Application of Magnetic Resonance Imaging to Radiotherapy Treatment Planning and Neurosurgery

Wayne Allan Beckham M.Sc.(Med.Phys.)

Department of Physics and Mathematical Physics
University of Adelaide

A thesis submitted in fulfilment of the degree of Doctor of Philosophy in the Faculty of Science

08 August 1997
# Table of Contents

Table of Contents ................................................................. i

List of Figures ................................................................. v

Abstract .............................................................................. xii

Statement of Authenticity ...................................................... xiii

Acknowledgments ............................................................... xiv

1. Chapter 1: Theory of MRI and its Application to Imaging .............. 1
   1.1 Introduction .............................................................. 1
   1.2 Historical Evolution of Magnetism, Nuclear Magnetic Resonance and Magnetic Resonance Imaging .......................... 2
   1.3 MRI Components ...................................................... 4
   1.4 Nuclear Magnetic Resonance Model .................................. 7
   1.4.1 Excitation .......................................................... 8
   1.4.2 Relaxation .......................................................... 12
   1.5 Localisation of the NMR Signal and Origins of System Related Distortion ......................................................... 18
   1.5.1 Slice Select Gradient (z-gradient) .................................. 19
   1.5.2 Phase Encode or Preparation Gradient (y-gradient) ....... 21
   1.5.3 Frequency Encode or Read-out Gradient (x-gradient) ...... 22
   1.5.4 Final Image Reconstruction ...................................... 25
   1.5.5 Eddy Currents ..................................................... 26
   1.6 Origins of Sample Dependent Distortion ............................ 27
   1.6.1 Magnetic Susceptibility and Chemical Shift Effects ......... 28
   1.7 Effects of Distortion of MRI in Radiotherapy Treatment Planning and in Neurosurgery ........................................ 29
   1.8 Summary of the Specific Aims for This Project .................. 34
2. Chapter 2: RAH MRI Distortion Description .............................................. 36
   2.1 Introduction .................................................................................. 36
   2.2 RAH Spatial Linearity Phantom ...................................................... 37
   2.3 MR Images of the Phantom .............................................................. 43
   2.4 Peak Search Algorithm Description ................................................. 44
      2.4.1 File Structure .................................................................... 44
      2.4.2 On the Nature of the Spatial Linearity Phantom Image Data .......... 46
   2.4.3 Development of the Peak Search Algorithm ................................. 50
   2.5 Peak Search Algorithm Results ...................................................... 58
   2.6 Methods of Distortion Magnitude Assessment ............................... 60
      2.6.1 3-D Plot of Distortion Magnitude and Associated Cartesian Coordinates ......................................................................................................................... 61
      2.6.2 American Association of Physicists in Medicine (AAPM) Distortion Definition ......................................................................................................................... 62
      2.6.3 Annular Distortion Assessment .................................................... 63
      2.6.4 Distortion Magnitude Distribution ............................................... 65
      2.6.5 Off Axis Distortion Assessment .................................................. 67
      2.6.6 Off Axis Dependence of Absolute Distortion and AAPM Percentage Distortion ......................................................................................................................... 70
      2.6.7 Off Axis Dependence of Annular Distortion .................................. 71
      2.6.8 Distortion Magnitude Distribution Dependence on Off Axis Position ................................................................................................................................. 73
      2.6.9 X and Y Components of Distortion and Variation with Off Axis Position ................................................................................................................................. 74
      2.6.10 Effect of Interchanging the Phase Encode and the Readout Directions ...................................................................................................................................... 74
2.6.11 Distortion Due to the Phantom Position in the Magnet Bore ...... 77
2.6.12 Temporal Stability of Observed Distortion .................... 78
2.7 Z Distortion Assessment ........................................ 81
2.8 Conclusions ..................................................... 85

3. Chapter 3: RAH MRI Machine Dependent Distortion Correction ...... 88
3.1 Introduction ...................................................... 88
3.2 Brief Theory of Image Transformation .............................. 88
3.3 Examples of Distortion Corrections Successfully Applied to Medical Images 93
3.4 Method Adopted for Distortion Correction at the RAH .............. 97
3.5 Distorted Image Resampling Coordinate Program (refer to appendix F) ................................................................. 99
3.6 Distorted Image Resampling Coordinate Program Results Analysis .. 101
3.7 Distortion Correction (Resampling) Program (refer to appendix G) .. 109
3.8 Results of the Distortion Correction (Resampling) Program ............. 111
3.8.1 3-D Plot of Corrected Distortion Magnitude and Associated Cartesian Coordinates .................................................. 114
3.8.2 AAPM Corrected Image Residual Distortion Assessment .......... 116
3.8.3 Corrected Image Annular Distortion Assessment .................. 117
3.8.4 Corrected Image Distortion Magnitude Distribution .................. 117
3.8.5 Corrected Image Off Axis Dependence of Absolute Distortion and AAPM Percentage Distortion ........................................ 118
3.8.6 Corrected Image Off Axis Dependence of Annular Distortion ........ 119
3.8.7 Corrected Spatial Linearity Distortion Magnitude Distribution Dependence on Off Axis Position ..................................... 121
3.8.8 Conclusions ..................................................... 122
List of Figures

Figure 1.1: Superconducting MRI System. Coordinate axes are also shown in the diagram which will be the convention assumed in this thesis. (Sprawls & Bronskill, 1992) . . . 5

Figure 1.2: Schematic representation of the y and z gradient coils. (Sprawls & Bronskill, 1992)

Figure 1.3: Origins of the Nuclear Magnetic Dipole Moment ........................................ 7

Figure 1.4: Precession of the magnetic dipole moment (µ) when placed in magnetic field \( H_0 \).

...................................................................................................................................... 10

Figure 1.5: Individual µ’s summing together to produce resultant \( M \) .......................... 11

Figure 1.6: Frame of reference rotating at the Larmor frequency (rotation is about the z axis) showing the effect on \( M \) of the field \( H_1 \). ................................................................. 12

Figure 1.7: Free induction decay of the NMR signal. The frequency of the signal is the Larmor frequency, and its initial amplitude in conjunction with successive 90° excitations can be used for \( T_1 \) assessment. \( T_2^* \) is discussed below (Horowitz, 1989). ......................... 15

Figure 1.8: \( T_1 \) assessment for two samples X (short \( T_1 \)) and Y (long \( T_1 \)) (Horowitz, 1989).

...................................................................................................................................... 16

Figure 1.9: The principle of the formation of a spin echo. F represents faster precessing spins (µ’s), and S represents slower ones. This is a laboratory (fixed) coordinate system.

...................................................................................................................................... 16

Figure 1.10: Pulse timing diagram for the spin echo sequence. ................................. 17

Figure 1.11: Pulse sequence for spin-warp imaging. TE is the time of echo signal production. ................................. 19

Figure 1.12: Relationships between \( M_{xy} \) and \( M'_{xy} \). (Sprawls & Bronskill, 1992). ........ 26

Figure 1.13: Overlay of two image outlines acquired from an actual patient on RAH CT and MRI scanners. ................................................................. 30
Figure 1.14: Assume tumour is seen on MRI scan but not on CT image. .................. 30

Figure 1.15: The tumour needs to be displaced to the left on the CT image to get edge correlation. .......................................................... 31

Figure 1.16: If a planning target volume was marked with a 1.0 cm boundary on the uncorrected MRI image, then a geometric miss would occur. ..................................................... 31

Figure 1.17: Schematic of image fusion process for use in stereotactic brain work. Note that the fiducial markers are from the CT scan data and are geometrically correct relative to the head outline. The MRI outline is then aligned to the CT outline, but, due to local spatial distortions in the MRI scan the relative position of the tumour may not be correct. .......................................................... 33

Figure 2.1: One of 11 spatial linearity plates. Each plate has 193 x 6.0 mm diameter holes with centres spaced 28 mm apart. .......................................................... 38

Figure 2.2: Spatial linearity phantom: clamping system. Each plate has three tie rods passing through each of the 10 mm holes and is separated from neighbouring plates by the spacers. The three tie rods are then tensioned with the nuts, which each bear on a washer to avoid plate damage. .......................................................... 39

Figure 2.3: Cross section of the outer case of the spatial linearity phantom. .............. 40

Figure 2.4: Spatial linearity phantom: Tank end plate. ........................................ 40

Figure 2.5: Photograph of the side view of the spatial linearity phantom. ............... 41

Figure 2.6: Photograph of the front view of the spatial linearity phantom. The precision in the plate location and the hole drilling is evident. ........................................ 41

Figure 2.7: Initial scan of RAH spatial linearity phantom. ................................... 43

Figure 2.8: Structure of a sample of the pixel information from the image of figure 2.7. .. 47

Figure 2.9: Structure of 9 peaks near the centre of the spatial linearity plate. .......... 48

Figure 2.10: Detail of the structure of a single peak in the spatial linearity image. ...... 49

Figure 2.11: Graph showing the projection of the peak from figure 2.10 onto the x-axis. The
pixel magnitude data has had a cubic spline fitted to it. The vertical dashed line represents the centre of gravity x-coordinate in fractional pixel units computed using equation 2.1. 55

**Figure 2.12:** Graph showing the projection of the peak from figure 2.10 onto the y-axis. The pixel magnitude data has had a cubic spline fitted to it. The vertical dashed line represents the centre of gravity y-coordinate in fractional pixel units computed using equation 2.2. 56

**Figure 2.13:** Scatter plot of the image data spatial linearity points for the image of figure 2.7 found with the peak search algorithm (this is the phantom’s central spatial linearity plate and is located at the magnet isocentre). Superimposed on the same axes are the coordinates (appropriately rotated) of the actual holes in the spatial linearity plate. 59

**Figure 2.14:** 3-D scatter plot of all spatial linearity points, with their respective distortion magnitudes, as found in the image of figure 2.7 (phantom’s central spatial linearity plate positioned at the magnet isocentre). 60

**Figure 2.15:** Maximum distortion detected within origin-concentric annuli of width 40 mm and with mean position 20, 60, 100, 140, 180 and 220 mm from the origin. 64

**Figure 2.16:** Frequency distribution of the percentage of the total number of spatial linearity points found in image 2.7 in 0.25 mm wide scoring bins of absolute value of distortion magnitude. 66

**Figure 2.17:** Scatter plot of the image and actual spatial linearity points for a position 25 mm off axis. 68

**Figure 2.18:** Scatter plot of the image and actual spatial linearity points for a position 50 mm off axis. 68

**Figure 2.19:** Scatter plot of the image and actual spatial linearity points for a position 75 mm off axis. 69
Figure 2.20: Scatter plot of the image and actual spatial linearity points for a position 125 mm off axis. .......................................................... 69

Figure 2.21: Annular distortion variation with off axis scanning slice. .................. 71

Figure 2.22: Off axis variation of distortion frequency distribution (0.25 mm wide scoring bins). .......................................................... 72

Figure 2.23: Bar graph showing the relative magnitudes of the mean x and y distortion as a function of off axis distance. Error bars represent the standard errors on the mean values of the distortions present at each off axis position studied. .................................. 73

Figure 2.24: Overlaid images of the central phantom spatial linearity plate. Black points are from an image with the phase encode direction along the y-axis, the blue points are for an image (of the same slice) but with the phase encode direction along the x-axis. ............................................................................. 76

Figure 2.25: Spatial linearity point positions for the phantom’s centre plate at the isocentre and for the phantom’s end plate at the same physical position within the magnet. .... 77

Figure 2.26: Scatter plot showing central slice spatial linearity stability over a 4.5 month period. .......................................................... 78

Figure 2.27: Scatter plot showing 100 mm off axis spatial linearity stability over a 4.5 month period. .......................................................... 79

Figure 2.28: Coronal MRI scan of the spatial linearity phantom showing the positioning of the desired transverse MRI scans, centred on each spatial linearity plate. ............. 81

Figure 2.29: Relationship of the image perceived slice z-position versus the actual z-position. ............................................................................. 83

Figure 2.30: Z slice position error magnitudes as a function of the actual position of the scanned slice within the magnet bore. .......................................................... 84

Figure 3.1: Summary of the various classes of transformation that map the coordinate system of one image onto another i.e. \((x,y)\)-(\(x',y'\)). (Van den Elsen et al, 1993) ............... 89
Figure 3.2: Method of Boone (Boone et al, 1991) applied to the RAH distorted image data, in order to perform warp corrections. ........................................... 97

Figure 3.3: Central image resampling coordinate x-position difference dependence on pixel location in $I_{\text{corrected}}$ .......................................................... 101

Figure 3.4: Central image resampling coordinate y-position difference dependence on pixel location in $I_{\text{corrected}}$ .......................................................... 102

Figure 3.5: Top centre image resampling coordinate x-position difference dependence on pixel location in $I_{\text{corrected}}$ .......................................................... 103

Figure 3.6: Top centre image resampling coordinate y-position difference dependence on pixel location in $I_{\text{corrected}}$ .......................................................... 105

Figure 3.7: Resampling coordinate x-position difference dependence on pixel position along the x-axis in $I_{\text{corrected}}$ .......................................................... 106

Figure 3.8: Resampling coordinate x-position difference dependence on pixel position along the y-axis of $I_{\text{corrected}}$ .......................................................... 107

Figure 3.9: Resampling coordinate y-position difference dependence on pixel position along the x-axis in $I_{\text{corrected}}$ .......................................................... 108

Figure 3.10: Resampling coordinate y-position difference dependence on pixel position along the y-axis of $I_{\text{corrected}}$ .......................................................... 109

Figure 3.11: Uncorrected patient image with an overlaid grid pattern to enable visualisation of the distortion correction process. ........................................... 111

Figure 3.12: Corrected patient image (image of 3.11 after the program resample.c was applied) with the grid showing the nature of the resulting distortion correction. .......... 112

Figure 3.13: Resampled image of figure 2.7 yielding a geometrically correct one. .......... 113

Figure 3.14: Scatter plot of the geometrically corrected central phantom spatial linearity coordinates (from figure 3.13) with the spatial linearity coordinates of the phantom. .......................................................... 114
Figure 3.15: 3-D scatter plot of all spatial linearity points in the corrected image (figure 3.13) with their respective residual distortion magnitudes. ........................................ 115

Figure 3.16: Corrected vs uncorrected maximum distortion detected within origin-concentric annuli of width 40 mm and with mean position 20, 60, 100, 140, 180 and 220 mm from the origin. ................................................................. 116

Figure 3.17: Frequency distributions, for the uncorrected image of figure 2.7 and the corrected image of figure 3.13, of the percentage of the total number of spatial linearity points having distortion magnitudes corresponding to 0.25 mm wide scoring bins. .... 118

Figure 3.18: Corrected spatial linearity image annular distortion variation with off axis scanning slice. ................................................................. 120

Figure 3.19: Corrected images: Off axis variation of distortion frequency distribution (0.25 mm wide scoring bins). ........................................... 121

Figure 4.1: Locally produced fiducial marker plate. .................................................. 125

Figure 4.2: Description of the use of the RAH fiducial marker plate. The value of $d_1/d_2$ is calculated for each CT and MRI slice to determine relative transverse slice offset. .................................................. 126

Figure 4.3: CT transverse slice of patient. Note fiducial plate and position of catheters. ................................................................. 127

Figure 4.4: Graph of the ratio $d_1/d_2$ versus nominal slice position for a patient scanned on CT with the fiducial marker plate in position. Data was fitted with a second order polynomial. .................................................. 129

Figure 4.5: Plot of the distance $d_2$ as measured from the CT scanner’s display console (actual dimension (mm)) versus the nominal slice position. .................................................. 130

Figure 4.6: Patient T1 weighted MRI scan (TR = 700 ms, TE = 15 ms). ..................... 133

Figure 4.7: Same scan as 4.6 which has had machine dependent geometric distortions removed.
Figure 4.8: Graph of the ratio \( d_1/d_2 \) versus nominal slice position for the same patient scanned on both CT and MRI. Both data sets were fitted with a second order polynomial. The MRI data set was offset by 9.0 mm from the nominal value which was read from the scans to give coincidence with the CT curve. ............................................. 135

Figure 4.9: CT slice of the patient through the kidney region. ............................................. 136

Figure 4.10: Corrected MRI scan of the same plane of the patient in figure 4.9. ............. 137

Figure 4.11: Resultant overlay of the CT and MRI slices from the same plane of the patient.

................................................................. 138

Figure 4.12: MRI slice which is coincident to the CT slice of figure 4.3. ...................... 139

Figure 4.13: Resultant overlay of CT (figure 4.3) and MRI (figure 4.12) slice outlines. Again the CT outlines are solid and the MRI ones are dashed. ............................................. 139

Figure 4.14: Position of a 420 mm x 260 mm “patient” placed on a flat couch insert for radiotherapy scanning in relation to the spatial linearity points which describe loco-regional distortions. ................................................................. 140
Abstract

The history of magnetism which forms the basis for contemporary magnetic resonance imaging (MRI) can be traced back some 4000 years. MRI is now accepted as a powerful diagnostic tool in medicine, and also is playing an ever increasing role in the detection and delineation of neoplastic and benign disease. Stereotactic neurosurgical techniques (including radiosurgery) and radiotherapy rely on accurate definition of tumour to ensure optimal patient treatment and hence prognosis.

There is still a heavy reliance on computed tomography (CT) x-ray imaging (especially when ionising radiation dose calculations are performed which use CT derived electron density information) but MRI is often able to offer unique target information not seen on CT. MRI suffers from an inherent problem which results in some degree of both machine dependent and non-machine dependent geometric distortion.

This thesis provides several methods for assessment of total machine dependent distortion which allows the clinical significance of that distortion to be established. This is a vital step before attempts are made to use any MRI information when the geometric integrity of the data will have a direct impact on the success of a treatment procedure. Further a method for removal of machine dependent distortions is presented, and shown to work for a locally produced, large field of view spatial linearity phantom.

Finally, a possible method is developed which is suitable for application of the distortion correction method to actual patients. This method was piloted on one patient who was undergoing radiotherapy treatment, and who had both CT and MRI scans as part of their planning procedures.
Statement of Authenticity

This work contains no material which has been accepted for the award of any other degree or diploma in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text.

I give consent to this copy of my thesis, when deposited in the University Library, being available for loan and photocopying.

Signed:_________________________ 8 August, 1997

Wayne A. Beckham
Acknowledgments

It is with great pleasure that I acknowledge the support of my boss, supervisor and friend, Assoc. Prof. Alun Beddoe. For without his encouragement and support I would not have been able to come to Adelaide to embark on this Ph.D. study.

I would also like to thank Dr. Peter Hoban for his friendship and always being there as a sounding board for my new ideas. His depth of understanding is remarkable, and has been of great help, especially with regard to suggestions for the write-up.

I am also indebted to my other colleagues and friends who were part of the Medical Physics and Radiation Oncology Departments at the Royal Adelaide Hospital for assisting in making my stay in Adelaide very special, and indeed, I have taken many pleasant memories with me as a result.

The assistance of Mr. Greg Brown, Senior Radiographer in charge of the MRI unit at the Royal Adelaide Hospital is also gratefully acknowledged.

I would also like to take this opportunity to recognise the support of the Anti-Cancer Foundation of the Universities of South Australia who provided funding for the construction and MRI scanning of the spatial linearity phantom.

Finally, I would like to record my gratitude to my wife Sue, who put up with me during the write-up, and my young sons Rhys and Connor, who saw a lot less of their dad than they should have during this period.
1. Chapter 1: Theory of MRI and its Application to Imaging

1.1 Introduction

Contemporary radiotherapy treatment involves the accurate delivery of radiation beams (x-rays or electrons generally) in order to maximise radiation absorbed doses to defined clinical target volumes, and at the same time minimising absorbed doses to tissues which are external to these volumes. The definition of the clinical target volume as defined by the International Commission on Radiation Units and Measurements (ICRU, 1993) is as follows: “The Clinical Target Volume (CTV) is a tissue volume that contains a demonstrable Gross Tumour Volume (GTV) and/or subclinical microscopic malignant disease, which has to be eliminated. This volume thus has to be treated adequately in order to achieve the aim of therapy, cure or palliation.” So under ideal circumstances one would attempt to deliver a uniformly distributed absorbed dose prescription to the CTV and zero dose outside this volume.

The ICRU, in the same report, also defines the Planning Target Volume: “The Planning Target Volume is a geometrical concept, and it is defined to select appropriate beam sizes and beam arrangements, taking into consideration the net effect of all the possible geometrical variations, in order to ensure that the prescribed dose is actually absorbed in the CTV.” The geometrical variations referred to in this definition result from reproducibility problems due to anatomical and technical factors in the treatment set up. An analysis of the magnitude of individual factors has been carried out by Goitein (Goitein, 1985).

Definition of the CTV involves the use of diagnostic imaging facilities, most commonly computed tomography (CT). CT alone however, is not always capable of complete definition of diseased tissue. An image modality such as Magnetic Resonance Imaging (MRI) is often able
to provide better tumour definition and so it should be considered for use in Radiotherapy Treatment Planning. Furthermore, there is also a significant function for MRI in some stereotactic neurosurgical procedures. One of the main problems associated with the use of MRI is that the images produced are to some extent spatially distorted (Fraass, 1987). A further limitation relates to the fact that MRI offers no information on relative electron density (easily obtainable from CT) which is necessary for the calculation of radiation dose distributions. A viable solution for MRI in radiotherapy treatment planning will therefore involve combining MRI and CT data. The principal aim of this study is to ascertain the efficacy of MRI for use clinically where geometric distortion is an important consideration.

1.2 Historical Evolution of Magnetism, Nuclear Magnetic Resonance and Magnetic Resonance Imaging

The following text is a summary of an excellent review, (Mourino, 1991) and serves to outline the key discoveries which led to the ability to produce magnetic resonance images. It was around 2000 BC that the discovery was made in the Western world of magnetite. Allegedly this was by a shepherd bearing the name of Magnes who found that the tacks in his sandals were drawn towards the earth. Upon digging to find the source of the attraction he uncovered the mineral magnetite or lodestone - a magnetic oxide of iron (Fe₃O₄). Around 500-300 BC it was postulated that the phenomenon of the lodestone was the same as that of amber rods when they were rubbed with fur, to attract pieces of paper etc. (ie. electrostatics). This belief continued for the next 2000 years or so. The lodestone was the subject of much mystique, superstition and even consideration that it may be useful in medicine. It was not until the year 1550 that the Italian mathematician Girolamo Cardano produced evidence that the lodestone attraction was a unique phenomenon, and was fundamentally different from electrostatics.
Electromagnetism was not explicitly discovered until, in 1820 or so, Oersted noticed that a magnet exerted a force on a current carrying wire and vice-versa. This provided the catalyst for a significant increase in research activity into the area of magnetism. Some of the key players were André Marie Ampère, Jean-Baptiste Biot and Felix Savart whose achievements are well known to all students of physics. The laws of classical electromagnetism were brilliantly summarised in 1865 by James Clerk Maxwell which resulted in the now well used electromagnetic theory of light.

Pieter Zeeman published the result (Zeeman, 1897) that magnetic fields had an effect on the electronic energy levels within an atom (Zeeman effect). This led to speculation that the atoms had associated magnetic moments. Classical theory at that time could not explain the phenomenon. Quantum mechanical theory (developed by Bohr and later Dirac) was to provide the solution and was supported by the experimental work of Otto Stern and Walter Gerlach (Gerlach & Stern, 1924). The result obtained was that the electrons of an atom had associated spins and orbital motion (which resulted in a tiny magnetic field), and that the orientation of the spins could be either parallel or antiparallel to an applied external magnetic field.

In 1924 Wolfgang Pauli suggested that odd-even and odd-odd atomic nuclei possess angular momentum (also called nuclear spin) which could explain the hyper-fine structure of atomic spectra. The nuclear spin resulted in a nuclear magnetic dipole moment. This dipole moment is fundamentally important for the nuclear magnetic resonance (NMR) effect, which is the underlying process of modern magnetic resonance imaging. Nuclear magnetic resonance was first measured by Isidor Rabi and co-workers in 1936. NMR in solids was independently reported in 1946 by two groups, one headed by Felix Bloch at Stanford University the other by Edward Purcell at Harvard University.
This discovery spawned a great deal of research work with NMR some of which involved studying NMR properties of various biological samples in the 1950's. An important discovery for medical physics was made by Raymond Damadian (Damadian, 1971) that NMR had the ability to discriminate between normal and malignant tissues. The ability to spatially discriminate NMR signals from a sample was first conceived and actually achieved by Paul Lauterbur (Lauterbur, 1973). This was the foundation of MRI as we know it today and within one year of Lauterbur’s work four separate groups had used four different methods to produce images.

1.3 MRI Components

In the following discussion, the values mentioned are typical for the MRI system employed at the Royal Adelaide Hospital and were derived from a comprehensive survey of MRI systems which was published in 1985 (Bradley, 1985). There are several principal components that are necessary to produce a magnetic resonance image as follows:

i) Main Magnet: This is usually of solenoid geometry with an air core of around 1 metre in diameter. Field strengths range (for superconducting coils) from 0.5 to 2.0 Tesla (5,000 - 20,000 Gauss) typically, and provide field uniformity of between 3-10 parts per million over a 40 cm sphere centred in the main magnet bore. Uniformity is achieved by the use of shim coils which can superimpose their effects on the main magnetic field. The magnetic field produced by the main magnet is designated $H_0$.1 A schematic of a superconducting MRI system is shown in figure 1.1. Proton spins are aligned parallel or antiparallel to the direction of $H_0$ producing a longitudinal magnetisation.

---

1Values in bold represent vectors, the same character in normal font is the modulus of that vector
ii) RF Transmitter: This delivers up to 15 kW of power to produce radiofrequency magnetic fields (designated $H_i$) via transmitter coils which are used in sample excitation. The transmitter coils may also act as receiver coils. Body coils are positioned within the main magnet coils to enable the entire body to be passed through their influence. Head coils may also be used which fit closely to the head dimensions for more efficient transmission/receiving from the subject. Some systems also employ surface coils of varying dimensions to enable localised imaging (e.g., orbit studies). The RF fields flip spins through a desired angle, primarily to produce a transverse magnetisation component which can be detected by the receiver coils.

iii) Magnetic Gradient System: This sub-system consists of a series of power supplies and coils which are used to superimpose time-varying magnetic fields onto the static field produced by the main magnet. Magnetic gradients produced can be up to a maximum value of around 1 Gauss per centimetre (typically 0.1-0.5 G/cm are used clinically). Rise times are typically 1-2 ms (modern systems can now have gradient strengths up to 2.5 G/cm and rise times of the order
of 300 µsec). Gradient linearity is usually better than 3%. The gradient system is capable of producing time-varying magnetic gradients in three dimensions. Figure 1.2 shows schematically how the y (saddle geometry) and z (Maxwell pair configuration) gradient coils are arranged. Note that in the figure the x gradient coils are not shown for clarity, but are also of saddle design. Magnetic gradients are used mainly to localise the origins of the RF signals.

![Schematic representation of the y and z gradient coils.](image)

**Figure 1.2**: Schematic representation of the y and z gradient coils. (Sprawls & Bronskill, 1992)

iv) Detection System: This consists of receiver coils (can be the same coils used for RF transmission in ii) above) connected to a radiofrequency receiver. The detection system results in the complex output signal $S(t)$.

v) Image Processing: This includes the computer which reconstructs and displays the resultant images (usually this computer also controls the other sub-systems already dealt with). The computer also performs the fundamental task of converting the output signal $S(t)$ to an image by computing the Fourier transform of $S(t)$.  

---

6
1.4 Nuclear Magnetic Resonance Model

This discussion will be limited to proton NMR, and as the reader will appreciate later most modern imaging is still proton based (i.e., using the nuclei of hydrogen atoms), with the exception of spectroscopic techniques which involve other nuclei.

Fundamental to the NMR phenomenon is the fact that (as was stated earlier) protons possess a magnetic moment \( \mu \), and can be thought of as spinning on an axis as in figure 1.3.

![Spinning Proton and Magnetic Dipole Moment](image)

**Figure 1.3:** Origins of the Nuclear Magnetic Dipole Moment

The magnetic moment \( \mu \) can be represented by the following equation:

\[
\mu = \gamma \frac{h}{2\pi} J
\]  

(1.1)

In this case \( \gamma \) is the gyromagnetic ratio and is a physical property of the nucleus giving the rate
of "precession" (see later) as a function of magnetic field strength. It gives the possibility of nuclear species selectivity (as in NMR spectroscopy) because different chemical elements (even different isotopes of the same element) possess quite different values of $\gamma$. For protons the gyromagnetic ratio has the value $4.26 \times 10^7$ Hz T$^{-1}$ (or $2.68 \times 10^8$ rad s$^{-1}$ T$^{-1}$). $J$ represents the spin angular momentum of the nucleus.

We now introduce this proton to a uniform magnetic field $H_0$, which is parallel to the z axis. The component of the proton's angular momentum in the z direction ($J_z$) is quantized, and for a proton can only have two values, namely $J_z = \frac{1}{2}$ or $J_z = -\frac{1}{2}$.

The potential energy ($E_p$) for the system under these circumstances obtained by taking the dot product of the nuclear magnetic dipole moment and the applied magnetic field viz:

$$E_p = -\mu \cdot H_0 \quad (1.2)$$

The negative sign in equation 1.2 indicates that the potential energy is maximised when the two vectors are anti-parallel (or when $J_z = -\frac{1}{2}$, ie $E_p$ is positive). This is important because it means that the lowest energy state is when the spins are parallel to the applied magnetic field. For equations 1.1 and 1.2, the possible values of $E_p$ are $\pm \frac{1}{2} \frac{\gamma}{2\pi} \frac{h}{2\pi} H_0$, and $\frac{\gamma}{2\pi} \frac{h}{2\pi} H_0$ is the difference in energy between the states.

1.4.1 Excitation

We are able to add energy to the system to cause transitions between these states by an external radiofrequency excitation. This can be expressed as:
\[ E_{\text{ext}} = hf = \frac{h}{2\pi} \omega \quad (1.3) \]

\[ \omega_0 = \gamma H_0 \quad (1.4) \]

If we want to induce a transition in the system from the lower energy state \((J_z = \frac{1}{2})\) to the higher energy state \((J_z = -\frac{1}{2})\) then we need to supply an amount of energy \(E_{\text{app}}\) which is equal to the difference in \(E_o\) between the two states which is \(\gamma \frac{h}{2\pi} H_0\). So we must have

\[ \frac{h}{2\pi} \omega = \gamma \frac{h}{2\pi} H_0 \]

which leads to the result:

Equation 1.4 is called the Larmor formula. Energy of angular frequency \(\omega_0\) (called the Larmor frequency) can be supplied by way of an alternating electro-magnetic field \((H_t)\) applied perpendicular to the static field \(H_0\) \((H_t\) lies in the \(xy\) plane).

The magnetic dipole moment \(\mu\) in the static magnetic field \(H_0\) experiences a torque \((\Gamma)\) which is given by:

\[ \Gamma = \mu \times H_0 \quad (1.5) \]

Since the nucleus is rotating, the torque acts to change the angular momentum of the system which results in precessional motion about the \(z\) axis as shown in figure 1.4.

