for edge detection (from testing, this was determined to be 75% of the maximum strength allowable: [0.4, 0.75] for Canny).

2. If auto_detect is set to 1 (default setting), the threshold will be detected based on the properties of the current study using the function set_thresh():

   (a) Determine the strength of the intensity gradient in a good image (i.e., the difference in intensity between empty space and the phantom). This will help determine how strong the edges will be, and therefore which strength threshold to use. These properties will determine which settings to use in order to determine the optimal threshold in the next steps.

   (b) Check the number of edge points that exist in each edge image.

   (c) Increase the threshold systematically with each iteration, based on the strength of the intensity gradient determined above. Stronger thresholds will remove more noise in the image; however, using too strong of a threshold will result in the removal of desirable data! The optimal threshold strikes a balance between these two.

   (d) Stop increasing the threshold once the number of edge points no longer varies more than a certain percentage (1% for x consecutive slices, with x being determined by the strength of the intensity gradient above.

3. Use the Canny edge detector with the newly detected threshold.

4. Convert the edge images to integer values (numerical 0 and 1), rather than logical values (boolean 0 and 1), to allow for calculations later on. Store all images into an array.

The result of this operation is a set of suitable edge images. Depending on the quality of the scan, and the amount of noise present in the study as a result, there is a slim chance that some of the edge images may not be perfect. Some of the images may have missing edge points, due to noise confusing the edge detector. In these areas, the edge strength is lower than in other areas without noise. This makes the determination of the actual edge quite difficult for the algorithm, and the detector removes vital data as a result.

Also keep in mind that the edge detector does ***not*** remove bubbles or other other similar features. These features also possess strong intensity gradients, and will affect the appearance of the edges.

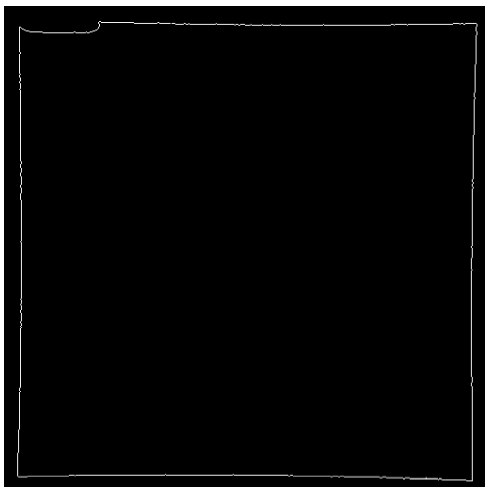


Fig. 3.5: Edge image of phantom, with large air bubble

It is because of these possible problems that the subsequent data processing func-

tions are necessary. Merely ignoring all of these bad features, and allowing the algorithms to use them as valid points for correction, is just ignorant practice. This directly violates the goals laid out at the beginning of the chapter. The presence of such features as air bubbles does not accurately represent the true shape of the phantom. To the algorithms that use this data, failing to remove those features indicates to them that this is what the phantom actually looks like. The correction results will likewise reflect these problems, and will not have any semblance of accuracy.

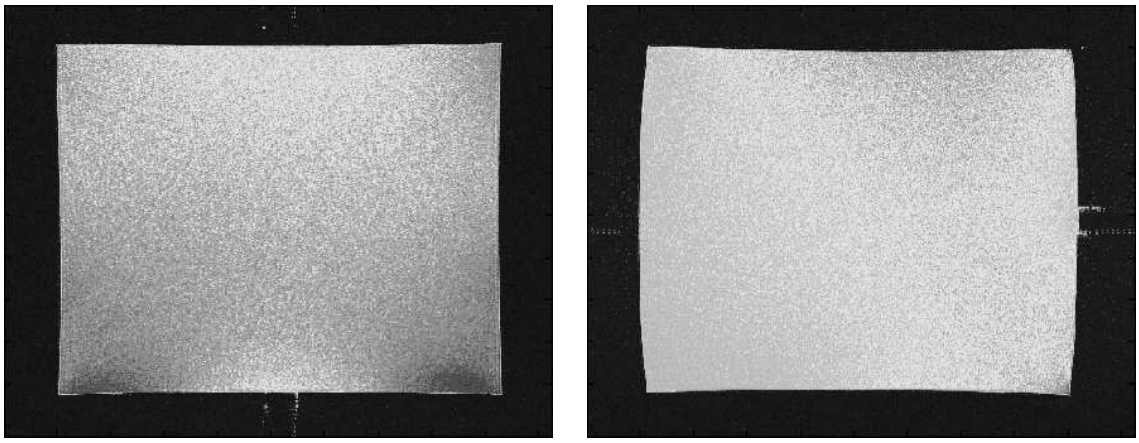
3.3.5 Removing Drain Plug

- **Function Name:** remove_plug
- **Input:** one edge image
- **Output:** one edge image, with plug removed
- **Syntax:** remove_plug(slice_edges)
- **Purpose:** Remove the image of the phantom's drain plug from all edge images.

Theory

This function does exactly what its name states: removes the image of the drain plug from all edge images. The space that the drain plug occupies, the threads of the plug itself, and the contours of the phantom edge in that region show up in the scans. Figure 3.6 demonstrates the effects of the plug on axial and sagittal slices.

These images make it apparent that there is a special technique used to process the signal to produce MRI images. Physically, the drain plug is located on the bottom of the phantom. In an axial image, the plug will appear on the bottom of the image; in a sagittal image, the plug will appear on the right edge. But notice how in both images, there is a *second* image of the plug on the opposite side of the image.This is no optical illusion! Additionally, this is not a form of image "ghosting". This phenomenon is a result of the method employed by the MRI scanner itself to sample the signal.



(a) Axial slice with drain plug (bottom). (b) Sagittal slice with drain plug (right).

Fig. 3.6: Typical slices with images of the phantom's drain plug.

Sampling the Signal: Circular Convolution

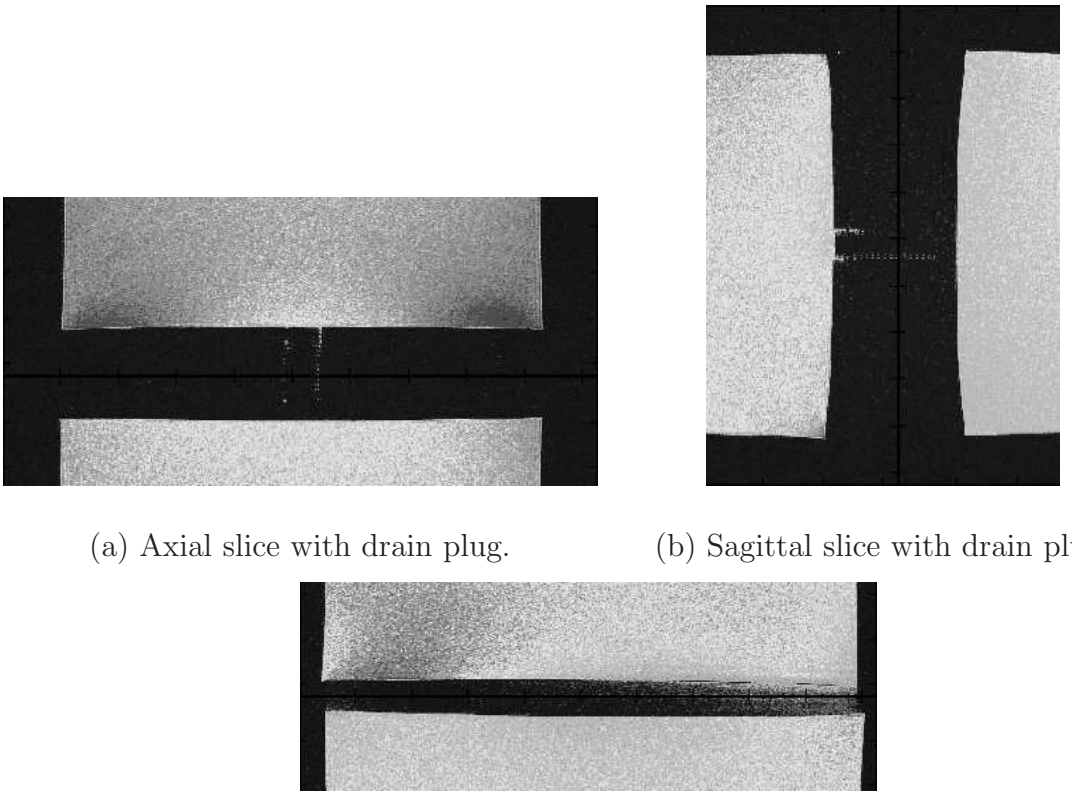
Circular convolution represents a particularly special case of the convolution method. There exist certain types of boundary conditions for convolution, all depending on the particular application of the discrete Fourier transform. A circular boundary is one possible type of boundary condition possible for convolution. As its name implies, circular convolution—also known as *cyclic convolution*—is a form of convolution that possesses a circular boundary condition.

During signal processing, signal sampling is subject to sharp discontinuities at the edges, thereby introducing errors into the sampled data. When reconstructing the sampled signal, data loss may occur to some degree in order to properly handle the effects caused by the sharp discontinuities [45, 46]. There exist many methods to handle these effects, using many well-known windowing techniques. By performing

circular convolution, the sharp discontinuities are handled in a unique manner: overlapping the initial input segments, thereby creating a "wrap-around" effect (hence *"circular"* convolution).

Performing a circular convolution involves using a discrete Fourier transform. Most computer algorithms implement the discrete Fourier transform by performing fast Fourier transforms, to promote computational efficiency. One of the side effects of using the discrete Fourier transform in this fashion is the data at the edges of the sampled signal have the same duration. This is contrary to the effect caused by performing linear convolution, where the data at the edges of the signal represent the start-up transient (or, latency) of the filter. Because of this effect using the discrete Fourier transform, the data appears to be periodic or cyclic. This is because the start-up transient uses data at both the beginning and end of the block. Circularly filtering the signal using this method and repeating it indefinitely effectively eliminates the sharp discontinuities at the edges of the signal, thereby creating seamless transitions between cycles.

The effects of circular convolution are not immediately apparent when looking at most of the images, since the images feature empty space at all four boundaries. The range of intensities of the empty space is not large, therefore the visual effects of the circular convolution are minimal. The naked eye is hard-pressed to notice the image warp-around. But in those images that contain features that are in close proximity to the boundaries of the image, the circular convolution continues that feature on the opposite side of the image. Figure 3.7 best illustrates the effect of circular convolution on the images demonstrated in Figure 3.6.



(a) Axial slice with drain plug.

(b) Sagittal slice with drain plug.

(c) Slice with oil leakage at bottom of image.

Fig. 3.7: Result of circular convolution on various slices.

This affect is even more apparent in the slice shown in Figure 3.7. In this image, the oil leaked out of the phantom, producing a signal response both inside and outside the phantom. Because of the location of the oil in this area, it appears as if the phantom is leaking oil out of this corner.

Notice how in all three images in Figure 3.7, the features that are at the edges are wrapped around, and continue seamlessly at the opposite sides of the image.

Because of this technique used, it is necessary to remove the plug at both ends of the image: on the top and bottom for axial images, and on the left and right for

sagittal images. No plug removal is necessary for coronal images, since the plug does not appear on the edges of the phantom. The plug appears in the center of the very first slices in the study, but these are removed by the use_slices() algorithm since these images do not contain a full view of the phantom.

After edge detection, a slice with the plug present will be far from perfect. The plug creates a discontinuity in the edge it is attached to, and the threads are detected as edges of the phantom, as shown in Figure 3.8.

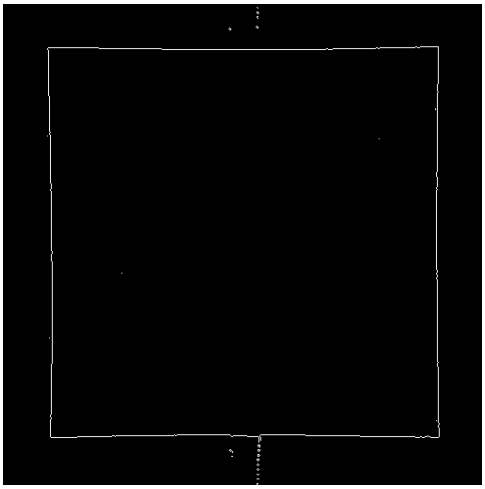


Fig. 3.8: Detected edges on a slice with the drain plug. The plug appears at the center of the top and bottom edges.

If the threads possess strong enough intensity gradients, as is the case in the image from Figure 3.8, the threads may appear at both sides of the edge image. Indeed, the edge detector removes the majority of the threads. But the edge in that location is not straight. The dips and curls that the edge contains in that region is not desirable. Simply removing the threads will not fully solve the problem. The edge is still not straight, as is the case in the middle region on the other three edges. Therefore, more drastic measures are necessary to correct this problem.

Application

Since erasing the threads will not completely solve the plug problem, a more sophisticated method is required. This method must perform the following operations:

- Remove the threads of the plug at both ends of the image.
- Remove the effect of the hole for the drain plug.
- Repair the edge in that region.

The execution of these steps is quite trivial. To simplify matters even more, this operation is performed on **all** edge images in the study, rather than just those afflicted by the plug. This also ensures that the plug is removed for all affected images, no matter how the phantom is positioned within the scanner.

To remove the plug from the edge images:

1. Determine where the phantom lies within the edge image. This ensures that the middle of the phantom edge is found, even if the phantom is placed off-center within the scanner.
2. Locate the middle of the bottom horizontal edge of the phantom in the image.
3. Clear out all the data in the middle of the image vertically. The width of the deletion window is equal to 25% of the horizontal size of the scan.
4. The top and bottom horizontal edges now have a large hole. Determine where the edges lie on either side of the hole.
5. Draw a straight line connecting both edges.

The result is an edge image that shows no sign of the plug ever existing in the first place, see Figure 3.9.

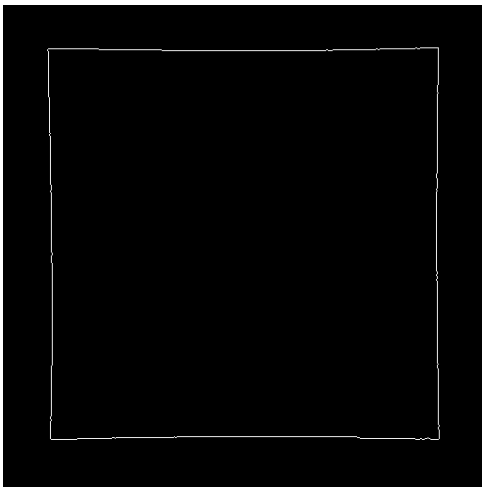


Fig. 3.9: Edge image after plug correction.

The assumption is that the edges of the phantom will most resemble a straight line in the middle regions. Because of this, it is feasible to simply draw a straight line connecting the edges within the deleted region. Even though this procedure is performed on all edge images in a study, the results are still nominal in all cases.

The edge images for 3 studies (axial, coronal, and sagittal) were corrected, and each corrected edge image was carefully analyzed. In all cases, the following observations were made:

- In the best case, the plug is removed, and an otherwise problematic image is corrected.
- In the worst case, the edge is shifted by one pixel in the erased region.
- For the majority of the images, there is no change before and after correction.

Because of the nature of the edge images, and also that of the drain plug removal method, it is not necessary to specifically identify which images to correct, and perform the correction only on those images. This implementation is much simpler, and can be performed much more quickly.

It becomes obvious that failure to remove the plug provides the subsequent algorithms with an inaccurate representation of the actual phantom, and–as a result–can even break some algorithms. Therefore, the importance of this step in preparing data properly for future processing becomes apparent.

3.3.6 Verifying Edges

- **Function Name:** verify_edges
- **Input:**
  - Array containing edge images
  - Array containing DICOM images
  - Vector of slice index ranges
  - Flag to turn on/off automatic detection of threshold
  - Flag to turn on/off report only/correct bad images
- **Output:** set of edge images, corrected
- **Syntax:** verify_edges(edges,study,range,correct)
- **Purpose:** Verify edge images contain four intact edges. Additionally, check for large noisy features in the edge images. In all cases, either repair or report any erroneous images.

Theory

This function performs a data integrity check on all edge images. The purpose here is to search for and handle severely damaged edge images. There are several scenarios that will set off the error flag in this verification step:

- An edge image contains a large discontinuity (many pixels in size).
- A large noisy area is present in the edge image.

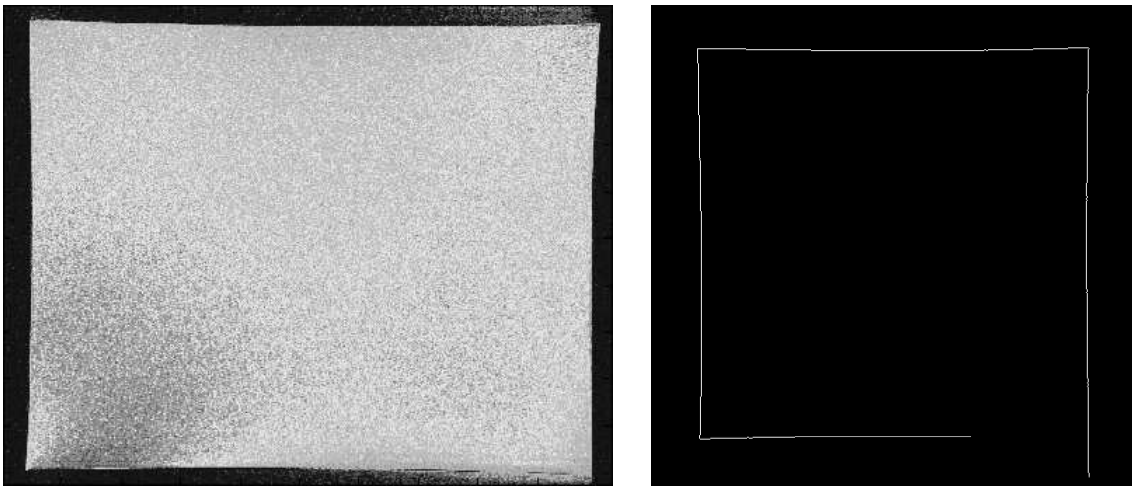
- An extremely large air bubble is present in the edge image.

The first case can occur during edge detection itself. If the test slice that is used for determining the edge detection threshold is very noisy, the edge detection threshold can be improperly configured. To handle the amount of noise in that image, the threshold would be accordingly set higher than in an image free of noise. Therefore, when the edge detector is run on all the other edge images, the strength of the threshold might remove useful regions of the phantom inadvertently. This results in large discontinuities in the edges.

Additionally, if there is enough noise at the edge of the image, the strength of that edge will become much weaker. The edge detector will therefore exclude that region of the image as being an edge, even though it actually does refer to the physical edge of the phantom. Such was the case in the study performed on the phantom that was leaking oil. Several slices displayed the effects of the oil leakage. The detected edges of this slice clearly demonstrate the serious problem that arises here. Refer to Figure 3.10 for these results.

The entire lower right quadrant of the edge image is nonexistent! The problems that this will create are quite severe. Correction of this problem is **absolutely necessary!**

In the case of the large air bubble, the edge may not have any semblance of the physical phantom edges. Recall that the signal response represented in the images is from the oil in the phantom. Oil provides a very strong signal response. Air, conversely, provides a very weak response. The consequences of having air inside the phantom are great. Since the oil is supposed to be providing the full representation of



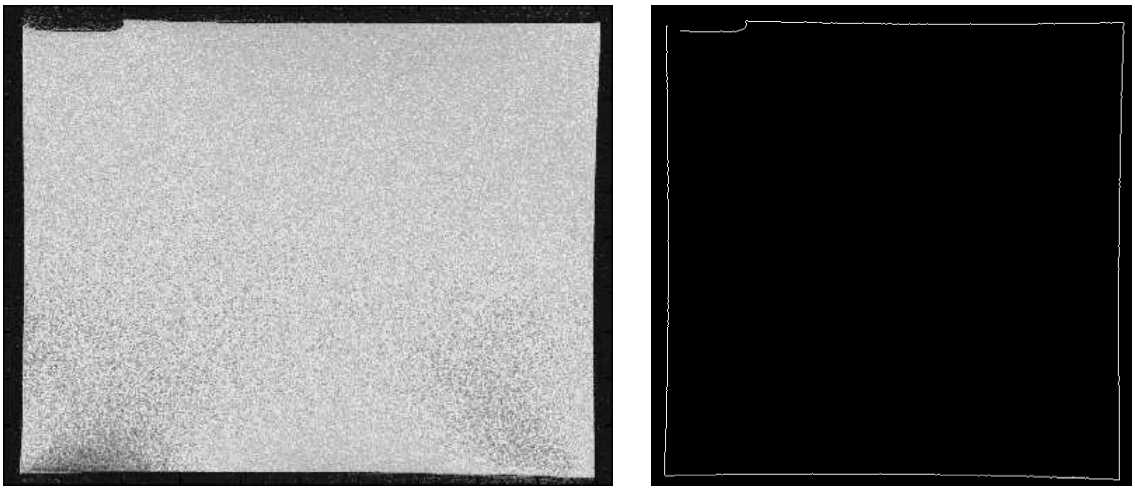
(a) Slice with oil leakage. (b) Result of edge detection.

Fig. 3.10: The adverse effects of phantom oil leakage.

the phantom, any air will disturb that representation, and corrupt the edge images. The large air bubble in Figure 3.11 demonstrates this point quite well.

Looking closely enough at the image will reveal a discontinuity at the top left corner of the image. This is not acceptable by any means. That hole will cause problems in nearly every single function that will use these edge images.

The severity of these data integrity problems cannot be stressed enough. If these errors are left in the data sets, one might believe that the algorithms are misbehaving due to design flaws. This was the case during the original testing and implementation stages. The functions were implemented correctly; it was the poor quality data that the functions were using that were breaking them.



(a) Slice with large air bubble. (b) Result of edge detection.

Fig. 3.11: Slices containing a large air bubble.

Application

Verifying edge integrity and quality is not as difficult as it may appear. Simple observations can be made about what constitutes a good edge image, and the bad edge images demonstrated above. In the case of the leaky slice, there is a large number of edge points missing in one quadrant of the image. Conversely, the image containing the bubble will contain extra edge points in that particular quadrant.

A good image will not have any of these features. The good image will have a nearly equal number of edge points in each of its four quadrants. No extra points will be present, and there will not be regions missing points.

These simple observations are the deciding criteria for the edge verification step:

1. Split the phantom edges into four equal quadrants.

2. Count the number of edge points present in each quadrant. A good edge image will not vary greatly in the number of edge points in each quadrant.

3. If a quadrant is found that varies greatly in the number of edge points (either less or more), flag the slice as being problematic.

4. Take the appropriate action based on the input flag:

   (a) If report-only is specified, do not copy flagged slices into the final returned array of slice images. Therefore, the returned array will only contain good images, and will be smaller than or equal in size to the original input array.

   (b) If correct is specified, repair all bad slices with fix_corner_bubble():

      i. Determine which quadrant is problematic (i.e., which one has a lot more/lot less edge points than the others).

      ii. Send the image to be fixed with fix_corner_bubble(), and fix that particular quadrant. For more information concerning this process, refer to Section 3.3.7.

      iii. Copy the repaired image into the final return array. The returned array will always be the same size as the input array.

A report-only method will remove all bad slices, and will only allow good slices to be used for processing later on. This method is much quicker, but is not a sure-fire method of removing all bad slices. Some small defects may still slip under the tests, and remain uncorrected in subsequent steps.

The correction method is much safer, but requires extra time to execute. This method ensures that **all** bad slices have been handled, and are suitable for use later

on. Each corrected slice will have the problematic quadrant replaced by an opposite quadrant, as described in fix_corner_bubbles().

To alert the user of the results of this data verification process, the function will output its findings. If no bad slices were found, it will report "No bad slices!" If bad slices were found, the function will report how many bad slices were found and repaired. It can be left up to the discretion of the operator whether or not to use this study for correction based on the number of slices that were found to be problematic. While a bad quality can still be saved at this stage, it may not be the smartest move to use a large number of corrected images. A large number of bad slices is indicative of the quality of the scan overall. The edges may now be free of serious problems, but may not be as crisp and straight as a good quality scan. This can result in accuracy issues in the gradient nonlinearity distortion correction.

3.3.7 Removing Bubbles

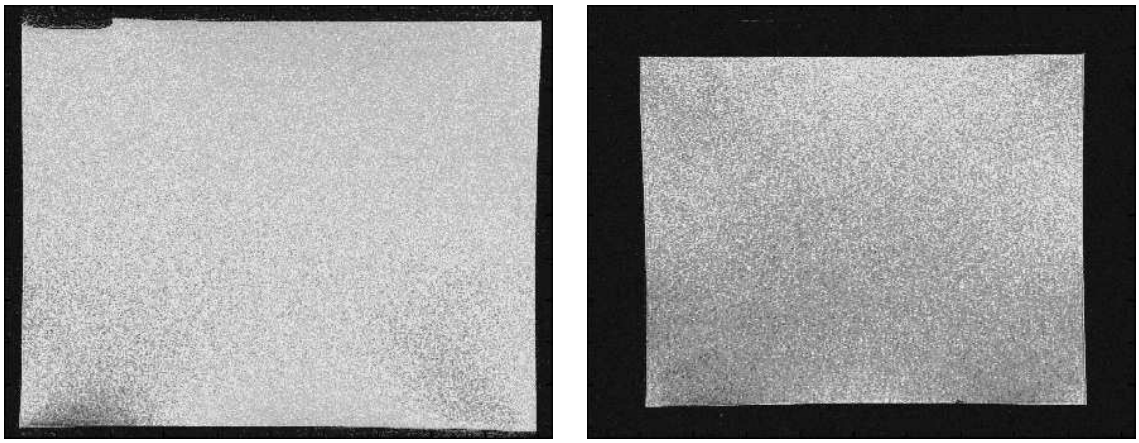
- **Function Name:** fix_bubbles
- **Input:**
  - One edge image
  - Edge to scan (1,2,3,4)
  - Point range to check
- **Output:** Corrected edge image
- **Syntax:** fix_bubbles(slice_edges,edge_number, separation)
- **Purpose:** Remove all bubbles from an edge image, on all four corners.

Theory

Air bubbles are one of the most annoying and most devastating problems to deal with in the phantom edge images. Bubbles corrupt edges and corners, and do not provide an accurate representation of the phantom. Correction methods will not yield accurate results if bubbles are present in the scan. In some cases, some functions will not yield **any** results with the presence of air bubbles. The air bubbles can cause many problems, including discontinuous edges, rounded imperfections, and inaccurate corner representation.

Recall the reason that bubbles cause such havoc with the edge images. The images that are used here represent the signal response of the oil within the phantom. The fundamental assumption of this procedure is that the phantom is filled *only* with

oil; therefore, the signal response is indicative of the interior volume of the phantom. Since it is impossible to fill the phantom 100% with oil, and remove **all** the air from inside, air bubbles will be present in some form. If the phantom ever leaks oil, more air will be present in the phantom. The air will displace the oil away from the edges of the phantom, thereby affecting the signal response observed in the scanner, and therefore the appearance of the images. Air does not provide a substantial signal response, as compared to oil. Therefore, the areas that the air resides in will appear to be empty, as if the phantom was physically shaped in that manner. Figure 3.12 demonstrates the appearance of air bubbles in various slices.



(a) Large corner air bubble. (b) Small air bubble on bottom edge.

Fig. 3.12: Small and large air bubbles, as they appear in a slice.

The issues caused by inaccurate edges have been mentioned numerous times already, so the importance of correcting those problems should be quite apparent now. The problem with inaccurate corner representation is a serious problem indeed as well. Knowing the locations of the corners of the phantom is crucial for:

- Edge tracking
- Tilt calculations
- Center offset calculations

The edge tracking technique will become important when it comes time to input the data for the gradient nonlinearity distortion correction.

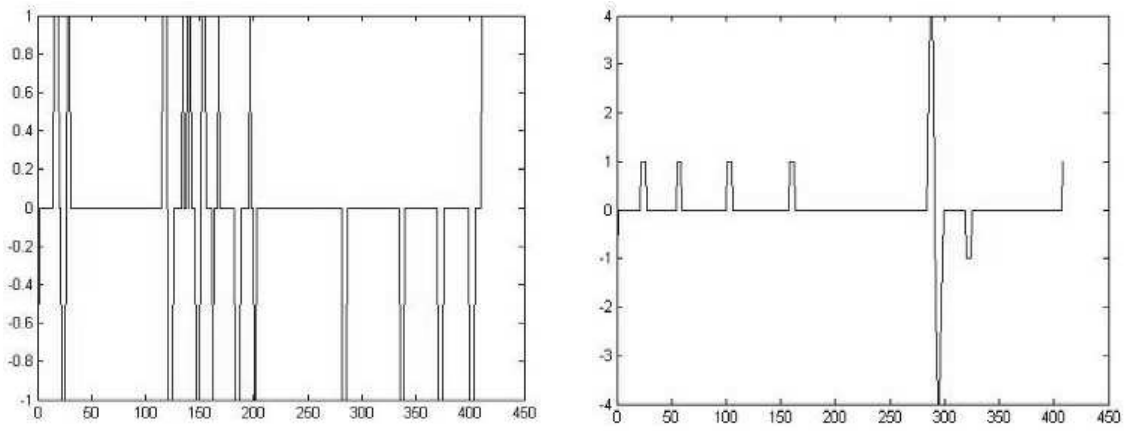
Locating bubbles is moderately difficult. The way to approach the problem is to make careful observations about the physical characteristics of a bubble:

- A bubble has curvature.
- A bubble will most likely have a flat spot at its peak.
- Bubbles cause deviations in an edge or corner.

As simple as these observations may sound, they are actually quite useful in identifying the bubbles. The fact that bubbles are usually not narrow and flat aids in their identification. When the term "flat" is used, it is to mean that the bubbles do not appear so short and abrupt that they mimic the edge they lie in. A quick analysis of each of the two images above will make it apparent of the one unique property of bubbles: their shape constitutes a drastic shift in the appearance of the edge. Edges in the images will not shift in any direction more than one pixel over small distances. Bubbles, on the other hand, will shift dramatically over a fairly small distance.

If the points on a given edge are analyzed, the difference between the shape of an edge containing a bubble and one that does not is more apparent. As previously stated, an edge without bubbles will not shift more than one pixel over a small

distance. Figure 3.13(a) shows the pixel shifts in a normal edge that does not contain any air bubbles. Following this edge shows that this edge will deviate at most 1 pixel in either perpendicular direction over a range of 5 pixels. Conversely, an edge with a bubble present will shift more than one pixel over a small distance. Figure 3.13(b) demonstrates an edge with a small air bubble.



(a) Edge without air bubbles.

(b) Edge with small air bubble.

Fig. 3.13: Pixel shifts across a 5-point span in 2 edges. Note the peaks of (a) are one pixel, while the peaks in (b) are four pixels.

The edge with the bubble experiences up to a 4-pixel shift across a 5-pixel distance. Immediately, this indicates a problem in the edge. The appearance of the edge on either side of the bubble is quite normal, only exhibiting at most a one pixel shift in any direction. Then suddenly, the bubble comes into view. Notice how the edge quickly undergoes a large positive pixel shift, then flattens, then undergoes a large negative pixel shift. This is indicative of the curvature of the bubble. The large pixel shift represents the increase in slope of the first edge of the bubble. When the slope

increases at the bottom portion of the bubble, the result is the positive pixel shift. As the curvature of the bubble begins to flatten out, the edge shifts back towards its original position before the bubble. The plot shows this as the return to zero from the positive spike. The top of the bubble is the zero region between the two spikes. As the second edge of the bubble is approached, the slope of the bubble becomes increasingly negative. The slope starts to increase again as the bubble flattens out. Finally, the edge is back at zero again after the bubble is passed.

This set of observations has identified three more important properties of bubbles:

- The first edge of a bubble is characterized by a large positive pixel shift.
- The top of the bubble is a flat region, that becomes parallel to the edge at some point.
- The second edge of a bubble is characterized by a large negative pixel shift.

Analyzing the graph in Figure 3.13(b) makes bubble location trivial. The determining criteria is the presence of the two opposing spikes in pixel shift. A bubble **always** have both pixel shifts because of its curvature. The location of these spikes determines the type of bubble that has been found: edge or corner. If the spikes are in the middle of the edge, the bubble is an *edge bubble*. If the first points of the edge are a part of the positive pixel shifts, then the bubble is considered to be a *corner bubble*, located at the beginning of the edge. Similarly, if the last points of the edge are a part of the negative pixel shifts, then the bubble is considered to be a corner bubble, located at the end of the edge.

These observations are verified by the plot of the large corner bubble, shown in

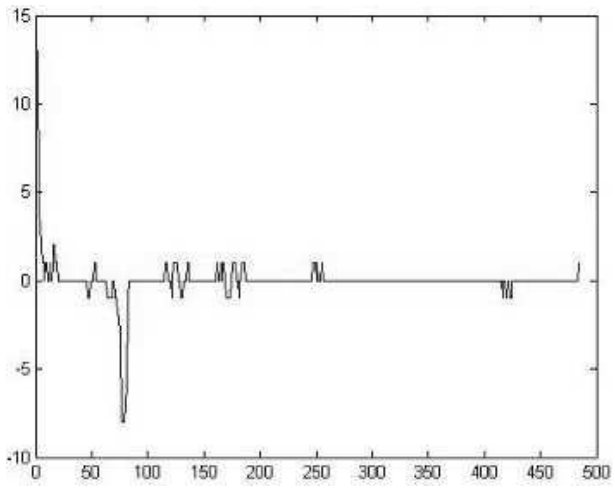


Fig. 3.14: Pixel shifts across a 5-point span in an edge with a large bubble. The positive peak is at 13 pixels, and the negative peak is at 8 pixels.

Figure 3.14. The pixel shift caused by the bubble in this slice is very dramatic. The edge undergoes a positive pixel shift of 13 pixels in the positive direction, and 8 pixels in the negative direction! This is also immediately apparent. The smaller jagged lines are the acceptable one-pixel shifts in the edge. These shifts pale in comparison to the shifts caused by the bubble.

Notice as well in this plot the characteristics of the bubble. There are multiple large pixel shifts present. The start of the bubble has a very large pixel shift (13 pixels). Moving to the right sees the pixel shifts becoming smaller, but then the pixels start shifting in the positive direction again. This is still part of the first edge of the bubble. The end of the bubble is designated as the end of the negative pixel shift, after the negative pixel spike. Determining this numerically is not difficult.

But what happens in the case that this image is flipped, such that the intermediate spike is now a negative spike? Observe the plot in Figure 3.15. The edge has been mirrored horizontally to produce this plot.

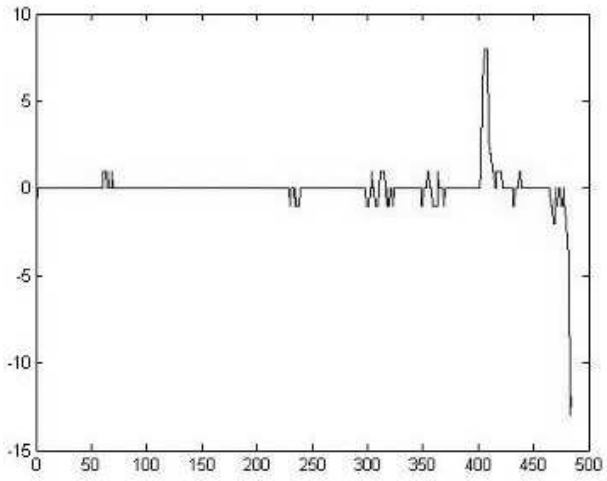


Fig. 3.15: Pixel shifts of slice with large air bubble from Figure 3.14, with the edge mirrored horizontally.

Which spike is indicative of the end of the bubble? If an algorithm simply looks for the end of the next negative spike, the algorithm will improperly identify the location of the bubble. In this case, the algorithm will choose the end of the bubble at a location roughly 10-15 pixels too short of the actual end. Therefore, it is necessary to find the **last** negative spike. In the event there are multiple bubbles in the slice, the last negative spike will be that spike occurring closest to the next positive spike. Or, in the event that this is the only bubble in the edge, the last negative spike will be just that: the last negative spike in the edge. As is the case in this image, the correct negative spike lies at the very end of the edge; identification using these criteria is not a big deal.

Application

A great deal of information can be derived from the simple edge point plots demonstrated earlier. All the necessary criteria to properly identify bubbles can be easily

determined from those plots.

There are several steps involved in generating those plots, for further analysis. These are handled by the edge_test() function:

1. Invoke the edge_test() function with the following information:

   - The current edge image
   - Edge to be scanned, identified as:
     - 1 for the top edge.
     - 2 for the bottom edge.
     - 3 for the left edge.
     - 4 for the right edge.
   - The distance to analyze for pixel shifts (default is 5 pixels)

   The output will be a vector that contains pixel shifts in the edge greater than one pixel over the specified pixel distance.

2. Send the vector containing large pixel shifts, along with the current edge image, to the locate_bubbles() function:

   (a) Locate the large pixel shifts.

   (b) Look for the paired positive and negative spikes in pixel shifts, as described earlier.

   (c) Mark the ranges of each paired pixel shift spikes, and return them in a vector. Additionally, return the type of bubble found (edge or corner).

3. Based on the type of bubble found, determine how to handle the bubble:

- For edge bubbles:
  (a) Consider only $\frac{1}{2}$ of the edge image, so that the opposite edge does not get erased.
  (b) Erase the edge image for the range of values that the bubble covers.
  (c) Send the edge image off to have the hole repaired by the repair_holes() function (Section 3.3.9).
- For corner bubbles:
  (a) Invoke the fix_corner_bubbles() function, with the following information:
    – The current edge image.
    – The edge that contains the bubble.
    – The pixel distance to check for the bubble (default is 5 pixels)
    – The location of the corner bubble (1 for beginning of the edge, any value greater than 1 for the end of the edge).
  (b) Based on the corner that is affected, grab data from the opposite bottom quadrant. That is:
    – The top left quadrant will use the bottom left quadrant.
    – The top right quadrant will use the bottom right quadrant.
    – The bottom left quadrant will use the bottom right quadrant.
    – The bottom right quadrant will use the bottom left quadrant.
  (c) Erase the quadrant containing the corner bubble.
  (d) Mirror the new data from the opposite quadrant.
  (e) Align the new data properly to match up with the old quadrant.

(f) Graft the new data into place.

The result is an edge image without bubbles, see Figure 3.16. Removing the edge bubbles are straight-forward: erase the bubble, and repair the hole created with the repair_holes() function. The specifics of the repair_holes() function is described in great detail in Section 3.3.9.

(a) Before correction. (b) After correction.

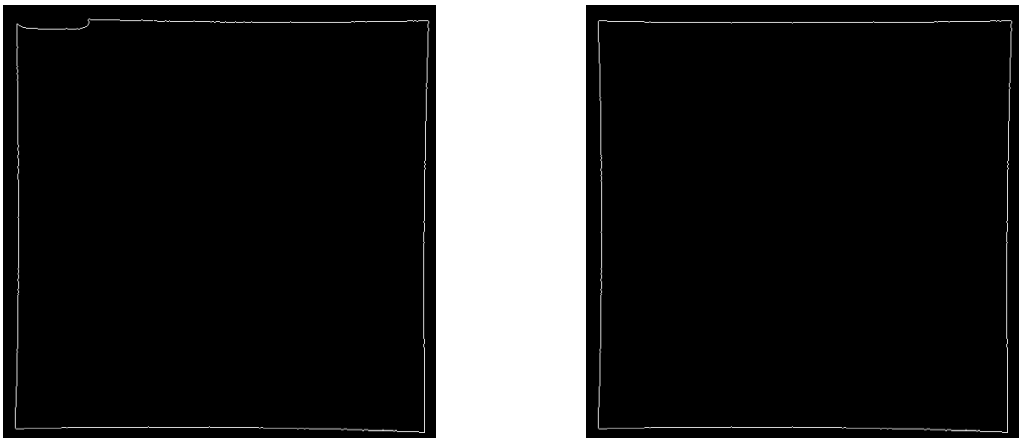
Fig. 3.16: Demonstration of bubble correction on small edge bubble.

The removal of edge bubbles is very trivial compared to removing corner bubbles. The retrieval of the opposite quadrant data and aligning it properly with the other side of the image is not an easy task whatsoever. When retrieving the data from the opposite quadrant, a smaller "sub-image" is created. To prevent problems with placing the new data overlapping past the boundaries of the entire edge image, the new data image must be cropped. It is necessary to then determine where the edges are at the boundaries of the image, and then determine where the edges are at the boundaries of the erased portion in the old quadrant. Knowing the locations of the edges in both images is required in order to line up the images perfectly. Finally, the proper range of rows and columns must be properly identified before grafting in the new data. If the erased portion is not ***exactly*** the same dimensions as the new data image, the new data will not be grafted into place properly, and the correction

cannot be performed.

Being able to mirror edges and achieve valid results follows the assumption that the distortion is symmetric. This means that the appearance of the phantom at the top will nearly mirror that of the phantom at the bottom, and vice versa. The same follows for the right and left sides of the phantom. Additionally, this also follows that the phantom has been correctly aligned and centered in the scanner. This test is performed later on, after the data is deemed usable and safe to process. These steps will be described in a later section.

Once these difficult challenges are overcome, the results are quite satisfying, as shown in Figure 3.17. Depending on the severity of the bubbles in the slice, the improvement made in affected slices can be absolutely striking. What used to be a nearly unusable image due to a large corner bubble is now perfectly suitable. The application of fix_bubbles() saved that slice, and allowed it to be usable again. Even though the difference in removing the small edge bubble is not as great, the usefulness of the function is not lessened by any means here. The edge in the corrected image is a much more accurate representation of the actual physical edge of the phantom. In any case, any corrected image is going to be much better than the uncorrected "bubbley" image, even if the image is not a 100% accurate representation of the phantom. Granted, the correction does give a *very* close approximation of the phantom in that region. And there is no other way to get a better approximation, unless the phantom is modified to remove the air bubbles completely (which is highly unlikely and nearly impossible).



(a) Before correction. (b) After correction.

Fig. 3.17: Demonstration of bubble correction on large corner bubble.

3.3.8 Verifying Corners

- **Function Name:** verify_corners
- **Input:**
  - One edge image
  - Corner to verify
- **Output:** Corrected edge image
- **Syntax:** verify_corners(slice_edges, corner_number)
- **Purpose:** Verify that the corners of the phantom in the edge images are in good condition. If the corners are corrupted by noise, replace them.

Edge images are easily susceptible to corruption by noise. The strength of the edge detector removes the majority of noise from the images. At this stage in the processing, many of the specific forms of noise have already been removed:

- Drain plug.
- Edge bubbles.
- Most forms of random noise in/around the image.

However, this does mean that the image is 100% free of noise. Yes indeed, the air bubbles and drain plug account for a good amount of noise in edge images. These are able to be detected and corrected with little issue. In studies that are affected by these features, **most**—if not, all—of the images no longer exhibit any signs of the noise. However, in extreme cases, there may be forms of noise that do not fall under any previously defined category.

Most forms of **"extreme noise"**—noise that corrupts the edge images and requires special correction methods—can be handled with the functions that have been introduced already. Disturbances in the edges can be handled in the same manner as edge bubbles. Any noise that is lucky enough to lie in the path of the plug correction will be removed by default. Corner bubbles can be detected and handled effectively. But what happens if there are other noisy features that are present in the images, that are not detectable in the same manner as a bubble?

Recall that air bubbles are characterized by relatively large pixel shifts over a small distance. Any other noise that exhibits this property will also be detected

and removed by the bubble removal algorithm. So these situations are not an issue. What does become an issue is noise around the image, that fools the edge detector into including it as part of the physical edges of the phantom. This may extend the edges past the physical corner of the phantom, as is the case of the noisy corner shown in Figure 3.18.

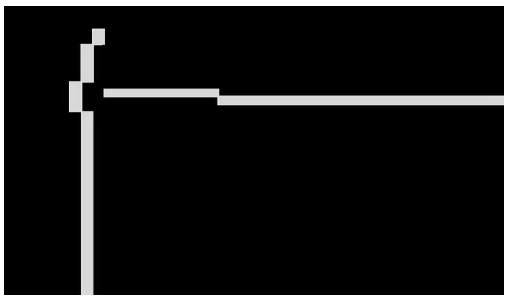


Fig. 3.18: Edge image with noisy corner.

This unique noise can be attributed to any of the following possibilities:

- A signal response from a large air bubble.
- Excess oil that was spilled on the outside of the phantom at the top of the phantom.
- The result of "wrap-around" from circular convolution.

In this particular study, the phantom was leaking oil, evident by the presence of the large corner bubble in a good number of the slices. This noise not only occurs in the slices with the corner bubble, but also occurs in slices adjacent to the air bubble. The noise that is present with the corner bubble gets removed when the corner bubble is removed. In those instances, the noise is not a problem. However, in those few

images where there is no bubble detected, the noise shown in Figure 3.18 still remains present.

A quick analysis of the edge in this area yields nothing out of the ordinary:

- There are no large pixel shifts in the edge.
- The edge is continuous.
- The detected corner is located in a probable area.

To the software, there is no problem evident here. So far, the software does not have a way of identifying that this is a problem. As a result, calculations will be performed using this erroneous data. This causes the detected corner to be at a point outside of the phantom, rather than at the correct physical location on the phantom. This greatly affects nearly all the subsequent functions that rely on corner detection:

- Edge tracking.
- Recursive hole repair.
- Tilt calculations.
- Center offset calculations.
- Final data preparation for gradient nonlinear distortion correction.

Upon closer examination, the only problem that is present is the fact that the horizontal edge in that location is discontinuous. But this is easily fixed with the repair_holes() function. But the extended corner is still present. Once again, the question still remains: what makes this corner bad? What properties can be attributed to this corrupted corner?

None of the previously defined characteristics will uniquely identify this problematic scenario. If the details are more closely examined about this corner, versus a good corner, one simple observation can be made:

- A good corner point has more than one point that lies in the same row and column, at some location in the image.
- A bad corner point only has points that lie in either the same row or column. The point is by itself in the column or row that it lies in.

Applying these observations to this particular case, we can see that using this criteria allows for positive identification of this erroneous region. The detected corner point, which is at the tip of the extended line, is only collinear with other points in the same row (vertical collinearity). However, that corner point is the only point that lies in that particular column (is not horizontally collinear). Close examination of a good edge image will reveal that all edge points have at least one other point that lies in the same column and row. Locating a point that is solitary in a particular row or particular column is a strong enough identifier that a problem exists in that region. Another example of this condition can be found in Figure 3.11(b).

This is the case for the extended vertical edge. If the image were rotated, such that the horizontal edge was extended in this same fashion, it would become apparent that the corner point in that region would be the only point in that particular column. Once again, that region can be positively identified as being problematic using this set of criteria.

Application

Based on the observations made concerning erroneous edges, and the criteria to properly identify these regions, it is now possible to perform correction on these regions. Because the problem lies in the corner of the image, the entire corner must be replaced. Both of the edges near this corner may also be influenced by the same noise that corrupted the corner itself. Therefore, a correction method similar to that used for handling corner bubbles is required.

To perform this correction, the verify_corners() function does the following:

1. Get the current corner using the get_corner() function, using the following information:
   - The current slice image.
   - The desired corner:
     - 1 for the top left corner.
     - 2 for the top right corner.
     - 3 for the bottom left corner.
     - 4 for the bottom right corner.
2. Check how many points lie in the same row and column as the detected corner point.
3. If there is only one point that lies in the same column and/or row as the corner point (which is the corner point itself), there is a problem with this particular corner.

4. Choose the opposite bottom corner for correction. That is:

    - The top left corner will use the bottom left corner.
    - The top right corner will use the bottom right corner.
    - The bottom left corner will use the bottom right corner.
    - The bottom right corner will use the bottom left corner.

5. Retrieve the corner data. The size of the sampling window for this region is 2% of the size of the image.

6. Mirror the new corner data, to align it properly with the corrupted corner.

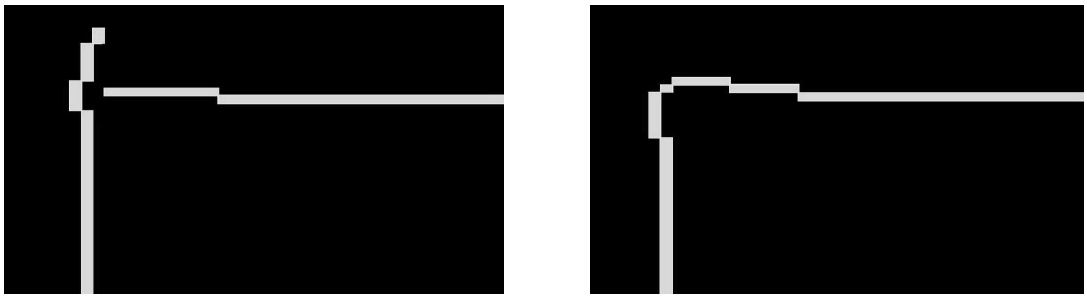
7. Crop the new corner data, to only leave the data that pertains to the phantom. This prevents the possibility of the new edge data lying outside the image boundaries when placed into its new location.

8. Erase the corrupted corner region, to remove the noise and make room for the new corner data.

9. Determine the locations of the edges at the boundaries of the new data image, as well as the locations of the edges at the boundaries of the erased region.

10. Line up the new corner data properly with the original image, and graft the data into place.

The method is very similar to the corner bubble correction. But this method only replaces the corner data, rather than an entire quadrant of data. Nevertheless, the correction results are just the same, as displayed in Figure 3.19.



(a) Before correction. (b) After correction.

Fig. 3.19: Corner correction on edge image with corrupted corner.

Now, the corners in the edge images are verified to be accurate approximations of those on the phantom. Noise in the corners is no longer an issue after the correction performed by this function. This ensures that the detected corners are accurate, and will provide correct information to the plethora of functions that rely on corner location data. Additionally, this will ensure that the edges are of high quality in this region, and will increase the accuracy of the results after the gradient nonlinearity distortion correction. The extraneous edge points removed by this function will adversely affect the results of any function that uses the data from this point on. Even though this function only ends up performing correction on a small handful of slices, and only in special cases, it does not degrade the important role this function plays in the overall quality assurance of data.

3.3.9 Repairing Holes

- **Function Name:** repair_holes, repair_holes_edge
- **Input:** One edge image
- **Output:** Corrected edge image
- **Syntax:** repair_holes(slice_edges), repair_holes_edges(slice_edges)
- **Purpose:** Repair any hole of (theoretically) any size found in the image.

The repair_holes() function is pretty straightforward, in both its purpose and its execution. This function will search for holes in the edge images, and will redraw the edge in that region to make the edge continuous.

There are two types of holes that can be present in an edge image:

- Holes caused by edge detection.
- Holes created from bubble correction.

In the case of the holes caused by edge detection, the holes are usually not very large, as shown in Figure 3.20. There is a caveat to this statement: by this point in the execution, large holes have already been addressed and corrected with the verify_edges() function. The smaller holes that may be left over in the edges will not be detected—and therefore corrected—by the verify_edges() function. These holes are usually very small in size: only one or two pixels in size. Although the holes are small in size, the problems they can create are large indeed. Here, the bite is *much* larger than the bark, figuratively speaking.

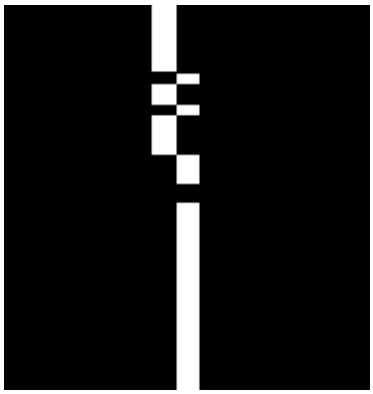


Fig. 3.20: Edge with a 1-pixel hole.

On the other hand, holes caused by bubble correction can be quite large in size, often spanning many pixels. Another possible scenario is multiple bubbles appearing on an edge, creating multiple holes. Figure 3.21 is an example of this scenario. The issues that these large holes will create are quite evident: leaving holes this large in the images creates data discontinuity, and will adversely affect the results of the gradient nonlinearity distortion correction. Due to the size of the hole, it is possible that the edge points on either side are not collinear, meaning that the edge undergoes some pixel shift somewhere within the expanse of the hole. Therefore, drawing a straight line from one edge to the next will not create a continuous edge image in every case.

Fig. 3.21: Edge image with 2 large holes, caused by bubble removal.

Because of the different natures of a hole one pixel in size, and a hole several pixels

in size, a universal method must be used that will handle both situations. Before the holes can be corrected, a few pieces of information must be determined:

- How large is the hole?
- How does the edge shift over the span of the hole?
- How to shift the edge to best approximate the edge within the hole?

One-pixel holes, or holes that occur in perfectly straight regions of the edge, are not difficult to deal with. The solution would be to simply draw a straight line from one side to the other. But if the hole is sufficiently large, it is possible that the edge might have shifted more than one pixel in one direction. Drawing a straight line to correct a hole in that region will result in yet another hole. Solving one problem creates another. In this instance, it is possible that this phenomenon will continue to occur: as one hole is corrected, another is created; as the second is hole is fixed, a third is created, and so on. Obviously, looping in such a manner is not conducive to any productivity. In any event, if that looping does stop, the quality of the hole repair will most likely be quite shoddy.

The goal of this correction is to be able to handle (theoretically) any size hole in an edge. Additionally, the correction should also be able to handle any number of holes in the images. The example image from Figure 3.21 will need special attention. So the procedure not only has to correct holes, but has to verify that its job is done before it returns a result. Therefore, recursion is required.

It is now clear that the solution to addressing holes needs to have the following characteristics:

- Fix one-pixel holes.
- Fix large holes, and properly approximate the shape of the edge, should the edge shift somewhere in the expanse of the hole.
- Fix **all** holes in the image.

Only then will the issue with holes be fully handled. Any compromise on these three ideals will eventually result in a set of edge images that still contain holes, even after "correction." The assumption with any function is that the function has completed its job *correctly* to the fullest extent by the time it has returned its share of data. The same is true here. Careful design of this correction is required to ensure that these requirements are met.

Application

Given the requirements listed above, it is clear to see that the solution will have to be quite robust in order to handle all possible cases. Dealing only with small holes is nearly a non-issue. Dealing with small holes *and* large holes makes the process more difficult. Then, in addition to that, dealing with *multiple* holes of *varying* sizes further complicates matters.

The hole repair algorithm performs the following steps to correct all holes in an edge image, no matter how they were created:

1. Get the corners of the phantom by invoking get_corner() with the following information:
    - The current edge image.

- The desired corner number:
  - 1 for the top left
  - 2 for the top right
  - 3 for the bottom left
  - 4 for the bottom right

2. Perform edge tracking with follow_edge(). The edge tracker will encounter the hole created earlier, and will quit, indicating the location of the hole in the edge.

3. Repair_holes() will scout out the continuation of the edge on the other side of the hole, and calculate the difference in rows and columns.

4. Fill in the edge points in the hole, gradually stepping the edge towards the other point. The edge is shifted by one pixel when the modulus of the length of the hole and the total pixel shift across the hole (*mod(hole_length,total_pixel_shift)*) equals 1. This prevents sharp pixel shifts in the edge, and gives the best possible approximation of the edge in that region.

5. Recursively call repair_holes() again to fix the next hole in the edge. If the edge is whole, quit.

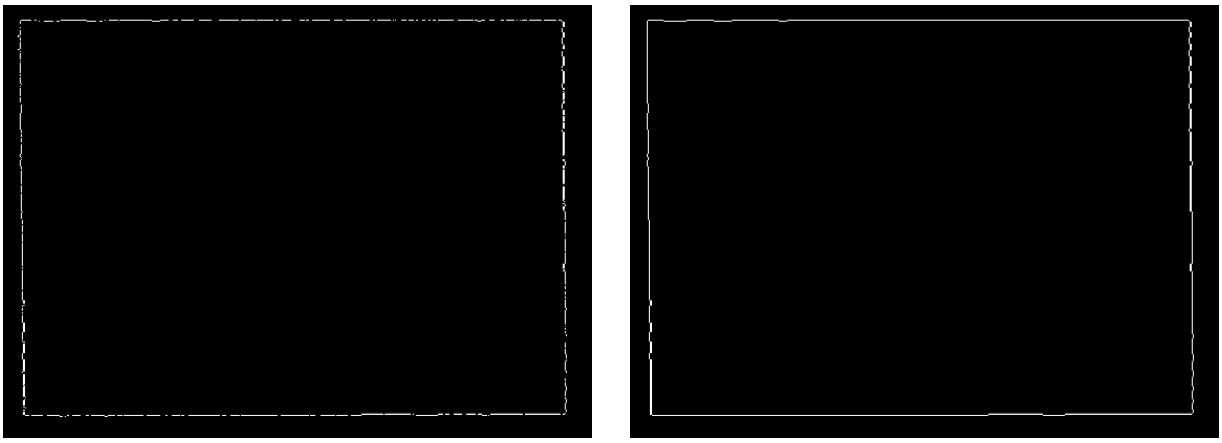
The repair_holes_edge() function performs the same steps outlined above, but only for one edge (rather than all four edges in an image). The function calls itself recursively to fix the other holes in the same edge, then quits. Other edges are not considered at this point. This is the implementation that is used by the bubble correction. This is because of the fact that during the bubble removal process, there

could be other bubbles in the other edges that would break the edge tracking algorithm used to identify holes.

The repair_holes() function goes one step further, by recursively checking for holes in **all** edges. The function will recursively check for holes in the specified edge. Once all those holes are repaired, then the function will check the other edges for holes. If any holes are found there, then the function will recursively fix each hole, one at a time, until all holes are fixed.

The repair_holes() function is the implementation used at the actual hole repair stage. This check will primarily handle the small holes caused by edge detection. The repair_holes_edge() function, on the other hand, will primarily deal with the larger holes created during the bubble correction process. All of the large holes, if any exist, will be fixed by the time repair_holes_edge() has completed its execution. Because of the recursive nature of these functions, it can take as long as 5 minutes to process one image, if that image is extremely perforated. But in the end, in *all* cases, the corrected image will be complete. Figure 3.22 demonstrates hole repair on a severely perforated edge image.

Pertaining specifically to larger holes, the key to each algorithm's robustness is its ability to gradually step the repaired edge section spanning across a shift in the original edge. Simply drawing a line to span the hole distance, then drawing another straight line to cover the shifted edge distance is the worst implementation possible. This creates edges that are greater than one pixel in size. The width of the edge should **never** be greater than one pixel. Having such an edge will adversely affect the distortion correction methods. Additionally, it is not feasible to diagonally com-



(a) Before correction. (b) After correction.

Fig. 3.22: Demonstration of hole repair on a severely perforated image.

pensate for the shifted edge distance over a short span of pixels. Doing so will create an edge similar in appearance to those that contain bubbles. Also, this creates a slanted portion in the edge, which is not a good representation of the actual edge of the phantom.

Therefore it is necessary to step the edge systematically over the span of the hole. The operator *mod(hole_length,total_pixel_shift)* ensures that when the edge is redrawn, the edge is shifted at even intervals across the span of the hole. This is the best possible approximation of the actual phantom edge in this region. The only way to get a better approximation is to rescan the phantom, and ensure that there is no noise or corruption in that region. The likelihood of a second scan coming out perfectly is very low; besides, this is not feasible whatsoever. The approximation of the edge serves its purpose well, and is more than suitable for the subsequent correction steps.

3.3.10 Verify Points

- **Function Name:** verify_points
- **Input:** One edge image
- **Output:** Corrected edge image with extraneous points removed.
- **Syntax:** verify_points(slice_edges)
- **Purpose:** Remove extra points in an edge image that do not lie in the edges.

Theory

At this point, the edge images are nearly free of unwanted features. Notice that the images are not ***entirely*** free of unwanted features. Indeed, bubbles, holes, and other noisy features have been removed. However, there is one other small feature that must be removed before the processing step can be completed:

extraneous points.

Extraneous points are simply points that exist in the edge image that are not part of the edges of the phantom. If these extra points are left in the image, subsequent computational steps will consider these extra points as valid points that are part of the phantom.

Any extra points that are left over in the phantom are a direct result of noise. The edge detection threshold chosen is sufficiently strong to remove nearly all the noisy edge points. After the previous preprocessing steps, most other extraneous points are

removed. By this stage in the processing, there is a slim chance of extra points still being present. In the very small number of images that do actually have extraneous points, there needs to be a correction method supplied to handle these points.

Extraneous points may be the result of some common image features:

- The edge of an air bubble.
- A portion of the drain plug.
- Random image noise.

Whatever the source is of the extraneous points, they all need to be removed. A simple example of this is demonstrated in Figure 3.23.

Fig. 3.23: Extraneous point below an edge.

Application

Removing extraneous points is relatively easy, thanks to the implementation of other algorithms available in the software package. The edge tracking algorithm is the tool employed to remove extraneous points, since it is immune to the presence of extraneous points in the image.

Extraneous point removal is performed in the following manner:

1. Get the corners of the phantom by invoking get_corner() with the following information:
    - The current edge image.
    - The desired corner number:
        - 1 for the top left
        - 2 for the top right
        - 3 for the bottom left
        - 4 for the bottom right
2. Perform edge tracking with follow_edge(), for all four edges.
3. Construct the edge image from these four lists of points.
4. Compare the newly-constructed edge image with the original image.
5. If the original image contains points not found on the newly-constructed image, remove all those points.
6. Return the cleaned-up image.

Prior to implementing this algorithm, the accuracy of rebuilding edge images using the edge tracker was tested using nine full studies, and found to be 100% accurate for *all* test images. Therefore, employing this method *will* yield valid results.

When comparing the two images, the method used here is based on the fact that the two edge images are large integer matrices. The empty space in the images has a value of 0, while each edge point has a value of 1. Comparing the edge points in the two edge images is as simple as subtracting the images. If the two images

are identical, the difference between them will be a matrix that contains only zeros. Any nonzero values in the difference of the two matrices indicates extraneous values. Checking for nonzero values is as easy as checking the maximum and minimum values of the result matrix: if the maximum and minimum values are zero (i.e., the same value), then the images are identical; if the values are not identical, the images are not identical.

In actual practice, the number of edge images that actually undergo correction by this algorithm are very few in number. In the three MRI studies used to develop this software package, it was found that less than 1% of the edge images (i.e., 2 out of approximately 240 images) required extraneous point removal. Although the number of images requiring correction was indeed low, the extra points will no doubt wreak havoc on all subsequent algorithms that make use of this edge data. In the end, the accuracy of the correction will suffer if these points are not removed.

3.3.11 Calculating Rotational Tilt

- **Function Name:** get_tilt
- **Input:** One edge image
- **Output:** Physical rotation of phantom from center plane (in degrees), and orientation of tilt.
- **Syntax:** get_tilt(slice_edges)
- **Purpose:** Calculate the amount of rotational tilt of the phantom, which reflects how it was placed within the MRI scanner.

Theory

The rotational tilt check is a verification step that is performed in order to ensure proper placement of the phantom within the MRI scanner. This check is very important because the entire distortion correction method is based around the assumption that the distortion is symmetric about the gradient isocenter. Proper placement of the phantom in the scanner will ensure that the center of the phantom lies as close as possible to the gradient isocenter, allowing the distortion to be as symmetric as possible in the images.

If the phantom is placed incorrectly in the scanner, the previous quality check results may not be accurate. Rotational tilt will cause an off-centering of the phantom, such that the distortion in the phantom is no longer symmetric. This means that the distortion of the phantom at the top of the image is no longer the same as—or similar to—the distortion at the bottom of the image. The correction methods for the corner

bubbles and the corner verification will therefore provide inaccurate results. Both of these functions rely heavily on this assumption.

An interesting question arises when considering this problem: why check the positioning of the phantom *after* performing so many data processing steps? The answer is quite obvious: the edge images are not suitable for accurate measurements for tilt at the early stages of data processing. Corner bubbles, noise, and incomplete edge images will not give any semblance of accuracy when calculating tilt. It is therefore necessary to ensure that the data that is used to calculate rotational tilt is as accurate as possible, and is free of errors.

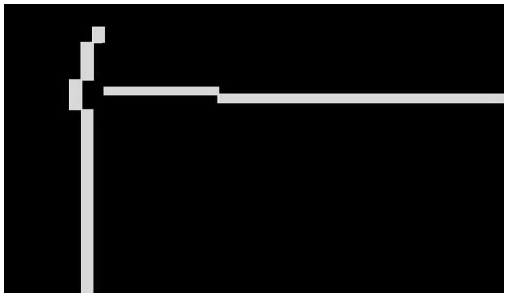


Fig. 3.24: Edge image with corrupted corner.

Take, for example, the edge image with the corrupted corner in Figure 3.24. The computed corner will be much higher than the actual physical corner of the phantom. Calculating the rotational tilt using that erroneous data point will likewise result in erroneous results for tilt. The operator will receive a message saying that the phantom is not positioned properly, and will wonder why the software reports this condition, even though the phantom may already be properly oriented within the scanner with the utmost precision. Data integrity becomes a very big issue at this stage of the processing. Poor decisions can be made, and incorrect actions may be taken, if the

data given here is poorly represented.

Application

The calculation method for the tilt is very simple:

1. Get the four corners of the phantom using the get_corners() function.
2. Using an imaginary line that connects each corner, determine the midpoint of that line. The midpoints will be used to determine the tilt of the phantom.
3. Compare the arctangents of the corresponding line midpoints (top and bottom, left and right).
4. Check the tilt values:
    - If the tilt of the horizontal line is negative and the tilt of the vertical line is positive, this shows a clockwise tilt for the phantom.
    - If the tilt of the horizontal line is positive and the tilt of the vertical line is negative, this shows a counterclockwise tilt for the phantom.

If the calculated tilt exceeds the threshold specified at the start of processing (when process() is invoked, default is 1 degree), an error message is displayed for the operator. Since this calculation is performed for all studies processed, the operator will have a good idea of which direction, and in which dimension, to rotate the phantom in order to get it properly centered. At this point, if the phantom is found to have an excessive tilt, the phantom must be re-centered, and the entire set of preprocessing steps must be executed again.

3.3.12 Calculating Center Offset

- **Function Name:** get_center_offset
- **Input:**
  - One edge image
  - Voxel size of particular scan
- **Output:** Distance of the center of the phantom from the center of the image.
- **Syntax:** get_center_offset(slice_edges,voxel_size)
- **Purpose:** Calculate the distance the center of the phantom lies from the center of the image.

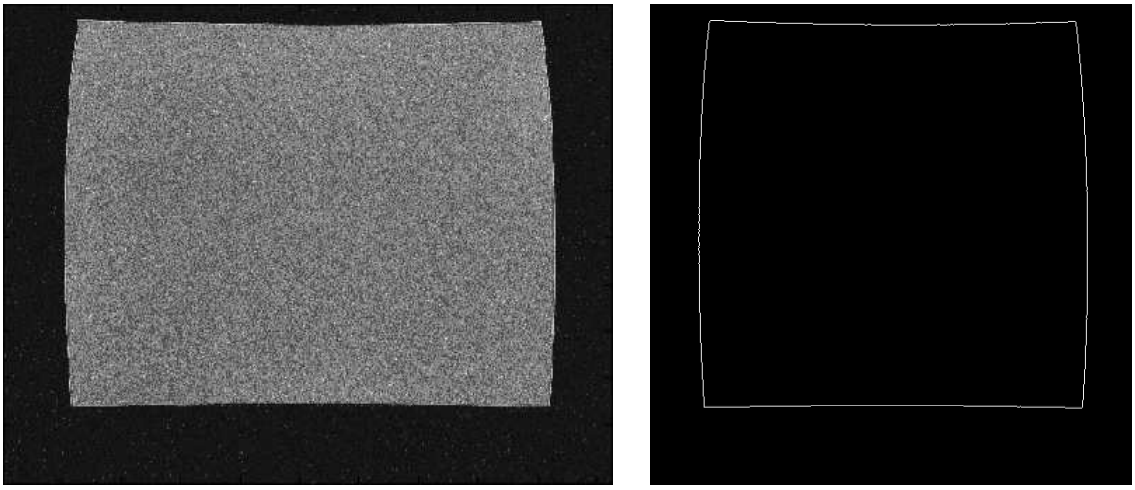
Theory

The center offset check is a verification step that is performed in order to ensure proper placement of the phantom within the MRI scanner. This check is very important because of the assumptions made concerning the distortion in the edge images. Recall that the distortion in each of the images is symmetric, meaning that the phantom should appear the same on either side of a line bisecting it.

If the phantom is placed incorrectly in the scanner, the previous quality check results may not be accurate. Incorrect centering of the phantom within the scanner will violate the assumption concerning the symmetry of the distortion in each image. In each image, the distortion at each side of the phantom is no longer the same as—or similar to—the distortion present in the opposite side. This means that the distortion of the phantom at the top of the image is no longer akin to the distortion

at the bottom of the image. The correction methods for the corner bubbles and the corner verification will therefore provide inaccurate results. Both of these functions make use of that assumption.

An off-centered phantom is quite apparent in the images displayed in Figure 3.25. This phantom is so off-centered that not only is it not physically centered in the image, but the distortion at the top of the phantom is much greater than that at the bottom, far from being symmetric.



(a) Original MR image. (b) Detected edges.

Fig. 3.25: Example of phantom offcentering in MRI scanner

As with the rotational tilt check, an interesting question arises when considering this problem: why check the positioning of the phantom *after* performing so many data processing steps? The answer is quite obvious: the edge images are not suitable for accurate measurements for centering at the early stages of data processing. Corner bubbles, noise, and incomplete edge images will not give any semblance of accuracy

when calculating center offset. It is therefore necessary to ensure that the data that is used to calculate center offset is as accurate as possible, and is free of errors.

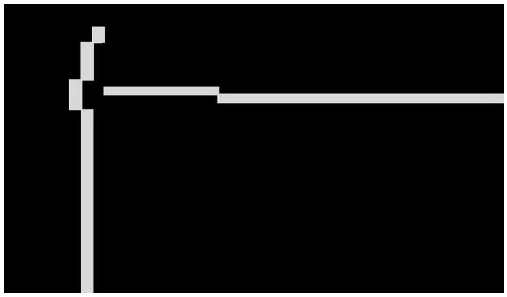


Fig. 3.26: Edge image with corrupted corner.

Take, for example, the edge image with the corrupted corner in Figure 3.26. The computed corner will be much higher than the actual physical corner of the phantom. Calculating the center offset using that erroneous data point will likewise result in erroneous results for center offset. The operator will receive a message saying that the phantom is not positioned properly, and will wonder why the software reports this condition, even though the phantom may already be properly oriented within the scanner with the utmost precision. Data integrity becomes a very big issue at this stage of the processing. Poor decisions can be made, and incorrect actions may be taken, if the data given here is poorly represented.

Application

The calculation method for the center offset is very simple:

1. Get the four corners of the phantom using the get_corners() function.
2. Average the horizontal and vertical components of the four corners. The result is an index that refers to the center of the phantom.

3. Determine the index of the center pixel in the image.

4. Take the difference of the two values.

5. Convert the result to mm, to give a physical offset.

If the calculated center offset exceeds the threshold specified at the start of processing (when process() is invoked, default is 1 mm), an error message is displayed for the operator. Since this calculation is performed for all studies processed, the operator will have a good idea of which direction, and in which dimension, to shift the phantom in order to get it properly centered. At this point, if the phantom is found to have an excessive center offset, the phantom must be re-centered, and the entire set of preprocessing steps must be executed again.

3.3.13 Auxiliary: Get Corners

- **Function Name:** get_corner
- **Input:**
  - One edge image
  - Corner to retrieve
- **Output:** Index of requested corner [row,column].
- **Syntax:** get_corner(slice_edges,corner_number)
- **Purpose:** Locate and return the location of a corner of the phantom.

Theory

One of the two auxiliary functions that is called very often during data preprocessing is the get_corner() function. This function does exactly what its name implies: returns the location of a specified corner of the phantom.

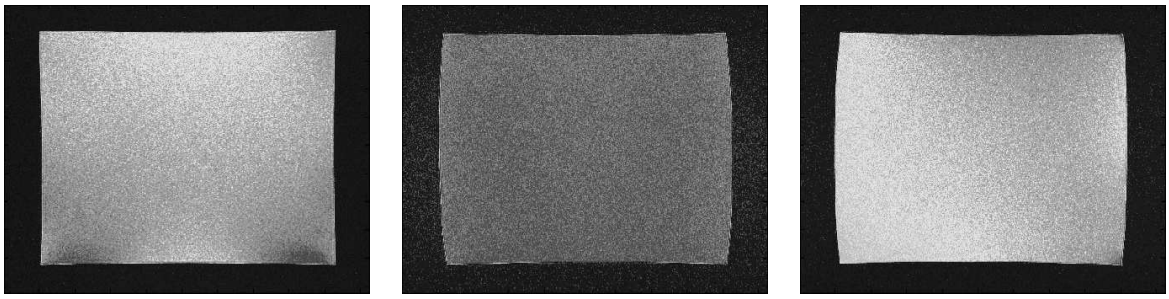
The importance of this function in preprocessing is paramount. Most of the preprocessing functions use the results returned by get_corner(). These include:

- verify_edges()
- remove_bubbles()
- verify_corners()
- repair_holes()
- get_tilt()

- get_center_offset()

Because of the heavy reliance on the location of the corners of the phantom, it is easy to see how critical accuracy is here. Incorrect measurements will greatly affect all of the functions listed above. Therefore, a very robust method is necessary in order to avoid errors, not only in corner detection, but all the results thereafter that rely on corner detection.

When analyzing the shapes of the phantom in the scans, shown in Figure 3.27, it will quickly become apparent that the phantom does not have the same shape in axial scans, as in sagittal or coronal. In coronal and sagittal scans, the left and right edges of the phantom are convex, rather than the concave edges found in axial scans. This poses a problem when edges are considered.



(a) Axial scan of phantom. (b) Coronal scan of phantom.(c) Sagittal scan of phantom.

Fig. 3.27: Typical slices from various MRI scans.

When considering points that are near the corners, it is possible to get more than two points in one column/row. In a perfect scan, with no distortion, there would only be two points per column/row, except for those columns and rows that contain the full edge. Since at least 2 of the edges are concave, that means that it is possible

to find a column that includes points not only from the top and bottom edges, but also from the nearest vertical edge. In the same manner, it is possible to find a row that contains points not only from the left and right edges, but also from the nearest horizontal edge.

Additionally, the convex vertical edges in sagittal and coronal studies exhibits points that lie farther to the left and right of the corners of the phantom. This eliminates other simple solutions to the corner location problem. It is therefore not possible to merely find the leftmost/rightmost points on the phantom, and consider those as the corners.

Based on these basic observations, it is necessary to analyze the corners and structure of the phantom more closely. The most basic question to ask is: *what is a corner?* A corner is simply the point at which two edges meet. This means that there are no other points that exist in either edge past the corner. As simple as these properties may be, they are actually quite helpful in developing the corner location algorithm.

Again, a corner is defined as:

- The point at which 2 edges meet.
- The termination of two joined edges.
- The common point between two edges.

These properties provide more than enough information to develop the corner detection algorithm. Making good use of these properties will ensure the highest level of accuracy in this algorithm.

Application

Now that a basis has been laid for the definition of a corner, the process performed by the get_corner() algorithm becomes quite clear:

1. Determine the location of the corner to be located. That is:
   - Corner 1 is the top left corner.
   - Corner 2 is the top right corner.
   - Corner 3 is the bottom left corner.
   - Corner 4 is the bottom right corner.
2. Consider only the quadrant that contains that corner ($\frac{1}{4}$ of the image), to prevent the edge points on the opposite sides of the image from confusing the process.
3. Starting at the vertical edge in that quadrant, move towards the corner in the following manner:
   - For corners 1 and 2, move up the vertical edge.
   - For corners 3 and 4, move down the vertical edge.
4. While moving, consider a window of points that is 6 pixels wide. This will be used in locating the horizontal edge.
5. If points are found to either side of the vertical edge, this may be a possible corner. Verify if this location is in fact a corner, or a bubble, by considering points within the 6-pixel-wide window. Continue all the way up/down to the edge of the image.

- If there are other points above the current "possible corner point," this means that the vertical edge continues beyond this point. Therefore the point that was located is part of a bubble or other noise, and is not the actual corner of the phantom. Continue searching.
- If there are no more points above the current "possible corner point," then this means that the vertical edge ends at this point, and the extra points found to the side are in fact points on the horizontal edge. This point is therefore the sought-after corner. Return this value.

6. If the corner is rounded, choose the point that is in the middle of the rounding. This point is not part of either edge, but lies between the termination of both edges (see below for a deeper explanation).

This method may seem more complex than what is necessary. However, the method can be fooled easily by bubbles or other irregularities in the edge. That is the reason why 2 different scouting windows are used. The scouting windows verify that the algorithm has in fact reached the actual corner of the phantom, rather than some other point.

Consider the example shown in Figure 3.28(a). Is this a real corner? Or is this a possible bubble? How can the algorithm tell the difference? The answer is simply: it can't! A human can't differentiate either! This is, in fact, part of a bubble. Zooming out in Figure 3.28(b) will make this more apparent.

Without the scouting windows, the algorithm would determine that particular point as the corner of the phantom. This will result in numerous unrecoverable data errors. With the scouting windows, the other side of the bubble would have

(a) Close-up of air bubble. (b) Full view of air bubble.

Fig. 3.28: Looks can be deceiving! An air bubble that can be mistaken for a corner of the phantom.

been located, along with the rest of the vertical edge. At this point, the algorithm would disregard this point, and continue searching for the real corner. The difference between a valid corner point and an invalid one, is whether or not other points exist beyond that proposed point.

(a) A sharp, straight corner. (b) A rounded corner.

Fig. 3.29: Typical corners in edge images.

Consider the two corners demonstrated in Figure 3.29. The corner in Figure 3.29(a) is very sharp, and features no rounding. The algorithm would choose the point that

is shared between the two edges. In the event that the corner is rounded, as in Figure 3.29(b), there is no point that is shared between both edges. Therefore, in this special case, the algorithm would choose the point that is *between* both edges. This point is unique in that it breaks all the definitions of a corner mentioned earlier:

- The point is *adjacent* to, but is not, the end of both edges.
- The point does not represent the termination of either edge.
- The point is not common between both edges.

The presence of this point is checked at the last stage of corner detection. Otherwise, the corner would be chosen as the first point in the horizontal edge. Clearly, this is not a point that lies in the vertical edge. Because of this, the point that is between both edges must be chosen. This point is the best approximation of the corner in that slice. As it is already, that particular corner is not an actual representation of the phantom. The rounding effect that is seen in some corners is actually the impression of the silicon bead on the oil inside the phantom. This silicon bead binds the faces of the phantom together. Unfortunately, the bead is large enough to interfere with the appearance of clean, sharp corners in the phantom.

Again, achieving the utmost accuracy here cannot be stressed enough. Requiring the algorithm to stringently test each possible detected corner point will ensure that the correct corner point is chosen. That will enable other algorithms to use the correct data, which will allow them to produce correct results.

3.3.14 Auxiliary: Follow Edges

- **Function Name:** follow_edge()
- **Input:**
  - One edge image
  - Edge to retrieve
- **Output:** List of points that comprise the requested edge.
- **Syntax:** follow_edge(slice_edges,edge_number)
- **Purpose:** Create a list of all points that comprise an edge, from corner to corner.

Purpose

The purpose and function of follow_edge() is very straight-forward: find and return a list containing all points in a given edge. This method can either be called *edge tracking*, *pixel walking*, or *edge following*. Whatever name is given, the method is the same.

For an edge to be continuous, every point must have at least one point adjacent to it. If this condition is not met, there exists a hole in the edge. Therefore, creating a list of edge points is fairly trivial. The only considerations that have to be made include:

- What action to take if no adjacent points are found.
- If multiple adjacent points exist, which point to take as the next point.

- Ensure that the algorithm does not oscillate between two points.

Paying close attention to the above criteria will ensure the accuracy of the edge tracker.

Because the edge tracker follows an edge, point-by-point, any holes that exist in the edge will be found. It is for this reason that the follow_edge() algorithm is used to search for holes. The hole is easily located, and its location marked, providing simple correction using repair_holes().

It is possible in some instances to have more than one point adjacent to the current point. Therefore, it is necessary to establish a priority for point decision. This will ensure that the edge tracker continues to move forward, instead of getting stuck in one position in the edge. Additionally, an unusually shaped edge (i.e., an edge that contains a large bubble) might confuse the edge tracker to oscillate between two edge points. In this case, the edge tracker might continue jumping between the two pixels indefinitely. A loop control must be set up to prevent the production of erroneous data.

As long as the above considerations are kept, the edge tracker will produce a correct set of edge points that comprise any edge in an edge image.

Application

The execution of this method is not complicated:

1. Determine the location of the edge specified. That is:

    - Edge 1 is the top edge.

- Edge 2 is the bottom edge.
- Edge 3 is the left edge.
- Edge 4 is the right edge.

2. Based on the edge chosen, retrieve the two corners that terminate the chosen edge.

3. Starting at the leftmost/topmost corner, find the next adjacent pixel to the current point. Use the following priority in choosing the next adjacent point:

   - For horizontal edges:
     (a) Right
     (b) Above-Right
     (c) Below-Right
     (d) Above
     (e) Below
   - For vertical edges:
     (a) Below
     (b) Below-Right
     (c) Below-Left
     (d) Right
     (e) Left

4. If no adjacent point is found at the current point, a hole exists. Quit execution and return [0,0] as the next entry in the edge list. [0,0] is not a valid index, since all matrices enumerate from [1,1].

5. If the second corner is reached, stop execution and return the completed list of edge points.

The result is a 3-dimensional matrix that contains all the edge points in an edge:

[row index, column index, edge point index].

In a complete edge, the first and last points in the list are the two corners of the chosen edge. If the last point of the list is [0,0], this is indicative of a discontinuous edge. A function that makes use of the list will then call repair_holes() to fix the error, and get a new corrected list.

For any corrected edge image (that is, an edge image that has gone through all the stages of data preprocessing), it is possible to make a list of all edge points, store them in four separate lists, and use those lists to recreate the edge image ***exactly***. In fact, this very method is implemented as the verify_points() function. Any points that exist in the original image and not in the reconstructed image are extraneous points not part of the edges. These are then removed, since they are merely noise corrupting the image.

Needless to say, the functionality provided by the follow_edge() algorithm will become quite useful in subsequent portions of this project.