The angular frequency of this precession is equal to the product \(\gamma H_0\) (from the definition of \(\gamma\)).
Precessional motion
of $\mu$

$H_0$

Fixed axis of rotation

**Figure 1.4:** Precession of the magnetic dipole moment ($\mu$) when placed in magnetic field $H_0$.

We can further write using equation 1.1 and the fact that the rate of change of angular momentum for the system is proportional to the torque acting on it:

$$\frac{d\mu}{dt} = \gamma \mu \times H_0$$  \hspace{1cm} (1.6)

In the case of a finite volume containing $i$ protons placed in a uniform magnetic field $H_0$, in which all of the individual $\mu$'s sum to give a resultant magnetisation $M$, equation 1.6 can be rewritten thus:

$$\frac{dM}{dt} = \gamma M \times H_0$$  \hspace{1cm} (1.7)

Under equilibrium conditions the vector $M$ is parallel to the $z$ axis and the component of $M$ in the $xy$ plane, $M_{xy}$, is equal to zero because the phases of the individual $\mu$'s are random, even though individual $\mu$'s have oscillating $x$ and $y$ components (and constant $z$ components). This is illustrated in figure 1.5.
Note that in the absence of any excitation by the field $H_1$, that there are a few more spins (1-3 per million) parallel to the z axis than there are anti-parallel ones. This is because the parallel orientation is the lowest energy state.

Upon excitation of the sample of protons with the rotating magnetic field $H_1$, applied at angular frequency $\gamma H_0$, all the individual nuclear magnetic dipole moments are forced to precess in phase (the resonance part of MRI), and further to this more and more spins are flipped into the anti-parallel higher quantum energy state, which effectively causes the vector $M$ to nutate away from it's rest state. This resultant transverse magnetisation (due to non-cancellation of $\mu_{xy}$'s) is time dependent (ie. $M_{xy} \neq 0$) and will induce a sinusoidally varying signal ($S(t)$) in a coil whose axis is placed in the xy plane. It is clear that $S(t)$ is dependent on $M$ (and hence on the number of protons spinning), and this relationship will be discussed later on. The $M_z$ (longitudinal) component decreases due to spins flipping into the antiparallel state and cancelling some of those in the parallel state. At the same time, due to the spins now precessing in phase, there is an oscillating transverse component $M_{xy}(t)$. This transverse magnetisation is not the direct result of decreasing the longitudinal magnetisation, though they go together during the RF excitation due to i) flipping and ii) in-phase precession. These are independent, however, and transverse magnetisation can completely disappear, whilst longitudinal does not increase much during
relaxation (following cessation of the RF excitation) which is further described in section 1.4.2.

If we now adopt a rotating coordinate frame the nutation of the magnetisation $M$ is simply described. Such a rotating frame is shown in figure 1.6.

During excitation, in the rotating frame system, $M$ simply tilts down into the xy plane. If half of the (excess) parallel spins flip to the antiparallel state, the $M_z$ component disappears, and if there is some transverse magnetisation there is then a 90° tilt. If all excess parallel spins flip, there is a 180° tilt of $M_z$. In the example shown in figure 1.6, the angle of tilt is 90°, and thus the radiofrequency (rf) pulse that causes $H_1$ to tilt $M$ by this amount is termed a 90 degree pulse. By applying the rf pulse for a longer period of time (or at a higher magnitude for the same time) it is possible to tilt $M$ through 180°.

1.4.2 Relaxation

Following excitation by an applied rf pulse the sample will eventually return to thermal equilibrium conditions (the lowest energy state). There are two principal phenomena which contribute to the relaxation process:

![Figure 1.6: Frame of reference rotating at the Larmor frequency (rotation is about the z axis) showing the effect on $M$ of the field $H_1$.](image)
The first is the individual spins re-aligning themselves with \( H_0 \), which results in the gradual recovery of \( M_x \). This is termed **longitudinal** or **spin-lattice** relaxation. Physically energy is transferred from the nuclear spin system to the surrounding atomic and molecular environment in a the sample. If the surrounding molecules are in a fluid environment (cf. body tissues) and the molecules are of such a size that they are subject to rotational and vibrational motion of frequency similar to the Larmor frequency then this energy transfer is most efficient, which in turn results in more rapid relaxation. This tends to be the case for larger molecules, which have slower motions than smaller molecules. For example, fat has a quicker relaxation rate than water, because water is a small molecule with re-orientation rates much greater than the Larmor frequency. The characteristic time constant of the recovery of \( M_x \) is designated \( T_1 \), and represents the time for \( M_x \) to recover to 63% \( (1 - 1/e) \) of its equilibrium value of \( M_0 \).

The other is the dephasing of the individual moments, after they were forced to precess in phase by \( H_1 \). This is called **transverse** or **spin-spin** relaxation, and governs the return of \( M_y \) to an equilibrium value of zero. The time constant associated with this spin-spin relaxation is designated \( T_2 \). Physically this dephasing is due to external applied magnetic field inhomogeneities, and in addition on the magnetic coupling which exists between neighbouring nuclei which slightly varies the precession frequencies of the protons in the sample under study. The latter effect can be cleverly isolated (see later) and is the phenomena which modulates “true” \( T_2 \) relaxation.

It is possible to add terms to equation 1.7 for \( T_1 \) and \( T_2 \) (Hinshaw & Lent, 1983) as follows:

\[
\frac{dM}{dt} = \gamma M \times H - \frac{(M_x i + M_y j)}{T_2} - \frac{(M_z - M_0)k}{T_1}
\]  

(1.8)
Where \( i, j \) and \( k \) are unit vectors pointing along the x, y and z axes respectively. Equation 1.8 is known as the Bloch equation (Bloch, 1946). This equation can be solved for the longitudinal \( (M_L) \) and the transverse (i.e. that component which leads to a measurable signal) components of magnetisation (Hinshaw & Lent, 1983).

Firstly the longitudinal:

\[
M_L(t) = M_L^0 \exp \left( -\frac{t}{T_1} \right) + M_0 \left( 1 - \exp \left( -\frac{t}{T_1} \right) \right)
\]  

(1.9)

In this case the longitudinal component decays from an initial value of \( M_L^0 \) (following application of \( H_1 \)) to its equilibrium value of \( M_0 \).

Now for the transverse:

\[
M(t) = M^0 \exp (i\omega t - \frac{t}{T_2})
\]  

(1.10)

Where: \( M^0 = M^0_\perp + iM^0_\parallel \)

The fact that different tissues exhibit different values of the relaxation times \( T_1 \) and \( T_2 \), gives the possibility for the nuclear magnetic resonance process to realise tissue differentiation. Note that this is quite different from the detection of different materials from their different Larmor frequencies as in NMR spectroscopy. Two examples will now be given to illustrate for our
sample of protons in a uniform magnetic field how one could, in principle, characterise that sample in terms of either $T_1$ or $T_2$.

For $T_1$, if we apply a 90° rf pulse we will tilt the vector $M$ completely into the xy plane, and our detected signal (called free induction decay (FID), see figure 1.7) will have an initial amplitude which will be directly proportional to $M_0$.

![Figure 1.7: Free induction decay of the NMR signal. The frequency of the signal is the Larmor frequency, and its initial amplitude in conjunction with successive 90° excitations can be used for $T_1$ assessment. $T_2^*$ is discussed below (Horowitz, 1989).](image)

If we then allow an interval of time TR to elapse (which will see $M_x$ recover to some extent) before applying a second 90° pulse, which in turn will tilt the recovered $M_x$ back into the xy plane, giving a detected signal magnitude which, when ratioed with the initial signal magnitude after the first rf pulse, gives a means of assessing $M_x$ recovery and hence $T_1$. Physically the correlation of $M_{xy}$ after the second 90° pulse to the degree of recovery of $M_0$ is due to the fact that a greater $M_x$ recovery means that there are more parallel spins that can be put into phase and hence produce a greater signal. Substances with a slow longitudinal relaxation (long $T_1$) such as water, will give a low signal after the second 90° pulse (unless a very long TR is used). Use of a short TR therefore allows differentiation of tissue types based on $T_1$ times. MRI sequences with a short TR are said to be $T_1$ weighted. Figure 1.8 shows a frame by frame analysis of such an NMR experiment carried out on two samples with different $T_1$'s.
By the time frame 4 is reached in the figure, sample X’s $M_z$ has fully recovered, whereas sample Y’s field has not, and after the second 90° pulse, sample X will exhibit a greater NMR signal than sample Y.

For $T_2$ assessment we need to first define $T_2^*$. This is shown in figure 1.7 as the positive envelope of the free induction decay curve. $T_2^*$ decay is due to spin-spin relaxation defined above, and in addition, can be due to subtle $H_0$ inhomogeneities, which are equipment dependent. It is possible to remove the latter effect and hence characterise pure $T_2$ decay by the formation of a spin echo. This is shown in figure 1.9.
At $t = 0$ a 90° rf pulse is applied to our sample which tilts $M_0$ into the xy plane ((a) in figure 1.9). Then at $t = \tau$ where partial dephasing of the $\mu$'s has occurred, and the faster precessing spins lead the slower ones, a 180° refocussing pulse is applied which produces a mirror image of the spins around the y-axis (called phase conjugation), thereby causing the slow spins to be leading the fast ones ((b) in figure 1.9). Now after the time $t = 2\tau$ has elapsed, the faster spins catch up with the slower ones which in turn will maximise the $M_y$ producing an echo (this time at which the echo occurs after the 90° pulse is known as TE, thus the 180° pulse is applied at a time TE/2). The magnitude of the echo, relative to the initial amplitude of the FID signal, is of course dependent on $T_2$, the true spin-spin relaxation time and TE: for a short TE, a small degree of spin-spin relaxation will have occurred and the echo will be large. A long TE gives a smaller echo (signal), but also, different tissues may then have undergone a vastly different degree of relaxation and there will be a very $T_2$-dependent signal. An MRI sequence with a long TE is thus said to be $T_2$ weighted. Watery fluids have a long $T_2$ and the echo will be larger than that of say fat for a given TE, therefore water gives a high signal on a $T_2$ image. Figure 1.10 shows the sequence of figure 1.9 in a pulse timing diagram. This provides a mechanism whereby $T_2$ can be assessed for different samples.

![Figure 1.10: Pulse timing diagram for the spin echo sequence.](image-url)
short TE do not differentiate tissues by either their $T_1$ or $T_2$ - only the number of protons (spins) affects the signal. Such sequences are said to be “proton density weighted”.

1.5 Localisation of the NMR Signal and Origins of System Related Distortion

Up until now the assumption has been made that our small sample of proton containing material is being studied in a uniform applied magnetic field $H_0$, allowing the NMR properties of that sample to be analysed. For a large, heterogeneous sample of protons distributed spatially, if the applied magnetic field is made non-uniform throughout the volume then protons at different positions will precess at different rates (eq. 1.4), and indeed, if $H_1$ is applied with a given frequency, then only those spins which are experiencing the correct magnetic field will be forced into resonance. This give rise to the possibility of spatial encoding, which is handled explicitly by the magnetic gradient sub-system of the imaging equipment as defined in iii) above. This sub-system is capable of producing spatially varying and time varying magnetic gradients in the x, y or z (or any combination thereof) directions (see figure 1.2, above).

It should be noted at this stage that discussion from now on will be in the context of the spin-warp image reconstruction algorithm (Edelstein et al, 1980). The reason for this is to describe the origins of distortion for the Royal Adelaide Hospital’s MRI equipment (RAH MRI) which utilises this algorithm. MRI machines exhibit non-linearities in their gradient fields, and have inhomogeneous static fields. The effects of these need to be considered as they can lead to image distortion.

The pulse sequence for spin-warp imaging is presented in figure 1.11 (based on O’Donnell & Edelstein, 1985).

---

2Siemens Magnetom 1.0 Tesla, Siemens Erlangen
1.5.1 Slice Select Gradient (z-gradient)

Initially a z-gradient (G(z)) (used for example here to excite a transverse slice) is turned on for a period, and during this time a selective 90° rf pulse is applied (in practice the rf pulse takes the form of a sinc function to enable the slice to be uniformly excited). The selective pulse excites only those spins in a slice perpendicular to the z axis where the magnetic field strength results in Larmor frequencies corresponding to the range of frequencies making up the pulse (ie. slice width is determined by the bandwidth of the rf pulse and by the gradient of the magnetic field).

The centre frequency of the pulse (ω(z)) corresponds to the z-coordinate of the centre of the slice. The relationship is described by the following equation:

\[
ω(z) = γ(H_0 + zG(z))
\]  

(1.11)

In the presence of z-gradient non-linearities and \(H_0\) inhomogeneities 1.11 can be rewritten for the new frequency \(ω'(z)\):
\[ \omega'(z) = \gamma(H_0 + \Delta H(x,y,z) + zG(z) + \Delta G(z)) \]  

(1.12)

Where \( \Delta H(x,y,z) \) is the amount of deviation of \( H_0 \) from being homogeneous, and \( \Delta G(z) \) is, for a given gradient strength, the amount that that gradient deviates from being linear. If the gradient strength \( G(z) \) is changed, then \( \Delta G(z) \) changes by the same proportion. The corollary to equation 1.12 is that the actual \( z \) position of the centre of the excited region may vary depending on the \( x,y \) position within the slice due to \( \Delta H(x,y,z) \). \( \Delta G(z) \) will also have an effect on the \( z \) position of excited spins.

The Royal Adelaide Hospital (RAH) MRI makes no allowance for \( \Delta H(x,y,z) \) or \( \Delta G(z) \), and so application of equation 1.11 leads to an erroneous position \( z' \) being assumed for the centre of the excited slice. The real position \( z \) for this, is found in equation 1.12. The relationship between \( z' \) and \( z \) is established by equating 1.11 and 1.12 yielding:

\[ z' = z + \frac{\Delta H(x,y,z) + \Delta G(z)}{G(z)} \]  

(1.13)

or:

\[ z' = z + \Delta z \]

In section 1.3 (i) and 1.3 (iii), it was seen that \( H_0 \) homogeneity of 3-10 ppm (specified for a sphere 50 cm in diameter for the RAH MRI Scanner) and \( G(z) \) linearity to 3% respectively is usually achievable. If we use the example of a 1.0 Tesla value for \( H_0 \), this implies a maximum value for \( \Delta H(x,y,z) \) of \( 10^{-5} \) T, and \( G(z) = 10^{-4} \) T cm\(^{-1} \) (typical value for \( G(z) \) for scans used later in this thesis), which in turn leads to a value for \( \Delta z \), the displacement in the \( z \) direction of 0.1 cm or 1.0 mm, due to \( H_0 \) inhomogeneity.
A gradient non-linearity of 3% (ie. $\Delta G(z)/G(z) = 0.03$ cm) would lead to a value of $\Delta z = 0.3$ mm.

Referring to figure 1.11 again one sees that $G(z)$ also comprises a negative component which is applied immediately after cessation of the 90° rf pulse. The purpose of this is to remove “phase creep” which was caused during rf excitation (centred on $t = 0$) due to the fact that the value of $H$ varied continuously in the z-direction, resulting in different spin resonance frequencies across the slice. If uncorrected, significant dephasing would occur, reducing the amplitude of the detected NMR signal.

1.5.2 Phase Encode or Preparation Gradient (y-gradient)

Phase encoding is carried out with a variable y-gradient as shown in figure 1.11. Typically 192 different values of $G(y)$ are used in the acquisition of the information necessary to produce 1 slice, ie. the whole sequence shown in figure 1.11 will be repeated 192 times. Whilst the y-gradient is being applied, spins precess at increasing frequencies along the y-axis, causing them to progressively go out of phase. The degree to which they go out of phase depends on the amplitude of the applied gradient. After application of the preparation gradient, the magnetic field becomes uniform once again (frequency is then constant), thus preserving the induced phase relationship. The phase encode gradient allows resolution of the MRI signals in the y-direction, and when this is complimented with the frequency encoded information from the readout gradient (see 1.5.3), individual voxels can be identified to ultimately produce an MRI intensity spatial map.

The phase encode direction is immune to static field inhomogeneities ($\Delta H(x,y,z)$), (O’Donnell & Edelstein, 1985, Gauvin, 1992). This is related to the fact that $\Delta H(x,y,z)$ is non-varying with the magnitude of the y-gradient, such that dephasing that results is recovered by the phase
conjugation process involved in the application of the $180^\circ$ rf pulse at time $t = \text{TE}/2$ after the $90^\circ$ pulse.

Y-gradient non-linearities because of their dependence on gradient strength, makes them non-reproducible and hence will result in some phase errors which will not be cancelled out by phase conjugation, resulting in distortions.

1.5.3 Frequency Encode or Read-out Gradient (x-gradient)

Once again referring to figure 1.11, one sees that a pulsed x-gradient of fixed amplitude is applied at the same time as the phase encode gradient. This is called the pre-phase lobe of the readout gradient. The purpose of this pre-phase lobe is to counter the effects of dephasing due to the readout gradient proper, which is to follow. The effects are exactly countered if the time integral of $G(x)$ equals zero at time $t = \text{TE}$, when all spins come into phase, though they have a different frequency.

Assume for now that we have not applied any phase encode gradient. The readout gradient is applied with a fixed amplitude and over a period of time long enough to cover the duration of the signal detected. Clearly, the received signal ($s_{\text{det}}(t)$) will be made up of many frequencies depending on the position that the individual spins are along the x-axis within our excited slice. This is described by equation 1.14:

$$\omega(x) = \gamma(H_0 + xG(x))$$  \hspace{1cm} (1.14)

Each frequency component of $s_{\text{det}}(t)$ corresponds to the summed contribution of transverse magnetisation ($M_{xy}$) along a strip (in the y-direction) of our excited slice with a particular x
coordinate. The receiver coil sees all of the frequencies present in $s_\text{det}(t)$, and the signal is an integral over x and is explicitly given by:

$$s_\text{det}(t) = e^{-iH_0 t} \int M_{xy}(x) e^{-i\gamma x G_z t} dx$$  \hspace{1cm} (1.15)$$

The part of equation 1.15 that is dependent on $H_0$, describes the carrier component of $s_\text{det}(t)$, and in MRI systems this carrier is removed (also called demodulation) by a quadrature phase detector (which involves combining $s_\text{det}(t)$ with a reference signal of frequency equal to the carrier) to reduce the sampling demands on the analogue to digital conversion (ADC) circuitry. So with the carrier removed, which results in $s(t)$, equation 1.15 becomes:

$$s(t) = \int M_{xy}(x) e^{-i\gamma x G_z t} dx$$  \hspace{1cm} (1.16)$$

In order to retrieve $M_{xy}$, the MRI system takes the Fourier transform of equation 1.16 (using the fast Fourier transform (FFT) (Bracewell, 1978)) which results in:

$$M_{xy}(x) = \int s(t) e^{i\gamma x G_z t} dt$$  \hspace{1cm} (1.17)$$

The effects of x-gradient non-linearities can be shown in a similar way as was done for the slice select gradient $G_z$. By analogous derivation equation 1.12 (in the context of the x-gradient) becomes:

$$\omega'(x) = \gamma(H_0 + \Delta H(x,y,z) + xG(x) + \Delta G(x))$$  \hspace{1cm} (1.18)$$

Again similarly, equation 1.13 for the x-gradient becomes:
\[ x' = x + \frac{\Delta H(x,y,z) + \Delta G(x)}{G(x)} \] (1.19)

Where as before: \[ x' = x + \Delta x \]

Once again the distortions as a result of the x displacements can be due to gradient field non-linearity and \( H_0 \) inhomogeneities.

Other workers (Ehricke & Schad, 1992) assessed the magnitude of this type of machine dependent displacement in the transverse plane, and found maximum values of 3-4 mm in a 300 mm field of view using a phantom filled with a rectangular grid of pipes filled with MR contrast medium. Further, this work also showed that, for a given acquisition protocol, the spatial distortion pattern was very stable, giving rise to the possibility of correcting for it (to less than 1.0 mm) in clinical images, based on previous phantom measurements. It should be noted at this point that the magnitude of machine dependent distortion depends on the MRI system condition (eg. the main magnetic field shimming state) and will vary from machine to machine. The values predicted earlier appear to be smaller than Ehricke & Schad’s work, and this could be due to the fact that the calculations assumed typical values for \( H_0 \) inhomogeneities and gradient field non-linearities, whereas, actual values may exceed these and therefore distortion magnitudes will increase. It should also be noted that this thesis made no attempt to measure directly \( H_0 \) inhomogeneities and gradient field non-linearities, rather their combinatorial net effect on geometric distortion magnitude.
1.5.4 Final Image Reconstruction

Consider a slice which has been excited and a specific y-gradient (phase encode) applied for a time T with a particular amplitude $G(y)$. An x-gradient (read-out) is then applied, and a demodulated signal $s(t)$ is detected and can be represented by:

$$s(t) = \int \int M_{xy}(x,y) e^{-iyG_yT} e^{-iyG_yT} dx dy$$  \hspace{1cm} (1.20)

For convenience one can define the spatial frequency variables $k_x$ and $k_y$, where:

$$k_x = \left(\frac{y}{2\pi}\right) G_x T \hspace{1cm} (1.21)$$

$$k_y = \left(\frac{y}{2\pi}\right) G_y T \hspace{1cm} (1.22)$$

It is evident from equation 1.20 that $s(t)$, for a particular phase encode gradient strength, is equivalent to a sample of the spatial Fourier transformation of $M_{xy}(x,y)$, where the complete spatial Fourier transformation is denoted $M_{xy}(k_x,k_y)$.

The relationships between $M_{xy}(x,y)$, $M_{xy}(k_x,k_y)$, $k_x$ and $k_y$ are shown diagrammatically in figure 1.12.
For a given repetition the signal $s(t)$ represents a complete line in the $k_x$ direction, the $k_y$ position of that line being determined from the area under the $G(y)$ gradient waveform. So for example the condition for no $y$-gradient results in the signal $s(t)$ coinciding with the $k_y$ axis ($s(t) = M_{xy}(k_x, 0)$). So after many repetitions, using both positive and negative $y$-gradients one gradually fills up $M_{xy}$ or “k-space” as it is sometimes called.

The interesting aspect to this type of imaging is that the signal being sampled is directly from Fourier space. The image ($M_{xy}$) is found by 2-dimensionally inverse fast Fourier transforming the resultant filled “k-space”.

1.5.5 Eddy Currents

Eddy currents can arise in conducting elements making up the MRI system and may include the cryostat, main magnet coils, gradient coils and rf coils (Henkelman & Bronskill, 1987). The cause of eddy currents is the pulsed nature of the magnetic gradients, which according to the
Lenz's law, oppose the change of the changing magnetic field and establish circular current flows. The net result of this is analogous to main magnetic field inhomogeneities in which some degree of geometric distortion results. Being a system dependent distortion, for a given image acquisition sequence, the magnitude of the distortion will be consistent from one patient to the next.

However, this will only apply if the eddy currents occur external to the imaging volume. There have been observations of distortions due to eddy currents in cases utilising stereotactic localising frames which act as a mount for fiducial marker systems to enable coordinate based neurosurgical or radiosurgical procedures to be performed. In such cases the distortion that results may not be totally independent from patient to patient due to the different possible positions of the stereotactic equipment within the main magnet bore. It may also be possible for eddy currents to be produced within the patient (or liquid filled phantom (see chapter 2)) thereby adding to distortion degree. This distortion effect would be difficult to predict and/or compensate for. No attempt was made in this work to measure this. This was justified to some extent by the fact that other workers (Ehrike and Schad, 1992, Moerland, 1996) have achieved acceptable distortion corrections without explicitly considering eddy current effects.

1.6 Origins of Sample Dependent Distortion

The previously mentioned distortions due to main magnetic field inhomogeneity and gradient field non-linearities can be classified as machine dependent distortions. A brief discussion will now be given concerning distortions associated with the type of matter that is present in the MRI system’s magnet bore (sample dependent distortions) and which is undergoing spin-warp imaging.
1.6.1 Magnetic Susceptibility and Chemical Shift Effects

The discussion so far has been involved with relating the spin precession frequency, $\omega$, to an applied magnetic field $H_0$. This applied magnetic field is altered by the magnetic susceptibility, $\chi$, of the sample under study. If there is a regional susceptibility change, this produces the equivalent of local main magnetic field inhomogeneities ($\Delta H(x,y,z)$). In human MRI studies, the worst area for this occurring is the lung to surrounding tissue interface. $\Delta \chi$ here is of the order of 5 ppm (Ehricke & Schad, 1992). Therefore for a 1.0 T value for $H_0$, $\Delta H(x,y,z)$ of $5 \times 10^{-6}$ T due to the susceptibility difference would result (Schenck, 1996). Using equation 1.19 one can calculate a value for $\Delta x$, a shift in the $x$-direction in the transverse plane of $5 \times 10^{-6} \times 1.0 \div 0.5 \times 10^{-4} = 0.1$ cm or 1.0 mm for a read-out gradient strength of $0.5 \times 10^{-4}$ T cm$^{-1}$. This figure is small enough to be ignored for spin-warp imaging at this gradient strength (typical of the values used for the imaging studies described in the ensuing chapters), but other imaging sequences are becoming more popular, for example echo planar imaging (which is a fast imaging technique), and have been reported as showing more severe distortions due to susceptibility than spin echo techniques (Moerland, 1996).

Chemical shift is another phenomena that results due to precessing spins having the magnetic field that they are experiencing altered by the shielding effect of the electron clouds that surround them, thereby slightly changing spin resonant frequencies. The degree of magnetic shielding depends on the position of the nuclei within the molecule, and indeed the molecular shape or chemical structure. A common example in the human body of this effect is the resonant frequency shift between water and fat ($\approx 3.3$ ppm). In a way that is similar to susceptibility effects above chemical shift results in an effective $H_0$ change (by causing the assumed position of a signal to be different to the actual position). The value of $\Delta H(x,y,z)$ however is once again small (ie. $3.3 \times 10^{-6}$), and therefore distortion displacements for spin-warp imaging around 0.66
mm (again for read-out gradient strengths of around $0.5 \times 10^{-4} \, \text{T cm}^{-1}$ (typical of those used for image acquisition later in this thesis)) would result in these circumstances. Again the magnitude here is relatively small and for most applications of MRI is negligible.

Knowledge of the displacements due to these two phenomena is obviously patient dependent and prediction would be extremely difficult. It is fortunate that the distortion magnitudes due to them are small in the context of spin-warp imaging, because this leaves the predictable machine dependent distortions as the dominant source, and hence offers the opportunity for correction. It should still be noted, however, that when smaller gradient strengths are used that the distortions due to susceptibility and chemical shift will change in inverse proportion and may well become significant.

1.7 Effects of Distortion of MRI in Radiotherapy Treatment Planning and in Neurosurgery

Having discussed the main causes of image distortion, it is appropriate now to consider the ramifications of it in the context of radiotherapy treatment planning (RTTP) and in neurosurgery applications.

One of the early papers which assessed the use of MRI in RTTP was published in 1987 (Frass et al, 1987) and this discussed the potential usefulness of delineating tumour volumes with MRI (where the tumour volumes were not visible in CT scanning which is now commonplace in RTTP and indeed essential for accurate radiation dosimetry calculations (Dobbs & Webb, 1990)). Frass pointed out the possible limitations to the use of MRI due to distortion especially for large fields of view which could result in a geometric miss of the tumour by the applied radiation beams and this would clearly cause a decreased probability of local tumour control and/or increased probability of complications, with disastrous effects on patient prognosis. Margins that are
placed around planning target volumes in RTTP range from 0.5 to 1.5 cm depending on the site of the body being treated and so the machine dependent distortions (up to 4 mm) experimentally determined by Ehrike and Schad (Ehrike and Schad, 1992) referred to in 1.5.3 above could clearly be of a magnitude that may compromise radiation therapy treatment, and clearly indicate that experimental assessment is necessary.

Shown in figures 1.13 to 1.16 is an example of how a geometric miss could occur in the case of treatment of a tumour in the abdomen.

**Figure 1.13:** Overlay of two image outlines acquired from an actual patient on RAH CT and MRI scanners.

**Figure 1.14:** Assume tumour is seen on MRI scan but not on CT image.
Figure 1.13 shows an example of a patient scanned by CT (solid outline) and MRI (dashed outline) at the Royal Adelaide Hospital in 1990. Even in this representation there is clear evidence of distortion. The CT scanner was subject to regular testing and can be considered geometrically accurate. Figure 1.14 assumes that some gross tumour volume (ICRU, 1993) is seen and marked on the CT scan, and further that the same tumour is not visible on the CT image (this happens in clinical practice).

From figure 1.15 it can be seen that in order for there to be edge correlation of the MRI and the CT outline, that the MRI outline needs to be “stretched” towards the left, which necessarily means that the associated tumour volume needs to be displaced along with it.

Figure 1.15: The tumour needs to be displaced to the left on the CT image to get edge correlation.

Figure 1.16: If a planning target volume was marked with a 1.0 cm boundary on the uncorrected MRI image, then a geometric miss would occur.
As can be seen in figure 1.16, if a planning target volume was marked surrounding the uncorrected MRI information with a margin of 1.0 cm, that the actual tumour position would still extend beyond those margins. This results in a geometric miss, which is one of the most serious errors in radiation therapy.

Several studies have been published discussing experiences with MRI as a tool in RTTP. One study (Pötter et al, 1992) used MRI for delineation of tumour volumes in 37 patients consisting of 13 brain tumours and 24 head and neck tumours. Distortion was assessed using a spatial linearity phantom for the MRI scanner used in that work and magnitudes of 1-3 mm were observed which the workers chose to ignore for external teletherapy beams, but acknowledged that stereotactic radiosurgery procedures should consider this. The same paper showed that in around 82% of the image planes investigated, the MRI information available resulted in change to the clinical target volume definitions that were obtained from CT information alone. The study alludes to the importance of assessing distortion for the specific MRI scanner being used as it is quite possible to exceed the 1-3 mm found on that equipment. Also, for larger fields of view the 1-3 mm of distortion could be exceeded due to the fact that main magnetic field homogeneity and gradient field’s linearity decrease as one moves away from the centre of the magnet.

A phantom study was also undertaken in work which involved use of MRI in subdiaphragmatic radiation therapy (Müller-Schimpfe, 1992) and distortions were found of up to 5 mm in this case due to the larger fields of view required for the abdomen area as opposed to head and neck as discussed above.

Stereotactic radiosurgery and neurosurgical procedures are now in widespread use clinically to very accurately target and treat small lesions. Geometric integrity is of paramount importance here and MRI images are often used as they offer clearer delineation of tumour or brain
structures. Various methods of image fusion are available (Vannier & Gayou, 1988, Henri et al 1991, Phillips et al, 1991, Chen & Pelizzari, 1989, Hemler et al, 1995) but none of these offer the ability to perform other than affine transformation operations to register the CT and MRI information. Thus no geometric corrections are carried out. A fusion method is schematically shown in figure 1.17.

![Diagram of image fusion process](image)

**Figure 1.17:** Schematic of image fusion process for use in stereotactic brain work. Note that the fiducial markers are from the CT scan data and are geometrically correct relative to the head outline. The MRI outline is then aligned to the CT outline, but, due to local spatial distortions in the MRI scan the relative position of the tumour may not be correct.

The fusion process illustrated here ensures that “macroscopically” the surface outline of the two images are aligned, and this is practically achieved in some systems by matching the bony table of the skull from both the CT and MRI data sets using for example least squares fitting. The transformations that achieve this matching consist of rotations, translations and scaling but not
warping. This means that any local distortions due to sharp changes in $H_0$ or in the field gradients will not be accounted for. The fiducial markers are used in the stereotactic planning process to define a coordinate system relative to a stereotactic frame (head ring or the like) which is generally clamped to the patient’s skull and also has the fiducial markers rigidly attached. The process of fusion described does not require a stereotactic frame to be applied to the patient during scanning as the coincidence of the data set with the CT information is achieved via the bony table match as mentioned previously.