4. STEP 2: PLANE CALCULATION

4.1 Midplane Calculation

4.1.1 Purpose

The final gradient nonlinear distortion correction method is based on a set of calculated "ideal" planes. These planes represent the appearance of the faces of the phantom, without subjection to distortion. Although these planes are estimations of the actual faces, they can be calculated to a high degree of accuracy. The errors that result from these calculations are quite minute, and do not adversely affect the final calculations.

4.1.2 Application

In order to calculate the ideal planes, it is first necessary to obtain the *midplanes* of each distorted face. The midplanes are composed of the midpoints of the edges in each edge image. The first step involves analysis of the edge images, prepared by the data preprocessing stage of execution, discussed previously in Chapter 3. The edge tracking algorithm (see Section 3.3.14) is used to generate four lists of edge points, one for each edge in the image. Then, the midpoints are calculated for each pair of opposite edges, as shown in Figure 4.1.

The algorithm excludes those points in the edge point lists that do not have oppo-

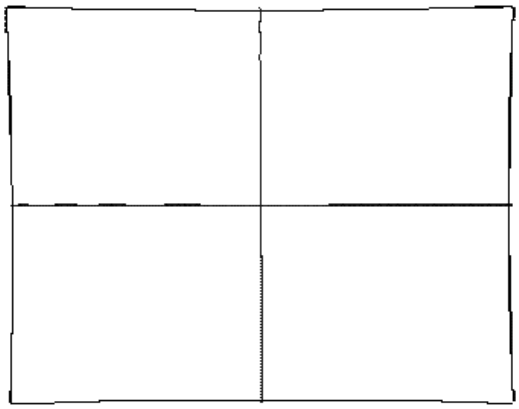


Fig. 4.1: The upper and lower edge points of the image are averaged to form two midlines, shown as the horizontal and vertical lines intersecting in the middle of the image. A plane is then fit to all the slices' midlines.

site neighbors, to ensure that the midpoint calculations generate valid results. The edge lengths can vary by a few pixels, depending on the appearance of the distortion in the image; therefore it is acceptable to have opposite edges of slightly different lengths; the algorithm handles these such edges to obtain valid results. The end result of midpoint calculation is two lines: one vertical and one horizontal. These are the intersecting lines shown in Figure 4.1. These midlines are nearly straight, and provide a suitable representation of the undistorted edges in each images, disregarding the positioning of the edges. These midpoints serve as the data that models the *ideal planes*.

Once the midlines are calculated for all edge images in a study, they are packed into matrices $m \times 3$ in size, where m is the number of points of a midline in each edge image. During the packaging process, the coordinates of the midpoints are organized into the three columns, denoting x-, y-, and z-dimensions. This process produces one

large array for each study, containing j- $m \times 3$ matrices, with j being the number of edge images in that particular study.

With the arrays from each study, the final step in data preparation is to create three large arrays to hold all the midpoints from the study, one array for each dimension (x, y, and z). Each array is n in size, with n being the total number of midpoints in the entire study. In a normal study, n can range between 28,000 and 32,000 points.

Depending on the study supplied, the algorithm determines the orientation of the plane that is to be fit to the data set, calculated using a linear least squares system. Planes are described by:

$$Ax + By + Cz + D = 0 \tag{4.1}$$

Each dimension can be isolated by rearranging the principle plane equation accordingly:

$$\begin{aligned} x &= -\left(\frac{D}{A}\right)1 - \left(\frac{B}{A}\right)y - \left(\frac{C}{A}\right)z \\ y &= -\left(\frac{A}{B}\right)x - \left(\frac{D}{B}\right)1 - \left(\frac{C}{B}\right)z \\ z &= -\left(\frac{A}{C}\right)x - \left(\frac{B}{C}\right)y - \left(\frac{D}{C}\right)1 \end{aligned} \tag{4.2}$$

For example, fitting a plane to the top and bottom edges of the images in an axial study results in a plane oriented in xz. Therefore, the plane can be expressed by:

$$y = -\left(\frac{A}{B}\right)x - \left(\frac{D}{B}\right)1 - \left(\frac{C}{B}\right)z$$

Based on the determined orientation, the algorithm organizes the data into one large matrix, $n \times 3$ in size, and one large array, n in size. The $n \times 3$ matrix is the *design matrix*, denoted by T, and the n-element array is the *observation vector*, denoted by L. The data structures are then arranged into a least squares system, described by:

$$TS = L$$

where S is the solution to the system, corresponding to the coefficients of the fitted plane. In the example mentioned above, L would contain y-data. T then contains data from the other two dimensions (x-data and z-data), plus a column of ones. The column of ones corresponds to the offset term in the rearranged plane equation (D/θ, $\theta = A, B, C$).

Solving for S—the plane's coefficients—is now trivial. In Matlab, solving this

linear least squares system is performed by using the backslash "\" operator, such that:

$$S = T \backslash L$$

Matlab employs QR factorization to solve this system (see Appendix B). The accuracy of the solution was found to be very high, based on analysis of the variance and standard deviation of the fitting results.

The fitted planes are known as the *midplanes*, since they are composed of the midlines of each edge image. The S vector contains the coefficients of the fit plane, expressed in terms of the dimension chosen:

Plane	Plane Equation	Plane Coefficients
x	$-\left(\frac{D}{A}\right)1-\left(\frac{B}{A}\right)y-\left(\frac{C}{A}\right)z$	$S=[-\left(\frac{D}{A}\right),-\left(\frac{B}{A}\right),-\left(\frac{C}{A}\right)]$
y	$-\left(\frac{A}{B}\right)x-\left(\frac{D}{B}\right)1-\left(\frac{C}{B}\right)z$	$S=[-\left(\frac{A}{B}\right),-\left(\frac{D}{B}\right),-\left(\frac{C}{B}\right)]$
z	$-\left(\frac{A}{C}\right)x-\left(\frac{B}{C}\right)y-\left(\frac{D}{C}\right)1$	$S=[-\left(\frac{A}{C}\right),-\left(\frac{B}{C}\right),-\left(\frac{D}{C}\right)]$

4.2 Ideal Plane Calculation

4.2.1 Purpose

The midplane represents the ideal shape of the faces of the phantom. However, it does not fully represent the ideal face because of its location in the center of the coordinate system. In order to properly represent the ideal faces of the phantom, the

midplanes must be shifted into the proper positions. The shifting distance for each face is $\frac{1}{2}$ the phantom dimension in that direction.

Shifting the midplanes involves a translation of $\frac{1}{2}$ the phantom dimension along the normal vector to that midplane. In terms of the coefficients of the midplane, the offset term must equal $\frac{1}{2}$ the phantom dimension, and the remaining coefficients must remain the same, in order to maintain the midplane's original orientation. This shifted midplane is known as the *ideal plane*, because it represents the ideal shape, orientation, size, and location of the phantom's faces without distortion.

4.2.2 Application

To calculate the shift, consider the plane equations mentioned earlier:

$$Ax + By + Cz + D = 0$$

$$\begin{aligned} x &= -\left(\frac{D}{A}\right)1 - \left(\frac{B}{A}\right)y - \left(\frac{C}{A}\right)z \\ y &= -\left(\frac{A}{B}\right)x - \left(\frac{D}{B}\right)1 - \left(\frac{C}{B}\right)z \\ z &= -\left(\frac{A}{C}\right)x - \left(\frac{B}{C}\right)y - \left(\frac{D}{C}\right)1 \end{aligned}$$

Considering the Hessian normal form of the plane equation:

$$N^T * p = -d \tag{4.3}$$

where $N = [A\ B\ C]$ is the *normal vector* to the plane, p is a point vector satisfying the plane equation, and $d = \frac{D}{\sqrt{A^2+B^2+C^2}}$. In terms of the rearranged equations, the normal vector N is expressed as:

$$N_x = [1, -\left(\frac{B}{A}\right), -\left(\frac{C}{A}\right)]$$
$$N_y = [-\left(\frac{A}{B}\right), 1, -\left(\frac{C}{B}\right)]$$
$$N_z = [-\left(\frac{A}{C}\right), -\left(\frac{B}{C}\right), 1]$$

Therefore, a plane oriented in xz (for example) can be expressed by:

$$y = -\frac{N_x}{N_y}x - \frac{N_z}{N_y}z + \frac{d}{N_y}$$

$$= C_x x + C_z z + C_c$$

$$= \begin{bmatrix} x & z & 1 \end{bmatrix} \begin{bmatrix} C_x \\ C_z \\ C_c \end{bmatrix} \tag{4.4}$$

The normal vector must be normalized in order to prevent any unnecessary weighting in the final result. Normalization of N produces the unit normal vector, $Nnorm$:

$$Nnorm = \frac{N}{||N||}$$

Consider a point $p = [x_0,\ y_0,\ z_0]^T$ that lies on the midplane. For each midplane, two of these dimensions are independent variables, and the third is a dependent variable. The two independent variables correspond to the dimensions represented in the design matrix T, and the dependent variable corresponds to that represented in the observation vector L.

To illustrate the derivation of shifting the midplanes, consider the top and bottom faces of a coronal study for all subsequent formulae. These faces represent z planes, oriented in xy. For this example, x and y are the independent variables, and z is the dependent variable. Given this, the simplest point p to choose for any plane is one that is located at the intercepts for the two independent dimensions. In the case of

the coronal study, point $p = [0,\, 0,\, z_0]^T$. Substituting p into the plane expression yields:

$$z_0 = -\left(\frac{A}{C}\right) * 0 - \left(\frac{B}{C}\right) * 0 - \left(\frac{D}{C}\right) 1$$
$$z_0 = -\left(\frac{D}{C}\right)$$

By isolating the dependent variable, it is now possible to determine the position of point p on the shifted plane. The shifted point, $p_2 = [x_1,\, y_1,\, z_1]^T$, is expressed by:

$$p_2 = p + d * Nnorm \tag{4.5}$$

where d is $\frac{1}{2}$ the phantom dimension. Substituting (in the case of the coronal study) $p = [0,\, 0,\, z_0]^T$ into this equation produces:

$$p_2 = \begin{bmatrix} 0 \\ 0 \\ -(\frac{D}{C}) \end{bmatrix} + d * Nnorm \tag{4.6}$$

Substituting further yields:

$$p_2 = \begin{bmatrix} 0 \\ 0 \\ -(\frac{D}{C}) \end{bmatrix} + d * [(\frac{A}{C}), (\frac{B}{C}), 1] * \frac{1}{\sqrt{(\frac{A}{C})^2 + (\frac{B}{C})^2 + 1)}}$$

where $\frac{1}{\sqrt{(\frac{A}{C})^2+(\frac{B}{C})^2+1)}}$ is the effect of normalizing N. Now the shifted point p_2 can be expressed as:

$$p_2 = p = \begin{bmatrix} 0 \\ 0 \\ -(\frac{D}{C}) \end{bmatrix} + \frac{d}{\sqrt{(\frac{A}{C})^2 + (\frac{B}{C})^2 + 1)}} * [(\frac{A}{C}), (\frac{B}{C}), 1]$$

$$p_2 = \begin{bmatrix} 0 + \frac{d}{\sqrt{\left(\frac{A}{C}\right)^2+\left(\frac{B}{C}\right)^2+1}} * \frac{A}{C} \\ 0 + \frac{d}{\sqrt{\left(\frac{A}{C}\right)^2+\left(\frac{B}{C}\right)^2+1}} * \frac{B}{C} \\ -\left(\frac{D}{C}\right) + \frac{d}{\sqrt{\left(\frac{A}{C}\right)^2+\left(\frac{B}{C}\right)^2+1}} * 1 \end{bmatrix} = [x_1, y_1, z_1]$$

Since this example considers a z-plane:

$$z = -\left(\frac{A}{C}\right)x - \left(\frac{B}{C}\right)y - \left(\frac{D}{C}\right)1$$

$$z_1 = -\left(\frac{A}{C}\right)x_1 - \left(\frac{B}{C}\right)y_1 - \left(\frac{D+}{C+}\right)1$$

Inputting p_2 from above into these plane expressions will make it possible to solve for the offset term for the shifted plane ($\frac{D}{C}$), denoted now as ($\frac{D+}{C+}$).

$$-\left(\frac{D}{C}\right) + \frac{d}{\sqrt{\left(\frac{A}{C}\right)^2 + \left(\frac{B}{C}\right)^2 + 1}} * 1 =$$

$$-\left(\frac{A}{C}\right)*\left(\frac{d}{\sqrt{\left(\frac{A}{C}\right)^2 + \left(\frac{B}{C}\right)^2 + 1}} * \left(\frac{A}{C}\right)\right) - \left(\frac{B}{C}\right)*\left(\frac{d}{\sqrt{\left(\frac{A}{C}\right)^2 + \left(\frac{B}{C}\right)^2 + 1}} * \left(\frac{B}{C}\right)\right) - \left(\frac{D+}{C+}\right)1$$

$$\left(\frac{D+}{C+}\right) - \left(\frac{D}{C}\right) + \frac{d}{\sqrt{\left(\frac{A}{C}\right)^2 + \left(\frac{B}{C}\right)^2 + 1}} = \left(\left(\frac{A}{C}\right)^2 + \left(\frac{B}{C}\right)^2\right) * -\frac{d}{\sqrt{\left(\frac{A}{C}\right)^2 + \left(\frac{B}{C}\right)^2 + 1}}$$

$$\left(\frac{D+}{C+}\right) = \left(\frac{D}{C}\right) - \frac{d}{\sqrt{\left(\frac{A}{C}\right)^2 + \left(\frac{B}{C}\right)^2 + 1}} * \left(\left(\frac{A}{C}\right)^2 + \left(\frac{B}{C}\right)^2 + 1\right)$$

$$\left(\frac{D+}{C+}\right) = \left(\frac{D}{C}\right) - d * \sqrt{\left(\frac{A}{C}\right)^2 + \left(\frac{B}{C}\right)^2 + 1}$$

Therefore, the amount to shift the midplane (i.e., the distance to be added to the original offset term) is equal to:

$$d * \sqrt{\left(\frac{A}{C}\right)^2 + \left(\frac{B}{C}\right)^2 + 1} \tag{4.7}$$

The same method follows as well for planes expressed in terms of x and y, but the final solution will be in terms of $\left(\frac{D+}{A+}\right)$ and $\left(\frac{D+}{B+}\right)$, respectively.

Since each midplane needs to be shifted twice (once in the positive and negative direction), the final expression for the z-planes will therefore be:

$$\left(\frac{D+}{C+}\right) = \left(\frac{D}{C}\right) \mp d * \sqrt{\left(\frac{A}{C}\right)^2 + \left(\frac{B}{C}\right)^2 + 1} \tag{4.8}$$

Following the same form for planes in terms of x and y, the final set of equations necessary to shift all midplanes can be derived:

Plane	Plane Expression	Offset Term
x	$-\left(\frac{D}{A}\right)1 - \left(\frac{B}{A}\right)y - \left(\frac{C}{A}\right)z$	$\left(\frac{D+}{A+}\right) = \left(\frac{D}{A}\right) \mp d * \sqrt{1 + \left(\frac{B}{A}\right)^2 + \left(\frac{C}{A}\right)^2}$
y	$-\left(\frac{A}{B}\right)x - \left(\frac{D}{B}\right)1 - \left(\frac{C}{B}\right)z$	$\left(\frac{D+}{B+}\right) = \left(\frac{D}{B}\right) \mp d * \sqrt{\left(\frac{A}{B}\right)^2 + 1 + \left(\frac{C}{B}\right)^2}$
z	$-\left(\frac{A}{C}\right)x - \left(\frac{B}{C}\right)y - \left(\frac{D}{C}\right)1$	$\left(\frac{D+}{C+}\right) = \left(\frac{D}{C}\right) \mp d * \sqrt{\left(\frac{A}{C}\right)^2 + \left(\frac{B}{C}\right)^2 + 1}$

4.3 Algorithms

Now that the mathematics and theory behind the plane calculation stage have been presented, an in-depth algorithm analysis can proceed. The algorithms follow the theory presented above in order to produce the desired result.

4.3.1 Invoking the "Calculate Planes" Algorithms

- **Script Name:** calculate_planes
- **Input:**
  - Arrays of processed edge images
  - Arrays containing study information (from DICOM headers)
- **Output:** Vectors containing midplane and ideal plane coefficients.
- **Syntax:** calculate_planes()
- **Purpose:** Master script that invokes all plane calculation steps automatically and sequentially.

Purpose

The plane calculation step commences after the preprocessing stage. After preprocessing, the software has a set of cleaned edge images, that will serve as the source of the data modeling the phantom's faces, in the presence of no distortion. Since there are so many steps involved in the actual processing, a master function was used to promote neatness and user-friendliness. This is nothing more than a script that properly calls each function in turn, and gives each function the proper data it needs to perform its duties.

As was the case with the data preprocessing stage, the plane calculation stage is highly sequential. The output of one function will feed directly as input for the next function. With each passing stage, the data undergoes various processing steps, to prepare it for its ultimate destination: midplane and ideal plane calculation.

The algorithms executed at this stage mainly involve packaging the data in various data structures, and ensuring that the data is arranged properly prior to plane calculation. Due to the sheer number of calculations and processing occurring at each stage, watching the console and waiting for each stage to complete is almost akin to watching water boil in a pot. Depending on the amount of data supplied, the plane calculation stage may take quite a while to complete. Additionally, the system that is running this software will ultimately decide how quickly these tasks will complete. For the record, the machine used to develop this software has the following specifications:

- Intel Pentium M 1.6 GHZ CPU (2 MB L2 Cache)
- 2 GB DDR2-533 MHZ RAM
- 80 GB 5400 RPM Hard Drive

When Matlab is processing the data, the CPU remains pegged at 100%. Matlab memory usage can vary anywhere between 300 MB (when only one study has been loaded, prior to processing), to over 1.5 GB (after processing has completed for 3 studies).

4.3.2 Reading Edge Images

- **Function Name:** read_all_edges
- **Input:** Array containing ALL edge images.
- **Output:** Arrays indexed by study, slice, and edge.
- **Syntax:** read_all_edges(edges)
- **Purpose:** Pack all the data points from the edge images obtained from the data preprocessing stage, to prepare for midplane calculation.

Purpose

As discussed in previous sections, the edge images contain the data that will be used to obtain the distortion correction parameters. The data preprocessing stage, described in Chapter 3, prepares all the edge data for these calculations. Now comes the time to put that data to use.

To aid in the handling of data, all edge points are packed into a series of arrays, encapsulated into one large array:

- The largest array contains arrays indexed by study.
- Each of those arrays contains arrays that are indexed by slice number.
- Each of these arrays contains four arrays that hold each of the four edges in each slice.

The studies used for software development contained 104 images, covering a slab of 208 mm. Therefore, the distance between images (slice thickness) is 2 mm. With the

phantom being a cube of approximately 160 mm, the phantom occupies approximately 80 images. Therefore, each study processed by the data preprocessing stage contains roughly 80 edge images. Given that each edge image contains 4 edges, it is possible to have up to 960 lists of edge points clustered in the largest array! With each edge containing at least 400 points, that brings the total number of data points to over 384,000!

Because of the sheer amount of data that is input to this stage, it is **absolutely crucial** to utilize a highly organized storage system, to prevent the incorrect processing and usage of these data points. There is a very high likelihood of confusing data points, and using the wrong points in certain calculations. Implementing a structured, indexed array system using read_all_edges() ensures data integrity and prevents problems in all subsequent data calculations.

Application

The packaging method employed by read_all_edges() is very straight-forward. There are three *for()* loops that iterate over each study, for each slice, and for each edge. The process is as follows:

1. Loop over each study provided (possible to have up to three studies total: axial, coronal, and sagittal).

2. Loop over the number of edge images provided in each study (determined by the size of the array containing the corrected edge images, produced during data preprocessing).

3. Loop over each of the four edges in each edge image.

- Using the edge tracking algorithm (refer to Section 3.3.14), obtain the list of edge points for each edge.
- Store the list of edge points into the final array, indexed by *study_number, image_number, edge_number.*

This triple-indexed array provides a very clean, simple method for accessing the edge data during all further calculations. Additionally, read_all_edges() takes care of over half of the data packaging necessary for the distortion correction process. The large array produced at this step will not only be used for midplane calculations, but is also used during the gradient nonlinearity distortion modeling, to obtain correction parameters.

Since the edge tracking algorithm is called so many times in this function, the array takes several minutes to pack. The process takes approximately 5 minutes to complete on the test machine:

- Intel Pentium M 1.6 GHZ CPU (2 MB L2 Cache)
- 2 GB DDR2-533 MHZ RAM
- 80 GB 5400 RPM Hard Drive

Obviously, the time required for read_all_edges() to pack the edge data will ultimately depend on the machine used. But it is important to note that no matter what machine is used, the process will indeed take several minutes to complete using this implementation.

4.3.3 Calculating Midpoints and Preparing Midplane Data

- **Function Name:** create_all_midline_data
- **Input:**
  - Array containing packed edge point data.
  - Study orientation (direction cosines).
  - Voxel size of study (mm/pixel).
  - Width of MRI scan (in pixels).
  - Height of MRI scan (in pixels).
  - Range of useful slices.
  - Slice thickness (mm).
  - Number of useful slices.
  - Toggle on/off graphing of data.
- **Output:** Array containing midpoints of edges.
- **Syntax:** create_all_midline_data(all_edges, orientation, voxel_size, scan_width, scan_height, ranges, thickness, number_slices, graph)
- **Purpose:**
  - Convert from pixel to mm coordinates.
  - Create and pack all midpoints.

Purpose

Most of the processing required to create the midplanes occurs in the create_all_midline_data() function, indicative of the large number of inputs required. The packed edge data and all the important MRI study parameters are needed to create the midplanes. The necessary pieces of data needed for each study include:

- Study orientation (direction cosines), to deteremine where the data lies in space.
- Voxel size of study (mm/pixel), to perform conversion from pixel to mm coordinates.
- Width of MRI scan (in pixels), for pixel to mm conversion.
- Height of MRI scan (in pixels), for pixel to mm conversion.
- Range of useful slices, to determine the positions of data points in each slice.
- Slice thickness (mm), for pixel to mm conversion.
- Number of useful slices, for array indexing and data conversion process.

This function calls upon four additional functions to perform each step of the midplane data creation process:

- remove_solitary_points()
- pack_edge_points()
- construct_dataset()
- cat_data_sets()

In-depth explanations of these functions will follow in later sections.

The **remove_solitary_points()** function performs the primary step in midpoint calculation. Analyzing each pair of edges in an edge image, the function will determine which points on the edges possess an opposite neighbor. In other words, the function identifies which points in each pair of edges that are useful for calculating midpoints. Obviously, calculation of a midpoint requires two data points. It is impossible to calculate a midpoint if there is only one data point given. Although this may sound extremely trivial, allowing solitary points (points without a neighbor on the opposite edge) to exist in the data sets will cause errors in midpoint calculations.

It is very common to have edges that are not equal in length. The difference in edge lengths can be caused by:

- The gradient nonlinearity distortion.
- The signal and image processing methods of the MRI scanner.
- The edge detection process.

The difference in lengths between pairs of edges is usually not more than a few pixels. Therefore, only a small number of edge points are actually removed at this stage. Once the solitary points have been removed, the useful data points are packaged into a series of arrays. These arrays feature the same structure that is employed by the read_all_edges() function. The only difference here is that the set of points from each edge is stored in a matrix, rather than a list. This greatly simplifies midpoint calculation, since all the midpoints for a pair of edges can be calculated all at once

simply by averaging the two matrices.

The next step involves converting the data points from pixel to mm coordinates and packing the data in a similar manner as read_all_edges(), using the **pack_edge_points()** function. Fitting planes to the midpoint data requires that the data be represented in a regular right-handed coordinate system: the DICOM coordinate system (see Figure 4.2). The data points that have been used thus far are indexed based on the matrix coordinate system, where the origin $[1, 1]$ is located in the top left corner of the system. In order to properly perform all subsequent calculations, the coordinates need to be converted.

In the first stage of the conversion, the data points are properly arranged into x, y, and z, based on their row, column, and slice number indexes. The conversion to $x-$, $y-$, and $z-$coordinates depends on the orientation of the study. For example, in an axial study, the row index corresponds to y, the column index corresponds to x, and the slice number corresponds to z. Once the data points are arranged, then the actual pixel to mm conversion takes place, using the pixel_to_mm() function. Additionally, slice numbers are converted to mm coordinates using the slice_pos() function. In the studies used for software development, one pixel was equivalent to 0.390625 mm, and the slice thickness was set at 2 mm. During this process, all coordinates are indexed from the origin $[0, 0, 0]$ located at the center of the scanning environment; that is, $[0, 0, 0]$ is the center row, center column, and center slice. Keep in mind that the origin may exist between pixels and slices, depending on the width and height of the images, and the number of slices included in the studies. For example, in the studies utilized for software development, the images were 512×512 pixels, and each study

contained 104 slices. Therefore, the origin of the DICOM coordinate system is located at pixel $[256.5, 256.5]$, at slice 52.5. When converted, all pixel values are indexed from this reference point.

Now that the edge points have been packed, processed for solitary point removal, and converted to a right-handed coordinate system, the midpoints can be calculated. For each edge image, there are now four matrices, one for each edge. Midpoint calculation is now trivial. Solitary point removal ensures that the matrices for each pair of edges are the *exact* same length. Therefore, the only step involved in midpoint calculation is to simply average pairs of matrices together. The result is a matrix that contains the set of midpoints for that particular pair of edges in the edge image. This set of midpoints can be thought of as a *midline.* Figure 4.1 demonstrates graphically the result of this process for one image. After calculation, the midline data is stored in the aforementioned array system, with each matrix describing one midline in one edge image.

In order to prepare the data for plane fitting, the **construct_dataset()** function assembles large matrices from the smaller matrices of midpoints. The small matrices produced by the pack_edge_data() function describe a midline in one edge image; the large matrices constructed by construct_dataset() represent the set of midlines *across an entire study.* In 3 dimensions, the set of midlines represent the *midplane.* For each study supplied, the construct_dataset() function will create 2 matrices, one for each midline represented in the edge images. Each matrix will contain approximately 32,000 data points. Given three studies (axial, coronal, and sagittal), a total of 6 matrices will be created. This equates to approximately 192,000 midpoints.

With the matrices prepared, the last step in data preparation involves combining the matrices together, to form 3 sets of midline data. Recall that when using axial, coronal, and sagittal studies at the same time, there will be two representations of each dimension given. The axial study provides $x-$ and $y-$data, the coronal provides $x-$ and $z-$data, and sagittal provides $y-$ and $z-$data. Since there are six faces on the phantom, there should only be three midplanes; therefore there should only be three sets of midlines. The **cat_data_sets()** function performs this step, by simply concatenating the appropriate data sets together. After completion, there will be one matrix that contains x midplane data, one for y midplane data, and a third for z midplane data.

The end result of the create_all_midline_data() function is these three large matrices, each containing approximately 64,000 data points, converted to the right-handed DICOM coordinate system, in millimeters. The matrices are packed into an array, so they can be easily indexed later. The edge data is now properly prepared for plane fitting, to obtain the midplanes.

Application

A great deal of processing occurs in the create_all_midline_data() function. This is the core of the midplane fitting process, and this function controls the sequential data processing steps in midpoint calculation and packaging. The secondary functions called here will be discussed in-depth in later sections. For now, the execution steps of create_all_midline_data() are as follows:

1. Given the large array of edge point data prepared by the data preprocessing

step, input each edge list into the remove_solitary_points() function. The output is two matrices, one for each processed edge.

2. Pack each of the two processed edges into another large array, once again indexed by *study_number, image_number, edge_number.*

3. Calculate the midpoints of the edges, by simply averaging pairs of matrices created by remove_solitary_points().

4. Create the large matrices that describe the midlines across an entire study, using construct_dataset().

5. Combine the matrices, to produce the final matrices that describe the three sets of midpoints of the phantom's faces, using cat_data_sets().

The amount of data being processed at this stage is enormous. Therefore, it is critical that the data is neatly and systematically organized. This will ensure that no errors will be introduced in the data sets prior to the core calculations. The plane fitting algorithms are sensitive to changes in the data sets. The results of plane fitting are assumed to be representative of the scan data, without any inaccuracies introduced as a result of data processing and packaging.

Needless to say, packaging errors were introduced during early stages of software development. All of the errors were a result of incorrect packaging procedures, and (eventually) incorrect usage of data during calculations. Therefore, it was found essential that the packaging and midpoint calculation processes be controlled by a highly organized function, that has data integrity as its primary goal. Using such an organized methodology also aids in clean, easy to read code, that is simple to

maintain and upgrade. Additionally, should errors be found in the data calculations, they are easier to track down in an organized system, rather than one that is not as structured.

4.3.4 Remove Solitary Points

- **Function Name:** remove_solitary_points
- **Input:**
  - Four arrays containing the edge points from one slice.
  - Direction to be processed (vertical or horizontal edges)
- **Output:** Two arrays of processed edges (either vertical or horizontal edges).
- **Syntax:** remove_solitary_points(edge_points,direction)
- **Purpose:** Ensure that each pair of edges in each slice contain the same number of points, in adjacent locations, prior to calculating midpoints and midplanes.

Purpose

Midpoint and midplane calculation are the two most important functions performed for the estimation of the gradient nonlinearity distortion. The midpoints provide a representation of the ideal shape, size, and orientation of the phantom's faces, in the presence of no distortion. In subsequent steps, these midpoints will be shifted $\frac{1}{2}$ the phantom dimensions, to create a set of ideal planes, that fully represent the phantom's faces without distortion.

Recall that the primary assumption given in calculating the correction for gradient nonlinearity distortion is the fact that the distortion is symmetric about the gradient isocenter. Given this condition, the appearance of opposite edges in an edge image should therefore be mirror images of each other. In other words, opposite edges (horizontal or vertical) should theoretically contain the same number of points.

However, this ideal condition is virtually impossible to achieve; there is no possibility of centering the phantom ***exactly*** in the center of the scanner, such that the center of the phantom lines up exactly with the gradient isocenter. The same holds true with rotational tilt: it is virtually impossible to position the phantom such that it is perfectly level across all three planes. Therefore, any slight rotational tilt that the phantom exhibits can also contribute to this condition.

Even though the threshold for center offset (the distance that the center of the phantom is from the gradient isocenter) is only 1 mm, and the threshold for rotational tilt is only 1 degree, *any* offset or tilt will introduce small asymmetries in the distortion seen in the edges. These asymmetries appear as edges of unequal length. It is very common—and perfectly acceptable—to have two opposite edges that differ in length by one or two pixels. Unless scrutinized closely, these additional distortions are not immediately apparent.

Other factors that can contribute to the difference in edge lengths include:

- The gradient nonlinearity distortion.
- The signal and image processing methods of the MRI scanner.
- The edge detection process.

Unfortunately, these small discrepancies can introduce many errors during midpoint calculation. By definition, a midpoint is the point that lies equidistant between *two* points. How, then, is it possible for a midpoint to be calculated with one point? Applying this principle to the uneven edges reveals the large problem that is present when calculating midpoints. If there are solitary points present in the midpoint cal-

culation (i.e., points on one edge that do not have another neighboring point on the opposite edge), the midpoint calculations will fail with errors. This principle makes the importance of solitary point removal prior to midpoint calculation very obvious.

Application

The solitary point removal method utilizes a simple but effective method of determining which points are suitable for midpoint calculation:

1. Determine which edges to use, given the direction specified.
   - If the direction is 1, use the horizontal (top and bottom) edges.
   - If the direction is 2, use the vertical (left and right) edges.
2. Calculate the range that both edges occupy:
   - For horizontal edges: compare the two leftmost points in the edges, and choose the rightmost of those two points.
   - For vertical edges: compare the two highest points in the edges, and choose the lowest of those two points.

   The method above of specifying the range will ensure that points on both edges will possess an opposite neighbor with which to calculate a midpoint.
3. Pack the edge points that lie within the calculated range. Return two arrays: one for each processed edge.

Keep in mind that this function does *not* perform the actual midpoint calculation; midpoint calculation occurs in create_all_midline_data(). remove_solitary_points()

merely prepares the edge points so that midpoint calculations can take place. When the actual calculations are performed, they are performed in one large batch. Large matrices are averaged, rather than calculating each individual point. Although the processing time for both methods would theoretically be the same, the matrix averaging method is much easier to program. It is important to note here as well the progression of data as the calculations are performed. Each pair of edge points produces a midpoint. The set of midpoints, calculated from a pair of edges, comprises a midline. And finally, the set of midlines, calculated across all slices, constitutes the midplanes.

4.3.5 Pack Edge Points

- **Function Name:** pack_edge_points
- **Input:**
  - Study orientation (direction cosines).
  - Slice number of image to be processed.
  - Voxel size of study (mm/pixel).
  - Width of MRI scan (in pixels).
  - Height of MRI scan (in pixels).
  - Range of useful slices.
  - Slice thickness (mm).
  - Number of useful slices.
  - Current edge being processed.
- **Output:** Matrix containing one edge, in mm coordinates.
- **Syntax:** pack_edge_points(orientation, slice_no, voxel_size, scan_width, scan_height, ranges, thickness, number_slices, current_edge)
- **Purpose:**
  - Convert from pixel to mm coordinates.
  - Create right-handed coordinate system.
  - Convert edge points from list storage to matrix storage.

Purpose

The pack_edge_points() function is the second of four secondary functions that is called by create_all_midline_data() to prepare the edge data points for further calculation. This function performs the most important task during this process: converting the pixel coordinates to mm coordinates. From this point forward, data points will no longer be considered as pixels, indexed in matrix-style form. They are now given real locations within the DICOM coordinate system, shown in Figure 4.2.

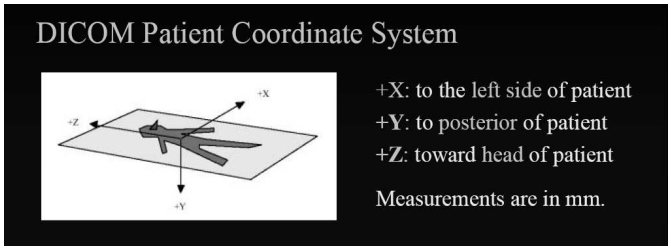


Fig. 4.2: DICOM coordinate system.

Recall that in the original pixel coordinate system, the origin is at $[1, 1]$ in the top left corner of the system. Therefore, all data points are positive, and are greater than $[1, 1]$. Additionally, each image is indexed from 1, and the direction of increasing slice indexes depends on which dimension the slices are encoded. This coordinate system quickly becomes confusing, and pinpointing a location within the coordinate system becomes difficult, at best. One other note to make is the fact that the units of the indexes are not consistent: row and column locations are expressed in pixels, while slice numbers are referenced (somewhat) in mm. Referencing a chosen location within the three studies will prove to be difficult, given that a 3-dimensional coordinate in this system uses two different units.

The use of the DICOM coordinate system eliminates this tricky problem by creating a system that is consistent, familiar, and easy to index. All three-dimensional coordinates in the system use only one unit type (millimeters), which allows trivial referencing of a location between different studies. The DICOM system makes plane fitting and gradient nonlinearity distortion modeling possible; the previous system will not allow for any semblance of comprehensible results when fitting planes or calculating distortion parameters.

In the first stage of the conversion process, the edge point coordinates are arranged properly as $x-$, $y-$, and $z-$coordinates, based on the orientation of the study provided. The following table illustrates the conversion process:

	Row Index	**Column Index**	**Slice Number**
Axial	y	x	z
Coronal	z	x	y
Sagittal	z	y	x

Once the data points are arranged, they can undergo pixel-to-mm conversion. The coordinates are input into the pixel_to_mm() and slice_pos() functions, which perform the mm conversion and properly orients the points in the right-handed coordinate system described earlier. The pixel_to_mm() function handles row and column indexes, while the slice_pos() function converts slice numbers to mm locations. The converted data points are then packed into a $m \times 3$ matrix, where m is the number of points in one edge. These matrices are then stored in the same array system used by read_all_edges(). The only difference between the two arrays is that the one created with converted mm coordinates stores the edge data points as matrices, not lists.

Application

The pack_edge_points() function performs a key step in data preparation for plane calculation and distortion modeling. The specific execution steps it implements are as follows:

1. Based on the orientation of the study provided, determine which dimensions the rows, columns, and slice numbers encode.
2. Extract each data point from the list of edge points.
3. Input the row and column data from the data point, along with all the necessary study information, into the pixel_to_voxel() function.
4. Input the slice number, along with all the necessary study information, into the slice_pos() function.
5. Arrange the converted row, column, and slice number indexes properly as $[x, y, z]$.
6. Pack each converted data point into a matrix. The size of this matrix is $m \times 3$, with m being the number of data points in that particular edge.

From this point forward, all data points are expressed as "real" coordinates, in that they exist in a regular right-handed coordinate system, and are expressed in terms of a real-world unit. The data is now suitable for use in subsequent calculations, not only because of the unit and coordinate system conversion, but also because the data has been packed into matrices. Recall that fitting a plane to a data set requires the data to be expressed as a matrix.

Visually inspecting the data points within the matrices at this stage may be confusing, since the points were always expressed in a different unit system prior to this stage. But after some analysis, and taking into consideration where the data points lie in three-dimensional space, the data will begin to make sense.

As mentioned earlier, the pack_edge_points() function is used in two different places in the distortion correction process. The first place is in the data preparation for midplane calculation. The data points processed for solitary points are packed into matrices, to enable trivial midpoint calculation. Since the remove_solitary_points() function has ensured that all pairs of edges are of the same length, the matrices constructed at this step also reflect that property. The next step after this process is to simply average the two matrices. The data in both edge point matrices are guaranteed to line up, so the midpoint calculations will always be 100% accurate.

The second place that pack_edge_points() is used is in the modeling of the gradient nonlinearity distortion. Here, *all* the edge points are packed using this function. Solitary points are not removed in that section; therefore, the pack_edge_points() function must be used twice to pack data throughout the course of the correction process. From here, the packed data points are used to both estimate the theoretical undistorted data points, and used as inputs to directly calculate the distortion parameters.

4.3.6 Construct Data Sets

- **Function Name:** construct_dataset
- **Input:** Array containing matrices of midpoints/edge points.
- **Output:** One large matrix containing all data points.
- **Syntax:** construct_dataset(dataset)
- **Purpose:** Consolidate all data points into one large matrix, to prepare for midplane fitting.

Purpose

The construct_dataset() function performs one of the final procedures in preparation for fitting planes to the midpoint data. Because of the method that the edge points are originally packed, the data needs to undergo a series of processing steps in order to make plane fitting possible. So far, the edge points have been sampled and packed into arrays. Next, the edge points were repacked into matrices, with one matrix holding the edge points on one image. Now, the task at hand is to combine all the matrices from a study into one large matrix.

One image yields two midlines, each consisting of over 400 midpoints. With each study containing on average 80 useful images, each set of midlines therefore contains roughly 32,000 midpoints. The 32,000-point matrices comprise a three-dimensional plane, to which a plane described by $Ax + By + Cz + D = 0$ can be fit. The construct_dataset() function produces these matrices.

Application

The process of assembling the large matrices is as follows:

1. Count the number of midpoints that were calculated for each image in the study.
2. Allocate the memory space for one matrix that can hold all the midpoints.
3. Reading each matrix of midpoints, copy the contents of each matrix into the single large matrix.

At this stage, it is now possible for planes to be fit to these matrices. The data that is stored in these matrices provide a representation of the shape, size, and orientation of the phantom's faces across the entire study.

4.3.7 Concatenate Data Sets

- **Function Name:** cat_data_sets
- **Input:**
  - Array containing the matrices that hold all the midpoints from each study.
  - Identifiers denoting the study represented in each matrix.
  - Dimension (i.e., orientation) of the data set to be concatenated.
- **Output:** One large matrix containing all data points in a given dimension.
- **Syntax:** cat_data_sets(data, orients, dimension)
- **Purpose:** Create one large matrix that contains all the data points describing one midplane.

Purpose

In the previous step, the construct_dataset() function created one large matrix that held all the midpoints across one study. It is possible to fit a plane to each of these matrices. However, if there is more than one study provided for processing, there will be more than one representation of the midplanes. The following table demonstrates the data provided by each study:

Sequence	**Row Data**	**Column Data**
Axial	y	x
Coronal	z	x
Sagittal	z	y

Therefore, when providing the software with one of each sequence, there will be two representations of each midplane data set provided. To maximize the accuracy of the distortion correction, it is highly recommended to perform all three sequences. Doing so provides twice the amount of information. Therefore, the midplane data sets will contain approximately 64,000 data points.

The task at hand now is to combine the correct data sets to produce the large $64,000 \times 3$ matrices. Concatenation of the data sets is as follows:

Plane Orientation	**Studies to Concatenate**
X	Axial and Coronal
Y	Axial and Sagittal
Z	Coronal and Sagittal

Obviously, concatenation only occurs in the event that there are multiple studies provided. When one study is provided, no concatenation occurs because only two dimensions are represented in the data. In order to provide representations in all three dimensions, a minimum of two studies must be provided. When two unique sequences are provided, there will be two data sets produced that contain 32,000 points, and one data set containing 64,000 points. In other words, concatenation of only one data set is necessary. Providing all three sequences (axial, coronal, and sagittal) will yield in three concatenations, since there are now two representations of each data set.

Application

The cat_data_sets() function performs data concatenation as follows:

1. Locate each data set matrix in the data array.

2. Determine how many data sets are provided. This will determine which data sets need concatenating, if any.

3. Based on the number of data sets provided for each plane type, perform concatenation. Return completed matrices.

The matrices produced at this point are finally suitable for plane fitting. Combining the data sets and then performing plane fitting is much more accurate than fitting planes separately to each data set, then averaging the plane coefficients. Therefore, the extra processing involved for this step is easily justified.

4.3.8 Fit Midplanes

- **Function Name:** calc_coeffs_midplane
- **Input:**
  - Matrix containing one set of midpoints across a given study.
  - Orientation of plane.
  - Toggle displaying of plane fitting results.
  - Toggle graphing of plane fitting results.
- **Output:** Vector containing coefficients of fitted plane.
- **Syntax:** calc_coeffs_midplane(edge_points, plane_type, display, graph)
- **Purpose:** Fit planes to midline data.

Purpose

The plane fitting function is one of the most important functions in the gradient nonlinearity distortion correction. All of the work performed in the previous processing steps pays off at this stage. The accuracy of all previous steps is directly reflected during plane fitting. The mathematical basis of the plane fitting procedures is described in Section 4.1.

The midplanes that are calculated at this step are planes fit to the midpoint data calculated by the create_all_midline_data() function, detailed in Section 4.3.3. Recall that each image yields two sets of midpoints: one for the horizontal edges, and one for the vertical edges. Each set of midpoints from an edge image constitutes a midline.

The sets of midlines across an entire study comprise a three-dimensional plane. The calc_coeffs_midplane() function fits planes to these data sets, processed and prepared by the construct_dataset() and cat_data_sets() functions (described in Sections 4.3.6 and 4.3.7, respectively).

Application

Midplane fitting is performed in the following manner:

1. Organize data to solve for plane coefficients using linear least squares, described by: $TS = L$, where L is a vector containing data from the dimension to be solved, T is a matrix containing data from the other two dimensions, plus a column of ones, and S is the solution to the system (i.e., the plane coefficients).
   (a) For an x plane (plane oriented in yz):
      i. Fill L with x data, from the first column of the midplane data matrix (the x column).
      ii. Fill the first column of T with ones, the second column with y midplane data, and the third column with z midplane data.
   (b) For a y plane (plane oriented in xz):
      i. Fill L with Y data, from the second column of the midplane data matrix (the y column).
      ii. Fill the first column of T with x midplane data, the second column with ones, and the third column with z midplane data.
   (c) For a z plane (plane oriented in xy):

i. Fill L with z data, from the third column of the midplane data matrix (the z column).

ii. Fill the first column of T with x midplane data, the second column with y midplane data, and the third column with ones.

2. Solve for S using QR factorization in Matlab: $S = T \backslash L$:

   (a) For an x plane (plane oriented in yz):

   - $S = [-\left(\frac{D}{A}\right), -\left(\frac{B}{A}\right), -\left(\frac{C}{A}\right)]$

   (b) For an y plane (plane oriented in xz):

   - $S = [-\left(\frac{A}{B}\right), -\left(\frac{D}{B}\right), -\left(\frac{C}{B}\right)]$

   (c) For an z plane (plane oriented in xy):

   - $S = [-\left(\frac{A}{C}\right), -\left(\frac{B}{C}\right), -\left(\frac{D}{C}\right)]$

The midplane fitting procedure is handled by Matlab's implementation of QR factorization, denoted by the backslash "\" operator (see Appendix B). This method was chosen because of its high accuracy and ability to adapt to the particular data set provided. The primary steps involved in calc_coeffs_midplane() involve organizing the data set properly in order to perform the actual plane fitting. The plane fitting procedure itself is rather trivial. However, do not underestimate the importance of the introductory steps in this function! Arranging the data properly is vital to achieving the desired results. Improper organization of the data will result in poor, incorrect, improper plane fits that are utterly useless to subsequent algorithms.

The resulting vector of coefficients describe the midplane, a surface that has the ideal shape, size, and orientation of the faces of the phantom, in the presence of no

distortion. This is the first time that the phantom can be represented without the effects of the gradient nonlinearity distortion. However, do keep in mind that these surfaces are ***not*** in the proper locations; they merely lie in the center of the coordinate system. They still need to be shifted the proper distance, in order to fully represent the undistorted faces of the phantom.

Depending on the options specified at runtime, the calc_coeffs_midplane() function can also display the fitting results numerically and graphically. The numerical results displayed include:

- Number of data points
- Total residuals
- Variance
- Standard Deviation

These results provide a good performance metric for the plane fitting procedure, and indicate the accuracy of the results. The calculations performed in the error analysis is described in Appendix A.

The following example output demonstrates typical plane fitting results. Figure 4.3 provides the graphical results of the same plane fitting procedure. The individual midpoints are marked as grey stars. The slant of the plane and data sets is greatly exaggerated for visualization purposes. Each "step" in the data represents 0.2 mm.

```
Midplane fitting results (in mm):


Number of data points:  62980
```

```
Total residuals (error) = 27.2311

MSE (Variance) = 0.011774

RMS (Standard Deviation) = 0.10851

Coefficients: [-(A/B) -(D/B) -(C/B)] = [0.00031188, 0.87814, 0.00
```

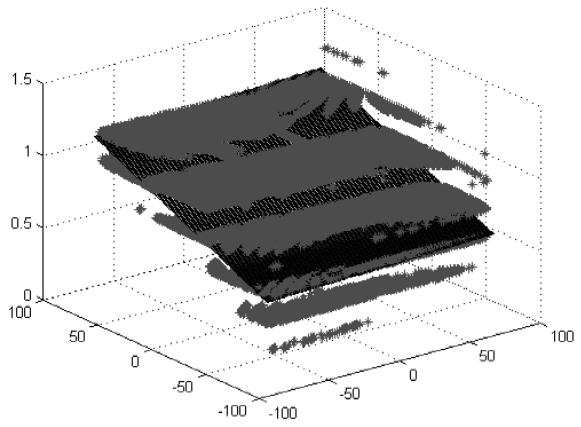


Fig. 4.3: Midplane fitting results.

In this example, the standard deviation of the plane fitting results is less than one-third of a pixel (0.130208 mm). Additionally, the coefficients provide an indication of the physical centering of the phantom in the MRI scanner. The offset term in the plane coefficients (in this case, the second term, indicating that this is a y plane, oriented in xz) shows an offset of 0.87814 mm in y. Given that the threshold for center offset is 1 mm, the phantom was properly centered in y in this particular study.

It is simple to see that the midplane coefficients provide more than just a representation of the phantom's faces. They also serve as a second check of the phantom's centering the scanner. Recall the importance of properly centering the phantom in the scanner. Since the distortion is symmetric about the gradient isocenter, it is of

the utmost importance to center the phantom as close to the gradient isocenter as possible. The midplane calculation is one of the many stages in the calculations that any mistakes in the original setup can be easily identified. By this time, it is assumed that any problems in the original setup have been resolved, and the coefficients should reflect a proper setup. Therefore, any discrepancies that are revealed in the plane fitting will most likely be the result of other systematic errors, or poor quality data.

4.3.9 Shift Midplanes / Create Ideal Planes

- **Function Name:** shift_midplanes
- **Input:**
  - Vector containing midplane coefficients.
  - Orientation of plane.
  - Toggle displaying of plane characteristics.
- **Output:** Two vectors containing coefficients of shifted planes.
- **Syntax:** shift_midplanes(coeffs, plane_type, display)
- **Purpose:** Create the ideal planes by shifting the midplanes calculated previously.

Purpose

The shifting of the midplanes is the final stage in plane calculation. The results of this stage will be used extensively to model the gradient nonlinearity distortion, based on the sum of spherical harmonics.

In the previous stage, the midplanes were calculated from the midpoint data sets. The midplanes provide the proper representation of the phantom's faces, in terms of shape, size, and orientation. The only aspect that is missing is the plane's proper *location.* The midplanes currently lie in the middle of the coordinate system. In order to provide a total representation of the phantom's faces in software, it is now necessary to shift the midplanes by one-half the phantom distance. The shifted midplanes

are then referred to as the *ideal planes.* The mathematical basis of the ideal plane calculation is described in Section 4.2.

Application

Ideal plane calculation is performed in the following manner:

1. Read the phantom's dimensions, provided in a text file. These dimensions are expressed in millimeters.
2. Based on the midplane provided, determine which direction the plane needs to be shifted, and choose the proper phantom dimension accordingly.
3. Shift the midplane by displacing the plane along the normal vector:
   - For x midplanes (midplanes oriented in yz):
     (a) Midplane coefficients = $[-\left(\frac{D}{A}\right), -\left(\frac{B}{A}\right), -\left(\frac{C}{A}\right)]$.
     (b) Calculate $\frac{1}{2}$ the phantom dimension in x, the distance by which the midplane will be shifted: dist = x_phantom_dimension / 2.
     (c) Calculate n, the effect of normalizing the normal vector: $n = \frac{1}{\sqrt{1+(\frac{B}{A})^2+(\frac{C}{A})^2)}}$
     (d) Shift the midplane in the negative direction: $\left(\frac{D}{A}\right) = \left(\frac{D}{A}\right) - dist * n$
     (e) Shift the midplane in the positive direction: $\left(\frac{D}{A}\right) = \left(\frac{D}{A}\right) + dist * n$
   - For y midplanes (midplanes oriented in xz):
     (a) Midplane coefficients = $[-\left(\frac{A}{B}\right), -\left(\frac{D}{B}\right), -\left(\frac{C}{B}\right)]$.
     (b) Calculate $\frac{1}{2}$ the phantom dimension in y, the distance by which the midplane will be shifted: dist = y_phantom_dimension / 2.

(c) Calculate n, the effect of normalizing the normal vector: $n = \frac{1}{\sqrt{1+(\frac{A}{B})^2+(\frac{C}{B})^2)}}$

(d) Shift the midplane in the negative direction: $\left(\frac{D}{B}\right) = \left(\frac{D}{B}\right) - dist * n$

(e) Shift the midplane in the positive direction: $\left(\frac{D}{B}\right) = \left(\frac{D}{B}\right) + dist * n$

- For z midplanes (midplanes oriented in xy):

(a) Midplane coefficients = $[-\left(\frac{A}{C}\right), -\left(\frac{B}{C}\right), -\left(\frac{D}{C}\right)]$.

(b) Calculate $\frac{1}{2}$ the phantom dimension in z, the distance by which the midplane will be shifted: dist = z_phantom_dimension / 2.

(c) Calculate n, the effect of normalizing the normal vector: $n = \frac{1}{\sqrt{1+(\frac{A}{C})^2+(\frac{B}{C})^2)}}$

(d) Shift the midplane in the negative direction: $\left(\frac{D}{C}\right) = \left(\frac{D}{C}\right) - dist * n$

(e) Shift the midplane in the positive direction: $\left(\frac{D}{C}\right) = \left(\frac{D}{C}\right) + dist * n$

As described in Section 4.2, the shifting of the midplanes is quite trivial. The primary work that is involved at this stage is determining which type of midplane is provided. The organization and proper identification of the orientation of the midplane is very crucial, as was the organization of the data during midplane fitting. The improper organization and identification of the midplane will result in an incorrect shift, and therefore a very incorrect representation of the phantom's faces.

The calculations reflect one primary concept that is outlined in the mathematical basis: the planes are only shifted in one dimension. The tilt and orientation of the original plane are preserved exactly. This is indicated by the fact that only the offset term of the midplane is modified, to produce the coefficients of the ideal plane.

A possible source of error in this function is in the specification of the phantom's dimensions. Unfortunately, this error can arise from human mistake, so there is little that the algorithm can do in order to prevent mistakes in this area. The algorithm

simply assumes that the phantom's dimensions specified in the text file are accurate. Any inaccuracies in the measurements, and therefore the specification, of the phantom's dimensions will result in inaccurate coefficients of the ideal planes.

Referring back to the example presented at the end of Section 4.3.8, the ideal planes were calculated as follows:

```
Ideal plane results:


Ideal Plane 1:
Coefficients: [-(A/B) -(D/B) -(C/B)] = [0.00031188, -78.9724, 0.0


Ideal Plane 2:
Coefficients: [-(A/B) -(D/B) -(C/B)] = [0.00031188, 80.7287, 0.00
```

The dimensions of the phantom were specified as 159.70 mm in y. The ideal plane coefficients calculated in this example indicate that the phantom is 159.7011 mm in y (78.9724 mm + 80.7287 mm = 159.7011 mm), nearly identical to the specified measurement. This result demonstrates the quality of the initial setup, the quality of the resulting acquired data, and the accuracy of the plane calculations. With results such as these, the operator can continue forward with confidence, knowing that the calculations are extremely accurate thus far.

5. STEP 3: DISTORTION ESTIMATION AND CORRECTION

5.1 Distortion Estimation and Correction

5.1.1 Purpose

The third and final stage involves performing the actual gradient nonlinearity distortion correction. The distortion correction model is based on the *sum of spherical harmonics*. Using the ideal planes obtained from the previous stage, theoretical undistorted data points will be calculated using the ideal plane equations and acquired data points. The undistorted data points are then used to calculate the coefficients of the distortion correction model. Three sets of coefficients are necessary, one for each dimension. Once the model is known it can be applied to all studies performed on the particular scanner.

The method employed to perform the distortion correction is loosely based on the implementation used by Langlois, et al. [33]. The distortion model is the same in both implementations: Like Langlois, only the order-two spherical harmonics are used. The higher order terms are least significant, and do not contribute greatly to the final solution.

The primary difference between Langlois' method and this implementation is the distortion correction itself. In Langlois' method, the distortion is modeled using the

sum of spherical harmonics (Equation 1.28), and the model model describes the displacement of the data points under the influence of the gradient nonlinearity distortion. To perform the distortion correction, Langlois inverted the distortion correction model, to obtain "undistortion" parameters (see Section 1.8.4). This method proves to be cumbersome and allows for the propagation and growth of calculation errors. Since there is only a finite amount of storage space for calculations, the results for each sub-calculation are subject to rounding or truncation error. These small errors build up over time, over numerous calculations, thereby skewing the final result.

The correction procedure employed in this work calculates the "undistortion" parameters directly. Instead of analyzing how the original data points were displaced in the presence of the gradient nonlinearity distortion, the software analyzes the displacement that the distorted data points must undergo in order to move them into their corrected undistorted positions. This is the exact opposite of the calculation performed by Langlois. Since the "undistortion" parameters are calculated directly, no matrix inversion is necessary. The benefits of this method are numerous:

- Reduces number of calculations necessary.
- Preserves accuracy of final result.
- Speeds up calculation process.

5.1.2 Application

The mathematical basis behind the gradient nonlinearity distortion correction is very straightforward. In particular, let x_i, y_i, and z_i be the distorted coordinates of some point i measured by an MR scanner. Further let $\bar{x}_i$, $\bar{y}_i$, and $\bar{z}_i$ be the undistorted

coordinates that corresponds to x_i, y_i, and z_i. That is: $\bar{p} = (\bar{x}_i, \bar{y}_i, \bar{z}_i)$ is the point on the ideal plane that produced the point $p = (x_i, y_i, z_i)$ in the MR image. Taking the first spherical harmonics, the correction model takes the form:

$$\bar{x}_i = x_i(1 + K_{x_0}(x_i^2 + y_i^2) + K_{x_1}z_i^2 + K_{x_2}z_i^2(x_i^2 + y_i^2) + K_{x_3}(x_i^2 + y_i^2)^2 + K_{x_4}z_i^4) \quad (5.1)$$

$$\bar{y}_i = y_i(1 + K_{y_0}(x_i^2 + y_i^2) + K_{y_1}z_i^2 + K_{y_2}z_i^2(x_i^2 + y_i^2) + K_{y_3}(x_i^2 + y_i^2)^2 + K_{y_4}z_i^4) \quad (5.2)$$

$$\bar{z}_i = z_i(1 + K_{z_0}(x_i^2 + y_i^2) + K_{z_1}z_i^2 + K_{z_2}z_i^2(x_i^2 + y_i^2) + K_{z_3}(x_i^2 + y_i^2)^2 + K_{z_4}z_i^4) \quad (5.3)$$

It is interesting to take note that nearly all the data represented in these equations is already known! The distorted data points, described by $p = (x_i, y_i, z_i)$, are obtained from the edge images processed earlier by the data quality assurance algorithms (Chapter 3). The ideal data points, described by $\bar{p} = (\bar{x}_i, \bar{y}_i, \bar{z}_i)$, are obtained from the ideal planes calculated in the previous step (Chapter 4). The only terms that are unknown at this time are the K-terms (K_x, K_y, K_z). The K-terms represent the desired correction parameters, that is, the parameters that will perform distortion correction (rearranging Equations 5.1-5.3 will provide distortion parameters, to distort a data set; more on this topic later). Solving for the K-terms is accomplished using least squares, presented in Section 5.2.6.

Calculation of the ideal data points ($\bar{p}$) is accomplished using the ideal planes. This is because the distortion in each face of the phantom is essentially in the direction

of the plane. For example, $\bar{z}_i$ can be calculated using the ideal planes oriented in the xy direction (i.e., the z ideal planes), since the distortion for those faces are essentially all in the $z$direction. The assumption can then be made that the ideal x and y coordinates ($\bar{x}_i$ and $\bar{y}_i$) will be the same as the distorted x and y coordinates (x_i and y_i).

Following the example with the z data presented in Section 4.2.2, obtaining ideal z data is accomplished by Equation 5.4. To simplify the expressions presented in Section 4.2.2, for readability and ease of comprehension, substitute the plane expression terms with simpler symbols:

$$C_{Z_x} = \left(\frac{A}{C}\right)$$
$$C_{Z_y} = \left(\frac{B}{C}\right)$$
$$C_{Z_z} = \left(\frac{D}{C}\right)$$

therefore yielding Equation 5.4:

$$\begin{aligned} C_{Z_z} \mp d * \sqrt{C_{Z_x}^2 + C_{Z_y}^2 + 1} &= N^T * \bar{p} \\ &= [C_{Z_x}, C_{Z_y}, 1] \begin{bmatrix} x_i \\ y_i \\ \bar{z}_i \end{bmatrix} \\ C_{Z_z} \mp d * \sqrt{C_{Z_x}^2 + C_{Z_y}^2 + 1} + C_{Z_x} x_i + C_{Z_y} y_i &= \bar{z}_i \end{aligned} \tag{5.4}$$

Note that the cases for x and y follow this example directly, substituting the correct values for c:

$$C_{X_x} = \left(\frac{D}{A}\right) \qquad C_{Y_x} = \left(\frac{A}{B}\right)$$
$$C_{X_y} = \left(\frac{B}{A}\right) \qquad C_{Y_y} = \left(\frac{D}{B}\right)$$
$$C_{X_z} = \left(\frac{C}{A}\right) \qquad C_{Y_z} = \left(\frac{C}{B}\right)$$

Now that the ideal points $\bar{p} = [x_i, y_i, \bar{z}_i]$ are known, and the distorted points $p = [x_i, y_i, z_i]$ are available, this information can be inserted into Equation 5.3. Note here that Equations 5.1 - 5.3 are all linear in the K-terms. Using all n points from both ideal planes oriented in the xy direction (i.e., both z ideal planes), the K-terms can be solved using least squares. Assume, once again, a least squares system, described by:

$$TS = L$$

The design matrix T contains each of the terms contained in Equations 5.1 - 5.3, the observation vector S holds all the ideal points $\bar{p}$ (i.e., data from both faces of the phantom in the particular dimension), and L is the solution to the system: the values of the K-terms.

$$S = \begin{bmatrix} \bar{p_0} \\ \bar{p_1} \\ \vdots \\ \bar{p_{2n-1}} \end{bmatrix}$$

$$S_i = \bar{z_i} - z_i$$

$$T = \begin{bmatrix} T_{0,0} & T_{0,1} & T_{0,2} & T_{0,3} & T_{0,4} \\ T_{1,0} & T_{1,1} & T_{1,2} & T_{1,3} & T_{1,4} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ T_{2n-1,0} & T_{2n-1,1} & T_{2n-1,2} & T_{2n-1,3} & T_{2n-1,4} \end{bmatrix}$$

$$\begin{aligned} T_{i,0} &= z_i(x_i^2 + y_i^2) \\ T_{i,1} &= z_i z_i^2 = z_i^3 \\ T_{i,2} &= z_i(x_i^2 + y_i^2)z_i^2 = z_i^3(x_i^2 + y_i^2) \\ T_{i,3} &= z_i(x_i^2 + y_i^2)^2 \\ T_{i,4} &= z_i(z_i^4) = z_i^5 \end{aligned} \tag{5.5}$$

$$L = \begin{bmatrix} K_{z0} \\ K_{z1} \\ K_{z2} \\ K_{z3} \\ K_{z4} \end{bmatrix}$$

Solving for L, the K-terms that describe the correction function in the z dimension, is accomplished using least squares (discussed in Appendix B). This process is repeated in the same fashion for the correction parameters K_x and K_y. Once all three sets of correction parameters are obtained, the correction can be applied to ***any*** MR image that originates from the same scanner[1], using Equations 5.1 - 5.3 to solve for the undistorted points $\bar{p} = (\bar{x}_i, \bar{y}_i, \bar{z}_i)$, given the distorted image points $p = (x_i, y_i, z_i)$.

The fact that the correction can be applied to any image that originates from the particular MRI scanner makes this entire correction process very feasible. The properties of the scanner do not change greatly over time, therefore the gradient nonlinearity distortion remains fairly constant. The only time that the characteristics of the distortion *may* change in a MRI scanner is after a bi-annual calibration that ensures proper operation of the scanner. Or, the distortion may change as a result of any other scheduled maintenance or physical modifications. Again, these occurrences are not common, and may occur at most every six months. Therefore, it would be recommended to run the software correction after any of these procedures; but the

[1] This assumes that all scans are performed using the identical configuration: TE, TR, flip angle, enabling/disabling of scanner correction, etc. A deeper discussion on this topic is presented in Chapter 6.

need to run the correction process more often has not been justified. Indeed, the correction process can be run as often as every scan, to ensure the maximum level of accuracy in the correction. However, any change in accuracy in the correction results may not be justifiable, due to the relative consistency of the gradient nonlinearity distortion inherent in each scanner.

5.2 Algorithms

Now that the mathematics and theory behind the distortion estimation and correction stage have been presented, an in-depth algorithm analysis can proceed. The algorithms follow the theory presented above in order to produce the desired result.

5.2.1 Invoking the "Calculate Distortion" Algorithms

- **Script Name:** calculate_distortion
- **Input:**
  - Arrays containing ideal plane coefficients.
  - Arrays containing processed edge data.
- **Output:** Matrix containing parameters of distortion correction model.
- **Syntax:** calculate_distortion()
- **Purpose:** Master script that invokes all distortion calculation steps automatically and sequentially.

Purpose

The distortion calculation step is executed after the plane calculation stage. Once the midplanes and ideal planes are calculated, the software has the necessary data to simulate the ideal shape, size, orientation, and position of the phantom's faces, in the presence of no distortion. The goal of this stage is to use this information to correctly model the distortion, in order to effectively "undistort" the original edge points. Since there are so many steps involved in the actual processing, a master function was used to promote neatness and user-friendliness. This is nothing more than a script that properly calls each function in turn, and gives each function the proper data it needs to perform its duties.

As was the case with both the data preprocessing and plane calculation stages, the distortion calculation stage is highly sequential. The output of one function will feed

directly as input for the next function. With each passing stage, the data undergoes various processing steps, to yield the final result: the coefficients of the distortion correction model.

The algorithms executed at this stage involve packaging the data in various data structures, ensuring the data is arranged properly prior to distortion calculation, and employing linear least squares to solve for the coefficients of the distortion correction model. Depending on the amount of data supplied, the plane calculation stage may take between a few seconds, to a few minutes to complete. Obviously, the system that is running this software will ultimately decide how quickly these tasks will complete. For the record, the machine used to develop this software has the following specifications:

- Intel Pentium M 1.6 GHZ CPU (2 MB L2 Cache)
- 2 GB DDR2-533 MHZ RAM
- 80 GB 5400 RPM Hard Drive

When Matlab is processing the data, the CPU remains pegged at 100%. Matlab memory usage can vary anywhere between 300 MB (when only one study has been loaded, prior to processing), to over 1.5 GB (after processing has completed for 3 studies).

5.2.2 Pack Edge Points

- **Function Name:** pack_edge_points
- **Input:**
  - Study orientation (direction cosines).
  - Slice number of image to be processed.
  - Voxel size of study (mm/pixel).
  - Width of MRI scan (in pixels).
  - Height of MRI scan (in pixels).
  - Range of useful slices.
  - Slice thickness (mm).
  - Number of useful slices.
  - Current edge being processed.
- **Output:** Matrix containing one edge, in mm coordinates.
- **Syntax:** pack_edge_points(orientation, slice_no, voxel_size, scan_width, scan_height, ranges, thickness, number_slices, current_edge)
- **Purpose:**
  - Convert from pixel to mm coordinates.
  - Create right-handed coordinate system.
  - Convert edge points from list storage to matrix storage.

Purpose

The pack_edge_points() function is actually used twice during the distortion correction process. The first time that this function is called is during the preparation of the acquired image data for plane calculation. There, the create_all_midline_data() function invokes pack_edge_points() to do just that: pack the acquired edge data points into matrices, while simultaneously converting from pixel to mm coordinates and creating a regular right-handed coordinate system.

The reason that the pack_edge_points() function needs to be used twice during the distortion correction process is due to the nature of the data being stored at each stage. When the pack_edge_points() function is first called during data preparation for midplane calculation, the data points processed for solitary points is packed into matrices, to enable trivial midpoint calculation. Since the remove_solitary_points() function has ensured that all pairs of edges are of the same length, the matrices constructed at this step also reflect that property. Therefore, the matrices created by pack_edge_points() do not contain *all* the acquired data, and are not suitable for modeling of the distortion at the distortion calculation stage.

When pack_edge_points() is called again during distortion calculation (i.e., this stage), *all* the edge points are packed into matrices. Solitary points are not removed here. Since a simple, straightforward, and reliable method of performing solitary point removal *after* packing data into matrices was not developed, the simple compromise is to use pack_edge_points() twice.

For the full discussion and explanation concerning pack_edge_points, refer to Section 4.3.5.

5.2.3 Create Edge Data Sets

- **Function Name:** create_data_set_edges()
- **Input:** Array containing detected edge points from one edge across an entire study.
- **Output:** Three large vectors containing all data points across that edge in a study, one for each dimension (x, y, z).
- **Syntax:** create_data_set_edges(edge_points)
- **Purpose:** Consolidate all data points across an edge in a study into three large vectors, one for each data set in each dimension (x, y, z).

Purpose

The create_data_set_edges() function is very similar in operation to that of the construct_dataset() function. Like its cousin, the create_data_set_edges() function performs one of the final procedures in preparation for distortion modeling using the acquired edge data. Because of the method that the edge points are originally packed, the data needs to undergo a series of processing steps in order to make distortion modeling possible. So far, the edge points have been sampled and packed into arrays. Next, the edge points were repacked into matrices, with one matrix holding the edge points on one image. Now, the task at hand is to combine all the matrices from each study, into three large vectors: one each for x, y, and z.

When create_data_set_edges() is called by calculate_distortion(), it is given data for one edge across an entire study. The function is slightly different from con-

struct_dataset(), in that the consolidated data is returned as three separate vectors of length n, instead of a $n \times 3$ matrix. This is done so that the data can be later consolidated one more time into large matrices that represent all the edge data for one face of the phantom. This greatly aids in distortion modeling, since the distortion is measured by the shape and curvature of each of the phantom's faces.

Application

The process of assembling the large vectors is as follows:

1. Count the number of midpoints that were calculated for each image in the study.
2. Allocate the memory space for the three vectors that can hold all the edge points.
3. Reading each matrix of edge points, copy the contents of each matrix into each of the large vectors, separating out each three-dimensional point into the appropriate vector.

At this stage, the data is one step closer to being suitably prepared for distortion correction calculation. The data that is stored in these vectors represents the exact appearance of the phantom as the result of the gradient nonlinearity distortion inherent in the scanner. Recall that in order to calculate the distortion correction parameters using least squares, the data must be stored in matrices and vectors, rather than cell arrays.

5.2.4 Create Face Data Sets

- **Function Name:** create_face_data()
- **Input:**
  - Matrix containing data from the phantom's x-faces.
  - Matrix containing data from the phantom's y-faces.
  - Matrix containing data from the phantom's z-faces.
  - Vector containing direction cosines, denotes scan orientation
  - Toggle graphing of data (1 = on, 0 = off)
- **Output:**
  - Cell array containing data for the two x-faces of the phantom (in matrix form).
  - Cell array containing data for the two y-faces of the phantom (in matrix form).
  - Cell array containing data for the two z-faces of the phantom (in matrix form).
- **Syntax:** create_face_data(x_points, y_points, z_points, orientation, graph)
- **Purpose:** Create three cell arrays (one each for x, y, and z), each holding the two faces of the phantom in that particular dimension.

Purpose

The create_face_data() function takes the construct_dataset() and create_data_set_edges() functions one step further. Both of those functions create vectors matrices that house the data points, for use in linear least squares calculations. In calculating the distortion correction parameters, the create_face_data() function combines the matrix data sets into three large cell arrays, one for each dimension. Each cell array, in turn, contains two matrices, each describing one face on the phantom. Utilizing this structure will prove to be very beneficial in calculating the distortion correction.

There are in fact two processes that are occurring at this stage: concatenation of data sets, and packaging of the large data sets for simple organization. The create_face_data() takes the vectors that were packed by create_data_set_edges(), and concatenates them together, being sure to properly combine data from each edge. Concatenation occurs in the following manner:

Data Orientation	**Studies to Concatenate**
X	Axial and Coronal
Y	Axial and Sagittal
Z	Coronal and Sagittal

The technique employed here is exactly the same as that used in cat_data_sets(). The vectors that are created as a result represent the all the data for one face of the phantom, in one dimension. Just as was the case in plane calculation, the number of data points that are processed in this stage is approximately 64,000.

And finally, the six $64,000 \times 3$ matrices are then packed one more time into cell arrays. Each cell array holds two matrices, and therefore contains the pair of faces

of the phantom in a particular dimension. As stated earlier, organizing the data one more time, such that the data can be referenced according to which face of the phantom it corresponds with, greatly simplifies the distortion calculation process.

Recall that there are six faces on the phantom: two faces oriented in x, two faces oriented in y, and two faces oriented in z. Rather than calculate the distortion separately for each face, it is much simpler and more straightforward to combine the faces in each dimension together, and treat the faces as pairs. Performing this step will therefore yield three sets of distortion parameters, describing the total distortion in x, y, and z. It would not make sense to perform two calculations for the distortion for each dimension, and therefore yield six sets of parameters, one for each face of the phantom. This is unnecessary, and is also a sloppy technique. The goal is to get three sets of parameters, not six. Errors can be introduced in the final calculation as a result of averaging or any other method to consolidate the results into three sets. Since the distortion is symmetric, and the phantom is assumed to be as close to the gradient isocenter as physically possible, it is perfectly legal to combine the data from each pair of faces, and use the larger data set to calculate the distortion correction. Essentially, the calculation method that is based on this data packaging technique models the distortion by processing the data in large batches. Each batch of data is oriented in a different dimension, and the parameters that satisfy that model represent the distortion in that particular dimension.

Application

The process of creating the face data sets occurs as follows:

1. Determine orientations of studies provided, to keep track of the data during concatenation.

2. Determine how many data sets are provided. This will decide which data sets need concatenating, if any.

3. Based on the number of data sets provided, perform concatenation. The concatenated data now exists in large matrices, approximately 64,000 elements in size.

4. Combine the concatenated data one more time, into the cell arrays. Return the cell arrays, each of which will contain two faces of the phantom.

The data is finally prepared for distortion correction calculation. The matrices that are contained within the cell arrays now hold data that represent one distorted face on the phantom. By having each face of the phantom separated into individual matrices, organizing the data to calculate the distortion correction parameters becomes very trivial. All that needs to occur is a simple concatenation of the two matrices in each dimension, to use both faces in each dimension to calculate the distortion in that particular dimension. All the preparation that has been performed thus far will pay off in the next stages.

5.2.5 Calculate Theoretical Undistorted Data

- **Function Name:** create_undist_data()
- **Input:**
  - Vector containing the coefficients of the ideal planes, calculated by calculate_planes().
  - Matrix containing the distorted data points from one face of the phantom.
  - Axis face lies on (x, y, or z)
  - Toggle graphing of data (1 = on, 0 = off).
- **Output:**
  - Matrix containing the theoretical undistorted data.
- **Syntax:** create_undist_data(coeffs, face, face_type,graph)
- **Purpose:** Calculate the theoretical undistorted data, which represents the theoretical positions of the phantom's faces.

Purpose

As discussed in Section 5.1.2, the distortion correction is based on the sum of spherical harmonics. Voxels are shifted in three dimensions in the presence of gradient nonlinearity distortion. Therefore, the correction technique needs to be able to shift the voxels in x, y, and z, in order to properly displace the voxels back to their original locations. Tackling this problem straight-on will not yield any success. Undertaking the task of shifting voxels independently in three dimensions requires solving for three

unknown variables: the displacements in x, y, and z. This is impossible to perform, mathematically-speaking, and also considering the information and tools provided.

So how is the three-dimensional displacement calculated for each voxel? Simple: isolate each dimension, calculate the displacement required to move the voxel from its distorted location to its corrected location *in that dimension*, and perform the shift one-by-one in each dimension. This method of divide-and-conquer greatly reduces the complexity of the problem, since now there is only one unknown to be resolved. What was originally a complex problem, is now a simple problem that can be solved trivially.

Applying this divide-and-conquer concept to the data that is currently available reveals how trivial the solution actually is. Recall that there are three midplanes (and therefore, three pairs of ideal planes) available for use at this stage in the calculation. On each of the phantom's distorted faces, the distortion is essentially in the dimension that corresponds to the orientation of that face. For example, for the y faces (oriented in xz), the distortion is essentially all in the y direction. Because of this, the y ideal planes can be used to calculate the theoretical undistorted locations of the distorted y face data. This undistorted y data is then used to determine the K-parameters that describe the distortion correction required in y. The same concepts follow as well for the x and z faces (oriented in yz and xy, respectively), using the x and z ideal planes and distorted faces.

The plane calculation stage (Chapter 4) produced general expressions for the ideal planes. Recall that these ideal planes represent the correct size, shape, orientation, and location of the phantom's faces, in the presence of no distortion. Therefore,

these ideal plane expressions can be used to create theoretical undistorted data sets. These data sets will be arranged in the exact same method as the distorted data, described in previous sections in this chapter. Plotting these data sets will produce a three-dimensional representation of the phantom's undistorted faces.

Since the displacement of each voxel is only being measured in one dimension at a time, the theoretical undistorted data points will contain a combination of distorted and corrected values. To elaborate:

- X data points: $\left[\bar{x}, y, z\right]$
- Y data points: $\left[x, \bar{y}, z\right]$
- Z data points: $\left[x, y, \bar{z}\right]$

where $\bar{x}, \bar{y}, \bar{z}$ represents the theoretical undistorted data points, and x, y, z represents the acquired distorted data points. In terms of the ideal plane expressions, each theoretical undistorted data point is equal to the offset term of the plane expression, which is nearly the same as δ, the distance by which the midplanes were shifted to create the ideal planes (which corresponds to $\frac{1}{2}$ the phantom's physical dimensions). Given the oil phantom's dimensions of approximately 160 mm^3, this value is approximately 80.

Application

Calculating theoretical undistorted data sets is accomplished by the following steps:

1. Initialize matrices that are the same size as the matrices that hold each distorted face data set.

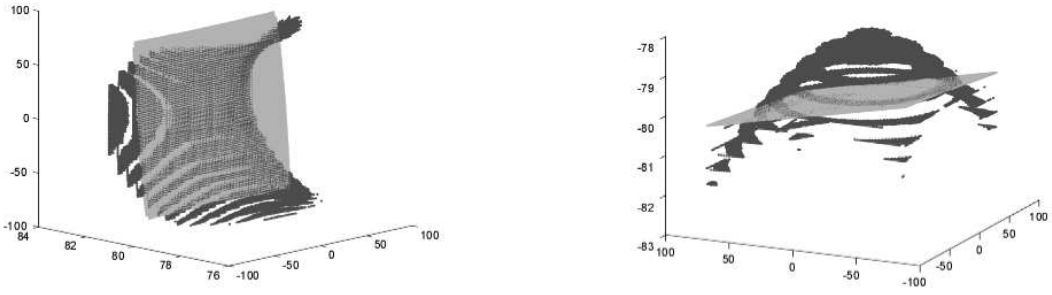


Fig. 5.1: Distorted faces produced by MR scanner (black), and the theoretical undistorted data sets calculated by create_undist_data() (grey). The axis dimensions are pixels, and note the curvature in the distorted faces is exaggerated so that the difference between the two faces is visible.

2. Extract each acquired distorted data point from the face matrix.

3. For each distorted data point, calculate the theoretical undistorted data value for each particular dimension in the following manner:

   (a) For x: $\bar{x_i} = (\frac{D}{A}) + (\frac{B}{A})y_i + (\frac{C}{A})z_i$

   (b) For y: $\bar{y_i} = (\frac{A}{B})x_i + (\frac{D}{B}) + (\frac{C}{B})z_i$

   (c) For z: $\bar{z_i} = (\frac{A}{C})x_i + (\frac{B}{C})y_i + (\frac{D}{C})$

4. Pack each new undistorted data point into the new matrix, and return.

Since this process uses the original distorted data points to calculate the theoretical undistorted data points, the number of theoretical undistorted data points is exactly the same as the number of distorted data points. Plotting both faces on the same graph, as in Figure 5.1, will demonstrate the difference between both data sets.

The create_undist_data() function supplies the last piece of the puzzle, for the distortion correction method. Now that theoretical undistorted locations are known for the voxels in the phantom's faces, all the information is now available to calculate the distortion correction parameters.

5.2.6 Calculate Distortion Correction Model

- **Function Name:** calc_distortion_params()
- **Input:**
  - Matrix containing distorted data from the phantom face that lies on the negative end of the axis
  - Matrix containing distorted data from the phantom face that lies on the positive end of the axis
  - Axis face lies on (x, y, or z)
  - Matrix containing calculated theoretical undistorted data from the phantom face that lies on the negative end of the axis
  - Matrix containing calculated theoretical undistorted data from the phantom face that lies on the positive end of the axis
- **Output:**
  - 5-element vector that contains the k-parameters.
- **Syntax:** calc_distortion_params(face_1, face_2, face_type, alpha_undist_1, alpha_undist_2)
- **Purpose:** Calculate the parameters for the distortion correction model, based on the sum of spherical harmonics.

Purpose

Now the big moment has arrived: time to calculate the distortion correction model, based on the sum of spherical harmonics. Equations 5.1 - 5.3 are the forms used by calc_distortion_params() to calculate the K-parameters, representing the coefficients of the distortion correction model for one dimension[2]. Arranging Equations 5.1 - 5.3 into matrices, for evaluation using a least squares system, yields a system described by:

$$A\chi = B$$

where the design matrix A contains each of the terms contained in Equations 5.1 - 5.3, the observation vector B holds the difference between all the ideal points $\bar{p}$ and the acquired distorted data points p, and χ is the solution to the system: the values of the K-terms. Take note that each pair of phantom faces (distorted and undistorted) are used during this calculation; therefore, given that each face contains n data points, A and B will now contain $2n$ data points:

[2] Keep in mind that this function needs to be run three times, to obtain all three sets of distortion parameters (K_x, K_y, and K_z). This follows from the fact that calculating the distortion requires isolating the distortion in each dimension, and creating a model that describes the distortion independently in X, Y, and Z. Refer to Section 5.2.5 for more information.

$$B = A\chi$$

$$\Downarrow$$

$$\begin{bmatrix} \bar{\alpha_1} - \alpha_1 \\ \vdots \\ \bar{\alpha_{2n}} - \alpha_{2n} \end{bmatrix} = \tag{5.6}$$

$$\begin{bmatrix} \alpha_1(x_1^2 + y_1^2) & \alpha_1 z_1^2 & \alpha_1(x_1^2 + y_1^2)z_1^2 & \alpha_1(x_1^2 + y_1^2)^2 & \alpha_1 z_1^4 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \alpha_{2n}(x_{2n}^2 + y_{2n}^2) & \alpha_{2n} z_{2n}^2 & \alpha_{2n}(x_{2n}^2 + y_{2n}^2)z_{2n}^2 & \alpha_{2n}(x_{2n}^2 + y_{2n}^2)^2 & \alpha_{2n} z_{2n}^4 \end{bmatrix} \begin{bmatrix} K_{\alpha 0} \\ K_{\alpha 1} \\ K_{\alpha 2} \\ K_{\alpha 3} \\ K_{\alpha 4} \end{bmatrix}$$

Because of the arrangement of the system, it is possible to calculate K-parameters that can either distort or correct a data set. The type of parameters that are created by this function is determined simply by the arrangement of the data in the matrices. The arrangement shown above in Equation 5.6 is used to calculate K-parameters that perform distortion correction on a given data set. Reversing the presentation of the data will produce K-parameters that distort a given data set. Providing such a result is accomplished by reversing the presentation of the ideal data points, $\bar{p}$, and the distorted data points, p, as such:

$$\bar{B} = \bar{A}\bar{\chi}$$

$$\Downarrow$$

$$\begin{bmatrix} \alpha_1 - \bar{\alpha_1} \\ \vdots \\ \alpha_{2n} - \bar{\alpha_{2n}} \end{bmatrix} =$$

$$\begin{bmatrix} \bar{\alpha_1}(\bar{x_1}^2 + \bar{y_1}^2) & \bar{\alpha_1}\bar{z_1}^2 & \bar{\alpha_1}(\bar{x_1}^2 + \bar{y_1}^2)\bar{z_1}^2 & \bar{\alpha_1}(\bar{x_1}^2 + \bar{y_1}^2)^2 & \bar{\alpha_1}\bar{z_1}^4 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \bar{\alpha_{2n}}(\bar{x_{2n}}^2 + \bar{y_{2n}}^2) & \bar{\alpha_{2n}}\bar{z_{2n}}^2 & \bar{\alpha_{2n}}(\bar{x_{2n}}^2 + \bar{y_{2n}}^2)\bar{z_{2n}}^2 & \bar{\alpha_{2n}}(\bar{x_{2n}}^2 + \bar{y_{2n}}^2)^2 & \bar{\alpha_{2n}}\bar{z_{2n}}^4 \end{bmatrix} \begin{bmatrix} K_{\bar{\alpha 0}} \\ K_{\bar{\alpha 1}} \\ K_{\bar{\alpha 2}} \\ K_{\bar{\alpha 3}} \\ K_{\bar{\alpha 4}} \end{bmatrix}$$

Application

Solving for the K-parameters in each dimension is very straightforward:

1. Concatenate the two provided faces, to create one large data set, $2n$ in size.
2. Create the B matrix by calculating the difference betweeen the undistorted and distorted data points (the order of the data in the subtraction is determined by what type of parameters are being calculated: distortion, or distortion correction).
3. Populate the A matrix by calculating each term of the spherical harmonics expansion. As shown above, each column in A represents one term in the expansion.
4. Solve for χ using least squares (see Chapter B).

5. Output the K-parameters to text file, using either the write_dist_params() or write_undist_params() functions.

Most of the work involved in this function is simply to pack the data sets that comprise the least squares system. The actual least squares calculation is very trivial in Matlab, thanks to the backslash "\" operator:

$$\chi = A \backslash B$$

Now, χ contains the five K-parameters that describe the distortion in one particular dimension. This function will need to be run three times—once each for x, y, and z—to obtain the entire distortion correction model (or, the entire distortion model, if the goal is to distort a data set, rather than undistort).

As mentioned in Section 5.1.2, once all three sets of correction parameters are obtained, the correction can be applied to ***any*** MR image that originates from the same scanner[3], using Equations 5.1 - 5.3 to solve for the undistorted points $\bar{p} = (\bar{x}_i, \bar{y}_i, \bar{z}_i)$, given the distorted image points $p = (x_i, y_i, z_i)$. The same holds true for the process of distorting a data set.

This is the last step in the software-based MRI gradient nonlinearity distortion correction. The K-parameters are the ultimate goal of this software method. Once the parameters are calculated, they are output to a tab-delimited text file, using the write_undist_params() and write_dist_params() functions. An example file is as follows:

[3] This assumes that all scans are performed using the identical configuration: TE, TR, flip angle, enabling/disabling of scanner correction, etc. A deeper discussion on this topic is presented in Chapter 6.

```
% Parameters for distortion correction model (to undistort data), ba
% the sum of spherical harmonics.  Refer to documentation
% for more information.

% Parameters are output as a 3x5 matrix, each row corresponds
% to one set of parameters:  row 1 = kx, row 2 = ky, row 3 = kz.

% Created on Jul 21, 2006 10:52:44 AM.

-3.30003e-007  6.50794e-006  -2.60814e-010  -6.91441e-011  2.48748e-
8.45944e-008   5.17323e-006  -2.3672e-010   -1.05688e-010  3.27553e-
-2.56371e-006  1.77455e-006  -1.94391e-010  8.20282e-011   -8.30052e
```

Given this file, any software program can easily read in the K-parameters, and perform the distortion or distortion correction methods. The only step that is required to apply the transformation on a data set is to input all the values properly into Equations 5.1, 5.2, and 5.3. All the long, hard, arduous work is now complete; the text files contain all the information necessary to recreate the distortion models, to distort and undistort data sets.

6. RESULTS

6.1 Introduction

Now that the concepts and details of the method have been presented, it is now time to analyze the effectiveness of the distortion correction method. Demonstrating the accuracy of the correction method will be handled in several ways.

Understanding the results requires both quantitative and qualitative analyses. The first section presents a numerical analysis, followed by an analysis of the images produced by the correction method. A basic survey of the numerical accuracy of the correction method will be demonstrated. Total error, variance, and standard deviations are calculated using the oil phantom. These provide an initial glimpse into the accuracy of the software correction method. Applying the correction model to the acquired MR images will provide a qualitative analysis of the correction method. Various corrected MRI scans of the oil phantom are presented, along with corrected edge images, to provide simple side-by-side comparisons.

These numerical and image analyses only serve as a basic overview, and do not serve as an all-conclusive verification of the accuracy of the system. The tests shown here are conducted using the calibration unit, and therefore does not provide total insight to the quality of the correction. Therefore, other independent tests are necessary to confirm the results shown here.

One independent test that provides further insight into the accuracy of the correction method is the utilization of a simulated data set. This data set will represent a perfect experimental setup:

- Perfectly sized phantom (160 mm^3).
- Perfectly centered phantom (with respect to gradient isocenter).
- Perfectly arranged data.
- Noiseless data (i.e., no air bubbles, no weak edges, no artifacts, etc.).

The purpose of using this perfect data set is to show the accuracy of the software correction method, without considering the various sources of experimental error. The software assumes that the phantom is perfectly centered with respect to the gradient isocenter, therefore making the distortion symmetric about the center of the images. To test the effectiveness of the software, the perfect phantom is created, and then distorted using a pre-defined set of distortion parameters. The distorted data set created here is therefore a perfect distorted data set, since it is based on a perfect phantom. Then, the software operates in the normal fashion, calculating the distortion correction model using the distorted data. The corrected data set is then compared to the original perfect data set. The differences between the two will provide insight to the accuracy of the method. In an ideal, perfect system, the two data sets will be exactly the same, since the correction method is essentially the inverse of the original applied distortion. However, due to the fact that the distortion correction method is based on a truncated expression of the spherical harmonics expansion (Equations 2.8 - 2.10), and not the full infinite series (Equation 1.28),

the correction method will not be perfect. Rather, the correction method provides the *best approximation* of the infinite spherical harmonics expansion. Therefore, this survey will analyze the quality of that approximation.

Additionally, the simulation includes an analysis on how improper phantom positioning affects the accuracy of the correction results. Since it is impossible to center the phantom *perfectly* with respect to the gradient isocenter, "improper" phantom centering will always be an issue. Quantifying these errors using the simulated data set will identify how important phantom centering is, and how this factor affects the overall accuracy of the correction method.

The second independent test is the most important step by far in proving the accuracy of the distortion correction method: localizing known fiducial markers. The Lucy phantom and stereotactic frame are used for this step, shown in Figure 6.31. The accuracy of target localization is accomplished by calculating the corrected positions of the Lucy's markers in an image, and comparing those positions to the actual measured positions on the Lucy phantom itself. The discrepancy between the two measurements will reveal the accuracy of the method, and provides a strong ***independent*** verification of the quality of the method. Achieving errors that are submillimeter at this stage will provide enough confidence to deploy this software system in functional proton radiosurgery.

The next section investigates the robustness of the software method. The software must be able to have the flexibility to accommodate various scanning configurations. Investigating the effectiveness of the distortion correction method based on the scan configuration is a very key step. Recall that the MRI technician has the option to

apply various image filtering techniques to the MR images during scanning (Section 1.8.1). These filtering techniques are designed to *reduce* the amount of visible distortion, rather than actually correct for distortion. Since it is common practice to acquire images using various filtering techniques, it is very important to present an analysis of the effectiveness of the software with filtered and unfiltered images. The software needs to be able to work under *all* conditions, regardless of the settings specified by the MRI technician.

Finally, the performance of the correction method is a very important aspect of its operation. A performance analysis is provided, to demonstrate the practicality of the method. Since the correction may be calculated often, it is best to provide insight as to how much time is required to calculate the coefficients of the distortion correction model. The analysis includes benchmarks across various platforms. The test machines used feature different hardware configurations, to help identify which hardware components best improve the performance of the software (CPU, memory, hard drive, etc.).

The first step in analyzing the effectiveness of the distortion correction method presented in this work is to perform an in-depth numerical analysis. The accuracy of the software-based MRI gradient nonlinearity distortion correction is determined by the results of error analysis (see Appendix A). Five MRI studies were performed to provide data for the distortion correction software, each denoted by the date they were acquired. With each new study performed, improvements were made to the experimental method, in order to achieve higher quality data sets, and therefore more accurate correction results. The software was tested on all five studies, to help identify sources of systematic error and improve the accuracy of the system.

Determining the position of the phantom with respect to gradient isocenter was accomplished by examining the position of the phantom in the images. The assumption made here is that the gradient isocenter exists at the center of each image. Therefore, in a 512 x 512 pixel image, the gradient isocenter would be located at pixel $[256.5, 256.5]$. Calculating the center offset is accomplished by the get_center_offset() function, presented in Section 3.3.12. Calculating the center of the image of the phantom and comparing it to pixel $[256.5, 256.5]$ determines the displacement of the phantom from gradient isocenter. The goal of phantom placement is to position the phantom to within one millimeter of gradient isocenter in all three dimensions.

Over the course of the five studies, modifications were made to the experimental setup, to improve the accuracy of the correction results and refine the consistency of the setup procedures. A detailed description of each setup is presented here. It is important to develop an easy, consistent method of experimental setup that can be

repeated with accuracy and confidence. Since this correction method will be employed in the medical field, it is vital that the setup can be repeated easily, and distortion correction results can be obtained with consistency.

6.2.1 7-17-05 Study

Several studies were performed prior to development of the software for this project. The data acquired from these studies were of relatively low quality. One of these studies was unfortunate enough to be performed when the phantom was leaking oil. Not only did the leakage cause corruption of the data, but the mess left by the phantom was difficult to clean up. Oil leakage is definitely not conducive to a user-friendly experimental setup! Ironically, this study proved to be very useful in developing and testing the bubble detection and removal algorithms presented in Section 3.3.7. The other studies, however, were not useful for software development. Either the images were of too low quality, the resolution was too low, or the phantom was not placed correctly in the scanner.

Because of the shortcomings of earlier studies, a fresh set of data was necessary in order to properly develop the software. The first study performed for this project was acquired on July 17, 2005. The purpose of this study was very simple: to acquire a data set for use in developing the software package, and to establish a groundwork for future improvements in the experimental method. Because of this, the setup procedure was very simple.

The acquisition parameters were as follows:

- TR: 1900 ms
- TE: 2.7 ms
- Flip Angle: 20 ms
- Field of View (FOV): 200 mm^2

- Matrix: 512 × 512
- Voxel Size: 0.391 × 0.391 × 2.000 mm
- 1 Slab, 104 partitions (slices) per orientation
- Sequences: axial, coronal
- All scanner filters turned off

At this point in time, the phantom was not properly centered in the scanner. Indeed phantom centering is an issue. However, the point of this study was to merely obtain data with which to develop the software. Later studies would address the issue of phantom centering.

Since the phantom was not centered properly in this study, this trial provided real data that would later quantify the sensitivity of the distortion correction method to phantom centering. Since the software assumes that the distortion is symmetric about the gradient isocenter, it was necessary to determine the severity of the errors that result from correcting data that deviates from this assumption.

Midplane	**Plane Coefficients**
X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{1.1254}, -0.0095131, 0.0014519]$
Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [0.0056433, \mathbf{-4.3254}, 0.0014944]$
Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [0.0021333, -0.0059547, \mathbf{-12.4474}]$

Tab. 6.1: Plane coefficients of midplane fitting in 7-17-05 study.

Numerical Results

Analyzing the bolded values in the plane fitting results in Tables 6.1 and 6.2 reveals the extent of the phantom's offcentering in the scanner. A properly centered phantom

Plane	Plane Coefficients
-X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{-78.6283}, -0.0095131, 0.0014519]$
+X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{80.879}, -0.0095131, 0.0014519]$
-Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [0.0056433, \mathbf{-84.1768}, 0.0014944]$
+Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [0.0056433, \mathbf{75.5259}, 0.0014944]$
-Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [0.0021333, -0.0059547, \mathbf{-91.504}]$
+Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [0.0021333, -0.0059547, \mathbf{66.6092}]$

Tab. 6.2: Plane coefficients of ideal planes fitting in 7-17-05 study.

would result in midplane coefficients that are approximately zero, and offset terms in the ideal plane equations approximately ± 80, given that the phantom is a 160 mm cube. These results show that the phantom is offcentered by approximately 1.1 mm in x, 4.3 mm in y, and 12.4 mm in z. These are quite apparent when observing the image of the phantom in the slices, see Section 6.3.

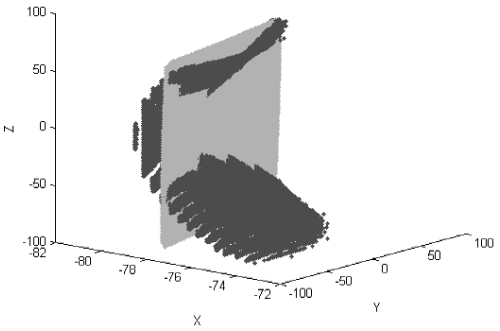
The great offset in z is the result of the shape of the phantom's foot. The size and shape of the foot prevents proper centering when the phantom is placed in the headcoil of the scanner. Below the headcoil is an inset rectangular tray, in which the foot rests. The foot is pushed up against the edge of the tray, to minimize the offset in z. However, the foot is too large, and the minimum offset achievable is 12 mm.

The results obtained here for the distortion correction were not very important at the time. As stated earlier, the primary purpose of this study was to provide real data for use in developing the software package. Several of the functions in the data preprocessing stage, discussed in Chapter 3, were developed using this data set. In fact, the entire distortion correction method was not tested with the 7-17-05 study until after software development was complete. The presentation of the distortion correction results of this study are merely for reference, and provide insight into

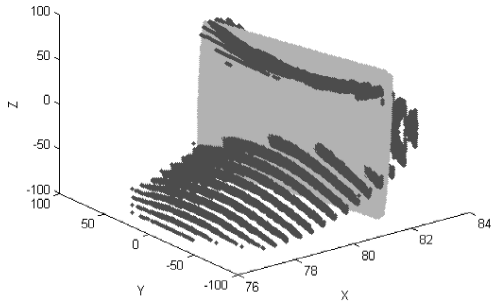
# Data Points Used (X)	31913
# Data Points Used (Y)	31816
# Data Points Used (Z)	29623
Total Residuals (Error) (X) (mm)	18.0845
Total Residuals (Error) (Y) (mm)	26.6412
Total Residuals (Error) (Z) (mm)	33.2983
MSE (Variance) (X) (mm)	0.010248
MSE (Variance) (Y) (mm)	0.022308
MSE (Variance) (Z) (mm)	0.03743
RMS (Standard Deviation) (X) (mm)	0.10123
RMS (Standard Deviation) (Y) (mm)	0.14936
RMS (Standard Deviation) (Z) (mm)	0.19347

Tab. 6.3: Errors calculated from plane fitting results from 7-17-05 study.

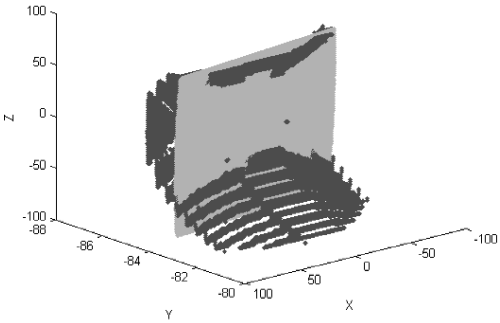
the effects of several factors inherent in the studies, primarily the issue of phantom centering. Improvements to the experimental procedure would be made based on the experienced gained in this study.



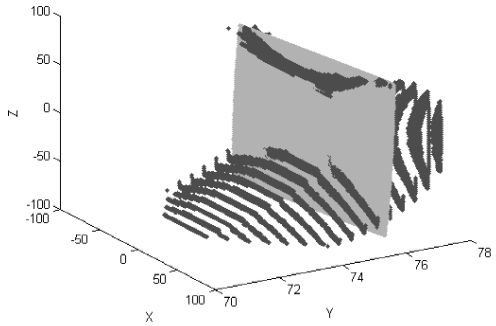
(a) -x face.



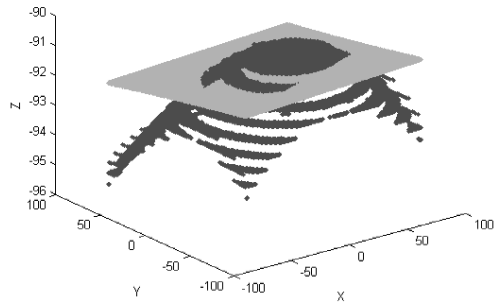
(b) +x face.



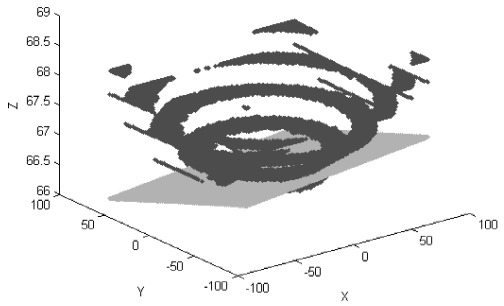
(c) -Y face.



(d) +Y face.



(e) -Z face.

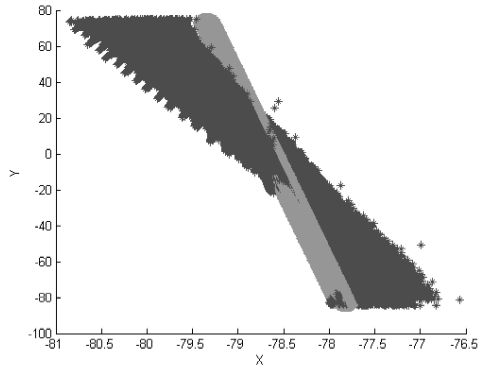


(f) +Z face.

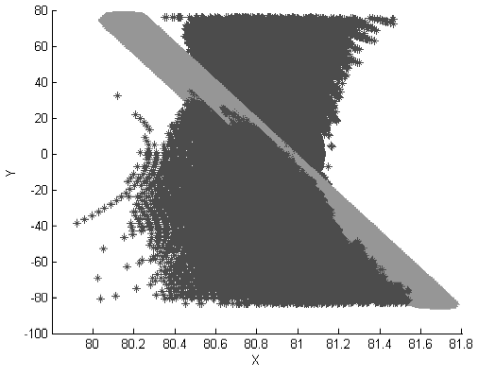
Fig. 6.1: Distorted data points (black) from the 7-17-05 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm.

# Data Points Used (-X)	63389
# Data Points Used (+X)	63387
# Data Points Used (-Y)	31894
# Data Points Used (+Y)	32305
# Data Points Used (-Z)	29623
# Data Points Used (+Z)	30544
Total Residuals (Error) (-X) (mm)	104.4025
Total Residuals (Error) (+X) (mm)	101.3568
Total Residuals (Error) (-Y) (mm)	41.6419
Total Residuals (Error) (+Y) (mm)	41.3696
Total Residuals (Error) (-Z) (mm)	71.4127
Total Residuals (Error) (+Z) (mm)	62.2361
MSE (Variance) (-X) (mm)	0.17195
MSE (Variance) (+X) (mm)	0.16207
MSE (Variance) (-Y) (mm)	0.054369
MSE (Variance) (+Y) (mm)	0.052978
MSE (Variance) (-Z) (mm)	0.17216
MSE (Variance) (+Z) (mm)	0.12681
RMS (Standard Deviation) (-X) (mm)	0.41467
RMS (Standard Deviation) (+X) (mm)	0.40258
RMS (Standard Deviation) (-Y) (mm)	0.23317
RMS (Standard Deviation) (+Y) (mm)	0.23017
RMS (Standard Deviation) (-Z) (mm)	0.41492
RMS (Standard Deviation) (+Z) (mm)	0.35611

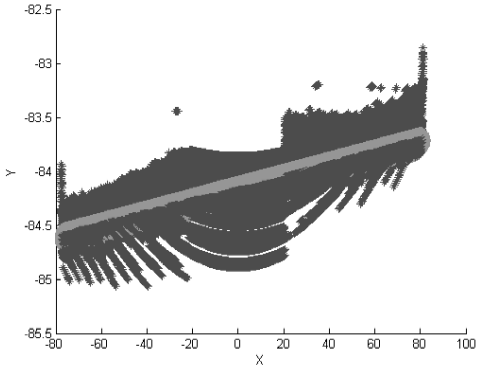
Tab. 6.4: Errors calculated from distortion correction results from 7-17-05 study.



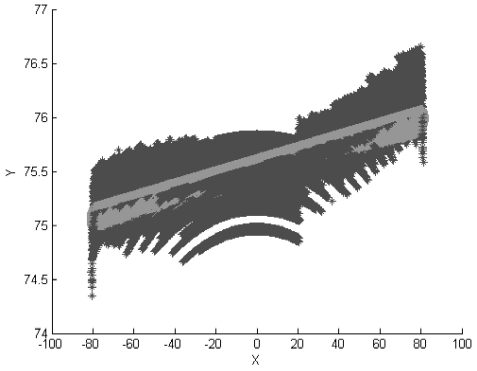
(a) -x face.



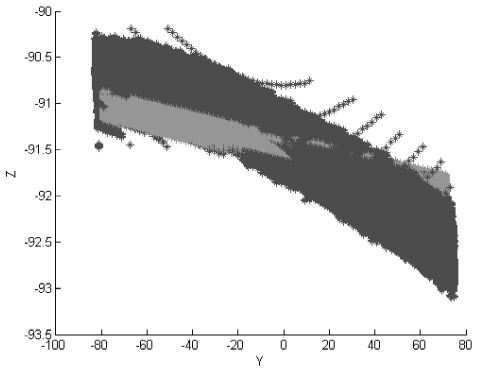
(b) +x face.



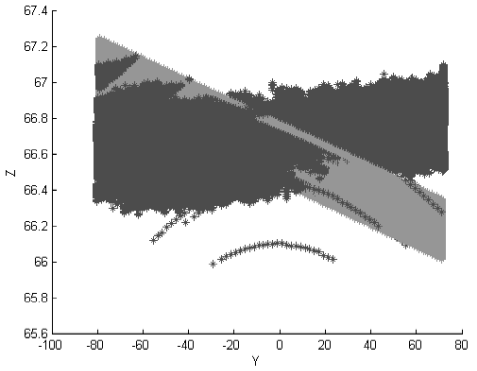
(c) -Y face.



(d) +Y face.



(e) -Z face.



(f) +Z face.

Fig. 6.2: Corrected data points (black) from the 7-17-05 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm.

Characteristics of the Data Set

Visualizing the data set acquired from the 7-17-05 study provides great insight into the characteristics of the study. The quality of the experimental setup is directly reflected in the appearance of the data set. Plotting both the distorted and corrected data points against the theoretical undistorted data points for all six phantom faces yields the three-dimensional graphs shown in Figures 6.1 and 6.2.

Analyzing the orientation of the distorted data points with the theoretical undistorted data points reveals the extent of the offcentering of the phantom in the MRI scanner. The implications of phantom centering are clearly demonstrated here in the form of asymmetric distortion. The distorted faces of the phantom appear quite lopsided, rather than proportionate. The farther away from gradient isocenter the phantom is placed (i.e., the farther offcentered the phantom is positioned), the greater the asymmetry in the gradient nonlinearity distortion. Recall that the software assumes that the distortion inherent to the data set is symmetric. Therefore, asymmetric distortion will reduce the accuracy of the distortion correction. The large errors observed in the numerical results in Tables 6.3 and 6.4 are directly attributed to the asymmetric distortion. Observing the location of the data in Figure 6.1 also indicates the position of the phantom in the scanner. Ideally, the positive and negative faces of the phantom should be equidistant from the origin in each dimension. Unfortunately, this trend is not demonstrated in this data set.

The corrected data points in Figure 6.2 also suffer because of the asymmetry in the distortion. Observing the plots of the corrected data points against the theoretical undistorted data points further demonstrates the quality of the correction. Ideally,

the corrected data points should be as close to the theoretical data points as possible. The farther away the corrected data points lie from the theoretical undistorted points, the larger the overall error in the correction, and the lower the accuracy of the correction. The spread of the corrected data points in this study is quite large. A higher quality distortion correction would result in corrected data points that lie closer to the theoretical undistorted points than demonstrated here. Because of the large differences between the theoretical and actual undistorted points, applying this calculated distortion correction model to other image sets is not recommended. The less-than-ideal experimental setup created scenarios that deviated from the assumed data model. This caused the distortion correction model calculated from the 7-17-05 study to be a poor representation of the actual gradient nonlinearity distortion of the MRI scanner.

6.2.2 10-2-05 Study

The study performed on October 2, 2005 had the goal of further investigating the issue of phantom centering. Additionally, this study was unique in that it also included a sagittal acquisition. The reason for including three scan orientations was to maximize the amount of data available to the software for correction. The acquisition parameters were as follows:

- TR: 2010 ms
- TE: 2.7 ms
- Flip Angle: 20 ms
- Field of View (FOV): 200 mm^2
- Matrix: 512 $\times$ 512
- Voxel Size: 0.391 $\times$ 0.391 $\times$ 2.000 mm
- 1 Slab, 104 partitions (slices) per orientation
- Sequences: axial, coronal, sagittal
- All scanner filters turned off

Again, the phantom was not properly centered in the scanner. The information gathered from this study would later be useful in modifying the phantom to accommodate the shape of the headcoil, to allow for freedom of movement in positioning the phantom.

Prior to scanning, the phantom was centered as best as possible using the MRI scanner's laser sight. The laser sight paints crosshairs on the headcoil (and the phantom), to align the gantry to the gradient isocenter. Again, this only aided the centering problem in two dimensions; the centering in z was still a problem, due to the size and shape of the phantom's foot.

Since the phantom was not centered properly in this study, this trial provided additional real data that would later quantify the sensitivity of the distortion correction method to phantom centering. Combining the knowledge gained from this study and the 7-17-05 study would provide more insight as to the sensitivity of the correction technique on data that does not feature symmetry in the distortion.

Midplane	**Plane Coefficients**
X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{1.7547}, 0.00051889, 0.0060593]$
Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [-0.0041635, \mathbf{-1.0045}, 0.0034399]$
Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [-0.0023869, -0.0067313, \mathbf{-13.2416}]$

Tab. 6.5: Plane coefficients of midplane fitting in 10-2-05 study.

Midplane	**Plane Coefficients**
-X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{-77.9968}, 0.00051889, 0.0060593]$
+X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{81.5061}, 0.00051889, 0.0060593]$
-Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [-0.0041635, \mathbf{-80.8557}, 0.0034399]$
+Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [-0.0041635, \mathbf{78.8466}, 0.0034399]$
-Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [-0.0023869, -0.0067313, \mathbf{-92.2986}]$
+Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [-0.0023869, -0.0067313, \mathbf{65.8155}]$

Tab. 6.6: Plane coefficients of ideal planes fitting in 10-2-05 study.

# Data Points Used (X)	62271
# Data Points Used (Y)	62163
# Data Points Used (Z)	58601
Total Residuals (Error) (X) (mm)	26.0861
Total Residuals (Error) (Y) (mm)	29.5959
Total Residuals (Error) (Z) (mm)	45.1725
MSE (Variance) (X) (mm)	0.010928
MSE (Variance) (Y) (mm)	0.014091
MSE (Variance) (Z) (mm)	0.034821
RMS (Standard Deviation) (X) (mm)	0.10454
RMS (Standard Deviation) (Y) (mm)	0.1187
RMS (Standard Deviation) (Z) (mm)	0.1866

Tab. 6.7: Errors calculated from plane fitting results from 10-2-05 study.

Numerical Results

Analyzing the bolded values in the plane fitting results in Tables 6.5 and 6.6 indicate the extent of the phantom's offcentering in the scanner. Since scans were performed in all three orientations, the 10-2-05 study provides greater insight to the software's sensitivity to phantom centering. Recall that a properly centered phantom would result in midplane coefficients that are approximately zero, and offset terms in the ideal plane equations approximately ± 80, given that the phantom is a 160 mm cube. These results show that the phantom is offcentered by amounts similar to those observed in the 7-17-05 study: approximately 1.7 mm in x, 1.0 mm in y, and 13.2 mm in z. The effects of the great displacements in the phantom centering are quite apparent when observing the image of the phantom in the slices, see Section 6.3.

Utilizing the scanner's laser sight helped align the phantom to within roughly 1 mm in x and y. This is very good, considering that the alignment was done merely by eyeballing; there were no discrete markings on the phantom that could be used accurately or confidently for precision measurement. The only problem is the centering in z, which cannot be remedied without modifying the phantom's foot.

The standard deviations of the correction results (Table 6.8) are smaller for all the faces compared to the 7-17-05 study. What is interesting to note is that the standard deviations shown here are actually quite acceptable, correlating to an accuracy of 0.39 - 0.72 mm, despite the great offcentering in z. These measurements indicate that this correction is easily capable of submillimeter accuracy.

The difference in standard deviations between the 7-17-05 and 10-2-05 studies could be the result of several factors:

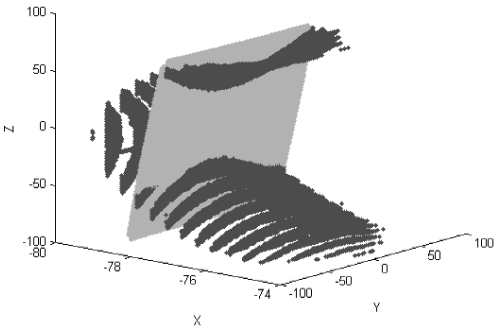
- Three scans (axial, coronal, sagittal) were used in this study, rather than just two (axial, coronal), thereby providing more data for calculation.
- The position of the phantom in the 10-2-05 study was closer to isocenter than in the 7-17-05 study.
- TR was changed from 1900 ms in the 7-17-05 study to 2010 ms in the 10-2-05 study.

Acquiring axial, coronal, and sagittal sequences provided more data for the distortion correction calculations. By providing only two studies, two of the dimensions are represented twice in the studies, while the third is only represented in one of the studies. In the case of the 7-17-05 study, the axial study provided x and y data,

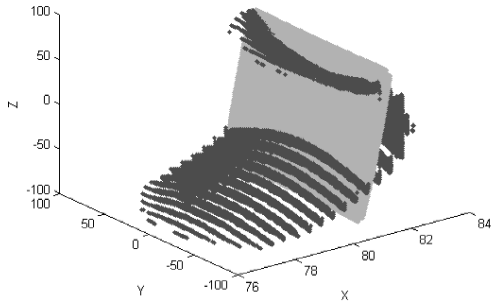
and the coronal study provided x and z data. Because of this arrangement, there were half as many data points representing y as there were x and z. This limitation resulted in a lower quality fit for the K_y parameters, since there were half as many points used in the least squares fits, which then affected all the correction results.

Taking the extra step to center the phantom in x and y provided a more ideal scenario, in terms of providing data with more symmetrical symmetry. Indeed the centering was still very far off in z (over 13 mm), the centering in x and y was much closer to isocenter in the 10-2-05 study than in the 7-17-05 study. Comparing results from both of these studies shows that taking the extra time to line up the phantom with the scanner's laser sight paid off in terms of improved accuracy.

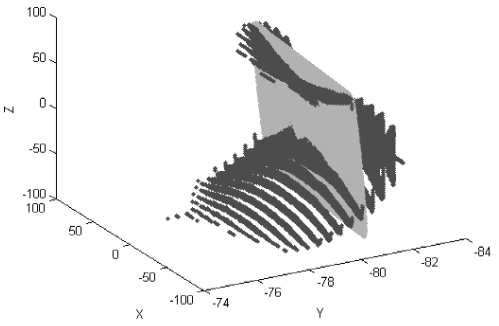
Changing the scan configuration (i.e., the TR) affected the appearance of the phantom in the images. Recall in Section 1.5 that it is possible to highlight the various regions in the scanned object using different weighting of the signals (i.e., various values for TE and TR). This change in TR produced different intensities in the images from the 7-17-05 and 10-2-05 studies, most particularly, the intensity gradients in the 10-2-05 studies were much higher than those in the 7-17-05 study. This may have improved the quality of the detected edges in the 10-2-05 images, thereby providing more accurate data for the correction software to use during calculations. Although there was an apparent difference in the image intensities, the impact that this might have had on data quality is probably very small.



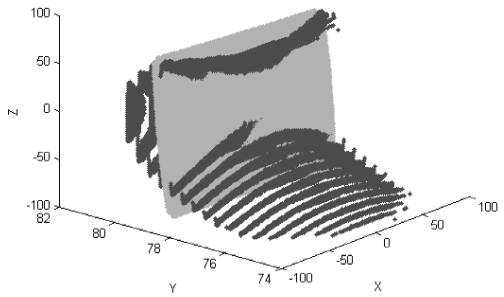
(a) -X face.



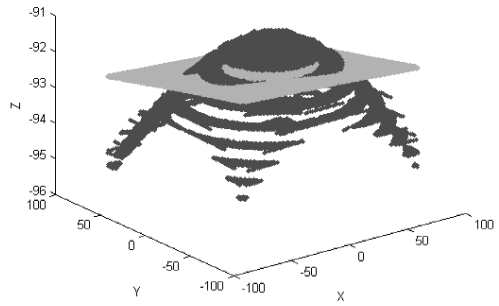
(b) +X face.



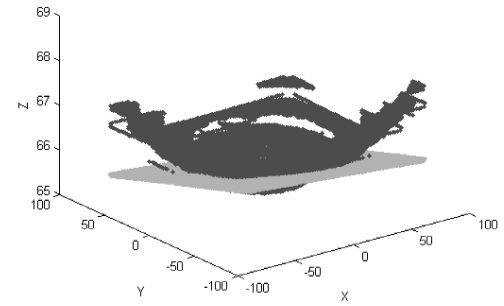
(c) -Y face.



(d) +Y face.



(e) -Z face.

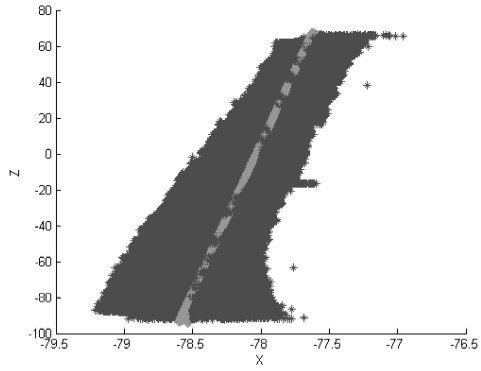


(f) +Z face.

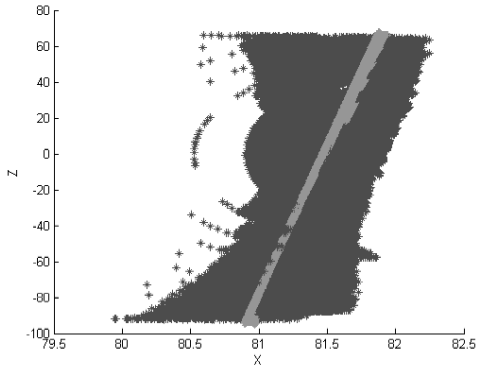
Fig. 6.3: Distorted data points (black) from the 10-2-05 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm.

# Data Points Used (-X)	62443
# Data Points Used (+X)	62441
# Data Points Used (-Y)	62362
# Data Points Used (+Y)	62572
# Data Points Used (-Z)	58601
# Data Points Used (+Z)	60297
Total Residuals (Error) (-X) (mm)	58.8591
Total Residuals (Error) (+X) (mm)	61.1452
Total Residuals (Error) (-Y) (mm)	37.2034
Total Residuals (Error) (+Y) (mm)	40.7898
Total Residuals (Error) (-Z) (mm)	39.9406
Total Residuals (Error) (+Z) (mm)	33.7984
MSE (Variance) (-X) (mm)	0.055481
MSE (Variance) (+X) (mm)	0.059876
MSE (Variance) (-Y) (mm)	0.022195
MSE (Variance) (+Y) (mm)	0.02659
MSE (Variance) (-Z) (mm)	0.027222
MSE (Variance) (+Z) (mm)	0.018945
RMS (Standard Deviation) (-X) (mm)	0.23554
RMS (Standard Deviation) (+X) (mm)	0.2447
RMS (Standard Deviation) (-Y) (mm)	0.14898
RMS (Standard Deviation) (+Y) (mm)	0.16307
RMS (Standard Deviation) (-Z) (mm)	0.16499
RMS (Standard Deviation) (+Z) (mm)	0.13764

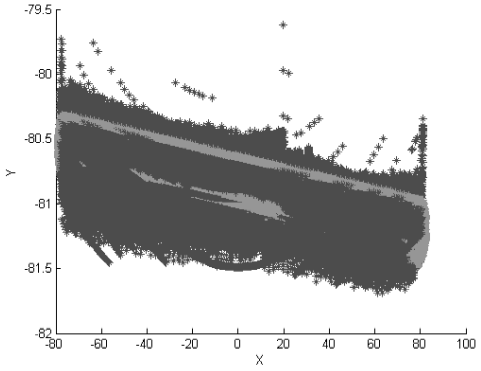
Tab. 6.8: Errors calculated from distortion correction results from 10-2-05 study.



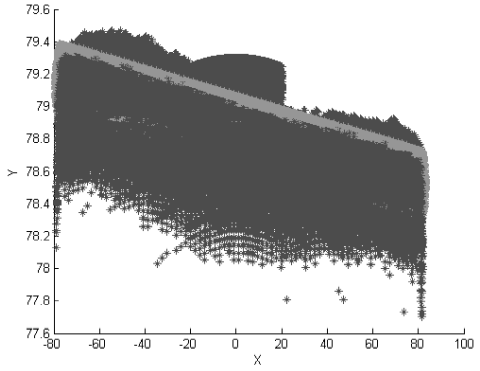
(a) -X face.



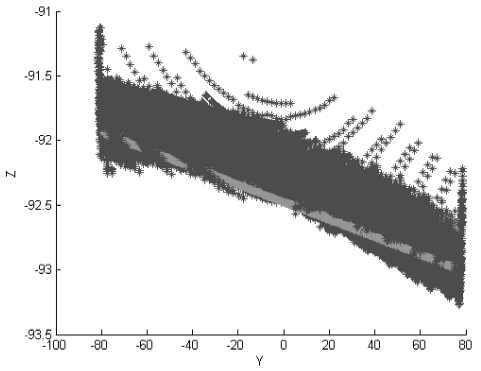
(b) +X face.



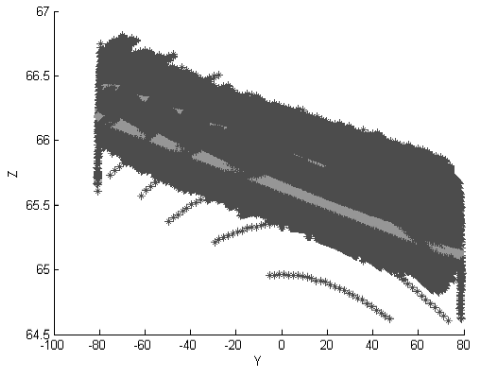
(c) -Y face.



(d) +Y face.



(e) -Z face.



(f) +Z face.

Fig. 6.4: Corrected data points (black) from the 10-2-05 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm.

Characteristics of the Data Set

Visualizing the data set acquired from the 10-2-05 study provides great insight into the characteristics of the study. The quality of the experimental setup is directly reflected in the appearance of the data set. Plotting the distorted and corrected data points against the theoretical undistorted data points for all six phantom faces yields the three-dimensional graphs shown in Figures 6.3 and 6.4.

Analyzing the orientation of the distorted data points with the theoretical undistorted data points reveals the extent of the offcentering of the phantom in the MRI scanner. As was the case with the 7-17-05 study, the implications of phantom centering are clearly demonstrated here in the form of asymmetric distortion. Like the 7-17-05 study, the 10-2-05 study was poorly centered, and the asymmetric distortion in the distorted faces of the phantom is quite apparent. The distorted faces of the phantom appear quite lopsided, rather than proportionate. The farther away from gradient isocenter the phantom is placed (i.e., the farther offcentered the phantom is positioned), the greater the asymmetry in the gradient nonlinearity distortion. Recall that the software assumes that the distortion inherent to the data set is symmetric. Therefore, asymmetric distortion will reduce the accuracy of the distortion correction. The large errors observed in the numerical results of the 10-2-05 study in Tables 6.7 and 6.8 are directly attributed to the asymmetric distortion. Observing the location of the data in Figure 6.3 also indicates the position of the phantom in the scanner. Ideally, the positive and negative faces of the phantom should be equidistant from the origin in each dimension. Unfortunately, the 10-2-05 study is similar to the 7-17-05 study in that this trend is not demonstrated in the data.

The corrected data points in Figure 6.4 also suffer because of the asymmetry in the distortion. Observing the plots of the corrected data points against the theoretical undistorted data points further demonstrates the quality of the correction calculated from the 10-2-05 study. Ideally, the corrected data points should be as close to the theoretical data points as possible. The farther away the corrected data points lie from the theoretical undistorted points, the larger the overall error in the correction, and the lower the accuracy of the correction. The spread of the corrected data points in this study is slightly smaller than that of the 7-17-05 study, but are still quite large. A higher quality distortion correction would result in corrected data points that lie closer to the theoretical undistorted points than demonstrated here. Because of the large differences between the theoretical and actual undistorted points, applying this calculated distortion correction model to other image sets is not recommended. The less-than-ideal experimental setup created scenarios that deviated from the assumed data model. This caused the distortion correction model calculated from the 10-2-05 study to be a unsatisfactory representation of the actual gradient nonlinearity distortion of the MRI scanner.

6.2.3 11-6-05 Study

After confirming the improper size of the phantom's foot, the foot was modified to allow for more freedom of movement in the scanner headcoil. Recall that the observed offset in z was about 12-13 mm. The foot was shortened by 15 mm in the front, to provide more flexibility in phantom centering. Based on the centering results from the 7-17-05 and 10-2-05 studies, it was determined that the phantom needed to be

adjusted in all three dimensions with the shims. The phantom would need to be raised 1.35 mm off the headcoil with shims. To further adjust the centering in z, the phantom was shimmed by 2.1 mm in the front. Due to the size of the phantom with respect to the headcoil, the phantom was very nearly centered in x, therefore no shimming was necessary in x.

To test the effectiveness of the shims, another study was performed on November 6, 2005. The scan parameters were identical to those of the 10-2-05 study.

- TR: 2010 ms
- TE: 2.7 ms
- Flip Angle: 20 ms
- Field of View (FOV): 200 mm^2
- Matrix: 512 $\times$ 512
- Voxel Size: 0.391 $\times$ 0.391 $\times$ 2.000 mm
- 1 Slab, 104 partitions (slices) per orientation
- Sequences: axial, coronal, sagittal
- All scanner filters turned off

After adjusting the phantom centering with the shims, the MRI scanner's laser sight was used to fine tune phantom centering. The primary goal of this study was to finally center the phantom to within 1 mm of gradient isocenter in all three dimensions, to observe the accuracy of the distortion correction method using properly centered phantom data.

Midplane	Plane Coefficients
X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{0.68583}, -0.00427, 0.0063581]$
Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [0.00031188, \mathbf{0.87814}, 0.0037484]$
Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [-0.0025594, -0.0068872, \mathbf{-0.16967}]$

Tab. 6.9: Plane coefficients of midplane fitting in 11-6-05 study.

Midplane	Plane Coefficients
-X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{-79.0665}, -0.00427, 0.0063581]$
+X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{80.4382}, -0.00427, 0.0063581]$
-Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [0.00031188, \mathbf{-78.9724}, 0.0037484]$
+Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [0.00031188, \mathbf{80.7287}, 0.0037484]$
-Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [-0.0025594, -0.0068872, \mathbf{-79.2268}]$
+Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [-0.0025594, -0.0068872, \mathbf{78.8875}]$

Tab. 6.10: Plane coefficients of ideal planes fitting in 11-6-05 study.

Numerical Results

The 11-6-05 study provides the first data set based on a phantom centered in an MRI scanner to within 1 mm from isocenter. The data results from the 7-17-05 and 10-2-05 studies, which represent improper centering of the phantom, can be compared to the results obtained here. The results from these three studies provide further insight into the sensitivity of the software to improper phantom centering.

For the first time, the phantom was centered to within 1 mm of gradient isocenter. Observing the bolded values in the midplane and ideal plane fitting results in Tables 6.9 and 6.10 shows that the greatest displacement of the phantom in terms of centering was only about 0.9 mm. With this setup, the distortion present in the data sets used during these calculations is about as symmetric as possible, therefore the data is as close to the ideal setup as possible.

# Data Points Used (X)	62776
# Data Points Used (Y)	62980
# Data Points Used (Z)	60463
Total Residuals (Error) (X) (mm)	26.5725
Total Residuals (Error) (Y) (mm)	27.2311
Total Residuals (Error) (Z) (mm)	22.122
MSE (Variance) (X) (mm)	0.011248
MSE (Variance) (Y) (mm)	0.011774
MSE (Variance) (Z) (mm)	0.0080939
RMS (Standard Deviation) (X) (mm)	0.10606
RMS (Standard Deviation) (Y) (mm)	0.10851
RMS (Standard Deviation) (Z) (mm)	0.089966

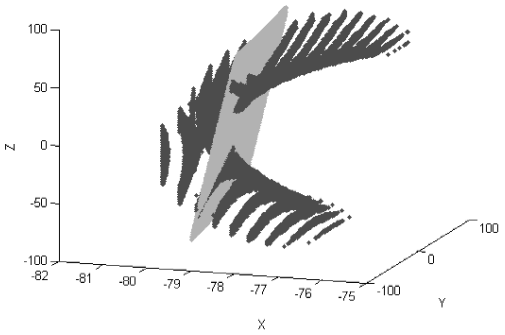
Tab. 6.11: Errors calculated from plane fitting results from 11-6-05 study.

Even with this very good centering, the standard deviations of the correction results shown in Table 6.12 are still slightly higher than what may be expected, most particularly in x. The standard deviations of the x faces are nearly twice as large as those in other faces. Even more interesting is the fact that the results of the 10-2-05 study (Table 6.12) are very close to the results obtained here. And in both studies, the standard deviations obtained are quite acceptable, correlating to an accuracy of better than 0.78 mm, despite the discrepancy in x in the 11-6-05 study. Both of these measurements indicate that this correction is easily capable of submillimeter accuracy.

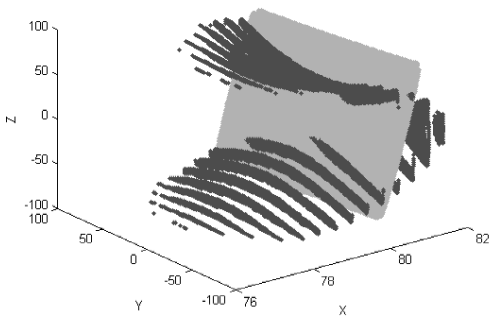
It would be expected that the correction results be very similar for all three faces in this study. The phantom was positioned *very* close to gradient isocenter in this study,

therefore the distortion would be expected to be very symmetric. But the results do not seem to reflect that property. Even more peculiar is the fact that the errors are not consistent across all dimensions: only the correction in x shows the higher errors. Although the y and z errors are higher than expected ($\frac{1}{3}$ of a pixel, or approximately 0.13 mm), they are still very good indeed. Despite the higher errors experienced in x, the errors are still small enough to show an accuracy of approximately 0.75 mm, which translates to about two pixels. Indeed, the correction results here do show the capability of the distortion correction technique to produce results with submillimeter accuracy.

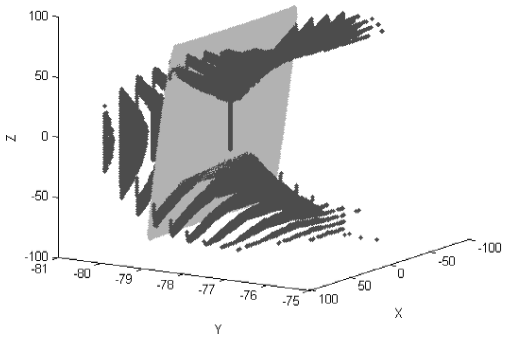
The discrepancies in these results prompted for an in-depth investigation into the cause of the increased error. A formal and thorough discussion of the investigation into this discrepancy is presented in Section 7.1.2. The results obtained here would prove to be very useful for all subsequent studies performed.



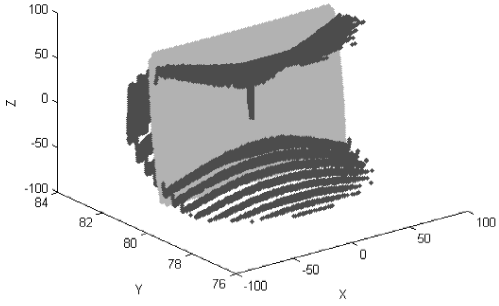
(a) -X face.



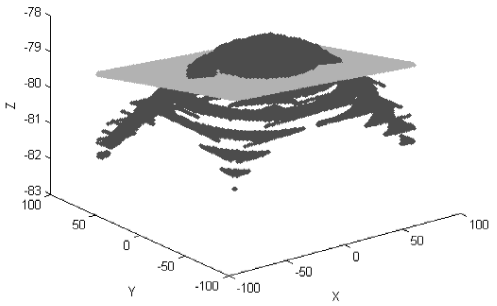
(b) +X face.



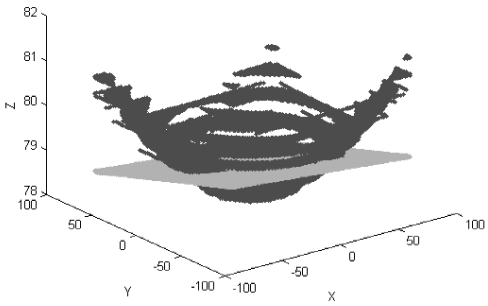
(c) -Y face.



(d) +Y face.



(e) -Z face.

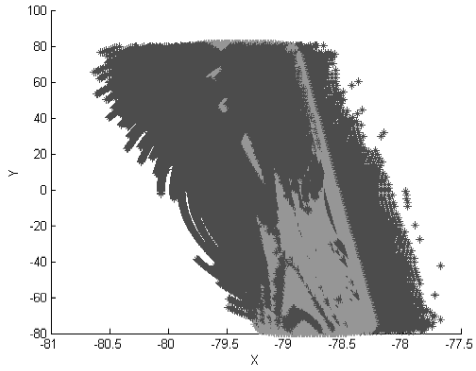


(f) +Z face.

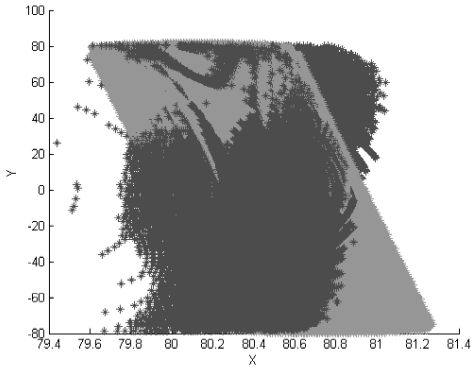
Fig. 6.5: Distorted data points (black) from the 11-6-05 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm.

# Data Points Used (-X)	62906
# Data Points Used (+X)	62895
# Data Points Used (-Y)	63205
# Data Points Used (+Y)	63474
# Data Points Used (-Z)	60819
# Data Points Used (+Z)	60678
Total Residuals (Error) (-X) (mm)	58.956
Total Residuals (Error) (+X) (mm)	63.1215
Total Residuals (Error) (-Y) (mm)	38.7808
Total Residuals (Error) (+Y) (mm)	40.0219
Total Residuals (Error) (-Z) (mm)	33.4214
Total Residuals (Error) (+Z) (mm)	34.9089
MSE (Variance) (-X) (mm)	0.055254
MSE (Variance) (+X) (mm)	0.063349
MSE (Variance) (-Y) (mm)	0.023795
MSE (Variance) (+Y) (mm)	0.025235
MSE (Variance) (-Z) (mm)	0.018366
MSE (Variance) (+Z) (mm)	0.020084
RMS (Standard Deviation) (-X) (mm)	0.23506
RMS (Standard Deviation) (+X) (mm)	0.25169
RMS (Standard Deviation) (-Y) (mm)	0.15426
RMS (Standard Deviation) (+Y) (mm)	0.15885
RMS (Standard Deviation) (-Z) (mm)	0.13552
RMS (Standard Deviation) (+Z) (mm)	0.14172

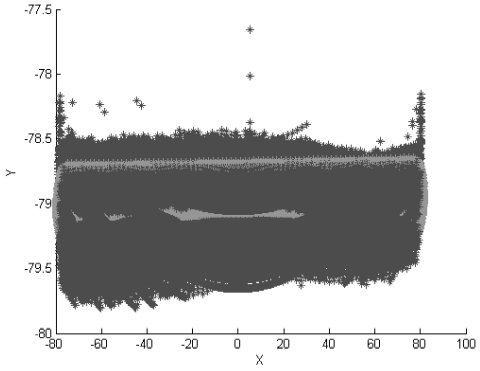
Tab. 6.12: Errors calculated from distortion correction results from 11-6-05 study.



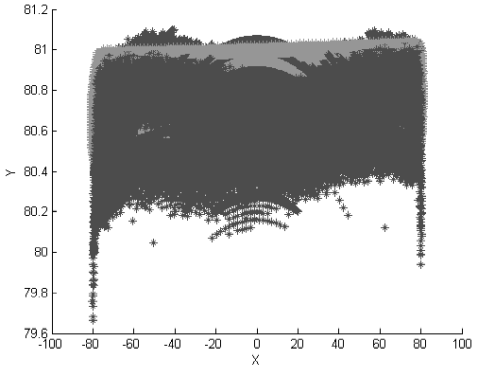
(a) -X face.



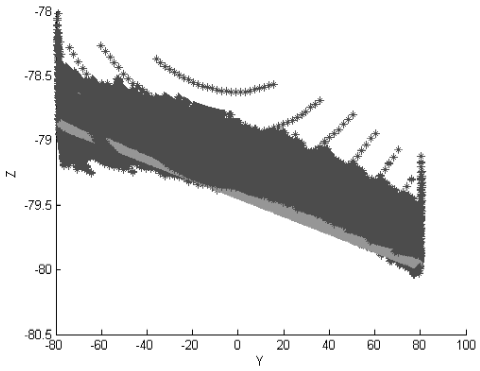
(b) +X face.



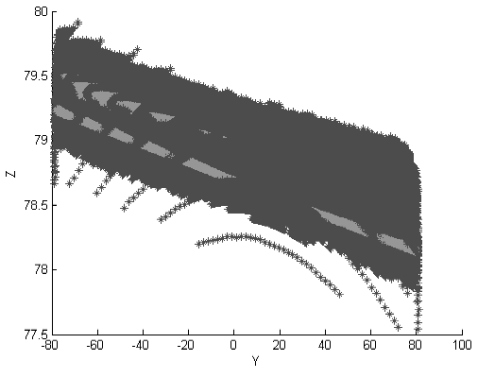
(c) -Y face.



(d) +Y face.



(e) -Z face.



(f) +Z face.

Fig. 6.6: Corrected data points (black) from the 11-6-05 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm.

Characteristics of the Data Set

Visualizing the data set acquired from the 11-6-05 study provides great insight into the characteristics of the study. The quality of the experimental setup is directly reflected in the appearance of the data set. Plotting the distorted and corrected data points against the theoretical undistorted data points for all six phantom faces yields the three-dimensional graphs shown in Figures 6.5 and 6.6.

Analyzing the orientation of the distorted data points with the theoretical undistorted data points reveals how well the phantom was placed inside the MRI scanner. Simply put: the higher quality experimental setup produced higher quality data. Likewise, the poor quality of the experimental setups in the 7-17-05 and 10-2-05 studies greatly reduced the quality and accuracy of the distortion correction. Since the phantom was placed to within one millimeter of gradient isocenter in all three dimensions in the 11-6-05 study, the distortion inherent to the data set is very symmetric. Indeed, the symmetry is not perfect, since the phantom was not placed perfectly centered with respect to gradient isocenter, but the symmetry in the distortion is much more ideal than either the 7-17-05 and 10-2-05 studies. For the first time, the distortion correction software has the opportunity to work with (nearly) symmetrically distorted faces, rather than faces that are disproportionate and lopsided. Positioning the phantom closer to gradient isocenter will increase the symmetry of the distortion in the data set, and will therefore increase the accuracy of the distortion correction. The smaller errors observed in the numerical results of the 11-6-05 study in Tables 6.11 and 6.12 are directly attributed to the increased symmetry in the distortion. Observing the location of the data also indicates the position of the phantom in the scanner. Ideally,

the positive and negative faces of the phantom should be equidistant from the origin in each dimension. The 11-6-05 study features data that is more ideally positioned in three-dimensional space, as indicated by Figure 6.5, and better matches this trend than the 7-17-05 and 10-2-05 studies.

The locations of the corrected data points in Figure 6.6 are improved because of the increased symmetry in the distortion. Observing the plots of the corrected data points against the theoretical undistorted data points further demonstrates the quality of the correction calculated from the 11-6-05 study. Ideally, the corrected data points should be as close to the theoretical data points as possible. The farther away the corrected data points lie from the theoretical undistorted points, the larger the overall error in the correction, and the lower the accuracy of the correction. The spread of the corrected data points in this study is much smaller than that of the 7-17-05 and 10-2-05 studies. The majority of the data points lie close to their theoretical undistorted locations. A handful of data points deviate away from the theoretical locations, as is apparent in some of the faces, but the important characteristic to observe is the close grouping of the majority of the points. Since the differences between the theoretical and actual undistorted points is quite reasonable, the calculated distortion correction model provides a suitable representation of the actual gradient nonlinearity distortion of the MRI scanner.

6.2.4 4-30-06 Study

The investigation into the discrepancies encountered in the 11-6-05 study (Section 7.1.2) revealed the need to adjust the shim coils in the MRI scanner prior to each scan.

Improper adjustment of the shim coils led to an extra distortion in axial sequences that was not accounted for in the distortion correction software, thereby increasing the size of the errors of the distortion correction. Shimming the coils produced images of noticeably higher quality.

To test the effectiveness of calibrating the shim coils, a fourth study was performed on April 30, 2006. The scan parameters were similar to those of the 10-2-05 and 11-6-05 studies.

- TR: 2010 ms
- TE: 2.75 ms
- Flip Angle: 20 ms
- Field of View (FOV): 200 mm^2
- Matrix: 512 $\times$ 512
- Voxel Size: 0.391 $\times$ 0.391 $\times$ 2.000 mm
- 1 Slab, 104 partitions (slices) per orientation
- Sequences: axial, coronal, sagittal
- All scanner filters turned off

The phantom was placed in the scanner in the same manner as in the 11-6-05 study, using the precision shims and the scanner's laser sight to fine-tune the phantom's position in the headcoil. The goal was to replicate the phantom's centering from the 11-6-05 study as best as possible, to fully realize the benefits gained from calibrating the shim coils.

Midplane	Plane Coefficients
X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{0.40857}, -0.0031346, 0.0034117]$
Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [-0.00086679, \mathbf{-0.066242}, 0.0032263]$
Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [0.00036201, -0.0063487, \mathbf{1.8855}]$

Tab. 6.13: Plane coefficients of midplane fitting in 4-30-06 study.

Midplane	Plane Coefficients
-X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{-79.3423}, -0.0031346, 0.0034117]$
+X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{80.1594}, -0.0031346, 0.0034117]$
-Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [-0.00086679, \mathbf{-79.9167}, 0.0032263]$
+Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [-0.00086679, \mathbf{79.7842}, 0.0032263]$
-Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [0.00036201, -0.0063487, \mathbf{-77.1711}]$
+Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [0.00036201, -0.0063487, \mathbf{80.9421}]$

Tab. 6.14: Plane coefficients of ideal planes fitting in 4-30-06 study.

Numerical Results

During the initial setup procedures of this study, serious care and attention were used when positioning the phantom in the headcoil, in order to position the phantom as close to gradient isocenter as possible. The same precision shims were used to adjust the phantom's position, and the laser sight was used to further guide phantom placement. However, the bolded values in the midplane and ideal plane results shown in Tables 6.13 and 6.14 that the phantom was slightly off-centered. The x center offset is shown to be approximately 0.4 mm, very accurate indeed. The y center offset is essentially right on the mark, there is very little room for improvement here. However, the z center offset is slightly higher, at about 1.9 mm. Although the z offset is not as great as what was seen in the 7-17-05 and 10-2-05 studies, the offset is indeed greater than one millimeter.

# Data Points Used (X)	63224
# Data Points Used (Y)	62966
# Data Points Used (Z)	60077
Total Residuals (Error) (X) (mm)	26.3768
Total Residuals (Error) (Y) (mm)	28.5108
Total Residuals (Error) (Z) (mm)	22.896
MSE (Variance) (X) (mm)	0.011004
MSE (Variance) (Y) (mm)	0.01291
MSE (Variance) (Z) (mm)	0.0087259
RMS (Standard Deviation) (X) (mm)	0.1049
RMS (Standard Deviation) (Y) (mm)	0.11362
RMS (Standard Deviation) (Z) (mm)	0.093413

Tab. 6.15: Errors calculated from plane fitting results from 4-30-06 study.

The most interesting observation from the 4-30-06 study is the fact that the accuracy of the correction results (Table 6.16) are still very similar to those obtained in the 11-6-05 study. Recall that the shim coils were calibrated prior to scanning in this study, while the 11-6-05 study did not feature calibration of the shim coils. But, the z center offset in the 4-30-06 study was over twice as much as in the 11-6-05 study, exceeding the ideal maximum offset of one millimeter.

The standard deviation values obtained in the 4-30-06 study are very similar to those observed in the 11-6-05 study, despite the extra procedures performed. Therefore, the larger values for the standard deviations in x, versus y and z, may not actually be caused by phantom centering or the shim coils. It could be that the gradient nonlinearity in x is simply greater than that of y or z. This seems logical be-

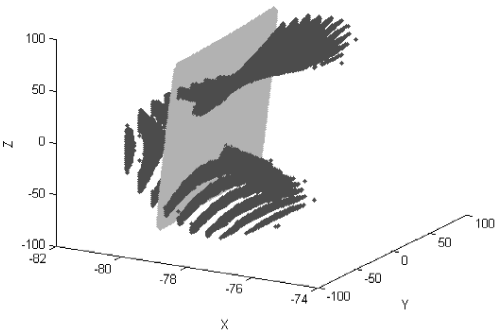
cause the tightness of fit of the spherical harmonics expansion by linear least squares is inversely proportional to the severity of the distortion. Therefore, the higher standard deviations in x may indicate stronger distortion in x in the scanner. Data points that exhibit stronger distortion must be displaced a farther distance to be placed in their ideal locations, and the spherical harmonics expansion will not exhibit as tight of a fit to that data set. This is simply a property inherent to the scanner used for this experiment. Actual observed results in different scanners of various makes and models will vary, since the gradients produced in each scanner will differ slightly.

Despite the similar results achieved here with the 4-30-06 study, the accuracy of the distortion correction is still very good indeed. All of the correction results further demonstrate the capability of the software to achieve submillimeter accuracy. However, the discrepancy in the x correction still remains. As stated in the investigation in Section 7.1.2, there are very few possible causes for this discrepancy. Calibration of the shim coils did indeed produce improved phantom images, and effectively eliminated the extra distortion seen in axial sequences. Theoretically, the accuracy of the distortion correction should be higher, and should be easily noticeable in the size of the standard deviations.

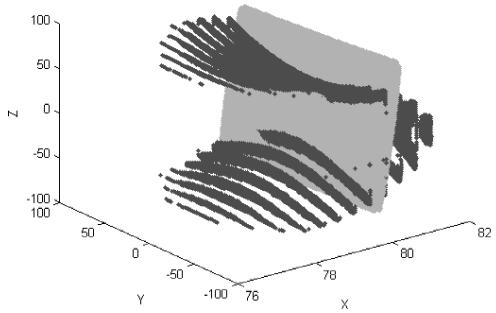
It is difficult to determine concrete conclusions by comparing the 11-6-05 study and the 4-30-06 study. The primary factor that is under scrutiny here is the phantom centering. The 11-6-05 study featured nearly ideal phantom centering, showing center offsets of less than one millimeter in x, y, and z. The 4-30-06 study, however, was only able to achieve submillimeter center offsets for x and y. The z offset was nearly twice that of the offset observed in the 11-6-05 study. Since the center offsets are

different, it is impossible to perform a direct comparison of the results. It may seem as if calibration of the shim coils did not improve the accuracy of the distortion correction. Even if this is actually the case, it would still be very wise to perform the shimming procedure, in order to remove as many artifacts and extra distortions as possible. The goal of achieving submillimeter accuracy is simply not to just obtain accuracies of less than one millimeter, but to get the size of the errors as small as possible. Every small step that can be taken to improve the accuracy of the correction method will indeed pay off in the end, and will contribute greatly to minimizing errors and maximizing accuracy.

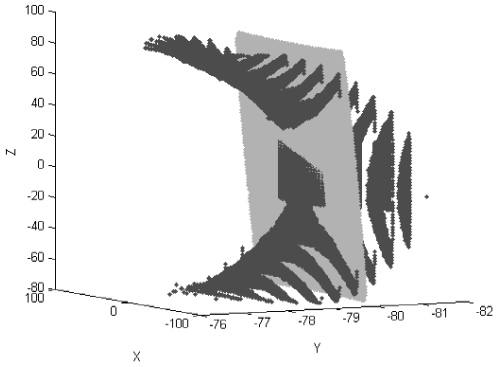
Analyzing the results of these four studies provides some insight into the sensitivity of the distortion correction method to improper phantom centering. In order to fully quantify this property, an investigation was performed using a simulated data set. The discussion of this simulation is discussed in Section 6.4.



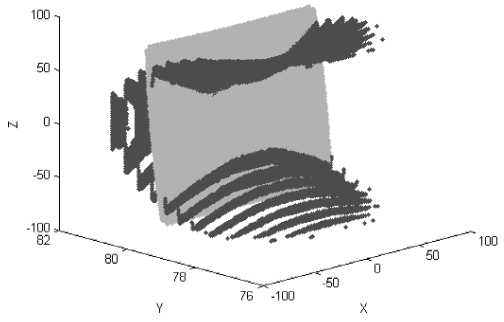
(a) -X face.



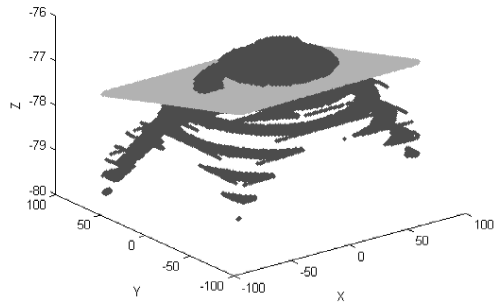
(b) +X face.



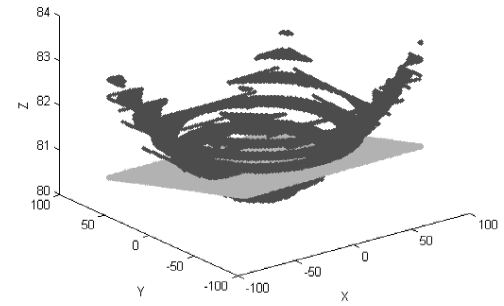
(c) -Y face.



(d) +Y face.



(e) -Z face.

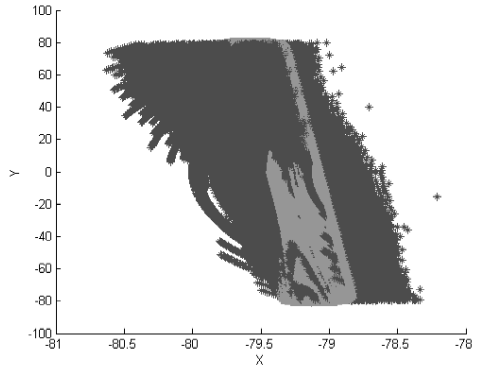


(f) +Z face.

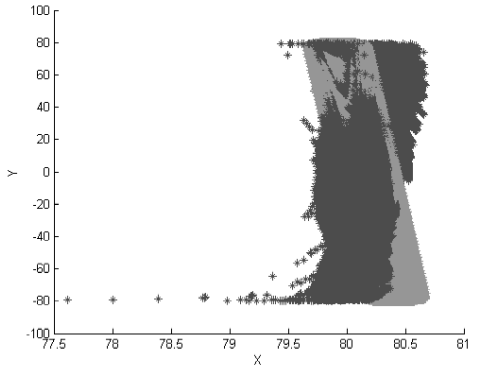
Fig. 6.7: Distorted data points (black) from the 4-30-06 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm.

# Data Points Used (-X)	63285
# Data Points Used (+X)	63246
# Data Points Used (-Y)	63175
# Data Points Used (+Y)	63411
# Data Points Used (-Z)	60601
# Data Points Used (+Z)	60091
Total Residuals (Error) (-X) (mm)	58.4885
Total Residuals (Error) (+X) (mm)	61.4923
Total Residuals (Error) (-Y) (mm)	42.6656
Total Residuals (Error) (+Y) (mm)	42.6498
Total Residuals (Error) (-Z) (mm)	37.2708
Total Residuals (Error) (+Z) (mm)	35.6394
MSE (Variance) (-X) (mm)	0.054055
MSE (Variance) (+X) (mm)	0.059787
MSE (Variance) (-Y) (mm)	0.028814
MSE (Variance) (+Y) (mm)	0.028686
MSE (Variance) (-Z) (mm)	0.022922
MSE (Variance) (+Z) (mm)	0.021137
RMS (Standard Deviation) (-X) (mm)	0.2325
RMS (Standard Deviation) (+X) (mm)	0.24451
RMS (Standard Deviation) (-Y) (mm)	0.16975
RMS (Standard Deviation) (+Y) (mm)	0.16937
RMS (Standard Deviation) (-Z) (mm)	0.1514
RMS (Standard Deviation) (+Z) (mm)	0.14539

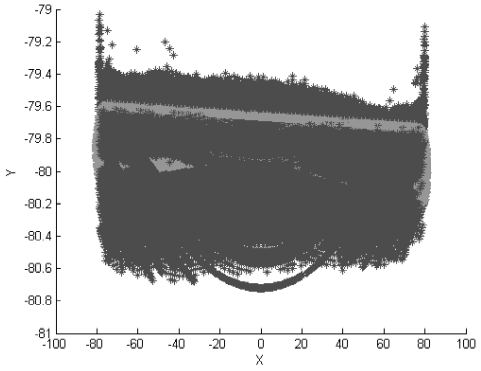
Tab. 6.16: Errors calculated from distortion correction results from 4-30-06 study.



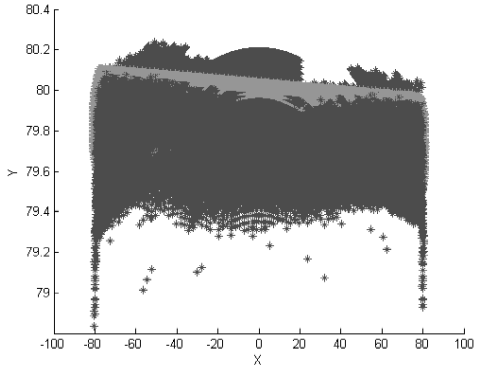
(a) -X face.



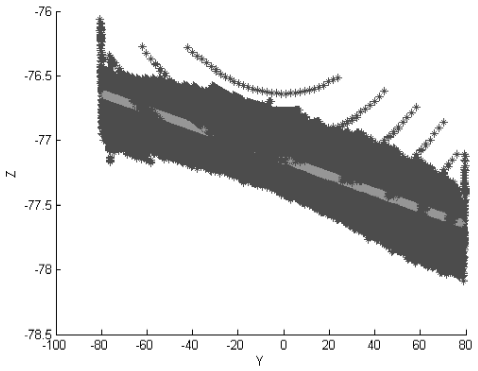
(b) +X face.



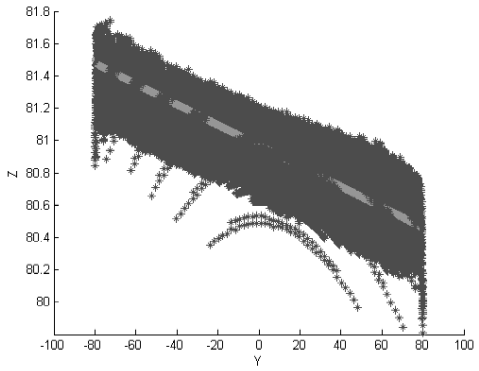
(c) -Y face.



(d) +Y face.



(e) -Z face.



(f) +Z face.

Fig. 6.8: Corrected data points (black) from the 4-30-06 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm.

Characteristics of the Data Set

Visualizing the data set acquired from the 4-30-06 study provides great insight into the characteristics of the study. The quality of the experimental setup is directly reflected in the appearance of the data set. Plotting the distorted and corrected data points against the theoretical undistorted data points for all six phantom faces yields the three-dimensional graphs shown in Figures 6.7 and 6.8.

Analyzing the orientation of the distorted data points with the theoretical undistorted data points reveals how well the phantom was placed inside the MRI scanner. Simply put: the higher quality experimental setup produced higher quality data. Likewise, the poor quality of the experimental setups in the 7-17-05 and 10-2-05 studies greatly reduced the quality and accuracy of the distortion correction. Since the phantom was placed to within one millimeter of gradient isocenter in two dimensions in the 4-30-06 study, the distortion inherent to the data set is very symmetric. The displacement of the phantom by nearly two millimeters in z adds a slight asymmetry to the distortion in the data set. Fortunately, the asymmetry is much less pronounced in this data set than in the 7-17-05 and 10-2-05 studies. Properly calibrating the shim coils also helps promote the consistency of the distortion in the data set. As in the 11-6-05 study, the distortion correction software has the opportunity to work with fairly symmetric distorted faces, rather than faces that are disproportionate and lopsided. Positioning the phantom closer to gradient isocenter will increase the symmetry of the distortion in the data set, and will therefore increase the accuracy of the distortion correction. The magnitude of the errors observed in the numerical results of the 4-30-06 study in Tables 6.15 and 6.16 are directly attributed to the nice symmetry

in the distortion. Observing the location of the data in Figure 6.7 also indicates the position of the phantom in the scanner. Ideally, the positive and negative faces of the phantom should be equidistant from the origin in each dimension. The 4-30-06 study features data that is positioned reasonably well in three-dimensional space, and better matches this trend than the 7-17-05 and 10-2-05 studies.

The locations of the corrected data points in Figure 6.8 are suitable because of the increased symmetry in the distortion. Observing the plots of the corrected data points against the theoretical undistorted data points further demonstrates the quality of the correction calculated from the 4-30-06 study. Ideally, the corrected data points should be as close to the theoretical data points as possible. The farther away the corrected data points lie from the theoretical undistorted points, the larger the overall error in the correction, and the lower the accuracy of the correction. The spread of the corrected data points in this study is slightly smaller than that of the 11-6-05 study. This is most likely the result of properly calibrating the shim coils in the 4-30-06 study, which was not performed for the 11-6-05 study. The majority of the data points lie close to their theoretical undistorted locations. A handful of data points deviate away slightly from the theoretical locations, as is apparent in some of the faces, but the important characteristic to observe is the close grouping of the majority of the points. Since the differences between the theoretical and actual undistorted points is quite reasonable, the calculated distortion correction model provides a suitable representation of the actual gradient nonlinearity distortion of the MRI scanner.

6.2.5 6-19-06 Study

Following the issues encountered in all previous studies with phantom centering, it was determined that additional procedures were required to promote consistency in phantom placement. This was accomplished by etching two lines—one horizontal and one vertical—on the anterior face of the phantom (which is the top face when the phantom is placed in the scanner) that intersect in the center of the face. The lines were inscribed by an engineering company, to ensure accurate inscribing of the lines. With these lines on the phantom face, there now exist reference marks for use with the scanner's laser sight, to more accurately align the phantom in the headcoil.

Using the precision shims utilized in the 11-6-05 and 4-30-06 studies, along with the inscribed lines and the laser sight, the phantom could now be placed and centered in the headcoil with greater confidence. To test the effectiveness of this positioning method, a fifth study was performed on June 19, 2006. The scan parameters were similar to those of the 10-2-05, 11-6-05, and 4-30-06 studies.

- TR: 2010 ms
- TE: 2.75 ms
- Flip Angle: 20 ms
- Field of View (FOV): 200 mm^2
- Matrix: 512 $\times$ 512
- Voxel Size: 0.391 $\times$ 0.391 $\times$ 2.000 mm
- 1 Slab, 104 partitions (slices) per orientation

- Sequences: axial, coronal, sagittal
- All scanner filters turned off

In addition to the extra steps performed in centering the phantom, the shim coils were calibrated, to ensure that no extra artifacts or distortions hindered the quality of the images acquired.

Midplane	Plane Coefficients
X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{1.0380}, -0.0060079, 0.0012984]$
Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [0.0069845, \mathbf{0.85202}, 0.0037581]$
Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [0.0024056, 0.000091784, \mathbf{-3.2980}]$

Tab. 6.17: Plane coefficients of midplane fitting in 6-19-06 study.

Midplane	Plane Coefficients
-X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{-78.7135}, -0.0060079, 0.0012984]$
+X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{80.7895}, -0.0060079, 0.0012984]$
-Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [0.0069845, \mathbf{-79.0005}, 0.0037581]$
+Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [0.0069845, \mathbf{80.7045}, 0.0037581]$
-Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [0.0024056, 0.000091784, \mathbf{-82.3533}]$
+Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [0.0024056, 0.000091784, \mathbf{75.7572}]$

Tab. 6.18: Plane coefficients of ideal planes fitting in 6-19-06 study.

Numerical Results

Aligning the inscribed lines on the phantom's face with the laser sight proved to be more difficult than originally planned. Theoretically, lining up the markings on the phantom with the projected laser beams should be a fairly simple task. If performed successfully, the phantom should be centered to within one millimeter of gradient

# Data Points Used (X)	63046
# Data Points Used (Y)	63444
# Data Points Used (Z)	60546
Total Residuals (Error) (X) (mm)	25.7921
Total Residuals (Error) (Y) (mm)	35.778
Total Residuals (Error) (Z) (mm)	23.4001
MSE (Variance) (X) (mm)	0.010552
MSE (Variance) (Y) (mm)	0.020176
MSE (Variance) (Z) (mm)	0.0090438
RMS (Standard Deviation) (X) (mm)	0.10272
RMS (Standard Deviation) (Y) (mm)	0.14204
RMS (Standard Deviation) (Z) (mm)	0.095099

Tab. 6.19: Errors calculated from plane fitting results from 6-19-06 study.

isocenter in all three dimensions. Unfortunately, this was not the case, as is indicative by the bolded midplane and ideal plane results shown in Tables 6.17 and 6.18. These results indicate a displacement of approximately 1.0 mm in x, 0.9 mm in y, and over 3 mm in z. These results are frustrating, to say the least, considering the extra modifications made in order to *improve* the accuracy of phantom placement in the headcoil. The difficulty experienced in centering the phantom using this reference system resulted in an increased center offset in z. There were several factors that contributed to this increased displacement.

The most frustrating factor in the centering of the phantom became apparent right away. The projected laser beams were not parallel with respect to the inscribed lines on the face of the phantom. Therefore it was impossible to line up both sides of the

lines with the laser beam. When the laser beam was aligned with the line on one side of the face, the beam would be about two to three millimeters away from the line on the other side of the face. Determining the true center position of the phantom in light of this situation proved to be guesswork at best.

The question that arises from this situation is: *what is causing the misalignment?* There are three possible factors contributing to this situation:

- The headcoil itself features a slight rotation that forces the phantom to be misaligned with respect to the laser beams.
- The laser sight is not calibrated properly, and therefore projects beams that are slightly rotated with respect to the central axes of the scanner.
- The inscriptions on the phantom feature a slight rotation, and therefore do not align properly with the laser sight on the scanner.

Investigating the validity of the above factors will be indeed difficult. Checking and measuring the headcoil and laser sight for alignment is nearly impossible. Performing such measurements require accurate, reliable, advanced tools that are simply not available. In addition to this difficulty, it is unknown if there is any possibility of repositioning either the headcoil or the laser sight to bring it back to center, in the event that either piece of equipment is determined to be offcentered *with a high level of confidence.* In other words, it would simply not be feasible to pursue this endeavor, considering the scope and timeframe of this project.

The third possibility of the inscribed lines on the phantom's face is not likely. The phantom was sent out to an engineering company (the same company that was con-

tracted to shorten the phantom's foot by 15 mm), which verified the alignment of the inscriptions with high accuracy. Additionally, the markings were verified for accuracy once again using basic measurement tools. Therefore, the possibility of skewed markings on the phantom's face most likely did not play a role in the offcentering of the phantom in the scanner.

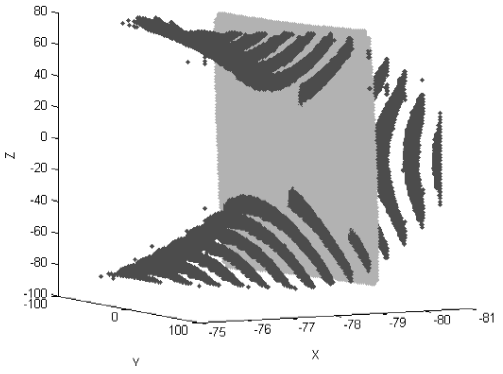
In addition to the improper alignment of the laser with the markings on the phantom, the second factor that contributed to the difficulty in centering the phantom was the way in which the laser beam appeared when shown on the phantom. Since the laser sight used in the scanner projects a *very* thin beam of red light, the light reflects off the phantom differently on the smooth face than inside the inscribed lines. The width of the markings were indeed small, but the width of the laser beams is much, *much* smaller. Shining the laser beam within the groove of the inscribed lines changed the appearance of the beam, making visual inspection of the phantom and the laser beams difficult. Considering this situation, it may be very likely that the phantom was not placed correctly in the scanner simply due to operator error.