Some workers have suggested (Schad et al, 1987) that distortion correction is a prerequisite for accurate stereotactic procedures, something which is not done in the above fusion process. Once again one can derive from this apparent variation in opinion, that it is essential to assess distortion for the particular MRI scanner that is intended to be used when geometric integrity is deemed to be important (ie for stereotactic work or large body cross sections for radiotherapy treatment planning).

1.8 Summary of the Specific Aims for This Project

This chapter has discussed the theory of MRI and used that theory to develop and explain the origins of geometric distortion in resultant images. The magnitudes of that distortion are somewhat variable from the reports in the literature, but this can be taken to mean that it is important to assess each MRI installation individually before using images in geometrically sensitive applications such as radiation therapy treatment planning for external teletherapy or radiosurgical beams or stereotactic neurosurgical procedures.
Two other forms of geometric distortion should be noted here. One is distortion that could result from the image processing algorithms of the computer (these could be flawed). This type of distortion could be regarded as machine dependent (and probably constant for a given image reconstruction method) making it analogous to eg main magnetic field inhomogeneity. Image processing distortions will be inherently dealt with in the following sections of this thesis, but will not be discussed specifically. The other type of distortion is related to the well known partial volume effect. This is where a spatially finite structure within the body of different MRI signal intensity to surrounding tissues only partially protrudes (from a neighbouring slice) into the imaged slice or is oblique to the plane of the slice. The result is that the 2-dimensional pixels making up the image of that slice, are assigned an MRI intensity which is the average of all the signals produced within the corresponding voxels. As a result contrast reduction can result, and this can reduce the ability to spatially resolve a structure (could be a poorly contrasting tumour). This type of distortion is very difficult to quantify in a the clinical setting, but slice thickness and fields of view (hence pixel size) should be judiciously chosen to minimise the effect.

The remainder of this thesis will explain the method developed for machine dependent distortion assessment and correction, and how in principle these methods can be applied clinically. An analysis was also attempted for a patient undergoing radiation therapy at the Royal Adelaide Hospital to assess the efficacy of the technique and the residual distortions due to magnetic susceptibility and chemical shift artifacts (ie. non-machine dependent distortions).
2. Chapter 2: RAH MRI Distortion Description

2.1 Introduction

This chapter serves to detail the work done to assess the magnitude of, and characterise the machine dependent distortions present for the Siemens Magnetom 1.0 T MRI machine installed at the Royal Adelaide Hospital in the Radiology Department, and which had been operational since January 1987. It was decommissioned in 1995.

A few technical details for the RAH MRI scanner are as follows: magnet - superconducting 1.0T with field uniformity (measured over a 30 cm diameter sphere) 5 ppm, temporal stability <0.1 ppm h⁻¹ with resistive shimming coils; RF system - 10 kW maximum power with frequency range 10 - 84 MHZ, usual RF pulse profile is sinc, automatic tuning performed for each patient (takes less than 30 sec), echo sampling time 7.6 ms; gradient system - gradient field strength 1 x 10⁻⁴ T cm⁻¹ with a 1.5 ms rise time, switching speed 2-4 T s⁻¹, quoted linearity x,y < 3% and z < 0.5%; MR image processing computer - DEC VAX 11/730 with Siemens BSP 11/ MR array processor (processing time for 256 x 256 image 6 s); image acquisition - 2D and 3D Fourier transform, FWHM slice thickness 3-200 mm, TR range 0.1 - 20 s, TE range 30 ms - 20 s, range of pixel sizes 0.5 - 3.0 mm; image quality - typical spatial resolution (head) 1.2 mm (body) 1.9 mm.

The literature provides examples of methods of distortion assessment which have been successfully carried out using a variety of spatial linearity phantoms. Work done at the University of Michigan in 1985 (Covell et al, 1986) involved construction of an acrylic phantom filled with MnCl₂ (400 μM), and a “checker board” array of square tubes which produced a regular pattern when imaged. Acrylic produces no signal in MRI whereas the MnCl₂ solution does. The image was then computer analysed and distortion quantified. This phantom was
designed for routine quality assurance purposes and had a large cross section to allow assessment of bigger fields of view, and a small cross section (obtained by re-orientating the phantom) to represent for example head scan fields of view. No attempt was made to correct distortion.

In 1987 a phantom was produced in Montreal and this formed the basis for the work of two master’s degree students (Drangova, 1987 and Gauvin, 1992) to assess distortion for stereotactic neurosurgery applications. This phantom was made of a series of parallel 1.5 mm thick acrylic plates with a series of regular holes drilled in them. The hole centres were separated by 1.0 cm and the holes were of diameter 1.5 mm. The plate assembly was placed in an acrylic tank and filled with 0.5 mM CuSO₄ solution. When the phantom was scanned the images could be assessed and the positions of the holes of the phantom could be measured against their respective positions in the image of the phantom.

In a 1995 publication (Prott et al, 1995), 27 MRI units in The Netherlands, Germany and Austria were assessed for geometric distortion with the aid of a large field of view acrylic phantom, consisting of a regular array of 477 tubes filled with an MRI signal producing solution. The phantom cross section was of similar dimension to an adult abdomen. In the same publication the following statement is made, “If the distortion of the MRI facility is known with the help of phantom measurements, it can be corrected with specific programs so that, after correction, the accuracy of the geometric information is only limited by the pixel resolution of the image.” this provided further motivation to develop a spatial linearity phantom at the Royal Adelaide Hospital.

2.2 RAH Spatial Linearity Phantom

In 1990 the American Association of Physicists in Medicine (AAPM) produced a report (Price,
Figure 2.1: One of 11 spatial linearity plates. Each plate has 193 x 6.0 mm diameter holes with centres spaced 28 mm apart.

1990) which among other things, describes the design basics for a spatial linearity phantom. This project was to initially assess distortion for the RAH MRI scanner for large fields of view. Thus a large phantom was proposed that would occupy most of the space available in the magnet bore. It was decided to build a phantom along similar lines to the Montreal phantom, except that it would be large enough to assess distortions for all body scan fields of view. The phantom was designed to comply with the AAPM report mentioned above and was manufactured at a local engineering company that had a computer controlled mill/drill for high precision (better than ±0.01 mm).

A total of 11 spatial linearity plates made out of 6.0 mm thick acrylic sheet were produced, and a drawing of one of these is shown in figure 2.1. The AAPM report (Price, 1990) identifies acrylic as a suitable material to use in conjunction with a signal producing fluid (see later) for spatial linearity assessment. Each plate was drilled with 193 x 6.0 mm holes in a regular array, with centres spaced 28 mm apart. In the completed phantom the plates will be in the xy-plane (transverse plane) of the magnet bore. The circular cross section magnet bore has a patient couch which occupies the lower section and this shape was chosen for the phantom in order to fill as
much as possible the remainder of the bore’s cross section. The three 10.0 mm diameter holes were for assembly purposes to allow all plates to be spaced apart and locked together with the aid of tie rods, spacers and locking nuts which comprise the clamping system for the arrangement. This is shown in figure 2.2.

![Figure 2.2: Spatial linearity phantom: clamping system. Each plate has three tie rods passing through each of the 10 mm holes and is separated from neighbouring plates by the spacers. The three tie rods are then tensioned with the nuts, which each bear on a washer to avoid plate damage.](image)

The plate spacing once assembled was 25.0 mm centre to centre. This gave a total length from the centre of the first plate to the centre of the eleventh plate of 250.0 mm. This length was chosen because the RAH MRI is capable of scanning a distance of 250 mm off axis from the centre of the bore. So in principle, if the entire useable length of the bore was to be assessed for distortion, then the phantom could be set up such that one end-most spatial linearity plate contained the magnet isocentre, and the most distant plate would be at a distance of 250 mm, ie. at the greatest position for image acquisition. The test could be repeated by shifting the phantom
Two sheets of 6.0 mm acrylic were heat formed to create the outer shell shown in figure 2.3. The overall length of the phantom was 290 mm. The joins in the tank were all “veed” out and welded to give maximum strength and avoid rupture. Two end plates were also manufactured (see figure 2.4) and these were welded to seal off each end of the phantom once the completed plate assembly was positioned inside. One of the end plates was fitted with a threaded bung with an
“O”-ring seal to enable filling of the phantom with liquid.

Figure 2.5 shows a photograph of the final phantom which was constructed. The 11 plates can clearly be distinguished. Another view of it is presented in figure 2.6. The high precision in the location of the plates with respect to each other is shown by the regular planes that are seen in this figure.

![Figure 2.5: Photograph of the side view of the spatial linearity phantom.](image)

![Figure 2.6: Photograph of the front view of the spatial linearity phantom. The precision in the plate location and the hole drilling is evident.](image)

The solution that was chosen for filling the phantom was 2 mM CuSO₄. The filling solution,
according to AAPM (Price et al 1990), should have:

\[ 100 \text{ ms} < T_1 < 1200 \text{ ms} \]

\[ 50 \text{ ms} < T_2 < 400 \text{ ms} \]

proton density = H\textsubscript{2}O density

It is important to have these parameters within these ranges, as they are similar to those for human tissues, which enables typical clinical pulse sequences to be used whilst scanning the phantom.

Drangova (Drangova, 1987) used 0.5 mM CuSO\textsubscript{4} at a field strength of 1.5 T from her phantom but did not report relaxation times. This same phantom was later used in further work (Gauvin, 1992) and it was reported that the CuSO\textsubscript{4} at this concentration made negligible contribution to overall magnetic susceptibility, which inspired its use in this project. CuSO\textsubscript{4} is also readily obtainable, very soluble, and acts as an anti-microbial agent. Other workers (Bucciolini et al, 1986) have shown that at 0.5 T, 2 mM CuSO\textsubscript{4} gave \( T_1 = 465 \pm 3 \text{ ms} \) and \( T_2 = 380 \pm 16 \text{ ms} \). It is expected that at 1.0 T (the field strength of the RAH MRI unit) for the same solution concentration that these relaxation times will be shorter, but still within the limits above.

The phantom once completed and filled weighed around 43 kg, which is about as heavy as one would wish from the point of view of manual handling. The total cost was around $3000.00 Australian. This compared more than favourably with commercially manufactured precision phantoms from a cost point of view, and besides, no commercial equivalent was known at the time of construction.
2.3 MR Images of the Phantom

The phantom was placed on the patient support assembly of the MRI scanner. It was carefully aligned such that the plates were parallel to the xy-plane (orthogonal to the axis of the main magnet solenoid) using the light beam alignment system. This light beam system also allows a point on the phantom (in this case the central spatial linearity plate) to be moved approximately to the isocentre of the magnet. An initial scan was carried out and the result is shown in figure 2.7.

![Figure 2.7: Initial scan of RAH spatial linearity phantom.](image)

The scan was a transverse $T_1$ weighted spin-echo sequence with $TR = 500$ ms and $TE = 15$ ms. The field of view was set at 500 mm to ensure complete coverage of the phantom. Slice thickness was set at 1.0 cm as this is typical for large field of view body scans. The preparation gradient ($0.47$ T cm$^{-1}$) is in the y-direction and the readout gradient ($0.63$ T cm$^{-1}$) was in the x-direction. Slice select gradient strength was $0.96$ T cm$^{-1}$. One can clearly see the bright signals corresponding to the 6.0 mm holes in the spatial linearity plate, in this case the central plate. It is also clear in the figure that some degree of displacement of the holes from a regular array has occurred giving a first glimpse of the geometric distortion.
At the time that the image of figure 2.7 was obtained the other spatial linearity plates were also scanned. In addition the effect of phantom position within the magnet bore was also looked at. This was done by doing a second scan at the magnet isocentre, but this time with the phantom offset by 250 mm such that one of the end-most spatial linearity plates was positioned at the isocentre. Results of this test will not be discussed until after the next section which explains the development of the algorithm which locates the position of the bright holes or peaks in the image.

2.4 Peak Search Algorithm Description

Once the MRI scans are reconstructed by the unit’s computer, they are stored digitally as files on the hard disk drive, there being one file per image. It was desired to download the image files for processing on a remote personal computer. This was done by connecting the MRI computer via an RS232 serial link to a notebook personal computer, and then using the shareware file transfer program known as KERMIT.

2.4.1 File Structure

Each image file is 135168 bytes long consisting of a 4096 byte header containing a large amount of information about the scan, and 131072 bytes of image pixel information (2 bytes per pixel). The pixel information is of 12 bit resolution which can be used to represent grey-scale values from 0 to 4095 (2^{12} = 4096). So each pixel is stored in binary format and is represented by two bytes, consisting of the twelve pixel bits, and the four remaining bits are used for user defined regions of interest (ROI). If the pixel in question has part of a region of interest associated, then one of the spare bits will be set. In implementing this system, ROI’s can be handled without compromising the underlying pixel’s grey-scale value.
In the file, byte locations 0 to 4095 consist of header information. The pixel information starts at byte offset 4096 from the beginning of the file. There are \(131072 \div 2 = 65536\) pixels represented, which corresponds to an image matrix of \(256 \times 256\) pixels. The first image pixel ie. that contained in bytes 4096 and 4097 of the file represents the top-left-most pixel (pixel 0,0) of the image. Subsequent byte pairs represent the next pixels in order along the top row of the image proceeding from left to right until the last pixel in the row is reached (pixels (1,0), (2,0),..., (255,0)). The pixel (255,0) is contained in byte locations 4606 and 4607. Bytes 4608 and 4609 represent pixel (0,1), which is the left hand pixel in the second row from the top of the image. This row by row storage pattern continues until pixel (255,255) at byte locations 135166 and 135167.

Using a binary file editor it is possible to analyse each byte location and read the hexadecimal (hex) value that is stored there. For example the file which contains the image in figure 2.7 has the hex values which correspond to the first ten pixels along the top row of the image as shown in table 2.1. Note that the third column has the bytes swapped around, which is required because the byte order is swapped in the file. Some computers do this when storing data.

Some of the header information was also extracted from the file, and as the description of the development of the algorithms proceeds, usage of header data will be discussed.
Table 2.1: The first ten pixels of the image of figure 2.7.

<table>
<thead>
<tr>
<th>Byte Offset Locations(^1)</th>
<th>Hexadecimal Values</th>
<th>Byte Swapped</th>
<th>Pixel (x,y)</th>
<th>Decimal Pixel Magnitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>4096/4097</td>
<td>03/00</td>
<td>00/03</td>
<td>(0,0)</td>
<td>3</td>
</tr>
<tr>
<td>4098/4099</td>
<td>05/00</td>
<td>00/05</td>
<td>(1,0)</td>
<td>5</td>
</tr>
<tr>
<td>4100/4101</td>
<td>12/00</td>
<td>00/12</td>
<td>(2,0)</td>
<td>18</td>
</tr>
<tr>
<td>4102/4103</td>
<td>11/00</td>
<td>00/11</td>
<td>(3,0)</td>
<td>17</td>
</tr>
<tr>
<td>4104/4105</td>
<td>0E/00</td>
<td>00/0E</td>
<td>(4,0)</td>
<td>14</td>
</tr>
<tr>
<td>4106/4107</td>
<td>0F/00</td>
<td>00/0F</td>
<td>(5,0)</td>
<td>15</td>
</tr>
<tr>
<td>4108/4109</td>
<td>15/00</td>
<td>00/15</td>
<td>(6,0)</td>
<td>21</td>
</tr>
<tr>
<td>4110/4111</td>
<td>0D/00</td>
<td>00/0D</td>
<td>(7,0)</td>
<td>13</td>
</tr>
<tr>
<td>4112/4113</td>
<td>08/00</td>
<td>00/08</td>
<td>(8,0)</td>
<td>8</td>
</tr>
<tr>
<td>4114/4115</td>
<td>07/00</td>
<td>00/07</td>
<td>(9,0)</td>
<td>7</td>
</tr>
</tbody>
</table>

Note: 1. The byte offset location corresponds to the position of the byte in question counted in whole bytes from the beginning of the file (starting with offset location 0 as the first byte in the file).

2.4.2 On the Nature of the Spatial Linearity Phantom Image Data

Shown in figure 2.8 is a surface plot of the pixel data in the block with corner pixel locations (100,0), (120,0), (100,40) and (120,40). For all pixels along the x-direction with y index values from 0 to about 12 one can see that these have a relatively low magnitude and this corresponds
Section of Spatial Linearity Pixel Data

Figure 2.8: Structure of a sample of the pixel information from the image of figure 2.7.

to the region of air in the magnet bore outside of the phantom. As the y pixel index increases from 13 through to around 20 there is a fairly steep increase in pixel magnitude. This corresponds to the volume of CuSO₄ solution between the spatial linearity plate (in this case the central plate in the phantom) and the phantom’s outer case. The other salient feature of this figure is the presence of four peaks in pixel magnitude. These peaks correspond to CuSO₄ solution in the 6.0 mm holes drilled in the spatial linearity plate. With reference back to figure
2.7, the actual holes which correspond to these particular peaks are the second and third holes in the top row of six holes, and the two holes directly below them. The pixel magnitudes between the peaks in figure 2.8 are raised above those corresponding to air outside the phantom because the slice width (1.0 cm) extends 2.0 mm either side of the spatial linearity plate into the CuSO$_4$ solution. If the spatial linearity plate was thicker than the scan slice width, then these pixels would have a lower value due to the fact that acrylic produces very little signal in MRI.

Central Section of Spatial Linearity Plate

![3D Graph](image)

**Figure 2.9:** Structure of 9 peaks near the centre of the spatial linearity plate.
Figure 2.9 shows the structure of the spatial linearity data in the vicinity of 9 peaks near the centre of the image. The corner pixels for this block of image data are (105,110), (150,110), (105,150), (150,150). It is quite clear from this figure that the peak data is highly discernable against the background and this provided the incentive to develop an algorithm that will automatically find these peaks.

**Single Peak of Spatial Linearity Plate**

![Diagram showing a single peak of spatial linearity plate](image)

**Figure 2.10:** Detail of the structure of a single peak in the spatial linearity image.

Shown in figure 2.10 is a detailed surface plot of a single peak. The boundary pixels for this block of data are (123,112), (131,112), (123,120) and (131,120). The header information for this
image was interrogated ascertain the scale factor which is stored as the number of whole pixels which make up 10 cm. In this case this number was 51, and so the pixel size was $100 \div 51 = 1.96$ mm. So the diameter of the holes (6.0 mm) covers about 3 pixels in any direction.

2.4.3 Development of the Peak Search Algorithm

The code for this algorithm was written in Borland Turbo C$^{3}$ version 2.0. A full listing of the code is presented in appendix A. The program name is “peak.c”. The main elements of the important subroutines in program peak.c will now be discussed and reference will be made to the relevant line numbers from the listing in appendix A.

Routine “main()”; Line 54

This routine starts by requesting the user for input of the name of the image file to be read, opens that file for binary reading. It also prompts the user for the name of the output file where the cartesian coordinates are to be placed of the found peaks. The output file is also opened in preparation for writing at this point (lines 56-72).

The file pointer is sent to byte offset 650 which contains is the start address of a pair of bytes that contain the size of the x-dimension of the image matrix, which is read (along with the size of the y-dimension). A test is then performed to check that the matrix is 256 x 256. If it is not the program terminates because it cannot handle other dimension sizes (lines 73-86).

The number of pixels that make up the distance of 10.0 cm is now read from the two bytes starting at address 694 bytes from the beginning of the file. This becomes the image scale factor.

---

$^{3}$Borland International, Scotts Valley, CA, USA
which is utilised later on (lines 87-91).

Memory space is now allocated for the 256 x 256 two dimensional array “matrix_raw”. A test is done to ensure that there is adequate memory available to do this. The array is then filled with the byte swapped pixel data (lines 92-116).

A second array called “matrix_mask” which is also a 256 x 256 two dimensional array has memory allocated (again with a test for adequate free memory) and has every location filled with zeros (lines 117-136). The use of this mask array will be discussed later.

Next the central zone of the spatial linearity image is assessed to determine the general magnitudes of the spatial linearity peaks, and also the magnitude of the MRI signal detected between the peaks. The central pixel block being checked has corner pixels (100,100), (150,100), (100,150) and (150,150) (lines 137-149).

The subroutine row_scan is now called (lines 150-151).

*Routine “row_scan()”; Line 175*

The principle purpose of this subroutine is to control the scanning from left to right, line by line of the image matrix to enable the searching for possible signal peaks, corresponding to spatial linearity points. When a possible candidate is found another routine local_search_max is called that further isolates a local maximum and also applies some rejection criteria to minimise the likelihood of artifactual peak detection.
Each line of the image is scanned, and each pixel read is tested to see if it has a value less than the inter-peak magnitude determined earlier. If this is the case then it is extremely unlikely that a peak is nearby and thus moves straight on to the next pixel. This will be the case for pixels that are in air i.e. outside of the spatial linearity phantom (lines 186-187).

The array matrix_mask is also checked for the presence of any previously detected peaks in the area. Clearly if a peak has been identified then another one will not exist for another 2.0 cm or so. The array matrix_mask is a 256 x 256 array that initially has zeros in all of its locations. When a peak is detected in matrix_raw, the same location (and all locations within the surrounding 30 x 30 mm² area, an area centred on a peak extending to approximately half way to the next peak) in matrix_mask are filled with ones. This happens in the subroutine local_max_search. This also speeds up the algorithm, as masked out pixels do not have to be tested (lines 188-193).

The algorithm is searching for rising trends in magnitude as it scans from pixel to pixel. If the next pixel magnitude divided by the current pixel magnitude exceeds the value of peak_threshold, which is defined on line 19 (in this case 1.3 was used), then there is the possibility of being on part of a peak. Once this possibility has been confirmed then the subroutine local_max_search is called (lines 194-203).

Routine "local_max_search()"; Line 210

The current pixel, once it has been determined to possibly be part of a spatial linearity peak, a zone 5 x 5 pixels in extent (the left corner of this zone is 2 pixels above and 2 pixels to the left) has the maximum value determined and stored as well as the location of that maximum (lines 52
213-227). This will cover any possible peak as the spatial linearity hole can only be 3 pixels in extent.

The structure of real peaks are approximately circular when viewed from above (see figure 2.10). The next sequence of code attempts to confirm this structure by determining the maximum pixel value in an “annulus” (this value will be referred to as annulus 1 max.) 1 pixel wide immediately surrounding the pixel containing the local maximum value. Then the maximum pixel value is determined from the “annulus”, again one pixel wide immediately surrounding the first one, and this will be referred to as annulus 2 max. The peak is accepted if annulus 1 max. divided by annulus 2 max. is greater than the value of reject_ratio which was set at line 22, and in this case has the value 1.17. Without this test, structures like that seen in figure 2.8 due to the gap between the spatial linearity plate and the phantom’s outer case would be counted as peaks. Further rejection occurs if the peak doesn’t exceed the value of peak_height_min defined on line 24, and has the value in this case of 50 (lines 228-269).

The 30 x 30 mm² area is masked out in array matrix_mask as previously defined, and the subroutine centre_of_gravity is called (lines 270-283).

Routine “centre_of_gravity()”; Line 286

It was decided to implement a centre of gravity estimate in order to better estimate the position of the centre of the peak that has been localised. This allows an estimate of the position of the centre to sub-pixel accuracy. If the pixel coordinate of the centre of gravity of the peak is \((\bar{x}, \bar{y})\), and the pixels in a 5 x 5 array centred on the maximum value found for the peak \((x_i, y_j)\) each have their own magnitude \(w_i\), and \(W\) is the sum of all 25 pixels magnitudes then the centre of gravity
is found by the following two equations:

\[ x = \frac{\sum w_i x_i}{W} \]  
(2.1)

\[ y = \frac{\sum w_i y_i}{W} \]  
(2.2)

This method was tested on the data for the peak shown in 2.10. The pixel matrix surrounding the peak magnitude pixel is:

<table>
<thead>
<tr>
<th>x – y</th>
<th>125</th>
<th>126</th>
<th>127</th>
<th>128</th>
<th>129</th>
</tr>
</thead>
<tbody>
<tr>
<td>114</td>
<td>438</td>
<td>375</td>
<td>405</td>
<td>447</td>
<td>471</td>
</tr>
<tr>
<td>115</td>
<td>445</td>
<td>532</td>
<td>1009</td>
<td>767</td>
<td>491</td>
</tr>
<tr>
<td>116</td>
<td>400</td>
<td>816</td>
<td>1635</td>
<td>1086</td>
<td>543</td>
</tr>
<tr>
<td>117</td>
<td>435</td>
<td>658</td>
<td>1143</td>
<td>913</td>
<td>592</td>
</tr>
<tr>
<td>118</td>
<td>467</td>
<td>409</td>
<td>412</td>
<td>525</td>
<td>529</td>
</tr>
</tbody>
</table>

Now:  
\[ W = 15943 \]  
\[ \sum w_i x_i = 2026591 \]  
\[ \bar{x} = 127.11 \]

\[ \sum w_i y_i = 1850297 \]  
\[ \bar{y} = 116.06 \]
So: \( (x, y) = (127.11, 116.06) \)

If the graph in figure 2.10 is projected onto the x-axis one sees the result in figure 2.11. A cubic spline has been fitted to the pixel magnitude data, and the x-coordinate for the centre of gravity

![Centre of Gravity: Projection onto X-axis](image)

**Figure 2.11:** Graph showing the projection of the peak from figure 2.10 onto the x-axis. The pixel magnitude data has had a cubic spline fitted to it. The vertical dashed line represents the centre of gravity x-coordinate in fractional pixel units computed using equation 2.1. It has also been plotted. It is clear that the centre of gravity method is going to be a valid estimator.
for the sub-pixel coordinates of the centres of the spatial linearity peaks. Shown in figure 2.12 is the same thing but for the y-axis projection.

**Centre of Gravity: Projection onto Y-axis**

![Graph showing the projection of the peak from figure 2.10 onto the y-axis. The pixel magnitude data has had a cubic spline fitted to it. The vertical dashed line represents the centre of gravity y-coordinate in fractional pixel units computed using equation 2.2.](image)

**Figure 2.12:** Graph showing the projection of the peak from figure 2.10 onto the y-axis. The pixel magnitude data has had a cubic spline fitted to it. The vertical dashed line represents the centre of gravity y-coordinate in fractional pixel units computed using equation 2.2.

The algorithm calculates the centre of gravity (lines 284-301). Ultimately the pixel coordinates that have been used to date need to be converted to cartesian coordinates, with the origin centred on the peak that is nearest to the centre of the image (which also coincides with the centre of the
main magnet bore of the MRI unit). Due to the reproducibility of phantom set-up from one spatial linearity measurement session to the next it was decided that in general this origin would always be made coincident with the peak near the pixel coordinate (125,130). In figure 2.7 this spatial linearity point is found in the centre column of points, and is the 6th from the bottom. A test is performed to see if the current peak is the one in this position, and if so, the variables origin_x, and origin_y are given the values of the centre of gravity x and y pixel coordinates respectively (lines 302-312). The variables origin_x and origin_y are used in order to carry out the coordinate transformation later on.

For all spatial linearity points found, the x and y pixel coordinates are passed to the arrays peak_x and peak_y respectively (lines 313-315).

Once all the peaks are found the routine row_scan terminates, and the large amount of memory used for arrays is freed up (lines 152-154). The subroutine output_cartesian is now called (line 160).

*Routine “output_cartesian()” ; Line 321*

This routine converts the coordinates of the found peaks in the pixel coordinate system (with the origin at the top left of the image, and x increasing to the right and y increasing downwards) to a conventional cartesian coordinate system, with the origin centred on the spatial linearity peak that is nearest to the image centre. The resultant transformed coordinates are printed in comma delimited form to an ASCII output file (lines 322-333).

Due to the fact that very small rotations of the phantom can occur when placing it on the couch
for scanning (which could result in significant displacements of spatial linearity points near the periphery of the phantom, when compared to expected positions of the points assuming no rotation) this average angle of rotation is assessed. This will allow superimposition of the known coordinates of the spatial linearity points onto the image ones. The angle is assessed by calculating the average rotation of the four nearest points to the origin in East, West, North and South directions with respect to that origin (lines 334-373). The tacit assumption is made that there is minimal distortion at the centre of the image, and was the same approach as Gauvin (Gauvin, 1992).

_Routine "sl_out_cartesian()"; Line 377_

This subroutine is called on line 160. The purpose of this routine is to print out the coordinates of the spatial linearity phantom’s 193 holes to the same ASCII file as used in the previous subroutine. Before the cartesian coordinate data is written to the file it is rotated according to the average angle corresponding to the rotation of the phantom during scanning, as determined in the previous subroutine (lines 377-482).

2.5 Peak Search Algorithm Results

Shown in figure 2.13 are the results of the peak search algorithm for the image in figure 2.7. The diamonds in the plot are the image spatial linearity points found by the peak search algorithm described above. Superimposed on the same axes are the coordinates of the holes in the spatial linearity phantom (appropriately rotated to take into account an actual rotation of the phantom when it was placed on the patient support assembly of the scanner.
Image and Actual Spatial Linearity Points

**Figure 2.13:** Scatter plot of the image data spatial linearity points for the image of figure 2.7 found with the peak search algorithm (this is the phantom’s central spatial linearity plate and is located at the magnet isocentre). Superimposed on the same axes are the coordinates (appropriately rotated) of the actual holes in the spatial linearity plate.

Distortion is clearly seen in this figure, and is characterised by the pairs of points not perfectly overlapping. It is clear that the distortion gets worse, the further one moves away from the origin.
2.6 Methods of Distortion Magnitude Assessment

Success with the method of finding the peaks in the MRI image of the spatial linearity phantom, allowed the work to proceed to quantifying the amount of distortion present. This was achieved by different techniques and each will be discussed now in turn, and the results for each method presented.

**3D Scatter Plot of Distortion Magnitude**

![3D Scatter Plot of Distortion Magnitude](image)

**Figure 2.14:** 3-D scatter plot of all spatial linearity points, with their respective distortion magnitudes, as found in the image of figure 2.7 (phantom’s central spatial linearity plate positioned at the magnet isocentre).
2.6.1 3-D Plot of Distortion Magnitude and Associated Cartesian Coordinates

Computer code, “dist_mag.c” was developed to prepare the data for the plot in figure 2.14. A listing is presented in appendix B. The description of this piece of code here will be brief because the bulk of the code is the same as that for the peak search algorithm “peak.c” in appendix A and which was described in section 2.4. Only the essential differences, and the subroutine which calculates the distortion magnitude will be discussed.

Changes Introduced to “dist_mag.c” Compared to “peak.c”

Subroutine output_cartesian (line 349) has been changed so that it no longer outputs the image spatial linearity points to a file, rather it just stores the coordinates in a pair of arrays peak_x and peak_y. Subroutine sl_cartesian (line 403), which was formerly named sl_out_cartesian in peak.c, likewise no longer outputs the phantom spatial linearity phantom coordinates to a file, but rather stores them in the pair of arrays sl_x and sl_y.