Whatever the case was, the distortion correction results shown in Table 6.20 are still quite consistent with those obtained in the 11-6-05 and 4-30-06 studies. As was the case with the 4-30-06 study, the standard deviations of the correction in x was higher than those observed in y and z. This time, the standard deviations in x are slightly smaller than those observed in the 11-6-05 and 4-30-06 studies. This is good news considering the fact that the phantom was offcentered in all three dimensions. In fact, this study provided one of the most accurate distortion correction results observed in all five studies, in terms of the size of the standard deviations in Table

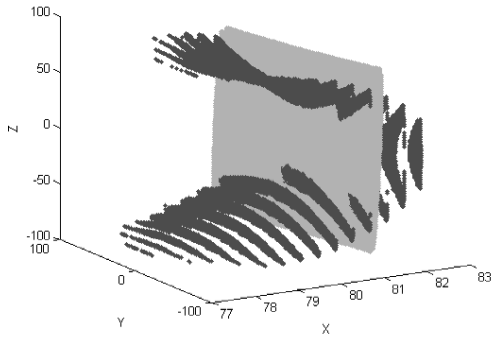
6.20.

The larger values for the standard deviations in x, versus y and z, may not actually be caused by phantom centering or the shim coils. It could be that the gradient nonlinearity in x is simply greater than that of y or z. The distortion correction results in this trial are very similar to those of the 4-30-06 study, despite the increased displacement of the phantom in z. This conclusion seems logical because the tightness of fit of the spherical harmonics expansion by linear least squares is inversely proportional to the severity of the distortion. Therefore, the higher standard deviations in x may indicate stronger distortion in x in the scanner. Data points that exhibit stronger distortion must be displaced a farther distance to be placed in their ideal locations, and the spherical harmonics expansion will not exhibit as tight of a fit to that data set. This is simply a property inherent to the scanner used for this experiment. Actual observed results in different scanners of various makes and models will vary, since the gradients produced in each scanner will differ slightly.

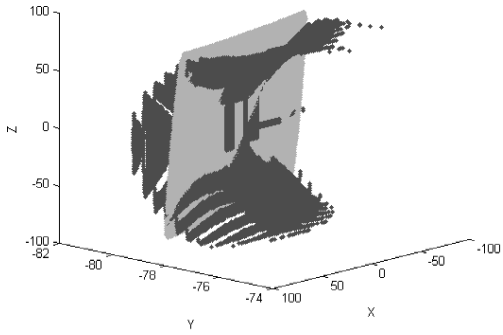
Perhaps the inscribed lines on the face of the phantom did in fact improve the accuracy of the correction results. The results obtained in the 6-19-06 study once again show the software's capability of achieving submillimeter accuracy in the distortion correction. The accuracy obtained in x is approximately 0.66 mm, correlating to slightly greater than 1.5 pixels. The accuracies obtained in y and z are very good as well, correlating to approximately 1.5 pixels and one pixel, respectively.



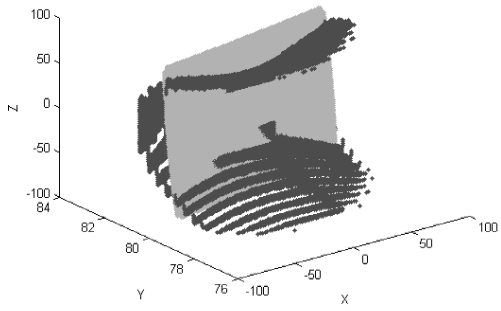
(a) -X face.



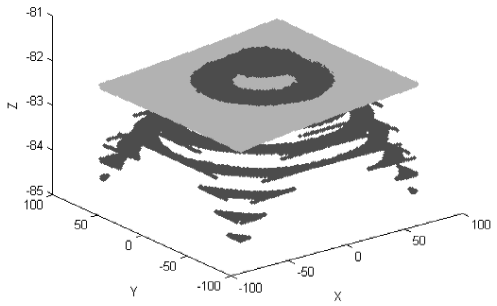
(b) +X face.



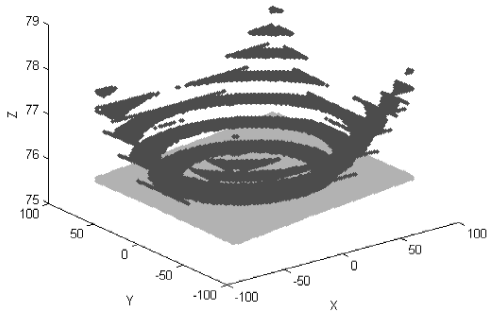
(c) -Y face.



(d) +Y face.



(e) -Z face.

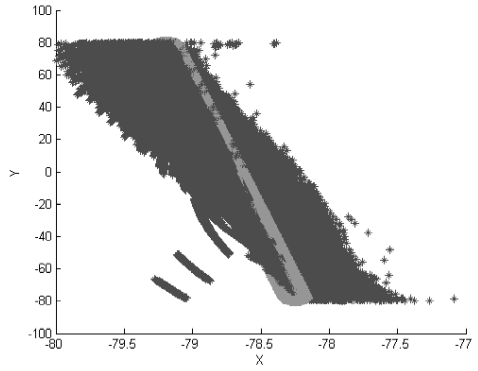


(f) +Z face.

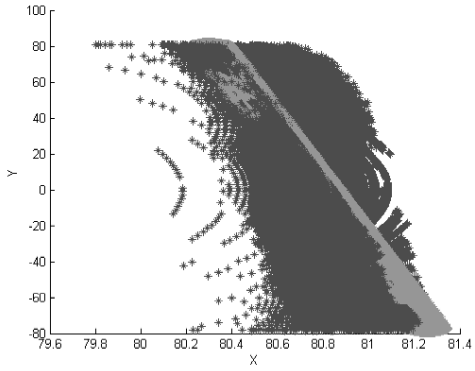
Fig. 6.9: Distorted data points (black) from the 6-19-06 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm.

# Data Points Used (-X)	63234
# Data Points Used (+X)	63363
# Data Points Used (-Y)	63553
# Data Points Used (+Y)	63860
# Data Points Used (-Z)	60643
# Data Points Used (+Z)	60877
Total Residuals (Error) (-X) (mm)	55.8341
Total Residuals (Error) (+X) (mm)	55.7088
Total Residuals (Error) (-Y) (mm)	48.9718
Total Residuals (Error) (+Y) (mm)	46.5682
Total Residuals (Error) (-Z) (mm)	34.2992
Total Residuals (Error) (+Z) (mm)	38.5495
MSE (Variance) (-X) (mm)	0.0493
MSE (Variance) (+X) (mm)	0.048979
MSE (Variance) (-Y) (mm)	0.037736
MSE (Variance) (+Y) (mm)	0.033959
MSE (Variance) (-Z) (mm)	0.019399
MSE (Variance) (+Z) (mm)	0.024411
RMS (Standard Deviation) (-X) (mm)	0.22204
RMS (Standard Deviation) (+X) (mm)	0.22131
RMS (Standard Deviation) (-Y) (mm)	0.19426
RMS (Standard Deviation) (+Y) (mm)	0.18428
RMS (Standard Deviation) (-Z) (mm)	0.13928
RMS (Standard Deviation) (+Z) (mm)	0.15624

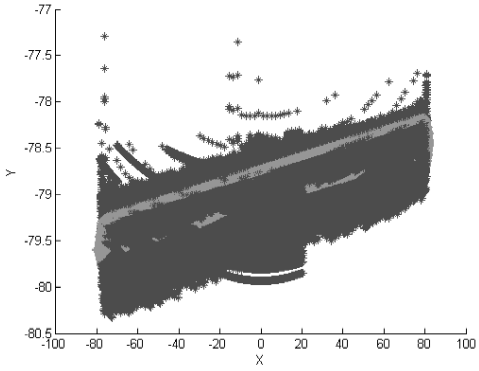
Tab. 6.20: Errors calculated from distortion correction results from 6-19-06 study.



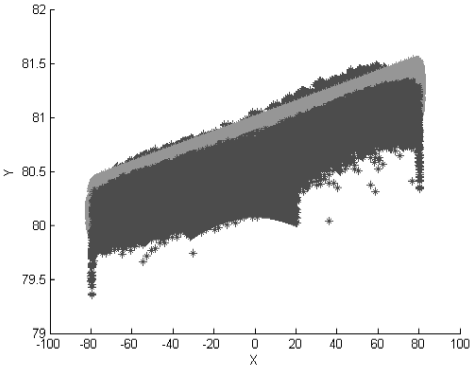
(a) -X face.



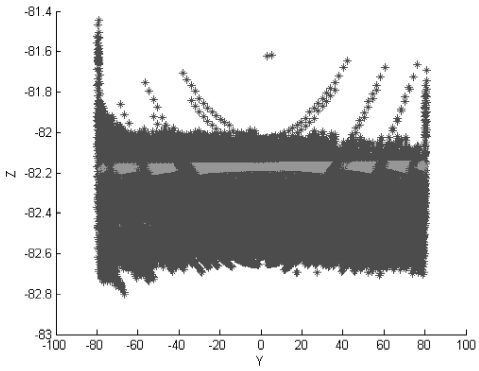
(b) +X face.



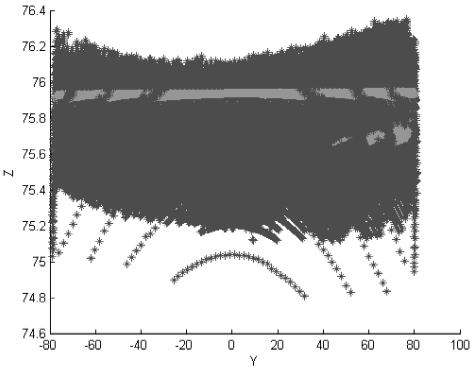
(c) -Y face.



(d) +Y face.



(e) -Z face.



(f) +Z face.

Fig. 6.10: Corrected data points (black) from the 6-19-06 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm.

Characteristics of the Data Set

Visualizing the data set acquired from the 6-19-06 study provides great insight into the characteristics of the study. The quality of the experimental setup is directly reflected in the appearance of the data set. Plotting the distorted and corrected data points against the theoretical undistorted data points for all six phantom faces yields the three-dimensional graphs shown in Figures 6.9 and 6.10.

Analyzing the orientation of the distorted data points with the theoretical undistorted data points reveals how well the phantom was placed inside the MRI scanner. Simply put: the higher quality experimental setup produced higher quality data. Likewise, the poor quality of the experimental setups in the 7-17-05 and 10-2-05 studies greatly reduced the quality and accuracy of the distortion correction. Since the phantom was placed to nearly one millimeter of gradient isocenter in two dimensions in the 6-19-06 study, the distortion inherent to the data set is very symmetric. The displacement of the phantom of over three millimeters in z adds a slight asymmetry to the distortion in the data set. Fortunately, the asymmetry is much less pronounced in this data set than in the 7-17-05 and 10-2-05 studies. Properly calibrating the shim coils also helps promote the consistency of the distortion in the data set. As in the 11-6-05 and 4-30-06 studies, the distortion correction software has the opportunity to work with fairly symmetric distorted faces, rather than faces that are disproportionate and lopsided. Positioning the phantom closer to gradient isocenter will increase the symmetry of the distortion in the data set, and will therefore increase the accuracy of the distortion correction. The magnitude of the errors observed in the numerical results of the 6-19-06 study in Tables 6.19 and 6.20 are directly at-

tributed to the nice symmetry in the distortion. Observing the location of the data in Figure 6.9 also indicates the position of the phantom in the scanner. Ideally, the positive and negative faces of the phantom should be equidistant from the origin in each dimension. The 6-19-06 study features data that is positioned reasonably well in three-dimensional space, and better matches this trend than the 7-17-05 and 10-2-05 studies.

The locations of the corrected data points are suitable because of the increased symmetry in the distortion. Observing the plots of the corrected data points against the theoretical undistorted data points in Figure 6.10 further demonstrates the quality of the correction calculated from the 6-19-06 study. Ideally, the corrected data points should be as close to the theoretical data points as possible. The farther away the corrected data points lie from the theoretical undistorted points, the larger the overall error in the correction, and the lower the accuracy of the correction. The spread of the corrected data points in this study is very similar to that of the 4-30-06 study. This is most likely the result of properly calibrating the shim coils prior to scanning the phantom. The majority of the data points lie close to their theoretical undistorted locations. A handful of data points deviate away slightly from the theoretical locations, as is apparent in some of the faces, but the important characteristic to observe is the close grouping of the majority of the points. Since the differences between the theoretical and actual undistorted points is quite reasonable, the calculated distortion correction model provides a suitable representation of the actual gradient nonlinearity distortion of the MRI scanner.

6.2.6 Calculated Distortion Correction Parameters

	7-17-05	**10-2-05**	**11-6-05**
K_{x0}	-1.6054×10^{-6}	-8.1403×10^{-7}	-2.9697×10^{-7}
K_{x1}	7.067×10^{-6}	7.1821×10^{-6}	6.6184×10^{-6}
K_{x2}	-2.7078×10^{-10}	-2.8644×10^{-10}	-2.5027×10^{-10}
K_{x3}	3.9457×10^{-11}	-2.8063×10^{-11}	-7.5172×10^{-11}
K_{x4}	1.7834×10^{-10}	1.2658×10^{-10}	1.9062×10^{-10}
K_{y0}	-1.0835×10^{-6}	-1.564×10^{-7}	1.7804×10^{-7}
K_{y1}	5.9438×10^{-6}	5.8519×10^{-6}	4.9858×10^{-6}
K_{y2}	-3.2484×10^{-10}	-2.685×10^{-10}	-2.1258×10^{-10}
K_{y3}	-8.6256×10^{-12}	-8.6393×10^{-11}	-1.156×10^{-10}
K_{y4}	2.8776×10^{-10}	2.2417×10^{-10}	3.0211×10^{-10}
K_{z0}	-2.3916×10^{-6}	-2.737×10^{-6}	-1.1895×10^{-5}
K_{z1}	2.5815×10^{-7}	1.4155×10^{-6}	1.9964×10^{-5}
K_{z2}	-1.954×10^{-10}	-1.5401×10^{-10}	1.4263×10^{-9}
K_{z3}	9.9373×10^{-11}	7.7077×10^{-11}	2.2744×10^{-11}
K_{z4}	7.0689×10^{-11}	-3.195×10^{-11}	-3.0273×10^{-9}

Tab. 6.21: Distortion correction parameters for all studies, part 1.

Due to the sheer number of values calculated for each set of distortion correction parameters, they are presented below in two large tables: Tables 6.21 and 6.22. This promotes ease of comparison between each study, and how each of the characteristics of each study contributed to the final correction results. To recap the highlights of each study:

- **7-17-05**:
  - Phantom severely offcentered in scanner.
  - Calibration of shim coils not performed.

	4-30-06	**6-19-06**	**Langlois' Parameters**
K_{x0}	-3.3×10^{-7}	-8.8382×10^{-7}	1.01×10^{-6}
K_{x1}	6.5079×10^{-6}	7.4406×10^{-6}	-6.33×10^{-6}
K_{x2}	-2.6081×10^{-10}	-2.9227×10^{-10}	1.24×10^{-10}
K_{x3}	-6.9144×10^{-11}	-2.6661×10^{-11}	-1.11×10^{-11}
K_{x4}	2.4875×10^{-10}	1.3204×10^{-10}	-3.64×10^{-12}
K_{y0}	8.4594×10^{-8}	-2.8245×10^{-7}	1.00×10^{-6}
K_{y1}	5.1732×10^{-6}	5.878×10^{-6}	-6.28×10^{-6}
K_{y2}	-2.3672×10^{-10}	-2.5701×10^{-10}	1.43×10^{-10}
K_{y3}	-1.0569×10^{-10}	-7.6522×10^{-11}	-8.42×10^{-12}
K_{y4}	3.2755×10^{-10}	2.1287×10^{-10}	-4.25×10^{-11}
K_{z0}	-2.5637×10^{-6}	-5.2356×10^{-6}	-1.43×10^{-6}
K_{z1}	1.7746×10^{-6}	2.8076×10^{-6}	-2.89×10^{-6}
K_{z2}	-1.9439×10^{-10}	2.8611×10^{-10}	2.28×10^{-10}
K_{z3}	8.2028×10^{-11}	5.102×10^{-11}	-1.02×10^{-11}
K_{z4}	-8.3005×10^{-11}	-2.8509×10^{-10}	1.05×10^{-10}

Tab. 6.22: Distortion correction parameters for all studies, part 2. Also includes Langlois' parameters.

  - Only two sequences performed.

- **10-2-05**:

  - Phantom severely offcentered in scanner.
  - Calibration of shim coils not performed.
  - All three sequences performed.

- **11-6-05**:

  - Phantom centered to within 1 mm of isocenter.
  - Calibration of shim coils not performed.
  - All three sequences performed.

- **4-30-06**:
  - Phantom slightly offcentered in scanner.
  - Calibration of shim coils performed.
  - All three sequences performed.
- **6-19-06**:
  - Phantom slightly offcentered in scanner, despite using inscribed lines with laser sight.
  - Calibration of shim coils performed.
  - All three sequences performed.

The size of each distortion correction parameter directly correlates to the amount of distortion observed in each study. When the distortion is greater in the study, the quality of the least squares fit for the distortion correction model degrades. This is because the spread of the data points is greater, and the trend that the data points follow deviates further from the spherical harmonics expansion. The least squares fit represents the best reconciliation of the spherical harmonics expansion to the data sets. Computing the distortion correction parameters on a data set with a smaller degree of distortion will result in a tighter least squares fit, yielding smaller errors and higher accuracy.

Consider the effects of the center offsets and differences caused by calibrating the shim coils in each data set. Observing the magnitude of each distortion correction parameter in each study will identify the trends and characteristics inherent in each data set. The magnitude of each parameter is an indication of the severity of the

distortion observed in each study. The larger the K value, the greater the distortion observed. This applies as well to other factors, such as phantom centering and calibration of the shim coils.

For purposes of further reference, the distortion correction parameters calculated by Langlois [33] are displayed in Table 6.22. A direct comparison of values is not wise, since the experimental setups are vastly different:

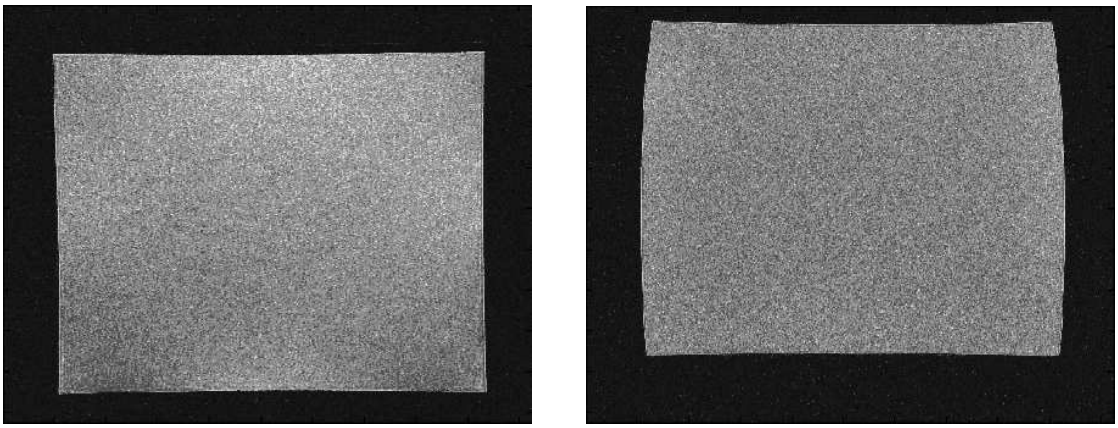
- TR: 2200 ms
- TE: 3.0 ms
- Flip Angle: not specified
- Field of View (FOV): 256 mm^2
- Matrix: 256 $\times$ 256
- Voxel Size: 1.0 $\times$ 1.0 $\times$ 1.7 mm
- 1 Slab, 124 partitions (slices) per orientation
- Sequences: coronal, sagittal
- All scanner filters turned off

Rather than comparing the actual values directly, a more suitable comparison method is to compare the magnitudes of each distortion correction value. Analyzing Langlois' calculated terms alongside the values obtained from the five studies shows that all the values follow the same general trend. In other words, the size of the values calculated by Langlois are similar to those calculated from the five studies performed here. This gives an indication to the validity of the results obtained here.

6.3 Distortion Correction Image Analysis

Analyzing the numerical results provides only half of the picture, literally. In order to fully understand the results obtained from the distortion correction software, it is necessary to also perform an image analysis. Numerical results can only go so far to demonstrate both the characteristics of a study and the effectiveness of the distortion correction. Performing a qualitative analysis will demonstrate how all the numbers and calculations come together to produce a practical, useful result.

6.3.1 7-17-05 Study

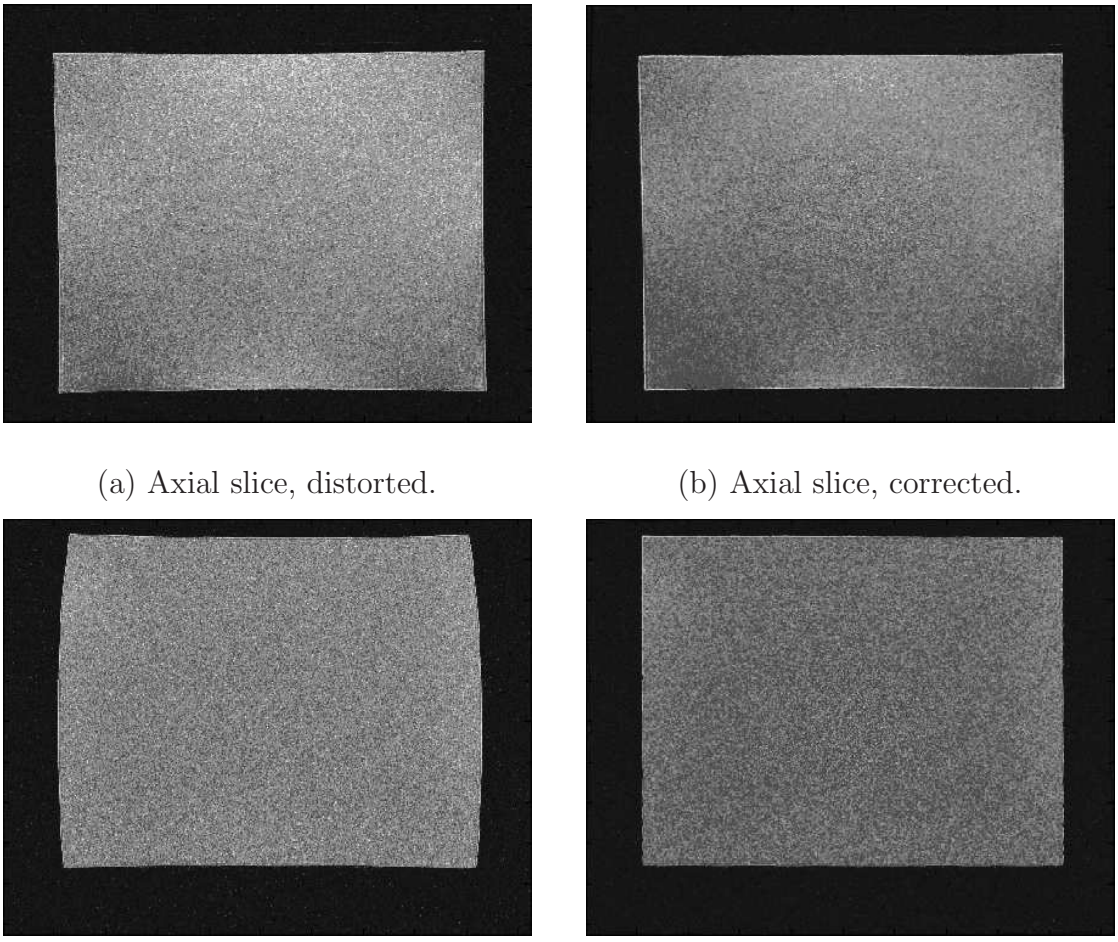


(a) Axial slice. (b) Coronal slice.

Fig. 6.11: Example images from the 7-17-05 study.

The study performed on July 17, 2005 was the first study acquired for this project. The experimental setup was quite simple, since this study was primarily a trial run to determine the best experimental setup. Because of this, the phantom was not centered properly in the scanner.

Observe the axial and coronal images of the phantom in Figure 6.11. The large center offset in z is very apparent in the coronal slice, Figure 6.11(b). A properly centered phantom will appear in the middle of the image. The axial slice, Figure 6.11(a), appears to be well-centered. This is reflected in the numerical results, which showed that the phantom was positioned 1 mm from isocenter in x, and positioned 4.5 mm from isocenter in y. The large offset in z becomes very apparent in the images. The observed 12 mm offset pushes the image of the phantom upwards in the image. Since the phantom is displaced by this relatively large amount, the distortion is no longer symmetric (the same holds true with the axial image as well, but the effect is much greater in the coronal image). Notice how the distortion present in the image of the phantom is much greater at the upper portion of the coronal image than in the lower portion. This characteristic creates some problems in terms of numerical accuracy due to the fact that the software assumes the distortion is symmetric. The numerical results show the implications of an offcentered phantom, but analyzing the images really shows the impact that improper centering of the phantom has on the images obtained. Calculating the distortion correction parameters and applying them to the example images shown in Figure 6.11 yields the results shown in Figure 6.12.



(a) Axial slice, distorted.

(b) Axial slice, corrected.

(c) Coronal slice, distorted.

(d) Coronal slice, corrected.

Fig. 6.12: Axial and coronal images from the 7-17-05 study, before and after distortion correction.

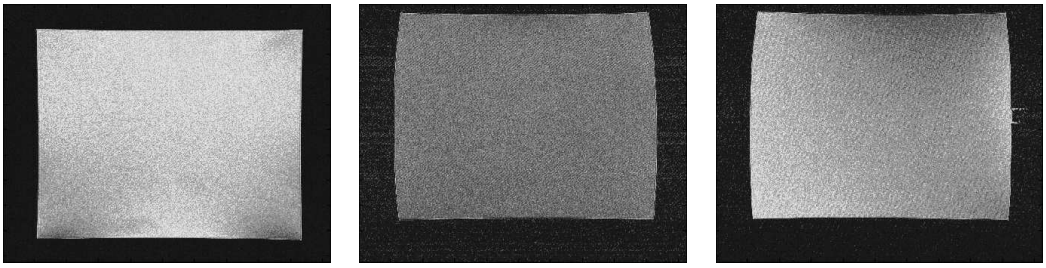
The quality of the correction results directly reflect the experimental setup. Recall that in this study, the phantom was offcentered by at least 1 mm from isocenter in all three dimensions, and the shim coils were not calibrated prior to scanning. Both of these factors contribute to the appearance of the phantom in the corrected images. Observe the appearance of the phantom in the corrected axial image, Figure 6.12(b). The horizontal faces of the phantom in the images are not parallel, giving the false impression that the phantom is not cubic, but asymmetrical. This extra distortion is the result of not calibrating the shim coils prior to scanning. The same characteristics were also found in axial images acquired in the 11-6-05 study, see Figure 7.1. The experimental studies performed on 3-29-06 showed the differences between images with and without calibration of the shim coils. For a detailed discussion concerning the importance of calibrating the shim coils prior to scanning, refer to Section 7.1.2.

Observing the images carefully also show that the edges are not perfectly straight. A small amount of the distortion is still present in the images, indicative of the curvature inherent in the images in Figures 6.12(b) and 6.12(d). Although the distortion is small, it is enough to influence accuracy in target localization. The margin of error in these studies is less than three pixels, each pixel equating to 0.391 mm. The curvature present in the edges of the phantom is about three pixels from endpoint to midpoint, thereby exceeding the level of accuracy required.

Both the quantitative and qualitative results demonstrate that using this study for calculating distortion correction parameters is not recommended. The parameters calculated do not correctly reflect the distortion inherent in the scanner. The lack of calibrated shim coils adds extra distortion in the bottom left portion of the phantom

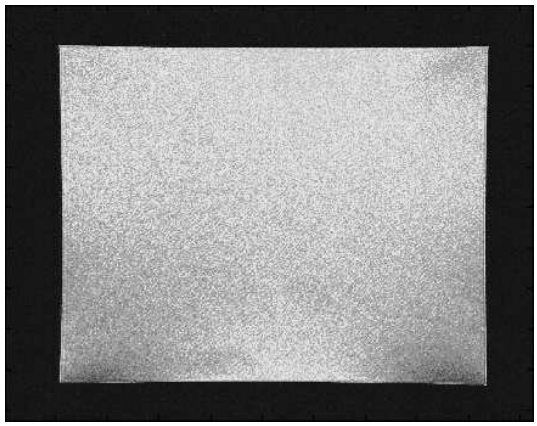
in axial images. Additionally, the great offset in z, as well as the > 1 mm offsets in x and y, create a data set that does not adhere to the assumption of symmetric distortion. Therefore, the distortion correction model calculated here cannot be confidently applied to a data set, in order to achieve submillimeter accuracy in target localization. It would be highly recommended to refine the experimental setup and recalculate the distortion correction model before proceeding.

6.3.2 10-2-05 Study

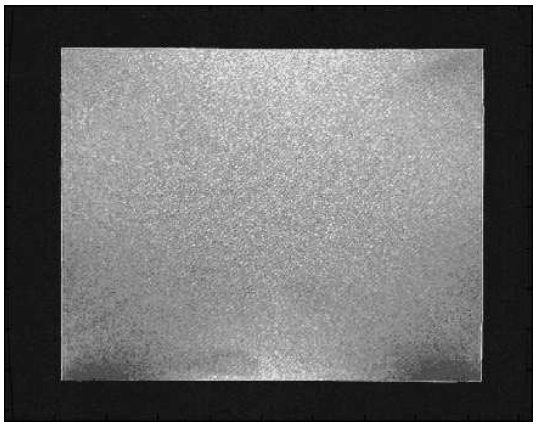


(a) Axial slice. (b) Coronal slice. (c) Sagittal slice.

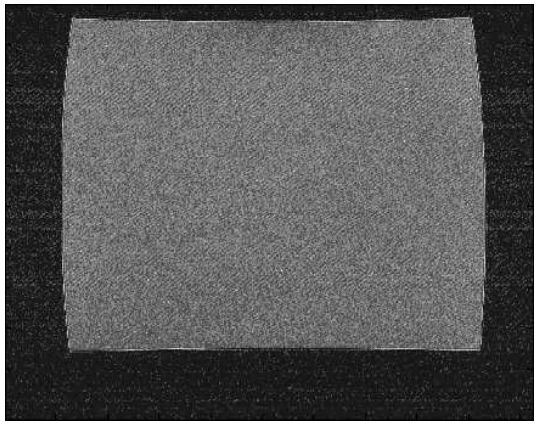
Fig. 6.13: Example images from the 10-2-05 study.



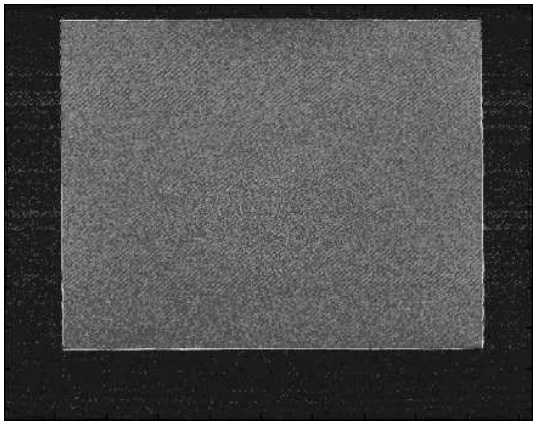
(a) Axial slice, distorted.



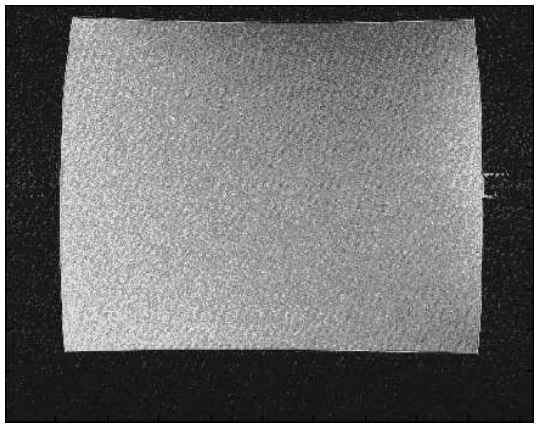
(b) Axial slice, corrected.



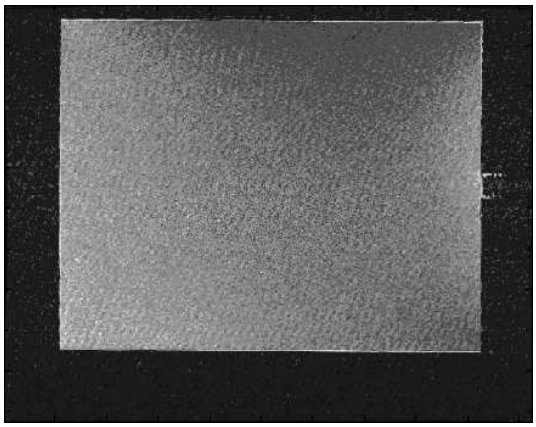
(c) Coronal slice, distorted.



(d) Coronal slice, corrected.



(e) Sagittal slice, distorted.



(f) Sagittal slice, corrected.

Fig. 6.14: Axial, coronal, and sagittal images from the 10-2-05 study, before and after distortion correction.

The study performed on October 2, 2005 is very similar to the 7-17-05 study. The experimental setup was the same in both studies. The primary difference between the two data sets is that the 10-2-05 study also provides sagittal images. The extent of the phantom offcentering becomes even more apparent in this study, now that all three acquisitions are available. Typical axial, coronal, and sagittal slices from the 10-2-05 study are shown in Figure 6.13.

The large center offset in z is very apparent in the coronal and sagittal slices, Figures 6.13(b) and 6.13(c). Recall that a properly centered phantom will appear in the middle of the image. The axial slice, Figure 6.13(a), appears to be very well-centered, even more so than in the 7-17-05 study. This observation is reflected as well in the numerical results, which confirm that the phantom was positioned approximately 1.5 mm from isocenter in x, and 1.0 mm from isocenter in y.

The large offset in z becomes very apparent in the images. The observed 13.5 mm offset pushes the image of the phantom upwards in the coronal and sagittal images. Since the phantom is displaced by this relatively large amount, the distortion is no longer symmetric in z. The phantom was centered well in x and y, so the asymmetry in the distortion in x and y is not very apparent. Notice how the distortion present in the image of the phantom is much greater at the upper portion of the coronal and sagittal images than in the lower portions. This characteristic creates some problems in terms of numerical accuracy due to the fact that the software assumes the distortion is symmetric. The numerical results show the implications of an offcentered phantom, but analyzing the images really shows the impact that improper centering of the phantom has on the images obtained. Calculating the distortion correction

parameters and applying them to the example images shown in Figure 6.13 yields the results shown in Figure 6.14.

As was the case in the 7-17-05 study, the quality of the correction results directly reflect the experimental setup. In this study, the phantom was offcentered by at least 1 mm from isocenter in all three dimensions, and the shim coils were not calibrated prior to scanning. Both of these factors contribute to the appearance of the phantom in the corrected images.

Of particular interest is the apparent absence of the telltale signs of not calibrating the shim coils. Observing the lower left portion of the phantom in the axial images yields nothing out of the ordinary, in terms of asymmetries. It was believed that the shim coils were not calibrated prior to acquiring these images; however, the axial images show no signs of distortion or artifacts caused by not calibrating the shim coils. It is very possible that the shim coils were adjusted differently in the 7-17-05 and 10-2-05 studies, causing the difference in characteristics of each set of images. It is unknown how many MRI scans were performed on the test scanner in that three month span, and how each scan was configured. Therefore, it is very possible that the shim coils were adjusted for a previous scan, and that configuration coincidentally was very close to, or exactly, the ideal configuration for the 10-2-05 study. A concrete explanation for this peculiarity is not possible, and the truth behind it is unknown.

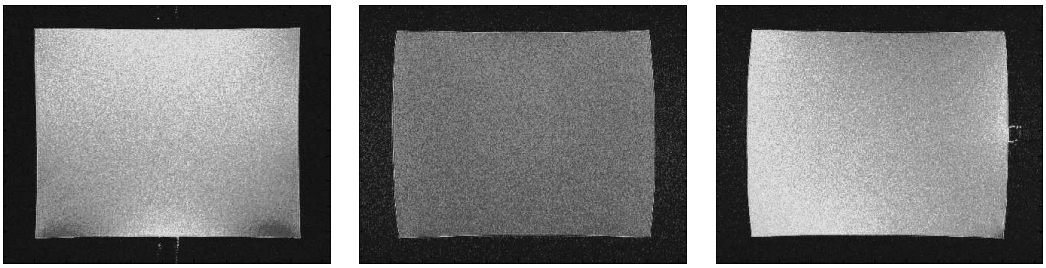
The corrected axial image, Figure 6.14(b), appears to be very accurate. The x and y correction models have yielded a symmetric image with straight edges. The fact that the phantom was centered well in x and y contributes to this result. However, observing the coronal and sagittal images yields a different result. Upon closer exam-

ination of Figures 6.14(d) and 6.14(f), the horizontal edges of the phantom exhibit slight curvatures. The curvature is symmetric on the right and left sides of the image, again confirming the good centering in x and y. Compare the corrected images to the original distorted images, and notice the appearance of the asymmetry in the distortion from the upper and lower portions of the images. It appears as if the coefficients of the distortion correction model were most influenced by the greater distortion in the upper portions of the images. As a result, the model handled the distortion present in the top portions of the images, where the distortion was greatest. However, the lower portions of the image were not corrected to the same effectiveness, since the distortion was much less in these areas. Since a good portion of the image exhibited a large amount of distortion due to the offcentering, the distortion correction model calculated is not consistent for all parts of the image. The model is only suitable for the upper portions of the image, because it tends to "over-correct" the lower portions. This is most noticeable by the lower corners of the phantom. Observe how the lower corners of the phantom are closer together than the top portions. The appearance of the lower portions of the coronal and sagittal images mimic the original images.

Analyzing the quantitative and qualitative results demonstrate that using this study for calculating distortion correction parameters is not recommended. The parameters calculated do not correctly reflect the distortion inherent in the scanner. Since the configuration of the shim coils is unknown in this scan, it is impossible to determine if there are any additional distortions or artifacts that are present in the data set that are not immediately noticeable to the eye. Additionally, the great offset in z creates a data set that does not adhere to the assumption of symmetric

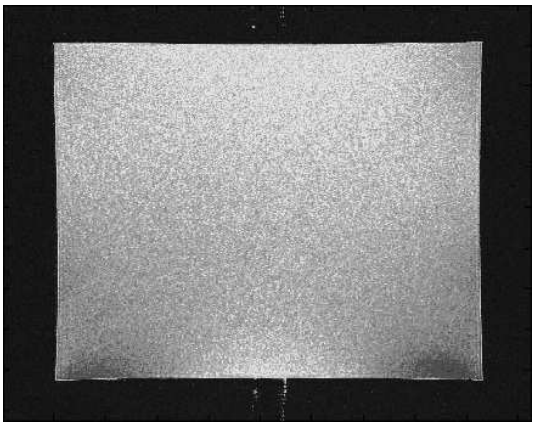
distortion. Even though the numerical results demonstrate that the study may be capable of calculating the distortion correction model with submillimeter accuracy, the images display features that reduce the confidence of performing accurate target localization. Therefore, it would not be recommended to use this data set to calculate the distortion correction model.

6.3.3 11-6-05 Study

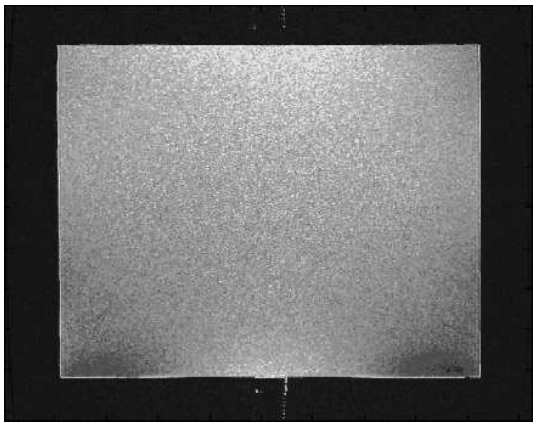


(a) Axial slice. (b) Coronal slice. (c) Sagittal slice.

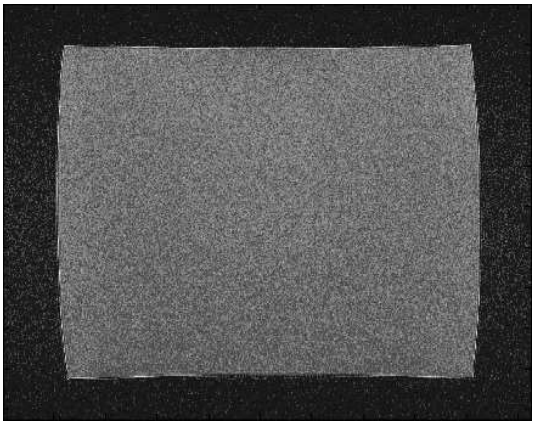
Fig. 6.15: Example images from the 11-6-05 study.



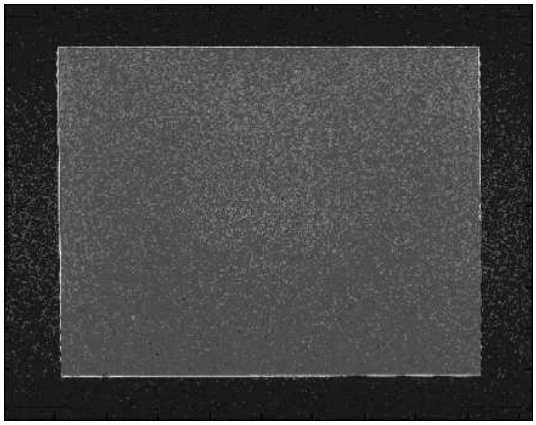
(a) Axial slice, distorted.



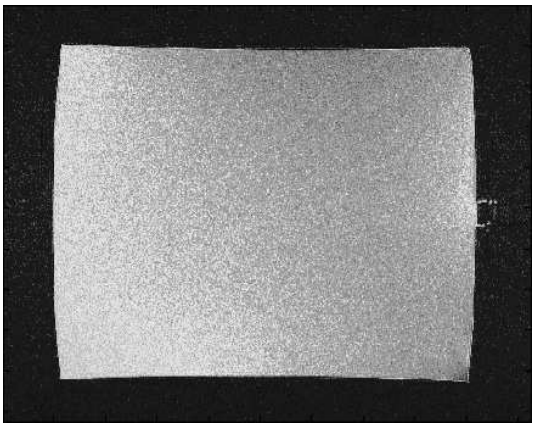
(b) Axial slice, corrected.



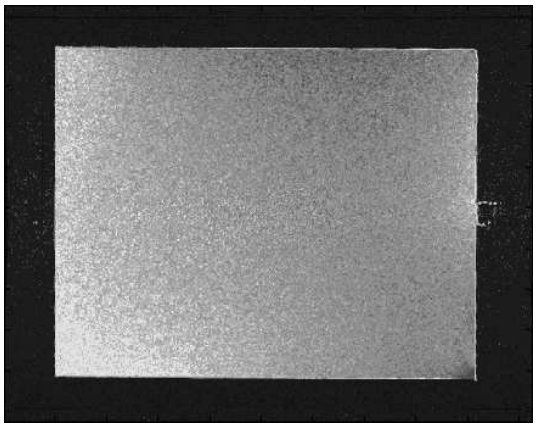
(c) Coronal slice, distorted.



(d) Coronal slice, corrected.



(e) Sagittal slice, distorted.



(f) Sagittal slice, corrected.

Fig. 6.16: Axial, coronal, and sagittal images from the 11-6-05 study, before and after distortion correction.

The November 6, 2005 study represents a properly centered data set. Recall that the foot of the phantom was too large to fit properly in the headcoil, resulting in a 12 - 13 mm displacement of the phantom away from the gradient isocenter. The resulting displacement adversely affected the numerical accuracy of the distortion correction. After modifying the phantom's foot, three new sequences were acquired. Typical axial, coronal, and sagittal slices from the 11-6-05 study are shown in Figure 6.15.

The most apparent feature of the images acquired in the 11-6-05 study is that the image of the phantom is centered in the slices. This clearly indicates that the phantom was successfully centered in all three dimensions with respect to the gradient isocenter. Having this setup created a data set that exhibits symmetric distortion about the center of the images—exactly what the software assumes. The numerical results confirm that the phantom was placed to within one millimeter of gradient isocenter in all three dimensions. Figures 6.16(a), 6.16(c), and 6.16(e) clearly demonstrate highly symmetric distortion. Gone are the extreme curvatures present in the coronal and sagittal images from the 7-17-05 and 10-2-05 studies.

The excellent phantom centering in the 11-6-05 study is directly reflected in the corrected images, Figures 6.16(b), 6.16(d), and 6.16(f). All the corrected images appear to be very consistent in shape and contour.

However, there remains the issue of calibrating the shim coils. Section 7.1.2 outlines the issues discussed in this study surrounding the calibration of the shim coils. The effects of improper shim coil adjustment are apparent in the corrected axial and coronal images, Figures 6.16(b) and 6.16(d). Observe the lower left portions in each image, and notice the asymmetry of the phantom. As was the case in the 7-17-05

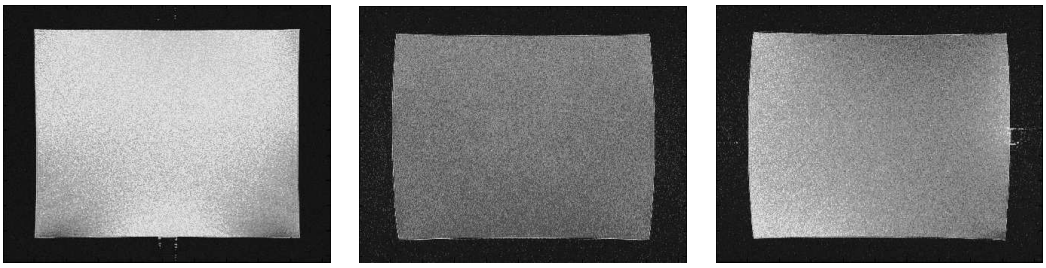
study, the horizontal faces of the phantom in the axial and coronal images are not parallel, giving the false impression that the phantom is not cubic, but asymmetrical. This extra distortion resulting from not calibrating the shim coils prior to scanning reduces the accuracy of the distortion correction.

Besides the extra distortion that is present in the axial and coronal images, the rest of the edges are very straight. Unlike the edges seen in the corrected images from the 7-17-05 and 10-2-05 studies, the edges of the corrected images from the 11-6-05 study do not exhibit any curvature that is indicative of the original distortion. The quality of the corrected images in the 11-6-05 study directly reflects the good numerical results observed after calculating the distortion correction. The corrected images from this study are undoubtedly superior to those corrected in the 7-17-05 and 10-2-05 studies.

Both the quantitative and qualitative results demonstrate that it is possible to use this study for calculating distortion correction parameters. The corrected images are very high quality, and provide an excellent representation of the phantom as it would appear in the absence of gradient nonlinearity distortion. The distortion correction parameters that were calculated from this study accurately reflect the distortion inherent in the scanner. Even without calibration of the shim coils, the quantitative and qualitative results are still quite good. Because of this, the distortion correction model calculated could be confidently applied to a data set, in order to achieve submillimeter accuracy in target localization. However, it would still be highly recommended to perform calibration of the shim coils and reposition the phantom in this exact manner, to achieve the highest accuracy possible.

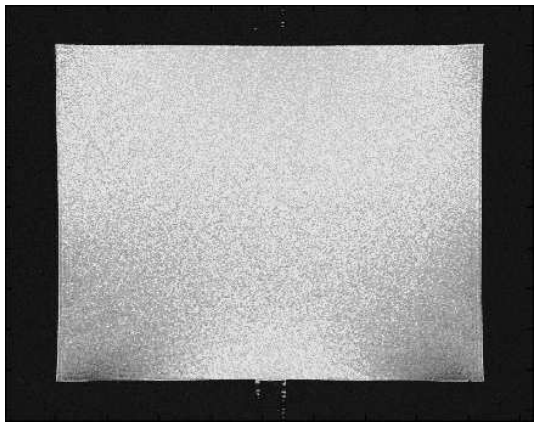
Drawing conclusions from the quantitative and qualitative results from the 11-6-05 study show that this data set is adequate for use in calculating the parameters for the distortion correction model. The parameters calculated sufficiently represent the distortion inherent in the scanner. Since the configuration of the shim coils is unknown in this scan, it is impossible to determine if there are any additional distortions or artifacts that are present in the data set that are not immediately noticeable to the eye. But, the excellent centering of the phantom in the scanner ensured that the distortion in the images was as symmetric as possible, increasing the overall accuracy of the correction method. Even with the extra distortion caused by improper adjustment of the shim coils, the images are still of high enough quality to produce accurate correction results. Therefore, the data acquired in the 11-6-05 study could be used with confidence to calculate a distortion correction accurate to less than one millimeter, for target localization.

6.3.4 4-30-06 Study

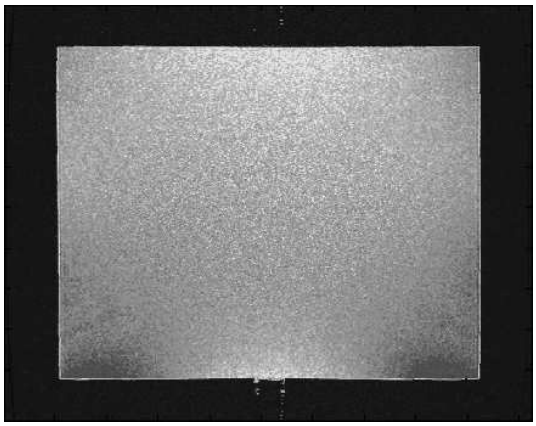


(a) Axial slice. (b) Coronal slice. (c) Sagittal slice.

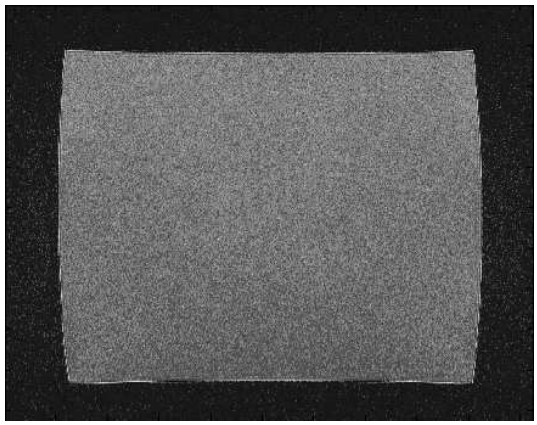
Fig. 6.17: Example images from the 4-30-06 study.



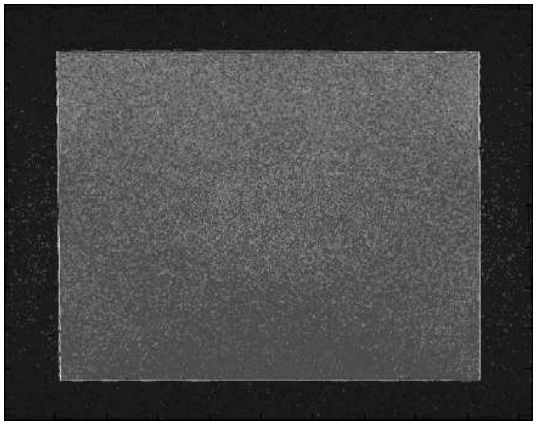
(a) Axial slice, distorted.



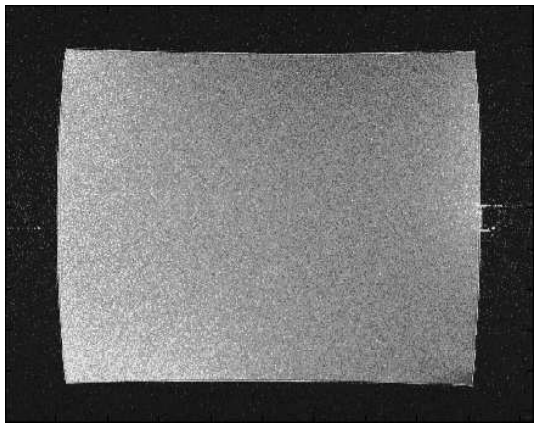
(b) Axial slice, corrected.



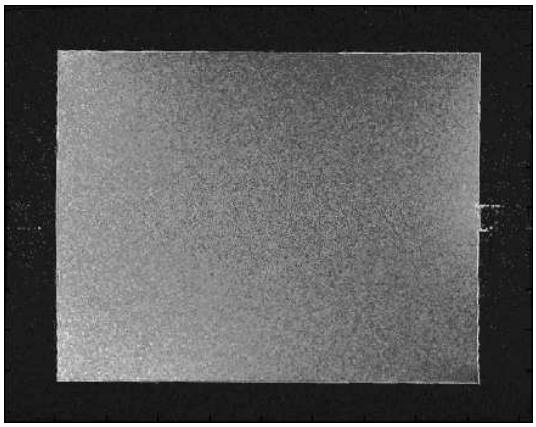
(c) Coronal slice, distorted.



(d) Coronal slice, corrected.



(e) Sagittal slice, distorted.



(f) Sagittal slice, corrected.

Fig. 6.18: Axial, coronal, and sagittal images from the 4-30-06 study, before and after distortion correction.

After the successful phantom centering achieved in the November 6, 2005, the goal was to reproduce the results in the next study. In addition to achieving centering the phantom less than one millimeter in all three dimensions with respect to the gradient isocenter, full calibration of the shim coils was also performed. Calibrating the shim coils was shown to remove the extra distortion present in the bottom left portions of axial and coronal images. The 4-30-06 represents the attempt to achieve both of these experimental setup conditions. Typical axial, coronal, and sagittal slices from the 4-30-06 study are shown in Figure 6.17.

Upon initial observation, it seems that the phantom was centered nearly as well as in the 11-6-05 study. The distortion present in all the images shown in Figure 6.17 appears symmetric. However, analyzing the numerical data shows that the phantom was slightly offcentered in z. The phantom was centered to within 0.4 mm of isocenter in x, and was nearly dead-center in y. The extra displacement in z is slightly noticeable in the images, although the effect is not as great as those seen in the 7-17-05 and 10-2-05 studies, where the phantom was displaced by a distance ten times larger. Figures 6.18(a), 6.18(c), and 6.18(e) demonstrate the symmetry in the gradient nonlinearity distortion.

The good phantom centering and the proper calibration of the shim coils in the 4-30-06 study are directly reflected in the corrected images, Figures 6.18(b), 6.18(d), and 6.18(f). All the corrected images appear to be very consistent in shape and contour.

However, the edges do not appear as straight as those observed in the 11-6-06 study. All the corrected images exhibit slight curvatures. Interestingly enough, the

bottom edge in the axial image, Figure 6.18(b), is quite noticeable, and follows the same general contour of the original distorted image shown in Figure 6.18(a). The curvatures of the corrected coronal and sagittal images, Figures 6.18(d) and 6.18(f), are not as noticeable, but are definitely present. In all three cases, the curvature in the edges reduces the accuracy of the distortion correction method.

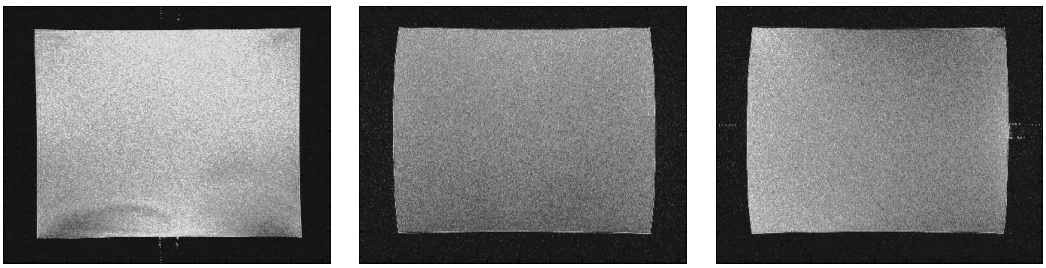
The good news is that the slanted edges in the axial and coronal images are finally absent. The lack of proper shim coil adjustment in the 7-17-05 and 11-6-05 studies produced an asymmetry in the images that could not be removed by the distortion correction. The result was a phantom that appeared to no longer be cubic, but highly asymmetric. The 4-30-06 study, like the 10-2-05 study, shows no signs of this asymmetry. But unlike that study, the shim coils are assured to be correctly calibrated specifically to the phantom in the 4-30-06 study, thereby increasing confidence in the results.

The quantitative and qualitative results demonstrate that it is possible to use this study for calculating distortion correction parameters. The corrected images are high quality, and provide an excellent representation of the phantom as it would appear in the absence of gradient nonlinearity distortion. The distortion correction parameters that were calculated from this study do a reasonable job of reflecting the distortion inherent in the scanner. Even with the slightly higher center offset in z, the quantitative and qualitative results are still quite good. Because of this, the distortion correction model calculated from this study could be confidently applied to other data sets, in order to achieve submillimeter accuracy in target localization. However, it would still be highly recommended to recenter the phantom, to position

the phantom less than one millimeter from gradient isocenter in all three dimensions.

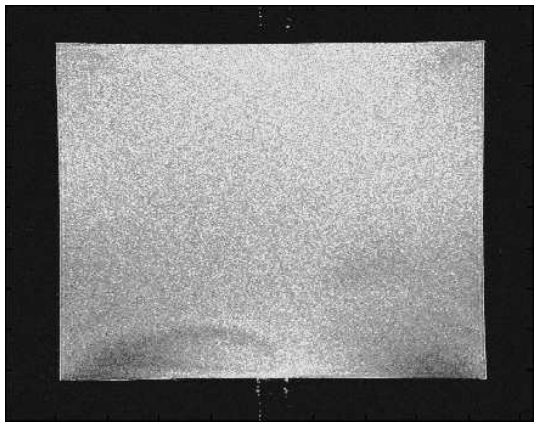
Making conclusions from the quantitative and qualitative results from the 4-30-06 study show that this data set is adequate for use in calculating the parameters for the distortion correction model. The parameters calculated sufficiently represent the distortion inherent in the scanner. Properly configuring the shim coils prior to scanning ensures that the gradient field in the scanner is as homogeneous as possible, increasing the symmetry of the distortion observed in the acquired images. The satisfactory centering of the phantom in the scanner helped ensure that the distortion in the images was as symmetric as possible, helping preserve the overall accuracy of the correction method. Even with the slight asymmetries present as a result of the higher z offcentering, the images are still of high enough quality to produce accurate correction results. Therefore, the data acquired in the 4-30-06 study could be used with confidence to calculate a distortion correction accurate to less than one millimeter, for the purposes of target localization.

6.3.5 6-19-06 Study

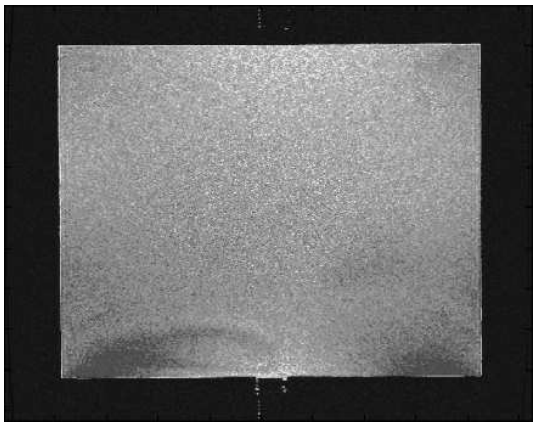


(a) Axial slice. (b) Coronal slice. (c) Sagittal slice.

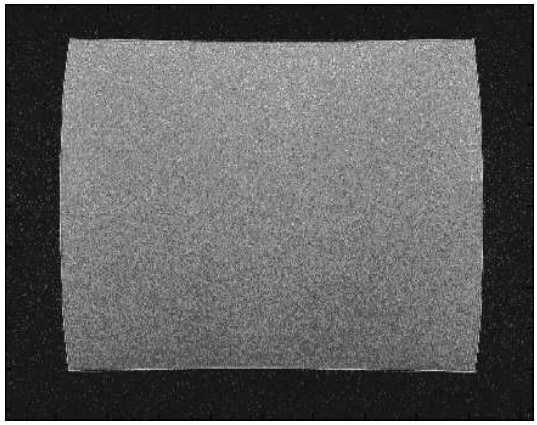
Fig. 6.19: Example images from the 6-19-06 study.



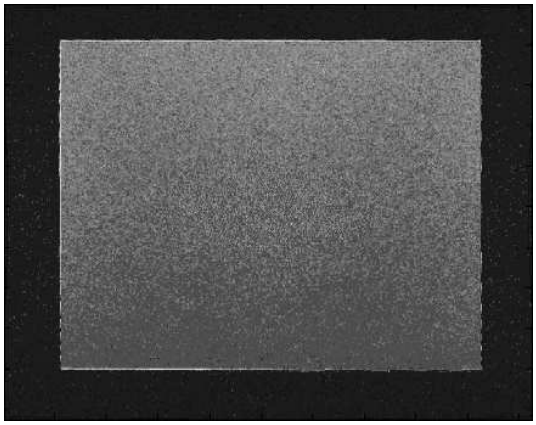
(a) Axial slice, distorted.



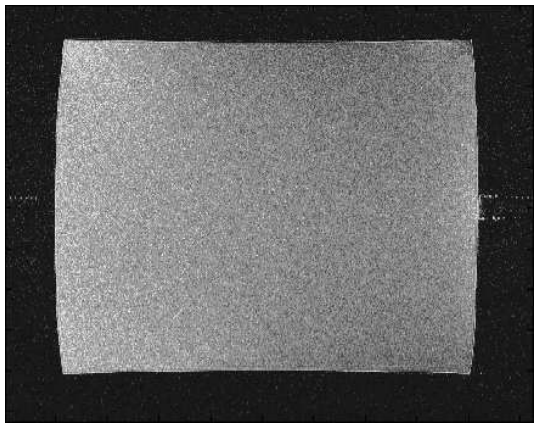
(b) Axial slice, corrected.



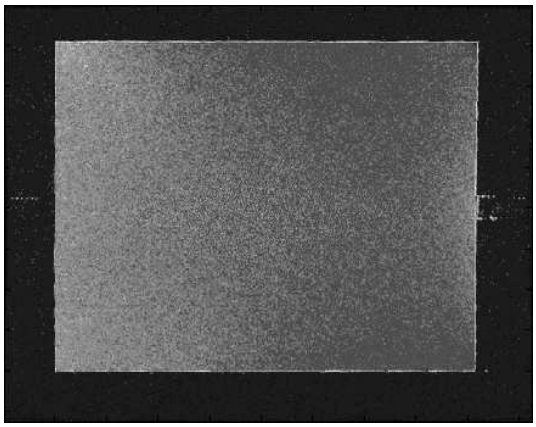
(c) Coronal slice, distorted.



(d) Coronal slice, corrected.



(e) Sagittal slice, distorted.



(f) Sagittal slice, corrected.

Fig. 6.20: Axial, coronal, and sagittal images from the 6-19-06 study, before and after distortion correction.

Once the results of the 4-30-06 study were analyzed, and it was shown that the phantom was once again not centered to within one millimeter of gradient isocenter, it was decided to inscribe straight lines on the anterior face of the phantom. These lines were used in conjunction with the scanner's laser sight, to increase the confidence of placing the phantom in the headcoil prior to scanning. Testing this modification was accomplished in the 6-19-06 study. As was the case in the 4-30-06 study, calibrating the shim coils was performed in order to remove the extra distortion present in the bottom left portions of axial and coronal images. Typical axial, coronal, and sagittal slices from the 6-19-06 study are shown in Figure 6.19.

Upon initial observation, the images seem to indicate that the phantom was centered nearly as well as in the 11-6-05 study. The distortion present in all the images shown in Figure 6.19 appears symmetric. However, analyzing the numerical data shows that the phantom was slightly offcentered in all three dimensions. The phantom was centered to within 1 mm of isocenter in x, 0.9 mm in y, and at least 3 mm in z. The extra displacement in all three dimensions is slightly noticeable in the images, although the effect is not as great as those seen in the 7-17-05 and 10-2-05 studies, where the phantom was displaced by much greater distances. Figures 6.20(a), 6.20(c), and 6.20(e) demonstrate the symmetry in the gradient nonlinearity distortion. The satisfactory phantom centering and the proper calibration of the shim coils in the 6-19-06 study are reflected in the corrected images, Figures 6.20(b), 6.20(d), and 6.20(f). All the corrected images appear to be consistent in shape and contour.

Once again, as was the case in the 4-30-06 study, the edges do not appear as straight as those observed in the 11-6-05 study. All the corrected images exhibit slight

curvatures. Interestingly enough, the bottom edge in the axial image, Figure 6.20(b), is quite noticeable, and follows the same general contour of the original distorted image shown in Figure 6.20(a). This curvature is even more pronounced here than is in the axial image of the 4-30-06 study, shown in Figure 6.18(b). Unlike the 4-30-06 study, the curvatures of the corrected coronal and sagittal images, Figures 6.20(d) and 6.20(f), are hardly noticeable. In all three cases, the curvature in the edges does reduce the accuracy of the distortion correction method.

It is indeed encouraging to see that the slanted edges in the axial and coronal images are finally absent. The 6-19-06 study, like the 10-2-05 and 4-30-06 studies, shows no signs of this asymmetry. But unlike 10-2-05 study, the shim coils are assured to be correctly calibrated specifically to the phantom in the 6-19-06 study, thereby increasing confidence in the results. The same holds true in the 4-30-06 study, which is (unfortunately) the only other study included that reflects proper calibration of the shim coils.

The quantitative and qualitative results demonstrate that it is indeed possible to use this study for calculating distortion correction parameters. The corrected images are high quality, and provide an excellent representation of the phantom as it would appear in the absence of gradient nonlinearity distortion. The distortion correction parameters that were calculated from this study do a good job of reflecting the distortion inherent in the scanner. Even with the slightly higher center offsets in all three dimensions, the quantitative and qualitative results are still quite good. In fact, the numerical errors calculated in this study are the smallest observed in all of the studies. Because of this, the distortion correction model calculated from this study

could be applied with confidence to other data sets, in order to achieve submillimeter accuracy in target localization. However, it would still be highly recommended to recenter the phantom, to position the phantom less than one millimeter from gradient isocenter in all three dimensions.

The conclusions drawn from the quantitative and qualitative results from the 6-19-06 study show that this data set is indeed suitable for use in calculating the parameters for the distortion correction model. The parameters calculated sufficiently represent the distortion inherent in the scanner. Properly configuring the shim coils prior to scanning ensured that the gradient field in the scanner was as homogeneous as possible, increasing the symmetry of the distortion observed in the acquired images. The satisfactory centering of the phantom in the scanner helped ensure that the distortion in the images was as symmetric as possible, helping preserve the overall accuracy of the correction method. Even with the slight asymmetries present as a result of the higher offcentering in x, y, and z, the images are still of high enough quality to produce accurate correction results. Therefore, the data acquired in the 6-19-06 study could be used with confidence to calculate a distortion correction accurate to less than one millimeter, for the purposes of target localization. Considering the corrected images produced by all five studies, the images produced from the 6-19-06 study are of the highest quality, in terms of consistency and accuracy.

Performing quantitative and qualitative analyses on real data provides a strong metric as to the performance of the software. Sources of error can be identified, and improvements can be made to the experimental setup to obtain higher quality data. In the case of this project, there are many sources of error that influence the accuracy of the distortion correction method. Among these include:

- Centering the phantom in the MRI scanner.
- Quality of the images acquired during scanning.
- Calibrating shim coils prior to scanning.

Analyzing the data produced in each of the five studies conducted demonstrates the extreme difficulty in achieving "the perfect data set." In fact, the "perfect" data set was not achieved in any of the five studies performed. The first two studies, the 7-17-05 and 10-2-05 studies, featured the phantom improperly centered in the scanner, causing the images of the phantom to exhibit highly asymmetric distortion. Additionally, the shim coils were not calibrated prior to scanning, thereby introducing additional distortions in the images. The 11-6-05 study achieved very ideal phantom centering, but was crippled by the extra distortions caused by not calibrating the shim coils before scanning. And finally, the 4-30-06 and 6-19-06 studies featured proper calibration of the shim coils, but the phantom was not centered as well as the 11-6-05 study in the scanner. The imperfections present in each data set makes it very difficult to positively identify the errors that were produced by each attribute in the experimental setup. Therefore, it is very difficult to fully interpret the numerical

results obtained from performing the distortion correction on each study.

So how can this task be accomplished? Simple: run the distortion correction on a perfect data set! By running a data simulation, it is possible to control nearly every aspect of the experimental setup, to isolate each factor and analyze the errors that are produced as a result.[1]Additionally, it is possible to eliminate *all* of the problems experienced in the experimental setup, to achieve the true perfect experimental setup, and therefore the perfect data set: one that is centered perfectly (and therefore achieves total symmetry in the distortion), one that is not subjected to any noise or artifacts as a result of image acquisition and signal sampling, and one that is not influenced by improperly configured shim coils.

In addition to simulating various experimental setups, the simulation also provides a secondary verification of proper operation of the software. Programming mistakes will show up in the error calculations when a perfect data set is used. Errors that cannot be accounted for in the calculations are most likely the results of incorrect code. Examining the numerical results during the simulations will help test the software and ensure that the algorithms are programmed correctly.

6.4.1 Producing a Simulated Data Set

Performing distortion correction calculations on a simulated data set is quite simple and straightforward:

1. Define the distortion model (i.e., choose the coefficients of the spherical harmonic

[1] The distortions caused by improperly calibrated shim coils cannot be simulated here, since the gradients produced by the coils vary depending on their configuration. The investigation into how shim coil configuration affects images is discussed in Section 7.1.2.

expansion that will distort the ideal data set).

2. Determine the phantom's position in the scanner (i.e., the distance the phantom lies from isocenter).

3. Create the matrices and arrays that contain the data points for the ideal data set.

4. Distort the ideal data set using the distortion model defined in Step 1.

5. Perform the distortion correction calculations as normal.

Define the Distortion Model

In the first step of the simulation, the parameters for the distortion model are defined. This model represents the characteristics of the gradient nonlinearity distortion in the scanner, that produce the geometric warping present in each MR image. The theoretical scanner in this simulation is defined here. Theoretically, any set of parameters can be used here; for example, it is possible to use the paramters Langlois [33] lists in his work.

Determine the Phantom's Center Offset

The primary issue when performing each of the five scans was phantom centering. Only one of the five studies achieved ideal phantom centering, where the phantom was placed within one millimeter of gradient isocenter in x, y, and z. The errors produced in each of the five studies reflect phantom positioning to some extent. The purpose of the data simulation is to determine to what extent phantom centering influences the

accuracy of the distortion correction method. If phantom centering is in fact a very serious issue, such that poor centering greatly decreases the accuracy of the correction method, then additional measures must be taken to ensure proper placement of the phantom in the scanner. If phantom centering is shown not to contribute much to numerical accuracy, then less emphasis will be placed on phantom centering in the future. It is possible to recreate each scenario exactly, such that the simulated phantom is displaced by the exact same amount as was observed in each of the five studies.

Creating the Ideal Data Set

Recall that the data used by the distortion correction method is obtained from edge images. The simulated data set created here represents edge image data. To conform with the formats used by the software, the simulation packs the calculated distorted data sets into the same data structures that are used to perform distortion correction calculations. This allows the simulation to function seamlessly in the software.

Distort the Ideal Data Set

Since the original data set is ideal, the distorted data set is therefore also ideal, being that the data points are not subject to noise, artifacts, or small errors as a result of image construction or signal processing. An additional advantage is knowing the original distortion model, since it was defined in the first step. In a real data set, the characteristics of the gradient nonlinearity distortion inherent in the scanner are not known. Having this information is a great advantage in performing data analysis

techniques.

Perform Gradient Nonlinearity Distortion Correction

Once the simulated data set is packed and distorted, it can then be processed by the software correction method in the same manner as real data. By producing the simulated data, only plane calculation and the actual distortion correction need to be executed. The initial stage of execution—data preprocessing and data quality assurance—is not necessary, since the data is perfect and does not need to be subjected to quality assurance tests.

6.4.2 Data Simulations

For each simulation, the phantom is defined to be a perfect cube of 160 mm. Therefore, a perfectly centered phantom will be reflected in the offset terms in the ideal plane equations as values of $\pm$ 80, depending on the location of the plane with respect to the coordinate axes. To promote consistency across the trials, each data set produced will be distorted using the distortion parameters calculated from the unfiltered 6-19-06 study, shown in Table 6.23.

	K_x	K_y	K_z
K_0	8.2819×10^{-7}	1.8569×10^{-7}	4.9965×10^{-6}
K_1	-7.4615×10^{-6}	-5.7053×10^{-6}	-2.7831×10^{-6}
K_2	2.273×10^{-10}	1.7915×10^{-10}	-2.9354×10^{-10}
K_3	3.7395×10^{-11}	9.2343×10^{-11}	-3.3674×10^{-11}
K_4	-1.0008×10^{-10}	-1.9753×10^{-10}	2.8046×10^{-10}

Tab. 6.23: Coefficients of distortion model used for all trials.

Trial # 1

The first trial simulates a perfectly centered phantom. The data set produced therefore exhibits perfectly symmetric distortion. Performing the distortion correction method on the data set created for Trial #1 yields the results displayed in Tables 6.24 - 6.27.

Midplane	**Plane Coefficients**
X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{0}, 0, 0]$
Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [0, \mathbf{0}, 0]$
Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [0, 0, \mathbf{0}]$

Tab. 6.24: Plane coefficients of midplanes fitting in Trial #1.

Plane	**Plane Coefficients**
-X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{-80}, 0, 0]$
+X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{80}, 0, 0]$
-Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [0, \mathbf{-80}, 0]$
+Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [0, \mathbf{80}, 0]$
-Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [0, 0, \mathbf{-80}]$
+Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [0, 0, \mathbf{80}]$

Tab. 6.25: Plane coefficients of ideal planes fitting in Trial #1.

Data Analysis

The bolded values in the plane coefficients calculated from the simulated data set, Tables 6.24 and 6.25, confirm that the data set is indeed centered perfectly with respect to gradient isocenter. Therefore the distortion inherent in the data is confirmed to be perfectly symmetric. The zero values in the plane coefficients also demonstrate the characteristics of the simulated phantom:

# Data Points Used (X)	66420
# Data Points Used (Y)	66420
# Data Points Used (Z)	66420
Total Residuals (X) (mm)	0
Total Residuals (Y) (mm)	0
Total Residuals (Z) (mm)	0
Variance (X) (mm)	0
Variance (Y) (mm)	0
Variance (Z) (mm)	0
Standard Deviation (X) (mm)	0
Standard Deviation (Y) (mm)	0
Standard Deviation (Z) (mm)	0

Tab. 6.26: Errors calculated from plane fitting results from Trial #1.

- The data is free from noise and artifacts.
- The simulated phantom is constructed perfectly, and does not contain any defects.
- The data is not subject to any errors as a result of signal processing and/or image construction.

The three items mentioned above are all inherent characteristics in a real data set acquired from MR images. Even the highest quality phantoms and the highest quality MRI scanners will not have the capability of totally eliminating these factors. As a result, these factors contribute to the errors observed in the data calculations. Therefore, any real data set that is processed by this software will be subjected to

these errors, since these errors are impossible to avoid.

The most important conclusion that can be drawn from the data in Tables 6.24, 6.25, and 6.26 is that the midplane and ideal plane algorithms do not introduce any errors into the calculations. The midplane and ideal plane coefficients, along with the zero values for the errors in Table 6.26, all indicate that the midplane and ideal plane algorithms do not produce any errors as a result of incorrect calculations. It is safe to conclude that the algorithms are programmed correctly, since the plane results are what would be expected given the characteristics of the simulation, and that errors are nonexistent.

Analyzing the data results of the distortion correction in Table 6.27 provides insight into the numerical accuracy of the distortion correction method. Recall back to Chapter 2 that one of the primary goals of this software is to preserve numerical accuracy in the correction method as much as possible. Regardless of the quality of the data and the experimental setup, if the distortion correction algorithms introduce large numerical errors as a result of data processing, the correction method is essentially useless.

Fortunately, the data simulation provides a verification of the numerical accuracies of the distortion correction method. Observing the size of the errors in Table 6.27 shows that the distortion correction algorithms introduce very small errors into the calculations. Recall that the spherical harmonics expressions used in this distortion correction method are themselves subject to error, since they are truncated versions of infinite expressions. Therefore, the errors that are displayed here only represent inaccuracies as a result of:

- Truncation errors from using the truncated spherical harmonics expansions.
- Rounding errors.
- Floating point errors.

Rounding and floating point errors are simply a property of every computer system, because no computer has an infinite amount of storage space with which to store the results of numerical computations.

Keeping these factors in mind helps interpret the results in Table 6.27. Notice the magnitude of the errors shown here. Keep in mind that the errors here represent the difference between the original simulated phantom data set and the corrected distorted data set. Errors on the order of 10^{-4} and 10^{-5} mm are very good indeed. Truncation, rounding, and floating point errors are totally unavoidable, regardless of the algorithm implementation. Realizing that these errors are only of this magnitude increases confidence in the results presented in Section 6.2. It is safe to conclude that the distortion correction algorithms implemented for this software do a very good job of preserving numerical accuracy, therefore assuring confidence in the correction results.

The results obtained from Trial #1 serve as a benchmark with which to compare other simulations. These results serve as the "perfect" results, since the phantom and experimental setup simulated here are perfect. The accuracy demonstrated here represent the highest achievable accuracy in this correction method.

Trial # 2

The second trial simulates a slightly offcentered phantom, one that is offcentered by the same amount as calculated in the 4-30-06 study. The observed center offset in the 4-30-06 study was 0.4 mm in x, essentially centered in y, and 1.9 mm in z. Performing the distortion correction method on the data set created for Trial #2 yields the results displayed in Tables 6.28 - 6.31.

Data Analysis

The bolded values in the ideal plane coefficients calculated from the simulated data set, Table 6.29, confirms the simulated phantom is offcentered with respect to gradient isocenter by nearly the same amount in x, y, and z as the 4-30-06 study. Therefore the distortion present in the simulated data is ensured to be similar to that inherent in the real data set. However, it is important to remember that the results given here will not be exactly the same as those observed in the real data set. Just as with Trial #1, the simulated data has the following characteristics:

- The data is free from noise and artifacts.
- The simulated phantom is constructed perfectly, and does not contain any defects.
- The data is not subject to any errors as a result of signal processing and/or image construction.

The three items mentioned above are all inherent characteristics in a real data set acquired from MR images. Even the highest quality phantoms and the highest quality

MRI scanners will not have the capability of eliminating these factors. As a result, these factors contribute to the errors observed in the data calculations. Therefore, any real data set that is processed by this software will be subjected to these errors, since these errors are impossible to avoid.

Compare the differences in magnitude of the numerical results between the simulated data set (Tables 6.28 - 6.31) and the real acquired data set from 4-30-06 (Tables 6.13 - 6.16). The standard deviation values of the plane fitting and distortion correction results for the test data are between one and two orders of magnitude smaller than those observed in the real data set. The larger errors observed in the real data set represent errors associated with the presence of the three factors mentioned above. Unfortunately, it is impossible to quantify exactly how much each of these factors contributes to the size of the errors in each study. Keep in mind that the acquired images represent a series of approximations, as the signals are sampled and the images are constructed. Unfortunately, it is very difficult to accurately identify and quantify each of these sources of error. Even if these sources of error were identified, little could be done to improve upon these errors, since these factors are controlled by the MRI scanner, and tweaking the scanner to improve these errors is a difficult task beyond the scope of this work. Regardless, the signal and image processing techniques used by the scanner do contribute to the errors observed in the real data set, as is easily identifiable by these results.

This test features a phantom that is slightly offcentered. Therefore, the distortion inherent in the data is slightly asymmetric. Compare the differences in magnitude of the numerical results between this trial (Tables 6.28 - 6.31) and the first trial with the

perfectly centered phantom (Tables 6.24 - 6.27). The standard deviations that are obtained in the second trial are between two and three orders of magnitude larger than those observed in the first trial. This increased error is strictly from the offcentering of the phantom, and therefore the asymmetry in the inherent distortion in the data set. The great increase in the magnitude of the errors indicates the sensitivity of the distortion correction method to phantom centering. Although the standard deviations are still one to two orders of magnitude smaller than those observed in real data sets, the substantial increase in size of the standard deviations is still clearly noticeable.

The centering simulated in this trial is similar to the setup achieved in the 11-6-05, 4-30-06, and 6-19-06 studies. It represents a suitable median for the center offset: the 11-6-05 study achieved closer centering than the simulation, and the centering in the 6-19-06 study was not as close as the simulation. This trial shows that slight offcentering (approximately 2 mm at the most) noticeably decreases the accuracy of the distortion correction, but still provides valid results. The errors observed in this simulation (on the order of 10^{-2} to 10^{-3} mm) provide a more realistic representation of the accuracy of the distortion correction method. The phantom centering simulated in this trial is very realistic, and will most likely be a common occurrence in future trials.

Trial # 3

The third trial simulates a very offcentered phantom, one that is offcentered by a similar amount as calculated in the 7-17-05 and 10-2-05 studies. The observed center offset in these studies was approximately 1.8 mm in x, 1.0 mm in y, and 13.2 mm in

z. Performing the distortion correction method on the data set created for Trial #3 yields the results displayed in Tables 6.32 - 6.35.

Data Analysis

The bolded values in the ideal plane coefficients calculated from the simulated data set, Table 6.33, confirms the simulated phantom is offcentered with respect to gradient isocenter by the same amount in x, y, and z as the 7-17-05 and 10-2-05 studies. Therefore the distortion present in the simulated data is confirmed to be similar to that inherent in the real data sets. Again, it is important to remember that the results given here will not be exactly the same as those observed in the real data sets. Just as with Trials #1 and #2, the simulated data has the following characteristics:

- The data is free from noise and artifacts.
- The simulated phantom is constructed perfectly, and does not contain any defects.
- The data is not subject to any errors as a result of signal processing and/or image construction.

The three items mentioned above are all inherent characteristics in a real data set acquired from MR images. Even the highest quality phantoms and the highest quality MRI scanners will not have the capability of eliminating these factors. As a result, these factors contribute to the errors observed in the data calculations. Therefore, any real data set that is processed by this software will be subjected to these errors, since these errors are impossible to avoid.

Compare the differences between the simulated data set (Tables 6.32 - 6.35) and the real acquired data set from 10-2-05 (Tables 6.5 - 6.8). The magnitudes of the standard deviations obtained in the plane fitting and distortion correction results are only one order of magnitude smaller in the simulated data set than in the real data set. Recall back to the previous trials, where the standard deviations of the correction results were on the average between two and three orders of magnitude smaller than those observed in the real data sets. Clearly the great center offset is degrading the accuracy of the correction results. Even though the standard deviations are much higher in this simulation than in the first simulation with the perfectly centered phantom, the correction results still seem within reasonable bounds.

Because this test features a phantom that is quite offcentered, the distortion inherent in the data is quite asymmetric. Compare the differences in magnitude of the numerical results between this trial (Tables 6.32 - 6.35) and the first trial with the perfectly centered phantom (Tables 6.24 - 6.27). The standard deviations that are obtained in the third trial are nearly four orders of magnitude larger than those observed in the first trial. This *very* large increase in the size of the errors is strictly from the offcentering of the phantom, and therefore the asymmetry in the inherent distortion in the data set. This apparent increase in the magnitude of the errors is a second indication of the sensitivity of the distortion correction method to phantom centering. Indeed, the standard deviations of the correction results are smaller than those observed in the real data set, but not much smaller.

The centering simulated in this trial is similar to the setup achieved in the 7-17-05 and 10-2-05 studies. The great offset in z of over 13 mm greatly increases

the size of the errors in the distortion correction results. Analyzing these results, and also considering the results obtained in the 7-17-05 and 10-2-05 studies, further demonstrate the sensitivity of the distortion correction method to phantom centering in the scanner.

6.4.3 Conclusions

It is very apparent that the accuracy of the distortion correction method is dependent on how well the phantom is centered in the scanner. The three simulations conducted demonstrate three important conclusions:

1. The distortion correction method itself is accurate, and does not introduce great errors into the results.

2. There are errors associated with the signal and image processing techniques used by the MRI scanner.

3. Improper phantom centering increases the size of the errors in the distortion correction, thereby degrading the accuracy of the results.

The quality of the distortion correction method was demonstrated with the first trial. Recall that the first trial created the perfect experimental setup, with a perfectly sized, perfectly centered phantom. Comparing the corrected data points to the original undistorted data points resulted in errors of approximately 10^{-5} in magnitude, quite small indeed considering that the units of measurement are millimeters. These results solidify confidence in the validity of the results generated by the correction method. The errors observed in the correction results of real data sets are

primarily from other sources, among these include the signal and image processing techniques of the scanner and phantom centering.

The method in which the scanner samples the electronic signals, processes the signals, and constructs images is, essentially, fixed. It may be possible to tweak the scanner's configurations to improve these methods, to increase accuracy, but the amount of work, research, and testing required to achieve such results are entirely beyond the scope of this work. To complicate matters, these efforts may not yield a noticeable improvement, thereby lending little justification required to achieve such results. Therefore, it is best to simply accept these errors as being a natural trait of the system.

Phantom centering, on the other hand, *is* a factor that can be controlled. The second and third trials clearly demonstrated that the accuracy of the correction method is influenced by phantom centering. Slight offcentering increases the size of the errors by two or three orders of magnitude, while severe offcentering increases the errors by four orders of magnitude. As was observed in the five studies performed, centering is an important issue to address. The accuracy of both the simulated and real data sets decreased as a result of improper phantom centering. Unfortunately, it is impossible to center any phantom perfectly in a scanner; therefore it is impossible to achieve the perfect experimental setup in the real world. Because of this unfortunate circumstance, the accuracies demonstrated in the first trial are unobtainable. But this does not mean that all is lost. Putting effort into phantom positioning will indeed benefit the accuracy of the correction results. The goal is therefore to position the phantom *as close as possible* to the center of the scanner, as close to the gradient isocenter.

These tests show that careless placement of the phantom in the scanner adversely affects the accuracy of the distortion correction.

Since perfect phantom centering is not possible, it is best to analyze a more realistic scenario. Trial #2 demonstrated typical, acceptable phantom centering, a setup that would not be an uncommon occurrence in the real world (recall that the centering achieved in the 4-30-06 study was used in this simulation). Therefore, the actual accuracy of the distortion correction in realistic scenarios is best represented here. The correction results exhibited errors on the order of 10^{-2} to 10^{-3} mm. The observed accuracies of the real data set from 4-30-06 were on the order of 10^{-1} mm, demonstrating that other sources of error are not large. The highest accuracy achievable in this scenario is 10^{-3} mm when only considering the asymmetry in the data set and the accuracy inherent to the distortion correction method itself. The errors caused by other sources are only responsible for approximately 10^{-1} mm, which is very promising. This further reinforces the concept of proper phantom centering, and demonstrates the importance of achieving symmetric distortion in the data sets.

A deeper analysis of phantom centering is presented in Section 7.1.1.

One of the key features offered by many MRI scanners is the ability to apply various filters to each MR image, to reduce the effects of distortion (including gradient nonlinearity distortion). Recall back to Section 1.8.1 that the filters applied by the MRI scanners are not true corrections, in that they *reduce* the amount of visible distortion in each image. The filters only apply corrections in two dimensions, unlike the software correction method described in this work. Therefore, using the scanner corrections by themselves will not provide submillimeter accuracy necessary for target localization in functional proton radiosurgery.

It is commonly preferred by physicians to use the scanner filters. The images that are created using the filters appear to have little or no distortion. For visual observation, the filters are excellent. A very important question arises: will the software correction method still work with filtered images, and does it improve upon the scanner filtering?

As stated many times previously, the scanner filtering techniques reduce the amount of distortion visible in each image. The filters perform two-dimensional corrections based on previously-defined characteristics of the magnetic gradients in the scanner. The filtering technique performed on each slice depends on the location of that slice within the scanner bore. Recall that in Section 6.2 and 6.4 that it was mentioned that the larger the distortion in the images, the greater the errors produced. This is because the fit of the spherical harmonics expansions is not as tight in a data set with greater distortion. Taking this into consideration, as well as the fact that the filters reduce the distortion in each image, a second question arises: is it possible to

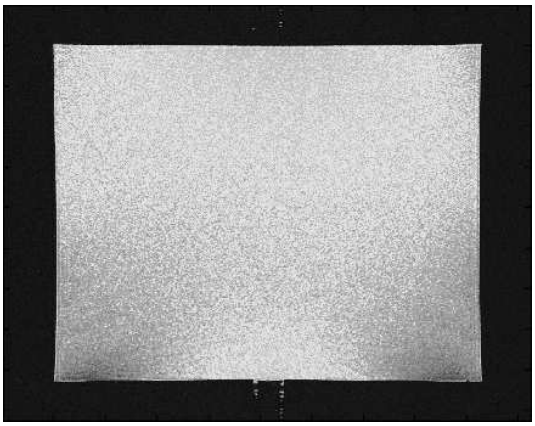
use the scanner filters to improve the quality of the corrected images, and therefore improve the accuracy of target localization?

6.5.1 Testing Filtered and Unfiltered Images

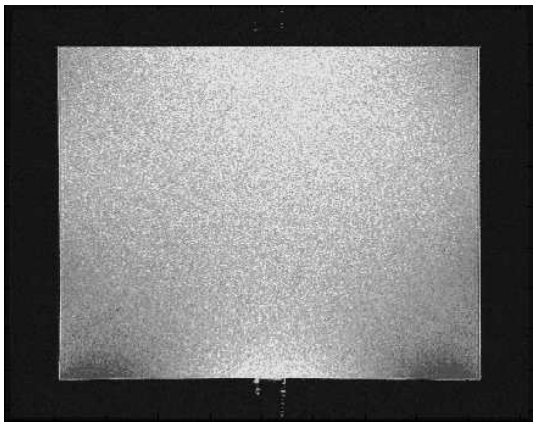
In order to answer both of these questions, two different studies were performed: one on April 30, 2006, and the second on June 19, 2006. The five studies presented in Section 6.2 all featured no scanner correction. To compare the differences between studies featuring scanner filtering and no scanner filtering, the final two studies, the 4-30-06 and 6-19-06 studies, were performed with the scanner filters on and off. The data that was acquired using no filters is presented in Section 6.2.

The benefit of performing the extra sequences during the 4-30-06 and 6-19-06 studies is the fact that the phantom and scanner are configured *exactly* the same in both sequences. In other words, no modifications were made to the experimental setup (besides the application of the filters) between each sequence. Therefore, a direct comparison is possible, since all the other variables in the experimental setup are identical.

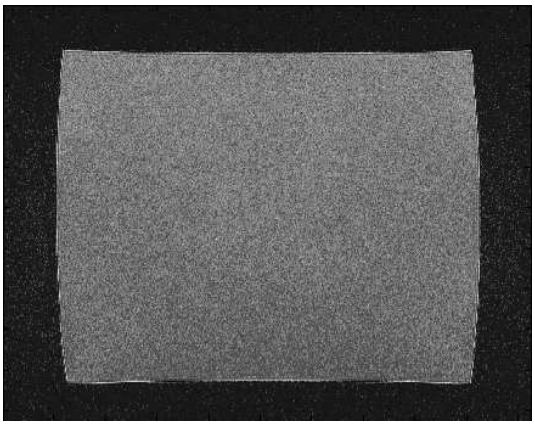
4-30-06 Study (with scanner correction)



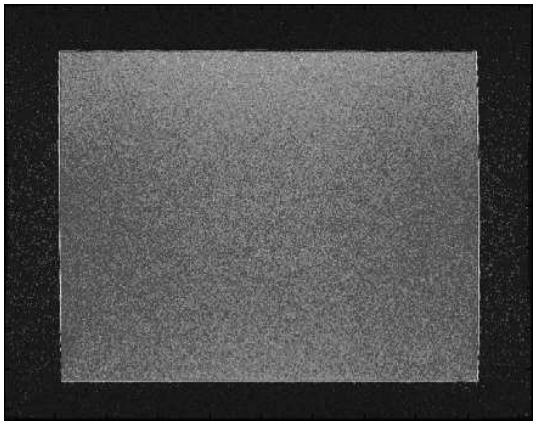
(a) Axial slice, unfiltered.



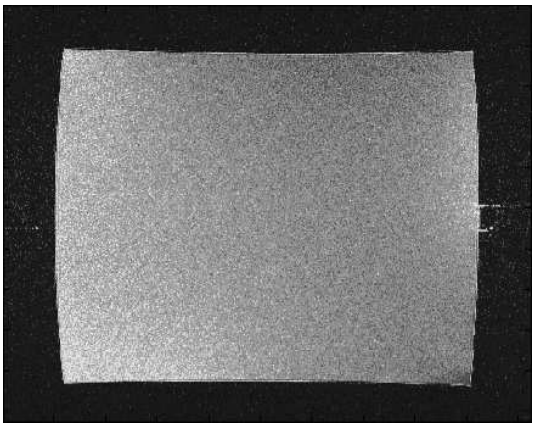
(b) Axial slice, filtered.



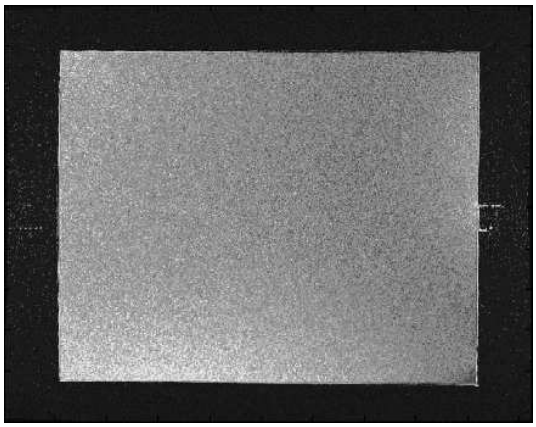
(c) Coronal slice, unfiltered.



(d) Coronal slice, filtered.



(e) Sagittal slice, unfiltered.



(f) Sagittal slice, filtered.

Fig. 6.21: Axial, coronal, and sagittal images from the 4-30-06 study, with and without scanner filtering.

# Data Points Used (-X)	66420
# Data Points Used (+X)	66420
# Data Points Used (-Y)	66420
# Data Points Used (+Y)	66420
# Data Points Used (-Z)	66420
# Data Points Used (+Z)	66420
Total Residuals (-X) (mm)	1.3125
Total Residuals (+X) (mm)	1.3125
Total Residuals (-Y) (mm)	1.3274
Total Residuals (+Y) (mm)	1.3274
Total Residuals (-Z) (mm)	0.019283
Total Residuals (+Z) (mm)	0.019283
Variance (-X) (mm)	2.5934×10^{-5}
Variance (+X) (mm)	2.5934×10^{-5}
Variance (-Y) (mm)	2.6529×10^{-5}
Variance (+Y) (mm)	2.6529×10^{-5}
Variance (-Z) (mm)	5.5984×10^{-9}
Variance (+Z) (mm)	5.5984×10^{-9}
Standard Deviation (-X) (mm)	5.0926×10^{-4}
Standard Deviation (+X) (mm)	5.0926×10^{-4}
Standard Deviation (-Y) (mm)	5.1506×10^{-4}
Standard Deviation (+Y) (mm)	5.1506×10^{-4}
Standard Deviation (-Z) (mm)	7.4822×10^{-5}
Standard Deviation (+Z) (mm)	7.4822×10^{-5}

Tab. 6.27: Errors calculated from distortion correction results from Trial #1.

Midplane	Plane Coefficients
X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{0.41769}, -2.2537 \times 10^{-8}, -5.0236 \times 10^{-6}]$
Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [-1.4296 \times 10^{-7}, \mathbf{-0.06772}, 8.1448 \times 10^{-7}]$
Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [7.8279 \times 10^{-6}, -2.7714 \times 10^{-6}, \mathbf{1.6209}]$

Tab. 6.28: Plane coefficients of midplanes fitting in Trial #2.

Plane	Plane Coefficients
-X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{-79.5823}, -2.2537 \times 10^{-8}, -5.0236 \times 10^{-6}]$
+X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{80.4177}, -2.2537 \times 10^{-8}, -5.0236 \times 10^{-6}]$
-Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [-1.4296 \times 10^{-7}, \mathbf{-80.0677}, 8.1448 \times 10^{-7}]$
+Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [-1.4296 \times 10^{-7}, \mathbf{79.9323}, 8.1448 \times 10^{-7}]$
-Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [7.8279 \times 10^{-6}, -2.7714 \times 10^{-6}, \mathbf{-78.3791}]$
+Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [7.8279 \times 10^{-6}, -2.7714 \times 10^{-6}, \mathbf{81.6209}]$

Tab. 6.29: Plane coefficients of ideal planes fitting in Trial #2.

# Data Points Used (X)	66420
# Data Points Used (Y)	66420
# Data Points Used (Z)	66420
Total Residuals (X) (mm)	0.8884
Total Residuals (Y) (mm)	0.14404
Total Residuals (Z) (mm)	7.3940
Variance (X) (mm)	1.1883×10^{-5}
Variance (Y) (mm)	3.1235×10^{-7}
Variance (Z) (mm)	8.2312×10^{-4}
Standard Deviation (X) (mm)	0.00344710
Standard Deviation (Y) (mm)	0.00055889
Standard Deviation (Z) (mm)	0.02869

Tab. 6.30: Errors calculated from plane fitting results from Trial #2.

# Data Points Used (-X)	66420
# Data Points Used (+X)	66420
# Data Points Used (-Y)	66420
# Data Points Used (+Y)	66420
# Data Points Used (-Z)	66420
# Data Points Used (+Z)	66420
Total Residuals (-X) (mm)	3.2459
Total Residuals (+X) (mm)	3.2492
Total Residuals (-Y) (mm)	2.2858
Total Residuals (+Y) (mm)	2.2800
Total Residuals (-Z) (mm)	0.57173
Total Residuals (+Z) (mm)	0.57540
Variance (-X) (mm)	1.5862×10^{-4}
Variance (+X) (mm)	1.5895×10^{-4}
Variance (-Y) (mm)	7.8665×10^{-5}
Variance (+Y) (mm)	7.8267×10^{-5}
Variance (-Z) (mm)	4.9213×10^{-6}
Variance (+Z) (mm)	4.9847×10^{-6}
Standard Deviation (-X) (mm)	0.012595
Standard Deviation (+X) (mm)	0.012607
Standard Deviation (-Y) (mm)	0.0088693
Standard Deviation (+Y) (mm)	0.0088469
Standard Deviation (-Z) (mm)	0.0022184
Standard Deviation (+Z) (mm)	0.0022326

Tab. 6.31: Errors calculated from distortion correction results from Trial #2.

Midplane	Plane Coefficients
X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{1.7949}, -1.0415 \times 10^{-5}, 0.00018361]$
Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [-1.0684 \times 10^{-5}, \mathbf{-1.0276}, -0.00010512]$
Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [-0.00030979, 0.00019398, \mathbf{-13.0567}]$

Tab. 6.32: Plane coefficients of midplanes fitting in Trial #3.

Plane	Plane Coefficients
-X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{-78.2051}, -1.0415 \times 10^{-5}, 0.00018361]$
+X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{81.7949}, -1.0415 \times 10^{-5}, 0.00018361]$
-Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [-1.0684 \times 10^{-5}, \mathbf{-81.0276}, -0.00010512]$
+Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [-1.0684 \times 10^{-5}, \mathbf{78.9724}, -0.00010512]$
-Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [-0.00030979, 0.00019398, \mathbf{-93.0567}]$
+Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [-0.00030979, 0.00019398, \mathbf{66.9433}]$

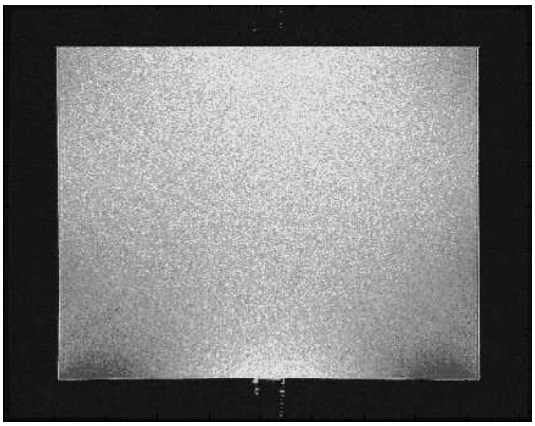
Tab. 6.33: Plane coefficients of ideal planes fitting in Trial #3.

# Data Points Used (X)	66420
# Data Points Used (Y)	66420
# Data Points Used (Z)	66420
Total Residuals (X) (mm)	3.9756
Total Residuals (Y) (mm)	2.2762
Total Residuals (Z) (mm)	60.3309
Variance (X) (mm)	2.3796×10^{-4}
Variance (Y) (mm)	7.8005×10^{-5}
Variance (Z) (mm)	5.4800×10^{-2}
Standard Deviation (X) (mm)	0.015426
Standard Deviation (Y) (mm)	0.0088321
Standard Deviation (Z) (mm)	0.23409

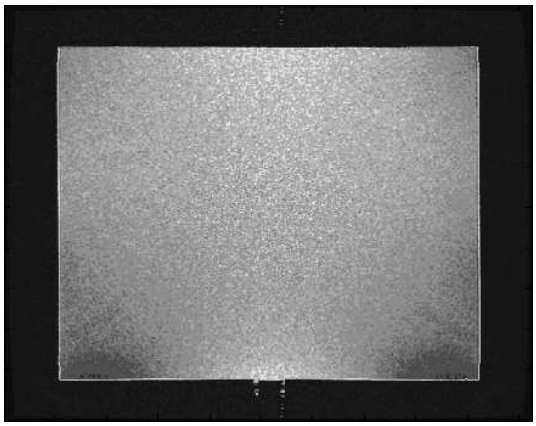
Tab. 6.34: Errors calculated from plane fitting results from Trial #3.

# Data Points Used (-X)	66420
# Data Points Used (+X)	66420
# Data Points Used (-Y)	66420
# Data Points Used (+Y)	66420
# Data Points Used (-Z)	66420
# Data Points Used (+Z)	66420
Total Residuals (-X) (mm)	11.0700
Total Residuals (+X) (mm)	10.7783
Total Residuals (-Y) (mm)	7.3615
Total Residuals (+Y) (mm)	7.3510
Total Residuals (-Z) (mm)	5.2100
Total Residuals (+Z) (mm)	5.8454
Variance (-X) (mm)	0.001845
Variance (+X) (mm)	0.001749
Variance (-Y) (mm)	0.00081589
Variance (+Y) (mm)	0.00081356
Variance (-Z) (mm)	0.00040868
Variance (+Z) (mm)	0.00051443
Standard Deviation (-X) (mm)	0.042953
Standard Deviation (+X) (mm)	0.041821
Standard Deviation (-Y) (mm)	0.028564
Standard Deviation (+Y) (mm)	0.028523
Standard Deviation (-Z) (mm)	0.020216
Standard Deviation (+Z) (mm)	0.022681

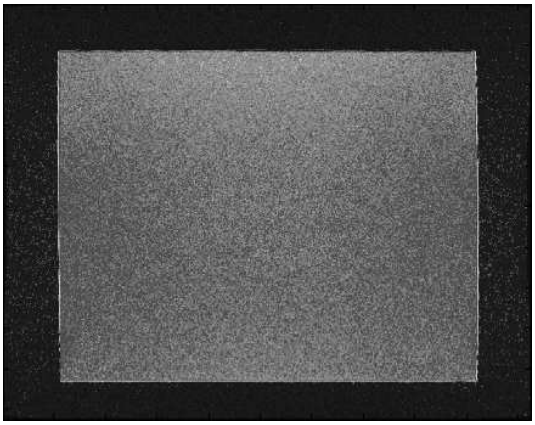
Tab. 6.35: Errors calculated from distortion correction results from Trial #3.



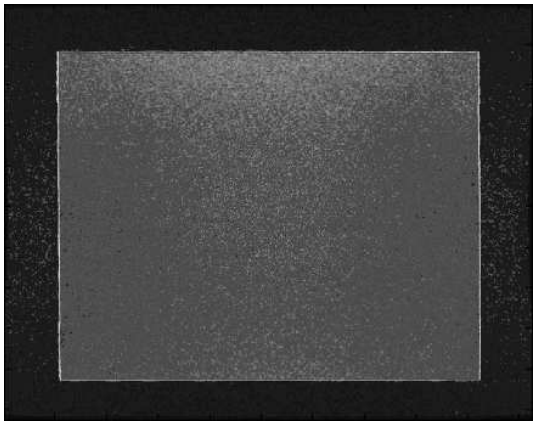
(a) Axial slice, distorted.



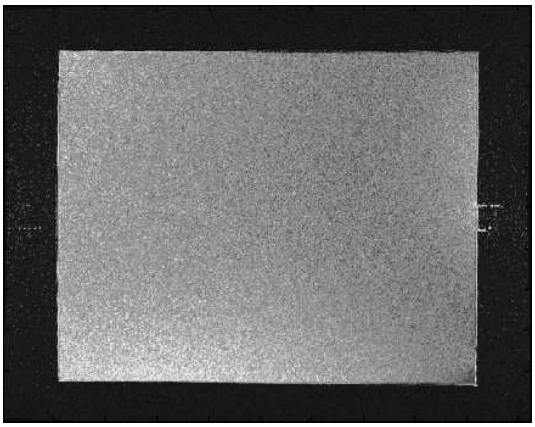
(b) Axial slice, corrected.



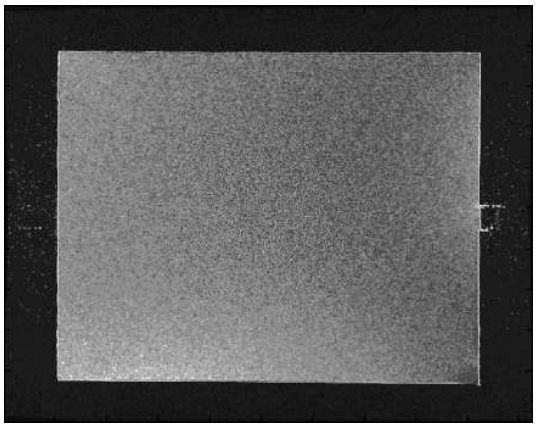
(c) Coronal slice, distorted.



(d) Coronal slice, corrected.



(e) Sagittal slice, distorted.



(f) Sagittal slice, corrected.

Fig. 6.22: Axial, coronal, and sagittal images from the 4-30-06 study with scanner filtering, before and after distortion correction.

The scanner configuration for the second 4-30-06 study was the same as the first, except for the addition of the scanner filtering:

- TR: 2010 ms
- TE: 2.75 ms
- Flip Angle: 20 ms
- Field of View (FOV): 200 mm^2
- Matrix: 512 $\times$ 512
- Voxel Size: 0.391 $\times$ 0.391 $\times$ 2.000 mm
- 1 Slab, 104 partitions (slices) per orientation
- Sequences: axial, coronal, sagittal
- Large Field of View (FOV) filter used

The difference between the filtered and unfiltered images are quite apparent, refer to Figure 6.21. But notice that the filtered images are still not totally free from distortion. Indeed the images do not exhibit as much distortion as the unfiltered images, so it is clear that the scanner filtering is successful in reducing the amount of distortion visible in the images. However, the telltale characteristics of the gradient nonlinearity distortion are still apparent in the filtered images. There is still room for improvement by the software distortion correction method.

When analyzing the numerical results of the filtered study, shown in Tables 6.36 - 6.38, refer back to the results obtained using the original unfiltered study presented in Tables 6.14 - 6.16.

Midplane	Plane Coefficients
-X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] =$ [**-79.3908**, $-0.0026601, 0.0036608$]
+X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] =$ [**80.110**8, $-0.0026601, 0.0036608$]
-Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] =$ [-0.00077895, **-80.0959**, 0.0030828]
+Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] =$ [-0.00077895, **79.6049**, 0.0030828]
-Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] =$ [$0.00077582, -0.0065026$, **-77.0907**]
+Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] =$ [$0.00077582, -0.0065026$, **81.0227**]

Tab. 6.36: Plane coefficients of ideal planes fitting in 4-30-06 study, with scanner correction.

Comparing the bolded values in the ideal plane results shown in Table 6.36 to those of the unfiltered study shown in Table 6.14 show that the reduction in gradient nonlinearity distortion slightly influenced the midplane and ideal plane coefficients. Although the values are nearly identical, the difference between the values does demonstrate the difference between filtered and unfiltered images. The size of the errors in the plane fitting results in Tables 6.37 and 6.15 are also nearly identical. Both of these observations shows that the calculated theoretical undistorted points for both data sets are also nearly identical.

The real difference between the data sets comes in at the stage where the distortion correction model is calculated. The calculated theoretical undistorted points for both data sets are nearly identical, but the positions of the original acquired data points are vastly different. The data points in the filtered images are much closer to their theoretical undistorted locations than the data points in the unfiltered images. Keeping this basic observation in mind will reveal the reason for the differences between the correction results in the filtered and unfiltered studies. Observing the magnitudes of the standard deviations in Tables 6.16 and 6.38 shows how different factors contribute to the errors produced in each study.

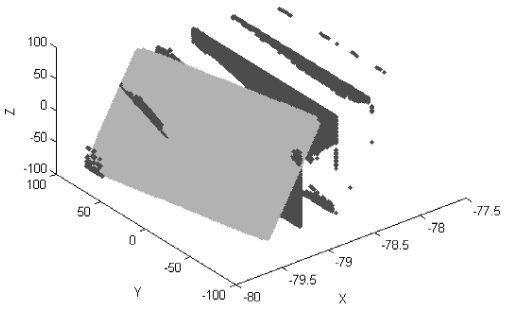
# Data Points Used (X)	62148
# Data Points Used (Y)	61938
# Data Points Used (Z)	61439
Total Residuals (Error) (X) (mm)	24.0145
Total Residuals (Error) (Y) (mm)	29.8867
Total Residuals (Error) (Z) (mm)	26.6469
MSE (Variance) (X) (mm)	0.0092794
MSE (Variance) (Y) (mm)	0.014421
MSE (Variance) (Z) (mm)	0.011557
RMS (Standard Deviation) (X) (mm)	0.09633
RMS (Standard Deviation) (Y) (mm)	0.12009
RMS (Standard Deviation) (Z) (mm)	0.1075

Tab. 6.37: Midplane fitting results from 4-30-06 study, with scanner correction.

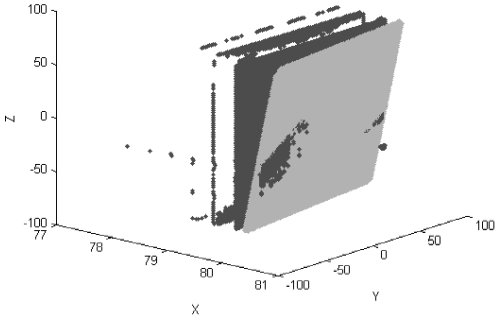
The standard deviation values obtained in the unfiltered study represent errors associated with the magnitude of the distortion and phantom centering. This is indicative of the consistently higher standard deviation values obtained in x in the 11-6-05, 4-30-06, and 6-19-06 studies, despite calibrating the shim coils and centering the phantom well in x. The magnitude of the standard deviations obtained in the filtered study are no longer subjected to the severity of the distortion, since the filtering has greatly reduced the distortion inherent in the images. Now, the standard deviations observed in the filtered study only represent phantom centering. This is very clear: recall that the phantom was displaced approximately 0.4 mm from isocenter in x, essentially centered in y, and displaced approximately 1.9 mm in z. As a result of this centering, the standard deviations observed in x and y are nearly identical, and

are very near ideal, being near $\frac{1}{3}$ of a pixel. This equates to an accuracy of very nearly one pixel, which is the highest accuracy that is theoretically obtainable in this system. On the other hand, the standard deviations observed in z are slightly higher, equating to an accuracy of nearly two pixels. Clearly, phantom centering affects the accuracy of the correction results in filtered studies. Therefore, addressing the issue of phantom centering is imperative to maximizing the accuracy of the distortion correction results.

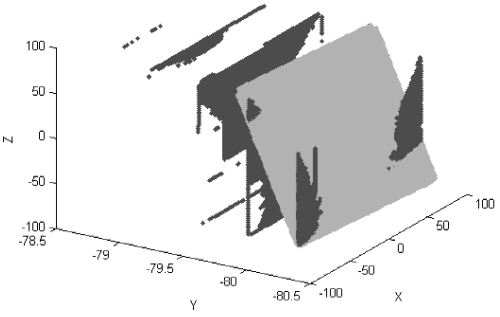
The accuracy of the distortion correction in the filtered study is very good, and surpasses that of the unfiltered study. It is interesting to note that the unfiltered study featured higher standard deviations in x, and the filtered study featured equally higher standard deviations in z. The standard deviations of the remaining two dimensions in both studies were nearly identical in magnitude, with the filtered study featuring slightly smaller values. The correction results in the filtered study further demonstrate the capability of the software to achieve submillimeter accuracy. Analyzing these results clearly show that applying the large FOV filter to all acquired images helps improve the accuracy of the distortion correction, to increase the accuracy in target localization.



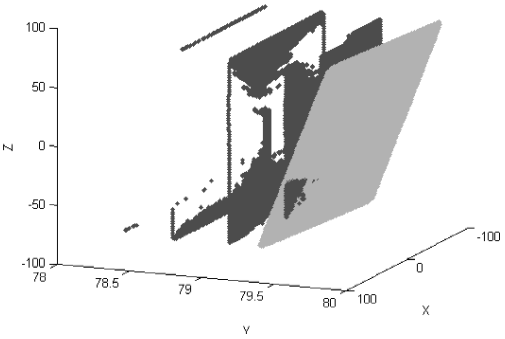
(a) -X face.



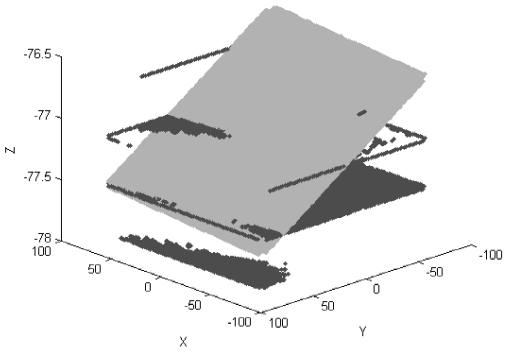
(b) +X face.



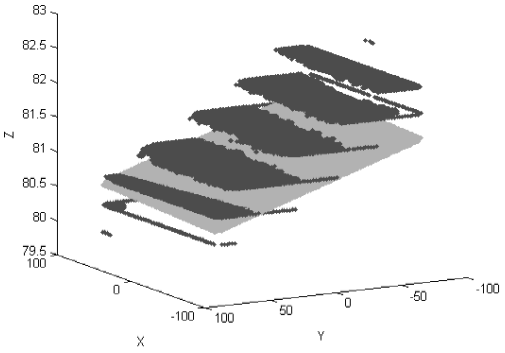
(c) -Y face.



(d) +Y face.



(e) -Z face.

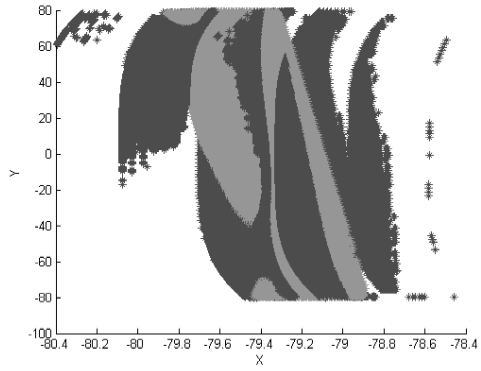


(f) +Z face.

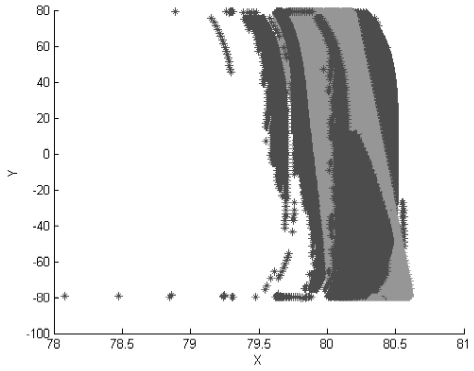
Fig. 6.23: Distorted data points (black) from the filtered 4-30-06 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm.

# Data Points Used (-X)	62233
# Data Points Used (+X)	62172
# Data Points Used (-Y)	62276
# Data Points Used (+Y)	62135
# Data Points Used (-Z)	61670
# Data Points Used (+Z)	61507
Total Residuals (Error) (-X) (mm)	37.5444
Total Residuals (Error) (+X) (mm)	37.7297
Total Residuals (Error) (-Y) (mm)	39.8923
Total Residuals (Error) (+Y) (mm)	37.2096
Total Residuals (Error) (-Z) (mm)	62.6028
Total Residuals (Error) (+Z) (mm)	59.1571
MSE (Variance) (-X) (mm)	0.02265
MSE (Variance) (+X) (mm)	0.022897
MSE (Variance) (-Y) (mm)	0.025554
MSE (Variance) (+Y) (mm)	0.022283
MSE (Variance) (-Z) (mm)	0.06355
MSE (Variance) (+Z) (mm)	0.056897
RMS (Standard Deviation) (-X) (mm)	0.1505
RMS (Standard Deviation) (+X) (mm)	0.15132
RMS (Standard Deviation) (-Y) (mm)	0.15986
RMS (Standard Deviation) (+Y) (mm)	0.14927
RMS (Standard Deviation) (-Z) (mm)	0.25209
RMS (Standard Deviation) (+Z) (mm)	0.23853

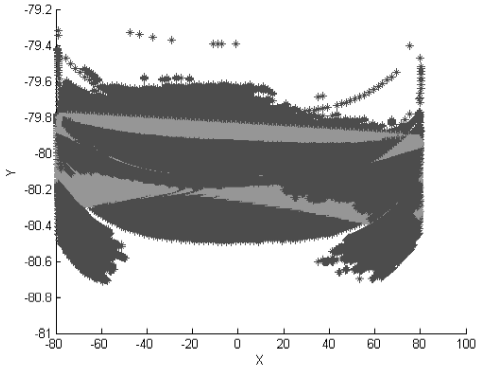
Tab. 6.38: Distortion correction results from 4-30-06 study, with scanner correction.



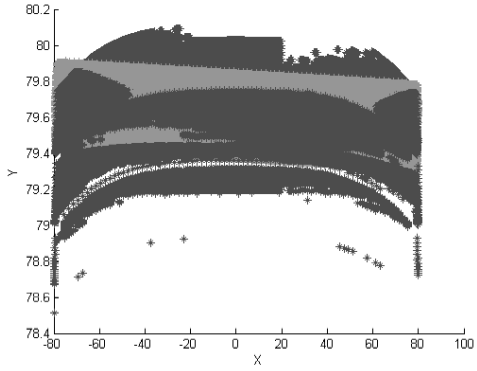
(a) -X face.



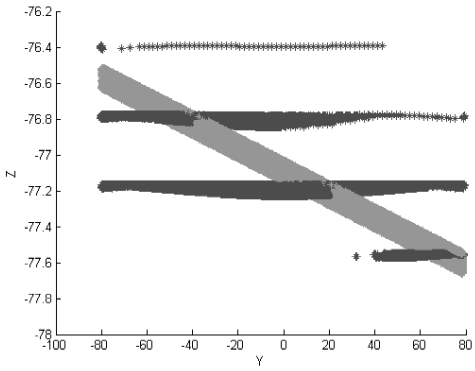
(b) +X face.



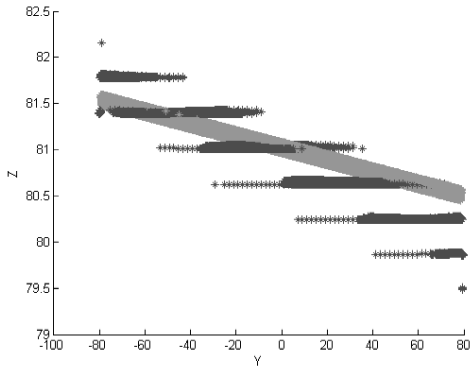
(c) -Y face.



(d) +Y face.



(e) -Z face.



(f) +Z face.

Fig. 6.24: Corrected data points (black) from the filtered 4-30-06 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm.

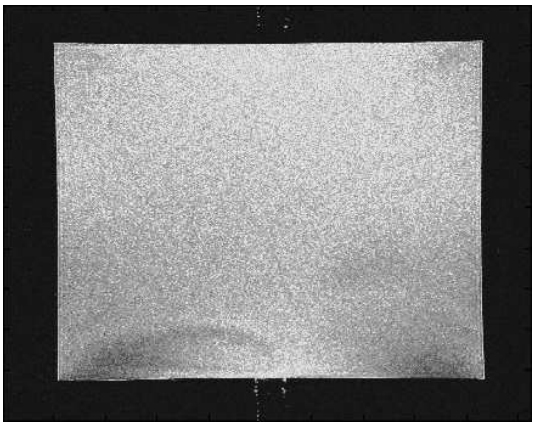
Characteristics of the Data Set

Visualizing the data set acquired from the 4-30-06 study provides great insight into the characteristics of the study. Plotting the distorted and corrected data points against the theoretical undistorted data points for all six phantom faces yields the three-dimensional graphs shown in Figures 6.23 and 6.24. Compare these plots to those acquired from the unfiltered study, shown in Figures 6.7 and 6.8.

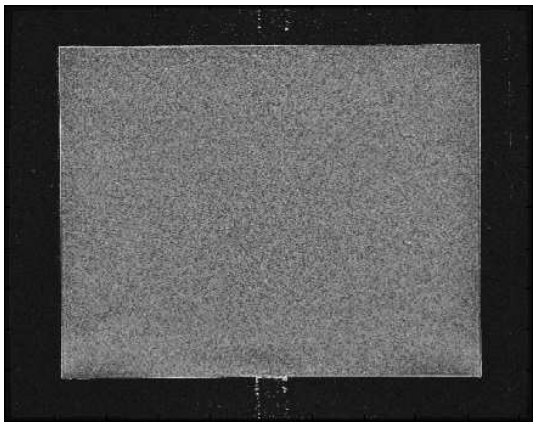
Analyzing the orientation of the distorted data points with the theoretical undistorted data points reveals several interesting properties of the data set. The most apparent difference between the filtered and unfiltered data sets is the shape of the data. Since the gradient nonlinearity distortion is greatly reduced by the filters, the undistorted data does not exhibit the curvature found in the unfiltered data sets. Instead, the data is very flat, and the partitions between the slices is more visible. The data appears to be more stepped, and experiences a smaller spread. Despite the tilt inherent to the theoretical undistorted face, the distorted data set more closely resembles the undistorted phantom.

The locations of the corrected data points also reveal interesting properties of the filtered data set. While the unfiltered data set featured more curvature in the distorted points and less curvature in the corrected points, the opposite is true for the filtered study. Here, the curvature that is characteristic to the gradient field nonlinearities is very apparent in the corrected data points. The spread of the corrected data points is quite small, and the points match closely to their theoretical undistorted locations. The corrected points in the filtered 4-30-06 study match their theoretical undistorted points more closely than the unfiltered 4-30-06 study. Indeed, the positions of the

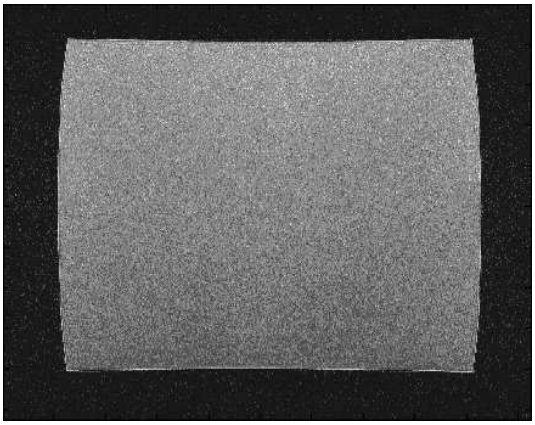
corrected points are not perfect. A handful of data points deviate away slightly from the theoretical locations, as is apparent in some of the faces, but the important characteristic to observe is the close grouping of the majority of the points. Since the differences between the theoretical and actual undistorted points is quite reasonable, the calculated distortion correction model provides a suitable representation of the actual gradient nonlinearity distortion of the MRI scanner, even more so than the unfiltered 4-30-06 study.



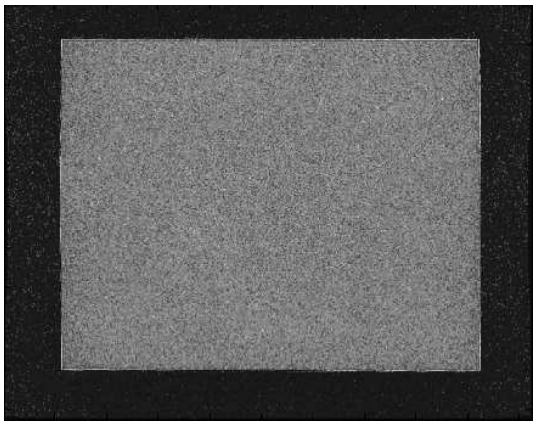
(a) Axial slice, unfiltered.



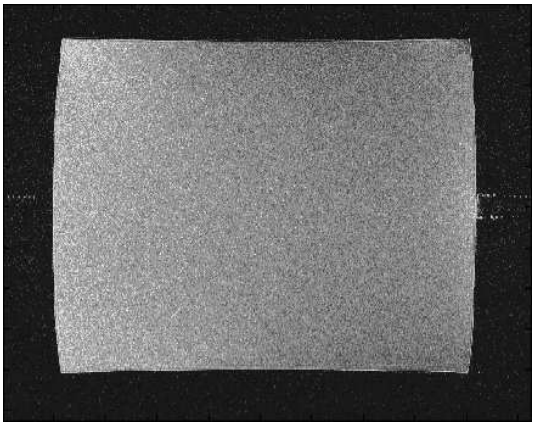
(b) Axial slice, filtered.



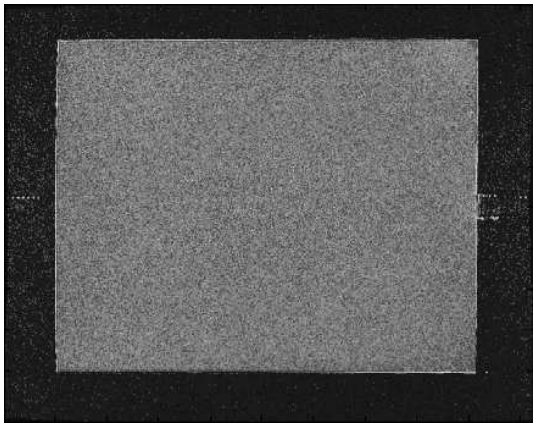
(c) Coronal slice, unfiltered.



(d) Coronal slice, filtered.

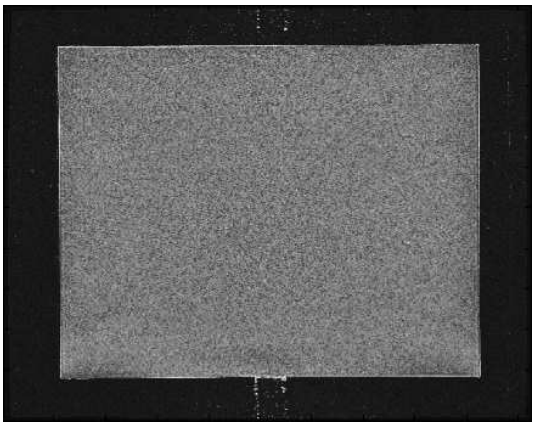


(e) Sagittal slice, unfiltered.

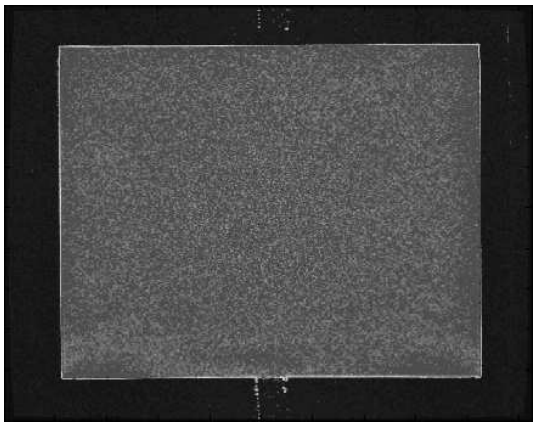


(f) Sagittal slice, filtered.

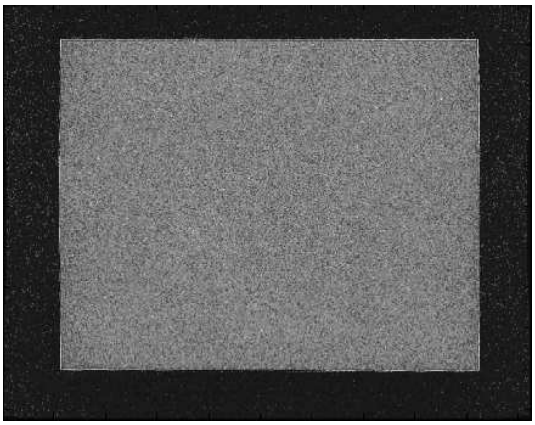
Fig. 6.25: Axial, coronal, and sagittal images from the 6-19-06 study, with and without scanner filtering.



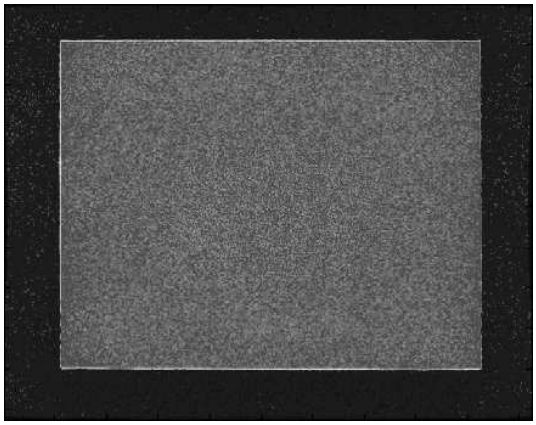
(a) Axial slice, distorted.



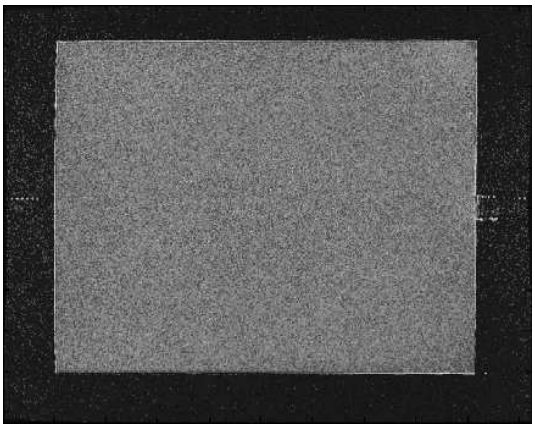
(b) Axial slice, corrected.



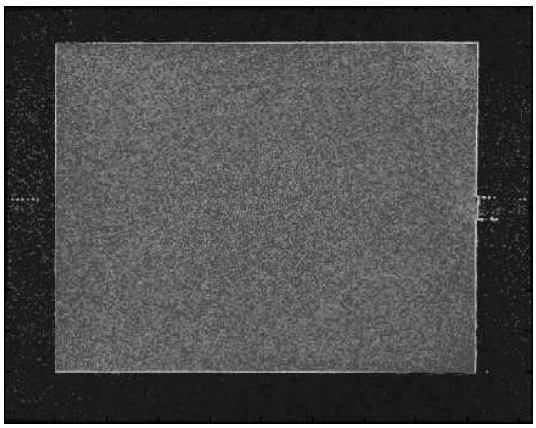
(c) Coronal slice, distorted.



(d) Coronal slice, corrected.



(e) Sagittal slice, distorted.



(f) Sagittal slice, corrected.

Fig. 6.26: Axial, coronal, and sagittal images from the 6-19-06 study with scanner filtering, before and after distortion correction.

The scanner configuration for the second 6-19-06 study was the same as the first, except for the addition of the scanner filtering:

- TR: 2010 ms
- TE: 2.75 ms
- Flip Angle: 20 ms
- Field of View (FOV): 200 mm^2
- Matrix: 512 $\times$ 512
- Voxel Size: 0.391 $\times$ 0.391 $\times$ 2.000 mm
- 1 Slab, 104 partitions (slices) per orientation
- Sequences: axial, coronal, sagittal
- Large Field of View (FOV) filter used

The difference between the filtered and unfiltered images are quite apparent, refer to Figure 6.25. But notice that the filtered images are still not totally free from distortion. Indeed the images do not exhibit as much distortion as the unfiltered images, so it is clear that the scanner filtering is successful in reducing the amount of distortion visible in the images. However, the telltale characteristics of the gradient nonlinearity distortion are still apparent in the filtered images. There is still room for improvement by the software distortion correction method.

When analyzing the numerical results of the filtered study, shown in Tables 6.39 - 6.41, refer back to the results obtained using the original unfiltered study presented in Tables 6.18 - 6.20.

Midplane	Plane Coefficients
-X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{-78.7486}, -0.0060346, 0.0013587]$
+X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [\mathbf{80.7544}, -0.0060346, 0.0013587]$
-Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [0.0071554, \mathbf{-79.2247}, 0.0032633]$
+Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [0.0071554, \mathbf{80.4802}, 0.0032633]$
-Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [0.0023423, 0.00021875, \mathbf{-82.3999}]$
+Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [0.0023423, 0.00021875, \mathbf{75.7105}]$

Tab. 6.39: Plane coefficients of ideal planes fitting in 6-19-06 study, with scanner correction.

Numerical Results

Comparing the bolded values in the ideal plane results shown in Table 6.39 to those of the unfiltered study shown in Table 6.18 show that the reduction in gradient nonlinearity distortion slightly influenced the midplane and ideal plane coefficients. Although the values are nearly identical, the difference between the values does demonstrate the difference between filtered and unfiltered images. The size of the errors in the plane fitting results in Tables 6.40 and 6.19 are also nearly identical. Both of these observations shows that the calculated theoretical undistorted points for both data sets are also nearly identical.

The real difference between the data sets comes in at the stage where the distortion correction model is calculated. The calculated theoretical undistorted points for both data sets are nearly identical, but the positions of the original acquired data points are vastly different. The data points in the filtered images are much closer to their theoretical undistorted locations than the data points in the unfiltered images. Keeping this basic observation in mind will reveal the reason for the differences between the correction results in the filtered and unfiltered studies. Observing the

# Data Points Used (X)	62771
# Data Points Used (Y)	62675
# Data Points Used (Z)	62235
Total Residuals (Error) (X) (mm)	26.1821
Total Residuals (Error) (Y) (mm)	29.7532
Total Residuals (Error) (Z) (mm)	24.5828
MSE (Variance) (X) (mm)	0.010921
MSE (Variance) (Y) (mm)	0.014124
MSE (Variance) (Z) (mm)	0.0097102
RMS (Standard Deviation) (X) (mm)	0.1045
RMS (Standard Deviation) (Y) (mm)	0.11885
RMS (Standard Deviation) (Z) (mm)	0.09854

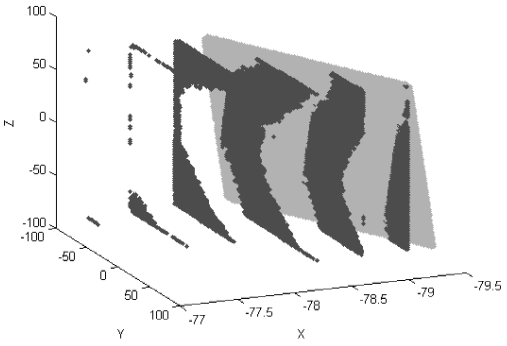
Tab. 6.40: Plane fitting results from 6-19-06 study, with scanner correction.

magnitudes of the standard deviations in Tables 6.20 and 6.41 shows how different factors contribute to the errors produced in each study.

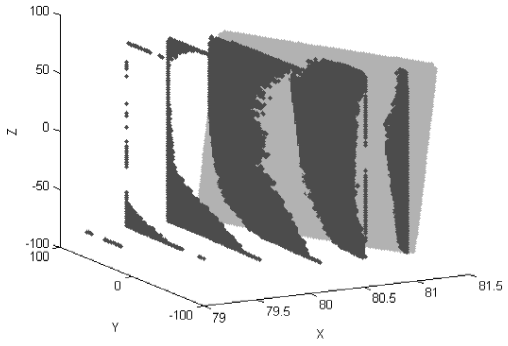
The standard deviation values obtained in the unfiltered 6-19-06 study represent errors associated with the magnitude of the distortion and phantom centering. This is indicative of the consistently higher standard deviation values obtained in x in the unfiltered 11-6-05, 6-19-06, and 6-19-06 studies, despite calibrating the shim coils and centering the phantom well in x. The magnitude of the standard deviations obtained in the filtered study are no longer subjected to the severity of the distortion, since the filtering has greatly reduced the distortion inherent in the images. Now, the standard deviations observed in the filtered study only represent phantom centering. This is very clear: recall that the phantom was displaced approximately 1.0 mm from

isocenter in x, 0.9 mm from isocenter in y, and displaced approximately 3.0 mm in z. As a result of this centering, the standard deviations observed in x and y are nearly identical, and are very near ideal, being near $\frac{1}{3}$ of a pixel. This equates to an accuracy of very nearly one pixel, which is the highest accuracy that is theoretically obtainable in this system. On the other hand, the standard deviations observed in z are slightly higher, equating to an accuracy of nearly two pixels. As was the case in the filtered 4-30-06 study, phantom centering clearly affects the accuracy of the correction results in filtered studies. Therefore, addressing the issue of phantom centering is imperative to maximizing the accuracy of the distortion correction.

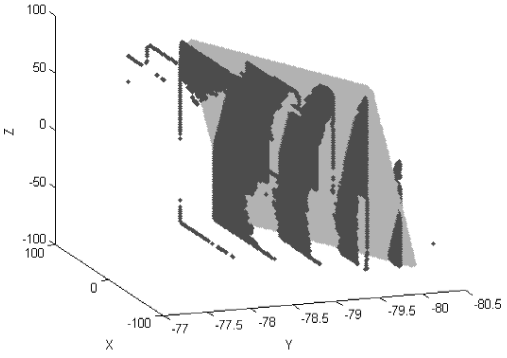
The accuracy of the distortion correction in the filtered study is very good, and surpasses that of the unfiltered study. It is interesting to note that the unfiltered study featured higher standard deviations in x, and the filtered study featured equally higher standard deviations in z. The standard deviations of the remaining two dimensions in both studies were nearly identical in magnitude, with the filtered study featuring slightly smaller values. The correction results in the filtered study further demonstrate the capability of the software to achieve submillimeter accuracy. Analyzing these results clearly show that applying the large FOV filter to all acquired images helps improve the accuracy of the distortion correction, to increase the accuracy in target localization.



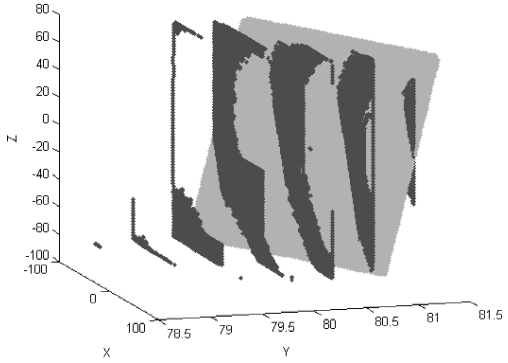
(a) -X face.



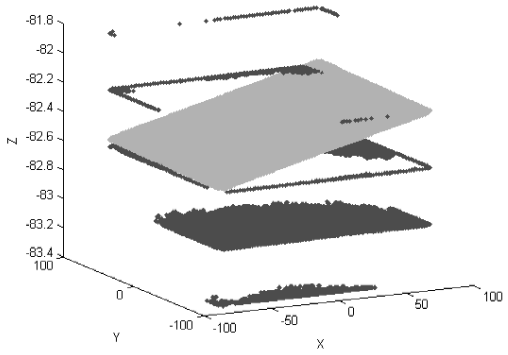
(b) +X face.



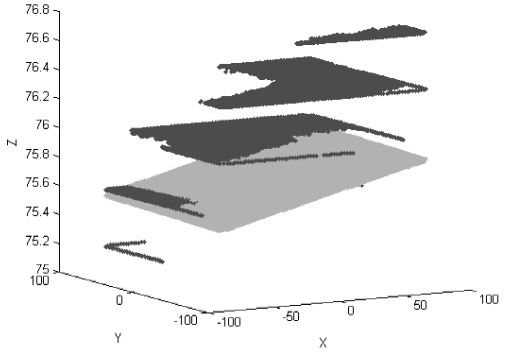
(c) -Y face.



(d) +Y face.



(e) -Z face.

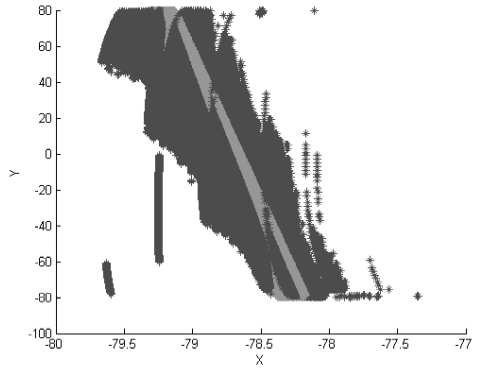


(f) +Z face.

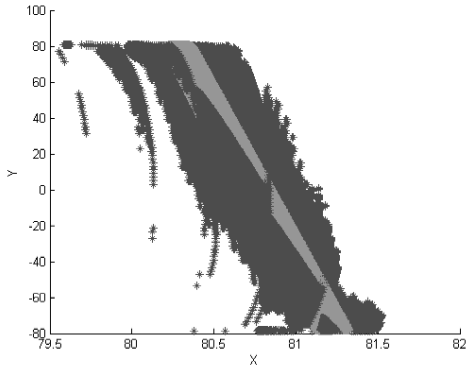
Fig. 6.27: Distorted data points (black) from the filtered 6-19-06 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm.

# Data Points Used (-X)	62970
# Data Points Used (+X)	63091
# Data Points Used (-Y)	63100
# Data Points Used (+Y)	62908
# Data Points Used (-Z)	62395
# Data Points Used (+Z)	62392
Total Residuals (Error) (-X) (mm)	39.0892
Total Residuals (Error) (+X) (mm)	36.5942
Total Residuals (Error) (-Y) (mm)	39.7922
Total Residuals (Error) (+Y) (mm)	38.8985
Total Residuals (Error) (-Z) (mm)	56.3031
Total Residuals (Error) (+Z) (mm)	60.5979
MSE (Variance) (-X) (mm)	0.024265
MSE (Variance) (+X) (mm)	0.021225
MSE (Variance) (-Y) (mm)	0.025094
MSE (Variance) (+Y) (mm)	0.024052
MSE (Variance) (-Z) (mm)	0.050806
MSE (Variance) (+Z) (mm)	0.058855
RMS (Standard Deviation) (-X) (mm)	0.15577
RMS (Standard Deviation) (+X) (mm)	0.14569
RMS (Standard Deviation) (-Y) (mm)	0.15841
RMS (Standard Deviation) (+Y) (mm)	0.15509
RMS (Standard Deviation) (-Z) (mm)	0.2254
RMS (Standard Deviation) (+Z) (mm)	0.2426

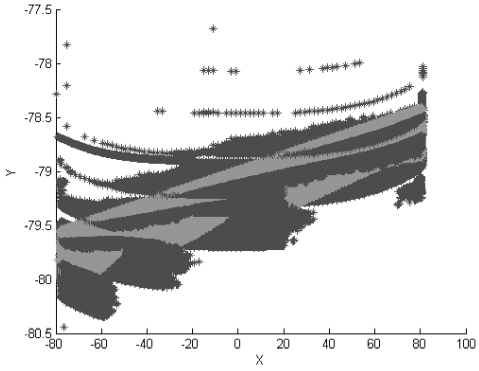
Tab. 6.41: Distortion correction results from 6-19-06 study, with scanner correction.



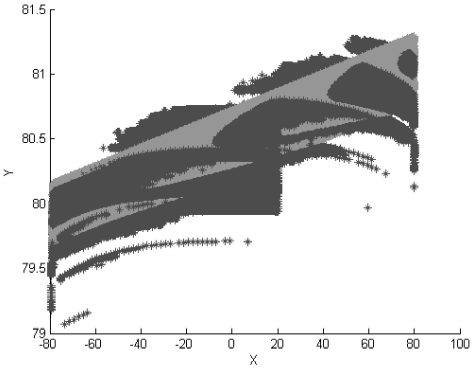
(a) -X face.



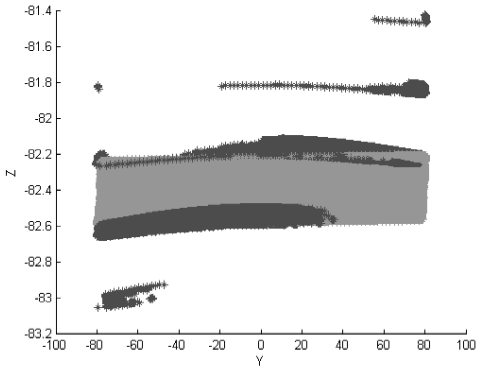
(b) +X face.



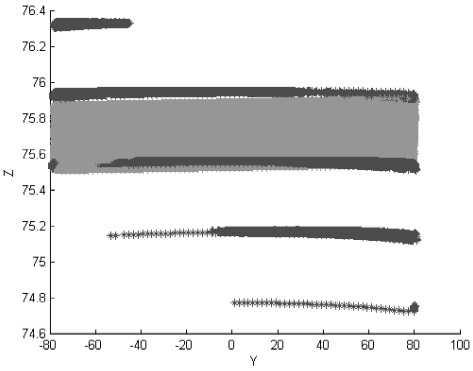
(c) -Y face.



(d) +Y face.



(e) -Z face.



(f) +Z face.

Fig. 6.28: Corrected data points (black) from the filtered 6-19-06 study versus theoretical undistorted data points (grey) for the six faces of the phantom. Axes are graduated in mm.

Characteristics of the Data Set

Visualizing the data set acquired from the 6-19-06 study provides great insight into the characteristics of the study. Plotting the distorted and corrected data points against the theoretical undistorted data points for all six phantom faces yields the three-dimensional graphs shown in Figures 6.27 and 6.28. Compare these plots to those acquired from the unfiltered study, shown in Figures 6.9 and 6.10.

Analyzing the orientation of the distorted data points with the theoretical undistorted data points reveals several interesting properties of the data set. The most apparent difference between the filtered and unfiltered data sets is the shape of the data. Since the gradient nonlinearity distortion is greatly reduced by the filters, the undistorted data does not exhibit the curvature found in the unfiltered data sets. Instead, the data is very flat, and the partitions between the slices is more visible. The data appears to be more stepped, and experiences a smaller spread. Despite the tilt inherent to the theoretical undistorted face, the distorted data set more closely resembles the undistorted phantom.

The locations of the corrected data points also reveal interesting properties of the filtered data set. While the unfiltered data set featured more curvature in the distorted points and less curvature in the corrected points, the opposite is true for the filtered study. Here, the curvature that is characteristic to the gradient field nonlinearities is very apparent in the corrected data points. The spread of the corrected data points is quite small, and the points match closely to their theoretical undistorted locations. The corrected points in the filtered 6-19-06 study match their theoretical undistorted points more closely than the unfiltered 6-19-06 study. Indeed, the positions of the

corrected points are not perfect. A handful of data points deviate away slightly from the theoretical locations, as is apparent in some of the faces, but the important characteristic to observe is the close grouping of the majority of the points. Since the differences between the theoretical and actual undistorted points is quite reasonable, the calculated distortion correction model provides a suitable representation of the actual gradient nonlinearity distortion of the MRI scanner, even more so than the unfiltered 6-19-06 study.

Distortion Correction Results

Observing the size of the errors inherent to the distortion correction results in Table 6.42 provides insight to the factors contributing to the accuracy of the distortion correction results. It becomes apparent that different factors contribute to the errors in each study. The unfiltered studies both experience larger errors in the x faces, while the filtered studies experience larger errors in the z faces. Recall that both of these studies featured the oil phantom that was positioned greater than one millimeter from isocenter in z. The results observed for the filtered studies seem to reflect this aspect of the experimental setup, being that the standard deviations in z are roughly 50% greater than those observed in x and y.

The larger observed standard deviations observed in the x faces of the unfiltered studies can be a result of several factors. The increased centering in z will definitely contribute to the size of the errors obtained in all three dimensions. Observing the size of the standard deviations from the 11-6-05 study also yields the same pattern: the x faces feature greater standard deviations than the y and z faces. The seemingly

logical contributing factor for this observation is simply the characteristics of the inherent gradient field nonlinearities of the MRI scanner. It is very possible that the distortion in x is greater than that of y and z, therefore resulting in correction results for x that are not as tight as those obtained in y and z. The greater the inherent distortion in the data set, the lower quality of fit that can be achieved, since the results produced by a least squares fit represents the best reconciliation of the data points to the distortion correction model. This idea holds consistent as well with the filtered studies, since this greater distortion in x is greatly reduced by the filters. The standard deviations in x for the filtered studies are much closer to the expected values of $\frac{1}{3}$ of a pixel.

Face	**4-30-06**	**4-30-06 Filtered**	**6-19-06**	**6-19-06 Filtered**
-X	0.2325	0.1505	0.22204	0.15577
+X	0.24451	0.15132	0.22131	0.14569
-Y	0.16975	0.15986	0.19426	0.15841
+Y	0.16937	0.14927	0.18428	0.15509
-Z	0.1514	0.25209	0.13928	0.2254
+Z	0.14539	0.23853	0.13928	0.2426

Tab. 6.42: RMS (standard deviations) (in mm) of distortion correction results for 4-30-06 and 6-19-06 studies, filtered and unfiltered.

Calculated Distortion Correction Parameters

Analyzing the distortion correction parameters will provide insight into the characteristics of the gradient nonlinearity distortion inherent in each study. Recall that the size of each distortion correction parameter correlates to the distance each distorted data point needs to move in order to place it in its ideal undistorted location. When

	4-30-06 Unfiltered	**4-30-06 Filtered**	**6-19-06 Unfiltered**	**6-19-06 Filtered**
K_{x0}	-3.3×10^{-7}	9.3676×10^{-7}	-8.8382×10^{-7}	4.6067×10^{-7}
K_{x1}	6.5079×10^{-6}	5.7956×10^{-7}	7.4406×10^{-6}	1.2131×10^{-6}
K_{x2}	-2.6081×10^{-10}	-2.5483×10^{-11}	-2.9227×10^{-10}	-4.7254×10^{-11}
K_{x3}	-6.9144×10^{-11}	-6.395×10^{-11}	-2.6661×10^{-11}	-2.8949×10^{-11}
K_{x4}	2.4875×10^{-10}	5.1175×10^{-11}	1.3204×10^{-10}	2.6523×10^{-12}
K_{y0}	8.4594×10^{-8}	1.1517×10^{-6}	-2.8245×10^{-7}	8.0342×10^{-7}
K_{y1}	5.1732×10^{-6}	3.9226×10^{-7}	5.878×10^{-6}	8.6938×10^{-7}
K_{y2}	-2.3672×10^{-10}	-6.3302×10^{-12}	-2.5701×10^{-10}	-2.9425×10^{-11}
K_{y3}	-1.0569×10^{-10}	-9.9173×10^{-11}	-7.6522×10^{-11}	-6.7613×10^{-11}
K_{y4}	3.2755×10^{-10}	6.4052×10^{-11}	2.1287×10^{-10}	1.4821×10^{-11}
K_{z0}	-2.5637×10^{-6}	-6.0376×10^{-7}	-5.2356×10^{-6}	-1.1055×10^{-6}
K_{z1}	1.7746×10^{-6}	-1.2778×10^{-7}	2.8076×10^{-6}	-1.1141×10^{-6}
K_{z2}	-1.9439×10^{-10}	6.41×10^{-11}	2.8611×10^{-10}	1.9345×10^{-10}
K_{z3}	8.2028×10^{-11}	1.444×10^{-11}	5.102×10^{-11}	-5.307×10^{-12}
K_{z4}	-8.3005×10^{-11}	-8.964×10^{-11}	-2.8509×10^{-10}	3.0989×10^{-11}

Tab. 6.43: Distortion correction parameters for 4-30-06 and 6-19-06 studies, filtered and unfiltered.

the distortion is greater in the study, the quality of the least squares fit for the distortion correction model degrades. This is because the spread of the data points is greater, and the trend that the data points follow deviates further from the spherical harmonics expansion. The least squares fit represents the best reconciliation of the spherical harmonics expansion to the data sets. Computing the distortion correction parameters on a data set with a smaller degree of distortion will result in a tighter least squares fit, yielding smaller errors and higher accuracy.

Consider the effects of the center offsets and differences caused by calibrating the shim coils in each data set. Observing the magnitude of each distortion correction parameter in each study will identify the trends and characteristics inherent in each

data set. The magnitude of each parameter is an indication of the severity of the distortion observed in each study. The larger the K value, the greater the distortion observed. This applies as well to other factors, such as phantom centering and calibration of the shim coils.

Conclusion

Since the filtered and unfiltered studies from 4-30-06 and 6-19-06 were performed without having to reconfigure the scanner or the phantom, more direct comparisons can be made in each case. Observing the filtered and unfiltered results in Tables 6.42 and 6.43 demonstrates the extent to which the scanner filters reduce the amount of inherent distortion in the acquired images. Some of the coefficients calculated in the filtered studies are one to two magnitudes of ten smaller than the corresponding coefficient calculated in the unfiltered study. In terms of the size of errors observed, the correction results from the filtered studies are much closer to pixel accuracy, which represents the highest theoretical accuracy of this system (being that the pixel is the smallest measurable unit in each image, and therefore in the data sets).

The results obtained here demonstrate the software's capability and flexibility in operation. Not only is the software distortion correction method able to correct filtered studies in the same manner as unfiltered studies, the correction results are just as valid with filtered studies, while featuring higher accuracy. This is a very useful observation, especially when considering the requirements set forth by LLUMC, where the utmost accuracy is top priority. Being that physicians and oncologists tend to prefer applying scanner filtering to the images acquired in a MRI scan, they all can

rest assured that doing so will not interfere with the proper operation of this distortion correction method.

The feasibility of a software-based MRI gradient nonlinearity distortion correction is determined by two primary factors: accuracy and speed. It is vital that the software strikes a nice balance between these two properties. On the one hand, the software needs to be accurate, such that the results it produces is usable in various medical applications. If, however, the software is so slow and inefficient that calculating the correction model requires a ridiculous amount of time, then the software is of little use. Examining the other side of the spectrum reveals a similar problem: the software may be able to quickly calculate a result, but the result is so invalid that accurate target localization is impossible. In both cases, the software would be a complete waste of time and hard drive space.

One of the goals of this project is to produce a software package that *feasibly* implements gradient nonlinearity distortion correction. The feasibility of the software is defined by the criteria mentioned above: accuracy and speed. The accuracy of the software package has already been identified and verified: the correction method is indeed capable of providing submillimeter accuracy for target localization in functional proton radiosurgery. The issue that remains is whether or not the software can calculate that answer in a timely fashion. The software package can potentially mimic the first scenario mentioned above, where the package is accurate but requires an exorbitant amount of time to compute a result. Will this software package fall into this category? Or will it strike a balance between accuracy and speed? The only way to answer these questions is to perform *performance benchmarks.*

6.6.1 Impetus

The speed and accuracy of any piece of software is always under scrutiny. It is impossible to have perfect amounts of both: the two attributes are typically inversely proportional values. Increasing accuracy usually compromises speed, and vice versa. The exact amounts of each are determined solely on the application. If the software in question is to be used in an environment where the highest accuracy achievable is of the utmost importance, then it will make little difference if the software requires insane amounts of time to compute the result. The ongoing race to compute the most digits of pi (π) most definitely requires the utmost accuracy, and calculation time is not important. Therefore, no researcher is concerned with striking a balance between speed and accuracy. The only concern in terms of speed is that the application completes its task in a finite amount of time. On the other hand, the software in question may be used in an extremely fast-paced environment, where results are needed immediately. It can go without saying that today's missile defense system would be entirely useless if the targeting computer required several hours (or, even, several seconds) to compute the trajectory of–and successfully intercept–an incoming enemy missile. Here, speed takes a much higher precedence over accuracy. Accuracy is indeed important as well, but not to the same degree as the calculation of pi.

These examples represent the two extremes in software development. The majority of applications developed will not have the speed-accuracy balance as skewed as these. The software-based MRI gradient nonlinearity distortion correction developed here is no different. Recall back to the design of the software presented in Chapter 2: this software was designed to be a feasible solution to gradient nonlinearity distortion,

meaning that the software will deliver accurate results in a timely fashion. In order to accomplish this goal, the software design considered many prior works, in order to improve upon the shortcomings of other methods. This was done to avoid the problems encountered in other solutions. The lessons learned from other works allows this software package to accomplish this goal.

The primary focus in this software is on the numerical calculations themselves. By concentrating on how the software performs complex calculations, it is possible to reasonably preserve both speed and accuracy. Most specifically, the method in which the gradient nonlinearity distortion correction is calculated determines the accuracy of the method as well. Performing a direct calculation and correction of the gradient nonlinearity distortion increases both speed and accuracy. This may sound slightly contradictory, but is in fact true. Calculating the inverse of a complex polynomial expansion is, to say the least, complex. Due to the finite precision of modern computer systems, inverting any complex mathematical model introduces error due to the rounding and truncation necessary to store the answers in memory. Eliminating the need to perform the inversions by calculating the correction model directly greatly simplifies the software, thereby increasing speed while preserving accuracy.

6.6.2 Test Setup

There exist nearly an infinite number of computer system configurations. Even though it is possible to run this software on essentially any modern computer system, identifying the ideal computer system configuration is beneficial. The benchmarks performed

here have the following goals:

1. To analyze the execution time of the software.

2. To demonstrate the performance of the software across various platforms.

3. To identify which system components are most beneficial in reducing execution time.

Designing the tests to take these issues into consideration will provide a conclusive analysis of the software's performance. It is impossible to predict or forsee the exact configurations of the computer systems that will run this software. Therefore, it is important that the software executes smoothly across various platforms. To accomplish these tasks, several machines were utilized for this test. The configurations of each of these machines is shown in Table 6.44.

CPU	**Clock Speed**	**FSB**	**RAM**	**Hard Drive**
AMD Athlon64 3800+	2.4 GHZ	800 mHz	1 GB DDR 400 [2]	74 GB 10,000 RPM
Intel Pentium M 730	1.6 GHZ	533 mHz	2 GB DDR2 533	80 GB 5400 RPM
Intel Pentium4 651	3.4 GHZ	533 mHz	1 GB DDR2 533	250 GB 7200 RPM
Intel Pentium D 820	2.8 GHZ	533 mHz	512 MB DDR2533	120 GB 7200 RPM
AMD Athlon64 3000+	2.0 GHZ	800 mHz	1 GB DDR 400	80 GB 7200 RPM

Tab. 6.44: Configurations of test machines.

Including such a vast array of machines in these benchmarks will help identify which hardware components are most beneficial to the distortion correction software's performance. These machines represent relatively modern machines (2004 and

[2] The Athlon64 3800+ machine uses very high performance memory, with CAS latencies of 2-2-2-5. These values represent the fastest (stable) timings available for DDR 400 memory. All other computers use generic memory with more relaxed timings.

newer). The first three machines are configured for video gaming, which generally represent some of the most computationally-intensive consumer applications available. Therefore, these systems feature fast processors and no less than one gigabyte of RAM. The Athlon64 3800+ machine is a dedicated gaming machine built in late 2004, and features the fastest components available on the market at build time. The Pentium M machine is the gaming laptop used to develop, test, and verify the software. The Pentium4 machine represents a typical higher-end desktop system available as of early 2006. The final two machines represent basic entry-level desktop machines of 2006 used primarily for common tasks, such as internet browsing, word processing, and some light video gaming.

The operating system of choice was Microsoft Windows XP Professional Service Pack 2. The Matlab code was compiled into a self-contained standalone Windows application using the Matlab compiler. This configuration allows the software package to be executed on any Windows system without the Matlab environment, and promotes consistency across various platforms.

Two benchmarks were performed with the distortion correction software. The first benchmark tests the performance of the software using two studies, provided by the unfiltered 6-19-05 study's axial and coronal sequences. The second benchmark measures the performance of the software using three studies, provided by the unfiltered 6-19-06.

To provide a baseline for comparisons between machines, a program to calculate the value of pi (π) was used. The program, called *Super PI*, calculates the value of π to a specified number of significant digits between 16,000 and 32,000,000. This appli-

cation is a Windows ported version of the algorithm that was used at the University of Tokyo in 1995 to calculate π on a supercomputer up to 4,294,960,000 decimal digits. The Super PI program is often used by computer enthusiasts as a performance benchmark to compare computer performance before and after CPU overclocking procedures. Although these tests do not involve CPU overclocking or other performance enhancements, the test will still be useful in observing the performance of each system in an independent software environment. According to the documentation included with Super PI, a computer equipped with an Intel Pentium 90 mHz processor, 40 MB RAM, and 340 MB hard drive space will require approximately 3 days to calculate π to 32 million digits. The computers featured in this test should have no problem calculating π to 32 million digits in much less time.

6.6.3 Benchmark Results

Super PI to 32 Million Digits

The intensity, complexity, and sheer number of calculations required by Super PI to calculate π to 32 million digits tests the capability of each test system. As mentioned previously, Super PI is used to measure CPU performance. Higher CPU performance generally correlates to lower execution times. The Super PI program does require a fairly large memory cache to operate, about 256 MB of RAM. However, this is roughly 1/7 of the amount of memory that is used during execution of the distortion correction software, primarily in the data quality assurance stage. Five test runs were executed, and the average of the execution times achieved by each test machine is shown in Figure 6.29.

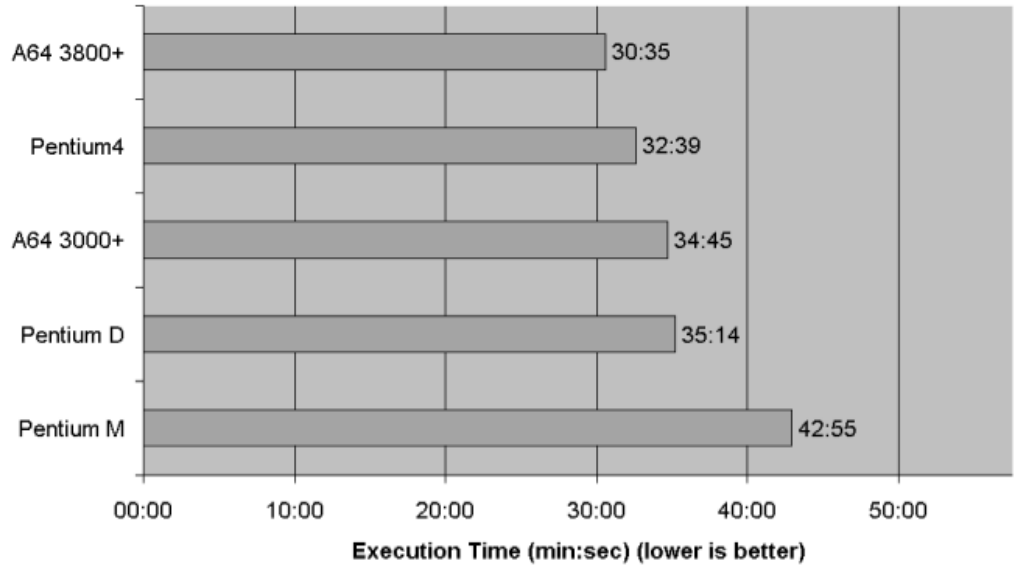


Fig. 6.29: Execution times of Super PI on the various test machines.