Routine “synchronise_arrays()”; line 508

In order to conveniently calculate the distance between the spatial linearity points in the image, and those of the rotated spatial linearity phantom data, it was decided that for any given spatial linearity point, that the array indices of peak_x and peak_y (the image spatial linearity coordinates) would be made to match those of sl_x and sl_y (the phantom spatial linearity coordinates). This was performed in lines 533-555 of the code. This in effect means that for the nth spatial linearity point, the coordinates (peak_x[n],peak_y[n]) and (sl_x[n],sl_y[n]) represent
the positions of that point in image and phantom space respectively.

Routine "xyz_mag()"; line 562

This routine loops through each of the 193 spatial linearity points and calculates the displacement of the image spatial linearity points from those of the phantom. The phantom coordinates and the associated distortion are then output to a comma delimited ASCII file.

Program dist_mag.c Results

The plot in figure 2.14 was produced using the output data from "dist_mag.c" to give some overall indication for the magnitude of the distortion present throughout the entire image space. It is apparent that the distortion gets worse the further one is away from the origin of the image. Some basic statistical analyses were conducted for the data of figure 2.14, and the results were as follows: maximum value of distortion = 11.6 mm, mean value of distortion = 3.4 mm and the standard deviation of distortion = 2.6 mm.

2.6.2 American Association of Physicists in Medicine (AAPM) Distortion Definition

This is given in AAPM report number 28 (Price et al, 1990). They define distortion in percentage terms as follows:

\[ \text{Percentage Distortion} = \frac{\text{True Dimension} - \text{Observed Dimension}}{\text{True Dimension}} \times 100\% \]  
(2.3)
The code of dist_mag.c was modified to produce distortion assessment as per equation 2.3. A listing of subroutine “xyz_mag()” is presented only as appendix C, as the rest of the code is identical to dist_mag.c in appendix B.

For any given spatial linearity point percentage distortion was assessed by calculating the distance from the origin to the spatial linearity point in the phantom (true dimension) and the distance from the origin to the same spatial linearity point in the image (observed dimension). This was done for all 193 spatial linearity points. The results for this were as follows: maximum percentage distortion = 4.6%, mean percentage distortion = 1.4% and the standard deviation of percentage distortion = 1.0%.

It is felt that this method of distortion quantification is somewhat limited for this instance in that it only calculates distortion in terms of radial distance from the origin. Distorted points in the image may not be displaced radially from where they should be if no distortion was present. That is they could still be at the same radial distance from the origin, but significantly displaced in the image. This method of distortion quantification is, however, useful for daily quality assurance (QA) measurements where a phantom of known dimension could be scanned, and two orthogonal dimensions could be measured using the graphics tools of the user interface of the MRI system, and compared to the known dimensions of the phantom. In this case equation 2.3 could be applied and the results stored for QA purposes.

2.6.3 Annular Distortion Assessment

Due to the apparent increased distortion as one moves away from the centre of the image, it was decided to define and measure maximum distortion in terms of millimetres displacement of
observed spatial linearity points from where they ought to be, within a series of concentric annuli centred on the origin and extending out to the maximum phantom dimension.

The subroutine used to calculate this annular distortion is listed in appendix D. It replaces the subroutine “xyz_mag()” from the program listing in appendix B. The results for this program on the image of figure 2.7 are shown in the bar graph of figure 2.15.

**Maximum Annular Distortion**

![Bar graph showing maximum annular distortion](image)

**Figure 2.15:** Maximum distortion detected within origin-concentric annuli of width 40 mm and with mean position 20, 60, 100, 140, 180 and 220 mm from the origin.
The increase in distortion as the distance away from the origin is increased is clearly shown in the graph. This type of graph could be useful in a radiation oncology treatment centre in order to assess the likely geometric error associated with utilising MRI scans. For example if a head scan was to be performed for use in clinical target volume definition, then the field of view would not exceed approximately 200 mm if the patient’s head were centred in the MRI magnet. From the graph in figure 2.15 it can be seen that distortion would not be expected to exceed 3.5 mm for the RAH MRI scanner (ie. at a radius of 100 mm from the origin, which corresponds to a field of view of 200 mm) which would probably be acceptable for some external beam radiation oncology treatments, but would definitely not be acceptable for stereotactic radiosurgical or neurosurgical procedures, as typically sub-millimetre precision is desired in these instances.

For radiation oncology treatments involving body scans, where the fields of view could be 400 mm (ie. corresponding to the annulus position of 220 mm in figure 2.15) distortion could be up to approximately 11.5 mm for the RAH scanner, which is of the order of the margin typically placed around the clinical target volume to form the planning target volume, and so would be unacceptable in so far as risk of geometric tumour miss is concerned.

2.6.4 Distortion Magnitude Distribution

A final property that was investigated was the frequency distribution of the absolute value of the distortion magnitude. A scoring bin size of 0.25 mm for the distortion magnitude was chosen and the total number of spatial linearity points falling in each bin was calculated. The subroutine which extracts this distribution is listed in appendix E. Once again this replaces the subroutine “xyz mag()” from the program listing in appendix B. The graph in figure 2.16 shows the resultant distribution from this subroutine where the scoring from the subroutine has been converted from the number of spatial linearity points, to the percentage of the total number of
spatial linearity points that fall within each bin. This conversion allows comparison of distortion distributions, which will prove useful later on.

**Distribution of Distortion Magnitude**

![Graph of distribution of distortion magnitude](image)

**Figure 2.16**: Frequency distribution of the percentage of the total number of spatial linearity points found in image 2.7 in 0.25 mm wide scoring bins of absolute value of distortion magnitude.

This provides another means of assessing the nature of the distortion present.
2.6.5 Off Axis Distortion Assessment

Distortion assessed so far only concerns the slice obtained from the image of figure 2.7 which was for the spatial linearity plate at the centre of the phantom, where that plate was placed at the isocentre of the MRI magnet. At the time that this scan was performed, other scans were made at slice positions corresponding to the other spatial linearity plate positions. The methods of distortion assessment studied so far were applied to these other scans and the results will be presented in this section. This will establish the effect of off-axis scanning on distortion. These results are presented in table 2.2.

For optimum performance of the algorithms in their ability to maximise the number of real peaks found for each spatial linearity plate image studied, the user defined variables peak_threshold (discussed in the section describing the routine “row_scan()”), reject_ratio and peak_height_min (in section describing routine local_max_search) were iteratively varied manually after viewing the results of the peak search algorithm. This had to be done because the pixel structure in the image matrix on occasions led to some artifactual peaks being found, and these three variables enabled rejection of those.

Previously obtained off axis spatial linearity scans at positions 25, 50, 75 and 125 mm from the magnet isocentre were studied.

Figures 2.17-2.20 show the locations of the image and the actual spatial linearity points for these other positions within the magnet bore.
Figure 2.17: Scatter plot of the image and actual spatial linearity points for a position 25 mm off axis.

Figure 2.18: Scatter plot of the image and actual spatial linearity points for a position 50 mm off axis.
Figure 2.19: Scatter plot of the image and actual spatial linearity points for a position 75 mm off axis.

Figure 2.20: Scatter plot of the image and actual spatial linearity points for a position 125 mm off axis.
2.6.6 Off Axis Dependence of Absolute Distortion and AAPM Percentage Distortion

Table 2.2 shows that from an absolute value of maximum distortion magnitude, and maximum percentage distortion (AAPM definition) that the off axis slices show worse distortion than the slice at 0 mm (i.e. the isocentre slice. This is not surprising because the MRI unit's main magnetic field is shimmed (made uniform) in a transverse plane corresponding to this slice. Also the gradient fields are referenced to "pivot" about the isocentre, such that gradient field non-linearities would in general be minimised in this slice (although x and y non-linearities would still exhibit themselves in this central slice the further one is away from the isocentre).

Further support for this is seen in that the mean values of distortion magnitude and AAPM percentage distortion are significantly greater for any other off axis slice (maximum values of \( p = 0.0005 \) and \( p = 3 \times 10^{-10} \) (2 tailed t-Test) respectively) than for the isocentre slice. However, all off axis slices showed no significant differences between means of either type of distortion.

**Table 2.2:** Distortion assessment results for different off axis positions.

<table>
<thead>
<tr>
<th>Off Axis Distance (mm)</th>
<th>Distortion Magnitude</th>
<th>AAPM Distortion</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Maximum (mm)</td>
<td>Mean (mm) ± Std. Err.</td>
</tr>
<tr>
<td>0</td>
<td>11.6</td>
<td>3.4±0.2</td>
</tr>
<tr>
<td>25</td>
<td>12.8</td>
<td>5.0±0.2</td>
</tr>
<tr>
<td>50</td>
<td>13.9</td>
<td>5.0±0.2</td>
</tr>
<tr>
<td>75</td>
<td>13.4</td>
<td>5.2±0.3</td>
</tr>
<tr>
<td>125</td>
<td>13.3</td>
<td>4.4±0.2</td>
</tr>
</tbody>
</table>
2.6.7 Off Axis Dependence of Annular Distortion

Off Axis Annular Distortion

![Graph showing annular distortion variation with off axis scanning slice.](image)

**Figure 2.21:** Annular distortion variation with off axis scanning slice.

Shown in figure 2.21 is the dependence of annular distortion on off axis distance. It is clear, as it was for the isocentre slice, that as one moves out in a radial direction in any given slice that the distortion magnitude increases. For any given annulus, there is a general increase in distortion as one moves further off axis along the magnet bore, again highlighting the possibility of greater main magnetic field inhomogeneity, and gradient field non-linearity at these off axis positions. The exception to this is the 125 mm off axis slice. The reason for this is likely due to the fact that
(as can be seen in figure 2.20) it was not possible to detect all spatial linearity points on the left hand side of the image, due to local distortions in the z-direction which effectively move the spatial linearity plate out of the imaged slice and flood that part of the image with high signal CuSO₄ solution, thus decreasing spatial linearity peak detectability by the peak search algorithm. The peak will decrease in intensity if the plate is partially out and will disappear entirely when the plate is totally out of the imaged slice.

**Figure 2.22:** Off axis variation of distortion frequency distribution (0.25 mm wide scoring bins).
2.6.8 Distortion Magnitude Distribution Dependence on Off Axis Position

The graph of figure 2.22 indicates the variation in the frequency distribution described in section 2.6.4, for 0 mm and 75 mm off axis. It can be seen that proportionally more of the spatial linearity points have a greater distortion magnitude for the 75 mm off axis slice as compared to the slice on the isocentre (0 mm of axis).

**Mean X and Y Components of Distortion**

![Graph showing the relative magnitudes of the mean x and y distortion as a function of off axis distance. Error bars represent the standard errors on the mean values of the distortions present at each off axis position studied.](image)

*Figure 2.23:* Bar graph showing the relative magnitudes of the mean x and y distortion as a function of off axis distance. Error bars represent the standard errors on the mean values of the distortions present at each off axis position studied.
2.6.9 X and Y Components of Distortion and Variation with Off Axis Position

Reference was made in section 1.5.3 to the fact that the frequency encode (or readout) gradient direction could have distortion components due to both gradient field non-linearities, and main magnetic field inhomogeneities. For spin-echo imaging, it was also pointed out in section 1.5.2 that the phase encode (or preparation) gradient direction was immune from main field inhomogeneity effects, but still suffered from gradient field non-linearities. For the spatial linearity scans performed on the RAH scanner, the readout gradient direction was along the x-axis, and the preparation gradient direction was in the y-axis direction.

If the distortion present in the images has a significant component of main magnetic field inhomogeneity, then the amount of distortion present in the x-direction should exceed that of the y-direction. Trivial modification to the code of program dist_mag.c was performed to assess x and y distortion magnitudes, and also study the dependence of these on of axis position.

The results of this assessment are shown in the graph of figure 2.23. In all scan positions there is significantly more distortion along the x-axis than the y-axis. This indicates significant main magnetic field inhomogeneity related distortion in addition to that due to x-gradient field non-linearities. Furthermore, there is a significant increase in mean x-distortion off axis as compared to that for the isocentre slice. This is most likely due to the fact that the main magnetic field is only made uniform by the equipment’s service personnel in the isocentre slice with the aid of a gauss meter, and it is therefore quite possible that field inhomogeneity will be different off axis.

2.6.10 Effect of Interchanging the Phase Encode and the Readout Directions

The effects discussed above should alter if the phase encode direction is now made to be in the
x-direction instead of the y-direction. This was only able to be shown qualitatively, as the electronic files of the relevant images were not down loaded from the MRI computer. However, it was possible to scan the images produced from film and colour process them such that when they were overlaid, relative spatial linearity point positions could be differentiated from one another.

The result is shown in figure 2.24. The images were overlaid and registered at the same origin point as before. The black points were from an image with the phase encode direction along the y-axis, and the blue points are with the phase encode direction parallel to the x-axis. With the phase encode and the frequency encode gradients swapped, the direction that distortion is due to the main magnetic field inhomogeneities is seen to swap also.

By looking at the left-hand and the bottom centre portions of the figure (2.24) one sees that for the y-phase encode direction (the black points) extend more to the left of the image (along the x-axis) which is due to the dependence on x only of the distortion due to the main magnetic field inhomogeneity. Conversely the blue points (phase encode direction along the x-axis) extend more towards the bottom of the image as a result of distortions due to the main magnetic field inhomogeneity having a y-dependence this time.
Figure 2.24: Overlaid images of the central phantom spatial linearity plate. Black points are from an image with the phase encode direction along the y-axis, the blue points are for an image (of the same slice) but with the phase encode direction along the x-axis.
2.6.11 Distortion Due to the Phantom Position in the Magnet Bore

This was assessed by comparing the spatial linearity image of the phantom’s centre slice (with that slice centred at the magnet isocentre), with the spatial linearity image of one of the phantom’s end spatial linearity plates centred at the isocentre.

![Image Spatial Linearity Points at Isocentre](image)

**Figure 2.25:** Spatial linearity point positions for the phantom’s centre plate at the isocentre and for the phantom’s end plate at the same physical position within the magnet.

This result is shown in figure 2.25. It is clear from this figure that the two images exhibit very little difference in terms of distortion present. In fact the maximum distortion difference
observed was 2.4 mm which is of the order of the pixel size of 1.96 mm. The mean distortion difference observed was only 1.12 ± 0.05 mm. Note that a few spatial linearity points in the end SL plate image were not detected by the peak search algorithm, which results in the “missing” diamonds in figure 2.25.

2.6.12 Temporal Stability of Observed Distortion

It has already been pointed out by other workers (Ehricke & Schad, 1992) that the geometric distortions associated with MRI equipment are relatively stable. It was planned to test this for

**Temporal Stability of Distortion (4.5 Months)**

Central Slice at Isocentre

![Scatter plot showing central slice spatial linearity stability over a 4.5 month period.](image)

**Figure 2.26:** Scatter plot showing central slice spatial linearity stability over a 4.5 month period.
the RAH MRI system for two spatial linearity scans performed approximately 4.5 months apart. Unfortunately service activity was performed over this period (eg. magnet shimming etc.) which may have affected distortion (over and above any changes that might have occurred as a result of instrument drift). Scans of the spatial linearity phantom were obtained on 28 June 1994 and 16 November 1994.

The results are presented in figure 2.26 in this case for the phantom's central spatial linearity slice located at the magnet isocentre. The images were processed to locate the spatial linearity points,

**Temporal Stability of Distortion (4.5 Months)**

![Image of scatter plot showing 100 mm off axis spatial linearity stability over a 4.5 month period.](image)

**Figure 2.27**: Scatter plot showing 100 mm off axis spatial linearity stability over a 4.5 month period.
and the two data sets were overlaid, with a rotation applied to one of the images in order to account for any differential phantom rotation from one scan to the next. It is clear that there have been some changes in distortion over this relatively long period, but the overall pattern of distortion is similar in both images, with a maximum difference observed of 5.0 mm (mean distortion difference = 1.71 ± 0.07 mm).

The assessment was repeated for a spatial linearity plate that was 100 mm off axis in the magnet bore to test stability here also. The result is shown in figure 2.27.

The stability of the pattern of distortion after this considerable time is still evident. The maximum distortion change measured in this case is 5.8 mm with mean of 2.00 ± 0.08 mm. It would appear therefore that the main field inhomogeneities and gradient field non-linearities exhibit some degree of temporal stability, but the tests would have to be repeated at shorter intervals in order to further characterise temporal stability. It is likely that the service activity earlier referred to has had some quantitative effect on the amount of distortion present, as magnet shimming was performed to produce main field uniformity in the xy-plane that contains the isocentre. There is evidence however that the variation in distortion over a short period of time is small. This is seen in the result discussed in section 2.6.11, which shows that even when scanning different spatial linearity plates at the same position in the magnet bore at different times (scan data were acquired 15 minutes apart), the distortion changes were minimal and certainly acceptable in the context of large field of view scanning (maximum distortion difference = 2.4 mm with a mean of 1.12 ± 0.05 mm).

Spatial linearity scans of the phantom may not be required before every patient scan that will ultimately be corrected for distortion. It may only be necessary to do phantom scans at some predetermined interval, and after each technical service instance. It should be pointed out that
temporal stability should be assessed on a machine by machine basis as the parameter may vary, and the frequency of assessment will have to be altered depending on the time frame of observed distortion changes.

2.7 Z Distortion Assessment

A series of scans was performed of all plates of the spatial linearity phantom. The scan position was chosen carefully in order to place the scan slice centre to be coincident with the spatial linearity plate being scanned. This was achieved by performing a coronal scan through the centre of the phantom. This scan is shown in figure 2.28. The bright bands in the image represent the inter-plate space filled with signal rich CuSO₄, and the dark bands are the non-signal producing spatial linearity plates. A utility exists as part of the user interface for the MRI system that enables the operator to mark the desired position of the transverse slices to be acquired. As can be seen in figure 2.28, the desired slice positions were placed as near as possible to the centre of

![Figure 2.28: Coronal MRI scan of the spatial linearity phantom showing the positioning of the desired transverse MRI scans, centred on each spatial linearity plate.](image-url)
the spatial linearity plates.

Another observation about the image in figure 2.28 is that the signal produced from the copper sulphate solution is not completely uniform. This is thought to be due to sub-optimal design of the body rf transceiver coils. Image non-uniformity is a problem in general for MRI images.

The resultant transverse scans produced showed good definition of the plates, in that no scans exhibited large regions of high signal CuSO₄ solution present which would indicate that the scans were non-coplanar with the plates. The plate spacing in the phantom is in 25 mm increments from -125 mm to +125 mm. Table 2.3 shows the actual phantom z-positions and the z-positions read from the images.

**Table 2.3**: Phantom z-positions and image z-positions for transverse scans of the spatial linearity phantom.

<table>
<thead>
<tr>
<th>Actual Phantom z-positions</th>
<th>Image Derived z-positions</th>
</tr>
</thead>
<tbody>
<tr>
<td>-125</td>
<td>-121.9</td>
</tr>
<tr>
<td>-100</td>
<td>-96.7</td>
</tr>
<tr>
<td>-75</td>
<td>-73.9</td>
</tr>
<tr>
<td>-50</td>
<td>-49.8</td>
</tr>
<tr>
<td>-25</td>
<td>-24.9</td>
</tr>
<tr>
<td>0</td>
<td>-1.0</td>
</tr>
<tr>
<td>25</td>
<td>24.4</td>
</tr>
<tr>
<td>50</td>
<td>49.8</td>
</tr>
<tr>
<td>75</td>
<td>73.1</td>
</tr>
<tr>
<td>100</td>
<td>98.6</td>
</tr>
<tr>
<td>125</td>
<td>122.1</td>
</tr>
</tbody>
</table>
The results were graphed to investigate the relationship of the image perceived position versus the phantom actual position. This graph is shown in figure 2.29. Clearly the relationship is linear ($R^2 = 0.9999$). There is however an offset of 3.88 mm which is due to an error associated with the positioning of the phantom accurately such that the centre slice is at the magnet isocentre. Positioning for the scans was performed by aligning the central plate with a light beam marker.

**Magnetom Z-axis Distortion**

![Graph showing linear relationship with R² = 0.9999](image)

**Figure 2.29**: Relationship of the image perceived slice z-position versus the actual z-position.
that is external to the magnet bore. A button is then depressed which drives the phantom to the nominal isocentre. The presence of the offset in figure 2.29 indicates that the light marker is in

**Magnetom Slice Position Error**

![Graph showing Magnetom Slice Position Error](image)

**Figure 2.30:** Z slice position error magnitudes as a function of the actual position of the scanned slice within the magnet bore.
need of adjustment.

The offset effect due to phantom placement error was removed and then the bar graph of figure 2.30 was produced which illustrates the MRI error associated with $z$-position. It can be seen that the maximum error expected is 3.3 mm for the centre of the slice. This magnitude of distortion in the $z$-direction is not considered significant in the context of conventional external beam radiotherapy, because the scan slice thickness used for both CT and MRI scans is typically 10.0 mm anyway. The same argument does not apply to stereotactic neurosurgery or radiosurgery, however, because slice thicknesses employed for these procedures are usually only 2.0 mm.

Furthermore, the assessment of $z$ distortion carried out here is only determining the average magnitude for one line in each slice (i.e. the lines described by the intersections of the coronal plane used to position the axial scans, with the plane of the transverse slices). For stereotactic use this assessment would have to be performed with a phantom which was capable of being orientated such that detailed spatial linearity data could be obtained in either the coronal or sagittal planes, hence allowing a full assessment of $z$-distortion.

### 2.8 Conclusions

This chapter has explained the method developed for distortion assessment involving a locally produced phantom, and computer software. The phantom and the software produced here will prove essential to the distortion correction technique to be described in the following chapter.

Results were produced as a result of various analyses of the magnitudes of distortion exhibited by the RAH MRI machine. It would appear that the concept of "annular distortion" is a useful one because, depending on the clinical field of view sizes to be employed, it is possible to
estimate maximum likely distortion magnitude to be observed, and this allows assessment as to whether or not this will compromise the procedure. It should be borne in mind, however, that annular distortion will be effected by interchanging x- and y-gradients and/or by applying different pulse sequences to those used to first assess it. For example a head and neck patient that is being planned for radiation therapy may only occupy the central portion of the magnet bore, and so the amount of actual distortion expected in any given slice may be clinically acceptable without any distortion correction. Until this sort of analysis is performed, however, such decisions are difficult to make.

The analysis of the frequency distribution of the absolute value of the distortion also was shown to offer information about the nature of the distortion present. This will also be used in the next chapter to assess the effectiveness of the correction algorithm that is to be discussed.

It is apparent that distortion varies with off axis scanning position within the magnet bore. This implies that any geometric correction procedure must take the off axis position of the image being corrected into account.

The results of the assessment carried out in this chapter also showed that the distortion causes were both due to main field inhomogeneities, and to a lesser extent to gradient field nonlinearities. These are the machine dependent distortions that it is hoped can be removed from images of real patients. The main field inhomogeneities appeared to be more significant due to the observed effect of distortion direction swapping following the exchange the frequency and phase encoding directions.

The phantom exhibited no differential susceptibility effects due to its position in relation to a given scanning position and this augers well for using the phantom as part of the distortion
Temporal stability of the distortion was assessed in a limited way, the study over a long period of time showed that the nature of the distortion was relatively unchanged, but there were some changes in magnitude. Stability over a short time (~15 minutes) was, however, acceptable in the context of conventional external beam radiotherapy. It is clear that distortion stability with time needs to be assessed for individual scanners where it is intended to employ phantom based geometric image correction, such that reliable and current spatial linearity data is available to perform such corrections.

Distortion along the z-axis was estimated and did not appear to be severe enough to compromise the use of trans-axial MRI information for external beam radiotherapy, especially in light of the fact that the CT slices typically employed (10 mm thick usually) for this application are over twice the thickness of the maximum MRI distortion observed in the z-direction. The magnitude of z-distortion observed here (3.0 mm) is of the same order of magnitude as that predicted in section 1.5.1 (1.3 mm from $H_0$ inhomogeneities and z-gradient non-linearity effects combined).

A more detailed assessment of z-distortion would be required in the context of stereotactic based procedures, however, where the slice thickness employed is typically 2-3 mm. The requirements of a phantom in this case would be that it is possible to rotate it in the magnet bore such that the spatial linearity plates could be orientated parallel to the sagittal and coronal planes. This would allow a true three dimensional assessment of MRI distortion, and thus the opportunity to correct this. The spatial linearity phantom described in this thesis was not capable of being re-orientated, unfortunately. This fact restricts the work that is to follow to the two dimensional case of geometric correction within a trans-axial MRI slice only, which limits the direct application of these results to external beam radiotherapy planning only.
3. Chapter 3: RAH MRI Machine Dependent Distortion Correction

3.1 Introduction

The previous chapter showed that the spatial linearity phantom built was useful in terms of characterising the distortion present in the magnetic resonance imaging system. It was able to be used to map distortions throughout the imaging volume in the magnet bore by scanning multiple spatial linearity plates at off axis z-positions. This chapter describes the development of a geometric image correction algorithm based on the distortion results from the spatial linearity phantom. This algorithm effectively removes the machine dependent distortion transverse component from the images, and its effectiveness will be discussed.

3.2 Brief Theory of Image Transformation

Shown in figure 3.1 is a summary of the various types of image transformations that map the coordinate system of one image onto another ie. \((x,y)\rightarrow(x',y')\). Note that the discussion in this section is limited to the 2-dimensional case as this is in context with the ultimate aim of this project which is to transform uncorrected MRI image coordinates into geometrically more accurate ones within a given MRI slice. One MRI slice is of course 2-dimensional, although it is acknowledged that a series of these slices makes up a three dimensional volume.

The transformation may be global (the top row of graphics in figure 3.1) or local (the bottom row). Global transformations result in the entire image space being affected by the mapping in a continuous fashion. Local transformations can affect varying amounts of the image space, and
parts of the image space may be affected entirely differently to others. Further sub-categories of global and local transformations are rigid, affine, projective and curved. These terms relate to the degree of elasticity of the transformation.

A transformation is rigid if the distance between any two points in the original and the transformed image remains the same. Rigid transformation comprise rotations, translation and reflections. Equation 3.1 is an example of a rigid transformation:

\[
\begin{bmatrix}
  x' \\
  y'
\end{bmatrix} = \begin{bmatrix}
  \cos \phi \pm \sin \phi & \sin \phi - \cos \phi \\
  \sin \phi - \cos \phi & \cos \phi \pm \sin \phi
\end{bmatrix} \begin{bmatrix}
  x \\
  y
\end{bmatrix} + \begin{bmatrix}
  t_x \\
  t_y
\end{bmatrix}
\]  

(3.1)
Where: $\phi$ is the rotation angle and $\begin{bmatrix} t_x \\ t_y \end{bmatrix}$ is the translation vector.

In fact a rotation was employed in the algorithms discussed in chapter 2 where it was desired to rotate the actual spatial linearity phantom data coordinates to match the rotation in the centre of the image spatial linearity data to account for slight rotations of the phantom when it was positioned in the MRI scanner.

In principal this type of method could be employed to register two images of the same 2D object assuming that the magnification of the images is the same i.e. a rotate/translate registration. This type of registration applied globally would be the preferred to match, for example, medical images from different modalities (e.g. CT, MRI, single photon emission tomography (SPECT)) where the scan planes within the patient were coincident (assuming that distortions in any of the modalities were clinically insignificant). Scan plane coincidence can be achieved by resampling the 3D data set (contiguous 2D slices) to produce new slices at any arbitrary plane orientation.

An affine transformation is defined by the fact that straight lines in the initial image are mapped to be straight lines in the transformed image. The 2D coordinate transformation can be expressed as:
Examples of affine transformations are uniform and non-uniform scaling and shearing. A scaling transform deals with the limitation in the example above concerning the matching of two coplanar image data sets, where the constraint was that the magnification was assumed to be the same in both images.

In projective transformations, straight lines in the first image are once again mapped to straight lines in the transformed image, except that, in general, parallelism between pairs of straight lines is not preserved. Projective transformations are examples of linear transformations in a higher dimensional space. An example used in radiation therapy is the production of digitally reconstructed radiographs (DRR’s), where a three dimensional image data set (consisting of eg. contiguous CT slice information), has ray-lines passed through it from a virtual source and the projection onto a plane of the rays exiting the data set is computed thus simulating a conventional plane radiograph. Projection transformations can be represented by:

\[
\begin{bmatrix}
  x' \\
  y'
\end{bmatrix} = \begin{bmatrix}
  a_{11} & a_{12} \\
  a_{21} & a_{22}
\end{bmatrix} \begin{bmatrix}
  x \\
  y
\end{bmatrix} + \begin{bmatrix}
  t_x \\
  t_y
\end{bmatrix}
\]
\[
\begin{bmatrix}
x' \\
y'
\end{bmatrix} = \begin{bmatrix} u/w \\ v/w
\end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33}
\end{bmatrix} \begin{bmatrix} x \\ y \\ 1
\end{bmatrix}
\] (3.3)

Where \( w \) represents a homogeneous coordinate.

Curved transformations or warps enable mapping of straight lines onto curves. A simplistic representation of such a transform is:

\[
(x', y') = F(x, y)
\] (3.4)

Where \( F \) can be any mapping function. The mapping function can be of a variety of forms, and one example is of the class polynomial. A 2D polynomial can be represented by:

\[
x' = a_{00} + a_{10} x + a_{01} y + a_{20} x^2 + a_{11} xy + a_{02} y^2 + \cdots
\] (3.5a)

\[
y' = b_{00} + b_{10} x + b_{01} y + b_{20} x^2 + b_{11} xy + b_{02} y^2 + \cdots
\] (3.5b)

It is clear that the distortions described in chapter two for the RAH MRI scanner exhibit local and curved variations of distortion, and that global transformations will not provide a solution to the image warp problem. This leaves local curved transformations where the mapping function \( F \) needs to be applied to local subsets of the entire image space, and will be based on information from the spatial linearity phantom which provides sufficient information to describe \( F \). This is the type of method which has been employed by others and examples will be given in the next
3.3 Examples of Distortion Corrections Successfully Applied to Medical Images

In 1987 a group in Heidelberg, Germany first published results of a method they developed for image distortions for use in stereotactic neurosurgery, and they made the point that image distortion correction is a prerequisite for this work (Schad et al, 1987) and followed it up with another paper in 1992 (Schad et al, 1992) which describes the same technique, but this time utilised in magnetic resonance angiography (MRA) for the treatment of arteriovenous malformations (AVM's). Details of MRA and its developments can be found elsewhere (Battocletti et al, 1981, Salles-Cunha et al, 1981, Wedeen et al, 1985 Laub et al, 1988 and Dumoulin, 1986). The papers (Schad et al, 1987 and Schad et al 1992) describe distortion correction based on a fourth order, two dimensional polynomial. They scanned a phantom which was very similar in construction to the RAH spatial linearity phantom and contained M (=249) spatial linearity points. The assumption was also made that the distortion at the centre of the image was nil and this became their origin, in a similar way that the origin was chosen in chapter 2 of this thesis for overlaying the distorted image and the phantom spatial linearity coordinates. The equations describing their polynomial correction are:

\[ u_k = \sum_{ij=0}^{N} U_{ij} \cdot x_k^i \cdot y_k^j \]  
\[ v_k = \sum_{ij=0}^{N} V_{ij} \cdot x_k^i \cdot y_k^j \]  

(3.6a)  
(3.6b)
with \( k = 1 \ldots M \).

Note: \( N = 4 \) in this case. See also equations 3.5 a) and b) for expanded form, where

\[
u, v = x', y'
\]

The spatial linearity points are represented by \((x, y)\), and their distorted positions within the image were \((u, v)\), with the origin of the \(xy\)-coordinate system coincident with that of the \(uv\)-coordinate system, and that point is in the geometric centre of the image. \( M \) is the number of spatial linearity points as previously mentioned and \( N \) is the expansion order of the 2D polynomial. Their selection of \( N \) was a compromise between computational burden and distortion reduction. They did not mention, but it is a consequence of using higher order polynomials that they can tend to oscillate unpredictably, and so if they had used an order much more than 4 (the value used) then this could have introduced greater distortions.

Equation 3.6a and 3.6b were applied to all \( M \) pin positions which formed a system of linear equations which allowed the coefficients \( u_y \) and \( v_y \) to be calculated and this enabled the distortion to be corrected.

Their correction method achieved geometric accuracy for head scans that was only limited by the pixel size (in their case \(~1 \text{ mm}\)). The pixel size for the RAH MRI for large fields of view (eg. thorax, abdomen and pelvis) is \(~2 \text{ mm}\). They only corrected their images in an axial or 2D sense, but then verified the correction with an elaborate 3D phantom which enabled them, via an iterative process which involved reshimming the magnet, to achieve very close to true axial scans restricting distortions in the \(z\)-direction to around 1.0 mm which is necessary for stereotactic
work. For this thesis involving larger field of view scans, no such 3D phantom was constructed due to the fact that clinical slices are 10 mm thick the z-distortions are not so significant in this context.

Another technique was proposed by Moshfeghi (Moshfeghi, 1991) which was to elastically match contours in the images from one modality to another. Contour information was a prerequisite for this technique, and so restricted it from being used as a general method for distortion correction.

A paper was published in 1991 (Boone et al, 1991) which described a method of distortion correction applied to image intensifier images from digital planar systems in radiology. A spatial linearity device with a grid pattern similar to the RAH phantom was employed to aid the distortion correction process. The distortion correction method was an unwarping technique.

Nearest neighbour techniques were employed to match the position of the spatial linearity points with the corresponding points in the distorted image (ie. identical to the process described in chapter 2 of this thesis which enabled the distortion assessment). The image of the spatial linearity points was broken down into the smallest possible triangular sections with a spatial linearity point at each apex. This is to facilitate the description of local regions in which to carry out local distortion corrections. Each triangular region had the following transformations defined:
\[ x' = \alpha_x x + \beta_x y + \epsilon_x \]  
\[ x' = \alpha_x x + \beta_x y + \epsilon_x \]  
\[ x' = \alpha_x x + \beta_x y + \epsilon_x \]  
\[ y' = \alpha_y x + \beta_y y + \epsilon_y \]  
\[ y' = \alpha_y x + \beta_y y + \epsilon_y \]  
\[ y' = \alpha_y x + \beta_y y + \epsilon_y \]  

For each triangular region the values for \( \alpha_x, \beta_x, \epsilon_x, \alpha_y, \beta_y, \epsilon_y \) were solved algebraically, by substituting in the spatial linearity coordinate values \((x_i, y_i), (x_r, y_r), (x_t, y_t)\) from the phantom and the corresponding coordinate values \((x'_i, y'_i), (x'_r, y'_r), (x'_t, y'_t)\) of the spatial linearity points in the distorted image into the equations 3.7a-f. The solved values were used to transform all pixels within each triangular region in the original distorted image \(I_{\text{distorted}}\) to their corresponding locations in the corrected image \(I_{\text{corrected}}\).

This transformation was carried out by mapping the grey scale values at each position, \((x', y')\) in \(I_{\text{distorted}}\) to position \((x, y)\) in \(I_{\text{corrected}}\) for each triangular region. This is known in image processing theory as image resampling (Wolberg, 1990).

This is a convenient method to develop computationally because it only involves solving two sets of three linear equation systems, even though this has to be done for every single triangular region of the image. The system is also able to extrapolate to pixels outside of the spatial linearity matrix (i.e. points which are not surrounded by three points which form a triangle). These were the main reasons for choosing to adopt this method of distortion correction for the work in this thesis. The next section details this development and describes the computer algorithms developed to perform the task of geometric correction.
3.4 Method Adopted for Distortion Correction at the RAH

Figure 3.2: Method of Boone (Boone et al, 1991) applied to the RAH distorted image data, in order to perform warp corrections.

Shown in figure 3.2 is a diagrammatic representation of how the distortion correction algorithm that was developed works. Full details will be presented in the next section, but it is useful to include an overview.

The graphic in the figure is an overlay of $I_{\text{distorted}}$ (the MRI image), and $I_{\text{corrected}}$ (the geometrically correct image that will result after applying the resampling technique). The spatial linearity points in $I_{\text{distorted}}$ are shown as the bold crosses, and the spatial linearity points in $I_{\text{corrected}}$ are shown as the circles. The grey square represents the pixel in $I_{\text{corrected}}$ that needs to be filled with a grey scale value which has been chosen from the appropriate location in $I_{\text{distorted}}$. That location in $I_{\text{distorted}}$ is represented by the smaller cross. The algorithm finds the nearest three spatial linearity
points to the pixel to be given a value in $I_{\text{corrected}}$, and by a nearest neighbour technique identifies the corresponding three spatial linearity points in $I_{\text{distorted}}$.

This provides sufficient information to enable the solution of equations 3.7a-f for the six unknown parameters $\alpha_x$, $\beta_x$, $\epsilon_x$, and $\alpha_y$, $\beta_y$, $\epsilon_y$. Equations 3.8a and 3.8b have the appropriate parameters substituted into them which is the coordinate required in $I_{\text{distorted}}$ from which to take the pixel magnitude and transfer it to $I_{\text{corrected}}$.

\[
x' = \alpha_x x + \beta_x y + \epsilon_x \quad (3.8a)
\]
\[
y' = \alpha_y x + \beta_y y + \epsilon_y \quad (3.8b)
\]

Within a triangular region the system of linear equations works in such a way that the nearer the pixel to be filled in $I_{\text{corrected}}$ is to a spatial linearity point pair, the more that point pair dictates the position of the resampling coordinate. However, if the pixel to be filled in $I_{\text{corrected}}$ is at the centre of the triangle, then the resampling coordinate is determined in equal weights by the three surrounding spatial linearity point pairs at each apex of the triangle.

When the image resampling point in $I_{\text{distorted}}$ is estimated a sub-pixel linear interpolation step is carried out in order to meet spatial sampling frequency requirements. This is necessary because some areas are reduced in size in the transformation from $I_{\text{distorted}}$ to $I_{\text{corrected}}$, so they are undersampled, which is in technical violation of sampling theory (Boone et al, 1991). Also the resampling position in $I_{\text{distorted}}$ will not in general correspond with the centre of a pixel, and so the magnitude of surrounding pixels needs to be taken into account.
The algorithm loops over every pixel in $I_{\text{corrected}}$ and resampling is governed always by the nearest three spatial linearity points. Obviously as the resampling proceeds, then new combinations of spatial linearity points are used and every time this is required, equations 3.7a-f need to be resolved. This accounts for the faceted nature of the surface plots (e.g., figure 3.3) (see later).

The actual implementation is done with two algorithms. The first called “dist_cor.c” which is listed in appendix F, creates a binary file of coordinates, which are derived as discussed above, for resampling a distorted image. The file structure is outlined in lines 7-20 in appendix F. The new aspects of this algorithm (i.e., ones not discussed in chapter two) will be presented in the next section. The second algorithm (“resample.c”) actually carries out the resampling of any image (using the coordinates derived by “dist_cor.c”) and a listing is given in appendix G. This will also be discussed in a later section.

3.5 Distorted Image Resampling Coordinate Program (refer to appendix F)

The first part of the algorithm operates once again on any spatial linearity image of the RAH phantom (see figure 2.7). It performs the same spatial linearity coordinate search (lines 24-530) as was described in chapter 2, section 2.4, in reference to appendix A. The routine synchronise_arrays (lines 531-583) is once again used and functions as described in section 2.6.1.

Routine “resampling_transform()”; line 588

This subroutine sets up memory space for two arrays (trans_x and trans_y) (lines 598-613) which will ultimately contain the x and y coordinates respectively for resampling a distorted image for the creation of a geometrically correct one. For every pixel in $I_{\text{corrected}}$ there will be a unique value
in each of trans_x and trans_y.

Now for every pixel in the spatial linearity image $I_{\text{distorted}}$, the following occurs: The pixel is converted to Cartesian coordinates (lines 619-621) with the origin centred on the spatial linearity point that is nearest the centre of the image. Next the nearest three spatial linearity points in $I_{\text{corrected}}$ are found to break the space up into triangular regions. Some peaks are rejected if there was no corresponding point found in $I_{\text{distorted}}$. This sometimes occurs if the spatial linearity plate of the phantom is not fully included in the scanned slice, in which case some spatial linearity points disappear and can't be detected. In this case the next nearest spatial linearity point is found until all three points in $I_{\text{corrected}}$ have corresponding spatial linearity points in $I_{\text{distorted}}$ (lines 630-649).

Another trap is set if the three nearest points lie along a straight line, as this means that there is no solution to equations 3.7a-f (lines 653-668). If the points are co-linear, then a third point is found which does not breach this condition (lines 669-720).

At this point there is enough information to define and solve equations 3.7a-f. The method for the solution is the well known one of Gauss elimination and back substitution. This method provides a general solution for simultaneous equations which lends itself to computer program implementation (lines 721-869).

This entire process is carried out for all $256 \times 256 = 65,536$ pixels in $I_{\text{corrected}}$. The resultant coordinates are then written to a binary output file (lines 870-896) for use in the resampling program which will be discussed next. Note that the output file is stored as integers (to avoid producing a massive file), but the decimal coordinates before being placed in the output file are multiplied by 100, giving an effective coordinate resolution of $1/100$ mm. The run time for this
program on an IBM compatible computer with a pentium processor running at 100 MHZ is approximately 1½ minutes for each spatial linearity image. Note that although this represents a significant amount of time to process all slices, the process only has to be done once (until the magnet is re-shimmied or a routine recreation of the distortion correction is made).

3.6 Distorted Image Resampling Coordinate Program Results Analysis

![Central Image Resampling X-Coordinate Change](image)

**Figure 3.3:** Central image resampling coordinate x-position difference dependence on pixel location in $I_{\text{corrected}}$. 
Central Image Resampling $Y$-Coordinate Change

![Graph showing central image resampling coordinate y-position difference dependence on pixel location in $I_{corrected}$.](image)

**Figure 3.4:** Central image resampling coordinate y-position difference dependence on pixel location in $I_{corrected}$.

The program was executed on the image of figure 2.7. The output from the program (ie. the resampling coordinates in $I_{distorted}$) was analysed with a small piece of c-code, which calculated for given pixel locations in $I_{corrected}$, the physical distance (in millimetres) away in x and y separately it is to the resampling position in $I_{distorted}$. A greater value of this distance implies that there is more distortion present in this region of the image. This type of analysis gives one a feel for the nature of the distortion correction that will be performed on MRI images.
The first result is for the same section of the image as shown in figure 2.9. The plot of x-coordinate resampling position difference is shown in figure 3.3. This graph gives an indication that the distortion correction regime is going to be continuous, and due to the faceted nature of the surface in the figure, the method takes regional distortion variations into account. The resampling coordinate difference distance in this central portion of the image is also relatively small, \( \sim \pm 1.0 \) mm, which is predictable because the actual magnitude of geometric distortion here is small also.

**Image Periphery Resampling X-Coordinate Change**

![Image Periphery Resampling X-Coordinate Change](image)

**Figure 3.5:** Top centre image resampling coordinate x-position difference dependence on pixel location in \( I_{\text{corrected}} \).
Similar results are presented for the same top centre section of the image as shown in figure 2.8. This was done because it shows the effects of extrapolating the distortion correction outside the region where spatial linearity points exist and the effects where significant distortions are present as shown in figure 2.13. A y-pixel index of 0 is the very top of the image, and therefore as the y-pixel value increases, this represents positions in a direction towards the bottom of the image. Similarly an x-pixel increase represents a direction in the image of left to right. X-coordinate resampling position difference is shown in figure 3.5.

Just considering the y-pixel index for the moment, one can see that in this direction, as we decrease from a pixel value of 40 down to 25 (the regions at the top of the image where the last spatial linearity points, and hence where explicit knowledge of distortion is) and the decreasing further to 0 (regions where distortion assessment is by extrapolating the spatial linearity results) that there is a steady increase in the x-coordinate resampling position difference, and so the extrapolated section behaves predictably, and would be a reasonable first approximation to peripheral distortion. This can be understood by looking at figure 2.13 which shows \( I_{\text{corrected}} \) overlaid with \( I_{\text{distorted}} \), indicating magnitude and direction of geometric distortion. With reference to the 4 spatial linearity point pairs in this figure which are identified as the second and third pairs from the left in the top row of six, and the two pairs immediately below them. It is this region that figure 3.5 corresponds to (with the extension to the very top of the image of course). It is clear that as we move from the bottom pair of spatial linearity coordinate pairs to the top pair in figure 2.13, that the position of the 4 points in \( I_{\text{distorted}} \) (the diamond symbols) move more to the right (the positive x-direction) of the corresponding points in \( I_{\text{corrected}} \), hence explaining the negative slope in the x-coordinate resampling difference with the increasing y-pixel index position of figure 3.5.

More dramatic (due to y distortion being more severe in this region) but equally explainable is
**Image Periphery Resampling Y-Coordinate Change**

![3D graph showing y-coordinate resampling position difference dependence on pixel location in $I_{corrected}$.](image)

**Figure 3.6:** Top centre image resampling coordinate y-position difference dependence on pixel location in $I_{corrected}$.

The result presented in figure 3.6 for y-coordinate resampling position difference for the region identical to that represented in figure 3.5. The same 4 spatial linearity point pairs apply and it is clear that (with reference to these) the y-direction distortion needs to be corrected by resampling in the negative y-direction in $I_{distorted}$ and that this goes more negative in going from a y-pixel...
index position of 40 down to 0. Once again there is no uncontrollable resampling coordinate estimate by extrapolating the measured distortion to the periphery of the image.

X- and Y-coordinate resampling position differences were also calculated for lines in the image corresponding to the x-axis and the y-axis. These transects give a global idea of the nature of the intended distortion correction, albeit only one dimensional. The results are presented in figures

*Difference in X-Resampling Coordinate Along X-Axis of* \( I_{\text{corrected}} \)

![Graph](image)

**Figure 3.7:** Resampling coordinate x-position difference dependence on pixel position along the x-axis in \( I_{\text{corrected}} \).
Figure 3.8: Resampling coordinate x-position difference dependence on pixel position along the y-axis of $I_{\text{corrected}}$. 3.7 to 3.10.

Note that the spatial linearity points exist from approximately pixel position 25 through to pixel position 225. Outside these values resampling is based on extrapolation. As will be shown in chapter 4, the vast majority of patients will not extend beyond the regions where spatial linearity points exist, and so the extrapolation that occurs will not play a major rôle.
These series of graphs can be viewed in conjunction with figure 2.13 to explain the trends that are present. In figures 3.7 and 3.8 this can be done by studying the x-displacement of the spatial

**Difference in Y-Resampling Coordinate Along X-Axis of** $I_{\text{corrected}}$

![Graph showing Y-Coordinate Resampling Position Difference](image)

**Figure 3.9:** Resampling coordinate y-position difference dependence on pixel position along the x-axis in $I_{\text{corrected}}$.

It is clear from figure 3.9 that there are only relatively minor y-direction corrections required along the x-axis if the image in this case. Once again this can be verified in figure 2.13 where there is very little y-displacement of the point pairs that lie along the x-axis.
3.7 Distortion Correction (Resampling) Program (refer to appendix G)

This program was produced to enable resampling of any distorted image according to the relevant resampling coordinate file which resulted from the program of appendix F. In effect this removes the machine dependent geometric distortions from the image. A brief description will now be given of this algorithm.

The program requires as input an MRI image file and a coordinate resampling file generated by dist_cor.c. The output is a corrected image file, with the same header as the original image, so that, in principle, the image could be re-read by the MRI system computer for display/hardcopy.
The input and output streams are initialised (lines 65-93). A test is performed to ensure that the image to be corrected has dimension 256 x 256 pixels (lines 94-104). Also the scale factor from the image to be corrected is read, and this is compared with the scale factor that is stored in the resampling coordinate file. These have to be the same, or the resampling will be invalid (lines 105-116).

The array matrix_ima is filled with the pixel data from the image to be corrected ($I_{\text{distorted}}$) (lines 117-143). Memory space is also allocated for the array matrix_cor, which will eventually contain the geometrically corrected pixel values ($I_{\text{corrected}}$) (lines 144-160). Two other arrays are established, x_sam_coord and y_sam_coord which contain the x and y coordinates respectively for resampling the undistorted image (lines 161-182). Thus we have the first entries of x_sam_coord and y_sam_coord representing the position in matrix_ima to compute a grey scale value to place in the first entry in the array matrix_cor, and so on for all pixels in matrix_cor.

Every pixel in $I_{\text{corrected}}$ now has a grey scale value calculated. It must be checked that the resampling coordinate is within the pixel bounds of $I_{\text{distorted}}$ (lines 193-198), and if not the pixel in $I_{\text{corrected}}$ is assigned the value 0 (line 406). Once it has been established that the coordinate to select the grey scale from is within the image bounds 4 way linear interpolation is performed to estimate the grey scale values from the surrounding pixels (lines 199-411). This was briefly discussed in section 3.4 and was necessary to avoid sampling theory violation. The possibility exists for the pixels in $I_{\text{distorted}}$ to contain regions of interest (ROI) information. If they do the underlying pixel magnitudes need to be extracted (lines 222-351). Any ROI's that were set in $I_{\text{distorted}}$ are transferred to the geometrically correct pixel locations in $I_{\text{corrected}}$. In principle this...
enables eg. tumour boundary marking on the MRI display system for possible transfer to the radiotherapy planning process (lines 367-404). This whole process is repeated until every pixel in $I_{\text{corrected}}$ has been assigned a grey scale value.

The final step is to transfer the header from the uncorrected image to the corrected one and then load the array matrix cor into the pixel value locations, yielding the machine dependent geometrically corrected image file (lines 412-487).

### 3.8 Results of the Distortion Correction (Resampling) Program

Shown in figure 3.11 is an MRI transverse slice image of a patient in the thorax region. This is straight off the MRI system and has not been geometrically manipulated. Overlaid on the image is a regular grid pattern which will prove useful in visualising the distortion correction process.

![Figure 3.11: Uncorrected patient image with an overlaid grid pattern to enable visualisation of the distortion correction process.](image)
Figure 3.12 is the same image of 3.11 but after the program resample.c has been applied, hence performing a spatial warp. Note that the deviation of the grid lines from those in 3.11 represents the opposite of the deviation caused by the distortion (ie the regular grid of figure 3.11 is actually in distorted image space). In the central regions of the image, the distortion correction appears to be relatively well behaved. However, at the image periphery it shows evidence of instability due to the breaking up of the grid lines. This is primarily due to the extrapolation process that is applied in these regions, and which was discussed in section 3.6. This will be later shown to have little effect on actual patient scans. Image resolution may be affected in the resampling process, but due to the sub-pixel interpolations being used as described in section 3.4 this is expected to be minimal. In addition, it would be prudent to mark ROI’s (eg tumour extent) on the MRI workstation before resampling such that any loss in image quality does not compromise diagnostic information. ROI’s can be dealt with in the resampling process (see section 3.7).
The image that was presented in figure 2.7 of the central plate of the spatial linearity phantom was also spatially resampled, and the results are shown in figure 3.13. Of course this image should be rendered geometrically correct as this was used to create the resampling coordinates in the first place, which the program resample.c uses as input data. On first inspection this appears to be true, as the pattern of spatial linearity points now appears to be reasonably regular.

The plot of figure 2.13 was reproduced here as figure 3.14, but this time for the corrected image. It is obvious that in this plot the degree of spatial distortion has been significantly reduced. The actual phantom and the corrected image spatial linearity data overlay much better. The data for figure 3.14 was created using the software developed to produce the data for figure 2.13 (peak.c, appendix A) which ran equally well on the image data of figure of 3.13.

The degree of geometric accuracy achieved was also tested by applying the methods defined in
section 2.6 which were used to assess distortion in uncorrected images. The results will be presented in the following sub-sections.

**Corrected Image and Actual Spatial Linearity Points**

![Scatter plot of the geometrically corrected central phantom spatial linearity coordinates (from figure 3.13) with the spatial linearity coordinates of the phantom.](image)

**Figure 3.14:** Scatter plot of the geometrically corrected central phantom spatial linearity coordinates (from figure 3.13) with the spatial linearity coordinates of the phantom.

### 3.8.1 3-D Plot of Corrected Distortion Magnitude and Associated Cartesian Coordinates

The program dist_mag.c was re-run on the corrected image data of figure 3.13, and the results
are presented in figure 3.15. This figure should be looked at in conjunction with figure 2.14, in

3D Scatter Plot of Distortion Magnitude

(Corrected Image)

Figure 3.15: 3-D scatter plot of all spatial linearity points in the corrected image (figure 3.13) with their respective residual distortion magnitudes.

order that the improvement in the distortion magnitude can be observed. The distortion magnitudes for the corrected image have been clearly reduced with a new maximum of 1.9 mm, mean value of distortion of 0.69 mm and the standard deviation of the distortion of 0.33 mm (cf. 11.6 mm, 3.4 mm and 2.6 mm respectively for the uncorrected image). This is quite acceptable given that the pixel size in the images is only 1.98 mm.
3.8.2 AAPM Corrected Image Residual Distortion Assessment

This is a repeat of the analysis carried out in section 2.6.2 to assess percentage distortion as per equation 2.3. The results for the corrected image were: maximum percentage distortion = 4.6%,

mean percentage distortion = 0.39% and the standard deviation of the percentage distortion = 0.47% (cf. 4.6%, 1.4% and 1.0% respectively for the uncorrected image).

**Figure 3.16:** Corrected vs uncorrected maximum distortion detected within origin-concentric annuli of width 40 mm and with mean position 20, 60, 100, 140, 180 and 220 mm from the origin.
The reason that the maximum percentage distortion as defined by the AAPM was not improved for the corrected image was because this related to a spatial linearity point that was adjacent to the origin, and hence even a small error in position will result in a significant percentage distortion value with respect to the origin. It is clear, however, that the mean value and the standard deviation have been significantly improved with the use of the resampling correction algorithm.

3.8.3 Corrected Image Annular Distortion Assessment

Annular distortion assessment was performed for the corrected image of figure 3.13 as described in section 2.6.3, and the results are presented in figure 3.16. The annular distortion from the uncorrected image is included for comparison. Once again it is clear that the residual distortion is less than the pixel dimension which is also marked on the graph.

3.8.4 Corrected Image Distortion Magnitude Distribution

Distortion magnitude distribution as described in section 2.6.4 was re-assessed for the corrected image of figure 3.13 and is shown in figure 3.17. The frequency distribution data for the corrected image is now restricted to the sub-pixel dimension, indicating a clear improvement. The data for the corresponding uncorrected image is included to allow direct comparison.
Distribution of Distortion Magnitude

(Uncorrected vs Corrected Images)

Figure 3.17: Frequency distributions, for the uncorrected image of figure 2.7 and the corrected image of figure 3.13, of the percentage of the total number of spatial linearity points having distortion magnitudes corresponding to 0.25 mm wide scoring bins.

3.8.5 Corrected Image Off Axis Dependence of Absolute Distortion and AAPM Percentage Distortion

This is the same analysis as was carried out in section 2.6.6 but this time for corrected images. This was done because it needs to be shown that the method being described is effective at correcting machine dependent distortions at off axis positions to ensure that it will be applicable
to patient slice data sets which contain off axis data. Geometric corrections were applied to various off axis positions based on the spatial linearity distortion results obtained from those slice positions. The results are presented in table 3.1.

**Table 3.1:** Distortion assessment results for corrected images for different off axis positions.

<table>
<thead>
<tr>
<th>Off Axis Distance (mm)</th>
<th>Distortion Magnitude</th>
<th>AAPM Distortion</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Maximum (mm)</td>
<td>Mean (mm) +Std.Err.</td>
</tr>
<tr>
<td>0</td>
<td>1.86</td>
<td>0.69±0.02</td>
</tr>
<tr>
<td>25</td>
<td>2.28</td>
<td>0.81±0.03</td>
</tr>
<tr>
<td>50</td>
<td>3.39</td>
<td>1.45±0.05</td>
</tr>
<tr>
<td>125</td>
<td>1.71</td>
<td>0.74±0.03</td>
</tr>
</tbody>
</table>

Table 3.1 shows that there is only minor changes to mean distortion magnitude in the corrected spatial linearity images with off axis position. The same is true for the mean percentage AAPM distortion. The overall magnitude of all results in this table are significantly less than those for the uncorrected images of table 2.2.

### 3.8.6 Corrected Image Off Axis Dependence of Annular Distortion

This was studied as it was in section 2.6.7 again for the corrected spatial linearity images at off axis positions. The reduction in maximum distortion within given annuli is quite marked when the results of figure 3.18 are contrasted with those of figure 2.21.
**Off Axis Annular Distortion**

*(Corrected Images)*

![Graph showing annular distortion variation with off-axis scanning slice.]

**Figure 3.18:** Corrected spatial linearity image annular distortion variation with off-axis scanning slice.

It is clear that the effectiveness of the correction algorithm is resulting in observed distortions which are generally of the order of, or less than the pixel dimension which is 1.98 mm in this case.
3.8.7 Corrected Spatial Linearity Distortion Magnitude Distribution Dependence on Off Axis Position

The same method to produce the frequency distributions described in section 2.6.4 was used here to display the dependence of this parameter on off axis position. Data for 0 mm and 125 mm off axis positions are presented in figure 3.19.

![Distribution of Distortion Magnitude](image)

**Figure 3.19:** Corrected images: Off axis variation of distortion frequency distribution (0.25 mm wide scoring bins).

Comparing this result with that of the graph of figure 2.22 which is for uncorrected images, once
again the effectiveness of the distortion correction algorithm is shown.

3.8.8 Conclusions

This chapter has described the development of a method to successfully remove the machine dependent distortions in the x-y plane using a priori knowledge from the distortion results obtained from the spatial linearity phantom. The corrections were shown to be effective for various slice positions throughout the magnet bore. Whilst temporal stability was demonstrated, it must be acknowledged that distortion integrity could be compromised at any time by for example, a small ferro-magnetic object becoming stuck in the bore, or a shimming power supply becoming unstable. Ongoing QA procedures would have to be implemented by way of regular spatial linearity assessment to ensure the reliability of any distortion correction application.

The distortion correction regime that was used has resulted in reductions of machine dependent distortion to similar magnitudes as the actual pixel size, or even better. It should be once again noted here that the method was only tested for large fields of view, and hence a relatively large pixel size (1.96 mm). The implication of this result is that in the majority of clinical situations involving large fields of view where this correction technique may be used, that the bulk of the distortion will be removed. Susceptibility and chemical shift distortions as discussed in chapter 1 (section 1.6.1) are likely to be small compared to the other distortion causes, and so would be able to be considered as second order effects which in turn may enable them to be ignored form the point of view of external beam radiotherapy. This is an assertion that would have to be tested for a number of clinical situations where CT data was available to benchmark the geometrically transformed MRI information against. This would enable assessment of the magnitude of this residual distortion, and whether or not it will compromise the intended aim of the radiotherapy treatment by ignoring it.
The situation will be different for stereotactic situations which typically involve smaller fields of view (hence smaller pixel sizes). It will still be possible to remove the machine dependent distortions in x-y plane for these slice sets, with the aid of an appropriate spatial linearity phantom (smaller cross section than the one used in this project) to an accuracy again approaching the pixel size, but the secondary effects of distortion due to chemical shift or susceptibility may not be insignificant here. Also a similar method to that described by others (Schad et al, 1987 and Schad et al, 1992) would have to be employed to ascertain that the presumed angulation of the slices (they may not always be transverse) was correct which is necessary in order that there are no significant z-distortions present (something that was ignored in this project due to the slice thicknesses being used of 10.0 mm). If other than transverse planes were required for the particular clinical procedure being undertaken, the transverse slices could be geometrically corrected and then reconstructed to from the desired slice planes. It would not be recommended to attempt to geometrically correct slices that were actually acquired obliquely from the MRI scanner, as the distortion that resulted from this type of acquisition could be different from true axial data collection (which is the mode used to obtain the distortion correction information).

The fact that it appears to be possible in principal, to significantly improve the geometric accuracy of the MRI data available for external beam radiotherapy treatment planning using the spatial correction method developed in this chapter, would allow more confidence to be given to the radiation oncologists about true position of tumours within the patient’s body. By transferring the MRI derived, geometrically correct, tumour delineation data to the planning CT information (which will have pixel by pixel dose calculations performed on it) will allow absorbed dose display to ensure adequate coverage of the intended treatment area. This would reduce the possibility of a geometric miss of the tumour (as described in section 1.7).
4. Chapter 4: Clinical use of the Distortion Correction Technique

4.1 Introduction

In this chapter will be described the application of the software developed in chapters 2 and 3 to a limited study of one patient who underwent radiotherapy treatment to the thoracic region at the Royal Adelaide Hospital and who had treatment planning work-up which consisted of both CT and MRI scan data being collected. The opportunity was taken to assess the applicability of the algorithms for distortion correction for real patient MRI data sets.

Comparison of the machine dependent distortion corrected MRI information with the geometrically accurate CT information\(^4\) was then able to be performed to enable a direct analysis as to whether the correction of machine dependent distortion alone on the MRI data is adequate for the purposes of radiotherapy treatment planning, at least in the context of thoracic imaging. A surface marker system was devised and implemented to provide quantitative information on transverse slice position along the z-axis for both scanning modes. This was useful from the point of view of ensuring that the slices being compared for geometric accuracy correspond to the same position within the patient’s body.

4.2 External Surface Fiducial Marker Plate

Fiducial marker systems are most commonly used in conjunction with stereotactic procedures, providing quantitative information which enables image data sets (usually transverse axial tomographic images from a variety of sources) to be mapped to a standard stereotactic coordinate

\(^4\)CT geometric accuracy was subject to routine quality assurance assessment by the clinical physics group at the RAH, and no distortion greater than 1.0 mm was recorded.
system. The result of this is that any point in the image data set can be transformed to a coordinate in the stereotactic coordinate system. These marker systems are mounted on rigid frames which are generally clamped directly to the patient's skull and can provide full three dimensional positional information. Further details on these can be found elsewhere (Lunsford et al, 1986 and Peters et al, 1986).

For this project only, a limited fiducial marker system was required which was designed to enable assessment of relative z-position of MRI and CT transverse scans to be determined. The assumption was made that the MRI and the CT slices were coplanar (both were obtained in a "true" trans-axial sense), which simplifies the design requirements for the fiducial system. To be strictly rigorous this assumption would have to be tested, requiring a phantom similar to that described elsewhere (Schad et al, 1987 and Schad et al, 1992) to be constructed. This was not attempted due to time constraints. It was necessary that the fiducial system was able to be quickly positioned on the patient, that it not interfere with the patient positioning and that the markers are able to be seen on both CT and MRI images. A picture of the device used is shown in figure 4.1.