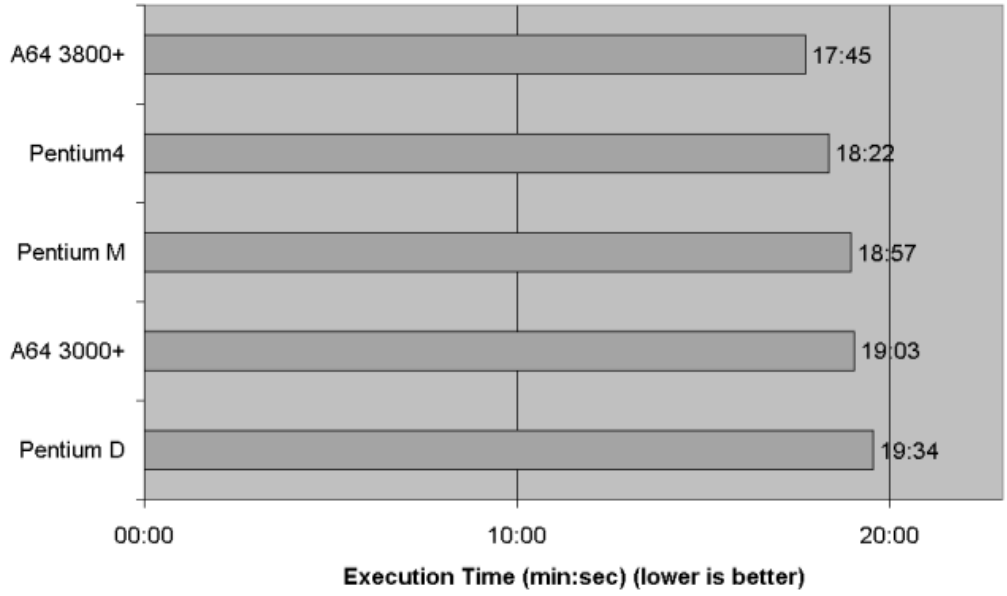
The results from this benchmark demonstrate the wide range of CPU's utilized in this test. The Athlon64 3800+ machine leads the pack by nearly 3 minutes over the next competitor. What may seem surprising is the fact that the Pentium4 machine, which boasts the highest clock speed of all the machines, cannot top the Athlon64 3800+. The reasons behind this observation is in the difference in CPU architectures utilized in the Pentium4 and Athlon64 CPUs. The same reason holds as well for the Pentium M laptop, which has the lowest clock speed of the modern test machines. A discussion of this topic will not be presented, as this subject is beyond the scope of this project. However, it is beneficial to note that there does exist architectural differences between the AMD and Intel machines, and will provide more insight when

analyzing the benchmarks for the distortion correction software.

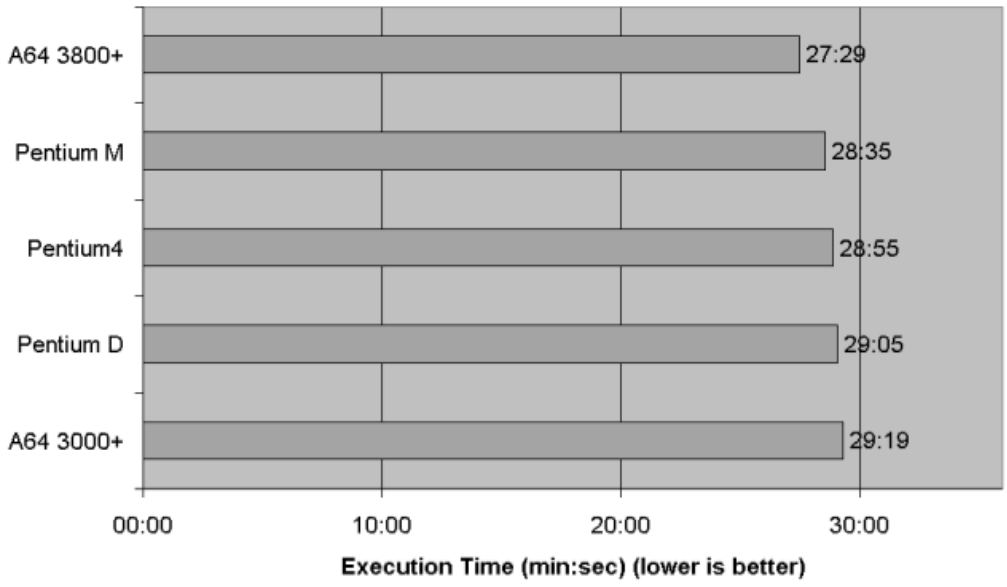
Distortion Correction Software

While the Super PI program is used to primarily quantify CPU performance, the distortion correction software benchmark will indicate the capability of the entire computer system. The speed of the CPU will determine how quickly calculations can be made. The amount of RAM available to the computer will decide how much data can be stored at one time. The speed of the hard drive will determine how quickly paging operations can be performed, in the event that all the data cannot be stored within RAM during execution. For all the test systems, save the Pentium M, the 1.5 GB of data that is manipulated during execution will not fit into RAM, and therefore the operating system will need to perform several paging operations in order to complete the process. Memory pages will be swapped out of RAM onto the hard drive. The relatively slow access and write time across the IDE or SATA bus, in addition to the latencies of the system memory, will greatly increase the execution time.

Consider these factors when analyzing the benchmark results shown in Figures 6.30(a) and 6.30(b). These results reflect the average execution time achieved after five test runs.



(a) Two studies.



(b) Three studies.

Fig. 6.30: Execution times of distortion correction software with two (a) and three (b) studies on the various test machines.

The distortion correction benchmark results reveal a particularly interesting characteristic: the software performs *very* consistently across all the modern systems. The difference between the fastest and slowest observed times in both tests is about two minutes, correlating to about 7 - 10% of the total execution time. This represents a stark contrast when analyzing the benchmark results of the Super PI program. The results shown in Figure 6.29 demonstrate a wide performance range between each of the test systems, indicating the differences in computational power in each system. The CPUs utilized in each system are by no means equal, and neither are the remaining system components, yet the software executes in nearly the same amount of time on each test machine.

6.6.4 Conclusions

Observing these benchmark results provides two primary conclusions. On the one hand, it becomes clear that the software is capable of performing very consistently on the average desktop computer system available in 2006. This is a very interesting result, especially when considering the differences in price for each test system. The prices of each test system range from $400 for the Athlon64 3000+ machine (in 2006), to over $2500 for the Athlon64 3800+ machine (in 2004).

On the other hand, the benchmark results show that there is one consistent factor inherent in each machine that contributes to the fairly consistent execution time on each machine. Two of the test machines stand out from the rest: the Pentium M and Athlon64 3800+ machines. The Pentium M machine boasts the highest available RAM, at least twice as much as the other machines. This machine has the capability

to store in RAM all the calculation data used during program execution. All the other test systems must perform paging, swapping out data values between memory and the hard drive. The CPU is quite capable, despite having the second slowest clock speed of the test group. The hard drive, however, is fairly slow, but the large amount of available system memory greatly minimizes the use of the hard drive during execution.

On the other hand, the Athlon64 3800+ machine features the highest performance components of all the test machines: the fastest CPU, the lowest latency memory, and the fastest hard drive. Therefore it should not be surprising that this machine performs at the top of all the benchmarks. Taking paging into consideration demonstrates the capability of the Athlon64 3800+ machine. Even though all the program data cannot be held in main memory at one time, the high performance system components greatly reduces the penalties incurred as a result of paging. All of these components working together provide a suitable system capable of handling the large calculations required of the distortion correction method.

The configuration of the Pentium M and Athlon64 3800+ machines should correlate to noticeably higher performance in the distortion correction. However, this is not demonstrated in the results. Indeed, these machines perform at the top of the group, but the performance gap is not large. All of the machines complete the distortion correction in very similar time spans. The execution time required to complete the calculations may seem somewhat lengthy (nearly 20 minutes for two studies, 30 minutes for three), but keep in mind the scale of the method. Recall back to Chapters 3 - 5 the sheer amount of data that is processed at each stage of the distortion correc-

tion process. The average number of three-dimensional data points processed during each execution of the software is roughly 384,000. And the calculations performed on these data values are by no means simple or small.

The question that now arises is: what contributes to the unusually consistent execution times across the various test platforms? There exists only one reasonable explanation: this peculiarity is a simple inherent trait of the distortion correction software itself. Not necessarily in terms of algorithm design, but the programming environment used to create the algorithms. The performance of the distortion correction software is a direct result of the nature of the software implementation. Matlab is a *scripting language*, rather than a traditional compiled language similar to C or C++. When programs created using scripting languages are executed, each command is loaded from memory, read as a string, interpreted, and then the appropriate assembly language call is made. This process is *extremely* slow in comparison to compiled programs, in which the high level code is translated directly into bytecode, then executed.

The primary differences between scripting and compiled languages are (1) the complexity of software development and (2) the performance of the software. Scripting languages greatly enhance the speed of software development simply because a compiler is not required to execute the program, therefore software development does not involve creating a compiler. Matlab code is never compiled before execution. The Matlab "Compiler" is in fact a misnomer, since this "compiler" is actually an interface used to execute the Matlab code outside of the Matlab environment. Unfortunately, the nature of Matlab severely limits computational performance, as demonstrated

in these benchmarks. On the other hand, compiled languages require more time to develop, since a capable compiler must be developed alongside the programs. The extra time necessary for software development definitely pays off in the form of greatly increased computational performance. The difference in execution time is commonly several factors of ten.

To verify this claim, the following algorithm was programmed in Matlab and C++, and then executed:

```
a = 100;
b = 200;
for i = 1:(512*512*104)
    result = i * (a * b);
end;
```

The C++ implementation of the program completed execution in 8 seconds. The Matlab implementation, on the other hand, required *20 minutes 21 seconds!* This is a *substantial* difference in performance, approximately 150 times! So indeed programs written in scripting languages perform *much* slower than programs written in compiled languages.

Even though the Matlab environment is extremely capable of handling large calculations, such as those required by this distortion correction method, the environment is clearly not the fastest or most computationally efficient solution out there. The fact tat Matlab is a scripting language creates a certain amount of computational overhead that is consistent across different systems, even being somewhat independent of the performance of each system. Indeed, the capability of the system does contribute to

the overall performance of the software, but the differences in performance between systems is not as great as what would be expected, especially after observing the performance differences using Super PI. These performance characteristics are inherent traits of scripting languages, and therefore little can be done to improve the execution time using this software setup.

What is important to gather from these benchmarks is the following:

1. The software executes successfully and provides the same answers across various platforms.

2. The accuracy of the software is consistent across various platforms.

3. The software executes in a reasonable amount of time across various platforms, despite differences in system performance.

Considering the original goals of this software, the benchmarks performed here demonstrate that the software has the capability to produce accurate results in a reasonable amount of time, and has the ability to work on various system configurations. Therefore, it is clear that any modern off-the-shelf computer system available for purchase at the present time (2006) will be capable of handling this software package. Since the distortion correction software is implemented using a scripting language, the execution of the program is heavily dependent on memory. It is for this reason that the Athlon64 3800+ and Pentium M machines perform at the top of the test group. The Athlon64 3800+ machine features the fastest (i.e., lowest latency) memory of the test group, while the Pentium M machine features the most memory of the group—at least 1 gigabyte more than the others. The computations themselves are

relatively simple, and are therefore not as dependent on CPU performance. This is the reason why the Pentium4 machine performed in the middle of the pack, despite it having the highest CPU clock speed of the group.

In order to gain the best performance, the benchmarks indicate that a computer possessing a robust memory setup will achieve the highest software performance. The system should either have very high performance, low latency memory (such as the memory used in the Athlon64 3800+ system), or should be equipped with a large amount of memory (such as the configuration used in the Pentium M machine). Obviously, the fastest system components will yield the highest performance. In terms of the components tested, the optimal system would feature a powerful CPU comparable to that of the Athlon64 3800+ test machine, a fast hard drive similar to that used in the Athlon64 3800+ test machine, and high quality memory similar to that used in the Athlon64 3800+ and Pentium M machines. Such a system will provide the highest level of performance; however, the price of such a system may not be easily justifiable, nor is such a system necessary to execute the software, since an average desktop system is more than capable of delivering similar performance at a much lower cost.

The detailed specifications of the computer system that will run this software cannot be determined at this time. The good news is that the detailed specifications do not need to be outlined at this time. This decision will be left totally to the medical institution. Whichever computer system the institution decides to execute this software, the institution can have confidence in the software's reliability and performance on the computer system they assemble. Further testing will need to commence to de-

termine the software's reliability and capability in Unix-based environments, but the benchmarks performed here confirm that the Windows-based implementation of the gradient nonlinearity distortion correction software executes reliably and correctly.

6.7 Accuracy of Target Localization

Verifying the accuracy and quality of the distortion correction using the calibration phantom only provides an initial verification. An independent verification must be performed to truly realize the accuracy and quality of the distortion correction method. The purpose of this project is not to merely provide undistorted images of the oil-filled phantom. Indeed, the bulk of the results presented here involve the oil phantom since the majority of the data acquired involved the oil phantom. When the distortion correction software is used for its original intended purpose, the corrected images will be used for localizing targets for use in functional proton radiosurgery. Therefore, it is necessary to verify the accuracy of target localization using images corrected by the distortion correction software.

6.7.1 Localizing Targeting Markers

The process of performing target localization to test the distortion correction software involves sophisticated hardware. A specialized phantom containing targeting markers with known locations was used for these tests. This phantom, the *Lucy phantom*, is a plastic sphere with provisions for 20 small spherical MRI markers. These markers are filled with oil in order to provide a clear signal in acquired images. To further enhance the appearance of the markers in the images, plastic bags filled with water were packed around the Lucy. Then the Lucy was affixed to the Leksell® MR Indicator frame and Leksell® Coordinate Frame Model G. This setup is demonstrated in Figure 6.31. The resulting images acquired during MRI scanning appear similar to those shown in Figure 6.32.

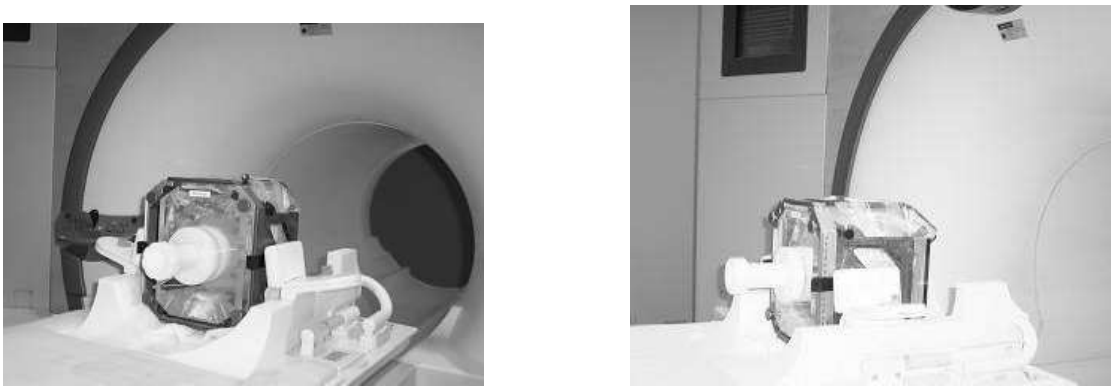
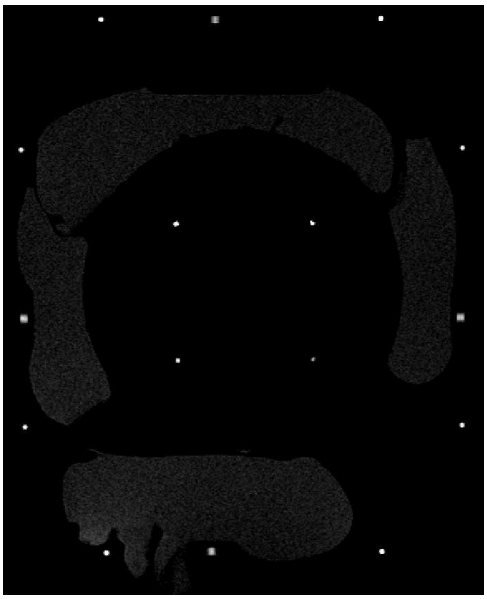


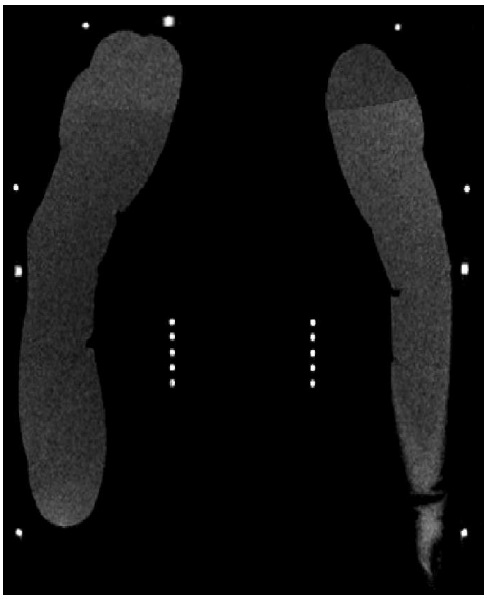
Fig. 6.31: Lucy phantom affixed to stereotactic frame, resting in head coil in MRI scanner.

The Leksell® MR Indicator frame and Leksell® Coordinate Frame are used to create a stereotactic coordinate system. During functional proton radiosurgery, all target regions will be identified using stereotactic coordinates. In order to calculate the stereotactic coordinates of all the targets, it is necessary to perform a *stereotactic transformation*. The transformation converts the target coordinates from the DICOM image coordinate system to the coordinate system created by the Leksell® Coordinate Frame. For the specifics concerning the Leksell® hardware and the stereotactic transformation, refer to Appendix C.

To aid in the localization of the markers, a commercially-available radiation treatment planning system was employed. The Odyssey® Radiation Treatment Planning Software[3]greatly simplifies the process of localizing target regions. Odyssey® contains the specifications of the Leksell® MR Indicator frame, and calculates the precise locations of the fiducial marks on the frame. Localizing target regions is a matter



(a) Axial scan.



(b) Coronal scan.

Fig. 6.32: Example scans of Lucy phantom ((a) and (b)). Observe the three dots produced by the stereotactic frame, on the outer edges of the images. The third dot indicates the position of the slice, relative to the frame.

of clicking on the desired target regions. Odyssey® then performs the stereotactic transformation and converts the target coordinates. The coordinates of the localized regions can then be exported in a file for use later on. Figure 6.33 provides an example of the Odyssey® program interface used for target localization.

6.7.2 Scan Details

Since the composition and structure of the Lucy phantom is vastly different than the oil-filled cube phantom, scanning the Lucy phantom was accomplished with slightly

[3] Odyssey® was used with permission by PerMedics, Inc., San Bernardino, CA.

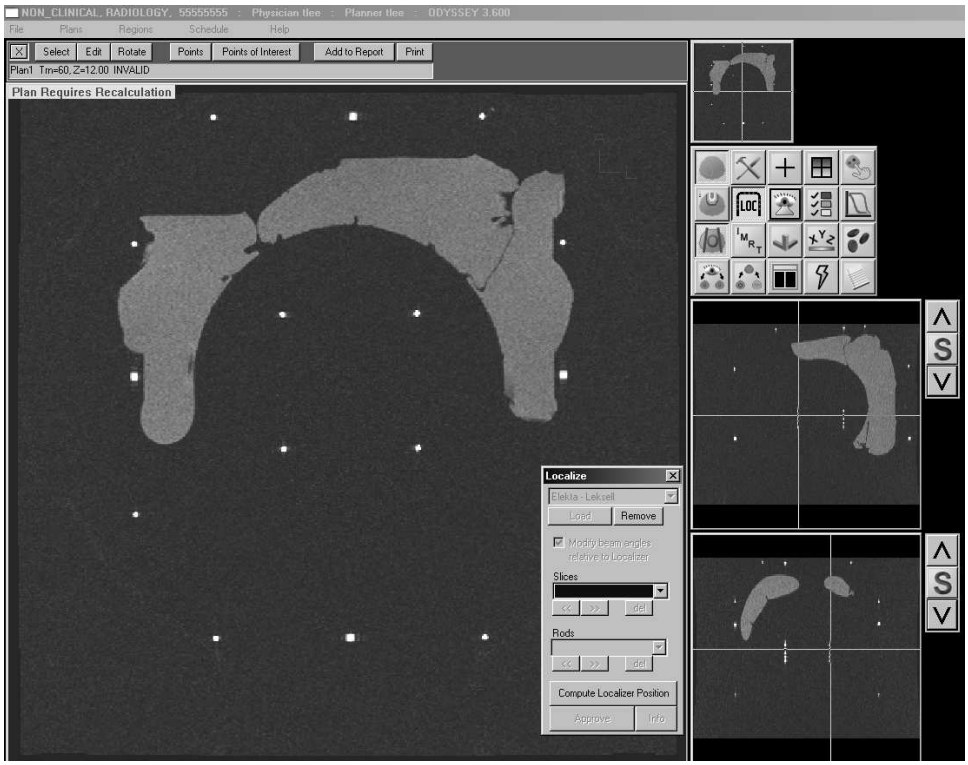


Fig. 6.33: Localizing target regions in the Odyssey® Radiation Treatment Planning Software.

different settings than those used for the oil phantom. This was done in order to obtain optimal image contrast and detail, especially for the markers inside the Lucy phantom. The MRI scanner was configured using the following settings:

- TR: 2010 ms
- TE: 2.4 ms
- Flip Angle: 20 ms
- Field of View (FOV): 300 mm^2
- Matrix: 512 x 512
- Voxel Size: 0.586 x 0.586 x 2.000 mm

- 1 Slab, 112 partitions (slices) per orientation
- Sequences: axial, coronal
- All scanner filters disabled

Images were acquired with the scanner filters disabled, to demonstrate the full potential of the gradient nonlinearity distortion correction. The Lucy phantom and the coordinate frame were placed inside the headcoil, and centered in a manner similar to that used for the oil phantom. Precision plastic shims were used to level the phantom assembly, and the orientation was checked with a level. The phantom was then centered using the scanner's laser sight indicator, to help ensure the phantom assembly is positioned as close to gradient isocenter as possible.

6.7.3 Specifications of Lucy Phantom

The Lucy phantom is a complicated piece of hardware, to say the least. The platform inside the spherical structure contains channels in which up to 20 markers can be placed. A detailed schematic of this structure is displayed in Figure 6.34. Observing the size of this structure, and therefore the size of each markers, indicates the difficulty of this step. The markers are quite small, and are placed only millimeters apart from one another.

To compare the measured locations of these markers in the images, the Lucy phantom was sent out to a local engineering company, which took the time to accurately measure the location of each marker, to the nearest thousandth of a millimeter. The measurements are expressed in frame coordinates, meaning that the three dimensional locations of the markers are in reference to the coordinate system created by

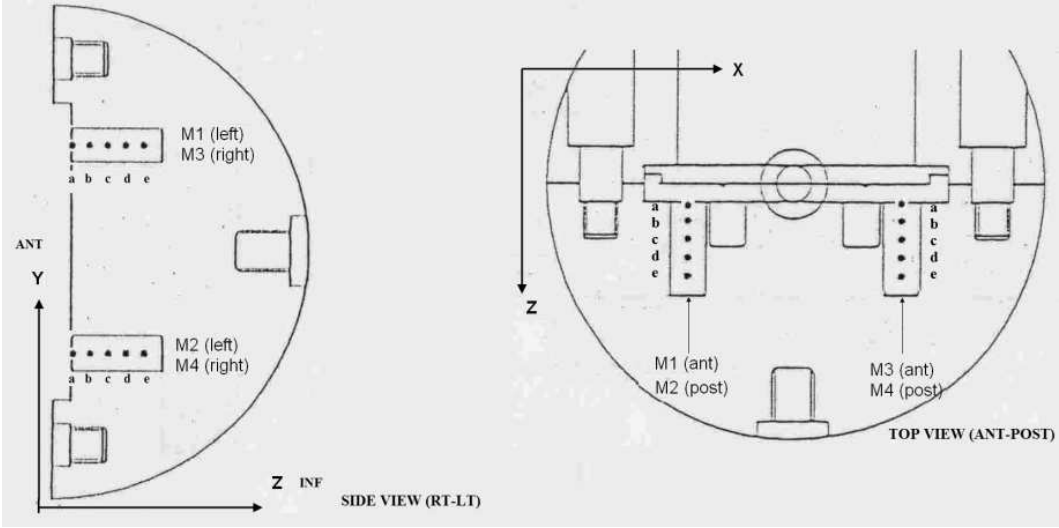


Fig. 6.34: Detailed schematic drawing of Lucy phantom, demonstrating provisions for targeting markers.

the Leksell® Coordinate Frame.

6.7.4 Numerical Results

The gradient nonlinearity distortion correction was applied to the images of the Lucy phantom, and the markers were localized in Odyssey®. Comparing the actual locations of the fiducials and their locations in the images yields the errors displayed in Table 6.45. In the uncorrected images, the fiducials are displaced up to 4.15 mm from their correct locations. Since the fiducials are located in the outer regions of the images (closer to the boundaries of the image), the effects of the gradient nonlinearity distortion are more pronounced. Offsets of approximately 4 mm are common in these regions, as noted by other works [33, 3, 36]. Correcting for gradient nonlinearity distortion reduces the error substantially to at the most 0.85 mm.

Due to the anisotropy of the resolution of the MRI studies (0.586 mm between pixels, 2.000 mm between slices), coordinates were defined using both the axial and

	Uncorrected	Corrected
Max Error (mm)	4.34	0.85
Average Error (mm)	2.15	0.45

Tab. 6.45: Errors calculated from localizing fiducials in uncorrected and corrected MR images.

coronal studies. The $x-$ and $z-$coordinates were defined using the coronal study and the $y-$ coordinates were defined in the axial study. This is to reduce the errors associated with localizing the markers in between slices, since the resolution between slices is nearly $\frac{1}{4}$ of the resolution between pixels. Because the phantom was positioned exactly the same in the axial and coronal studies, the coordinates in both sets of images directly correspond, allowing accurate target localization to be possible between the studies.

To determine the effectiveness of the gradient nonlinearity distortion correction, the markers were localized in both uncorrected and corrected studies. The reported positions of the markers in images with and without distortion correction are displayed in Table 6.46.

To provide a reference, the markers were also localized using a high-resolution CT scan of the Lucy phantom. The measurements acquired from the CT scan will serve as the reference benchmark, since the CT images are not subject to distortions, inhomogeneities, or artifacts. To promote consistency in target localization between the CT and MR images, the CT scan was configured very similarly to the MRI scans. The acquisition parameters were as follows:

- Matrix: 512 x 512
- Voxel Size: 0.585938 × 0.585938 × 0.625000 mm

- Field of View (FOV): 300 mm^2
- 1 Slab, 256 partitions (slices)

Therefore the resolution of the scan is nearly isotropic, allowing for nearly the same degree of accuracy of target localization in all three dimensions throughout the study. This is important because the markers can only be localized in the axial plane in CT. Since the markers are separated by small distances in z, high resolution is necessary in z to obtain accurate measurements.

The reported positions of the markers are displayed in Table 6.47

6.7.5 Error Analysis

Performing an error analysis (see Appendix A) on the calculated positions of the Lucy markers yields the results shown in Table 6.48. The difference between target localization using uncorrected and corrected MR images is quite apparent. Target localization in the distortion-corrected MR images provides accuracy that is very similar to that of CT. To determine the overall accuracy of both methods, consider three standard deviations from the mean. Calculating three standard deviations will include 99.5% of the measurements, presuming that the measurements follow a Gaussian (normal) distribution.

When considering the standard deviations from CT, the highest value was calculated in x (0.36 mm), while the lowest standard deviation was calculated in y (0.17 mm). Therefore, 99.5% of the targets localized in CT will possess an accuracy of between 0.51 mm to 1.08 mm. Observing the standard deviations from uncorrected MR images, the standard deviations range from 0.355 mm in y to 1.11 mm in z.

Therefore, 99.5% of the targets localized in the uncorrected images is between 1.07 and 3.33 mm. Performing target localization on distortion-corrected images yield standard deviations ranging from 0.16 mm in x, to 0.33 mm in y. Therefore, 99.5% of the targets localized in MRI will possess an accuracy of between 0.48 mm to 0.99 mm.

The errors calculated in Table 6.48 show that the accuracy of target localization in distortion-corrected MR images is slightly better than that of CT. This conclusion is *very* substantial, being that CT is currently the gold standard in most medical imaging applications. Additionally, the accuracy of target localization using distortion-corrected MR images is submillimeter, possessing an accuracy of between 0.48 mm and 0.99 mm. The accuracy of target localization using corrected images is noticeably higher when compared to the accuracy obtained when using uncorrected images. The gradient nonlinearity distortion reduces the accuracy of target localization by approximately $\frac{1}{3}$. Clearly, submillimeter accuracy is not possible due to the gradient nonlinearity distortion, since the accuracy ranges between 1.07 and 3.33 mm.

Keep in mind that unlike MRI, CT is not subject to any distortions or artifacts. The primary disadvantage to using CT for target localization is the limit to the accuracy in all three dimensions. Since CT only provides images in the axial plane, it is very difficult to obtain isotropic resolution in localizing targets. Since MRI provides scanning in all three planes, it is possible to localize targets using two different studies, to obtain more accurate measurements. This is precisely the method used to perform this verification: the x and z coordinates were obtained from the coronal study, while the y coordinates were obtained from the axial study. Therefore, the limit of accuracy

for target localization using this method was one pixel, corresponding to 0.586 mm. The resolution of the CT scan was very close to isotropic, with pixels being 0.586 mm and the distance between slices being 0.625 mm. Therefore, the resolution of target localization was very similar in both studies.

6.7.6 Conclusion

Calculating the locations of the markers in the Lucy phantom serves as a solid independent verification of the accuracy of the software-based MRI gradient nonlinearity distortion correction. The distortion correction needs to perform consistently across numerous experimental setups. Verifying this characteristic will provide an additional metric with which to measure the success of the method.

Even though the accuracy of target localization calculated for MRI is indeed submillimeter, the results are far from perfect. The calculated locations of the markers are subject to several factors:

- The quality of the images.
- The characteristics of the study (most specifically, pixel size and slice thickness).
- The accuracy of locating the markers and fiducials in Odyssey®.
- The Leksell® Coordinate Frame Model G.

Obviously, the quality of the images will ultimately determine the accuracy of target localization. If the images of the markers are very poor, then the task of locating the center of the marker becomes very difficult. The appearance of the markers was very different in the axial and coronal studies, see Figure 6.32. The

quality of appearance of the markers in the axial study were very low. Observe in Figure 6.32(a) how the markers appear to be rectangular, rather than circular. In contrast, the image of the markers were very sharp in the coronal study, Figure 6.32(b). Because of this factor, the coronal study was used to calculate the x and z positions of the markers. Unfortunately, the axial study was still needed to obtain the y coordinates, which reduced the accuracy of all the measurements in y. The coronal study was also used for x and z because of the arrangement of the markers in the Lucy phantom. The markers are separated by approximately 5 mm in z. With the axial study featuring a resolution of approximately 2 mm in z, localizing the center of the marker proved to be a very difficult task. Locating the markers required traversing between ten different slices. Due to the thickness of each slice, the centers of most of the markers did not lie directly in the slice; rather, the centers often existed between slices. In contrast, the markers appeared in two slices in the coronal study, making target localization very simple. The measurements obtained for the markers in Table 6.46 are about as accurate as possible, considering the quality of these images and the arrangement of the markers in the phantom.

Another closely-related factor to image quality is the manner in which the study was configured. The most important factors that determine the accuracy of target localization are pixel size and slice thickness. Keep in mind that in order to maintain suitable signal-to-noise ratio in MRI, it is necessary to keep the slices relatively thick compared to the pixel size. Localizing targets using two different studies will overcome the issue of anisotropic resolution caused by thick slices. Using this method, the obtainable accuracy is now based only on the pixel size. Obviously, increasing the

size of each pixel will degrade the resolution of the images, and therefore degrade the accuracy of target resolution. The pixel size for the studies performed were 0.586 mm. This means that obtaining submillimeter accuracy leaves very little room for error. Localizing a target with an error of two pixels will compromise submillimeter accuracy. If the pixel size were reduced, such that the studies were performed with the same pixel size as used for the oil phantom (0.391 mm), then the size of the errors calculated in Table 6.48 would most likely be reduced.

The third factor that contributes to the accuracy of target localization is the accuracy of defining the locations in Odyssey®. This is the most critical step in the target localization process, and is also the factor that is subject to the most error. Improper specification of target regions in Odyssey® will yield inaccurate results. The user simply clicks on each location in order to define each target point. Clicking on the incorrect location is quite easy to do. Thankfully, Odyssey® allows the user to magnify portions of the image in order to increase the accuracy of clicking on the target regions with the mouse. Like any system, practicing target localization in Odyssey® is essential to achieving accurate measurements.

The final factor is specific to the studies performed for this verification. During target localization, an inhomogeneity was identified in the inferior region of the coronal images. After some investigation, it was determined that the Leksell® Coordinate Frame Model G used during these scans introduced an extra inhomogeneity in the images that could not be accounted for using the gradient nonlinearity distortion correction. Coincidentally, the frame is metallic. A discussion concerning this issue is presented in Section 7.1.3. To work around this problem, the fiducial marks

were defined in their expected locations in the inferior portions of the coronal images. Drawing a straight line on top of the fiducial rod yields the "expected" position of the inferior fiducials. Performing this method resulted in the fiducials being defined with submillimeter accuracy, compared to the actual measurements of the indicator frame. Thankfully, the markers existed superior to the affected region, far enough away that the inhomogeneity did not influence the image of the markers. This was trivial to accomplish, since the remaining portions of the fiducial rods were very straight.

Addressing these issues will yield more accurate target localization results. With more refinements to the experimental setup, and with more practice localizing the markers using the treatment planning software, obtaining more accurate results for target localization in MRI is possible. Even so, the current accuracy of MRI-based target localization using this experimental setup is submillimeter, and therefore meets the original requirements specified by LLUMC.

Marker	Actual X	U X	C X	Actual Y	U Y	C Y	Actual Z	U Z	C Z
M1a	70.022	71.200	70.200	128.800	128.700	128.400	129.450	129.600	129.300
M1b	70.019	71.200	70.200	128.800	128.500	128.300	134.450	133.600	134.300
M1c	70.017	70.700	70.200	128.800	128.700	128.900	139.450	137.600	139.300
M1d	70.014	70.900	70.200	128.800	128.400	128.200	144.450	143.600	144.300
M1e	70.012	70.500	70.200	128.8	128.700	128.600	149.45	149.600	149.300
M2a	70.312	71.600	70.400	68.836	68.900	68.600	129.460	128.700	129.100
M2b	70.309	71.400	70.300	68.837	68.700	68.600	134.460	132.700	134.600
M2c	70.307	71.200	70.300	68.838	68.700	68.500	139.460	138.700	139.100
M2d	70.304	71.100	70.300	68.839	68.400	68.400	144.460	142.700	143.600
M2e	70.302	71.300	70.300	68.84	68.600	68.800	149.46	148.700	149.100
M3a	129.980	131.500	130.000	129.090	128.900	128.800	129.480	127.700	129.700
M3b	129.980	131.200	130.000	129.090	129.000	128.800	134.480	133.700	134.700
M3c	129.980	131.000	130.000	129.090	128.700	128.800	139.480	139.700	139.700
M3d	129.970	130.800	130.000	129.090	128.900	129.200	144.480	143.700	144.700
M3e	129.97	130.600	130.000	129.09	128.300	129.100	149.48	149.700	150.200
M4a	130.270	131.500	130.500	69.126	68.800	68.600	129.490	128.800	129.600
M4b	130.270	130.900	130.500	69.127	68.900	69.100	134.490	132.800	134.600
M4c	130.270	131.100	130.500	69.128	68.800	69.000	139.490	138.800	139.600
M4d	130.260	130.900	130.500	69.129	68.900	68.900	144.490	142.800	144.100
M4e	130.26	130.800	129.900	69.13	68.300	68.800	149.49	148.700	149.600

Tab. 6.46: Actual and measured locations of the Lucy markers in uncorrected (U) and corrected (C) MR images, in mm.

Marker #		Actual X	CT X		Actual Y	CT Y		Actual Z	CT Z
M1a		70.022	69.900		128.800	128.700		129.450	129.700
M1b		70.019	69.900		128.800	128.700		134.450	134.700
M1c		70.017	69.900		128.800	128.800		139.450	139.700
M1d		70.014	69.900		128.800	128.800		144.450	144.700
M1e		70.012	69.900		128.800	128.900		149.450	149.700
M2a		70.312	70.000		68.836	68.900		129.460	129.600
M2b		70.309	70.000		68.837	68.900		134.460	134.600
M2c		70.307	70.000		68.838	69.000		139.460	139.600
M2d		70.304	69.900		68.839	69.000		144.460	144.600
M2e		70.302	69.900		68.840	69.100		149.460	149.600
M3a		129.980	129.700		129.090	128.700		129.480	129.100
M3b		129.980	129.700		129.090	128.800		134.480	134.800
M3c		129.980	129.700		129.090	128.800		139.480	139.800
M3d		129.970	129.700		129.090	128.900		144.480	144.800
M3e		129.970	129.700		129.09	128.900		149.48	149.800
M4a		130.270	129.700		69.126	68.900		129.490	129.700
M4b		130.270	129.700		69.127	69.000		134.490	134.700
M4c		130.270	129.700		69.128	69.000		139.490	139.700
M4d		130.260	129.700		69.129	69.100		144.490	144.700
M4e		130.260	129.700		69.130	69.100		149.490	149.700

Tab. 6.47: Actual and measured locations of the Lucy markers in CT images, in mm.

	CT	MRI Uncorrected	MRI Corrected
X			
Total Residuals (Error) (mm)	1.63189092772773	4.33281294311213	0.724339699312406
MSE (Variance) (mm)	0.133153400000004	0.938663400000001	0.0262333999999993
RMS (Standard Deviation) (mm)	0.364901904626441	0.968846427458966	0.16196728064643
Y			
Total Residuals (Error) (mm)	0.797282885806541	1.59162181437678	1.40002142840745
MSE (Variance) (mm)	0.0317830000000003	0.126663000000002	0.0980030000000013
RMS (Standard Deviation) (mm)	0.178277872996063	0.355897457141803	0.313054308387541
Z			
Total Residuals (Error) (mm)	1.08857705285386	4.96094749014747	1.47410990092327
MSE (Variance) (mm)	0.0592499999999995	1.23055000000002	0.108650000000001
RMS (Standard Deviation) (mm)	0.243413228892761	1.10930158207767	0.329620994476991

Tab. 6.48: Errors calculated from localizing Lucy markers in CT and MR images, with and without distortion correction.

7. SUMMARY, CONCLUSIONS, AND RECOMMENDATIONS

The quantitative and qualitative results demonstrate the capability of the software-based MRI gradient nonlinearity distortion correction. However, these results are useless if little can be derived from them. Simply throwing out numbers and showing pretty pictures will do little as far as forwarding knowledge in this research topic area. In the same manner, failure to provide explanations behind the results will greatly reduce the validity of the project. Therefore, the conclusive analysis will include several key features, in order to maximize the usefulness of the results acquired from this project.

Regardless of the effectiveness or success the software, it is imperative that the sources of error be identified. No matter how robust the distortion correction method, the implementation will always be prone to errors. Even the best software distortion correction method can produce invalid results if the errors associated with experimental factors are too large. What sources of error need to be controlled? How can errors creep into the correction results, and potentially invalidate the distortion correction? Blindly implementing this distortion correction method without understanding where problems can occur will cause trouble in the future. Many problems were identified and dealt with during the development of this project. To minimize future issues, these problems are discussed, and their solutions presented.

Interpreting the results is a key step in determining the true effectiveness of the distortion correction method. It is necessary to determine if the software has met the original requirements set forth by LLUMC. If the software has indeed met these requirements, then future researchers can be assured that the information provided in this thesis is useful and valid for future endeavors in MRI research. On the other hand, if the software has not met the original requirements, then it is necessary to identify the reasons for this failure, be it shoddy data, faulty software design, or poor experimental setup. Without this analysis, the results are utterly useless.

The distortion correction method described here is certainly not the only one in existence today. There are many other implementations that handle the issue of gradient nonlinearity distortion for MRI. All of these methods have a certain performance level. Now that the success of the distortion correction software described in this work has been demonstrated, how does it compare to previous methods? Has the distortion correction method improved upon previous implementations, to provide greater accuracy in MRI-based applications?

It goes without saying that no distortion correction method is perfect. Even the method described in this work is not perfect. There will always be room for improvement in any method, be it in algorithm design, software design, or experimental setup. All of these factors contribute to the effectiveness of the distortion correction method. Failure to properly address any one of these areas will degrade the quality of the distortion correction. Regardless of the effectiveness of the distortion correction method described in this work, there are areas that could use some improvement. Future research in this area can take advantage of these improvements to produce

an even more accurate, more effective distortion correction method, to obtain even better results than described here.

7.1 Primary Sources of Error, and Solutions

Analysis of the numerical results obtained from the distortion correction software reveals that the results are prone to errors. This should not be a surprising observation, given that the software and experimental setup are based on imperfect conditions. As mentioned in Section 6.4, the perfect experimental setup is impossible to achieve. Therefore, the software MRI gradient nonlinearity distortion correction will always be prone to numerical inaccuracies. Thankfully, the numerical analyses presented in Sections 6.2 and 6.4 show that these errors are not sufficiently large, such that submillimeter target localization is still feasible.

The primary sources of these errors have been identified as the following:

1. Rounding and truncation errors.
2. Quality of scanner signal and image processing.
3. Accuracy of spherical harmonics expansion.
4. Phantom centering in MRI scanner.
5. Proper calibration of shim coils in MRI scanner.
6. The stereotactic coordinate frame.

The first three sources of error are relatively fixed, and little can be done to improve the accuracy of the distortion correction method by way of modifying these

attributes. As with every experimental system, there are sources of error that cannot be eliminated, and must simply be accepted as being a natural part of the system. The MRI gradient nonlinearity distortion correction method described in this work is no different.

Any scientific software application will be prone to rounding and truncation errors. Finite precision is a simple inherent trait of all computer systems. Until the advent of infinite computer systems—machines that possess infinite storage space and can represent values with infinite precision—all calculations performed on computers will have a limit to their accuracy. Since *all* calculations possess some degree of error due to rounding and truncation, the goal is therefore to *reduce* the errors present in the calculations. To put it simply: there exist efficient and inefficient methods of performing calculations. Inefficient methods are careless and do not consider the limitations of finite storage space on computers, and therefore do not preserve numerical accuracy in the results. Efficient methods, on the other hand, take special precautions to avoid error propagation, thereby minimizing the size of errors in calculation results. The methods described in this work focus heavily on preserving numerical accuracy, and the numerical results presented in Sections 6.2 and 6.4 demonstrate this ideal in action.

Considering rounding and truncation errors once again, there is also the issue of the quality of the signal and imaging processing techniques employed by the scanner to produce MR images. The software relies on the output of the MRI scanner in order to perform its duties. Many complex procedures are performed by the MRI scanner to produce images, see Section 1.5. All of these operations are handled by a

computer, and are therefore subject to finite precision. The results of the signal and image processing performed by the scanner exhibit numerical errors themselves. The quality of the techniques is determined by the accuracy of the results. Although the signal and image processing techniques employed by the scanner are most likely of very high quality, and the errors inherent in the results are most likely quite small, it is still necessary to mention that the errors are indeed present. The presence of these errors will indeed influence the accuracy of the distortion correction method to some degree. But given that there is probably little room for improvement in terms of numerical accuracy of the scanner signal and image processing techniques, and given that there is probably no way to modify the scanner software, it would be of little use to modify these techniques in order to improve accuracy. The errors associated with sampling MR images is simply an inherent trait of the data, and cannot be changed.

The accuracy of the distortion correction method is also heavily dependent on the accuracy of the spherical harmonics expansion, used to define the distortion (and distortion correction) models. The true model for the gradient nonlinearity distortion is described by Equation 1.28, an infinite series. However, due to the finite precision of computers, it is impossible to fully represent this expression on a computer. Because of this, it is necessary to represent this model as a truncated expression, described by Equations 2.8 - 2.10. Truncating the result will influence the accuracy of the model. The expressions used for this work are the same as those used by Langlois, et al. [33], and include the first five orders. Including the higher order terms will indeed increase the accuracy of the model. However, there needs to be a limit to the number of terms to include. As noted by Langlois, the higher order terms contribute

less and less to the final result, therefore it is sufficient to only consider the first five terms [33]. GE follows this practice; Siemens, on the other hand, includes the first eleven terms [36]. While the Siemens implementation may be more accurate because of the inclusion of the higher order terms, the increased accuracy obtained by the larger expressions may not be justifiable. The same holds true for this software: the data simulations demonstrated that the correction method is capable of producing submillimeter accuracy using five terms, therefore increasing the number of terms in the spherical harmonics expansion may not be justifiable to increase the accuracy of the system. The errors produced as a result of truncating the infinite spherical harmonics expression are not sufficient large, such that the correction results are greatly affected.

The final three sources of error, unlike the first three, are factors that can be controlled. This is very good news indeed, because both of these factors can greatly influence the overall accuracy of the correction results. Improper phantom centering creates asymmetry in the gradient nonlinearity distortion in the datasets, which contradicts the software's assumption of symmetric distortion. Both the real and simulated numerical results show that proper phantom centering is crucial to preserving the accuracy of the correction method, see Sections 6.2 and 6.4. Likewise, proper calibration of the shim coils in the MRI scanner is important to preserving accuracy. The shim coils are used to reduce artifacts, distortions, and noise in the acquired images, see Section 1.7.4. If the shim coils are not properly adjusted prior to scanning, there is an increased likelihood of artifacts and distortions in the datasets that are not taken into consideration by the software, thereby causing discrepancies

in the correction results. Finally, the use of a metallic stereotactic coordinate frame introduced inhomogeneities in the main gradient field, thereby creating extra distortions in the images. These inhomogeneities need to be properly addressed in order to obtain accurate correction results and, therefore, accurate localization of anatomical targets.

Phantom centering and calibrating the shim coils are two crucial factors that must be taken into account prior to scanning. Failure to do so will compromise numerical accuracy. Additionally, the use of a metallic stereotactic coordinate frame creates difficulties in performing stereotactic localization, since the images are prone to additional distortions not handled by the gradient nonlinearity distortion correction. Analyzing each of these factors will provide insight into how accuracy is affected, as well as possible solutions to these problems.

7.1.1 Phantom Centering

The issue of phantom centering is—and always will be—a "hot topic", in terms of this application. As was the case in the five studies performed for testing, ideal phantom centering was achieved only once, despite efforts to increase the accuracy and confidence of the process. The numerical results in Sections 6.2 and 6.4 clearly demonstrate the sensitivity of the distortion correction method to phantom centering. Improper phantom centering is also very apparent in the acquired images themselves. Observe the several examples shown in Section 6.3. Therefore, it is vital that this issue be addressed so that consistent results can be achieved from this distortion correction method in the future.

Implications of Phantom Centering

The results of the distortion correction on both real and simulated data clearly demonstrated that the quality of the correction is dependent on the position of the phantom in the scanner. Most specifically, the phantom needs to be as close to the gradient isocenter as possible, in order to promote symmetry in the gradient nonlinearity distortion inherent to the dataset. Obviously, it is impossible to center the phantom *perfectly* in the MRI scanner, therefore the phantom will always exhibit some displacement from isocenter. Because of this, the data will also exhibit some asymmetry in the inherent distortion. Since it is impossible to achieve perfect phantom centering, and therefore perfectly symmetric distortion, the goal of the experimental setup is to position the phantom as close as possible to gradient isocenter.

The five experimental trials and the data simulation demonstrated that the quality of the distortion correction is based heavily on the position of the phantom in the scanner. In the experimental trials, the most accurate correction results were achieved when the phantom was placed to within a couple of millimeters of the isocenter. The trials that were conducted with the phantom displaced relatively far off center (12-13 mm) exhibited much larger errors. In the case of the 7-17-05 study, the quality of the distortion correction does not promote submillimeter accuracy in target localization. The accuracy of the correction is between 0.7 - 1.2 mm, depending on the dimension. On the other hand, the accuracy of the distortion correction is much higher in studies with properly centered phantoms. Improved phantom centering was achieved in the 11-6-05, 4-30-06, and 6-19-06 studies, where accuracies of 0.39 - 0.78 mm were observed.

A similar trend was also observed in the data simulation. In the simulation performed with a well-centered phantom, the accuracy of the distortion correction is on the order of 10^{-2} to 10^{-3} mm, while the accuracy of an improperly centered study is at best 10^{-2} mm. An entire order of magnitude is lost in the accuracy of the distortion correction as a result of improper phantom centering. Even though the accuracy of both trials is high, the degree of accuracy sacrificed by poor phantom centering is unacceptable.

It is apparent that in order to confidently achieve submillimeter accuracy in the distortion correction, proper phantom centering is necessary. Careless setup procedures will yield improper results, thereby denying the possibility of submillimeter accuracy in target localization. Therefore, strict measures must be taken in order to obtain consistent phantom centering in the MRI scanner. Several steps were taken to help promote consistent phantom centering for this project.

The Goal: Obtain Consistent Phantom Centering

Developing a method to achieve consistent proper phantom centering is a difficult task, to say the least. The oil cube phantom used in the experimental trials underwent several modifications in order to achieve proper phantom centering. In the initial studies, placing the phantom in the scanner was largely guesswork. Without proper visual references and knowledge, it is possible to place the phantom any which way in the scanner's headcoil. The modifications made to the phantom promote consistent phantom centering in all future studies.

The original plan was to produce a base for the phantom that would allow for

perfectly consistent and accurate placement of the phantom in the scanner, every time. The benefits of such a foot would be numerous:

1. The phantom would be placed in the phantom in the same position *every time,* thereby promoting consistency in the distortion correction results.
2. Placing the phantom would not require tedious procedures to ensure proper centering, saving valuable time and energy.
3. The foot would make it possible for any person to perform the distortion correction procedure using the phantom, without special training on how to center the phantom prior to scanning.

However, this solution proved to be too difficult, too expensive, and too specialized to implement. The reasons for this were as follows:

1. Contracting an engineering company to measure the headcoil and produce such a foot with very high accuracy would require a large sum of money that simply exceeded the operational budget of this project.
2. Placing the phantom to within one millimeter of isocenter for the purposes of obtaining the measurements for the foot is too difficult.
3. The foot produced would only match the particular phantom, and would only match the particular headcoil in the scanner. Therefore, the phantom would require different feet for various scanners.

Even though this solution would be ideal, it was simply out of reach for this project. To help make up for this shortcoming, several small, basic modifications were made to the phantom in an attempt to achieve consistent phantom centering.

Phantom Foot Modification

The studies conducted on 7-17-05 and 10-2-05 featured a phantom that was ill-shaped for the scanner's headcoil. The foot at the base of the phantom was too large to fit into the tray of the headcoil, displacing the phantom by 12-13 mm in Z. This large displacement caused a great degradation in the accuracy of the distortion correction. After conducting the 10-2-05 study, the foot was shortened by an engineering company to better fit into the tray in the headcoil. The cost to perform this seemingly basic and simple procedure was quite high (but nothing in comparison to what would be required to produce a custom-fit foot). By shortening the foot, and providing a better fit in the headcoil, it was now possible to position the phantom closer to gradient isocenter, thereby improving the symmetry in the distortion and increasing the accuracy of the distortion correction.

Fine-Tuning Phantom Position

Placing the phantom to within a couple of millimeters of gradient isocenter in the headcoil is a very difficult task. Due to limited time and funds, obtaining the *exact* shape and size necessary for the phantom's foot was impossible in this situation. Therefore, in order to position the phantom with a higher degree of accuracy, a set of high-precision plastic shims of known thicknesses were used. Placing the shims around the phantom in various locations fine-tuned the phantom's position. Using these shims, the phantom was placed to within one millimeter of isocenter in all dimensions in the 11-6-05 study. The shims present a simple, cost-effective method to position the phantom with a high degree of accuracy, without "breaking the bank"

by obtaining a custom made foot, which proved to be simply outside operational budgets for this project. With that consideration in mind, the plastic shims are a great alternative.

Creating Reference Marks

The MRI scanner utilized for this project featured a laser sight that project two red laser beams onto the headcoil, to provide a reference for centering the headcoil prior to scanning. To help achieve consistent phantom centering, the phantom was modified by etching straight lines on the anterior face of the phantom (which is the top face when the phantom is placed in the scanner). These lines corresponded to the laser beams projected by the scanner. Using the controls on the scanner, it was now possible to center the phantom in relation to the projected laser beams prior to performing a scan. The plastic shims and modified foot provided a limited degree of confidence and accuracy, since there was no visual reference to determine the phantom's position in the scanner before acquiring images. With the etched lines on the phantom's face, more consistent phantom centering is now possible when used in conjunction with the laser sight on the MRI scanner.

Conclusions

Phantom centering is a key factor in the accuracy of a distortion correction model. The various steps listed above help address the issue of consistently achieving proper phantom centering when calculating a gradient nonlinearity distortion correction. While each particular experimental setup will require specific steps to achieve this

goal, the knowledge and experience gained from this project can serve as a reference for future implementations. Not all MRI scanners are made equal: the headcoils of various makes and models may have slight differences in design that will ultimately determine the size, shape, and type of phantoms that can be used. Therefore, it is impossible to devise one generic, all-encompassing solution; each solution must be tailored specifically to the particular setup.

Ultimately, the solution to the phantom centering problem will depend on available resources. The idea of custom fabricating a phantom that will fit perfectly into a headcoil and will always be ideally centered requires lots of time, money, patience, and a knowledgeable engineering company. In the case where such a solution is not feasible, due to a lack of any of these resources, other creative solutions must be implemented. The actual solution that is devised for each situation does not matter, as long as the problem with phantom centering is addressed properly.

Before approaching the issue of phantom centering, it is first necessary to analyze the specific requirements of each project. Most notably, the important question to answer is: *what accuracy is acceptable for the gradient nonlinearity distortion correction?* If, in fact, submillimeter accuracy is not the goal, then phantom centering is not as important of a factor in the accuracy of the distortion correction. The distortion correction still works even with offcentered phantoms; the only difference between centered and offcentered phantoms is the accuracy obtainable in the distortion correction. For example, in the studies where the phantom was displaced by a relatively large distance from isocenter, the accuracy of the distortion correction was approximately 1.2 mm, which is on par with or more accurate than other published

methods. This accuracy may be sufficient depending on the application. If, however, the application requires the high precision that this project demands, then phantom centering is a very important issue that requires special attention. In these cases, the extra time, effort, and money put into providing consistent phantom centering is definitely worthwhile.

7.1.2 Effects of Magnetic Shimming

Applying a shimming procedure is an essential step in achieving accurate data during scanning. As discussed in Section 1.7.4, the shim coils are used to further improve the homogeneity of the main field, as well as reduce the field effects due to susceptibility differences in the scanned objects [2].

During software development, peculiarities in select images prompted an investigation into other sources of distortion. After performing additional studies, it was determined that the peculiarities found in the images were a result of failure to adjust the shim coils prior to scanning. As a result, the homogeneity of the main field was slightly compromised, increasing the errors in the correction procedure. Executing the MRI scanner's computer-based shimming procedure adjusted the shim coils properly, to produce a more homogeneous field. This effectively removed the discrepancies found in the images. An account of the investigation follows.

Investigating Image Peculiarities

The bulk of the development of the gradient nonlinearity distortion correction software package was conducted using the 11-6-05 MRI study. This study was the first

study that achieved proper physical centering of the phantom in the MRI scanner: the phantom's center was calculated to be less than 1 mm away from the gradient isocenter. Ensuring the phantom's proper centering is conducive to producing accurate distortion correction results. Even though the phantom was positioned to within 1 mm of the gradient isocenter, there were discrepancies in the correction results that could not be accounted for. The source of the increased error present in the calculations were uncovered only after in-depth numerical and qualitative analyses.

Numerical Analysis

Midplane fitting calculations confirmed that the phantom was indeed positioned very close to the gradient isocenter, as shown in Tables 7.1 and 7.2.

Midplane	**Variance (mm)**	**Standard Deviation (mm)**
X	0.011248	0.10606
Y	0.011774	0.10851
Z	0.0080939	0.089966

Tab. 7.1: Calculation error of midplane fitting in 11-6-05 study.

Midplane	**Plane Coefficients**
X	$[-(\frac{D}{A}) - (\frac{B}{A}) - (\frac{C}{A})] = [0.68583, -0.00427, 0.0063581]$
Y	$[-(\frac{A}{B}) - (\frac{D}{B}) - (\frac{C}{B})] = [0.00031188, 0.87814, 0.0037484]$
Z	$[-(\frac{A}{C}) - (\frac{B}{C}) - (\frac{D}{C})] = [-0.0025594, -0.0068872, -0.16967]$

Tab. 7.2: Plane coefficients of midplane fitting in 11-6-05 study.

Since the midplanes are the average of the distorted faces, they are positioned in the center of the coordinate system. The offset coefficient $(-\frac{D}{\alpha}, (\alpha = A, B, C))$ is therefore indicative of the centering of the phantom in the scanner. Considering this

value provides an accurate secondary representation of the positioning of the phantom in the scanner, to confirm the initial centering measurements performed during data preprocessing. The offset term of the x midplane shows that this plane is roughly 0.68 mm away from the isocenter ($-\frac{D}{A} = 0.68583$). The y midplane's offset term indicates that the y midplane is approximately 0.88 mm away from the isocenter ($-\frac{D}{B} = 0.87814$). Finally, the offset term for the z midplane indicates a distance of approximately 0.17 mm ($-\frac{D}{C} = -0.16967$).

The ideal planes also reflect this same positioning, since they are calculated from shifting of the midplanes. At this point, it seems safe to assume that the data provided in this study is as accurate as can possibly be, given the difficulty of positioning the phantom properly in the scanner.

To check the accuracy of the theoretical undistorted points, a plane was fit to each data set. The planes fit to the theoretical data sets were compared to the ideal planes calculated previously. The calculation of theoretical undistorted data points proved to be about as accurate as possible: the difference between the fit ideal plane and calculated theoretical undistorted points (i.e., the size of the errors) is on the order of machine precision, $\epsilon \approx 10^{-16}$:

Phantom Face	**Error**
-X	$9.94765710951133 \times 10^{-14}$
+X	$2.84281812164297 \times 10^{-14}$
-Y	$5.6845765581073 \times 10^{-14}$
+Y	$1.42108812320234 \times 10^{-13}$
-Z	$1.42490998938069 \times 10^{-14}$
+Z	$4.26334583052364 \times 10^{-14}$

Tab. 7.3: Accuracy of theoretical undistorted points in 11-6-05 study.

The parameters of the distortion correction model were then calculated based on the theoretical undistorted points. Again, the accuracy of the theoretical undistorted data points used in this calculation seems high, as shown in Table 7.3.

Finally, the distortion parameters were applied to the distorted data points, to produce corrected undistorted points. After thorough analysis, it was determined that a reasonable goal of both plane fitting and distortion correction calculations is to achieve a standard deviation of $\frac{1}{3}$ of a pixel (0.13021 mm). Generally speaking, three standard deviations (1 pixel) away from the mean will consist of 99.5% of the data points.

Phantom Face	Variance (mm)	Standard Deviation (mm)
-X	0.055254	**0.23506**
+X	0.063349	**0.25169**
-Y	0.023795	0.15426
+Y	0.025235	0.15885
-Z	0.018366	0.13552
+Z	0.020084	0.14172

Tab. 7.4: Results of distortion correction in 11-6-05 study.

However, this trend is not reflected in the distortion correction. When comparing the corrected data points to the theoretical undistorted points (i.e., the theoretical locations of the corrected points), the standard deviations are higher than expected, as indicated by the bolded values in Table 7.4. The corrections for all six faces possess errors that are higher than expected, most particularly, the x faces have standard deviations that are nearly twice as high as the other results. Because of the discrepancy in the x correction, all the other corrections also suffer, since all three sets of distortion parameters (K_x, K_y, K_z) are used to correct each face. An error in

one set of parameters will affect *all* correction procedures.

The interesting point to note here is the fact that all previous calculations were verified for their correctness. Although they cannot be fully proven to be "correct," they can be shown to not contain additional systematic errors. Such was the case in each step in the distortion correction methods, except for the last step.

Identifying the Culprit

Once these errors were discovered, an intense, in-depth analysis was conducted to locate the source of the discrepancy. The only two possible sources of the discrepancy were:

1. Programming errors that introduce errors into the data, which propagate through calculations and grow over time.
2. A systematic error represented in the data that was not accounted for in any previous step.

After a thorough investigation of the software package, the possibility of programming errors was eliminated because:

1. Error analyses indicated that there were no (significant) systematic errors introduced during each stage of data processing, as shown above.
2. During calculation of the distortion parameters and the actual correction procedure, data from all 3 dimensions pass through the same portions of code.

Therefore, an error in the code would produce similarly-sized errors in all 3 dimensions, not just in one. The fact that the errors are isolated to just one dimension

makes this problem truly unique.

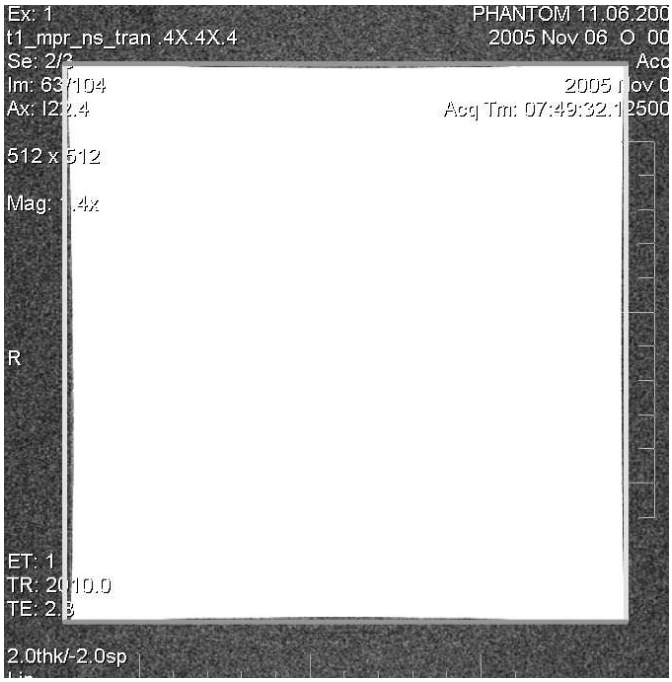


Fig. 7.1: Axial slice from 11-6-05 study, with a box drawn around the image of the phantom. Observe how the lower left corner is not collinear with upper left corner, indicating a problem.

The only possibility left is an additional systematic error in the data, prior to calculating planes and correcting the distortion. This makes sense because errors that occur prior to these steps would not be detectable in the methods used in the error analysis. Only upon visual inspection did the problem become apparent. Observe the apparent asymmetry on the left side of the phantom in Figure 7.1. The lower left corner of the phantom in the axial images is not vertically aligned with the upper left corner. The right corners of the phantom, on the other hand, are perfectly aligned. This phenomenon can be the result of either of two situations:

1. The phantom's faces are not the same length on the top and bottom in these images, causing the left face to have a slight tilt.

2. There is an additional distortion being introduced as a result of the scanner.

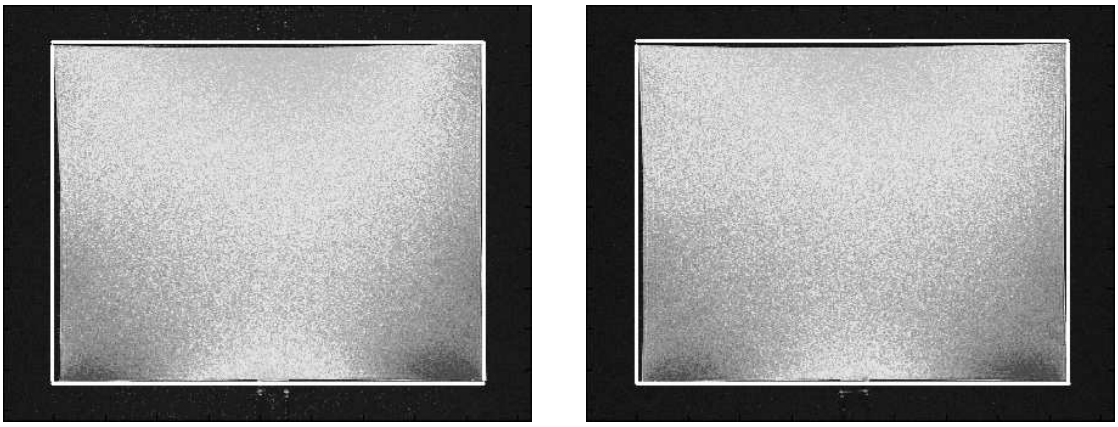
Upon careful analysis and measurement of the phantom, the faces were not found to possess a tilt as great as what was demonstrated in the MR images. Therefore, the discrepancy had to have been created as a result of an additional distortion in the scanner unaccounted for by previous steps.

Determining a Solution

To further investigate this problem, the corners in axial images were observed. The phantom was placed in the scanner, and one axial sequence was acquired. Then, the phantom was rotated such that the front face was pointing to the back of the scanner, therefore essentially producing a flipped set of images. Performing these two scans will determine the true cause of the discrepancy. If the corner that is affected appears on opposite sides of the images in the two scans (i.e., the bottom left corner in one scan, and the bottom right corner in the other scan, or vice versa), this indicates that the tilt is inherent to the phantom. On the other hand, if the corner that is affected in the images is on the same side of the image, then the scanner can be identified to be at fault. It is physically impossible for one face to be both tilted and straight at the same time.

An additional study was performed on 3-21-06, to determine the cause of the tilt. Example slices from both studies are shown in Figure 7.2. Visual examination of the two sets of axial images resulted in a misaligned bottom left corner *in both studies*.

Observe how these slices appear nearly the same as that shown in Figure 7.1. This observation eliminates the possibility that the phantom's physical structure is at fault, and therefore identifies that the scanner itself is the cause of the error.



(a) Axial slice.

(b) Axial slice, phantom reversed.

Fig. 7.2: Axial studies identifying source of discrepancy in lower left corners of both images.

After discussing the problem with physicists at LLUMC, it was decided that shimming the scanner's magnets might help eliminate the added distortion. In the two previous studies conducted during the development of this software package, no shimming procedures were performed prior to scanning. As a result, one of those studies, performed on 7-17-05, shows the exact same phenomenon. The second study, performed on 10-2-05, oddly enough, does not show this effect. The reason for this still remains a mystery, since there was no account of shimming being performed in any of the studies earlier than 11-6-05.

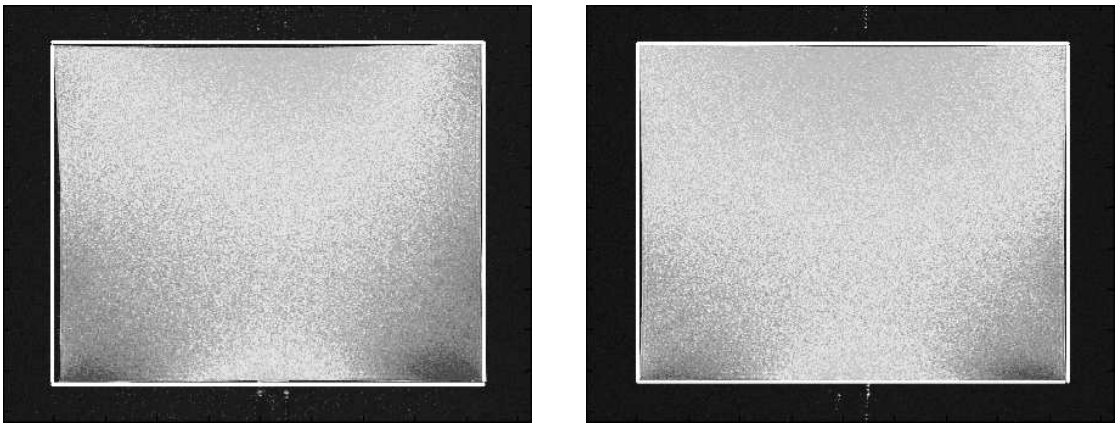
To further test the effect of the shimming procedure, MRI studies were performed on 3-29-06. Three axial studies were performed:

1. No magnet shimming, MRI scanner's distortion correction was applied to all images.

2. No magnet shimming, MRI scanner's distortion correction was disabled, all images feature the full effects of gradient nonlinearity distortion.

3. Magnet shimming procedure applied, MRI scanner's distortion correction was disabled, all images feature the full effects of gradient nonlinearity distortion.

A quick visual analysis showed that the study performed after shimming the magnets did indeed correct for the misaligned corners, as shown in Figure 7.3. The distortion that was caused as a result of that was no longer present in the images.



(a) Axial slice, no shimming. (b) Axial slice, shimming.

Fig. 7.3: Example images produced with calibration of the shim coils (b) and without (a). A box is drawn around the image of the phantom. Notice the location of the lower left corner in each image.

Conclusion

The differences in the acquired images acquired with and without calibration of the shim coils is quite noticeable. Properly calibrating the shim coils removed the extra distortion present in the lower left corner of axial images, and therefore created a more ideal dataset. The extra distortion that results from improperly calibrating the shim coils prior to scanning degrades the quality of the distortion correction. This is due to the fact that the distortion correction method assumes that the distortion inherent in the data is symmetric. The extra distortion in the images creates an apparent asymmetry in the images, thereby making this assumption incorrect. Because of the results observed in these tests, it was determined that prior to every MRI study conducted for gradient nonlinearity distortion correction, it is necessary to perform the magnet shimming procedure, to ensure the shim coils are in their ideal locations, to promote homogeneity of the main field. Performing this extra step prior to acquiring a set of images will help promote symmetry in the distortion inherent to the datasets, thereby increasing the quality and accuracy of the gradient nonlinearity distortion correction.

7.1.3 Stereotactic Coordinate Frame

After experiencing a wide range of problems with the acquired images for use in this project, several steps were taken to maximize the quality of the data obtained from the images. Proper centering of the phantom with respect to gradient isocenter promotes symmetry in the gradient nonlinearity distortion. Performing calibration of the shim coils increases the homogeniety of the main gradient field, while simultaneously

reducing additional distortions and artifacts in the image. Despite these measures, additional inhomogeneities were discovered when scanning the Lucy phantom. It was therefore a great surprise to discover that the inhomogeneities were caused by the stereotactic localization equipment.

The stereotactic coordinate frame used for the testing and verification of the distortion correction software was the Leksell® Coordinate Frame Model G, pictured in Figure 7.4. The frame is engraved with a rectilinear coordinate scale, in which the origin $(X, Y, Z) = (0, 0, 0)$ is located superior, lateral, and posterior to the frame on the patient's right side. The coordinates are graduated in millimeters and conforms with the coordinate system used in CT and MR scanning (the DICOM coordinate system, Figure 3.1).

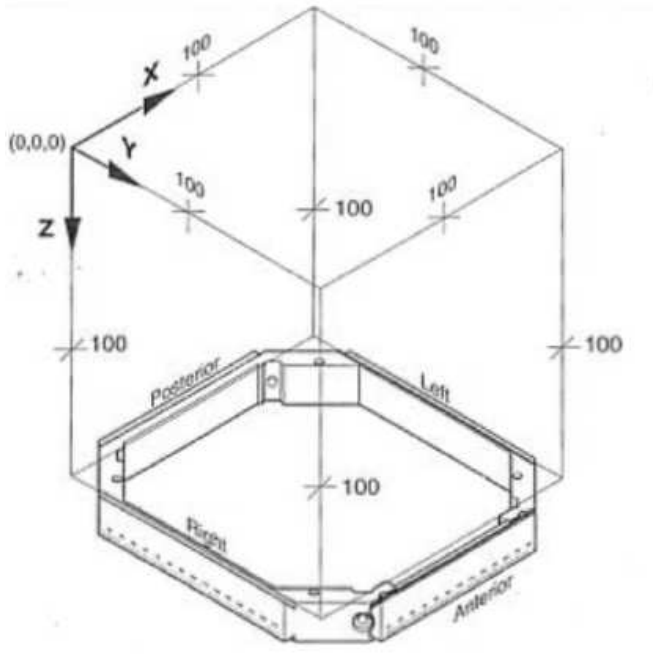


Fig. 7.4: Leksell® Coordinate Frame Model G.

The frame itself is a useful tool that provides a consistent system for performing

accurate target localization. The same frame is used in CT for performing target localization. When used in conjunction with the Leksell® CT Indicator Frame, the Coordinate Frame Model G allows physicians to accurately localize anatomical regions within CT images. The same theory holds true for MRI, when the coordinate frame is used in conjunction with the Leksell® MR Indicator Frame. However, there exists one fundamental problem: the coordinate frame is *metallic*.

7.1.4 Metallic Objects

In Section 1.5.3, the unique characteristics of magnetic resonance imaging is discussed and presented. One of the primary concerns with MRI is the fact that strong magnetic fields are utilized to produce images. Because of the strength of these magnetic fields (ranging between 1 and 20 tesla, depending on the scanner), the presence of metallic objects in and around the scanner causes several problems.

One hazard arises when ferromagnetic objects are placed near the scanner. Ferromagnetic objects can be attracted to the center of the magnet at very high velocities, causing bodily injury and serious mechanical damage. The magnetic field is constantly active, therefore special caution is necessary at all times when working around the MRI scanner. Obviously, it is not possible to eliminate *all* ferromagnetic objects from the scanning room. Objects such as tooth fillings, belt buckles, buttons, rings, and cardiac pace makers cannot always be removed prior to scanning. Their presence in and around the scanner poses potential dangers for the patient, as well as other personnel.

Besides the physical damage that can be caused by ferrometallic objects, there

exists another difficulty: inhomogeneities in the main magnetic field. The issue of resolving inhomogeneities in the magnetic field is the topic of great importance, discussed in many previous works [35, 33, 3, 36, 39, 37, 38]. Most specifically, calibration of the shim coils is recognized to be one of the most important—and one of the most effective—methods to reduce gradient field inhomogeneities [35]. Recall in Section 1.7 that producing a homogeneous, uniform magnetic field is crucial in order to properly correct gradient nonlinearity distortion. The existence of secondary inhomogeneities causes additional problems that will adversely affect the results of correcting for gradient nonlinearity distortion.

Keep in mind that the software-based gradient nonlinearity distortion correction is only designed to correct for gradient nonlinearity distortion. Distortions caused by other sources are not considered, since each individual source of distortion requires a specific method of correction. The six primary sources of distortion are discussed in brief in Section 1.6. Correcting for gradient field nonlinearities is only one step in handling MRI distortions; the other five source of distortion also need to be addressed.

In addition to these sources of distortion is the inhomogeneities caused by metallic objects in the scanner. Metallic objects perturb the magnetic field, creating inconsistencies in the gradient nonlinearity distortion. Without proper care and attention, these point sources of distortion will adversely affect the quality of any distortion correction method, especially the method described in this work. As is the case with other gradient nonlinearity distortion corrections [33, 40, 41, 3, 37, 38, 36, 39], gradient field inhomogeneities caused by the presence of metallic objects are not considered, and therefore are not handled properly in the software. Methods such as calibrating

shim coils and applying scanner filtering techniques only reduce the effects of these inhomogeneities, and are not effective in eliminating these problems.

The implications of the presence of metallic objects in and around the scanner should be quite clear now. Indeed simple steps can be taken to reduce the number of metallic objects (both ferromagnetic and non-ferromagnetic) prior to scanning. Objects that are permanently affixed to the patient are of course unable to be removed. But what about objects that comprise of the stereotactic reference system? Most specifically, what about the metallic Leksell® Coordinate Frame Model G?

7.1.5 A Metallic Stereotactic Coordinate Frame?

A quick visual inspection of the Leksell® Coordinate Frame Model G reveals one very important observation: the frame is metallic. The frame works well when used with the CT indicator frame, and provides accurate stereotactic localization using CT. However, the frame causes unique problems when used in MRI. The presence of the frame inside the scanner introduces inhomogeneities in the main gradient field, thereby adding additional distortions in the images that cannot be accounted for in the distortion correction method. The coordinate frame is affixed to the inferior portion of the MRI indicator frame, only a few centimeters away from the bottom of the right and left fiducial rods. As a result, the inferior regions of images experience an extra distortion that cannot be resolved using the gradient nonlinearity distortion correction method. Therefore, the accuracy of target localization is much less in the inferior portions of the image than that of the superior regions.

Thankfully, the markers in the Lucy phantom exist near the middle of the phantom.

The most inferior of the markers lie approximately 4 centimeters away from the coordinate frame. The inhomogeneities caused by the coordinate frame do not extend to the regions where the markers are positioned. Therefore, the markers can still be targeted accurately, without experiencing the inhomogeneities to as great of an extent as the inferior portions of the fiducial rods.

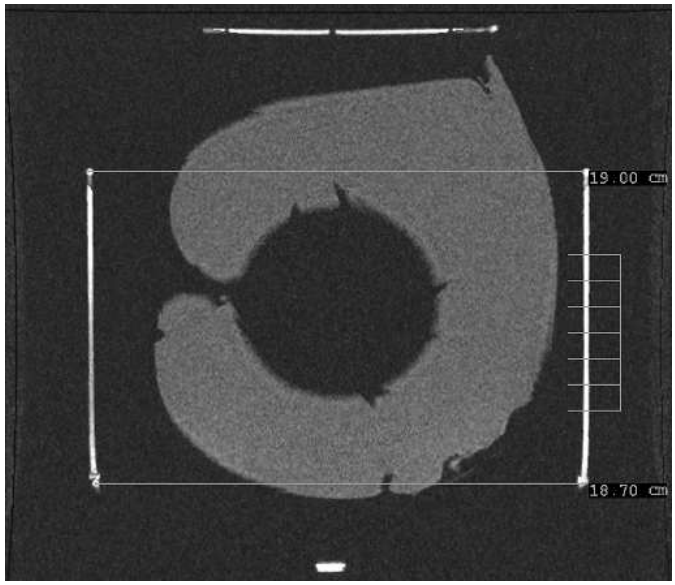


Fig. 7.5: Coronal scan of Lucy phantom and Leksell® MR Indicator Frame, after gradient nonlinearity distortion correction. Observe the slight curvature at the bottom of the fiducial rods, causing the distance of the rods to be inconsistent.

Even though the frame is non-ferromagnetic, the inhomogeneity caused by the frame is very apparent in Figure 7.5. This coronal image has been corrected by the software-based MRI gradient nonlinearity distortion correction method. Unfortunately, the software was unable to resolve this extra distortion, resulting in the curvature at the bottom of the fiducial rods. Even more disconcerting is the fact that measuring the distance between the rods at the bottom of the image results in a significant discrepancy. At the top of the image (the superior end of the frame), the rods

are at the correct distance from each other: 190 mm. The same measurement should be observed at the bottom of the image (the inferior end of the frame), however the curvature results in the rods appearing to be only 187 mm apart. This 3 millimeter discrepancy is quite severe, being that the goal of the distortion correction is to provide submillimeter accuracy in target localization. This 3 millimeter discrepancy is definitely outside that acceptable range.

Discussing the matter with Leksell did not reveal any concrete solutions to the problem. Leksell claims that the Coordinate Frame Model G is non-ferromagnetic, which is indeed true, since the frame was able to be used properly when scanning the Lucy phantom. Additionally, Leksell claims that the frame does not introduce any additional distortions or inhomogeneities in the main gradient field. Observing the image in Figure 7.5 clearly invalidates that claim. The presence of the frame in the scanner *does* create an inhomogeneity in the main gradient field that needs to be resolved separately from the gradient nonlinearity distortion correction.

It is important to keep in mind that ***all*** objects will influence the characteristics of a magnetic field to some degree. Some objects influence the field more than others will. Section 1.6 points out the fact that biological substances in the human body—such as fat, water, and tissue—will even influence the characteristics of the magnetic field. The same holds true for non-ferromagnetic objects, such as the Leksell® Coordinate Frame Model G. Using this frame in CT scans does not introduce any problems, since CT does not make use of magnetic fields for scanning. On the other hand, the frame creates problems in MRI, and these problems need to be addressed properly in order to provide accurate target localization.

7.1.6 Conclusion

The problems caused by using the Leksell® Coordinate Frame Model G in stereotactic localization are very unfortunate. To add to the difficulties, this problem was discovered very late in the course of this project. Therefore, little could be done to properly address the problem. In the future, it may be necessary to use a different system for stereotactic localization. Unfortunately, it is impossible to scan the Lucy phantom without affixing the coordinate frame to the MRI indicator frame, since the coordinate frame secures the Lucy phantom in place within the indicator frame. Perhaps another frame can be utilized in the future, one that is not metallic. It is unknown whether or not such a frame exists. Unfortunately, the only way to proceed from here is to make do with the equipment available, and resolve the gradient field inhomogeneity with other software-based methods. These solutions are out of the scope of this work, in terms of content and feasibility. However, it is important to point out the importance of implementing such methods in order to successfully resolve this issue.

7.2 Success or Failure?

The ultimate success of this software package is defined by many different factors. Creating a straightforward, feasible solution that features the robustness to operate flawlessly in a medical environment is no easy task. The only method to determine the success of the software is to analyze the original requirements and specifications detailed by Loma Linda University Medical Center, and analyze the software to ensure conformance to these standards.

7.2.1 Meeting LLUMC Requirements

This software-based MRI gradient nonlinearity distortion correction was developed in conjunction with Loma Linda University Medical Center, to create a MRI-based system for target localization. As outlined in Section 1.3.2, LLUMC requires that the software be able to correct gradient nonlinearity distortion with submillimeter accuracy, thereby allowing for submillimeter accuracy in target localization. Before the software can be put in service, it is necessary to analyze the performance of the software and verify that it has indeed met the requirements outlined by LLUMC.

The basic requirements of the software were outlined as the following:

1. A three-dimensional correction method must be implemented, that goes one step beyond the scanner filtering techniques. Additionally, the correction method must be able to handle all images that a scanner produces, filtered and unfiltered.

2. The correction method must be specifically designed for MRI to calculate a distortion correction, without fusing CT image data.

3. Dynamic calculation of the gradient field nonlinearity is required to achieve submillimeter accuracy, rather than relying on static specifications and values provided by the scanner manufacturer.

4. Calculation of the gradient field nonlinearity needs to be performed directly, rather than inverting the spherical harmonics expansion.

5. Efficient algorithm design must be the foremost concern in implementing a distortion correction method, to reduce the number of calculations necessary to

compute the result. Recall that the size and number of rounding and truncation errors is directly proportional to the number of calculations performed.

6. The correction must be accurate to less than one millimeter, in order to have the ability of promoting submillimeter accuracy in target localization.

7. The software should produce valid results in a timely fashion.

8. The software should have the flexibility of running on a wide range of computer systems, without requiring special hardware.

Meeting each of these requirements is vital to the success of the software in the real world. Failure to meet any of these may not allow LLUMC to achieve the high precision it seeks in functional proton radiosurgery. Therefore, the true success of the software is determined by analyzing the completed package, checking for conformance with the aforementioned characteristics.

The first five requirements are inherent features of the design. These design aspects are discussed in great detail in Chapters 2 - 5. The final three requirements reflect the actual operation of the completed software. The performance of the software package must meet these requirements in full, or the system will not provide the capability and support LLUMC is requiring.

7.2.2 Accuracy of Distortion Correction

The numerical results presented in Section 6.2 demonstrate the obtainable accuracy of the distortion correction method. Referring to the data tables listed in these sections will reveal that the software was indeed able to provide submillimeter accuracy in

the distortion correction. When considering the three high quality datasets—the 11-6-05, 4-30-06, and 6-19-06 studies—the distortion correction results demonstrated an accuracy of 0.39 to 0.78 mm. This equates to an accuracy of one to two pixels, given that each pixel in the studies equals 0.390625 mm. Theoretically, the highest accuracy obtainable in any image is one pixel, provided that the pixel is the smallest unit of measurement in the image. Observing an accuracy of one to two pixels in the distortion correction is very good indeed.

The data simulations of Section 6.4 also demonstrated the accuracy of the distortion correction method. The simulations demonstrated that the distortion correction algorithms possess an accuracy of 10^{-5} mm, when using a perfect dataset. This accuracy is the upper limit to the accuracy of the system, meaning that the correction method is at best accurate to 10^{-5} mm. This scenario, obviously, will never occur, because this measure was obtained with a perfect dataset: a dataset featuring a perfectly centered, perfectly sized phantom with no noise or artifacts. Even though this accuracy is not obtainable in real-world scenarios, this measure does demonstrate that the errors observed in real datasets are primarily caused by other sources (listed in Section 7.1), not by the calculations performed by the distortion correction software. This increases confidence in the validity and accuracy of the results provided by the software.

The accuracy obtained from the distortion correction was better than one millimeter in all but one of the test cases: the 7-17-05 study. In this study, the accuracy observed was 0.7 to 1.2 mm. The reason for the lower accuracy in this test case was three-fold:

- Only two sequences were used (axial and coronal), instead of all three. The smaller size of the dataset may have contributed to the size of the errors observed.
- The phantom was offcentered by a large amount, creating quite asymmetrical distortion in the data.
- The shim coils were not calibrated prior to scanning, which introduced additional distortions and artifacts in the dataset not accounted for by the distortion correction.

Because of the problems inherent with the 7-17-05 dataset, this scenario should be disregarded from consideration. The accuracy obtained in the distortion correction is a directd reflection of the quality of the original dataset, which is in turn dependent on the quality of the original experimental setup. Since the experimental setup was not as refined in this study as in future studies, the input data was therefore of lower quality, which resulted in lower quality the correction results. Curiously enough, the 10-2-05 study was also plagued by these very problems, yet the accuracy of the results in the distortion correction was still better than one millimeter. Technically, the 10-2-05 study should also be disregarded from consideration because of the problems inherent in the data. However, since the results obtained using this data were still acceptable, it may not be necessary to disregard these results from consideration.

7.2.3 Accuracy of Target Localization

Determining the accuracy of the distortion correction by localizing the target markers in the Lucy phantom provided an independent verification of the accuracy of the software, see Section 6.7. The Odyssey® treatment planning software performed the

stereotactic transformation necessary to convert the coordinates of the markers to the coordinate system defined by the Leksell® Coordinate Frame. After computing the locations of the markers, the calculated locations of the markers were also within one millimeter of the actual measured locations. The markers were localized with an accuracy of between 0.48 mm and 0.99 mm. This is a very significant improvement over the accuracy of target localization using the uncorrected images, which yielded accuracies between 1.07 mm and 3.33 mm. Using the corrected images, the accuracy of target localization was even slightly better than the accuracy obtained from localizing in CT images.

Obtaining submillimeter accuracy in target localization using the distortion-corrected images is a significant achievement, especially considering the fact that the coordinate frame introduced inhomogeneities into the main gradient field. These inhomogeneities added additional distortions in the inferior portions of the images, and made target localization more difficult (see Section 7.1.3 for a more detailed analysis). Regardless of the situation, the markers were localized properly to within one millimeter of their actual locations, and therefore passes the test.

The results obtained from localizing the Lucy markers confirm that the software has met the primary requirement of obtaining submillimeter accuracy. The calculations performed in these tests fully demonstrate the capability of the software to obtain submillimeter accuracy in both the distortion correction and target localization. The results produced by the software can now be used with confidence in real-world scenarios.

7.2.4 Software Performance

Section 6.6 discusses the never-ending struggle of balancing accuracy and performance in software systems. One of the considerations made when developing this software solution was to strike a great balance between speed and accuracy. The accuracy of the distortion correction is already verified, and meets the requirements outlined by LLUMC. The next challenge the software had to overcome was to deliver those accurate results in a timely fashion.

The performance benchmarks show that the software does indeed have the capability of delivering submillimeter accuracy in a relatively short amount of time. Since the operating environment of this software is not time-critical, meaning that calculations are not required after a specified amount of time, there is essentially no time limit for the software to deliver its results. However, requiring the user to wait a ridiculous amount of time is not good practice. Considering the sheer number of complicated operations performed by the software, the fact that the software performs the distortion correction in about 20-30 minutes is very good indeed. Even more interesting is the fact that the software performs quite consistently across the various test platforms, despite the large differences in hardware configurations. The expected result would be that the differences in performance of the distortion correction software would be similar to that of the control program across the various test machines. The control program, SuperPI, demonstrated how the differences in hardware on the test machines influenced the execution time of calculating π to 32 million digits. Observing very consistent software performance results across vastly different hardware configurations is somewhat uncommon in the PC world.

Obviously, there are some factors at work in the software that contribute to this phenomenon. The Matlab programming environment is not a typical development suite, in that the programming syntax is designed with user-friendliness in mind. Matlab is primarily used by engineers and scientists who do not have a formal computer science background, and therefore do not have native programming experience. This by no means degrades the quality of Matlab. The expansive application support provided by Matlab is unsurpassed. The Matlab libraries include full DICOM and image processing support—perfect for this project. Although Matlab contains all of these wonderful features, it is not the most efficient programming environment available today. The performance benchmarks are indicative of this. The fact that the software performs nearly identical across vastly different hardware configurations indicates a design aspect inherent to the Matlab environment. There is some form of computational overhead specific to Matlab programs that is imposed on program execution, regardless of the hardware configuration. This seems to be the only logical explanation for the very consistent performance results observed in the benchmarks.

Another factor that contributes to the performance of the software is the Matlab compiler. The distortion correction software was developed in Matlab, then converted to a standalone Windows executable using the Matlab compiler. The method in which the Matlab compiler creates the standalone executable will definitely influence the execution time of the software. The exact method in which the compiler creates standalone executables is unknown. What is known is that the compiler creates a wrapper written in C/C++, created by the system C/C++ compiler. These two factors definitely contribute to the finished product. The efficiency of the system

C/C++ compiler definitely comes into question at this stage, as well as the quality of the C/C++ wrapper code. By no means is this discussion designed to criticize these methods or processes; the only intention here is to make a point to discuss the Matlab compiler as a probable factor in the performance of the distortion correction software.

The only method in which to determine the efficiency of the program created by Matlab is to program the entire software from scratch strictly in a C/C++ environment, and perform the performance benchmarks again. Unfortunately, performing such a test is out of the question for the following reasons:

1. All of the built-in Matlab library functions will require reprogramming in C/C++. These include all the edge detection algorithms, DICOM image and data support, and matrix support. The time required to *correctly and efficiently* code all of these algorithms is simply too great.

2. The efficiency and correctness of freely-available functions to perform edge detection, DICOM data manipulation, and matrix operations becomes a very serious issue, one that might compromise the performance and accuracy of the software.

3. Matlab features many data visualization tools that are useful for program development, which would simply not be available if the software were to be programmed from scratch in C/C++.

Even though the energy and work required to port the software to C/C++ is definitely great, the result would be worthwhile if the goal is to decrease computation time. With a C/C++ implementation of the distortion correction software, several

benefits would result:

1. The software could be optimized for specific machine platforms, thereby increasing the performance on particular machines.
2. The software would no longer be limited to the Windows operating system, but could be used in theoretically any operating system.
3. The programmer has more freedom to choose the compiler to generate the software.

Given the time, budget, and resources allocated to this project, the straight C/C++ implementation was not possible. The Matlab implementation does indeed serve its purpose, and proves that the software is capable of producing the necessary results in a reasonable amount of time. The idea of porting the software to C/C++ is definitely an avenue of research for the future, which may increase the appeal and feasibility of the software beyond its current state.

7.2.5 Hardware Compatibility

The performance benchmarks were successful in demonstrating two concepts: the computational time required of the software to deliver results, and the compatibility of the software on various computer hardware platforms. The benchmarks were completed using several machines, each with different hardware configurations. The machines range from being relatively out-dated and slow by today's standards, to being one of the most powerful machines available on the current market. In all cases, the software was able to complete its operations and deliver the same numerical results without experiencing any errors. Hardware incompatibility is a serious issue to

consider when developing software. Various hardware platforms can result in inconsistent program execution, which may lead to incorrect computation by the program. The fact that the software has the capability to run correctly and consistently on several different platforms is testament to the quality of the software design.

It is impossible to say at this time what kind of computer system will be running this software. The system can range anywhere from an average desktop system similar to the midrange systems featured in the benchmark tests, to a very powerful multiprocessor server-grade machine. Whatever system is chosen to execute the software, it is highly likely that the software will be able to run correctly and consistently on that system. The benchmarks instill confidence in this claim by demonstrating the software's capability across various systems.

Indeed, the software was only tested using the Windows XP operating system. The ability of the software to run in other environments is unknown at this time, simply because other operating systems were not tested. The Matlab compiler has a series of configurable options that may allow the software to be run in other environments, and therefore ensure compatibility with other hardware configurations. Again, these claims will have to be investigated to determine for certain if these are possible. Considering the results demonstrated here with the performance benchmarks, the likelihood of successfully accommodating these situations is very high.

7.2.6 Scanner Filtering

One of the main concerns when developing this software was the compatibility of the software with images that have been filtered by the MRI scanner. Many physicians

and oncologists tend to prefer applying scanner filtering techniques when acquiring images during an MRI scan. This makes sense, since the filters greatly reduce the amount of inherent distortion in the images. However, these filters are not sufficient in correcting for gradient field nonlinearities, and therefore do not provide the accuracy necessary to perform target localization, as per LLUMC's specifications. The nature of these filters is discussed in Section 1.8.1.

Since the filters used in the experimental setup are not true three-dimensional correction methods, the filters simply reduce the amount of visible distortion present in the images. Using this correction alone is not sufficient in providing the accuracy necessary for target localization. However, the story changes when the scanner filtering techniques are used in conjunction with the distortion correction software.

In Section 6.5, the accuracy of the distortion correction using filtered and unfiltered was analyzed. The numerical and image results show that the distortion correction is able to further improve upon the scanner filtering and provide even more accurate correction results. This result should not be surprising. Reducing the amount of visible distortion in the original dataset will increase the accuracy of the correction method. Since the correction method makes use of linear least squares, the results represent the best reconciliation of the distortion correction model to the acquired distorted data and the theoretical undistorted data. The data acquired using the scanner filters more closely matches the undistorted data than the data acquired without using scanner filters, therefore the least squares fit will provide a more accurate model of the corrected data.

The distortion correction results obtained from the filtered studies demonstrate

the sensitivity of the correction method to phantom centering in the MRI scanner. If the phantom was properly centered, that is, positioned to within one millimeter of the gradient isocenter in x, y, and z, the accuracy of the correction results would improve. The increased errors observed in the correction results using the filtered studies is the direct result of improper phantom centering. In these studies, the phantom was offcentered in z; as a result, the errors observed in z of the correction results were higher than those of x and y. The offset in z most likely influenced the accuracy of the correction in x and y as well, since the method is a three-dimensional correction.

Taking all of these factors into consideration will reveal the true ideal setup to use in the real world: acquire images of a properly centered phantom (positioned to less than one millimeter from gradient isocenter), properly calibrate the shim coils, and filter all images with all scanner filtering techniques enabled. Even though this experimental setup was not obtained during the testing procedures, analyzing the numerical results clearly indicate that providing such a setup will yield the most accurate results. Indeed, accurate results can still be obtained with a slightly-offcentered phantom, and using images without scanner filtering. Submillimeter accuracy is still possible even with these less-than-ideal conditions, to a certain extent. Placing the phantom carelessly in the scanner will yield inaccurate results. The resulting large center offsets such as those observed in the 7-17-05 and 10-2-05 studies will not promote submillimeter accuracy in target localization.

Indeed, careful attention is necessary to achieve this ideal scenario. Again, this refers back directly to the issues of phantom centering and calibrating shim coils,

which are the two greatest contributors to errors observed in the correction results. If the scanner filters were applied, and these two factors were properly addressed, the correction results would be as accurate as can be. The phantom placement would promote symmetry in the gradient nonlinearity distortion inherent in the data, the shim coils would remove extra distortions and artifacts, and the scanner filters would greatly reduce the amount of inherent gradient nonlinearity distortion. These factors combined will yield a high-quality dataset that will produce very accurate results. Therefore, it is highly recommended to follow these guidelines in all future scans. Despite the difficulty of reproducing such a scenario, the high-quality correction results that would be obtained definitely justify the extra work involved.

7.3 Comparison of Other Solutions

Today, there exist several different MRI gradient nonlinearity distortion correction methods. All of these methods experience some degree of success in tackling the problem of gradient nonlinearity distortion. Recall back in Chapter 2 that current published methods do not provide the high level of accuracy LLUMC is requiring for its latest generation of functional proton radiosurgery systems. That is the reason why this software package was developed: to produce a more accurate, more robust system to correct for gradient nonlinearity distortion. The numerical results shown in Section 6.2 demonstrate that the software does indeed provide the accuracy LLUMC requires, and also meets other specified requirements.

One of the most important factors of the software's success is its contribution to this particular research field. The primary difference between this implementation

and others is the difference in the accuracy obtained in the correction results. At this point, the question that must be answered is the following:

How does the accuracy obtained from this system compare to those of other published works?

Comparing the accuracy of the software to that of other published methods will further demonstrate this software's success.

7.3.1 Previous Works

There are many works that provide solutions to gradient nonlinearity distortion in MRI [3, 33, 36, 37, 38, 40, 41]. These works confirm that developing and deploying such a system is indeed feasible. The numerical and image analyses in Sections 6.2 and 6.3 demonstrate this software's effectiveness in handling gradient nonlinearity distortion. Each implementation represents a different method to tackle the distortion problem, in order to provide accurate images for target localization. While the work presented in this thesis discusses the same topic, the implementation is very different.

Scanner Correction

One of the most well-known "correction" methods are those provided by the MRI scanners themselves. General Electric (GE) and Siemens, the two most prevalent manufacturers of MRI scanners, both provide distortion correction methods, which can be activated within the scanner software. The correction takes into account the properties of the magnetic field in each scanner, and implements a number of

correction filters. As discussed previously in Section 1.8.1, these filters do not provide a true three-dimensional correction. This is because the filtering process is based on the location of the slice relative to the gradient coil. Therefore, this method simply filters the image, to *reduce* the amount of visible distortion in the images. Because this method does not consider the data in three dimensions, the pixels in each image possess only a limited degree of geometric accuracy.

The accuracy obtained from applying scanner filters is not suitable for accurate target localization. Localizing target regions in the filtered images will not provide much improvement in geometric accuracy as would localization on unfiltered/uncorrected images. The filters are indeed successful in reducing the amount of visible distortion in the images. To the untrained eye, the filtered images may appear to be "distortion-free." However, the distortion is still present in the images.

One of the tests performed in the results analysis was processing filtered and unfiltered images. After applying the distortion correction on both filtered and unfiltered images, it became apparent that the characteristics of gradient nonlinearity distortion are indeed present in both image sets. The only difference between the filtered and unfiltered images is the severity of the distortion in each. The effects of gradient nonlinearity distortion are more pronounced in the unfiltered images, resulting in the three-dimensional "potato chip" warping. The filtered images, on the other hand, may appear to not have any distortion to the naked eye. But looks are definitely deceiving: the distortion correction method still displaces the voxels in the filtered datasets to their correct theoretical undistorted locations. The magnitude of the displacements of each voxel is definitely much smaller in the filtered datasets than those

in the unfiltered datasets, but they are present nonetheless. This simple observation demonstrates the incapability of the scanner filtering techniques to provide a true three-dimensional correction for gradient nonlinearity distortion.

Manufacturers are currently implementing three-dimensional correction techniques. Unfortunately, none are available currently on the market. Additionally, these filtering techniques are designed specifically for each make and model of scanner, since it takes into account the characteristics of the gradient fields in each. This severely limits the deployment of a three-dimensional distortion correction to only those scanners that feature the new software. Not all scanners will be able to benefit from this technology, further limiting the scale of the deployment of a distortion correction method. To further complicate matters, the distortion correction will require integration into the scanner software, requiring medical facilities to perform software upgrades on the scanner system. It becomes clear that any scanner filtering technique simply will not fit the bill.

Fusing CT and MRI

In his work, Sumanaweera [3] presents a method of fusing CT and MR images together, to provide geometric accuracy. This seems logical, and is actually common practice today. Since CT is the gold standard, MR has been primarily used to supplement CT, to provide more complete patient information. This implementation is slightly backwards, so to speak, in that CT is used to supplement MRI. Historically, MRI was secondary to CT. Using CT and MRI together provides successful results since CT is not subject to distortions as MRI. Combining the two together provides

a method that utilizes the best properties of each modality.

When properly configured, implementing such a system will provide submillimeter accuracy in the correction results, and therefore target localization. Indeed, the results do meet the requirements LLUMC specified. However, this system requires much more work and resources in order to obtain this level of accuracy. This system will require the patient to be scanned twice: once by CT, and once by MRI. This greatly complicates all aspects of the the treatment process, most notably the treatment planning process. Implementing a system of target localization that is based solely on MRI will greatly reduce the complexity of the treatment process, and will help promote the ease and success of treatment.

A second consideration to keep in mind is the age of the work. Being that the work was published well over a decade prior, medical technologies have changed greatly over that period of time. Indeed, the method is useful and successful in accounting for gradient nonlinearity distortion, but on MRI scanners of that particular era. What is important to keep in mind is the infancy of MRI technologies in the early 1990's. Many of the issues needing to be dealt with at that time, as well as some of the shortcomings of MRI, are nearly nonexistent today. Advances in technology have successfully eliminated most of the initial problems with MRI. Technological advances have also improved the quality of MR images, which promotes the accuracy of target localization. At that time, the obtainable resolution on MRI was half of that obtainable on CT (256×256 pixels, vs 512×512 pixels), limiting the amount of information obtainable from each image. Today, MR image resolution is on par with CT. Perhaps the greatest change introduced with MRI advancements is an increase

in severity of gradient nonlinearity distortion. Since today's scanners utilize wider bores and stronger gradient fields than scanners of old, the problem is even greater today than in the past.

The distortion correction method developed for this work successfully meets or exceeds the accuracy obtained from a system based on the fusing of CT and MRI, without relying on CT image data, thereby making this system more robust and much simpler to implement, deploy, and maintain.

Using Pre-Defined Specifications and Parameters

The distortion correction method proposed by researchers at the NMR facility at Massachusetts General Hospital, Harvard (MGH) [36] is based on knowledge of the characteristics of the magnetic gradient coils in the scanner. This is very similar to the method used by the scanner filtering techniques. Their premise was that if the properties of the gradient coils are known, then the distortion caused by the coils could be calculated directly. This is known as one of the two universally accepted correction methods [36, 33, 37, 38, 39].

The method used to calculate the distortion correction is very straightforward and direct, not requiring a great deal of complex calculations. Successful implementation of the procedure is accomplished by obtaining the manufacturer's specifications of the magnetic gradient coils. Included with these specifications is an expansion of the spherical harmonics model (see Section 1.7 and Equation 1.28). This information is then used to geometrically transform each image to remove the effects of the distortion. The results presented do indeed reflect the feasibility of such an implementation.

Submillimeter target localization was accomplished using this system. The accuracies published are very similar to those observed in the numerical analyses of this work.

Even though this implementation is successful in providing submillimeter accuracy in target localization, there exist many limitations to this system. This is because the entire correction method is based solely on one source: the manufacturer specifications. This proves to be a very significant shortcoming in a distortion correction method, due to the following reasons:

- The spherical harmonics expansion provided by the manufacturer is based on calculations performed on one machine. Each machine will vary slightly from the machine tested by the manufacturer. Therefore, each scanner produced will differ from the manufacturer's specifications.

- Performing regular maintenance on the scanner will slightly change the characteristics of the gradient coils, thereby increasing the inaccuracy of the factory specifications. Indeed, the magnetic field is very stable over time, but every change in the system will produce small variations in the field characteristics.

- The correction requires the information to be acquired from the manufacturer, increasing the complexity and difficulty of performing such a method.

Regardless of the amount of work required to obtain the factory specifications (which may not be much of an issue), the fact that the correction method is based on *static* values opens the door to calculation errors. The characteristics of the gradient coils simply do not remain perfectly constant over time. This is the sole reason for the very expensive annual maintenance costs required to properly deploy MRI: to

ensure that the scanner is *as close* to factory specifications as possible. No machine will ever match the manufacturer's specifications *exactly*. The specifications may be loose enough such that the errors introduced in the correction method do not allow for submillimeter accuracy in target localization.

Even though the method is successful in its implementation, there is not enough confidence in terms of being able to achieve high geometric accuracy in localizing targets across various MRI scanners. The methods employ here seem eerily similar to those employed by the scanner filtering techniques, in that the correction is based on static, predefined measurements. The distortion correction method described in this thesis overcomes these limitations by providing a dynamically-calculated distortion correction model that is based on the *current* condition of the gradient field. If the characteristics of the gradient field change, even slightly, the distortion correction software will not have a problem calculating another distortion correction model to fit those changes. In other words, the distortion correction software will adapt itself to the characteristics of the gradient field of *any* MRI scanner, at *any* time. This correction method is machine-independent, therefore it can be deployed across various systems without requiring manufacturer specifications for the gradient coils. The distortion correction method described in this thesis obtains similar accuracy to the MGH implementation, but represents a much more robust, flexible implementation that does not require detailed manufacturer specifications concerning each MRI scanner prior to execution.

Calculating Distortion Correction Indirectly

The distortion correction method that stands out amongst the rest in this research topic area is that of Langlois, et al [33]. The method proposed by Langlois was perhaps the first work that discussed a more direct approach to correcting gradient nonlinearity distortion. Today, this work is widely recognized as the primary source for this subject, and is referenced by other works [36].

To perform the correction, Langlois followed the lead of several other researchers, using a truncated spherical harmonics model to mathematically model the distortion. Calculating the coefficients of the model in this method provides a representation of the distortion inherent in the images. The correction method is then accomplished by inverting this model, to obtain an "undistortion" model, or a mathematical expression used to remove distortion from the images.

The accuracy of the correction produced by this system is 1-3 millimeters. For this very reason, this implementation is not suitable for LLUMC. Simply put: this implementation does not provide the accuracy sought after by LLUMC. Submillimeter accuracy is clearly not possible using this system. The factors that contribute to this lower accuracy are numerous. Although this method seems simple and straightforward, it is not without great limitations. There are many dangers associated with inverting large polynomial functions; the spherical harmonics model is definitely one of them. Even though the model used is only a truncated version (the full mathematical expression is an infinite sequence of terms), the complexity of the expression is still quite high. Inverting *any* polynomial function, no matter how complex or large, will always result in calculation errors. This is not so much a product of the algo-

rithm used (although the algorithm design does in fact contribute to the size of the errors produced); it is a simple inherent trait of computer design. In mathematical theory, a true inversion of numbers, matrices, or expressions is always perfect, in that the inverse provides a true opposite of the original input. However, computer science would say otherwise. In order to perform a full inverse, an infinite amount of storage space is required. Unfortunately, today's computers do not have an infinite amount of storage space at their disposal. This limitation leads to one often-disregarded fact: numbers cannot be stored with infinite precision, because the machine is not infinite. Therefore, every number that is stored must be truncated or rounded, in order to fit into a specified storage space. This finite precision is the inherent problem with all modern computers. ALL complex calculations are subject to rounding and truncation errors. Therefore, the goal in algorithm design is not just to implement an algorithm, but to provide the *most accurate* method of implementation, one that minimizes the errors produced.

In terms of the correction method proposed by Langlois, the inversion of the spherical harmonics model is subject to significant rounding and truncation errors. A general rule of thumb to follow is: the more calculations performed, the more the errors grow, since more values are being passed, stored, and truncated. Inverting a complex polynomial function involves many, *many* calculations. Therefore, the inversion is also prone to many, *many* rounding and truncation errors. Indeed, the individual errors themselves are quite small. However, over time, as more calculations are performed, the size of the total error can grow quite quickly. The *error propagation* can get out of control, leading to an inaccurate result, and therefore an improper

distortion correction.

By no means is the distortion correction method described in this thesis free from rounding and truncation errors. In fact, no distortion correction method (or, any complex computer algorithm, for that matter) will be immune to these types of errors. However, this does not mean these errors should be taken lightly. The distortion correction software developed for this thesis takes extra care to preserve numerical accuracy by avoiding cumbersome calculations such as these. In order to achieve high numerical accuracy in any application, it is vital to properly address the issue of error propagation. The algorithms implemented in this software were designed in such a way as to *reduce* the amount of error produced during the calculation process. The high accuracy obtained from the distortion correction method and target localization described in this thesis is primarily due to the careful attention to numerical accuracies. The primary method used to achieve this goal is using linear least squares to calculate the distortion correction parameters. Calculating the least squares result is accomplished using QR factorization, which is a very robust method that preserves numerical accuracies (see Appendix B). By reducing the number of calculations performed in the distortion correction method, the accuracy of the final result was greatly enhanced, allowing the software to meet LLUMC requirements by providing submillimeter accuracy. Many opportunities for rounding and truncation errors were avoided simply by avoiding the inversion of complex polynomial functions, and calculating the distortion correction model directly. This greatly improved numerical accuracy, yielded higher quality calculations, and increased the efficiency of each algorithm.

Conclusion

When comparing the distortion correction method developed for this thesis to current implementations, the quality of this method becomes apparent. Some current methods simply do not provide enough geometric accuracy, and therefore will not be able to support accurate localization of increasingly small anatomical targets of less than one centimeter in size. This software meets LLUMC's requirements not only in terms of achieving a high level of accuracy, but also in terms of simplicity and robustness. Even though there are existing systems that can provide submillimeter accuracy in target localization, the feasibility and complexity of such systems limits the effectiveness of these techniques. Developing a system based solely on MRI greatly reduces the complexity of the treatment procedure by not requiring additional CT image data. Providing a solution that is machine-independent, and not requiring manufacturer specifications greatly increases the range of deployment of such a system at medical institutions. Although there are several other correction methods currently available, none of these are sufficient for the needs specified by LLUMC. So indeed, the distortion correction method described in this work does contribute to this widely discussed research topic, and does forward knowledge in this area.

7.4 Suggestions for Improvement

The requirements set forth by LLUMC are quite stringent. In order to properly deploy their new system for functional proton radiosurgery, they needed a distortion correction method that can provide submillimeter accuracy in target localization. The analyses and discussions presented in Chapters 6 and 7 demonstrate that the

software is successful in meeting these requirements. There should be little doubt of the software's performance when it is integrated into the final system.

Even though the software accomplishes all of the original design goals, it is far from perfect. In a similar fashion, the results that it provides are far from perfect. There are several areas that can be improved, in order to further refine the software technique and increase overall accuracy. With more research, time, and resources, there should be little problem in implementing these minor changes.

7.4.1 Phantom Centering Techniques

The numerical and image results presented in Sections 6.2 and 6.3 clearly demonstrated the software's sensitivity to the manner in which the oil phantom was placed in the scanner. The primary assumption that the software employs while calculating the distortion correction model is that the gradient nonlinearity distortion inherent to the dataset is symmetric. To promote this property in the data, it is necessary to position the phantom as close as possible to the gradient isocenter of the MRI scanner. Obviously, it is impossible to position the phantom *perfectly* with respect to isocenter. Therefore, a reasonable goal is to place the phantom to within one millimeter of isocenter in all three dimensions, since the level of accuracy desired is also less than one millimeter.

The experimental trials performed for this work demonstrate the high difficulty in achieving proper phantom centering. Even after performing several modifications to both the phantom and the experimental setup, it was still not possible to achieve consistent centering of the phantom to within one millimeter of isocenter. As discussed

in Section 7.1.1, the following modifications were implemented:

- Shortened the phantom foot to better fit the headcoil.
- Utilized precision plastic shims to fine-tune phantom placement.
- Etched reference marks on the phantom to use in conjunction with the laser sight on the MRI scanner.

Even with these modifications, only one of the five studies performed was able to achieve ideal phantom placement. The primary frustration in performing all of the studies was the inability to check phantom placement prior to acquiring several sequences. Therefore, placing the phantom properly in the scanner involved some degree of guesswork. There was no absolute level of confidence that the phantom was placed properly in the scanner. The only way to check the placement of the phantom was to perform a full sequence, and then check the measurements afterwards. Unfortunately, this was the only surefire method of verifying phantom positioning. The feeling of frustration and disappointment of once again failing to achieve proper phantom centering is quite understandable when considering each sequence takes almost 20 minutes to perform.

Rather than perform full sequences and waiting for a long period of time to verify phantom centering, a new idea came about: etch reference marks on the phantom to use with the scanner laser sight. The etched marks on the phantom were designed to fine-tune phantom placement by using a generic reference system. However, this still did not seem to improve the situation. The 6-19-06 study was performed on a phantom that was centered using these reference marks. The study achieved submillimeter

centering in x and y, but not in z. At this point, it would be unwise to rule out the usefulness of the reference marks, since only one study was performed thus far with these marks. Further testing is necessary to determine the true usefulness of using this system for phantom placement.

The issue of phantom centering has always existed as a challenging problem to overcome. Unfortunately, this problems remains unsolved. Even though strong efforts were made to solve this problem, they have all remained unsuccessful, since a system for consistent phantom placement was not fully developed. Again, the small number of trials performed with the refinements to the experimental setup do not provide enough information to draw educated conclusions. In order to develop a consistent experimental setup, more experimental trials are necessary. This includes testing the modifications to the phantom, as well as practicing the setup procedures with the MRI technician. There are so many variables that come into play concerning phantom placement that it is difficult to control all of them with only a small number of experimental trials.

A consistent method of phantom centering is vital to the success of this software. Even though the numerical and image results show that the software still produces valid results with less-than-ideal phantom centering, the accuracy of the correction is indeed compromised as a result. The greater the phantom offset, the greater the chance of sacrificing submillimeter accuracy in target localization. Indeed, there are other factors that contribute to the accuracy of the correction result (see Section 7.1 for more details), but none have as great of an influence as phantom centering. Developing a refined, consistent method of phantom centering will promote consistency

in the distortion correction results, and increase confidence in the capability of the system.

7.4.2 MRI Scanner Configuration

Besides phantom centering, there are other issues that must be properly addressed in the experimental setup. The manner in which the MRI scanner is configured will definitely determine the quality of the images acquired. Improperly configuring the basic parameters of the MRI scanner will result in low quality data, thereby compromising the accuracy of the distortion correction method. Such parameters as signal weighting, sampling methods, and image filters will all affect the appearance and quality of the acquired images.

The distortion correction software works best with a high quality dataset in order to calculate a distortion correction model with submillimeter accuracy. In order to achieve this high level of accuracy, extra steps must be taken to properly configure the MRI scanner. These steps may deviate slightly from normal practice, and the extra procedures may be unfamiliar to the MRI technician. The needs of this project are slightly different than those of routine MRI scans, and therefore require a specific set of parameters to acquire the data properly.

Perhaps the most important issue to address is proper calibration of the shim coils. This topic has been discussed many times in previous sections, so it should be very apparent by now that this issue should not be taken lightly. Because there are so many steps required to achieve the proper experimental setup, it is very simple to forget to perform small steps such as shim coil calibration. All of the parameters of

the scan are configured using the MRI scanner software. Due to the vast number of configurable parameters, overlooking one or two small details is quite easy to do. Take, for example once again, shim coil calibration. Three of the five trials performed for this project were completed without any knowledge of shim coil calibration. The 11-6-05 study indicated problems with the experimental setup in the form of extra distortions in the datasets. Thankfully, the problem was resolved, and the quality of the data was preserved in later studies.

The most difficult problem to overcome during the experimentation stage was the fact that the studies were performed using two different MRI technicians. The 7-17-05, 10-2-05, and 11-6-05 studies were performed with a seasoned MRI technician that was quite familiar with the scanner software suite. Unfortunately, the MRI technician left shortly after the 11-6-05 study, and a new MRI technician took the position. The replacement technician was unfamiliar with this project, and was therefore unfamiliar with the setup procedures. Familiarizing the technician with the experimental procedures of this project took some time and experimentation. Although the change in technicians added difficulty to the experimental process, the proper procedures were eventually carried out to obtain useful data.

In order to properly execute this project in the future, it is necessary to have clear understanding of the configurable parameters on the MRI scanner. This responsibility falls down on both the researcher and the MRI technician. The researcher must fully understand the setup procedures and exactly how each parameter will affect the acquired images. Obtaining desirable results in the images also requires coordination between the researcher and the MRI technician. Before scanning a volume, the MRI

technician must also understand the implications of configuring the myriad of options in the scanner software. And most importantly: good communication between the researcher and the MRI technician is necessary to properly carry out the experimental procedures and obtain desired results.

Due to the relatively short timespan and limited budget of this project, it was not possible to spend more time experimenting with the MRI scanner, to learn more about how the configurable options will affect the appearance and quality of the images. If more time were allotted, then more knowledge and experience could be gained, to further improve the results obtained from this distortion correction. The change in MRI technician only complicated matters, since there was an adjustment period necessary between the researchers and the technician. Fortunately, the desired data was acquired and all the requirements were satisfied. But, learning more about MRI scanner configuration will greatly enhance the experimental procedure, the quality of the acquired images, and therefore the accuracy of the distortion correction method.

7.4.3 Image Quality

The issue of image quality is closely related to the configuration of the MRI scanner. The quality of acquired images determines the accuracy of both the gradient nonlinearity distortion correction and target localization. Poor quality images will greatly degrade the ability of the distortion correction software to model the gradient field nonlinearity. The data that is used to calculate the distortion correction model is derived only from the image of the oil phantom. If the images of the oil phantom are inaccurate, the distortion correction model that is calculated will also be inaccurate.

In the same manner, the accuracy of target localization is also based on the quality of the images. Poor image quality will make target localization difficult. Determining the proper location of a desired region is nearly impossible in noisy images. Localizing the markers in the Lucy phantom in Section 6.7 proved to be very difficult in the axial study, but was much easier in the coronal study. The markers appeared rectangular and lopsided in the axial study, greatly increasing the difficulty in accurately defining the center of the markers. On the other hand, the markers appeared very sharp and clear in the coronal study, thereby promoting ease in accurate target localization.

The importance of image quality in all future studies cannot be expressed enough. Proper configuration of both the MRI scanner and the experimental setup are crucial in obtaining accurate results, both in distortion correction and target localization. Submillimeter accuracy cannot be expected from poor quality images. Providing high quality images for use in distortion correction and target localization will help promote submillimeter accuracy in both processes.

7.4.4 Programming Languages/Environments

The choice of programming environment definitely played a role in the development of this software. Theoretically speaking, this software could have been programmed in any language or in any environment. The implementations cited in this thesis made use of different programming languages to produce their distortion correction methods. The researchers at the NMR facility at Massachusetts General Hospital, Harvard (MGH) developed their software using Matlab [36], while Langlois, et al programmed their method in C [33]. Given the time constraints and resource available

for this project, Matlab seemed like the logical choice for software development. The excellent support for DICOM data formats increased the speed at which the software could be developed. Matlab's extensive imaging toolkit library eliminated the need to implement edge detection and other image processing methods. The native support for matrices greatly increased the ease of software development by allowing for simple, effective data storage. Without the tools made available by Matlab, development of this software package may not have been possible in the short time allotted.

Indeed, the functionality provided by Matlab is great, and the flexibility of the environment allowed the software to meet all of the software design goals. However, Matlab is not the ultimate solution. The performance benchmarks of Section 6.6 revealed that the "compiled" software is subject to some degree of computational overhead, due to the fact that Matlab is a scripting language. Because Matlab provides so much capability for a wide range of applications in a very simple programming environment, the programs it produces will not exhibit as high of a computational performance level as other implementations, particularly those programmed using compiled languages. The performance benchmarks in Section 6.6 clearly demonstrate the slower execution of Matlab due to it being a scripting language. Here, a basic program implemented in Matlab was shown to be over 150 times slower than the same program implemented in C++.

Matlab is also known for its high numerical accuracy in computations, which also lends to its slower computational speed. Recall back to Sections 6.2 and 6.6 that speed and accuracy are generally inversely proportional to each other. It is impossible to have an infinite amount of both quantities. Favoring one factor will cause the other

to suffer. Such is the case with Matlab: in order to achieve the high accuracy and simplicity in software design, it is apparent that Matlab considers accuracy much more important than speed. This is by no means a bad decision: the accuracy of the distortion correction software and the ease of software development are testaments to Matlab's unique design.

This issue may be relatively minor, and may not concern most people. The future is uncertain, and requirements may change over time. If it is determined that the software needs to perform the distortion correction in less time, then this issue must be addressed properly. Increasing the computational performance of the software while maintaining numerical accuracy will not be a simple task. This does not mean, however, that this goal is impossible to reach. Further research must be conducted to determine the ideal path to take, and to determine the exact requirements for the improvements.

7.4.5 Time

All of the suggestions for improvement listed above seem to center around one key aspect: *time.* Not only were the requirements specified by LLUMC quite stringent, but the time period in which these deadlines needed to be met added even more pressure. This project was funded by the Henry L. Guenther foundation, which provided additional funding for further research in the other aspects of the functional proton radiosurgery project. By the time this project started, the timeline for completion was only one year. Considering the relative infancy of the experimental procedures, one year is a very short time indeed to develop new experimental procedures, develop

the software distortion correction method from scratch, and verify the correctness and accuracy of the software. All of these goals were met within the time period, and the software has demonstrated its success. However, providing more time to devote to the project will definitely improve the quality of the distortion correction, and therefore increase the accuracy of the method.

The knowledge gained from performing these experimental procedures is indeed priceless. With each experimental trial, more knowledge was acquired which help improve the quality of the data acquired at the next trial. Compare the quality of the first and last scans performed: the 7-17-05 study only included axial and coronal sequences, featured a phantom that could not be centered properly in the headcoil, and did not take advantage of proper shim coil calibration. As a result, the quality of the images acquired from this study were the worst out of the five studies, and the accuracy of the distortion correction based on this study was the lowest of the group. In contrast, the 6-19-06 study included axial, coronal, and sagittal sequences (thereby maximizing the amount of data available for distortion correction modeling), featured a phantom that was centered much closer to the gradient isocenter, and took advantage of proper shim coil calibration. The three studies in between featured small refinements to the experimental procedure, which resulted in increasingly higher quality results each time. The same pattern would hold true if more scans were performed at this stage. Even though the most accurate results were obtained from the 6-19-06, the study is far from being perfect. There are improvements than can still be made to the experimental setup, but this requires more time to properly implement and test.

Even though the software-based MRI gradient nonlinearity distortion correction method was successful in achieving its goals, the project will exist as an ongoing work. Clearly, there is still room for improvement. The only way that these improvements can be realized and fully implemented is with more time. Time is truly of the essence, especially when considering this project. Allocating and devoting more time to this project will result in an increase in knowledge and experience, which can be directly applied to the development of this project, to further enhance the performance of the software in all aspects.

7.5 Future Work

Research in magnetic resonance imaging is ongoing. Many advances have been made in the field over the past ten years, and more advances will be realized in the future. Because the field is so fast-paced and ever-changing, the software-based MRI gradient nonlinearity distortion correction must have the ability to keep up with the changing demands of the field. Even though the software was able to meet the requirements specified by LLUMC, the project is not completed. As pointed out in Section 7.4, there is room for improvement in several areas. This does not mean the software is of low quality. There is no perfect solution in existence today, and no perfect solution will ever exist in the future. However, there are several areas where the software can be modified, in order to produce a more perfect system, and improve upon the results gained here.

7.5.1 Increasing User-Friendliness

The final destination for the software-based MRI gradient nonlinearity distortion correction is unknown at this time. The correction may become part of a treatment planning system, and the user can apply the distortion correction to images at the treatment planning stage. The software may exist as a standalone process that corrects images directly after scanning, to provide DICOM images that natively contain the correction. The correction may even be used as part of a proprietary software suite that controls aspects of the functional proton radiosurgery system. Depending on the application, the user-friendliness of the software may come into play.

The current implementation of the software-based MRI gradient nonlinearity distortion correction exists as primarily a console application, with some visual program elements. The user is presented with a graphical folder browser at program startup, to locate the images to be used for distortion correction modeling. The software provides data visualization support by displaying two- and three-dimensional plots of the calculated data. Other than these interactive features, the software does not feature a full-fledged graphical user interface. Theoretically, a graphical user interface is not necessary, but the majority of end users would prefer a graphical user interface over a command line interface, no matter the application.

The manner in which the software is used in the future will determine whether or not the current implementation of the software provides the correct level of user friendliness. If the software is to be integrated into another application, the software will not need additional graphical user interfaces to control its operation. The parent application would be the provider of the interfaces, since it is manipulating the dis-

tortion correction method in the background. If, on the other hand, the software is to be used as a standalone application, then user friendliness may become an issue. Depending on the type of users that will operate the software (most specifically, how proficient the users are with computers), modifications may be required to increase user friendliness of the software. The user may not want to type in commands at a command prompt, but may rather click buttons and traverse menus to achieve the same result. It all depends on the final destination of the software.

If the decision is made to increase the user friendliness of the software, there are a few key areas where graphical user interfaces will increase the user friendliness of the software. The first aspect is the configuration of the software. In its current iteration, the software adjusts its computational behavior based on a group of predefined settings. The configuration of the software is performed manually using a text file. In order to verify or change the software settings, the user must locate the settings file, open the file in an appropriate text editor, and modify the desired values. Handling this procedure using a graphical user interface will greatly enhance the ease of configuring the software for use. Many of the configuration settings specified in the text file are boolean values. The controls for these options would be quite easy to implement in a graphical user interface, in the form of radio buttons or check boxes.

The second key area that may benefit from a graphical user interface is the actual execution of the software. In its current implementation, invocation of the software occurs at a command line prompt. In Windows, this requires the user to open up a command prompt window, locate the directory that contains the software, and invoke the software with the appropriate command line request. A graphical user

interface could be used here to automate this process. Rather than forcing the user to manipulate a command prompt to invoke the software, a clickable shortcut and graphical window could be used. Invoking the software through the command prompt in Windows XP may seem very old-fashioned to today's users that are familiar with the point-and-click functionality of the operating system. Extending this feel and look to the distortion correction software may increase the ease at which users start up the program to perform distortion correction.

The final area that may benefit is user notifications. Since the software requires at least 20 minutes to perform distortion correction, the user is left with little notification on the progress of the software. Implementing progress bars and timers to indicate to the user how much time is required to complete the software operations may help make the software more visually appealing. In its current implementation, the software only gives simple text messages in the command prompt to notify the user of its progress. The user has the option to turn these options off, only to leave a relatively blank command prompt. Impatient or inexperienced users may have no idea of the time required of the software to complete its tasks, and therefore may assume the software has crashed because it has not finished executing in a short amount of time. Providing progress bars and timers would inform users that the software is indeed working properly, but requires a certain amount of time to execute.

The current design of the software promotes user friendliness to a certain extent. The software only requires the user to setup the configuration file with the desired values, and then prompts the user to locate the source image files. No further interaction from the user is necessary. Therefore, no further effort was made to increase

user friendliness. Depending on the final destination of the software, the needs of the system, and the types of users manipulating the program, user friendliness may become a future issue that might need to be addressed.

7.5.2 Porting to Different Languages

The software-based MRI gradient nonlinearity distortion correction was developed using Matlab. Matlab provides great support for the types of operations performed in the distortion correction method. However, the Matlab environment does not present the highest performance solution available, in terms of computational speed. The performance benchmarks in Section 6.6 indicate machine-independent computational overhead due to Matlab being a scripting language.

In the original specifications set forth by LLUMC, execution time was not a determining factor in the success of the software. Therefore, the required 20 - 30 minutes to calculate the distortion correction result may not be an issue. If, however, more speed is desired in the future, it will be necessary to investigate methods to speed up the software. Porting the software to different programming languages (most specifically, compiled languages) will help increase the speed of the software, while maintaining the high level of accuracy already observed. In a sense, this will allow researchers to "have their cake and eat it too." Optimizing the performance of the software in particular programming languages will require an in-depth, time-consuming effort that will entail an entirely separate project of its own. Many of the operations performed in the distortion correction software are library functions included with Matlab. Therefore, the distortion correction method is not the only code that has to

be ported, but also all the library functions. This includes DICOM image support, edge detection, image processing techniques, and matrix support.

Although porting the distortion correction software to different languages may require a large, concerted effort, the end result will definitely be well worth it. Matlab is a scripting language, which severely limits its computational speed. Recall back to Section 6.6 that the performance difference between a simple program written in Matlab and C++ was over 150×! Porting the distortion correction software to a compiled language such as C/C++ will greatly enhance performance, most likely by at least several factors of ten. This performance increase would result in the distortion correction to be calculated in a matter of seconds, rather than 20-30 minutes, thereby increasing the effectiveness and robustness of the software. Boosting computational speed by this great margin will also enhance the feasibility of the software, in terms of integrating the distortion correction into other software packages.

The size and complexity of the Matlab libraries also hinders its computational speed. By programming only what is required of the software, the computational performance of the distortion correction method may greatly increase. This process is very similar to that of setting up an operating system. By modifying the kernel to include only the necessary hardware and application support for a particular use, the performance of the system greatly increases. By the same token, reprogramming the software to include only the functions that it requires to execute will result in a more compact, streamlined method. Rather than building and including vast code libraries to invoke one or two key functions, programming only the necessary functions from scratch will greatly improve the performance of the software.

Porting the software to different languages will also allow researchers to take advantage of environment- or platform-specific enhancements. Developing all of the program code for the distortion correction method will allow researchers to control every aspect of the program's execution, even down to basic mathematical operations. Depending on the particular type of machine that will execute the distortion correction, there may exist specific commands that take advantage of the computer architecture, thereby unlocking various performance enhancements. The possibilities are essentially endless. If this endeavor is investigated, it is important to take note that the Matlab code developed for this project can serve as a blueprint for future implementations, to aid in future software development in any programming language.

7.5.3 Supporting Unix-Based Operating Systems

The distortion correction software was tested using the Windows XP operating system. Since the Matlab distribution that was used for software development was also designed for Windows, this operating system seemed like the logical choice for testing the performance of the software. Unfortunately, due to time constraints imposed on this project, it was impossible to test the functionality of the software in other operating systems, most notably, Unix-based operating systems.

The performance benchmarks from Section 6.6 demonstrate that the software has the capability to run consistently and correctly across various hardware platforms. The only factor that was not tested was the operating system of those platforms. To keep the tests consistent, all of the machines were configured with Windows XP.

Further testing will be required in order to measure the performance of the software in other operating systems.

Matlab is available for various operating systems, not just Windows. There exist Unix distributions of Matlab which provide the same functionality as the Windows version. It is therefore possible for the program code to be used on a Unix version of Matlab, and compiled to run in a Unix environment. Performing this task would not require a great deal of time or effort, just simple knowledge of the operation of Matlab. If the current implementation of the distortion correction software is deemed suitable for a particular application, then creating a Unix version of the software will not be difficult.

If the software is ported to a different programming language, then the operating system used to execute the software will become a very important issue. Various programming environments provide functionality on particular systems. Likewise, certain environments benefit greatly from certain programming languages. Porting the software to various languages will also allow for support of a wider range of operating systems, thereby increasing the deployment of the software in the medical community.

7.5.4 The Calibration Phantom

The greatest factor that determines the quality of the distortion correction is the calibration phantom. A poorly designed phantom will provide a poor representation with which the software calculates the distortion correction model, thereby creating invalid or inaccurate results. The oil phantom underwent several design modifica-

tions throughout the course of the development of this project. These modifications included:

- Shortened the phantom foot to better fit the headcoil.
- Utilized precision plastic shims to fine-tune phantom placement.
- Etched reference marks on the phantom to use in conjunction with the laser sight on the MRI scanner.

The reasons for performing these seemingly basic and crude modifications were time and budget constraints. Manufacturing a custom-built phantom to fit perfectly within the headcoil of the test MRI scanner exceeded operational budgets. Additionally, the time necessary to create such a phantom would have greatly reduced the amount of data obtained for the project. Therefore, to overcome these problems and shortcomings, basic modifications were made to the existing oil phantom. It was unfortunate, to say the least, that the modifications made to the phantom were not capable of solving the problem of phantom centering.

Further testing would be necessary to refine the experimental procedures prior to each scan. Centering the phantom is a difficult task with the current setup. There exist two possibilities to solve this problem:

1. Continue testing with the current phantom, and make more modifications to the phantom to (hopefully) achieve ideal phantom centering consistently.
2. Manufacture a custom phantom that fits perfectly with the headcoil of the scanner, such that placing the phantom in the scanner allows it to be centered ideally with respect to the gradient isocenter each and every time.

The path to take at this point depends on both the time and budget available to undertake such endeavors. If the budget allotted for this research is large enough to support a custom-made phantom, then this would be the ideal option. Creating a custom phantom will eliminate all of the guesswork involved with placing the phantom in the scanner. No special training will be required to place the phantom in the scanner correctly each time. The possibility of placing the phantom in the scanner incorrectly, such that the phantom is not centered properly with respect to gradient isocenter, would be greatly diminished. If the operational budget for this project would have allowed for such a solution, then a different phantom would have been used, and the results presented here would be much different.

If, unfortunately, there exist time and budget constraints that simply do not allow a custom-made phantom to be produced, then a simple phantom such as the oil-filled cube phantom used in this project will have to be modified extensively. Modifications such as those performed on this phantom will greatly increase the accuracy phantom placement in the scanner. Even though the phantom was still not placed to within one millimeter of gradient isocenter with all of these modifications, this does not mean all hope is lost. With more experience, time, and experimentation, it should be possible to make do with the current tools available, and achieve a dataset based on an ideally-centered phantom. Perhaps additional modifications are necessary in order to achieve consistent phantom centering. Developing a different phantom foot may resolve the issue by limiting the range of movement the phantom experiences in the headcoil. Additional markings etched onto the phantom may help center the phantom more precisely using the laser sight on the scanner. Creating a precisely

measured cradle for the phantom using precision shims may help in achieving ideal phantom placement in the scanner.

The exact solution to this problems cannot be determined at this time. Further research into this topic is necessary. The implications of phantom centering have been discussed at length in this work, in order to provide future works with the information necessary to make informed decisions on how to rectify the situation. With enough resources and knowledge, consistent phantom centering will eventually be a reality rather than a dream.

7.5.5 The Metallic Stereotactic Coordinate Frame

It was unfortunate to experience problems with the stereotactic coordinate frame during the testing and verification stage of this work. Section 7.1.3 discusses the inhomogeneities created by the presence of the Leksell® Coordinate Frame Model G in the scanner. Even more disconcerting was discovering that the manufacturer's claim was false, and that the frame does indeed introduce inhomogeneities in the images. The markers in the Lucy phantom were still able to be accurately located, despite the additional inhomogeneity caused by the coordinate frame. Luckily, the markers were positioned several centimeters superior of the coordinate frame, and therefore existed outside of the range of the inhomogeneity. Small steps were taken in order to "salvage" the images for use in target localization, as discussed in Section 6.7. However, this problems needs to be properly addressed in order to realize the full potential of MRI-based stereotactic localization.

There are two possible solutions to this potentially serious problem. The first

is to investigate the usage of other stereotactic reference systems. Due to the time constraints imposed on this project, locating other stereotactic reference systems was not possible. It is unknown at this time if other MRI stereotactic reference systems exist on the market. If there are indeed other commercial products available, testing the effectiveness of these products could greatly enhance the accuracy of target localization in corrected images. Most specifically, if a non-metallic coordinate frame was available, the inhomogeneities observed in the images of the Lucy phantom would no longer exist, thereby solving the problems experienced during stereotactic localization for this project.

The second solution is more complex, but would handle a host of other potential problems. The inhomogeneities caused by the coordinate frame are no different than those created by other metallic objects in the scanner. Scanning patients with metallic objects in and on their bodies is a regular occurrence. There will always be patients who have metallic objects such as tooth fillings, belt buckles, buttons, rings, cardiac pace makers, and implants. The presence of these objects in the scanner will also introduce inhomogeneities in the images that must be resolved separately from gradient field nonlinearities. These point sources of distortion require sophisticated detection and correction techniques. Modeling these sources of distortion, as well as implementing a software-based correction, will require a great deal of research and knowledge. This step is necessary in order to deploy an accurate MRI-based stereotactic localization system.

Undertaking this task will definitely be necessary in the future. The software developed for this thesis only provides correction for one source of MRI distortion.

Other software methods will be required to handle the other various sources of distortion (described in brief in Section 1.6), including inhomogeneities caused by metallic objects in the scanner. Until these sources of distortion are handled, the accuracy of target localization on scans including metallic objects is limited.

7.5.6 Further Testing and Verification

Even though the software-based MRI gradient nonlinearity distortion correction has been shown to meet the requirements set forth by LLUMC, the method still must go through many stages of rigorous testing and verification before it will ever be used on human beings. The most important obstacle to overcome is obtaining approval from the Food and Drug Administration (FDA), which needs to verify that the system is indeed safe for use on humans. In order to reach this critical stage, the distortion correction software must pass a vast number of rigorous tests. Among these tests include the localization of markers in various phantoms, and the treatment of anatomical regions on animal subjects.

Applying the distortion correction method to other phantoms will ensure the accuracy of the method is consistent across different experimental setups. There exist numerous phantoms of various shapes and sizes, and each has provisions for performing different tests in a range of experimental conditions. Previous works have made use of different phantoms to prove the effectiveness of their techniques [3, 36, 33]. Performing target localization using various phantoms will serve to demonstrate the effectiveness of the software, as well as identify areas that need improvement.

In addition to testing the distortion correction method using various phantoms,

LLUMC currently has plans to test the accuracy of both the distortion correction method and the proton delivery system by delivering doses to animal subjects. LLUMC is making the final preparations to perform functional proton radiosurgery on rats. The experiments will target the pituitary glands, which are only a few millimeters in size. Recall that the goal LLUMC set forth was to create a system for functional proton radiosurgery that could accurately treat volumes less than 1.0 cm^3; therefore the rat pituitary gland provides an ideal target for testing. If the experiments prove to be successful, such that the glands were treated effectively without damaging other tissues, then the tests clearly demonstrate the capability of both the proton delivery system and the distortion correction software.

After the successful completion of these (and other) tests, then LLUMC can consult the FDA for approval in treating human subjects. Even though the distortion correction method has demonstrated its effectiveness in the tests performed in this work, these tests are very preliminary. Providing more evidence that the system works as advertised is necessary in order to obtain FDA approval, and therefore begin treatment on humans.

7.6 Final Remarks

Nearly every aspect of the software-based MRI gradient nonlinearity distortion correction has been discussed and presented. This thesis has discussed several different topics, from details concerning the historical developments in the field, to the requirements of the software set forth by LLUMC, to the actual software development process, to finally the results obtained from the distortion correction method. One

of the goals of this thesis, and the accompanying program code, is to serve as a basis of knowledge for future endeavors. A great deal of knowledge was gained from performing the various experimental trials. This knowledge will aid future researchers in achieving accurate distortion correction results for use in many medical applications, including stereotactic radiosurgery.

The accuracy and efficiency of the software-based MRI gradient nonlinearity distortion correction has been demonstrated. Observing the accuracy of the distortion correction using the oil phantom, the software correction yields an accuracy of 0.39 to 0.78 mm. In terms of localizing the targeting markers in the Lucy phantom, the distortion correction allows the markers to be localized with an accuracy of between 0.48 mm and 0.99 mm. The software is successful in meeting the initial requirements set forth by LLUMC. With further testing and verification, the software will one day be worthy of FDA approval for use on humans. Only then will LLUMC be able to commission the system for use in its proton treatment facility. Until that time, more work is necessary in order to ensure the ultimate success of the distortion correction software.

It is important to keep in mind that the software developed in this thesis is far from perfect. Several key areas of improvement have been identified. If these aspects are properly handled, the accuracy, effectiveness, and efficiency of the software will improve. With these modifications, the quality of the distortion correction will also improve, thereby providing a more accurate distortion correction to support higher quality results in medical applications.

Another key aspect to keep in mind is the scope of this project. This software

distortion correction method was designed to only handle gradient nonlinearity distortion. There exist many other sources of distortion, caused by various factors during the MRI scanning process. In order to obtain submillimeter accuracy from an MRI-based stereotactic localization system used on humans, it is necessary to correct for the other sources of distortion in MRI. The work described in this thesis eliminates one of these sources of distortion, and brings the system one step closer to achieving its ultimate goal.

Knowledge and research in MRI is constantly evolving. New technologies will become available over the next several years that will change the face of the medical imaging community. MRI has experienced a great surge in uses and applications over the past five to ten years, and will continue to experience this trend in the future. The software-based MRI gradient nonlinearity distortion correction described in this work provides the flexibility to handle future endeavors in this field. The use of the software-based gradient nonlinearity distortion correction is not strictly limited to functional radiosurgery. Theoretically, *any* medical techniques that make use of MRI can greatly benefit from a robust gradient nonlinearity distortion correction, to provide more detailed, more accurate image information. With the appropriate modifications and updates, the software will be able to adapt to the particular needs and requirements of various medical applications. The lessons learned—and the knowledge acquired—through this thesis will no doubt aid future research in this field, enabling future scientists and engineers to effectively deploy MRI-based patient treatment systems.

APPENDIX

A. ERROR ANALYSIS

Performing error analyses is a very crucial step in determining the quality and accuracy of the software-based gradient nonlinearity distortion correction. The primary goal specified by LLUMC is to obtain submillimeter accuracy in target localization using MRI. This also implies that the gradient nonlinearity distortion correction technique must also possess submillimeter accuracy. Expressing the accuracy of the numerical results is accomplished with several formulae.

Calculating the accuracy of mathematical computations is a straightforward process. For this example, let x be an n-vector of "true" measurements and $\hat{x}$ be a vector of estimates of x. Therefore, $\tilde{x}$ is the *error* in the estimates, corresponding to the differences between x and $\hat{x}$. In the numerical analyses in Chapter 6, $\hat{x}$ is also referred to as the *total residuals*.

Calculating the *mean squared error (MSE)* will yield the "expected" value of the square of the errors:

$$MSE \;=\; \frac{\|\tilde{x}\|_2^2}{n}, \tag{A.1}$$

The MSE is also referred to as the *variance* in Chapter 6.

Taking the square root of the MSE will yield the *root mean squared error,* or simply

the *RMS error:*

$$RMS = \sqrt{MSE} \tag{A.2}$$

The RMS error is commonly referred to as the *standard deviation.* The same holds true for the data analyses in this work.

The accuracy of the gradient nonlinearity distortion correction is expressed in terms of the RMS error (standard deviation). Assuming that the data points follow a Gaussian (normal) distribution, extending out to three standard deviations (RMS $\times$ 3) will include 99.5% of the data points. From this, it can be concluded that 99.5% of the data points will be less than three times the RMS.

B. LEAST SQUARES

B.1 Introduction

Calculating the coefficients of the spherical harmonics expansion for the distortion correction method requires a unique problem solving technique in mathematics. The goal is to determine a set of numbers that best approximates a much larger group of numbers. Representing the gradient nonlinearity distortion as a polynomial expansion effectively creates a simplified expression of the dataset, rather than using some very complex function. Indeed, the approximation is not exact, but the representation created does still retain the same characteristics of the original dataset. Viewing this problem in terms of linear algebra, it is simple to view this process as projecting a vector from a higher-dimensional space onto some lower-dimensional space. This process is known as *least squares,* and is a very popular method widely used in scientific computing.

There are several different types of least squares problems, each with their own characteristics and appropriate applications. The first, and most simple, type is *square least squares,* which involves a square linear system: there exist the same number of unknowns as equations. Here, a unique solution always exists, as long as the matrix is nonsingular. The second type of problems are known as *nonlinear least squares* problems, and are more complicated to compute. Here, the model function

f is nonlinear in the components of $\boldsymbol{x}$, therefore this problem represents a special case of nonlinear optimization. The third, and final, type of least squares problem is *linear least squares,* in which the model function f is linear in the components of $\boldsymbol{x}$, therefore this situation exists as a linear optimization problem. Calculating the coefficients of the spherical harmonics expansion for this distortion correction method involves a linear least squares problem.

B.2 Characteristics of Linear Least Squares

There are several characteristics of linear least squares that makes it the ideal method of calculation for the distortion correction method. Recall back to Chapters 4 and 5 the properties of the problems that make use of linear least squares. The primary characteristics of the plane fitting and distortion correction calculations was the fact that there exist more equations than unknowns. In fact, there exist *several thousand times* as many equations than unknowns. This type of system is unlike a square linear system, in which the number of unknowns and equations are the same. Since there are more equations than unknowns, a unique solution does not exist for the system. Do not panic! This does not mean the problem cannot be solved. In this situation, being that there are so many equations, the presumption is that the given data contains noise or irrelevant points, therefore an exact fit to the data set is not desired. [4] Therefore, the solution to the problem is not an exact fit to the dataset; the extra data points can be thought of as extra measurements that will help smooth out the noise inherent to the data set. This configuration represents an *overdetermined system,* in which there exist more equations than unknowns. Expressing this system

in matrix-vector notation yields Equation B.1:

$$\boldsymbol{Ax = b,} \tag{B.1}$$

where $\boldsymbol{A}$ is an $m \times n$ matrix with $m > n$, $\boldsymbol{b}$ is a vector of size m, and $\mathbf{x}$ is a vector of size n. However, recall that an exact solution to this overdetermined system does not exist in the usual sense. Since there are only n parameters in the vector $\boldsymbol{x}$, it is not possible to reproduce vector $\boldsymbol{b}$ containing m elements as a linear combination of the n columns of $\boldsymbol{A}$**.** Rather than calculating the exact solution to the system, as the "=" symbol in Equation B.1 would imply, the goal is therefore to minimize the distance between the left and right sides. Expressing this mathematically, the goal is to minimize the Euclidian norm (also known as the 2-norm) of the *residual vector* $\boldsymbol{r = b - Ax}$ as a function of $\boldsymbol{x}$. Recall that the Euclidian norm of x is expressed by:

$$||x||_2 = \sum_{i=1}^{n} \sqrt{(x_i)^2} \tag{B.2}$$

Theoretically, any norm can be used here, but the 2-norm offers many advantages over other norms:

- The 2-norm possesses a strong mathematical relationship with the inner product and orthogonality.
- The 2-norm is very smooth and possesses strict convexity.
- Calculating the 2-norm is quite simple and straightforward, thereby simplifying the problem.

The name *least squares* is derived from the use of the 2-norm to calculate the result. Solving the system yields a vector $\boldsymbol{x}$ that minimize the sum of square differences between the components of the left and right sides of the linear system.

Because an exact equality does not exist for this problem, the expression in Equation B.1 does not best reflect the characteristics of the system. Therefore, the linear least squares problem needs to be rewritten as Equation B.3:

$$\boldsymbol{A}\boldsymbol{x} \cong \boldsymbol{b} \tag{B.3}$$

Here, the approximation is in the 2-norm, or least squares, sense.

B.3 Characteristics of Distortion Correction Problem

Now that the properties of linear least squares are understood, it is time to analyze the characteristics of the MRI gradient nonlinearity distortion correction problem. Understanding the requirements and specifications of the distortion correction problem will reveal the reasons that linear least squares was used to calculate the solutions to the problems. During the distortion correction process, the method of linear least squares is used in two primary steps: plane fitting and distortion correction modeling.

B.3.1 Overdetermined Systems

The most unique property of the plane fitting and distortion correction modeling steps is the size of the system. Both systems are extremely overdetermined. During midplane calculation, there exist over 32,000 data points for each midplane. The plane equation only contains three unknowns! Clearly, an exact solution to this

system is impossible, being that there exist over 10,000 times as many equations than unknowns. The situation is very similar when the distortion correction model is calculated. Here, over 64,000 data points are used to calculate the five terms of the spherical harmonics expansion. Again, there exist over 10,000 times as many equations than unknowns. And again, an exact solution to this system does not exist.

There are two methods that can be used to solve both of these problems. The first would be to reduce the problem down to a square system, such that there exist the same number of equations as unknowns. The implications of such a maneuver should be quite apparent. Doing so would be foolish at best. Throwing away so much valuable data is downright inane. To further complicate matters, determining the data to be kept and thrown away would prove to be a very daunting task. And who is to say that the data sampling best reflects the actual model? Being that the measurements are prone to errors, basing the final result on only a small number of points will prove to be very unreliable. Therefore, the only other logical solution would be to use linear least squares to provide the best approximation of the system in the 2-norm sense. Using this method, the large amount of data points will serve to provide a more accurate approximation of the result, smoothing out the errors to obtain more accurate values for the parameters. The least squares approximation reduces the tens of thousands of observations that lie in a correspondingly high-dimensional space, down to either the three- or five-dimensional parameter space of the plane fitting or distortion correction calculations.

B.3.2 Data Fitting

In both plane calculation and distortion correction calculation, the goal is to essentially fit an expression to the data set. This fit represents the best approximation to the data set in the 2-norm sense. Fitting these expressions to the data sets is a very wise move indeed. Being that the data points are acquired from images produced by the MRI scanner, the data will always be susceptible to noise and other small errors. Simply put: the acquired data will never be perfect. These errors are impossible to avoid, due to the numerical inaccuracies involved with sampling a signal and converting that data to a digital image.

Does this mean that the data should be discarded because of these errors? Definitely not! Linear least squares provides the capability to handle these errors. It is here that the benefits of the overdetermined system are realized. Since there exist a number of data points that exhibit noise or random error, sampling a larger number of points will provide a better representation of the actual system. MRI is a mature system, and the methods employed to acquire images are very accurate. Therefore, the number of data points in the images that will be subject to noise or error are reasonably small. By sampling a larger number of data points, the negative effects that these noisy data points will have on the accuracy of the final result is greatly reduced. Fitting the expression to the overdetermined data set will best reconcile these errors, such that the final result is a suitable representation of the actual system.

B.3.3 Linear System

Probably the nicest property of both the plane calculation and distortion correction calculation steps is the fact that both of these problems are linear systems. Modeling the solutions to these systems is therefore very straightforward using linear least squares. Equation B.3 demonstrates the simplicity of the expression. There are three key components to the system. The first is the matrix $\boldsymbol{A}$, which represents the independent variables. Indicator variables (ones and zeros) indicate group membership with each set of observations. This matrix is known as the *design matrix.* In the plane calculation stage, $\boldsymbol{A}$ is a $n \times 3$ matrix where n typically exceeds 32,000. In the distortion correction stage, $\boldsymbol{A}$ is a $n \times 5$ matrix where n typically exceeds 64,000. The second component of the linear least squares system is the vector $\boldsymbol{b}$, which contains all of the measured data points, or observations, from the sample. This vector is called the *observation vector.* In the plane calculation stage, $\boldsymbol{b}$ contains more than 32,000 elements, while this vector contains more than 64,000 elements in the distortion correction stage. The solution to the system is represented by the vector $\boldsymbol{x}$. The three elements of $\boldsymbol{x}$ in the plane calculation stage represent the corresponding plane coefficients of the particular midplane. The five elements of $\boldsymbol{x}$ in the distortion correction stage represent the corresponding terms of the spherical harmonics expansion.

B.4 Process of Least Squares

Since the plane calculation and distortion correction stages feature highly overdetermined systems, least squares provides a very unique method to determine a solution to the system. The least squares process is a minimization problem, to reduce the

problem from a high-dimensional space down to a three- or five-dimensional parameter space. Due to the complexity of the inherent processes behind least squares, only the basics of these concepts will be covered.

Essentially, the goal of least squares is to transform the overdetermined problem expressed by Equation B.3 into a triangular linear system. This process is the method in which least squares reduces the high-dimensional space of the observations down to the smaller dimension of the parameter space. The conversion method occurs over a series of steps, in which the problem is reduced into successively easier ones having the same solution. Since the system is highly overdetermined, the problem exists as a rectangular system in its original state. Through successive steps, the problem is eventually reduced to a square system, in which the number of equations is equal to the number of unknowns. Even at this simple stage, the problem can be further reduced to yield a triangular system. This process involves introducing zeros successively into the design matrix $\boldsymbol{A}$, to convert $\boldsymbol{A}$ into an upper triangular matrix. The result is a much simpler problem that still preserves the least squares solution of the original system.

Reducing the matrix $\boldsymbol{A}$ to upper triangular form is accomplished by minimizing the squared Euclidian norm (2-norm) of the residual vector $\boldsymbol{r = b - Ax}$. In other words, the process minimizes the residual errors of the differences between the vector $\boldsymbol{b}$ and the product of matrix $\boldsymbol{A}$ and vector $\boldsymbol{x}$. Creating an upper triangular system makes the system quite trivial to solve, since all matrix elements below the diagonal in an upper triangular matrix are zero. The overdetermined ($m > n$) least squares problem containing an upper triangular matrix can be expressed as:

$$\begin{bmatrix} \boldsymbol{R} \\ \boldsymbol{O} \end{bmatrix} \boldsymbol{x} \cong \begin{bmatrix} c_1 \\ \\ c_2 \end{bmatrix} \tag{B.4}$$

where $\boldsymbol{R}$ is an $n \times n$ upper triangular matrix, and vector $\boldsymbol{c}$ is partitioned to match the left-hand-side of the expression. Expressing the residual in terms of Equation B.2, the least squares residual is then described by:

$$||r||_2^2 = ||c_1 - \boldsymbol{R}\boldsymbol{x}||_2^2 + ||c_2||_2^2. \tag{B.5}$$

In the sum, the second term, $||c_2||_2^2$ is independent of $\mathbf{x}$, therefore there is no control over this term. But, the first term is dependent on $\boldsymbol{x}$, can can therefore be forced to zero by choosing $\boldsymbol{x}$ to satisfy the triangular system:

$$\boldsymbol{R}\boldsymbol{x} = c_1 \tag{B.6}$$

The solution of $\boldsymbol{x}$ is the least squares solution. In this expression, $\boldsymbol{x}$ can be found by back-substitution. Therefore, the minimum sum of squares is:

$$||\boldsymbol{R}||_2^2 = ||c_2||_2^2. \tag{B.7}$$

B.4.1 Orthogonal Transformation

Converting the least squares system to upper triangular form is made possible by *orthogonal transformations*. Using orthogonal transformations allows the original least squares solution to be preserved as the matrix $\boldsymbol{A}$ is converted from a rectangular, to a square, to an upper triangular matrix. The method of orthogonal transformation

that is employed is known as the *QR factorization.* For an $m \times n$ matrix $\boldsymbol{A}$, where $m > n$, the QR factorization takes the form:

$$\boldsymbol{A} = \boldsymbol{Q} \begin{bmatrix} \boldsymbol{R} \\ \boldsymbol{O} \end{bmatrix}, \tag{B.8}$$

where $\boldsymbol{Q}$ is a square orthogonal matrix of size m and $\boldsymbol{R}$ is a square upper triangular matrix of size n. The reason that this factorization preserves the original least squares solution when converting the least squares problem (described by Equation B.3) into a triangular least squares problem is because of the series of transformations the linear least squares problems undergoes. These steps can be described by the following:

$$||\boldsymbol{r}||_2^2 = ||\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}||_2^2 = ||\boldsymbol{b} - \boldsymbol{Q} \begin{bmatrix} \boldsymbol{R} \\ \boldsymbol{O} \end{bmatrix} \boldsymbol{x}||_2^2 = ||Q^T \ \boldsymbol{b} - \begin{bmatrix} \boldsymbol{R} \\ \boldsymbol{O} \end{bmatrix} \boldsymbol{x}||_2^2 = ||c_1 - \boldsymbol{R}\boldsymbol{x}||_2^2 + ||c_2||_2^2. \tag{B.9}$$

In Equation B.9, the transformed right-hand side of the expression is given by:

$$Q^T \ \boldsymbol{b} = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} \tag{B.10}$$

Equation B.10 is partitioned such that c_1 is an n-vector. Solving for x in this system satisfies the $n \times n$ triangular linear system $Rx = c_1$. The residual norm is then expressed by $||\boldsymbol{r}||_2 = ||c_2||_2$.

QR factorization is the method employed by Matlab to solve overdetermined least squares problems. Therefore, expressing the solution to a linear least squares problem in Matlab is as simple as computing:

$$x = A\backslash b.$$

Depending on the characteristics of the least squares problem, Matlab will employ various QR factorization methods to compute the final result. This makes the computational process nearly invisible to the user, and ensures that the optimal solution method is used.

B.4.2 Implementing Orthogonal Transformation

There exist several common methods of orthogonal transformation. Each method is similar to LU factorization using Gaussian elimination, where zeros are introduced into the matrix $\boldsymbol{A}$ successively to transform $\boldsymbol{A}$ into upper trianglular form [4]. To preserve the Euclidian norm, orthogonal transformation are used instead of elementary elimination matrices. Three of the most commonly used methods include:

- Householder transformations
- Givens transformations
- Gram-Schmidt orthogonalization

Matlab employs Householder transformations, therefore this discussion will only present this method.

Householder Transformations: Vectors

Householder transformation are the most popular form of orthogonal transformations. Usually, these transformations are the most effective approach to performing orthogonal transformations. By performing the orthogonal transformation, the goal is to

annihilate desired components of a given vector [4]. The Householder method uses *elementary reflectors* to accomplish this task. The elementary reflectors are matrices that take on the form:

$$\boldsymbol{H} = \boldsymbol{I} - 2\frac{vv^T}{v^Tv}, \tag{B.11}$$

where $\boldsymbol{v}$ is a nonzero vector. From this definition, it becomes clear that $\boldsymbol{H}$ is both orthogonal and symmetric, such that:

$$\mathbf{H} = H^T = H^{-1}.$$

Now, consider a vector $\boldsymbol{a}$ which is to undergo orthogonal transformation by annihilating every component except the first. In order to accomplish this, choose a vector $\boldsymbol{v}$ such that:

$$\boldsymbol{Ha} = \begin{bmatrix} \alpha \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \alpha \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \alpha e_1 \tag{B.12}$$

Substituting the definition of $\boldsymbol{H}$ from Equation B.11 into Equation B.12 yields:

$$\alpha e_1 = \boldsymbol{Ha} = \left(I - 2\frac{vv^T}{v^Tv} \right) \boldsymbol{a} = \boldsymbol{a} - 2v\frac{v^Ta}{v^Tv}. \tag{B.13}$$

Therefore, the value of $\boldsymbol{v}$ can be derived from Equation B.13:

$$\boldsymbol{v} = (\boldsymbol{a} - \alpha e_1)\frac{v^Tv}{2v^Ta}. \tag{B.14}$$

Since the final factor is a scalar, this value is irrelevant in determining $\boldsymbol{v}$ because it simply divides out in the formula for $\boldsymbol{H}$ in Equation B.11.[4] Therefore, the final expression for $\boldsymbol{v}$ is:

$$\boldsymbol{v} = \boldsymbol{a} - \alpha e_1, \alpha = \pm||a||_2 \tag{B.15}$$

The α term in each of these equations is used to preserve the Euclidian norm of the result. Here, the sign of α is chosen properly to avoid cancellation.

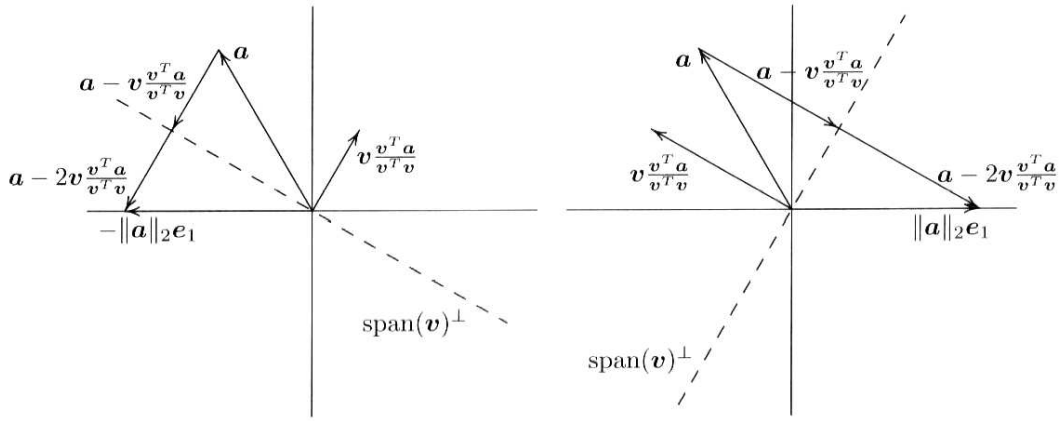


Fig. B.1: Graphical representation of Householder transformations, demonstrating the transformation as reflections [4].

A graphical representation of these processes is demonstrated in Figure B.1, to provide a more visual reference of the concepts presented in Equations B.11 - B.15. In Figure B.1, the vector $\boldsymbol{a}$ is transformed onto the first coordinate axis, making its other components become zero. Then, $\boldsymbol{a}$ is reflected across either of the two hyperplanes (dashed lines) that bisect the respective angles between $\boldsymbol{a}$ and the first coordinate axis. This reflection is performed in order to preserve the norm, and the hyperplane that is chosen determines the sign of the α component described in

Equation B.15. The hyperplanes are given by:

$$span(\boldsymbol{v})^{\perp} = \{x : v^T x = 0\}. \tag{B.16}$$

Here, $\boldsymbol{v}$ is a nonzero vector that is parallel to $\boldsymbol{a} - \alpha e_1$, where $\alpha = \pm||a||_2$, depending on the hyperplane chosen. The orthogonal projector onto span($\boldsymbol{v}$) is given by:

$$\boldsymbol{P} = \boldsymbol{v}(\boldsymbol{v}^T\boldsymbol{v})^{-1}\boldsymbol{v}^T = \frac{\boldsymbol{v}\boldsymbol{v}^T}{\boldsymbol{v}^T\boldsymbol{v}}, \tag{B.17}$$

and the projector onto $span(\mathbf{v})^{\perp}$ is simply given by $\boldsymbol{I}$ - $\boldsymbol{P}$. Therefore, the projection of $\boldsymbol{a}$ onto the hyperplane is given by:

$$\boldsymbol{(I - P)a = a - v}\frac{\boldsymbol{v}^T\boldsymbol{a}}{\boldsymbol{v}^T\boldsymbol{v}}. \tag{B.18}$$

However, to get to the first coordinate axis, it is necessary to project twice as far [4]:

$$a - 2v\frac{vv^T}{v^Tv}. \tag{B.19}$$

Technically, choosing either hyperplane will work; however, the limitations of finite precision arithmetic on computers makes it necessary to choose the hyperplane that yields the point on the first coordinate axis farther away from $\boldsymbol{a}$ [4]. In terms of the α component, this step requires the proper sign for α to be chosen. Failure to properly perform this procedure will result in cancellation in computing the vector $\boldsymbol{v}$. Therefore, considering the graphical example in Figure B.1, the graph on the right should be chosen, meaning that $\alpha = ||a||_2$, because a_1 is negative.

Householder Transformations: Matrices

Expanding upon the concepts of performing Householder transformations on vectors, it is possible to perform the same operations to successively annihilate the elements in a matrix. Consider a m-vector $\boldsymbol{a}$ with the following partitioning:

$$\boldsymbol{a} = \begin{bmatrix} a_1 \\ \\ a_2 \end{bmatrix}, \tag{B.20}$$

where a_1 is a $(k-1)$-vector, $1 \leqslant k < m$. Now, consider the Householder vector $\boldsymbol{v}$:

$$\boldsymbol{v} = \begin{bmatrix} \boldsymbol{o} \\ \\ a_2 \end{bmatrix} - \alpha e_k, \tag{B.21}$$

where α = *-sign(a_k)*$||a_2||_2$. This Householder transformation will then be able to annihilate the last $m-k$ components of $\boldsymbol{a}$. By creating a sequence of Householder transformations defined in the same manner as in Equation B.21, it is then possible to annihilate all the subdiagonal elements of an $m \times n$ matrix $\boldsymbol{A}$ [4]. The process occurs column by column from left to right, eventually reducing the matrix into upper triangular form. With each successive Householder transformation performed on the matrix $\boldsymbol{A}$, the remaining unreduced portion of the matrix undergoes transformation. These steps do not affect the prior columns already reduced, thereby preserving the zeros through each successive transformation.

Applying the Householder transformation $\boldsymbol{H}$ to a vector $\boldsymbol{p}$ yields the following:

$$\boldsymbol{Hp} = \left(I - 2\frac{vv^T}{v^T v} \right) p = p - \left(2\frac{v^T p}{v^T v} \right) v. \tag{B.22}$$

The advantage of computing the transformation in this method is the savings in computational complexity. This method is much simpler to compute than a general matrix-vector multiplication, since it only requires knowledge of the vector $\boldsymbol{v}$. Additionally, no explicit formation of the matrix $\boldsymbol{H}$ is required [4].