![Figure 4.1: Locally produced fiducial marker plate.](image-url)
Radio-opaque catheters filled with CuSO₄ were used in the device such that they would be visible on x-ray CT scans, and also will produce a signal in MRI scanning hence making them detectable for this mode as well. The catheters were placed in milled slots in a triangular shaped, 6.0 mm thick piece of acrylic sheet to form a "vee" and another catheter was inlaid such that it was parallel to one of the legs of the "vee". The marker plate was placed on the anterior surface of the patient such that it covered the extent of the body that was to be scanned. The apex of the "vee" was orientated towards the patient's head. The overall useful length of the plate is about 280 mm. The plate was placed in the same position relative to the patient for each of the MRI and CT scans. This was achieved by making use of a localising tattoo that was on the anterior surface of the patient, and positioning the plate as accurately as possible relative to it.

It is clear that as successive scans are performed, the distance between the legs of the "vee", where they intercept the transverse slice planes, will change. This change of distance will be able to be used (as shown in figure 4.2) to calculate relative slice offset.

![Diagram of catheter and acrylic plate](image)

**Figure 4.2:** Description of the use of the RAH fiducial marker plate. The value of $d_1/d_2$ is calculated for each CT and MRI slice to determine relative transverse slice offset.

The reason that the ratio $d_1/d_2$ was used is that it provides an index of relative slice z-position...
which is independent on the magnification of the images that they are being measured from (unlike if, say, the variation in \(d_2\) were used). This makes it possible to measure \(d_1\) and \(d_2\) on hard copy films, which can for different modalities, have different magnifications.

4.3 Patient CT Scans

A patient was imaged on the Toshiba (TCT-900S) CT scanner at the RAH with the fiducial marker plate in position. Scanning was in the thoracic region (the patient was suffering from a cancer of the oesophagus) and resulted in a contiguous set of 10 mm thick slices. The plate and the catheters can be seen in one of the scans in figure 4.3.

The ratio \(d_1/d_2\) was measured from the hard copy films of the CT slices and this was graphed against the nominal slice z-position that the CT scanner assigns to the slices. This result is shown in figure 4.4, and is as expected. Mathematically the curve can be represented by the following:
\[ d_1 = \text{const} = c, \quad d_2 = -kx + b \]

\[
\frac{d_1}{d_2} = \frac{c}{-kx + b} = \frac{c}{kx - b} = \frac{c}{k\left(x - \frac{b}{k}\right)} = \frac{c}{B\left(x - \frac{b}{k}\right)}
\]

Where \(c, k, b\) are positive

This is the negative of a hyperbolic function, shifted left by \(\frac{b}{k}\).

Ultimately the corrected MRI data set will have \((d_1/d_2)\) assessed to enable synchronisation of the transverse slices from both modalities. This will be described later. It can be seen in figure 4.4 that the ratio \(d_1/d_2\) increases relatively smoothly with increasing nominal slice position.

At this stage the opportunity was taken to assess the accuracy of the nominal slice position for the CT scanner. This can be done quite simply from the fiducial marker plate measured data, and involves graphing the parameter \(d_2\) vs nominal slice position. One would expect a linear relationship for this graph which is shown in figure 4.5.
**Figure 4.4:** Graph of the ratio \( d_1/d_2 \) versus nominal slice position for a patient scanned on CT with the fiducial marker plate in position. Data was fitted with a second order polynomial.
Patient CT Slice Series with Fiducial Marker Plate

Figure 4.5: Plot of the distance \( d_z \) as measured from the CT scanner’s display console (actual dimension (mm)) versus the nominal slice position.

The slope of this straight line is also of quantitative use. The slope should theoretically be given by:

\[
\frac{\Delta d_z}{\Delta z} = 2 \tan \frac{\theta}{2}
\]  

(4.1)

Where \( \theta \) is the angle between the catheters which define \( d_z \). In this case this angle is 45°, giving
a slope of 0.8284 (cf 0.8273 from the graph).

Note that for figure 4.5, the distance \( d \) was measured using software utilities on the CT scanner console, such that the dimension assessed in this case was actual. This was done to enable the slope of the linear regression line to be compared to the theoretical slope calculated using equation 4.1. The two values are in close agreement which indicates that the CT table movement is accurate, and the value of \( r^2 \) (0.9995) indicates that it is linear as well. If this were not the case, the fiducial marker system could have been used on the CT scanner data to calculate the true z-offset of the slices. This will be seen to be useful for the MRI data for the patient which will be dealt with later on.

### 4.4 MRI Spatial Linearity Scans

On the morning of the planned patient MRI scanning (the same patient as mentioned in section 4.3) a spatial linearity scan of the phantom was performed to provide the most up to date distortion information. Scanning consisted of obtaining data for the 11 spatial linearity plates (T1 weighted scans \( TR = 700 \) ms, \( TE = 15 \) ms, slice thickness 10.0 mm with a 500 mm field of view, preparation gradient (0.47 T cm\(^{-1}\)) in the y-direction and the readout gradient (0.63 T cm\(^{-1}\)) in the x-direction) with the central plate of the phantom placed at the nominal magnet bore origin, as defined by the external light beam alignment system. These same scan parameters were applied to the patient later on that day as was the method of positioning for the reference point of the fiducial marker system (which was placed on the patient) in the centre of the magnet bore.

All resultant slices from the phantom were dealt with using the methods of chapter 3 (running program “dist_cor.c”) which resulted in a set of resampling coordinates to allow machine dependent geometric distortion correction to be achieved for any subsequent patient slice planes.
which were coincident with the spatial linearity plate positions.

Clearly not all patient slices will be coincident, as the MRI spatial linearity scans were non-contiguous, with a slice separation of 25 mm. This necessitated the development of an algorithm which enabled, via linear interpolation between adjacent spatial linearity slice planes, the resampling coordinates to be derived for any nominal slice offset z-position. The code listing for the program “interpol.c” which performs this task is given in appendix H.

The user has to define the two output files of the program “dist_cor.c” which have z-coordinates as read from the MRI spatial linearity scans which are adjacent to the patient MRI scan to be geometrically corrected. These two files are opened for binary reading (lines 24-41). The output file is opened for binary writing which will contain the interpolated resampling coordinates for the patient MRI file which requires the machine dependent distortions to be removed (lines 42-49).

Next a distance weight factor for the z-coordinate as read from the patient file requiring correction needs to be input to the program (lines 50-54). This weight factor is calculated as follows:

\[
WF = \frac{z(\text{patient}) - z(sl_{lower})}{z(sl_{upper}) - z(sl_{lower})}
\]  

Where:

\(WF\) = the required distance weight factor.
\( z(\text{patient}) \) \ = \ z\)-coordinate as read from the MRI patient scan requiring geometric correction.

\( z(\text{sl}_{\text{lower}}) \) \ = \ z\)-coordinate as read from the nearest (in \( z \) terms) MRI spatial linearity scan which has a lower \( z \)-coordinate than that of \( z(\text{patient}) \).

\( z(\text{sl}_{\text{upper}}) \) \ = \ z\)-coordinate as read from the nearest (in \( z \) terms) MRI spatial linearity scan which has a higher \( z \)-coordinate than that of \( z(\text{patient}) \).

Linear interpolation (taking into account the distance weight factor) is then performed between the two bounding spatial linearity phantom derived resampling coordinate sets to yield a further resampling coordinate set at the required \( z \)-slice position to enable correction of the patient image, and this is output to a file (lines 55-80).

Figure 4.6: Patient T1 weighted MRI scan (TR = 700 ms, TE = 15 ms).
This allows the program "resample.c" to be run to correct the patient slice. This process has to be repeated for all patient slices requiring correction.

4.5 MRI Patient Scans

The same patient as described in section 4.3 which had CT scans performed, also underwent MRI scanning later on in the same day that the spatial linearity scans were performed using exactly the same pulse and gradient parameters as for the spatial linearity scans. The methods of section 4.4 were applied to all of the MRI scans for this patient, which resulted in a slice set that was free from machine dependent geometric distortion.

Shown in figures 4.6 and 4.7 is an example of one of the MRI slices from the patient uncorrected and corrected respectively. It can be seen that the distortion correction has minimal effect on the features able to be seen in the image.

Figure 4.7: Same scan as 4.6 which has had machine dependent geometric distortions removed.
Measurements were performed on the set of MRI scans to measure the ratio \( d_1/d_2 \) for the fiducial markers as was described in section 4.3. (Note that the reproductions of the scans in figures 4.6 and 4.7 are of relatively poor resolution and do not show the fiducial markers. They were, however, visible on the hard copy films, from which the measurements were made). The result is shown in figure 4.8.

**Figure 4.8**: Graph of the ratio \( d_1/d_2 \) versus nominal slice position for the same patient scanned on both CT and MRI. Both data sets were fitted with a second order polynomial. The MRI data set was offset by 9.0 mm from the nominal value which was read from the scans to give coincidence with the CT curve.
It would appear from the graph that it is only necessary to offset the MRI information by 9.0 mm to achieve slice registration. Due to the similarities in the shapes of the two curves it would appear that no differential adjustment of individual MRI slice offsets is necessary. The 9.0 mm offset arises from the fact that the light beam system which is used to place the patient at the magnet isocentre was maladjusted. This was noticed when the spatial linearity phantom was scanned earlier in the day, when the light beam alignment system was used to position the central plate of the phantom at the isocentre. The nominal slice position for the central slice was 9.0 mm. The central slice plane of the patient (plane that coincided with the CT slice obtained with a z value of 0.0 mm) was likewise placed in the centre of the magnet using this alignment system hence giving rise to the 9.0 mm offset described in figure 4.8. The limitation here of course is that true slice registration only occurred where the marker plate was positioned. Differential z-offsets that may exist elsewhere in the slice plane cannot be considered with an external marker system. Likewise, if the slices were non-coplanar (CT to MRI) as discussed in section 4.2.

With the slice sets being registered it was now possible to start to compare different slices and observe the features which are identifiable on both to assess the effectiveness of the warp correction. Figures 4.9 and 4.10 show the same section of the patient scanned with CT and MRI.

Figure 4.9: CT slice of the patient through the kidney region.
respectively.

Features that are immediately identifiable in both scans are the kidneys, vertebral body, spinal cord, aorta and liver on the left hand side of the image (actually on the patient’s right as these images are viewed from the feet looking towards the head).

Figure 4.10: Corrected MRI scan of the same plane of the patient in figure 4.9.

There are two differences to be noted also on the scans. One is that the MRI scan has arms present lateral to the body outline whereas the CT scan does not. This is due to the fact that the CT scan for the patient had to be performed with the arms raised above the head for radiotherapy treatment purposes to prevent the right and left posterior oblique fields, that were to be employed to treat the oesophagus, from irradiating the arms. It was not possible for the patient to achieve this position in the MRI scanner due to the smaller bore size. This was unfortunate for the purposes of total comparison of the images due to the fact that this difference in patient position is known to affect the geometry of body cross sections. This was unavoidable because ethical reasons prevented repeating the CT scan with the arms beside the body, as this data was not useful for the patient’s treatment. The other difference (and in fact another limiting factor for the
Figure 4.11: Resultant overlay of the CT and MRI slices from the same plane of the patient.

Purpose of cross section comparison is that the CT scan was obtained on a flat couch top, and none was available for the MRI scanner. This also has the effect of distorting the anatomy. In fact this has been studied elsewhere (Herman et al, 1986) which actually proposed a warp algorithm to correct the anatomy changes.

Despite these limitations the two images were traced onto clear acetate film along with the structures that were visible on both CT and MRI and the results are presented in figure 4.11.
The two slices were matched by eye to give the closest overall co-registration by process of...
translation and rotation. It is clear that the superimposition of the two slices produces an inconclusive overall result as to residual MRI distortion (ie non-machine dependent) and its significance in the context of this method of geometric correction, due to the non-flat couch top of the MRI scanner, and the raised arm position which as mentioned earlier distort the anatomy.

Nevertheless, the vertebral body, kidneys, aorta and the liver provided some degree of coincidence.
The corrected MRI slice (figure 4.12) which corresponded to the CT slice of figure 4.3 was also studied. The resultant overlay of outlines for these two scans is shown in figure 4.13.

Once again this result suffers due to the curved couch top in MRI and the different arm position. The vertebral body, aorta and some sections of the lung outlines showed some degree of coincidence. Clearly more work needs to be performed on patients who have CT and MRI scans in the same body positions, with identical couch tops to unequivocally show by experiment that the corrected MRI scans coincide with the CT scans.

4.6 MRI Patient Scan Position Relative to the Spatial Linearity Data

The type of geometric correction employed in this project works best if the object occupies image space that is surrounded by spatial linearity points, such that no extrapolation of the image correction resampling coordinates is required. Shown in figure 4.14 is a diagrammatic representation of the transverse external surface contour of a patient resting on a flat couch insert, such as would be employed in radiation therapy treatment. The dimensions of the ellipse is 420 mm x 260 mm (and even dimensions of 450 mm x 290 mm would still not extend beyond the spatial linearity point field). This is a relatively large patient and as a result, at least in principle, would allow good geometric correction of a high proportion of radiation therapy patients.

4.7 Impact of Other Imaging Sequences

The limited study presented in this thesis was performed on the basis that the spatial linearity studies and the actual patient studies were performed using exactly the same spin echo MR pulse sequences and field gradient strengths. This was done to avoid introducing any other variables into the mechanism which influences geometric distortion. Investigation of the effects of other
imaging sequences was considered outside the scope of this thesis, but it has been looked at by others (Moerland, 1996) and it is instructive to briefly discuss the implications of these results.

In gradient echo imaging, distortion magnitudes (for given gradient strengths) are similar to spin echo techniques. If the gradient strengths are varied between spatial linearity scanning and clinical scanning, however, this can effect the amount of distortion present. Echo planar techniques can have greater distortion magnitudes associated with them (Moerland, 1996) and the distortions are more sensitive to susceptibility differences than spin or gradient echo. This fact alone may render this type of machine dependent distortion correction unacceptable, but would have to be specifically assessed to investigate this.

It is clear that varying pulse sequence (and indeed parameters within a given pulse sequence) can influence distortion. Hence it is essential to assess this for all imaging sequences that are likely to be used clinically in order that reliable distortion corrections can be performed based on previously determined spatial linearity data.

4.8 Conclusions

This chapter described the application of the distortion assessment and correction algorithms which were developed in chapters 2 and 3 for a very limited set of circumstances, and as it turned out, under sub-optimal conditions for a fair comparison of the geometrically accurate CT information with the corrected MRI data. The anatomical displacements that resulted from the different position of the patient in the MRI and CT imaging system severely compromised the assessment. Organ movement in these circumstances is very difficult to predict with any certainty. Clearly further work needs to be performed in order to unequivocally prove this method of correction, but others (Schad et al, 1987 and Schad et al, 1992) have performed similar
corrections and found them satisfactory in the perhaps more demanding (at least in terms of geometric accuracy required) area of stereotactic neurosurgical and radiosurgical techniques. This tends to reinforce the notion that machine dependent distortions are dominant in introducing geometric inaccuracies into MRI data. Nevertheless the system of correction developed in this project will be implemented and thoroughly investigated at Liverpool Hospital (New South Wales) which will be purchasing an MRI scanner towards the end of 1997, and which will provide information that will be used for targeting in radiotherapy.

The development of a surface marker system to determine z-coordinates independent of the MRI system’s z-coordinate derivation was also explained. This surface marker system was also useful for QA of the CT scanner’s z-coordinate accuracy.

Although the results of the final comparison of the corrected MRI data and the CT data was not ideal (due to the fact that any residual non-machine dependent distortion was impossible to detect due to unavoidable differences in the patient position) a method was nonetheless presented whereby the warp correction can be applied to real patients via the interpolated resampling routine, and that the slices from both modalities can be registered in terms of their z-offset. At this point it is feasible to use the method clinically subject to associated comprehensive quality assurance procedures being performed to ensure that patient treatments are not compromised as a result.

Such procedures would consist of for example regular spatial linearity tests to ensure constancy of the geometric distortion that is being corrected for. Also routine benchmarking by overlay of all corrected MRI outline and ROI information onto CT (geometrically correct (assuming QA in place for this)) data for stringent analysis of the appropriateness of the corrected MRI data. This should be done for all transverse slices, and be conformed by the treating radiation oncologist,
possibly with input from a radiologist.
5. Chapter 5: Summary and Future Work

5.1 MRI use in Radiotherapy and Stereotactic Neurosurgery

MRI has shown itself to be a useful tool in medical diagnosis due to its ability to compliment other imaging modalities by offering unique diagnostic information. This is especially true for brain or central nervous system (CNS) imaging, where disease processes (such as neoplasms) are better delineated with MRI. Some of these CNS lesions are focal in nature and therefore lend themselves to surgical excision (for operable areas) or radiosurgical treatment involving accurate placement (using stereotactic techniques) of radiation beams in order to concentrate dose, hence obliterating diseased tissue, whilst minimising the damage to surrounding normal tissues. For smaller lesions, techniques involving direct and precise (sub-millimetre accuracy) introduction of a stereotactically guided probe possessing a tip which can produce high intensity microwaves to destroy tissue in its immediate vicinity (in a controlled fashion) are used.

MRI information is presently used in a significant number of stereotactic neurosurgical procedures in conjunction with CT scanning. Some institutions have assessed the geometric distortion associated with the MRI system that is used in these cases and have found displacements which are small enough to ignore for this work. Conversely others have reported distortions which are of sufficient magnitude to force the introduction of a distortion correction technique similar to the one described in this thesis. It is clear, therefore, that all institutions need to assess their MRI system initially using methods similar to those described in chapter 2, to ensure the geometric accuracy of the MRI image data. This assessment also needs to be part of a regular quality assurance programme, as spatial linearity performance can vary due to instrument drift or malfunction.
As far as use of MRI in radiotherapy treatment planning at other sites involving larger body cross-sections goes, there is a higher likelihood of requiring geometric correction of the images and so the techniques of chapters 3 and 4 provide a mechanism whereby it is possible (at least in principle) to incorporate geometrically correct MRI information, which may enhance the radiation oncologist’s ability to accurately delineate and hence treat tumours. As a minimum, the degree of distortion should be assessed for each MRI system that is proposed to be used for general radiotherapy procedures, and the magnitude of the distortion found should be carefully evaluated with respect to the likely effect that it will have on the accurate targeting and treating of the neoplastic disease.

Diagnostically MRI has been recognised to be better than CT in some cases involving liver tumours (Boechat et al, 1988), rectal cancer recurrence (Krestin et al, 1988), detection of lung metastases in bone (Jelinek et al, 1990), enlarged node detection in hilar and mediastinal cancers (Heelan et al, 1985). In addition, MRI has aided in the process of target definition for radiotherapy for head and neck tumours (Toonkel et al, 1988 and Pötter et al, 1992), carcinoma of the uterine cervix (Mayr et al, 1993 and Mayr et al, 1996), subdiaphragmatic tumours (Müller-Schimpfle, 1992) and for other pelvic, chest, spine and abdominal wall tumours (Shuman et al, 1985). As new MRI/MRA techniques are developed (this is a major and productive research area) further improvements in the ability of MRI to delineate tumours from other areas of the body will occur, thus increasing the demand for the use of it in radiotherapy.

Whether the MRI system is used for stereotactic or general radiotherapy treatment work, a priori knowledge of distortion is especially important when the MRI system offers unique target structure information (ie the target is not visible on CT for example, which has provable geometric integrity) and so no cross-check verification of the final target position is possible. This “flying blind” approach could seriously jeopardise the treatment and hence the patient’s
prognosis if significant geometric distortion is present in the MRI data and is not corrected for.

5.2 Future Work

As was mentioned in section 4.7 it is intended to assess the geometric integrity of the proposed MRI scanner at Liverpool Hospital, Sydney, Australia. The distortion correction technique of chapter 3 will be invoked, and applied to actual patients (who have both MRI and CT scans performed) undergoing radiotherapy as per the methods defined in chapter 4.

It is then planned to assess the non-machine dependent residual MRI distortions by direct comparison to the corresponding CT data sets. A phantom study is also proposed that will induce non-machine dependent distortions to images of that phantom, thus enabling quantitative confirmation of the estimates for these types of distortion which were made in chapter 1. Such a phantom will consist of neighbouring compartments of precise physical dimension that will allow samples to be introduced of, for example, differing magnetic susceptibility, or chemical structure.
Bibliography


Bloch F 1946 Nuclear induction Phys. Rev. 70 460-474


Bracewell R N 1978 The Fourier transform and its physical applications (New York, USA: McGraw Hill)

Bucciolini M, Ciraolo L and Renzi R 1986 Relaxation rates of paramagnetic solutions: Evaluation by nuclear magnetic resonance imaging Medical Physics 13 298-303
Chen G T Y and Pelizzari C A 1989 Image correlation techniques in radiation therapy treatment planning *Computerized Medical Imaging and Graphics* 13 235-240


Damadian R 1971 Tumour detection by nuclear magnetic resonance *Science* 171 1151-1153


Drangova 1987 Stereotactic neurosurgical planning using magnetic resonance imaging: Image distortion evaluation (Master of Science Thesis, Department of Physics, McGill University: Montreal)


Ehricke H H and Schad L R 1992 MRA-guided stereotactic radiation treatment planning for cerebral angiomas *Computerized Medical Imaging and Graphics* 16 65-71

Gauvin A 1992 Geometrical distortion of magnetic resonance images (Master of Science Thesis, Department of Physics, McGill University: Montreal)

Gerlach W and Stern O 1924 On the position of the quantum lines in a magnetic field Ann. Phys. 74 675-699


Heelan R T, Martini N, Westcott J W et al Carcinomatous involvement of the hilum and mediastinum: Computed tomographic and magnetic resonance evaluation Radiology 156 111-115

Hemler P F, Napel S, Sumanaweera T S et al 1995 Registration error quantification of a surface-based multimodality image fusion system Medical Physics 22 1049-1056

Henkelman R M and Bronskill M J 1987 Artifacts in magnetic resonance imaging Reviews in Magnetic Resonance Imaging 2 1-126


Hinshaw W S and Lent A H 1983 An introduction to NMR imaging: From the Bloch equation to the imaging equation *Proceedings of the IEEE* 71 338-350

Horowitz A L MRI physics for physicians 1989 (New York, USA: Springer-Verlag)

ICRU 1993 Prescribing, recording and reporting photon beam therapy *ICRU Report 50* (Bethesda, Maryland: ICRU)


Lauterbur P C 1973 Image formation by induced local interactions: examples employing nuclear magnetic resonance *Nature* 242 190-191
Lunsford L D, Martinez A J and Latchaw R E 1986 Stereotaxic surgery with a magnetic
resonance- and computerized tomography-compatible system *Journal of Neurosurgery* **64** 872-
878

radiation therapy *Radiology* **189** 601-608

Mayr N A, Magnotta V A, Ehrhardt J C et al 1996 Usefulness of tumour volumetry by magnetic
resonance imaging in assessing response to radiation therapy in carcinoma of the uterine cervix
*Int. J. Radiation Oncology Biol. Phys.* **35** 915 924

Moerland M A 1996 Magnetic resonance imaging in radiotherapy treatment planning (PhD
Thesis, University of Utrecht, The Netherlands)

Moshfeghi M 1991 Elastic matching of multimodality medical images *Graphical Models and
Image Processing* **53** 271-282

Mourino M R 1991 From Thales to Lauterbur, or from the lodestone to MR imaging: magnetism
and medicine *Radiology* **180** 593-612

Müller-Schimpfle M, Layer G, Köster A et al 1992 MRI and MRA in treatment planning of
subdiaphragmatic radiation therapy *Journal of Computer Assisted Tomography* **16** 110-119

O'Donnell M and Edelstein W A 1985 NMR imaging in the presence of magnetic field
imhomogeneities and gradient field nonlinearities *Medical Physics* **12** 20-26
Peters T M, Clarke J A, Oliver A et al 1986 Integrated stereotaxic imaging with CT, MR imaging and digital subtraction angiography Radiology 161 821-826


Pötter R, Heil B, Schneider L et al 1992 Sagittal and coronal planes from MRI for treatment planning in tumours of brain, head and neck: MRI assisted simulation Radiotherapy and Oncology 23 127-130


Prott F J, Haverkamp U, Willich N et al 1995 Comparison of imaging accuracy at different MRI units based on phantom measurements Radiotherapy and Oncology 37 221-224


Schenck J F 1996 The role of magnetic susceptibility in magnetic resonance imaging: MRI magnetic compatibility of the first and second kinds *Medical Physics* 23 815-850


Sprawls P and Bronskill M J (eds) 1992 The physics of magnetic resonance imaging *Proceedings of the 1992 AAPM Summer School (AAPM)*


Vannier M W and Gayou D E 1988 Automated registration of multimodality images *Radiology* 169 860-861


Zeeman P 1897 On the influence of magnetism on the nature of light emitted by a substance Phil. Mag. 43 226-239
Appendix A: Peak Search Algorithm

/*********************/
/*
/* peak.c
/* This program searches for the peaks in MRI images and produces an
/* output file of cartesian coordinates of the found peaks. This can
/* then be read into e.g. sigma plot to enable plotting of the data.
/*
/* W. Beckham August 1994
/*
/*********************/

#include <stdio.h>
#include <dos.h>
#include <alloc.h>
#include <math.h>
#define matrix-raw(n,m) matrix l((unsigned)m + n * 256)]
#define matrix-mask(n,m) mmatrix l((unsigned)m + n * 256)]
#define TRUE 1
#define FALSE 0
/*Threshold ratio i+1th/th for detection of possible peak*/
#define peak_threshold 1.3
/*Threshold for rejection of annulus1 max/annulus2 max ratio which signifies
a false peak due to artifacts*/
#define reject_ratio 1.17
/*Peak height rejection threshold*/
#define peak_height_min 50
FILE *infile;
FILE *outfile;
void pause();
void row_scan();
void local_max_search();
void centre_of_gravity();
void output_cartesian();
void sl_out_cartesian();

unsigned short huge *matrix;
char huge *mmatrix;
float far *sl_x;
float far *sl_y;
unsigned x_dim,y_dim,dc_floor,local_max;
unsigned ann1_max,ann2_max,local_max,sl_out_cartesian();
long sum_wx,sum_wy,total_wt;
int i,j,x,y,m,n,local_pimmax_x,local_pimmax_y,num_peak,fifteen_mm,sl_out_cartesian();
char input[12],output[12],out_ans[5];
float cog_x,cog_y,origin_x,origin_y,peak_x[250],peak_y[250],coord_x,coord_y;
float ang_ave;
/*Set up structure to enable easy byte swap from MRI VAX*/
union
{
unsigned short word;
struct
{
    unsigned char first;
    unsigned char second;
} rvalue;

main()
{

    /*User input of filename*/
    printf("Input image filename --> ");
    scanf("%s",input);

    /*Opens file for read and exits if not found*/
    if ((infile = fopen(input,"rb")) == NULL)
    {
        printf("Could not open file %s",input);
        exit(1);
    }

    /*Prompt user for output file name*/
    printf("Enter name of output file --> ");
    scanf("%s",output);
    if ((outf,rle=fopen(output,"w")) == NULL)
    {
        printf("Could not open output file");
        exit(1);
    }

    /*Sends file pointer to position of x matrix dimension*/
    fseek(infile,650,0);
    fread(&rvalue.word,sizeof(unsigned short),1,infile);

    /*Perform byte swap*/
    x_dim=rvalue.byte.second << 8 | rvalue.byte.first;

    /*Repeat for y dimension*/
    fread(&rvalue.word,sizeof(unsigned short),1,infile);
    y_dim=rvalue.byte.second << 8 | rvalue.byte.first;

    /*Test that matrix size is 256 x 256, if not then exit*/
    if (x_dim != 256 || y_dim != 256)
    {
        printf("FILE PROBLEM: Matrix is %u x %u, not 256 x 256",x_dim,y_dim);
        exit(1);
    }

    /*Extract the number of pixels for 10.0 cm*/
    fseek(infile,694,0);
    fread(&rvalue.word,sizeof(unsigned short),1,infile);
    scale_fact = rvalue.byte.second << 8 | rvalue.byte.first;
    printf("There are %u Pixels for 10 cm in image space",scale_fact);

    /*Allocate memory space for array "matrix_raw"*/
    printf("%lu bytes left before array space allocated",farc oreleft());
if ((matrix =(unsigned short *))
   faralloc((long)x_dim*y_dim,sizeof (unsigned short))) == NULL)
{  
    printf("nInsufficient space for pixel array");
    exit(1);
}
printf("n\nStarting address of pixel array = %Fp",matrix);
/*Fill array "matrix_raw" with pixel data*/

/*Set pointer at first pixel location in input file*/
printf("\n\nLoading pixel array with pixel data...");
ffseek(infile,4096,0);
for (y=0; y < y_dim; y++)
{  
    for (x=0; x < x_dim; x++)
     { 
      fread(&rvalue.word,sizeof(unsigned short),1,infile);
      matrix_raw(y,x) = rvalue.byte.second << 8 ! rvalue.byte.first;
    }
  
  printf("nFilled pixel array");
  
  /*Test if array loaded correctly*/
  /*for (x=200; x < 256; x++)
   printf("(7ou,255)= %d",matrix_raw(255,x));*/
  
  /*Allocate memory space for array matrix-mask to prevent a peak from being
   found more than once*/
  if ((mmatrix =(char *))
     faralloc((long)x_dim*y_dim,sizeof (char))) == NULL)
  {  
    printf("nInsufficient space for mask array");
    exit(1);
  }
  printf("n\nStarting address of mask array = %Fp",mmatrix);
  printf("n\n%-lu bytes after all array space allocated",farcoreleft());
  /*Fill array "matrix_mask" with mask data*/
  printf("\n\nLoading mask array with mask data...");

  printf("n\nFilled mask array\n");
  
  /*Go to zone at centre of matrix and find max. and min. to establish
   height of peaks and DC noise floor*/

  czone_max = 0;
  dc_floor = 65535;
  for (y=100; y <=150; y++)
   { 
    for (x=100; x <=150; x++)
     { 
      matrix_mask(y,x) = 0;
    }
   }
  printf("nFilled mask array\n");

  
  /*Go to zone at centre of matrix and find max. and min. to establish
   height of peaks and DC noise floor*/

  czone_max = 0;
  dc_floor = 65535;
  for (y=100; y <=150; y++)
   { 
    for (x=100; x <=150; x++)
     { 
      if (matrix_raw(y,x) < dc_floor) dc_floor = matrix_raw(y,x);
  
  printf("nFilled mask array\n");