B.5 Computer Algorithm

The process for performing QR factorization on matrices using Householder transformations can be summarized by the following computer algorithm [4]:

```
for k = 1 to n      // loop over columns

    // compute Householder vector for current column
    alpha[k] = -sign(a[k][k]) *
        sqrt((a[k][k])^2 + ... + (a[m][k])^2);

    v[k] = [0 ... 0 a[k][k] ... a[m][k]^T - (alpha[k] * e[k]);
    beta[k] = (v[k])^T * v[k];

    // skip current column if it's already zero
    if beta[k] = 0 then
        continue with next k

    // apply transformation to remaining submatrix
    for j = k to n
```

```
            gamma[j] = (v[k])^T * a[j];
            a[j] = a[j] - (2 * gamma[j] / beta[k]) * v[k];
        end;
    end;
```

The important step to perform in this algorithm is the central check:

```
    // skip current column if it's already zero
    if beta[k] = 0 then
        continue with next k
```

If this test returns true, such that at index k $\beta_k = 0$, then the subdiagonal entries that are to be annihilated are already zero. Therefore the algorithm can ignore these entries and move onto the next column. Adding this check allows the QR factorization to complete successfully. This algorithm produces a factorization of the form:

$$H_n \dots H_1 A = \begin{bmatrix} \boldsymbol{R} \\ \boldsymbol{O} \end{bmatrix}, \tag{B.23}$$

where $\boldsymbol{R}$ is upper triangular. Each successive Householder transformation produces an orthogonal matrix, indicated by $H_n \dots H_1$ in Equation B.23. Therefore, taking:

$$Q^T = H_n \dots H_1. \tag{B.24}$$

then

$$\boldsymbol{A} = \boldsymbol{Q} \begin{bmatrix} \boldsymbol{R} \\ \boldsymbol{O} \end{bmatrix}. \tag{B.25}$$

The result shown in Equation B.25 indeed demonstrates that using this form will compute the QR factorization of the matrix $\boldsymbol{A}$ to solve the linear least squares problem. In order to ensure that the solution is properly preserved, it is necessary to transform the right-hand-side vector $\boldsymbol{b}$ using the same transformation [4]. Therefore, the triangular least squares problem becomes:

$$\begin{bmatrix} \boldsymbol{R} \\ \boldsymbol{O} \end{bmatrix} \boldsymbol{x} \cong Q^T b = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}. \tag{B.26}$$

When this computer algorithm is executed, it is not necessary for the product of $\boldsymbol{Q}$ to be explicitly formed. The Matlab implementation, as well as other versions, of QR factorization store $\boldsymbol{R}$ in the upper triangle of the array originally containing $\boldsymbol{A}$ [4]. In the meantime, the nonzero portions of the Householder vectors v_k used for each Householder transformation are stored in the lower triangular portion of $\boldsymbol{A}$. Keep in mind that during the successive Householder transformations, the portion of $\boldsymbol{A}$ that is used to store the vectors v_k are now zero, and are not considered by the algorithm. Using this process conserves storage space and helps increase the efficiency of the algorithm. In some cases, an additional vector of size n is required for storage, because each Householder vector has one more nonzero component than the subdiagonal portion of $\boldsymbol{A}$, therefore the vector will not fit neatly inside this section of $\boldsymbol{A}$ [4].

Because of this implementation, the only components necessary to solve the least squares problem are the vectors v_k. It is for this reason that Householder transformations are much more computationally efficient than performing explicit matrix-vector multiplication. If at some point $\boldsymbol{Q}$ is needed explicitly, calculating $\boldsymbol{Q}$ is accomplished

simply by multiplying each Householder transformation in sequences times a matrix by the identity matrix $\boldsymbol{I}$ [4].

B.6 Conclusion

The software-based MRI gradient nonlinearity distortion correction is a very complex method, requiring the manipulation of tens of thousands of data points. Two key processes are executed during the distortion correction process: plane fitting, and distortion correctio modeling. In plane fitting, a plane equation is fit to the data set. When calculating the distortion correction model, an approximation of the spherical harmonics expression is fit to the data. Calculating an exact solution to this system is impossible, being that the system is grossly overdetermined, meaning that the number of equations greatly outnumbers the number of unknowns. Analyzing the problem further reveals that both of these data models are linear, meaning that both processes make use of linear expressions. Under these conditions, it becomes clear that the goal is to calculate the coefficients of the expression that best fits the mathematical model to the data set. The only accurate method of calculating the coefficients of a plane equation and the distortion correction model is to use linear least squares. The robust characteristics of linear least squares makes these complex calculations possible.

Programming the software-based MRI gradient nonlinearity distortion correction in the Matlab environment allows the backslash "\" operator to be used to calculate the linear least squares solution. The Matlab implementation uses Householder transformations to implement QR factorization, which is proven to be a very accurate, highly efficient method. The algorithm design utilized by Matlab gives confidence

in the calculation results, and provides reassurance that the method of linear least squares used to implement the software-based MRI gradient nonlinearity distortion correction is nearly as accurate and efficient as can possibly be.

C. STEREOTACTIC TRANSFORMATION

C.1 Stereotactic Frame

In order to properly administer treatment to a patient, the target region must be accurately identified. Localization of anatomical targets is made possible with the use of a *stereotactic frame,* which provides an external coordinate reference system that discretizes anatomical regions inside the patient. The frame that is used is called the *Leksell® MR Indicator frame* (named after Lars Leksell, inventor of the Gamma Knife technique, see Section 1.3.). The MR Indicator frame is used in conjunction with the Leksell® Coordinate frame to define stereotactic coordinates. These are shown in Figures C.1 and C.2, respectively.

The stereotactic transformation required to properly localize targets is based on the coordinate system defined by the Leksell® Coordinate frame, Figure C.2. The frame is engraved with a rectilinear coordinate scale, in which the origin $(X, Y, Z) = (0, 0, 0)$ is located superior, lateral, and posterior to the frame on the patient's right side. The coordinates are graduated in millimeters and conforms with the coordinate system used in CT and MR scanning (the DICOM coordinate system, Figure 3.1).

The Leksell® MR Indicator frame is used to perform target localization in stereotactic radiosurgery. The MR Indicator frame is used with the Leksell® Coordinate frame, which provides spatial reference for surgery using x-ray, CT, or MR image

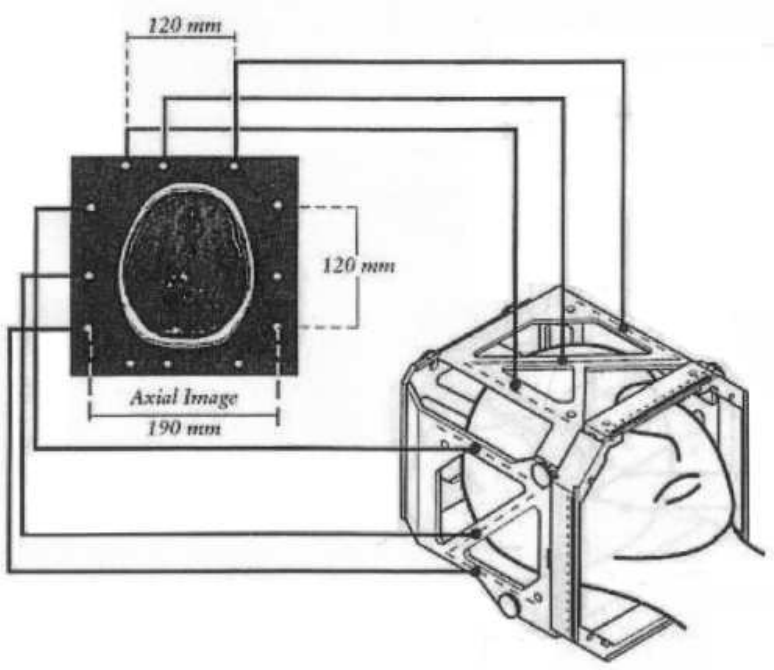


Fig. C.1: Leksell® MR Indicator frame.

data. The MR Indicator frame imposes fiducials on MR images of the patient, which are then used to determine target coordinates within the images. Additionally, the fiducials are used for image slice alignment when the image data is manipulated by treatment planning systems. To ensure that the fiducial markers are visible in the MR images, the frame consists of Z-shaped channels filled with copper sulfate ($CuSO_4$) solution. There are four sets of fiducials imposed on each MR image. The distances and locations of these fiducials are known, and show the relative position of the patient's skull in stereotactic space. The position of the center fiducial in each image, comprising of the diagonal segment of the "Z", indicates the position of that particular slice in stereotactic space.

Localizing target regions using this system requires a *stereotactic transformation* in order to properly define target coordinates. Once these coordinates are calculated,

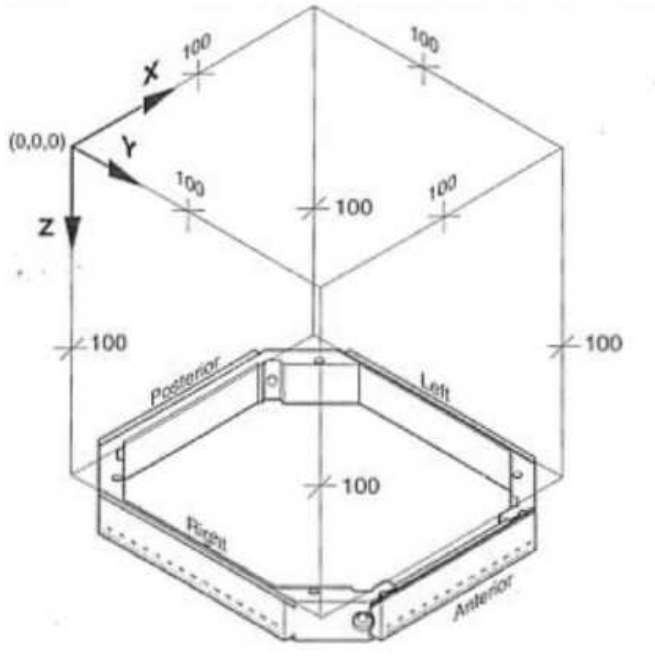
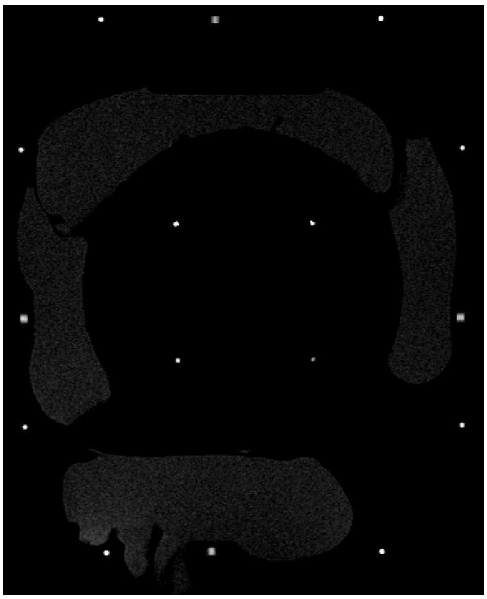


Fig. C.2: Leksell® Coordinate Frame Model G.

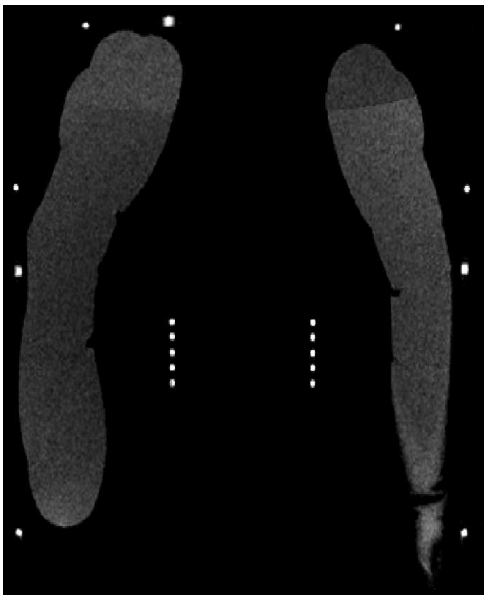
accurate dose delivery is then possible.

C.2 Stereotactic Transformation: Background

In order to properly (and consistently) localize anatomical targets to deliver a dose, two tools are required: a fiducial reference frame, and a stereotactic transformation. The *stereotactic transformation* is a mathematical transformation that converts image coordinates (the coordinate system defined by the MRI scanner) to the stereotactic reference system (SRS). The SRS is defined by the Leksell® Coordinate frame, Figure C.2. The transformation is somewhat complex, given the fact that the axes of the two different coordinate systems are not always parallel with respect to each other. Therefore, in order to properly execute the transformation, mapping coordinates be-



(a) Axial scan.



(b) Coronal scan.

Fig. C.3: Example scans of Lucy phantom ((a) and (b)). Observe the three dots produced by the stereotactic frame, on the outer edges of the images. The third dot indicates the position of the slice, relative to the frame.

tween each system requires both translations and rotations. In order to calculate the expressions that describe the conversion between the coordinate systems, at least three linearly independent points are required.

C.3 Mathematical Basis

The stereotactic transformation presented here is based on the method described in [47]. Before describing the mathematical expressions used to perform the transformation, a brief overview must be provided. The following conventions are used:

- *superscript g:* Global coordinates of SRS

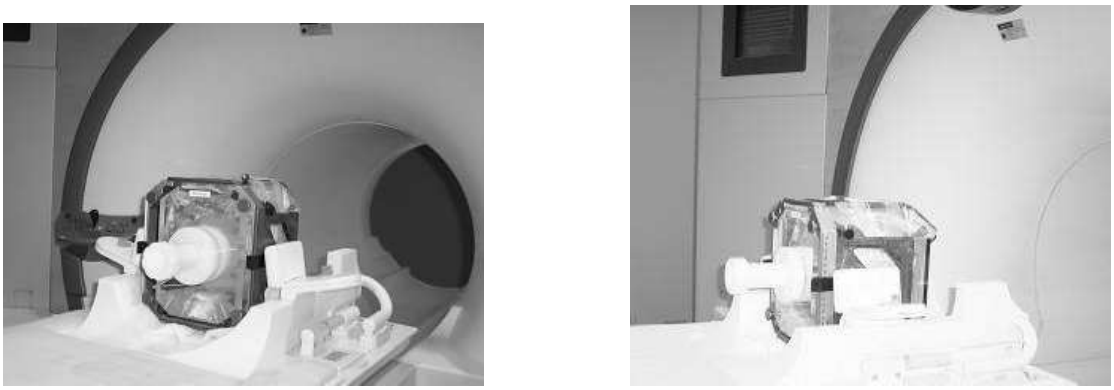


Fig. C.4: Lucy phantom affixed to stereotactic frame, resting in head coil in MRI scanner.

- *superscript l:* Local coordinates of imaging system
- $p_{1,i}^{(l)}$, $p_{2,i}^{(l)}$, $p_{2,i}^{(l)}$: Locations of three distinct markers in local coordinate system (i = 1-3).
- $p_{1,i}^{(g)}$, $p_{2,i}^{(g)}$, $p_{3,i}^{(g)}$: Locations of three distinct markers in global coordinate system (i = 1-3).
- Lower-case bold letters denote vectors, and upper-case bold letters denote matrices.
- All coordinates systems referenced are right-handed coordinate systems.

C.3.1 Coordinate Translation

Consider the coordinate system shown in Figure C.5. The three markers in the local coordinate system $p_{1,i}^{(l)}$, $p_{2,i}^{(l)}$, and $p_{3,i}^{(l)}$ form the triangle $P_1^{(l)}$, $P_2^{(l)}$, $P_3^{(l)}$. Now, let $\mathbf{p}_1^{(l)}$, $\mathbf{p}_2^{(l)}$,

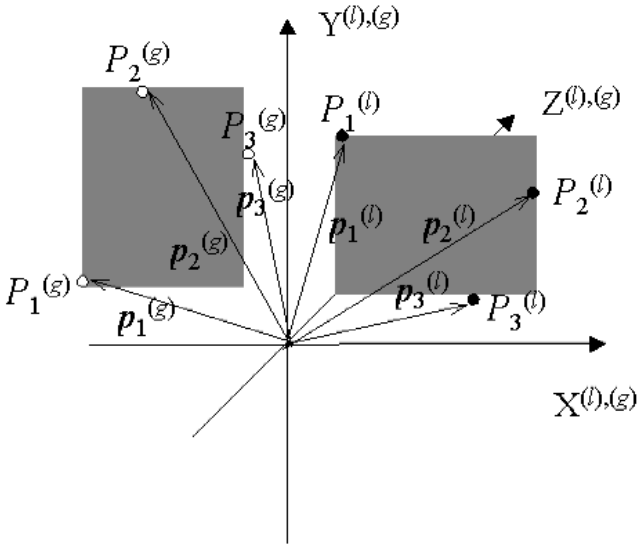


Fig. C.5: Two marker sets in global and local reference systems.

$\mathbf{p}_3^{(l)}$ be position vectors, pointing from the origin of the local reference system to the center of each marker. In the global coordinate system, the three markers $p_{1,i}^{(g)}$, $p_{2,i}^{(g)}$, and $p_{3,i}^{(g)}$ form the triangle $P_1^{(g)}$, $P_2^{(g)}$, $P_3^{(g)}$. Now, let $\mathbf{p}_1^{(g)}$, $\mathbf{p}_2^{(g)}$, $\mathbf{p}_3^{(g)}$ be position vectors, pointing from the origin of the global reference system to the center of each marker. Finally, assume that the origins and axes of both coordinate systems coincide, and that the vectors $\mathbf{p}_1^{(l)}$, $\mathbf{p}_2^{(l)}$, $\mathbf{p}_3^{(l)}$ and $\mathbf{p}_1^{(g)}$, $\mathbf{p}_2^{(g)}$, $\mathbf{p}_3^{(g)}$ represent two different marker sets. With this, the coordinate transformation between the two systems becomes trivial, since the transformation can be performed in the same manner as mapping the local marker set to the global marker set, expressed in Equation C.1.

$$\mathbf{p}_n^{(g)} = \boldsymbol{M}_B \cdot \boldsymbol{M}_A \cdot \mathbf{p}_n^{(l)} + \boldsymbol{t}, n = 1 - 3 \tag{C.1}$$

Here, $\boldsymbol{M}_A$ is a 3 x 3 matrix representing a rotation that orients the plane formed by the l marker set parallel to the plane formed by the g marker set, $\boldsymbol{M}_B$ is a 3 x 3 matrix representing an "in-plane" rotation that aligns corresponding triangle sides with respect to each other, and $\boldsymbol{t}$ is a vector used to correct for the residual translational difference between l points and corresponding g points after performing the rotations on the l triangle.

C.3.2 Coordinate Rotation

Derivation of M_A

In addition to translation, the stereotactic transformation also requires rotating vectors about other non-collinear vectors. To describe this, consider the following situation:

Unit vector $\boldsymbol{v}$ is to be rotated around another unit vector $\boldsymbol{o}$ in a Cartesian coordinate system formed by three orthogonal vectors:

$$\mathbf{o}, \mathbf{p} = \frac{(\mathbf{v} \times \mathbf{o})}{sin(\theta)}, \mathbf{q} = \frac{[\mathbf{o} \times (\mathbf{v} \times \mathbf{o})]}{sin(\theta)} \tag{C.2}$$

where $\frac{1}{sin(\theta)}$ is a necessary factor to preserve unit length of the vectors. Additionally, take note that the angle θ between $\boldsymbol{v}$ and $\boldsymbol{o}$ is described by $cos(\theta) = \boldsymbol{v} \cdot \boldsymbol{o}$. After rotating $\boldsymbol{v}$ around $\boldsymbol{o}$ by angle ϕ, $\boldsymbol{v}$ now becomes $\boldsymbol{v'}$, and can be described in terms of $\boldsymbol{v}$, $\boldsymbol{o}$, and $\boldsymbol{p}$ as follows:

$$v' = (\mathbf{v} \cdot \mathbf{o})\mathbf{o} + sin(\theta)sin(\phi)\mathbf{p} + sin(\theta)cos(\phi)\mathbf{q} \tag{C.3}$$

Now, consider the relationship in Equation C.4:

$$\mathbf{o} \times (\mathbf{v} \times \mathbf{o}) = \mathbf{v} - \mathbf{o}(\mathbf{v} \cdot \mathbf{o}) \tag{C.4}$$

Substituting the expressions in Equation C.2 and considering the relationship in Equation C.4 will yield the following:

$$v' = \mathbf{v}cos(\phi) + \mathbf{o}(\mathbf{v} \cdot \mathbf{o})[1 - cos(\phi)] + (\mathbf{v} \times \mathbf{o})sin(\phi) \tag{C.5}$$

Equation C.5 can be expressed in matrix form as:

$$\mathbf{v'} = \mathbf{M}\ \mathbf{v}$$

where $\boldsymbol{M}$ is the rotation matrix given by:

$$\begin{bmatrix} cos(\Phi) + \mathbf{o}_1^2(1 - cos(\Phi)) & \mathbf{o}_3 sin(\Phi) + \mathbf{o}_1\mathbf{o}_2(1 - cos(\Phi)) & \dots \\ -\mathbf{o}_3 sin(\Phi) + \mathbf{o}_1\mathbf{o}_2(1 - cos(\Phi)) & cos(\Phi) + \mathbf{o}_2^2(1 - cos(\Phi)) & \dots \\ \mathbf{o}_2 sin(\Phi) + \mathbf{o}_1\mathbf{o}_3(1 - cos(\Phi)) & -\mathbf{o}_1 sin(\Phi) + \mathbf{o}_3\mathbf{o}_2(1 - cos(\Phi)) & \dots \end{bmatrix}$$

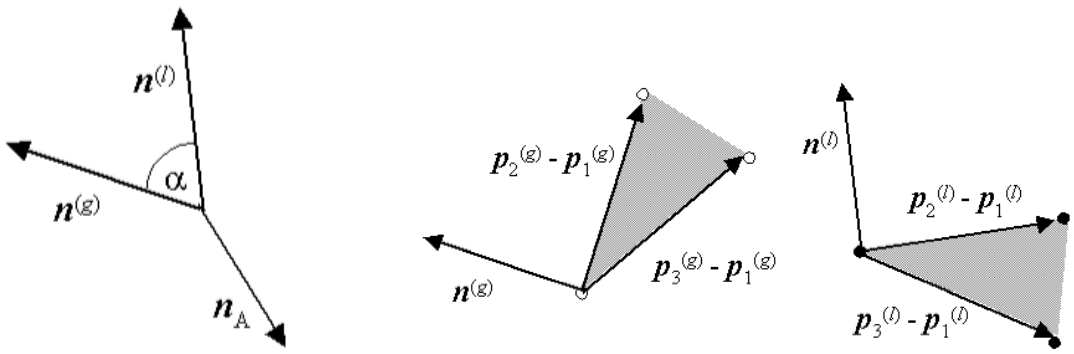
$$\begin{bmatrix} \dots & -\mathbf{o}_2 sin(\Phi) + \mathbf{o}_1\mathbf{o}_3(1 - cos(\Phi)) \\ \dots & -\mathbf{o}_1 sin(\Phi) + \mathbf{o}_2\mathbf{o}_3(1 - cos(\Phi)) \\ \dots & cos(\Phi) + \mathbf{o}_3^2(1 - cos(\Phi)) \end{bmatrix} \tag{C.6}$$

The rotation matrix M_A transforms triangle l such that it becomes coplanar with triangle g. To derive M_A, consider the unit normal vector of triangle l, denoted by $\mathbf{n}^l$, and the unit normal vector of triangle g, denoted by $\mathbf{n}^g$. The transformation performed by M_A corresponds to a rotation of $\mathbf{n}^l$ about the orthogonal vector $n_A = (\mathbf{n}^l \times \mathbf{n}^g)$ by angle α, where $cos(\alpha) = \mathbf{n}^l \cdot \mathbf{n}^g$. This is represented graphically in Figure C.6(a).

Calculating $\mathbf{n}^l$ and $\mathbf{n}^g$ is accomplished by computing and normalizing the following vector products:

$$(\mathbf{p}_3^{(l)} - \mathbf{p}_1^{(l)}) \times (\mathbf{p}_2^{(l)} - \mathbf{p}_1^{(l)}), (\mathbf{p}_3^{(g)} - \mathbf{p}_1^{(g)}) \times (\mathbf{p}_2^{(g)} - \mathbf{p}_1^{(g)}), \tag{C.7}$$

These products are demonstrated graphically in Figure C.6(b).



(a) Rotation of $\mathbf{n}^l$ about n_A.

(b) Calculation of $\mathbf{n}^l$ and $\mathbf{n}^g$.

Fig. C.6: Vector products to derive M_A.

Obtaining the final expression for M_A (Equation C.8) involves normalizing $\mathbf{n}_A$ to produce $\mathbf{o}_A = \frac{\mathbf{n}_A}{sin(\alpha)}$, and combining this result with Equation C.6:

$$M_A = \begin{bmatrix} cos(\alpha) + \mathbf{o}_{A1}^2(1 - cos(\alpha)) & \mathbf{o}_{A3}sin(\alpha) + \mathbf{o}_{A1}\mathbf{o}_{A2}(1 - cos(\alpha)) & \dots \\ -\mathbf{o}_{A3}sin(\alpha) + \mathbf{o}_{A2}\mathbf{o}_{A1}(1 - cos(\alpha)) & cos(\alpha) + \mathbf{o}_{A2}^2(1 - cos(\alpha)) & \dots \\ \mathbf{o}_{A2}sin(\alpha) + \mathbf{o}_{A3}\mathbf{o}_{A1}(1 - cos(\alpha)) & -\mathbf{o}_{A1}sin(\alpha) + \mathbf{o}_{A3}\mathbf{o}_{A2}(1 - cos(\alpha)) & \dots \end{bmatrix}$$

$$\begin{bmatrix} \dots & -\mathbf{o}_{A2}sin(\alpha) + \mathbf{o}_{A1}\mathbf{o}_{A3}(1 - cos(\alpha)) \\ \dots & \mathbf{o}_{A1}sin(\alpha) + \mathbf{o}_{A2}\mathbf{o}_{A3}(1 - cos(\alpha)) \\ \dots & cos(\alpha) + \mathbf{o}_{A3}^2(1 - cos(\alpha)) \end{bmatrix} \tag{C.8}$$

Derivation of M_B and $\mathbf{t}$

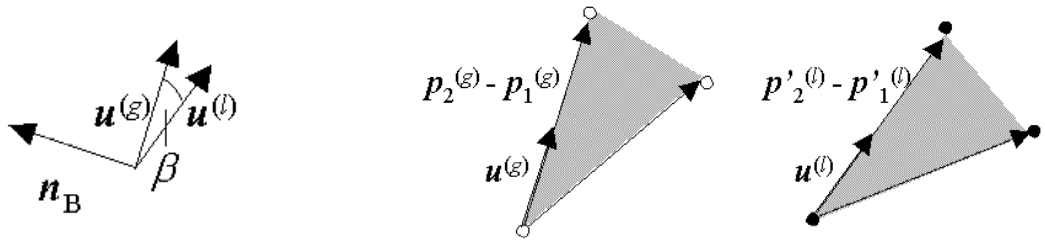
Performing the coordinate rotation described by M_A on $\mathbf{p}_1^{(l)}$, $\mathbf{p}_2^{(l)}$, and $\mathbf{p}_3^{(l)}$ produces three new vectors: $\mathbf{p'}_1^{(l)}$, $\mathbf{p'}_2^{(l)}$, and $\mathbf{p'}_3^{(l)}$. The rotated vectors now form a triangle that is coplanar with the triangle $\boldsymbol{g}$, formed by $\mathbf{p}_1^{(g)}$, $\mathbf{p}_2^{(g)}$, and $\mathbf{p}_3^{(g)}$. In order to complete the stereotactic transformation, a second rotation and a final translation are required. The second rotation orients triangle $\boldsymbol{l}$ identically with respect to triangle $\boldsymbol{g}$, and the translation moves triangle $\boldsymbol{l}$ into the proper location. This rotation is handled by a second rotation matrix, denoted M_B, and the translation is handled by vector $\mathbf{t}$.

Derivation of M_B is accomplished by normalizing the vectors rotated by M_A, to yield the non-collinear unit vectors $u^{(l)}$ and $u^{(g)}$:

$$u^{(l)} = (\mathbf{p'}_2^{(l)} - \mathbf{p'}_1^{(l)}), u^{(g)} = (\mathbf{p}_2^{(g)} - \mathbf{p}_1^{(g)})$$

A graphical representation is shown in Figure C.7(b). The next task is to align $u^{(l)}$ with $u^{(g)}$, which involves rotating $u^{(l)}$ about the orthogonal vector $n_B = (\mathbf{u}^{(l)} \times \mathbf{u}^{(g)})$ by angle β, where $cos(\beta) = \mathbf{u}^{(l)} \cdot \mathbf{u}^{(g)}$. This operation is expressed graphically in Figure C.7(a).

Obtaining the final expression for M_B (Equation C.9) involves normalizing $\mathbf{n}_B$ to produce $\mathbf{o}_B = \frac{\mathbf{n}_B}{sin(\beta)}$, and combining this result with Equation C.6:



(a) Rotation of $\mathbf{u}^{(l)}$ about n_B.

(b) Calculation of $\mathbf{u}^{(l)}$ and $\mathbf{u}^{(g)}$.

Fig. C.7: Vector products to derive M_B.

$$M_B = \begin{bmatrix} cos(\beta) + \mathbf{o}_{B1}^2(1 - cos(\beta)) & \mathbf{o}_{B3}sin(\beta) + \mathbf{o}_{B1}\mathbf{o}_{B2}(1 - cos(\beta)) & \dots \\ -\mathbf{o}_{B3}sin(\beta) + \mathbf{o}_{B2}\mathbf{o}_{B1}(1 - cos(\beta)) & cos(\beta) + \mathbf{o}_{B2}^2(1 - cos(\beta)) & \dots \\ \mathbf{o}_{B2}sin(\beta) + \mathbf{o}_{B3}\mathbf{o}_{B1}(1 - cos(\beta)) & -\mathbf{o}_{B1}sin(\beta) + \mathbf{o}_{B3}\mathbf{o}_{B2}(1 - cos(\beta)) & \dots \end{bmatrix}$$

$$\begin{bmatrix} \dots & -\mathbf{o}_{B2}sin(\beta) + \mathbf{o}_{B1}\mathbf{o}_{B3}(1 - cos(\beta)) \\ \dots & \mathbf{o}_{B1}sin(\beta) + \mathbf{o}_{B2}\mathbf{o}_{B3}(1 - cos(\beta)) \\ \dots & cos(\beta) + \mathbf{o}_{B3}^2(1 - cos(\beta)) \end{bmatrix} \tag{C.9}$$

Performing the coordinate rotation described by M_B on $\mathbf{p'}_1^{(l)}$, $\mathbf{p'}_2^{(l)}$, and $\mathbf{p'}_3^{(l)}$ produces three new vectors: $\mathbf{p''}_1^{(l)}$, $\mathbf{p''}_2^{(l)}$, and $\mathbf{p''}_3^{(l)}$. The rotated vectors now form a triangle that is identical in orientation with respect to triangle $\boldsymbol{g}$, formed by $\mathbf{p}_1^{(g)}$, $\mathbf{p}_2^{(g)}$, and $\mathbf{p}_3^{(g)}$. Translating these vectors by vector

$$\mathbf{t} = \mathbf{p}_1^{(g)} - \mathbf{p''} \tag{C.10}$$

moves the rotated vectors into the correct positions.

When calculating M_B and t, it is possible to use the other two triangle vectors to obtain arithmetic means of the values. In M_B, the other vectors can be used to obtain an estimate of β, to then calculate arithmetic means for the three values in Equation C.9. Doing so will help spread out inaccuracies in the individual points. Similarly, the other two triangle vectors can be used to calculate the arithmetic mean of t.

C.3.3 The Final Expression

Expressing the rotations and translations described by Equations C.8, C.9, and C.10 yields the final expression for the stereotactic transformation of any vector $\mathbf{v}$ from the local coordinate system to the global coordinate system:

$$v^{(l)} = \mathbf{M}_{AB} \cdot \mathbf{v}^{(l)} + \mathbf{t} \tag{C.11}$$

where $\mathbf{M}_{AB} = \mathbf{M}_B \cdot \mathbf{M}_A$, which represents a combination of the rotations described earlier. The inverse translation, which translates any vector $\mathbf{v}$ in the opposite direction, from the global coordinate system to the local coordinate system, is expressed as:

$$v^{(l)} = \mathbf{M}_{AB}^{-1} \cdot \mathbf{v}^{(g)} - \mathbf{t} \tag{C.12}$$

C.3.4 Verifying Correctness

The stereotactic transformations described in Equations C.11 and C.12 properly preserve distances and and angles between points and vectors. This is due to the fact

that the translations involve only translations and proper rotations. Therefore, verifying the accuracy of the translation is quite simple. Verification of the preservation of distances is checked by calculating the determinants of the rotation matrices:

$$det(\mathbf{M}_A) = det(\mathbf{M}_B) = det(\mathbf{M}_{AB}) = 1$$

If this condition holds, then the distances are preserved properly after performing the stereotactic translation. Any other value than 1 indicates that the distances are *not* preserved. A special case occurs when the determinants are equal to -1; this is an indication that the orientation of the coordinate systems is not preserved. In other words, one of the coordinate systems is a correct right-handed system, while the other is a left-handed coordinate system.

A second verification method involves using the inverse transformation, Equation C.12. Applying the transformation to any position vector, then applying the inverse transformation to the translated vector, should yield the original position vector *exactly.* Any deviation from the original position vector indicates errors in the system.

Keep in mind that the stereotactic transformation described here is not perfect by any means. In an ideal system, with ideal data, the transformation will translate the triangles exactly, and the above checks will produce perfect outcomes. However, this is obviously not the case in the real world. Since it is very possible for systematic and/or random errors to creep into the data values, the transformation will not translate the vectors perfectly. Measurements of the marker positions are always subject to errors in the form of random uncertainties. Therefore, the resulting triangles will be slightly different. It is impossible to produce triangles that are *exact,* therefore,

the transformation is designed so that the triangles are *as close as possible.* This condition is contingent upon systematic and random errors. Quantifying and measuring these errors can be accomplished by calculating the geometric distance between the transformed triangle points and the global triangle points. In a properly configured system, these errors are less than 1 mm. Taking this fact into account shows that this transformation is suitable for target localization for use in functional proton radiosurgery: the transformation described here has the capability of maintaining the submillimeter accuracy necessary. An assertion of this fact is presented in Section 6 in the form of an in-depth error analysis, complete with all obtained numerical data.

D. EDGE DETECTION

D.1 Theory

Edge detection is a common method in image processing. The purpose of all edge detection algorithms is to locate strong intensity contrasts in the image, where there is a large jump in intensity from one pixel to the next. There are many different methods of edge detection that exist. Matlab has edge detection functionality built-in. The supported methods include Sobel, Prewitt, Roberts, Laplacian of Gaussian, Zero-Cross, and Canny.

Although each of these methods are similar in output, the method that they use to detect edges can be categorized into one of two families: *gradient* and *Laplacian*.

The *gradient* method searches for the edges in an image by taking the first derivative of the image, and then finding the maxima and minima. For example, consider the signal shown in Figure D.1. Taking the gradient signal, which is the first derivative with respect to t in one dimension, results in Figure D.2.

The derivative shows quite clearly that a maximum exists in the center of the graph, where t = 0. Edges will possess higher intensity values than the pixels that surround them. Depending on the configuration of the method used, this will be considered an edge if the value of the gradient exceeds the specified threshold value. Otherwise, the intensity jump will be disregarded. It is very important to set the

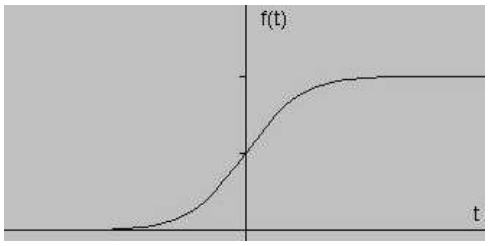


Fig. D.1: Example signal.

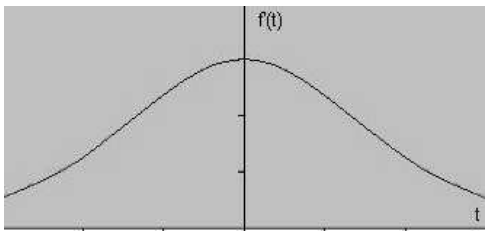


Fig. D.2: First Derivative of Example Signal.

threshold to a value that will detect a suitable number of edges, while not including too much or too little information.

Taking this concept one step further will yield the principle behind the *Laplacian* family of methods. By taking the derivative of the gradient signal (which corresponds to the second derivative of the image), all the aforementioned maxima become zeros, as shown in Figure D.3. The Laplacian methods search for locations in the image where the second derivative is zero. These locations correspond to the locations of edges in the image.

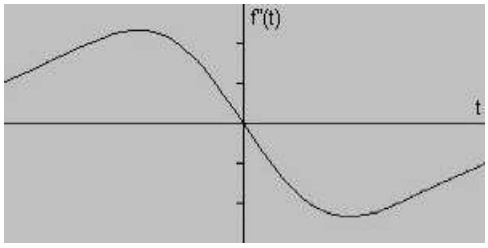


Fig. D.3: Second Derivative of Example Signal.

D.2 Application

Now that the theory has been explained for one dimension, these concepts can be applied to two-dimensional images. Each of the six methods supported in Matlab follows the assumption that there is an accurate method to approximate either the first or second derivatives of two-dimensional images. For each of the six methods described, the sample image shown in Figure D.4 has been used for edge detection.

For the purposes of this project, the object is to find the edges of the phantom–the orange-yellow section–and disregard the empty space around it. The edges obtained

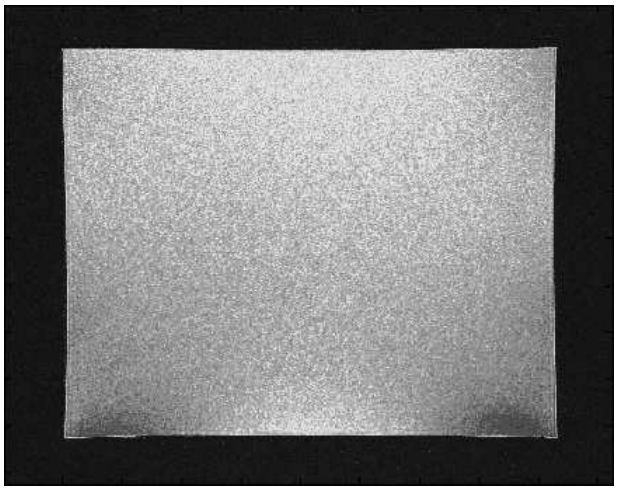


Fig. D.4: Sample MRI Slice.

in each edge image provides the software with the data necessary to calculate the distortion correction model, to correct for gradient nonlinearity distortion (discussed in Chapters 4 and 5). The results of each edge detection algorithm will be shown for comparison purposes.

D.2.1 Sobel

The first (and default) method used in Matlab is the *Sobel* method. Sobel edge detection follows the gradient family of edge detection methods, which means the approximation of the first derivative of the image is used.

The Sobel method returns the approximate absolute gradient maximum at each point in an image. In other words, the points that are returned are those points where the gradient of the image is maximum. Those points that are not stronger than the specified threshold value are not considered edges. The Sobel method uses

two masks to detect edges: one for scanning horizontally, and one for scanning vertically. The masks are then convolved with the image. The Sobel masks are as follows:

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & 2 & 1 \end{bmatrix}$$

To calculate the magnitude of the gradient, the following calculation is performed:

$$|G| = \sqrt{Gx^2 + Gy^2}$$

Matlab Configuration

In Matlab, the Sobel method can be used either with a specified threshold value, or Matlab can determine an appropriate value. Additionally, other options can be specified to configure the Sobel method. The scanning direction-horizontal, vertical, or both-can be used to return specific image results. The default option is to scan in both horizontal and vertical directions, thereby using both masks. Specifying the Sobel method can also allow the user to perform edge thinning on the image. Edge thinning is a technique used to aid the possibility of success in detecting objects in the image, and can help reduce extra processing when the edges are used for labeling and image transformation. Edge thinning is turned on by default, but can be turned off by specifying 'nothinning' when calling the function. If it is deemed necessary to analyze the gradient responses in the vertical and/or horizontal directions, the

following commands will yield these results:

```
[BW,thresh,gv,gh] = edge(I,'sobel',...)
```

Or, the following lines can be used:

```
if ~(isa(I,'double') || isa(I,'single')); I = im2single(I);
gh = imfilter(I,fspecial('sobel') /8,'replicate');
gv = imfilter(I,fspecial('sobel')'/8,'replicate');
```

Keep in mind that in order to achieve desirable results that best suit the application, it is necessary to test different values and options, to fully understand how the Sobel method works.

Results

Figure D.5 demonstrates the result of using the Sobel method on the sample image, with all default options (where threshold = 2.0554×10^{-4} specified by Matlab).

Through further experimentation by increasing default threshold value, it was found that using a threshold value of approximately 1.5 times stronger than the default removed the extra noise in the image, to leave just the physical edges of the phantom. Figure D.6 demonstrates the result of using a threshold value 1.5 times stronger than that specified by Matlab (threshold = 3.0830×10^{-4}).

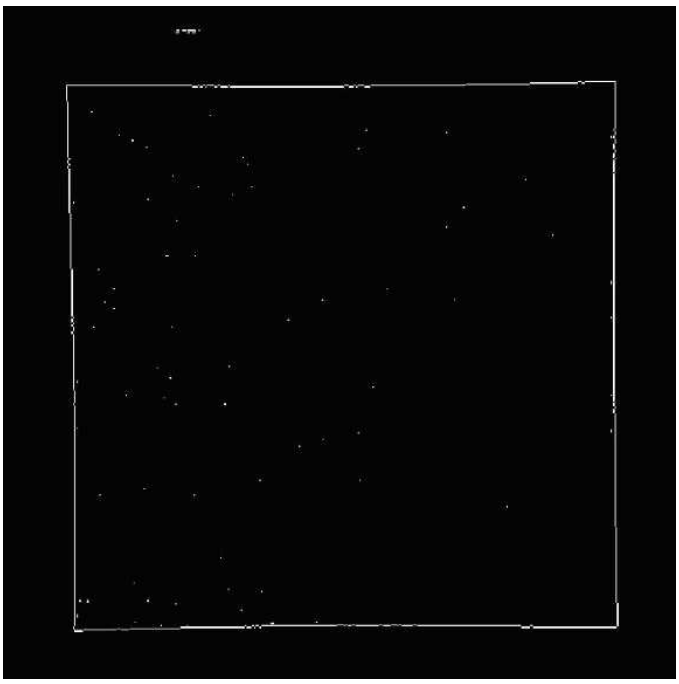


Fig. D.5: Results of Sobel with Default Threshold Value = 2.0554×10^{-4}.

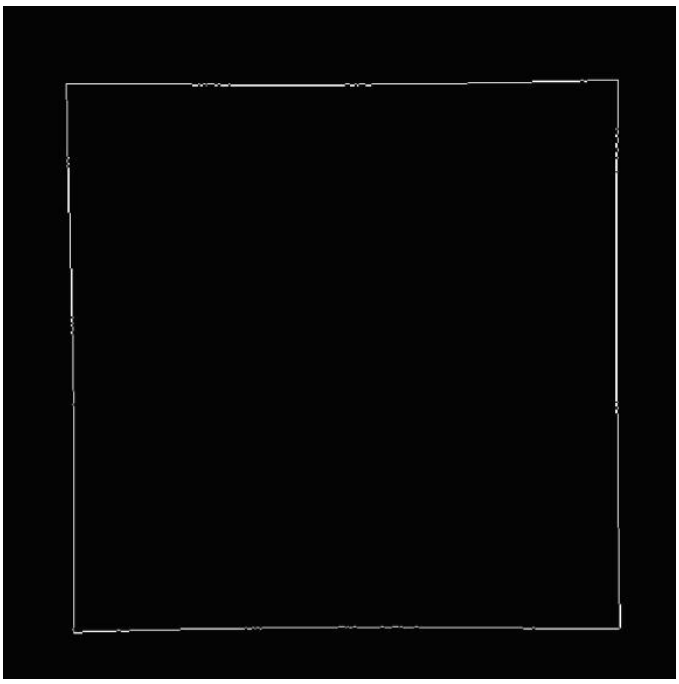


Fig. D.6: Results of Sobel with Threshold Value = 3.0830×10^{-4}.

D.2.2 Prewitt

Another method that Matlab supports is known as the *Prewitt* method. It is similar to the Sobel method in that it is a member of the gradient family, making use of the first derivative approximation of the images it manipulates. The Prewitt method is specifically used to highlight areas in the image that have high spatial frequency. The Prewitt method is a variation of the Sobel method, following the same process as the Sobel method, but uses different masks:

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}, G_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

The results that are returned by the Prewitt method are similar to those by Sobel, but the results are not as isotropic as Sobel. In the Sobel method, the masks used on the image apply equally well over all parts of the image, and does not possess any bias on any particular set of compass directions. Because of this higher degree of isotropy, Sobel produces brighter images than Prewitt.

Matlab Configuration

In Matlab, the Prewitt method can be used either with a specified threshold value, or Matlab can determine an appropriate value. Additionally, other options can be specified to configure the Prewitt method. The scanning direction-horizontal, vertical, or both-can be used to return specific image results. The default option is to scan in both horizontal and vertical directions, thereby using both masks.

Keep in mind that in order to achieve desirable results that best suit the application, it is necessary to test different values and options, to fully understand how the Prewitt method works.

Results

Figure D.7 shows the result of using the Prewitt method on the sample image, with all default options (where threshold = 1.9985×10^{-4} specified by Matlab).

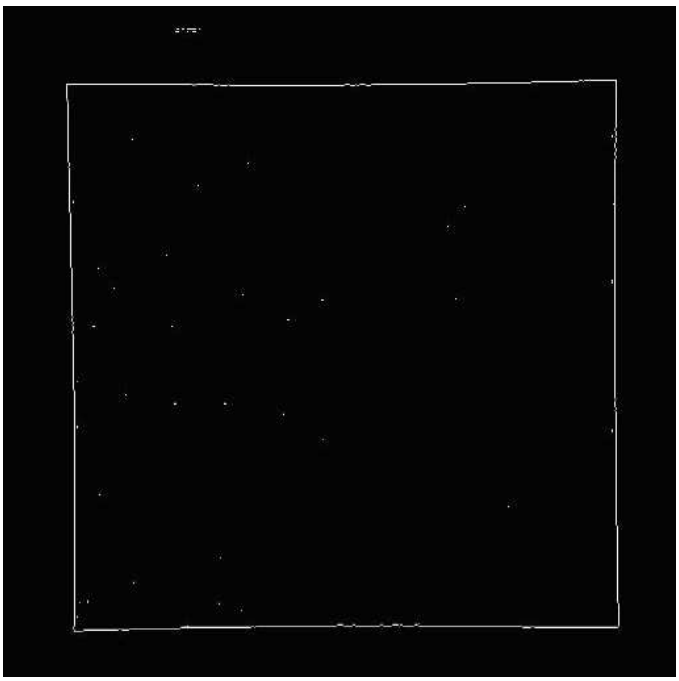


Fig. D.7: Results of Prewitt with Default Threshold Value = 1.9985×10^{-4}.

Through further experimentation by increasing default threshold value, it was found that using a threshold value of approximately 1.5 times stronger than the default removed the extra noise in the image, to leave just the physical edges of the phantom. Figure D.8 shows the result of using a threshold value 1.5 times stronger

than that specified by Matlab (threshold = 3.0830×10^{-4}).

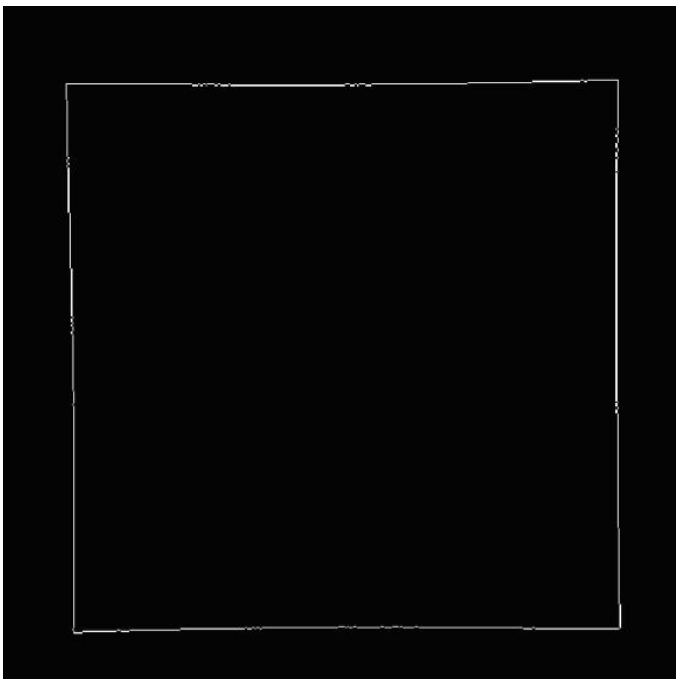


Fig. D.8: Results of Prewitt with Threshold Value = 3.0830×10^{-4}.

D.2.3 Roberts

The third method supported by Matlab is the *Roberts* method. The Roberts method is also similar to the Sobel method, in that it belongs to the gradient family, using the first derivative approximation of images to detect edges. A similar method to Sobel is used in Roberts, but the masks used on the image are quite different:

$$G_x = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}, G_y = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

The result of the Roberts edge detector is sharp edges that run at 45 degrees according to the pixel grid. The advantage to using the Roberts edge detector is that it is faster than most other methods. Using 2x2 kernels versus 3x3 kernels reduces the number of inputs to be examined by over 50 percent for each pixel. However, the disadvantage to this method is its susceptibility to noise. Additionally, the results only show strong edges, so there is a possibility that some of the features in the image will be lost. Sobel considers softer edges, and might give a better representation of the actual original image. To compensate for this property, proper threshold values are key to decent performance when using Roberts.

Matlab Configuration

In Matlab, the Roberts method can be used either with a specified threshold value, or Matlab can determine an appropriate value. Additionally, other options can be specified to configure Roberts. Specifying the Sobel method can also allow the user to perform or not perform edge thinning on the image. Edge thinning is turned on by default (or can be forced by specifying 'thinning'), but can be turned off by specifying 'nothinning' when calling the function. If it is deemed necessary to analyze the gradient responses at 45 and 135 to the Roberts gradient operators, the following commands will yield these results:

```
[BW,thresh,g45,g135] = edge(I,'roberts',...)
```

Or, the following lines can be used:

```
if ~(isa(I,'double') || isa(I,'single'));
    I = im2single(I);
end
g45  = imfilter(I,[1 0; 0 -1]/2,'replicate');
g135 = imfilter(I,[0 1;-1  0]/2,'replicate');
```

Keep in mind that in order to achieve desirable results that best suit the application, it is necessary to test different values and options, to fully understand how the Roberts method works.

Results

Figure D.9 demonstrates the result of using the Roberts method on the sample image, with all default options (where threshold = 2.8711×10^{-4} specified by Matlab).

Through further experimentation by increasing default threshold value, it was found that using a threshold value of approximately 1.6 times stronger than the default removed the extra noise in the image, to leave just the physical edges of the phantom. Figure D.10 demonstrates the result of using a threshold value 1.6 times stronger than that specified by Matlab (threshold = 4.5937×10^{-4}).

D.2.4 Laplacian of Gaussian (LOG)

The fourth method supported by Matlab is the *Laplacian of Gaussian (LOG)* method. As the name implies, the LOG method belongs to the Laplacian family, in that it searches for zero crossings in the second derivative of the image. The Laplacian and LOG operators make use of two 3x3 convolution kernels. The kernels need to be

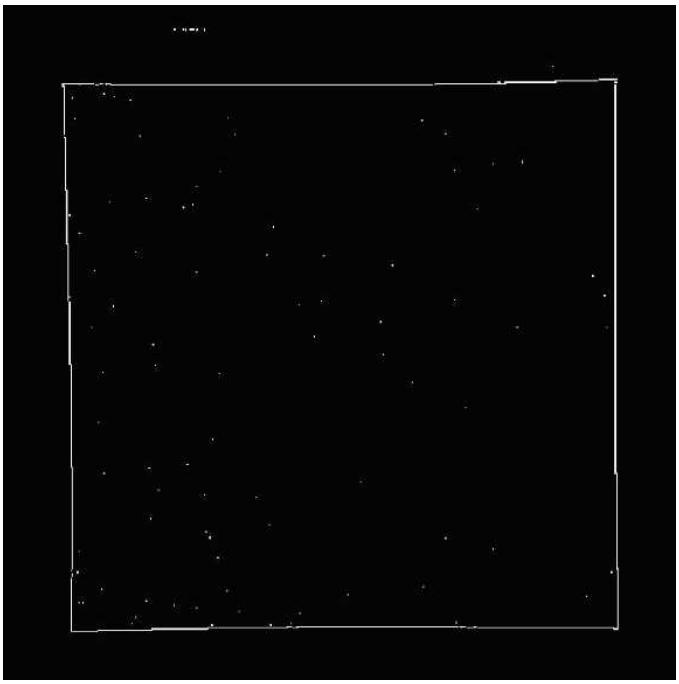


Fig. D.9: Results of Roberts with Default Threshold Value = 2.8711×10^{-4}.

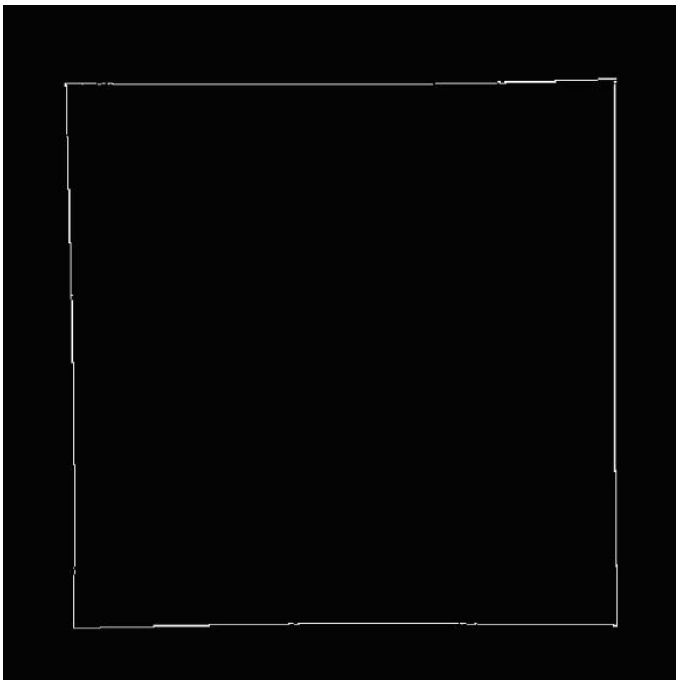


Fig. D.10: Results of Roberts with Threshold Value = 4.5937×10^{-4}.

computed by sampling a discrete Gaussian function, so that they best approximate the second derivative of the image, as described in the definition of the Laplacian. Because the image is a set of discrete pixels, the convolution performed on the image is also discrete. The following are possible kernels that may be used by LOG (after sampling a Gaussian):

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 1 & 1 \\ 1 & 8 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \begin{bmatrix} -1 & 2 & -1 \\ 2 & -4 & 2 \\ -1 & 2 & -1 \end{bmatrix}$$

Because Laplacian and LOG use an approximation of the second derivative of the image, they are both very susceptible to noise. If a noisy image is first smoothed using Gaussian smoothing, the noise can be suppressed to a certain extent, thereby increasing the effectiveness of the algorithm. The LOG method goes one step further than the basic Laplacian method, in that it performs Gaussian smoothing on the image prior to applying the Laplacian method (hence the name "Laplacian of Gaussian"). However, using LOG will reduce the number of edges detected, since applying Gaussian smoothing softens images. In addition to a reduction in the number of detected edges, those detected edges will not be as bright as those produced by the basic Laplacian, or any other edge detection method.

Matlab Configuration

In Matlab, there are not many options available for configuring the LOG method. LOG can be used either with a specified threshold value, or Matlab can determine

an appropriate value. Additionally, the standard deviation s for the Gaussian kernel can be specified. Otherwise, the default standard deviation is 1, and the size of the filter is chosen automatically by Matlab, based on the standard deviation.

Keep in mind that in order to achieve desirable results that best suit the application, it is necessary to test different values and options, to fully understand how the LOG method works.

Results

Figure D.11 shows the result of using the LOG method on the sample image, with all default options (where threshold = 5.9078×10^{-6} specified by Matlab):

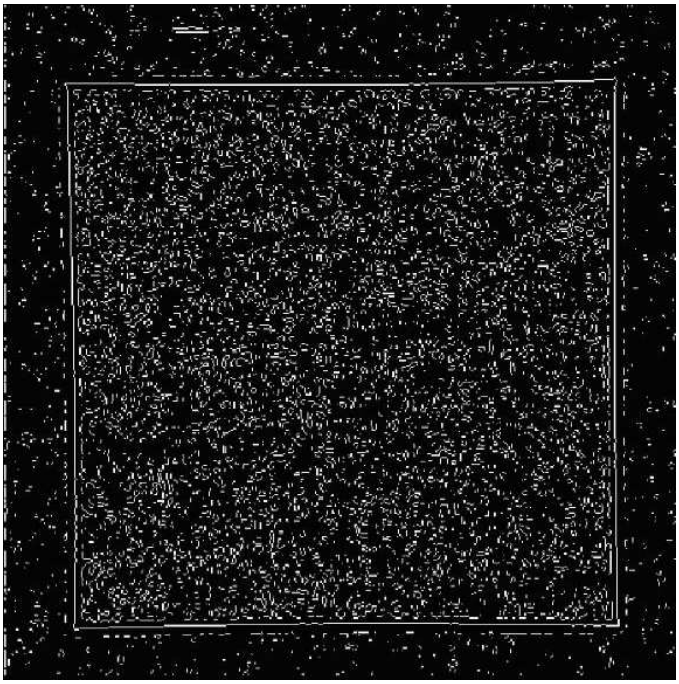
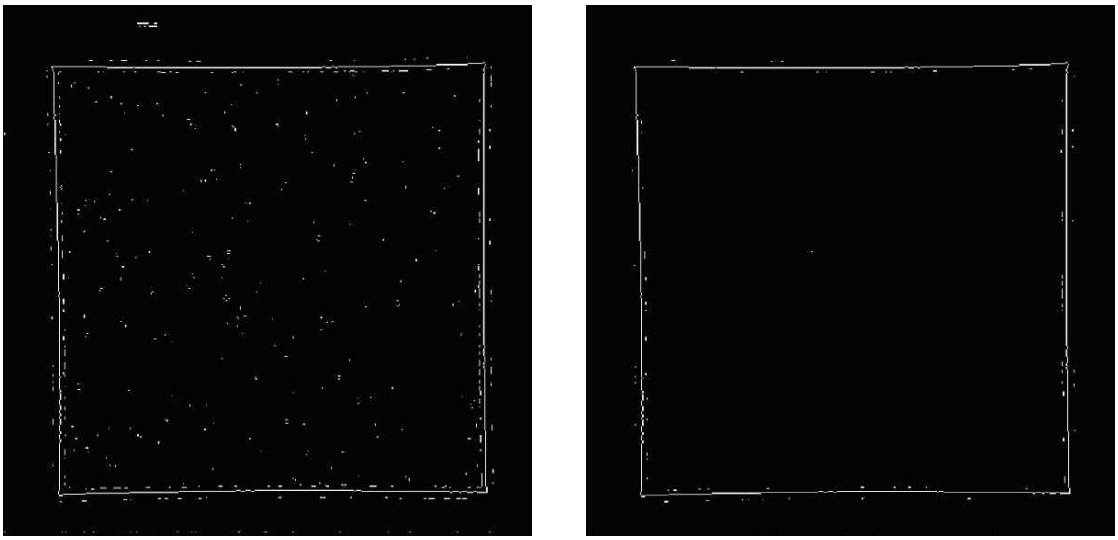


Fig. D.11: Results of LOG with Default Threshold Value = 5.9078×10^{-6}.

It is clear to see that with the default options, the LOG method detects too many

edges in the image. The physical edge of the phantom is detected quite well, but the image also includes other small variations in intensity throughout the image.

It is here that it becomes apparent the importance of properly specifying threshold values. Figure D.12 demonstrates two images, representing doubling and tripling the Matlab-specified threshold values, respectively (threshold = 1.1816×10^{-5} and threshold = 1.7724×10^{-5}):



(a) Threshold = 1.1816×10^{-5}. (b) Threshold = 1.7724×10^{-5}.

Fig. D.12: Results of LOG with increased threshold values.

In order to remove the rest of the noise and leave the physical edges of the phantom present in the image, the threshold value must be approximately 5.1 times greater than the originally specified threshold value. Figure D.13 represents LOG using a threshold 5.1 times stronger than the original threshold specified by Matlab (threshold = 3.0130×10^{-5}).

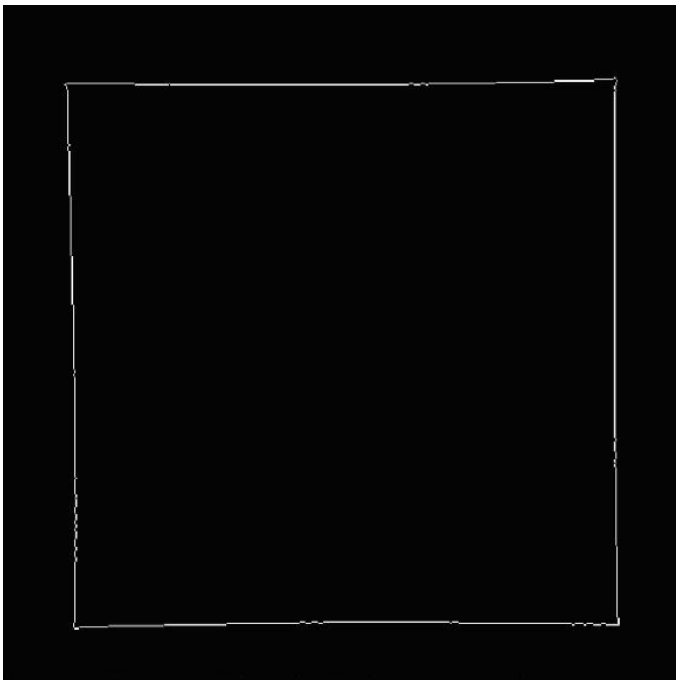


Fig. D.13: Results of LOG with Threshold = 3.0130×10^{-5}.

D.2.5 Zero-Cross

The fifth method is the *Zero-Cross* method. The Zero-Cross is quite similar to the Laplacian and LOG methods, in that it searches for zero crossings in the second derivative of the image. The method that is used here is exactly the same as that employed by Laplacian and LOG. The only difference with Zero-Cross is that Matlab allows the user to specify the filter to be applied. Zero-cross is a general Laplacian method that can filter an image using any specified filter, not just with a Gaussian filter, as is used with LOG. As with LOG, Zero-Cross is very susceptible to noise because it makes use of an approximation of the second derivative of the image.

Matlab Configuration

In Matlab, there are not many options available for configuring the Zero-Cross method. LOG can be used either with a specified threshold value, or Matlab can determine an appropriate value. Additionally, the filter h can be specified to filter the image using the Zero-Cross method. Otherwise, Matlab will revert to using a Gaussian filter, and the default Zero-Cross method will act just the same as a LOG.

Keep in mind that in order to achieve desirable results that best suit the application, it is necessary to test different threshold values and kernels, to fully understand how the Zero-Cross method works.

Results

Since the default settings for the Zero-Cross method result in a default LOG method, refer to the images output by the LOG method (Figures D.11 - D.13.

D.2.6 Canny

The sixth and final method used in Matlab is the *Canny* method. The Canny method is quite different than the rest of the methods described here. The Canny edge detector was designed to be the optimal edge detector, in that it tracks and plots intensity discontinuities. Because of its unique properties, and the advanced methods it employs, Canny does not fit into either of the aforementioned families of edge detectors.

The Canny is much more advanced than the previously mentioned methods. Detecting edges with Canny involves three separate steps. The first step is to smooth the image using a Gaussian filter, similar to the initial step of LOG. This step is called the **canny enhancer**. The output of this step is a strength image (which describes the edge strength at each pixel), and an orientation image (which describes the orientation of pixels to the edge normal).

The second step is to perform a process called **nonmaximum suppression**. The strength image output by the first step may contain wide ridges around local maxima. Nonmaximum suppression thins these ridges to produce 1-pixel wide edges. This is similar to the edge thinning method described in Sobel and Roberts. The output is a set of thinned edge points resulting from suppressing nonmaxima edge points.

The third and final step is to remove local maxima created by noise, using a process called **hysteresis thresholding**. In the other 5 methods, utilizing only one threshold value can create false contours or streaking in the final image. To prevent this from occurring, Canny utilizes two threshold values: a maximum and a minimum. These maximum and minimum threshold values are used by hysteresis thresholding to

produce a set of lists, which describe the position of a connected contour in the image. These lists, as well as the strength and orientation images from the canny enhancer, describe the properties of all edge points. The purpose of hysteresis thresholding is to also perform edge tracking, in that it finds chains of connected edge maxima, or connected contours. This data can be used with edge point descriptors to aid in curve detection.

Matlab Configuration

In Matlab, Canny edge detection can be configured using several options. Because of the use of two threshold values by hysteresis thresholding, the user must specify a two-element matrix, containing a low and high threshold value. If one value is specified for the threshold (i.e., a scalar rather than a vector), this value is used for the high threshold and 0.4*value is used for the low threshold. If the threshold is not specified, Matlab automatically chooses a low and high value for threshold. Additionally, the standard deviation s of the Gaussian filter can be specified. The default standard deviation is 1, and the size of the filter is chosen automatically by Matlab, based on the standard deviation.

Keep in mind that in order to achieve desirable results that best suit the application, it is necessary to test different threshold values and s values, to fully understand how the Canny edge detection method works.

Results

Figure D.14 is the result of using the Canny method on the sample image, with all default options (where low threshold = 0.0188 and high threshold = 0.0469 specified by Matlab).

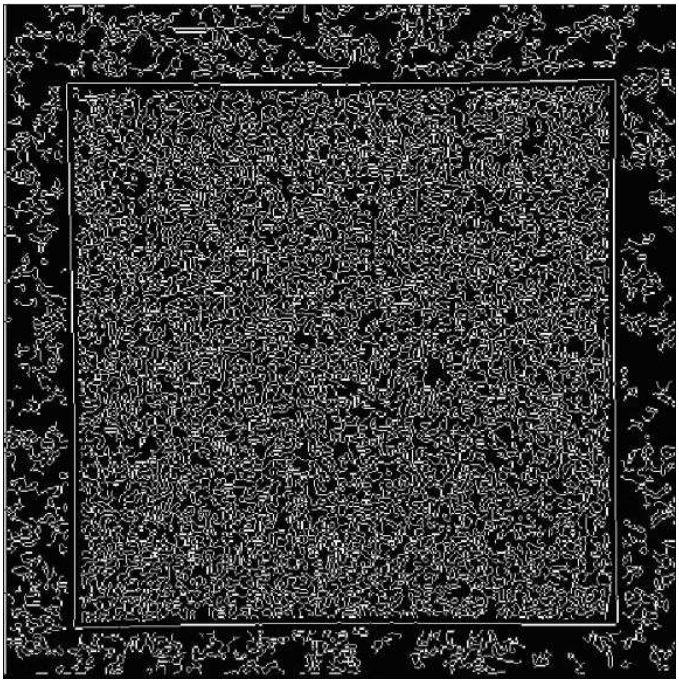
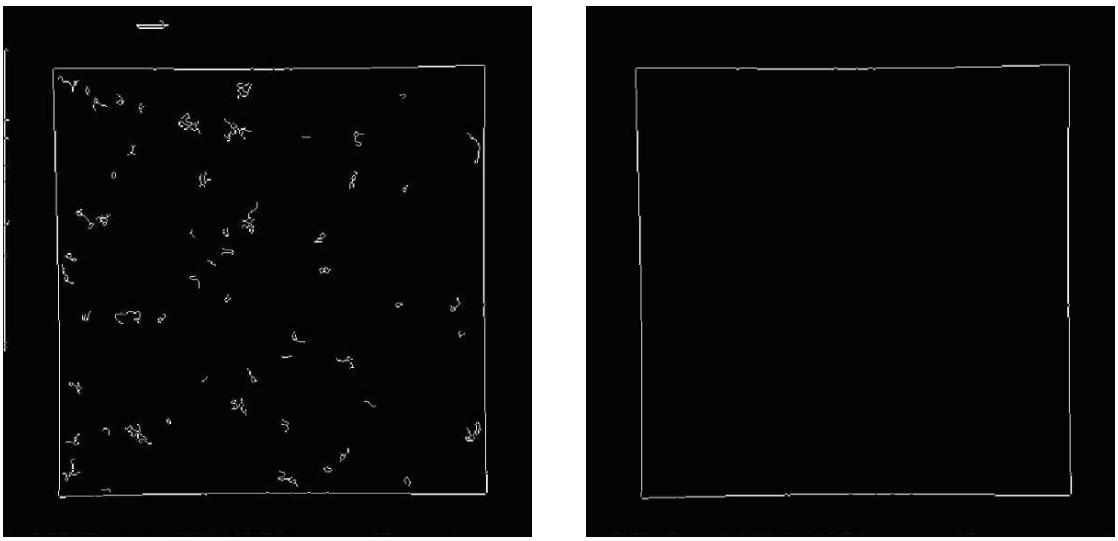


Fig. D.14: Results of Canny with Default Threshold Values = 0.0188 and 0.0469.

It is clear to see that with the default options, the Canny method detects too many edges in the image. The physical edge of the phantom is detected quite well, but the image also includes other small variations in intensity throughout the image.

It is here that it becomes apparent the importance of properly specifying threshold values. Figure D.15 contains two images, representing a doubling (low threshold = 0.0375 and high threshold = 0.0938) and tripling (low threshold = 0.0563 and high

threshold = 0.1406) the Matlab-specified threshold values, respectively.



(a) Thresholds = 0.0563 and 0.1406.

(b) Thresholds = 0.0563 and 0.1406.

Fig. D.15: Results of Canny with increasing threshold values..

The approximate minimum threshold values that yield just the phantom edges and no noise was found to be approximately 2.9 times of those specified by Matlab (where low threshold = 0.0544 and high threshold = 0.1359), as demonstrated in Figure D.16.

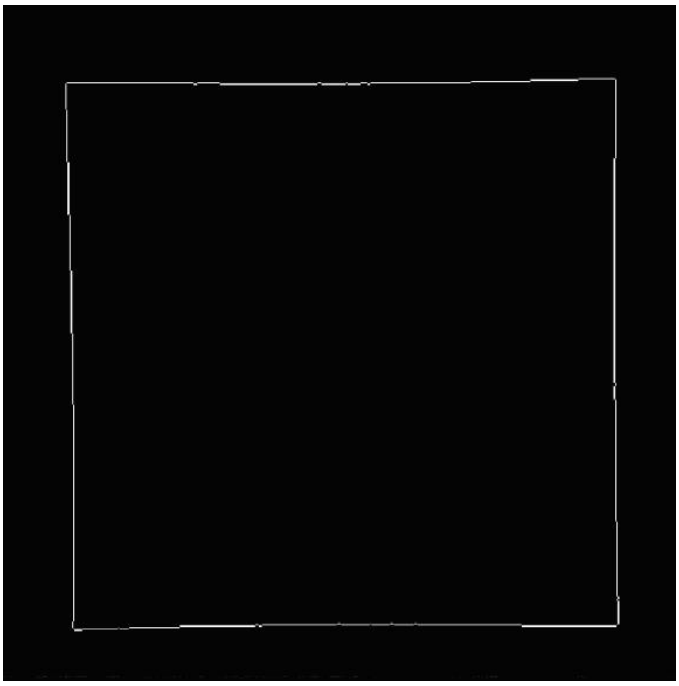


Fig. D.16: Results of Canny with Thresholds = 0.0544 and 0.1359.

D.3 Conclusion

Each of the six edge detection methods specified by Matlab uses different methods to detect the edges within an image. Some are simple and straightforward, others go to great lengths to determine edges. The method that best suits the needs of a particular application must be decided by the user. Proper configuration of each method is critical to ensuring optimal and correct operation for the intended application.

In terms of this project, the Canny edge detector was chosen to generate all edge images. After analyzing the above results, the reasons for this decision become clear:

- Canny was least sensitive to noise.

- The resulting images were the most consistent amongst all methods.
- Edges produced by Canny were of higher quality than other methods.

Indeed, any of these methods can be used to produce the edge images necessary for this project. However, Canny is the most robust (and the most complicated) edge detection method supported by Matlab. Even though any of the methods will suffice, Canny is the safest bet. Minimizing susceptibility to noise is of the utmost importance, since ***all*** the calculations performed in the distortion correction are based directly on the quality of the edge images. If the edge images created are of poor quality, then the correction will also be of poor quality.

REFERENCES

[1] P. C. Shrimpton, M. C. Hillier, and M. A. Lewis, "Doses from Computed Tomography (CT) Examinations in the UK - 2003 Review," *Health Protection Agency*, vol. NRPB-W67, 2003.

[2] S. Clare, "Functional MRI : Methods and Applications," *Doctoral dissertation, University of Nottingham*, 1997.

[3] TS. Sumanaweera, "Segmentation and Distortion Correction in Medical Imaging," *Doctoral dissertation, Stanford University*, 1992.

[4] Michael T. Heath, "Scientific Computing - An Introductory Survey," pp. 105–140, 2002.

[5] Wilson RR, "Radiological Use of Fast Protons," *Radiology*, vol. 47, pp. 487–91, 1946.

[6] Tobias CA., Anger HO., Lawrence JH., and Am. J. Roentgenol., "Radiological Use of High Energy Deuterons and Alpha Particles," *Radiat. Ther. Nucl. Med.*, vol. 67, pp. 1–27, 1952.

[7] Tobias CA., van Dyke DC., Simpson ME., Anger HO., Huff RL., Koneff AA., and Am. J. Roentgenol., "Irradiation of the Pituitary of the Rat with High Energy Deuterons," *Radiat. Ther. Nucl. Med.*, vol. 72, pp. 1–21, 1954.

[8] Tobias CA., Roberts JE., Lawrence JH., Low-Beer BVA., Anger HO., Born JL., McCombs R., and Huggins C., "Irradiation Hypophysectomy and Related Studies Using 340-MeV Protons and 190-MeV Deuterons," *Peaceful Uses of Atomic Energy*, vol. 10, pp. 93–106, 1956.

[9] Tobias CA., Lawrence JH., Born JL., McCombs R., Roberts JE., Anger HO., Low-Beer BVA., and Huggins C., "Pituitary Irradiation with High Energy Proton Beams: a Preliminary Report," *Cancer Res.*, vol. 18, pp. 121–134, 1958.

[10] van Dyke DC., Simpson ME., Koneff AA., and Tobias CA., "Long-term Effects of Deuteron Irradiation on the Rat Pituitary," *Endocrinology*, vol. 64, pp. 240–257, 1959.

[11] Larsson B., Leksell L., Rexed B., and Sourander P., "Effect of High-energy Protons on the Spinal Cord," *Acta Radiologica*, vol. 51, pp. 52–64, 1959.

[12] Falkmer S., Larsson B., and Stenson S., "Effects of Single-dose Proton Irradiation of Normal Skin and Vx2 Carcinoma in Rabbit Ears," *Acta Radiologica*, vol. 52, pp. 217–234, 1959.

[13] Rexed B., Mair W., Sourander P., Larsson B., and Leksell L., "Effect of High-energy Protons on the Brain of the Rabbit," *Acta Radiologica*, vol. 53, pp. 289–299, 1960.

[14] Larsson B., Kihlman BA., and Int. J., "Chromosome Aberrations Following Irradiation with High-energy Protons and their Secondary Radiation: a Study of Dose Distribution and Biological Efficiency Using Root-tips of Vicia faba and Allium cepa," *Radiat. Biol.*, vol. 2, pp. 8–19, 1960.

[15] Larsson B. and Brit. J., "Pretherapeutic Physical Experiments with High Energy Protons," *Radiol.*, vol. 34, pp. 143–151, 1961.

[16] Larsson B., "On the Application of a 185 MeV Proton Beam to Experimental Cancer Therapy and Neurosurgery: a Biophysical Study," *Abstracts of Uppsala Dissertations in Science. Uppsala, Sweden: Almqvist and Wiksells Boktryckeri AB*, 1962.

[17] Raju MR., "Proton Radiobiology, Radiosurgery and Radiotherapy," *Int. J. Radiat. Biol.*, vol. 67, pp. 237–259, 1995.

[18] Mahesh Neupane, "Optimization of Sequential Alignment Verification and Positioning System (SAVPS) for Proton Radiosurgery," M.S. thesis, CSUSB, September 2005.

[19] Wikipedia, "Computed tomography — wikipedia, the free encyclopedia," 2006, [Online; accessed 10-October-2006].

[20] Avinash Kak and Malcolm Slaney, "Principles of Computerized Tomographic Imaging," *IEEE Press*, 1988.

[21] Stanley R. Deans, "The Radon Transform and Some of Its Applications," *New York: John Wiley & Sons.*, 1983.

[22] Frank Natterer, "The Mathematics of Computerized Tomography," *Classics in Applied Mathematics, Society for Industrial and Applied Mathematics.*, p. 32.

[23] Frank Natterer and Frank Wubbeling, " Mathematical Methods in Image Reconstruction.," *Society for Industrial and Applied Mathematics.*

[24] Ambrose J. and Hounsfield G., "Computerized Transverse Axial Tomography," *Br J Radiol.*, vol. 46(542), pp. 148–9, 1973 Feb.

[25] Hounsfield GN., "Computerized Transverse Axial Scanning (Tomography). 1. Description of system.," *Br J Radiol.*, vol. 46(552), pp. 1016–22, 1973 Dec.

[26] "Glossary of methodologic terms.," vol. Archives of Physical Medicine and Rehabilitation, 2006.

[27] Wikipedia, "Magnetic resonance imaging — wikipedia, the free encyclopedia," 2006, [Online; accessed 10-October-2006].

[28] Tweig D., "The K-Trajectory Formulation of the NMR Imaging Process with Applications in Analysis and Synthesis of Imaging Methods," *Med Phys*, vol. 10, pp. 610–621, 1983.

[29] G.E Martin and A.S. Zekter, "Two-Dimensional NMR Methods for Establishing Molecular Connectivity," *VCH Publishers, Inc: New York*, p. 59, 1988.

[30] J.W. Akitt and B.E. Mann, "NMR and Chemistry," *Stanley Thornes: Cheltenham*, pp. 273, 287, 2000.

[31] J.M. Tyszka, S.E. Fraser, and R.E. Jacobs, "Magnetic Resonance Microscopy: Recent Advances and Applications," *Current Opinion in Biotechnology*, vol. 16(1), pp. 93–99, 2005.

[32] Ljunggren S., ," *J Magn Reson*, vol. 54, pp. 338, 1983.

[33] S. Langlois, M. Desvignes, J.M. Constans, and M. Revenu, "MRI geometric

distortion: a simple approach to correcting the effects of non-linear gradient fields," *J. Magn. Reson. Imaging*, vol. 9, pp. 821–831, 1999.

[34] R. A. Serway and R. J. Beichner, "Physics for Scientists and Engineers, with Modern Physics," pp. 938–943, Harcourt, 2000.

[35] Vincent Guffens, "Study and Correction of Field Inhomogeneity in Magnetic Resonance Imaging," *Dissertation, Université Catholique de Louvain-La-Neuve*, 2000.

[36] J. Jovicich, S. Czanner, D. Greve, E. Haley, A. van der Kouwe, R. Gollub, D. Kennedy, F. Schmitt, G. Brown, J. MacFall, B. Fischl, and D. Anders, "Reliability in multi-site structural MRI studies: Effects of gradient non-linearity correction on phantom and human data," *Neuroimage*, vol. 30, pp. 436–443, 2006.

[37] D. Wang, W. Strugnell, G. Cowin, D.M. Doddrell, and R. Slaughter, "Geometric distortion in clinical MRI systems Part I: evaluation using a 3D phantom," *J. Magn. Reson. Imaging*, vol. 22, pp. 1211–1221, 2004.

[38] D. Wang, W. Strugnell, G. Cowin, D.M. Doddrell, and R. Slaughter, "Geometric distortion in clinical MRI systems Part II: correction using a 3D phantom," *J. Magn. Reson. Imaging*, vol. 22, pp. 1223–1232, 2004.

[39] D. Wang, D.M. Doddrell, and G. Cowin, "A novel phantom and method for comprehensive 3-dimensional measurement and correction of geometric distortion in magnetic resonance imaging ," *Magn. Reson. Imaging*, vol. 22 (4), pp. 529542, 2004.

[40] T.S. Lee, K.E. Schubert, and R.W. Schulte., "Software Development For Correction of Gradient-Nonlinearity Distortions in MR Images.," *Accepted to ICSSA 06*, 2006.

[41] T.S. Lee, K.E. Schubert, and R.W. Schulte., "Gradient Non-Linearity Correction of MR Images for Functional Radiosurgery," *Accepted to ICIS 06*, 2006.

[42] D. Kondziolka, "Functional radiosurgery," *Neurosurgery*, vol. 44, pp. 12–20, 1999.

[43] G. Bednarz, M.B. Downes, B.W. Corn, W.J. Curran, and H.W. Goldman, "Evaluation of the spatial accuracy of magnetic resonance imaging-based stereotactic target localization for gamma knife radiosurgery of functional disorders," *Neurosurgery*, vol. 45, pp. 1156–1161, 1999.

[44] S.J. Doran, L. Charles-Edwards, S.A. Reinsberg, and M.O. Leach MO, "A complete distortion correction for MR images: I. Gradient warp correction," *Phys. Med. Biol.*, vol. 50, pp. 1343–1361, 2005.

[45] G. Arfken, "Convolution Theorem," *Mathematical Methods for Physicists, 3rd ed.*, pp. 810–814, Orlando, FL: Academic Press, 1985.

[46] R. Bracewell, "The Fourier Transform and Its Applications, 3rd ed.," pp. 108–112, New York: McGraw-Hill, 1999.

[47] K. Weaver, V. Smith, J.D. Lewis, B. Lulu, and et. al, "A CT based computerized treatment planning system for I-125 stereotactic brain implants," *Int. J. Radiat. Oncol. Biol. Phys.*, vol. 18, pp. 445–454, 1990.