  
  /*Go to zone at centre of matrix and find max. and min. to establish
   height of peaks and DC noise floor*/

  czone_max = 0;
  dc_floor = 65535;
  for (y=100; y <=150; y++)
   { 
    for (x=100; x <=150; x++)
     { 
      if (matrix_raw(y,x) < dc_floor) dc_floor = matrix_raw(y,x);
  
  printf("nFilled mask array\n");
if (matrix_raw(y,x) > czone_max) czone_max = matrix_raw(y,x);
}
/*printf("ndc_floor = %u, czone_max = %u",dc_floor,czone_max);*/
/*Start scanning the pixel matrix*/
row_scan();
/*Free up array space*/
farfree((unsigned short far *)matrix);
farfree((char *)mmatrix);
/*Place output file cent. of gravity data on a Cartesian coord. system to be viewed as in the image on the scanner*/
output_cartesian();
/*Produce file of coords of actual positions (appropriately rotated) of the S.L. phantom*/
sl_out_cartesian();
/*Close input and output files*/
fclose(infile);
fclose(outfile);
}
void pause()
{
printf("Hit return to continue ...");
getchar();
getchar();
}
/*This routine governs the scanning of the image matrix to enable searching for possible signal peaks. When one is found another routine is called (local_max_search) that finds a local maxima and some rejection criteria for artifacts*/
void row_scan()
{
num_peak = 0;
printf("\nStarting the search...");
for (y=10; y<240; y++)
{
for (x=3; x<253; x++)
{
/*printf("nx=%u,y=%u, Value=%u",x,y,matrix_raw(y,x));
divisor = ((float)matrix_raw(y,x+1)/(float)matrix_raw(y,x));
printf(" divisor=%f",divisor);*/
/*Rejects pixel if in air*/
if ((float)matrix_raw(y,x) < (float)(dc_floor/1.5)) continue;
/*Check mask matrix to reject already found peaks*/
while(matrix_mask(y,x)==1)
x++;
/*printf("x = %u, y = %u, matrix_mask = %i", x, y, matrix_mask(y, x));*/

/*Checks to see if rising trend meets possible peak criteria*/
if (x != 255 && (float)matrix_raw(y, x + 1)/(float)matrix_raw(y, x) > peak_threshold)
{
    /*printf("x = %u, y = %u, calling local_max_search...", x, y);
    printf("ith+1th=%u, ith=%u", matrix_raw(y, x + 1), matrix_raw(y, x));*/
    local_max_search();
}

/*This routine finds a local pixel maxima near to the pixel that was
identified as being a possible peak in routine row_scan. The top left of the
area defined for the search is 2 pixels to the left, and 2 pixels above this
pixel. Further it is 5 x 5 pixels in extent. The routine also contains some
rejection criteria to take into account artifacts due to e.g. bright signal
bands around the edges of the spatial linearity plates*/

void local_max_search()
{
    local_max = 0;

    /*This nested loop finds local max pixel positions in x and y*/
    for(j = y - 1; j < y + 4; j++)
        for(i = x - 2; i < x + 3; i++)
            /*printf("i=%u, j=%u", i, j);*/
            if (matrix_raw(i, j) > local_max)
                local_max = matrix_raw(i, j);
    local_pixmax_x = i;
    local_pixmax_y = j;

    /*printf("local_max = %u, at (%u,%u)", local_max, local_pixmax_x, local_pixmax_y);*/

    /*Find the pixel maximum in the annulus immediately surrounding the
    pixel maximum found above*/
    ann1_max = 0;
    for(j = local_pixmax_y - 1; j < local_pixmax_y + 2; j++)
        for(i = local_pixmax_x - 1; i < local_pixmax_x + 2; i++)
            if((local_pixmax_x == i && local_pixmax_y == j) continue;
            if(matrix_raw(i, j) > ann1_max) ann1_max = matrix_raw(i, j);

    /*printf("\nann1_max = %u", ann1_max);*/

    /*Find the pixel maximum in the annulus immediately outside the one above*/
    ann2_max = 0;

    /*Scan the top & bottom rows of the annulus*/
    for(j = local_pixmax_y - 2; j < local_pixmax_y + 3; j++)
        {
for(i=local_pixmax_x-2; i<local_pixmax_x+3; i++)
{
    if(matrix_raw(i,j) > ann2_max) ann2_max = matrix_raw(i,j);
}

/*Scan the remaining 6 pixels in the second annulus*/
for(i=local_pixmax_x-2; i<local_pixmax_x+3; i=i+4)
{
    for(j=local_pixmax_y-1; j<local_pixmax_y+2; j++)
    {
        if(matrix_raw(i,j) > ann2_max) ann2_max = matrix_raw(i,j);
    }
    /*printf("\nann2_max = %lu",ann2_max);*/
}

/*Check if need to reject peak due to surrounding high pixel values i.e.
artifacts*/
if(((float)ann1_max / (float)ann2_max > reject_ratio) && (local_max > 
peak_height_min))
{
    num_peak++;
    printf("\nNumber of peaks found = %lu",num_peak);
    /*for(i=1; i<=sizeof(num_peak); i++)
    {
        printf("\n\b\n");
    }*/

    /*Mask out 30mm x 30mm area surrounding the maximum pixel in matrix_mask*/

    /*Work out how many pixels for 15mm*/
    fifteen_mm = (int)floor((float)scale_fact/100 * 15);
    for(j=local_pixmax_y-fifteen_mm; j<=local_pixmax_y+fifteen_mm; j++)
    {
        for(i=local_pixmax_x-fifteen_mm; i<=local_pixmax_x+fifteen_mm; i++)
        {
            matrix_mask(j,i) = 1;
        }
    }

centre_of_gravity();
/*else(printf("\nPeak rejected at (%lu,%lu)\n",local_pixmax_x,local_pixmax_y));*/

/*This routine finds the centre of gravity of the 24 pixels surrounding the
maximum pixel of the peak found in the previous routine*/
void centre_of_gravity()
{
    sum wx = 0;
    sum wy = 0;
    total wt = 0;
    for(j=local_pixmax_y-2; j<=local_pixmax_y+2; j++)
    {
        for(i=local_pixmax_x-2; i<=local_pixmax_x+2; i++)
        {
            sum wx = sum wx + (long)matrix_raw(j,i) * (long)i;
            sum wy = sum wy + (long)matrix_raw(j,i) * (long)j;
        }
    }
    /*Use moment formula to find centroid*/
    sum_xy = sum wx + sum wy;
    idefinition = sum_xy / total wt;
    /*Output centre of gravity*/
    printf("\nCentre of gravity\n= (%lu,%lu)\n",idefinition,idefinition);
    return;
}
total_wt = total_wt + (long)matrix_raw(j,i);
}

cog_x = (float)sum_wx/(float)total_wt;
cog_y = (float)sum_wy/(float)total_wt;

/* Checks to see if peak found is near the position x=125 y=130 for the purpose of calculating offsets to place the Cartesian origin over the centre of this peak */
if((local_pixmax_x > 117 && local_pixmax_x < 133 &&
    local_pixmax_y > 122 && local_pixmax_y < 138)
{
    origin_x = cog_x;
    origin_y = cog_y;
    printf("Origin (%f,%f),origin_x,origin_y);
    pause(

    peak_x[num_peak] = cog_x;
    peak_y[num_peak] = cog_y;
    }

/* This routine converts the output file peak pixel position data into Cartesian data as viewed on the MRI scanner. The origin is centred on the peak at approx. pixel_x = 125, pixel_y = 130. It also finds an appropriate angle of rotation to be used when placing the phantom actual peak positions onto the same coordinate axes. */
void output_cartesian()
{
    int i,num_fnd;
    float ang_north,ang_east,ang_south,ang_west;
    num_fnd = 0;
    printf(outfile,"Image Space S.L. Data
    /* Converts arrays to Cartesian system of coords. and writes to outfile */
    for(i=1; i<=num_peak; i++)
    {
        peak_x[i] = (peak_x[i] - origin_x)/(float)scale_fact/100.0;
        peak_y[i] = (-peak_y[i] + origin_y)/(float)scale_fact/100.0;
        printf(outfile,"%f,%f

    /* Calculate any rotation required to position phantom actual peak positions on same Cartesian axes as image data. Rotation is calculated by taking the mean angle of rotation about the origin, of the four nearest peaks to the origin. NB the image data is NOT rotated */
    for(i=1; i<=num_peak; i++)
    {
        if(peak_x[i] > 20.0 && peak_x[i] < 36.0 &&
            peak_y[i] > -8.0 && peak_y[i] < 8.0)
            ang_east = -atan(peak_y[i]/peak_x[i]);
            num_fnd++;
        else if(peak_x[i] > -8.0 && peak_x[i] < 8.0 &&
            peak_y[i] > -36.0 && peak_y[i] < -20.0)
            ang_south = atan(peak_x[i]/peak_y[i]);
            num_fnd++;
else if(peak_x[i] > -36.0 && peak_x[i] < -20.0 &&
    peak_y[i] > -8.0 && peak_y[i] < 8.0) {
    ang_west = -atan(peak_y[i]/peak_x[i]);
    num_fnd++; 
}
else if(peak_x[i] > -8.0 && peak_x[i] < 8.0 &&
    peak_y[i] > 20.0 && peak_y[i] < 36.0) {
    ang_north = atan(peak_x[i]/peak_y[i]);
    num_fnd++; 
}
printf("num_fnd = %i",num_fnd);

if(num_fnd != 4) {
    printf("Not all peaks found for rotation in routine output_cartesian");
    exit(1);
    ang_ave = (ang_east + ang_south + ang_west + ang_north)/4.0;
    printf("Average angle of rotation = %f radians",ang_ave);
}

/*This routine loads an array with the coords of the holes in the sl phantom.
The data set is rotated by the angle ang_ave calculated in the previous
routine*/
void sl_out_cartesian() {
    
    int i,j,count;
    
    /*Allocate memory space for arrays "sl_x" and "sl_y"*/
    printf("%lu bytes left before array space allocated",farcoreleft());
    if (((sl_x = (float *)
        farcalloc((long)193,sizeof (float))) == NULL ||
        (sl_y = (float *)
        farcalloc((long)193,sizeof (float))) == NULL)) {
        printf("Insufficient space for pixel array");
        exit(1);
    }
    count = 0;
    printf("putting 2nd header in output file");
    printf("ncount = %i",count);
    
    /*Load the central data*/
    for(j=-4; j<=6; j++) {
        for(i=-6; i<=6; i++) {
            sl_x[count] = 28.0 *(float)i;
            sl_y[count] = 28.0 *(float)j;
            count++;
        }
    
    printf("top row");
    printf("ncount = %i",count);
    
    /*Load top row*/
    for(i=count; i<=count+6; i++) {
    
}
sl_y[i] = 224.0;
}
count = count + 6;

sl_x[count-6] = -84.0;
sl_x[count-5] = -56.0;
sl_x[count-4] = -28.0;
sl_x[count-3] = 28.0;
sl_x[count-2] = 56.0;
sl_x[count-1] = 84.0;

printf("second row");
printf("count = %i", count);
/*Load second row*/
for(i=-4; i<=4; i++)
{
    sl_x[count] = 28.0 * (float)i;
    sl_y[count] = 196.0;
    count++;
}

printf("bottom row");
printf("count = %i", count);
/*Load bottom row*/
for(i=-4; i<=4; i++)
{
    sl_x[count] = 28.0 * (float)i;
    sl_y[count] = -140.0;
    count++;
}

printf("First col.");
printf("count = %i", count);
/*Load first column*/
for(i=-2; i<=2; i++)
{
    sl_x[count] = -224.0;
    sl_y[count] = 28.0 * (float)i;
    count++;
}

printf("2nd col.");
printf("count = %i", count);
/*Load second column*/
for(i=-3; i<=4; i++)
{
    sl_x[count] = -196.0;
    sl_y[count] = 28.0 * (float)i;
    count++;
}

printf("2nd last col.");
printf("count = %i", count);
/*Load second last column*/
for(i=-3; i<=4; i++)
{
    sl_x[count] = 196.0;
    sl_y[count] = 28.0 * (float)i;
    count++;
}
printf("\nlast col\n");
printf("\ncount = %i\n",count);

/*Load last column*/
for(i=-2; i<=2; i++)
{
  sl_x[count] = 224.0;
  sl_y[count] = 28.0 * (float)i;
  count++;
}
printf("\ncount = %i\n",count);

/*Rotate data by angle ang_ave using standard rotation transformation. Note
that in rot. transformation the angle convention is negative of standard.*/
for(i=0; i<=192; i++)
{
  sl_x[i] = (sl_x[i] * cos(-ang_ave)) - (sl_y[i] * sin(-ang_ave));
  sl_y[i] = (sl_x[i] * sin(-ang_ave)) + (sl_y[i] * cos(-ang_ave));
}

/*Print S.L. Data to file*/
fprintf(outfile,"Phantom Space S.L. Rotated Data...\n");
for(i=0; i<=192; i++) fprintf(outfile,"%f,%f\n",sl_x[i],sl_y[i]);

}
Appendix B: Distortion Magnitude Assessment Algorithm

/**************************************************************************/
/*dist_mag.c
/*PRODUCES AN OUTPUT FILE OF MAGNITUDES OF DISTORTION IE. X,Y,Z
/*
/*
/* W. Beckham November 1996
/**************************************************************************/

#include <stdio.h>
#include <dos.h>
#include <alloc.h>
#include <math.h>

#define matrix-raw(n,m) matrix[ ((unsigned)m + n * 256) ]
#define matrix-mask(n,m) mmatrix[ ((unsigned)m + n * 256) ]
#define trans-x(n,m) nmatrix[ ((unsigned)m + n * 256) ]
#define trans-y(n,m) omatrix[ ((unsigned)m + n * 256) ]
#define TRUE 1
#define FALSE 0

/*Threshold ratio ith/i-1th for detection of possible peak*/
#define peak_threshold 1.3

/*Threshold for rejection of annulus1 max/annulus2 max ratio which signifies
a false peak due to artifacts*/
#define reject_ratio 1.17

/*Peak Height Rejection Threshold*/
#define peak_height_min 50

FILE *infile;
FILE *outfile;
void pause0;
void initialise0;
void row-scan0;
void local_max_search0;
void centre_of_gravity0;
void output_cartesian0;
void sl_cartesian0;
void synchronise_arrays0;
void xyz_mag0;

unsigned short huge *matrix;
char huge *mmatrix;
float huge *nmatrix;
float huge *omatrix;
float far *sl-x;
float far *sl-y;
unsigned x_dim,y_dim, scale_fact, czone_max, dc_floor, local_max;
unsigned ann1_max, ann2_max, pix_sum_ann1, next_x;
long sum_wx, sum_wy, total_wt;
int i, j, x, y, m, n, local_pixmap_x, local_pixmap_y, num_peak, fifteen_mm, file_out;
char input[12], output[12], out_ans[5];
float cog_x, cog_y, origin_x, origin_y, peak_x[200], peak_y[200], coord_x, coord_y;
float ang_ave;

/*Set up structure to enable easy byte swap from MRI VAX*/
union
{
    unsigned short word;
struct
{
    unsigned char first;
    unsigned char second;
} byte;
} rvalue;

main()
{

/*User input of filename*/
printf("Input image filename --> ");
scanf("%s", input);

/*Opens file for read and exits if not found*/
if ((infile = fopen(input,"rb")) == NULL)
{
    printf("Could not open file %s", input);
    exit(1);
}

/*Prompt user for output file name*/
printf("Enter name of output file --> ");
scanf("%s", output);
if((outfile = fopen(output,"w")) == NULL)
{
    printf("Could not open output file");
    exit(1);
}

/*Set up initial conditions for the image search and also load the input
file into an array*/
initialise();

/*Start scanning the pixel matrix*/
row_scan();

/*Free up mask array space & image array space*/
free((char *)mmatrix);
free((unsigned short *)matrix);

/*Place cent. of gravity of the found peaks data on a Cartesian coord.
system to be viewed as in the image on the scanner and place in arrays
peak_x and peak_y*/
output_cartesian();

/*Produce arrays (sl_x, sl_y) of coords of actual positions (appropriately
rotated) of the S.L. phantom*/
sl_cartesian();

/*Synchronise the arrays sl_x --> peak_x and sl_y --> peak_y such that
for each point in undistorted space the corresponding point in distorted
space is easily found*/
synchronise_arrays();
/*Calculate the distortion magnitude at each spatial linearity point*/
xyz_mag();

/*Close input and output files and free up array space*/
fclose(infile);
fclose(outfile);

/*This routine sets up initial conditions for the search for peaks in the
distorted image space. It also loads the input file to an array in order to
speed up the search routine. A mask array is also loaded to prevent a peak
being found more than once.*/
void initialise()
{
/*Fill arrays peak_x and peak_y with 1000.0. This is required later to
take into account that there may be more (or less) peaks found in the
peak search than 193*/
for(i=0; i<=199; i++)
{
peak_x[i] = 1000.0;
peak_y[i] = 1000.0;
}
/*Sends file pointer to position of x matrix dimension*/
fseek(infile,650,0);
fread(&rvalue.word,sizeof(unsigned short),1,infile);
/*Perform byte swap*/
x_dim=rvalue.byte.second << 8 | rvalue.byte.first;
/*Repeat for y dimension*/
fread(&rvalue.word,sizeof(unsigned short),1,infile);
y_dim=rvalue.byte.second << 8 | rvalue.byte.first;
/*Test that matrix size is 256 x 256, if not then exit*/
if (x_dim != 256 || y_dim != 256)
{
printf("FILE PROBLEM: Matrix is %u x %u, not 256 x 256",x_dim,y_dim);
exit(1);
}
/*Extract the number of pixels for 10.0 cm*/
fseek(infile,694,0);
fread(&rvalue.word,sizeof(unsigned short),1,infile);
scale_fact = rvalue.byte.second << 8 | rvalue.byte.first;
printf("\nThere are %u Pixels for 10 cm in image space",scale_fact);

/*Allocate memory space for array "matrix_raw"*/
printf("\n%u bytes left before array space allocated",farcoreleft());
if ((matrix =(unsigned short *))
  farcalloc((long)x_dim*y_dim,sizeof (unsigned short))) == NULL)
{
  printf("\nInsufficient space for pixel array");
  exit(1);
}
printf("\nStarting address of pixel array = %F",matrix);

/*Fill array "matrix_raw" with pixel data*/
/*Set pointer at first pixel location in input file*/
printf("\nLoading pixel array with pixel data...");
for (y=0; y < y_dim; y++)
{
  for (x=0; x < x_dim; x++)
  {
    fread(&rvalue.word,sizeof(unsigned short),1,infile);
    matrix-raw(y,x) = rvalue.byte.second << 8 | rvalue.byte.first;
  }
}
printf("\nFilled pixel array");
/*Test if array loaded correctly*/
/*for (x=200; x < 256; x++)
printf("(Vou,255)=Vou",x,matrix-raw(255,x));x*/
/*Allocate memory space for array matrix-mask to prevent a peak from being
found more than once*/
if ((mmatrix =(char *))
  farcalloc((long)x_dim*y_dim,sizeof (char))) == NULL)
{
  printf("\nInsufficient space for mask array");
  exit(1);
}
printf("\nStarting address of mask array = %F",mmatrix);
printf("\nBytes after all array space allocated",farcoreleft());

/*Fill array "matrix_mask" with mask data*/
/*Go to zone at centre of matrix and find max. and min. to establish
height of peaks and DC noise floor*/
czone_max = 0;
dc_floor = 65535;
for (y=100; y<=150; y++)
{
    for (x=100; x<=150; x++)
    {
        if (matrix_raw(y,x) < dc_floor) dc_floor = matrix_raw(y,x);
        if (matrix_raw(y,x) > czone_max) czone_max = matrix_raw(y,x);
    }
    /*printf("nde_floor = %u, czone_max = %u",dc_floor,czone_max);*/
}

/*This routine governs the scanning of the image matrix to enable searching
for possible signal peaks. When one is found another routine is called
(local_max_search) that finds a local maxima and some rejection criteria
for artifacts*/

void row_scan()
{
    num_peak = 0;
    printf("\n\nStarting the search...\n");
    for (y=0; y<256; y++)
    {
        for (x=0; x<256; x++)
        {
            /*printf("nx=%u,y=%u, Value=%u",x,y,matrix_raw(y,x));
            divisor = ((float)matrix_raw(y,x+1))/(float)matrix_raw(y,x));
            printf(" divisor=%f",divisor);*/
            /*Rejects Pixel if in air*/
            if ((float)matrix_raw(y,x) < (float)(dc_floor/1.5)) continue;
            /*Check mask matrix to reject already found peaks*/
            while(matrix_mask(y,x)==1)
            {
                x++;
                /*printf("nx=%u, y=%u, matrix_mask = %i",x,y,matrix_mask(y,x));*/
            }
            /*Checks to see if rising trend meets possible peak criteria*/
            if (x!=255 & & (float)matrix_raw(y,x+1)/(float)matrix_raw(y,x) > peak_threshold)
            {
                /*printf("nix= %u, y= %u, calling local_max_search...",x,y);
                printf("nth+1th= %u, ith= %u",matrix_raw(y,x+1),matrix_raw(y,x));*/
                local_max_search();
            }
        }
    }
    /*This routine finds a local pixel maxima near to the pixel that was
identified as being a possible peak in routine row_scan. The top left of the
area defined for the search is 2 pixels to the left, and 2 pixels above this
pixel. Further it is 5 x 5 pixels in extent. The routine also contains some
rejection criteria to take into account artifacts due to e.g. bright signal
bands around the edges of the spatial linearity plates*/

void local_max_search()
{
local_max = 0;

/*This nested loop finds local max pixel positions in x and y*/
for(j=-1; j<4; j++)
{
for(i=-2; i<3; i++)
{
/*printf("ni=%u, j=%u",i,j);*/
if (matrix_raw(j,i) > local_max)
{
local_max = matrix_raw(j,i);
local_pixmap_x = i;
local_pixmap_y = j;
}
}
/*printf("local_max = %u, at (%u, %u)",local_max,local_pixmap_x,local_pixmap_y);*/

/*Find the pixel maximum in the annulus immediately surrounding the
pixel maximum found above*/
ann1_max = 0;
for(j=local_pixmap_y-1; j<local_pixmap_y+2; j++)
{
for(i=local_pixmap_x-1; i<local_pixmap_x+2; i++)
{
if(local_pixmap_x == i && local_pixmap_y == j) continue;
if(matrix_raw(j,i) > ann1_max) ann1_max = matrix_raw(j,i);
}
/*printf("\nann1_max = %\u",ann1_max);*/

/*Find the pixel maximum in the annulus immediately outside the one above*/
ann2_max = 0;

/*Scan the top & bottom rows of the annulus*/
for(j=local_pixmap_y-2; j<local_pixmap_y+3; j=j+4)
{
for(i=local_pixmap_x-2; i<local_pixmap_x+3; i++)
{
if(matrix_raw(j,i) > ann2_max) ann2_max = matrix_raw(j,i);
}
/*printf("\nann2_max = %\u",ann2_max);*/

/*Scan the remaining 6 pixels in the second annulus*/
for(i=local_pixmap_x-2; i<local_pixmap_x+3; i=i+4)
{
for(j=local_pixmap_y-1; j<local_pixmap_y+2; j++)
{
if(matrix_raw(j,i) > ann2_max) ann2_max = matrix_raw(j,i);
}
/*printf("\nann2_max = %\u",ann2_max);*/

/*Check if need to reject peak due to surrounding high pixel values i.e.
artifacts*/
if((float)ann1_max / (float)ann2_max > reject_ratio) & & (local_max >
peak_height_min))
{
num_peak++;
printf("\nNumber of peaks found = %\u",num_peak);
}
for(i=1; i<=sizeof(num_peak); i++)
{
    printf("\b");
}*/

/*Mask out 30mm x 30mm area surrounding the maximum pixel in matrix_mask*/

/*Work out how many pixels for 15mm*/
fifteen_mm = (int)floor((float)scale_fact/100 * 15);
for(j=local_pixmax_y-fifteen_mm; j<=local_pixmax_y+fifteen_mm; j++)
{
for(i=local_pixmax_x-fifteen_mm; i<=local_pixmax_x+fifteen_mm; i++)
{
    matrix_mask(j,i) = 1;
}
}

centre_of_gravity();
else(printf("nPeak rejected at (%u,%u)",local_pixmax_x,local_pixmax_y));

}/*This routine finds the centre of gravity of the 25 pixels surrounding the
maximum pixel of the peak found in the previous routine*/
void centre_of_gravity()
{
    sum.wx = 0;
    sum.wy = 0;
    total.wt = 0;
    for(j=local_pixmax_y-2; j<=local_pixmax_y+2; j++)
    {
for(i=local_pixmax_x-2; i<=local_pixmax_x+2; i++)
{
    sum.wx = sum.wx + ((long)matrix.raw(j,i) * (long)i);
    sum.wy = sum.wy + ((long)matrix.raw(j,i) * (long)i); 
    total.wt = total.wt + (long)matrix.raw(j,i);
}
}
cog.x = (float)sum.wx/(float)total.wt;
cog.y = (float)sum.wy/(float)total.wt;

/*Checks to see if peak found is near the position x=125 y=130 for the
purpose of calculating offsets to place the Cartesian origin over the
centre of this peak*/
if(local_pixmax.x > 117 && local_pixmax.x < 133 &&
    local_pixmax.y > 122 && local_pixmax.y < 138)
{
    origin.x = cog.x;
    origin.y = cog.y;
}

    peak.x[num_peak-1] = cog.x;
    peak.y[num_peak-1] = cog.y;
}

/*This routine converts the peak pixel position data into
Cartesian data as viewed on the MRI scanner. The origin is centred on the peak at approx. pixel_x = 125, pixel_y = 130. It also finds an appropriate angle of rotation to be used when placing the phantom actual peak positions onto the same coordinate axes. */

void output_cartesian()
{
    int i, num_fnd;
    float ang_north, ang_east, ang_south, ang_west;

    num_fnd = 0;

    /*Converts arrays to Cartesian system of coords.*/
    for(i=0; i<=num_peak-1; i++)
    {
        peak_x[i] = (peak_x[i] - origin_x)/((float)scale_fact/100.0);
        peak_y[i] = (-peak_y[i] + origin_y)/((float)scale_fact/100.0);
    }

    /*Calculate any rotation required to position phantom actual peak positions on same Cartesian axes as image data. Rotation is calculated by taking the mean angle of rotation about the origin, of the four nearest peaks to the origin. NB the image data is NOT rotated*/
    for(i=0; i<=num_peak-1; i++)
    {
        if(peak_x[i] > 25.0 && peak_x[i] < 31.0 &&
           peak_y[i] > -5.0 && peak_y[i] < 5.0)
        {
            ang_east = atan(peak_y[i]/peak_x[i]);
            num_fnd++;
        } else if(peak_x[i] > -5.0 && peak_x[i] < 5.0 &&
            peak_y[i] > -31.0 && peak_y[i] < -25.0)
        {
            ang_south = atan(peak_x[i]/peak_y[i]);
            num_fnd++;
        } else if(peak_x[i] > -31.0 && peak_x[i] < -25.0 &&
            peak_y[i] > -5.0 && peak_y[i] < 5.0)
        {
            ang_west = atan(peak_y[i]/peak_x[i]);
            num_fnd++;
        } else if(peak_x[i] > -5.0 && peak_x[i] < 5.0 &&
            peak_y[i] > 25.0 && peak_y[i] < 31.0)
        {
            ang_north = atan(peak_x[i]/peak_y[i]);
            num_fnd++;
        }
    }
    printf("num_fnd = %i", num_fnd);

    if(num_fnd != 4)
    {
        printf("Not all peaks found for rotation in routine output_cartesian");
        exit(1);
    }
    ang_ave = (ang_east + ang_south + ang_west + ang_north)/4.0;
printf("nAverage angle of rotation = %f radians",ang_ave);
}

/*This routine loads an array with the coords of the holes in the sl phantom.
The data set is rotated by the angle ang_ave calculated in the previous
routine*/

void sl_cartesian()
{
  int i,j,count;

  /*Allocate memory space for arrays "sl_x" and "sl_y"*/
  printf("n\n%lu bytes left before array space allocated",farcrolef());
  if ((sl_x = (float *)
      farcalloc((long)193,sizeof (float))) == NULL ||
      (sl_y = (float *)
       farcalloc((long)193,sizeof (float))) == NULL)
  {
    printf("nInsufficient space for pixel array");
    exit(1);
  }

  count = 0;
  printf("nputting 2nd header in output file");
  printf("ncount = %i",count);

  /*Load the central data*/
  for(j=-4; j<=6; j++)
  {
    for(i=-6; i<=6; i++)
    {
      sl_x[count] = 28.0 * (float)i;
      sl_y[count] = 28.0 * (float)j;
      count++;
    }
  }
  printf("ntop row");
  printf("ncount = %i",count);

  /*Load top row*/
  for(i=count; i<=count+6; i++)
  {
    sl_y[i] = 224.0;
  }
  count = count + 6;

  sl_x[count-6] = -84.0;
  sl_x[count-5] = -56.0;
  sl_x[count-4] = -28.0;
  sl_x[count-3] =  28.0;
  sl_x[count-2] =  56.0;
  sl_x[count-1] =  84.0;

  printf("nsecond row");
  printf("ncount = %i",count);

  /*Load second row*/
  for(i=-4; i<=4; i++)
  {
    sl_x[count] = 28.0 * (float)i;
    sl_y[count] = 196.0;
    count++;
  }
printf("bottom row");
printf("count = %i", count);
/*Load bottom row*/
for(i=-4; i<=4; i++)
{
    sl_x[count] = 28.0 * (float)i;
    sl_y[count] = -140.0;
    count++;
}
printf("First col");
printf("count = %i", count);
/*Load first column*/
for(i=-2; i<=2; i++)
{
    sl_x[count] = -224.0;
    sl_y[count] = 28.0 * (float)i;
    count++;
}
printf("2nd col.");
printf("count = %i", count);
/*Load second column*/
for(i=-3; i<=4; i++)
{
    sl_x[count] = -196.0;
    sl_y[count] = 28.0 * (float)i;
    count++;
}
printf("2nd last col");
printf("count = %i", count);
/*Load second last column*/
for(i=-3; i<=4; i++)
{
    sl_x[count] = 196.0;
    sl_y[count] = 28.0 * (float)i;
    count++;
}
printf("last col.");
printf("count = %i", count);
/*Load last column*/
for(i=-2; i<=2; i++)
{
    sl_x[count] = 224.0;
    sl_y[count] = 28.0 * (float)i;
    count++;
}
printf("nlast col.");
printf("ncount = %i", count);
/*Load last column*/
for(i=0; i<=192; i++)
{
    sl_x[i] = (sl_x[i] * cos(-ang_ave)) - (sl_y[i] * sin(-ang_ave));
    sl_y[i] = (sl_x[i] * sin(-ang_ave)) + (sl_y[i] * cos(-ang_ave));
}
/*This routine ensures matching array indices for corresponding points in
distorted and undistorted space*/

void synchronise_arrays()
{
    int i,j,num;

    /*Need a scratch array for reordering purposes*/
    float far *tmp_peak_x;
    float far *tmp_peak_y;

    /*Allocate space for temp arrays*/
    if (((tmp_peak_x = (float*))
        farcalloc((long)200,sizeof(float))) == NULL ||
        (tmp_peak_y = (float*)
        farcalloc((long)200,sizeof(float))) == NULL)
    {
        printf("Insufficient space for temporary arrays");
        exit(1);
    }

    /*Load temp arrays with peak_x and peak_y data. Then put values of 1000.0
     in peak_x and peak_y, as the number of peaks found may be more or less
     than the number of sl points (=193)*/
    for (i=0; i<=199; i++)
    { 
        tmp_peak_x[i] = peak_x[i];
        tmp_peak_y[i] = peak_y[i];
        peak_x[i] = 1000.0;
        peak_y[i] = 1000.0;
    }

    /*For each actual sl point find the corresponding point in tmp_peak_x
     and tmp_peak_y*/
    for (i=0; i<=192; i++)
    { 
        num = 0;
        for (j=0; j<=199; j++)
        {  
            /*Finds point in image space that is within 14mm of the point in
             undistorted space*/
            if(sqrt(pow((sl_x[i]-tmp_peak_x[j]),2) + pow((sl_y[i]-tmp_peak_y[j]),2)) < 14.0)
            {
                num++;
                if(num > 1)
                {  
                    printf("Error, more than one peak found near actual sl point!!!");
                    printf("Coordinate in undistorted space = (%f,%f)\n",sl_x[i],sl_y[i]);
                    exit(1);
                }
                peak_x[i] = tmp_peak_x[j];
                peak_y[i] = tmp_peak_y[j];
            }
        
        } 
    
    pause();
    /*Free up temporary array space*/
    farfree((float far *)tmp_peak_x);
    farfree((float far *)tmp_peak_y);
}
void xyz_mag()
{
    float dist_mag;

    /*Calculate the magnitude of distortion at each SL coord. and output to file*/
    fprintf(outfile,"X,Y,Distortion Magnitude:\n");

    for (i=0; i <= 192; i++)
    {
        /*Don't output if peak not found from above*/
        if (peak_x[i]==1000 || peak_y[i]==1000) continue;

        /*Calculate distortion magnitude*/
        dist_mag=sqrt((pow((sl_iil-peak_x[i]),2))+(pow((sl_y[i]-peak_y[i]),2)));

        /*Output sl_x,sl_y,dist_mag to file*/
        fprintf(outfile, "%f,%f,%f\n", sl_x[i], sl_y[i], dist_mag);
    }
}
Appendix C: AAPM Distortion Assessment Subroutine

Note that this subroutine replaces the subroutine of the same name in dist_mag.c in appendix B.

```c
void xyz_mag()
{
    float true_dim, observed_dim, percent_dist;

    /* Calculate the magnitude of distortion at each SL coord. and output to file */
    fprintf(outfile, "X,Y,AAPM % Distortion:\n");

    for (i=0; i <= 199; i++)
    {
        /* Don't output if peak not found from above */
        if (peak_x[i] == 1000 || peak_y[i] == 1000) continue;

        /* Calculate True Dimension: Origin --> Phantom SL Point */
        true_dim = sqrt(pow((sl_x[i]-0.0), 2) + pow((sl_y[i]-0.0), 2));

        /* Calculate Observed Dimension: Origin --> Image SL Point */
        observed_dim = sqrt(pow((peak_x[i]-0.0), 2) + pow((peak_y[i]-0.0), 2));

        /* Trap for origin to avoid divide by zero */
        if (sl_x[i] == 0.0)
        {
            fprintf(outfile, "0,0,0\n");
            continue;
        }

        /* Calculate % Distortion */
        percent_dist = fabs((true_dim - observed_dim) / true_dim * 100.0);

        /* Output sl_x,sl_y,dist_mag to file */
        fprintf(outfile, "%f,%f,%f\n", sl_x[i], sl_y[i], percent_dist);
    }
}
```
Appendix D: Annular Distortion Assessment Subroutine

Note that this subroutine replaces the subroutine called "xyz_mag()" in appendix B. The variable "inc" (line 12) is defined at the top of the program (not listed in this appendix) and was given the arbitrary value of 40 mm.

```c
/*This routine studies annuli concentric about the origin of the S.L. phantom, and quantifies in millimetres the maximum distortion present in each annulus.*/
void max_radial_distortion()
{
  int i,j,k,r;
  float dist_err,max_err;

  k=0;
  fprintf(outfile,"Radial_Distortion
          ");
  fprintf(outfile,"Radius     Distortion/mm
          ");

  /*Max radius for any point = 239.23mm*/
  for(r=0; r<=200; r=r+inc)
  {
    max_err = 0;
    for(i=0; i<=192; i++)
    {
      /*Tests if S.L. point is within the current annulus*/
      if(sqrt(pow(sl_x[i],2) + pow(sl_y[i],2)) >= (float)r &&
        sqrt(pow(sl_x[i],2) + pow(sl_y[i],2)) < (float)(r+inc))
        {
          for(j=0; j<=192; j++)
          {
            /*Tests if point in image space is < 14mm from the current S.L. point*/
            if(sqrt(pow((sl_x[j]-peak_x[i]),2) + pow((sl_y[j]-peak_y[i]),2)) < 14.0)
              {
                dist_err=sqrt(pow((sl_x[j]-peak_x[i]),2) + pow((sl_y[j]-peak_y[i]),2));
                if(dist_err > max_err) max_err = dist_err;
                k++;
                printf("%k=%d",k);
              }
          }
        }
    }
  }
  fprintf(outfile,"%f     %f
          ",((float)r+((float)inc/2.0)),max_err);
}
```
Appendix E: Distribution of Distortion Magnitude Subroutine

Note that this subroutine replaces the subroutine called "xyz_mag()" in appendix B. The variable "mmbin_width" (line 7) is defined at the top of the program (not listed in this appendix) and was given the arbitrary value of 0.25 mm as the width of each scoring bin. The variable "array_dim" was also defined at the top of the program and was set to 100.

```c
void dist_distrib()
{
    float dist_mag;

    /*fill array distrib with zeros*/
    for (i=0; i < array_dim; i++) distrib[i] = 0;
    dist_num = 0;

    fprintf(outfile,"Bin width = \%f\n",mmbin_width);

    for (i=0; i <= 192; i++)
    {
        /*Don't output if peak not found from above*/
        if (peak_x[i]==1000 || peak_y[i]==1000) continue;

        /*Calculate distortion magnitude*/
        dist_mag=sqrt((pow((sl-x-lil-peak_x[i]),2))+(pow((sl_y-peak_y[i]),2)));

        /*Increment appropriate bin*/
        distrib[abs((dist_mag)/mmbin_width)] =
        distrib[abs((dist_mag)/mmbin_width)] + 1;
        dist_num = dist_num + 1;
    }

    /*Output number of points making up distribution*/
    fprintf(outfile,"No.in Dist. = %\n",dist_num);
    fprintf(outfile,"Bin Lower, No.in Bin\n");

    for (i = 0; i < array_dim; i++)
    {
        /*Output Distribution data to file*/
        fprintf(outfile,"%f,%\n",((i*mmbin_width),distrib[i]);
    }
}```
Appendix F: Distorted Image Resampling Coordinate Program

/*dist_cor.c*/

/* THIS PROGRAM PRODUCES AN OUTPUT FILE OF COORDINATES FOR */
/* RESAMPLING THE */
/* DISTORTED IMAGE */

/* THE OUTPUT FILE IS BINARY AND IS IN WORD FORMAT (i.e. 16 bit per entry): */
/* (NB: The coords are multiplied by 100 so as to use integers in program */
/* RESAMPLE) */

byte address 0 : scale_fact (# of pixels in 10 cm)
byte address 2 : x pixel resampling coord for pixel [0,0]
byte address 4 : y pixel resampling coord for pixel [0,0]
byte address 6 : x pixel resampling coord for pixel [1,0]
byte address 8 : y pixel resampling coord for pixel [1,0]
byte address 262142 : x pixel resampling coord for pixel [255,255]
byte address 262144 : y pixel resampling coord for pixel [255,255]
<eof>

W. Beckham
April 1994

W. Beckham
April 1994

#include <stdio.h>
#include <dos.h>
#include <alloc.h>
#include <math.h>
#include <stdlib.h>

#define matrix-raw(n,m) matrix l((unsigned)m + n * 256)
#define matrix-mask(n,m) mmatrix l((unsigned)m + n * 256)
#define trans-x(n,m) nmatrix l((unsigned)m + n * 256)
#define trans-y(n,m) omatrix l((unsigned)m + n * 256)
#define TRUE 1
#define FALSE 0

/*Threshold ratio ith/i-1th for detection of possible peak*/
#define peak_threshold 1.3

/*Threshold for rejection of annulus1 max/annulus2 max ratio which signifies */
/* a false peak due to artifacts*/
#define reject_ratio 1.17

/*Peak height rejection threshold*/
#define peak_height_min 50

FILE *infile;
FILE *outfile;
void pause(void);
void initialise(void);
void row_scan(void);
void local_max_search(void);
void centre_of_gravity(void);
void output_cartesian(void);
void sl_out_cartesian(void);
void synchronise_arrays(void);
void resampling_transform(void);

unsigned short huge *matrix;
char huge *mmatrix;
float huge *mmatrix;
float huge *omatrix;
float far *sl_x;
float far *sl_y;
unsigned x_dim,y_dim, scale_fact, czone_max, dc_floor, local_max;
unsigned ann1_max, ann2_max, pix_sum_ann1, next_x;
long sum_wx, sum_wy, total_wt;
int i,j,x,y,m,n,local_pixmax_x, local_pixmax_y, num_peak, fifteen_mm, file_out;
char input[40], output[40], out_ans[5];
float far *sl_x;
float far *sl_y;
float cog_x, cog_y, origin_x, origin_y, peak_x[200], peak_y[200], coord_x, coord_y;
float ang_ave;

/*Set up structure to enable easy byte swap from MRI VAX*/

union
{
unsigned short word;
struct
{
unsigned char first;
unsigned char second;
} byte;
} rvalue;

main()
{

/*User input of filename*/
printf("Input image filename -- > ");
scanf("%s", input);

/*Opens file for read and exits if not found*/
if ((infile = fopen(input, "rb")) == NULL)
{
printf("Could not open file %s", input);
exit(1);
}

/*Prompt user for output file name*/
printf("Enter name of output file -- > ");
scanf("%s", output);
if ((outfile = fopen(output, "wb")) == NULL)
{
printf("Could not open output file");
exit(1);
}

/*Set up initial conditions for the image search and also load the input
file into an array*/
initialise();

/*Start scanning the pixel matrix*/
row_scan();
/*Free up mask array space & image array space*/
farfree((char *)mmatrix);
farfree((unsigned short *)matrix);

/*Place cent. of gravity of the found peaks data on a Cartesian coord. system to be viewed as in the image on the scanner and place in arrays peak_x and peak_y*/
output_cartesian();

/*Produce arrays (sl_x, sl_y) of coords of actual positions (appropriately rotated) of the S.L. phantom*/
sl_out_cartesian();

/*Synchronise the arrays sl_x --> peak_x and sl_y --> peak_y such that for each point in undistorted space the corresponding point in distorted space is easily found*/
synchronise_arrays();

/*Construct geometrically corrected image space by sampling in the correct position in distorted space. Sampling position is governed by solution of a series of three linear eqns. for x and a furthur three for y which describe the distortion of the three nearest known points to the current pixel in corrected space. Two arrays are produced which contain coordinate transformations for resampling of distorted image space. These become lookup values for the correction of future patient images*/
resampling_transform();

/*Close input and output files and free up array space*/
close(infile);
close(outfile);

} /*End main function*/

/*This routine sets up initial conditions for the search for peaks in the distorted image space. It also loads the input file to an array in order to speed up the search routine. A mask array is also loaded to prevent a peak being found more than once.*/
void initialise()
{
/*Fill arrays peak_x and peak_y with 1000.0. This is required later to take into account that there may be more (or less) peaks found in the peak search than 193*/
for(i=0; i<=199; i++)
{peak_x[i] = 1000.0;
peak_y[i] = 1000.0;

/* Sends file pointer to position of x matrix dimension */
seek(infile,650,0);

fread(&rvalue.word,sizeof(unsigned short),1,infile);

/* Perform byte swap */
x_dim=rvalue.byte.second << 8 | rvalue.byte.first;

/* Repeat for y dimension */
fread(&rvalue.word,sizeof(unsigned short),1,infile);
y_dim=rvalue.byte.second << 8 | rvalue.byte.first;

/* Test that matrix size is 256 x 256, if not then exit */
if (x_dim != 256 || y_dim != 256)
{
    printf("\nFILE PROBLEM: Matrix is %u x %u, not 256 x 256",x_dim,y_dim);
    exit(1);
}

/* Extract the number of pixels for 10.0 cm */
seek(infile,694,0);
fread(&rvalue.word,sizeof(unsigned short),1,infile);
scale_fact = rvalue.byte.second << 8 | rvalue.byte.first;
printf("\n\nThere are %u Pixels for 10 cm in image space",scale_fact);

/* Allocate memory space for array "matrix_raw" */
/* printf("\n\n%u bytes left before array space allocated",farcoreleft()); */
if ((matrix = (unsigned short *)farcalloc((long)x_dim*y_dim,sizeof (unsigned short))) == NULL)
{
    printf("\nInsufficient space for pixel array");
    exit(1);
}
printf("\n\nStarting address of pixel array = %Fp",matrix);

/* Fill array "matrix_raw" with pixel data */

/* Set pointer at first pixel location in input file */
printf("\n\nLoading pixel array with pixel data...");
seek(infile,4096,0);
for (y=0; y < y_dim; y++)
{
    for (x=0; x < x_dim; x++)
    {
        fread(&rvalue.word,sizeof(unsigned short),1,infile);
        matrix_raw(y,x) = rvalue.byte.second << 8 | rvalue.byte.first;
    }
    printf("\nFilled pixel array");
}

/* Test if array loaded correctly */
/* for (x=200; x < 256; x++)
printf("(%u,255) = %u",x,matrix_raw(255,x)); */

/* Allocate memory space for array matrix_mask to prevent a peak from being found more than once */
if ((mmatrix = (char *)farcalloc((long)x_dim*y_dim,sizeof (char))) == NULL)
{ printf("Insufficient space for mask array");
exit(1);
}
printf("Starting address of mask array = %fp\n",m_matrix);
/*printf("\n\n%lu bytes after all array space allocated",farcoreleft());*/

/*Fill array "matrix_mask" with mask data*/
printf("\n\nLoading mask array with mask data...");

for (y=0; y < y_dim; y++) {
for (x=0; x < x_dim; x++) {
matrix_mask(y,x) = 0;
}
}
printf("\nFilled mask array\n");

/*Go to zone at centre of matrix and find max. and min. to establish
height of peaks and DC noise floor*/
czone_max = 0;
dc_floor = 65535;
for (y=100; y<=150; y++) {
for (x=100; x<=150; x++) {
if (matrix_raw(y,x) < dc_floor) dc_floor = matrix_raw(y,x);
if (matrix_raw(y,x) > czone_max) czone_max = matrix_raw(y,x);
}
}
/*printf("\ndc_floor = %u, czone_max = %u",dc_floor,czone_max);*/

/*This routine governs the scanning of the image matrix to enable searching
for possible signal peaks. When one is found another routine is called
(local_max_search) that finds a local maxima and some rejection criteria
for artifacts*/
void row_scan()
{
num_peak = 0;
printf("\n\nStarting the search...\n");
for (y=10; y<240; y++)
for (x=3; x<253; x++) {
/*printf("\nx=%u, y=%u, Value=%u",x,y,matrix_raw(y,x));
divisor = ((float)matrix_raw(y,x+1)/(float)matrix_raw(y,x));
printf(" divisor=%f",divisor);*/
/*Rejects pixel if in air*/
if ((float)matrix_raw(y,x) < (float)(dc_floor/1.5)) continue;
/*Check mask matrix to reject already found peaks*/
while(matrix_mask(y,x)==1)
{
x++;  
/*printf("\nx=%u, y=%u, matrix_mask = %i"%,x,y,matrix_mask(y,x));*/
}

/*Checks to see if rising trend meets possible peak criteria*/
if (x!=255 && (float)matrix_raw(y,x+1)/(float)matrix_raw(y,x) > peak_threshold)  
{  
/*printf("\nx=%u, y=%u, calling local_max_search...",x,y);  
print("\nith+1th=%u, ith=%u",matrix_raw(y,x+1),matrix_raw(y,x));*/

local_max_search();
}
}
/*This routine finds a local pixel maxima near to the pixel that was  
identified as being a possible peak in routine row_scan. The top left of the  
area defined for the search is 2 pixels to the left, and 2 pixels above this  
pixel. Further it is 5 x 5 pixels in extent. The routine also contains some  
rejection criteria to take into account artifacts due to e.g. bright signal  
bands around the edges of the spatial linearity plates*/
void local_max_search()
{
  local_max = 0;
  /*This nested loop finds local max pixel positions in x and y*/
  for(j=y-1; j<y+4; j++)  
  {  
    for(i=x-2; i<x+3; i++)  
    {  
      /*printf("\ni=%u, j=%u",i,j);*/
      if (matrix_raw(j,i) > local_max)  
      {  
        local_max = matrix_raw(j,i);
        local_pixmax_x = i;
        local_pixmax_y = j;
      }
    }
  }
  /*printf("local_max = %u, at (%u,%u)\n",local_max,local_pixmax_x,local_pixmax_y);*/

  /*Find the pixel maximum in the annulus immediately surrounding the  
pixel maximum found above*/
  ann1_max = 0;
  for(j=local_pixmax_y-1; j<local_pixmax_y+2; j++)  
  {  
    for(i=local_pixmax_x-1; i<local_pixmax_x+2; i++)  
    {  
      if(local_pixmax_x == i && local_pixmax_y == j ) continue;
      if(matrix_raw(j,i) > ann1_max) ann1_max = matrix_raw(j,i);
    }
  }
  /*printf("\nann1_max = %u, ann1_max\n",ann1_max);*/
  /*Find the pixel maximum in the annulus immediately outside the one above*/
  ann2_max = 0;
  /*Scan the top & bottom rows of the annulus*/
for(i=local_pixmax_x-2; i<local_pixmax_x+3; i++)
{
    if(matrix_raw(i) > ann2_max) ann2_max = matrix_raw(i);
}

/*Scan the remaining 6 pixels in the second annulus*/
for(i=local_pixmax_x-2; i<local_pixmax_x+3; i=i+4)
{
    for(j=local_pixmax_y-1; j<local_pixmax_y+2; j++)
    {
        if(matrix_raw(i,j) > ann2_max) ann2_max = matrix_raw(i,j);
    }
    /*printf("\nann2_max = \%,ann2_max);*/
}

/*Check if need to reject peak due to surrounding high pixel values i.e. artifacts*/
if( ((float)ann1_max / (float)ann2_max > reject_ratio) && (local_max >
    peak_height_min))
{
    num_peak++;
    printf("\nNumber of peaks found = \%,num_peak);
    /*for(i=1; i<=sizeof(num_peak); i++)
    {
        printf("\b");
    }*/

    /*Mask out 30mm x 30mm area surrounding the maximum pixel in matrix_mask*/
    /*Work out how many pixels for 15mm*/
    fifteen_mm = (int)floor((float)scale_fact/100 * 15);
    for(j=local_pixmax_y-fifteen_mm; j<=local_pixmax_y+fifteen_mm; j++)
    {
        for(i=local_pixmax_x-fifteen_mm; i<=local_pixmax_x+fifteen_mm; i++)
        {
            matrix_mask(j,i) = 1;
        }
    }

    centre_of_gravity();
    /*else(printf("\nPeak rejected at (\%,\%),local_pixmax_x,local_pixmax_y));*/
}

/*This routine finds the centre of gravity of the 25 pixels surrounding the
maximum pixel of the peak found in the previous routine*/
void centre_of_gravity()
{
    sum_wx = 0;
    sum_wy = 0;
    total_wt = 0;
    for(j=local_pixmax_y-2; j<=local_pixmax_y+2; j++)
    {
        /*...*/
    }
}
for(i=local_pixmax_x-2; i<=local_pixmax_x+2; i++)
{
    sum_wx = sum_wx + (long)matrix_raw(j,i) * (long));
    sum_wy = sum_wy + (long)matrix_raw(j,i) * (long));
    total_wt = total_wt + (long)matrix_raw(j,i);
    }

cog_x = (float)sum_wx / (float)total_wt;
cog_y = (float)sum_wy / (float)total_wt;

/*Checks to see if peak found is near the position x=125 y=130 for the
purpose of calculating offsets to place the Cartesian origin over the
centre of this peak*/
if(local_pixmax_x > 117 && local_pixmax_x < 133 &&
    local_pixmax_y > 122 && local_pixmax_y < 138)
{
    origin_x = cog_x;
    origin_y = cog_y;
}

peak_x[num_peak-1] = cog_x;
peak_y[num_peak-1] = cog_y;

/*This routine converts the peak pixel position data into
Cartesian data as viewed on the MRI scanner. The origin is centred on the
peak at approx. pixel_x = 125, pixel_y = 130. It also finds an appropriate
angle of rotation to be used when placing the phantom actual peak positions
onto the same coordinate axes.*/

void output_cartesian()
{
    int i,num_fnd;
    float ang_north,ang_east,ang_south,ang_west;

    num_fnd = 0;

    /* Converts arrays to Cartesian system of coords.*/
    for(i=0; i<=num_peak-1; i++)
    {
        peak_x[i] = (peak_x[i] - origin_x)/((float)scale_fact/100.0);
        peak_y[i] = (-peak_y[i] + origin_y)/((float)scale_fact/100.0);
    }

    /*Calculate any rotation required to position phantom actual peak positions
on same Cartesian axes as image data. Rotation is calculated by taking the
mean angle of rotation about the origin, of the four nearest peaks to the
origin. NB the image data is NOT rotated*/
    for(i=0; i<=num_peak-1; i++)
    {
        if(peak_x[i] > 20.0 && peak_x[i] < 36.0 &&
            peak_y[i] > -8.0 && peak_y[i] < 8.0)
            ang_east = -atan(peak_y[i]/peak_x[i]);
            num_fnd++;
        else if(peak_x[i] > -8.0 && peak_x[i] < 8.0 &&
            peak_y[i] > -36.0 && peak_y[i] < -20.0)
397     if (peak_x[i] > -31.0 && peak_y[i] < -25.0 &&
398         peak_x[i] > -5.0 && peak_y[i] < 5.0)
399     {  
400         ang_south = atan(peak_x[i]/peak_y[i]);
401         num_fnd++;
402     }
403     else if (peak_x[i] > -31.0 && peak_x[i] < -25.0 &&
404              peak_y[i] > -5.0 && peak_y[i] < 5.0)
405     {  
406         ang₶st = atan(peak_y[i]/peak_x[i]);
407         num_fnd++;
408     }
409     else if (peak_x[i] > -8.0 && peak_x[i] < 8.0 &&
410            peak_y[i] > 20.0 && peak_y[i] < 36.0)
411     {  
412         ang_north = atan(peak_x[i]/peak_y[i]);
413         num_fnd++;
414     }
415     printf("num_fnd = %i",num_fnd);
416     if (num_fnd != 4)
417     {  
418         printf("Not all peaks found for rotation in routine output_cartesian");
419         exit(1);
420     }
421     ang_ave = (ang_east + ang_south + ang_west + ang_north)/4.0;
422     printf("Average angle of rotation = %f radians",ang_ave);
423     /*This routine loads an array with the coords of the holes in the sl phantom.
424        The data set is rotated by the angle ang_ave calculated in the previous
425        routine*/
426     void sl_out_cartesian()
427     {  
428         int i,j,count;
429     /*Allocate memory space for arrays "sl_x" and "sl_y"*/
430     /*printf("%lu bytes left before array space allocated",farcoreleft());*/
431     if ((sl_x=(float *))
432         faralloc((long)193,sizeof (float))) == NULL ||
433         (sl_y=(float *))
434         faralloc((long)193,sizeof (float))) == NULL)
435         {  
436             printf("Insufficient space for pixel array");
437             exit(1);
438         }
439     count = 0;
440     printf("Putting 2nd header in output file");
441     printf("%i",count);
442     /*Load the central data*/
443     for(j=-4; j<=6; j++)
444     {  
445         for(i=-6; i<=6; i++)
446         {  
447             sl_x[count] = 28.0 * (float)i;
448             sl_y[count] = 28.0 * (float)j;
449             count++;
450         }
451     }
printf("top row");
printf("count = ", count);
/*Load top row*/
for (i = count; i <= count + 6; i++) {
    sl_y[i] = 224.0;
    count = count + 6;
}
sl_x[count-6] = -84.0;
sl_x[count-5] = -56.0;
sl_x[count-4] = -28.0;
sl_x[count-3] = 28.0;
sl_x[count-2] = 56.0;
sl_x[count-1] = 84.0;
printf("second row");
printf("count = ", count);
/*Load second row*/
for (i = -4; i <= 4; i++) {
    sl_x[count] = 28.0 * (float)i;
    sl_y[count] = 196.0;
    count++;
}
printf("bottom row");
printf("count = ", count);
/*Load bottom row*/
for (i = -4; i <= 4; i++) {
    sl_x[count] = 28.0 * (float)i;
    sl_y[count] = -140.0;
    count++;
}
printf("First col.");
printf("count = ", count);
/*Load first column*/
for (i = -2; i <= 2; i++) {
    sl_x[count] = -224.0;
    sl_y[count] = 28.0 * (float)i;
    count++;
}
printf("2nd col.");
printf("count = ", count);
/*Load second column*/
for (i = -3; i <= 4; i++) {
    sl_x[count] = -196.0;
    sl_y[count] = 28.0 * (float)i;
    count++;
}
printf("2nd last col.");
printf("count = ", count);
/*Load second last column*/
for (i = -3; i <= 4; i++) {
    sl_x[count] = -196.0;
    sl_y[count] = 28.0 * (float)i;
    count++;
}
printf("n2nd last col.");
printf("count = ", count);
/*Load second last column*/
for (i = -3; i <= 4; i++) {
    sl_x[count] = -196.0;
    sl_y[count] = 28.0 * (float)i;
    count++;
}
sl_x[count] = 196.0;
sl_y[count] = 28.0 * (float)i;
count++;
}

printf("\nlast col.");
printf("\ncount = %i",count);

/*Load last column*/
for(i=-2; i<=2; i++)
{
sl_x[count] = 224.0;
sl_y[count] = 28.0 * (float)i;
count++;
printf("\ncount = %i",count);
}

/*Load data by angle ang_ave using standard rotation transformation. Note
that in rot. transformation the angle convention is negative of standard.*/
for(i=0; i<=192; i++)
{
sl_x[i] = (sl_x[i] * cos(-ang_ave)) - (sl_y[i] * sin(-ang_ave));
sl_y[i] = (sl_x[i] * sin(-ang_ave)) + (sl_y[i] * cos(-ang_ave));
}

/*This routine ensures matching array indices for corresponding points in
distorted and undistorted space*/

void synchronise_arrays()
{
    int i,j,num;

    /*Need a scratch array for reordering purposes*/
    float far *tmp_peak_x;
    float far *tmp_peak_y;

    /*Allocate space for temp arrays*/
    if ((tmp_peak_x = (float*)farcalloc((long)200,sizeof(float))) == NULL)
        if ((tmp_peak_y = (float*)farcalloc((long)200,sizeof(float))) == NULL)
            printf("\nInsufficient space for temporary arrays\n");
        exit(1);

    /*Load temp arrays with peak_x and peak_y data. Then put values of 1000.0
    in peak_x and peak_y, as the number of peaks found may be more or less
    than the number of sl points (=193)*/
    for (i=0; i<=199; i++)
    {
        tmp_peak_x[i] = peak_x[i];
        tmp_peak_y[i] = peak_y[i];
        peak_x[i] = 1000.0;
        peak_y[i] = 1000.0;
    }

    /*For each actual sl point find the corresponding point in tmp_peak_x
    and tmp_peak_y*/
for (i=0; i<=192; i++)
{
    num = 0;
    for (j=0; j<=199; j++)
    {
        /*Finds point in image space that is within 14mm of the point in
        undistorted space*/
        if(sqrt(pow((sl_x[i]-tmp_peak_x[j]),2) + pow((sl_y[i]-tmp_peak_y[j]),2)) < 14.0)
        {
            num++;
            if(num > 1)
            {
                printf("Error, more than one peak found near actual sl point!!!!");
                printf("In Coordinate in undistorted space = (\%f,\%f), sl_x[i],sl_y[i]);
                exit(1);
            }
            peak_x[i] = tmp_peak_x[j];
            peak_y[i] = tmp_peak_y[j];
        }
    }
    /*Free up temporary array space*/
    farfree((float*)tmp_peak_x);
    farfree((float*)tmp_peak_y);
}

/*This routine forms two arrays of lookup tables describing where to resample
the distorted image space to reconstruct a geometrically corrected image.
One array contains the x coordinate for resampling and the other the y coord.
The arrays are written to a file for use in transforming distorted images*/
void resampling_transform()
{
    int i,j,k,index_min[3],valid_coord_cnt,iteration_cnt,loop,line_flag;
    int index_sml,index_min_dum,int_trans_x,int_trans_y;
    double idiff,diff,fdiff;
    float diff,diff1,fdiff1;
    float x_corr,y_corr,sl_dist[193],dist_init,dist_init_last,dist_min[3];
    float x_cart_samp,x_pix_samp,grad1,grad2,dist1,dist2,ave_dist_sml;
    float m2,m3,Y2,Z2,X2p,Y3,Z3,X3p,m3p,Zeta3,Chi3p,eta_x,beta_x,alpha_x;
    float Y2_dum,Z2_dum,X2p_dum,alpha_y,eta_y,eta_y_cart_samp,y_pix_samp;

    /*Allocate space for arrays trans_x and trans_y*/
    /*printf("\n\n%lu bytes left before array space allocated",farcoreleft());*/
    if ((inmatrix = (float *))
    faralloc((long)x_dim*y_dim,sizeof (float)) == NULL)
    {
        printf("Insufficient space for trans_x array");
        exit(1);
    }
    /*Switch off array allocation for now*/
    /*printf("%lu bytes left before array space allocated",farcoreleft());*/
    if ((omatrix = (float *))
    faralloc((long)x_dim*y_dim,sizeof (float)) == NULL)
    {
        printf("Insufficient space for trans_y array");
        exit(1);
    }
/*Loops to cover all pixels (0,0) to (255,255)*/
for (j=0; j < y_dim; j++)
{
  for (i=0; i < x_dim; i++)
  {
    /*Convert pixel location to (x,y) Cartesian coords*/
    x_cor = ((float)i - origin_x)/(float)scale_fact100;
    y_cor = (-float)j + origin_y)/(float)scale_fact100;
    /*print("%f, %f, %f", x_cor, y_cor, x_cor,y_cor);*/

    /*Need to search sl_x and sl_y for the three nearest spatial linearity coordinates*/
    for (k=0; k <= 192; k++)
    {
      /*Measure all distances*/
      sl_dist[k] = sqrt(pow((x-cor-sl_x[k]),2)+pow((y_cor-sl_y[k]),2));

      /*Determine the closest 3 distances and reject sl points that don't have a corresponding peak found in distorted space*/
      dist_init_last = 0.0;
      valid_coord_cnt = 0;
      while (valid_coord_cnt <= 2)
      {
        dist_init = 10000.0;
        for (k=0; k <= 192; k++)
        {
          if (sl_dist[k] < dist_init && sl_dist[k] > dist_init_last &&
              peak_x[k] != 1000.0)
          {
            dist_init = sl_dist[k];
            dist_min[valid_coord_cnt] = sl_dist[k];
            index_min[valid_coord_cnt] = k;
          }
        }
        dist_init_last = dist_min[valid_coord_cnt];
        valid_coord_cnt++;
      }

      /*Need to trap for the condition of the three points lying in a straight line. If they do then a new point has to be chosen after one has been rejected*/

      /*Reset line flag*/
      line_flag = 0;

      /*Do they form a straight line?*/

      /*Check for Infinite slope!*/
      if (sl_x[index_min[1]]-sl_x[index_min[0]] == 0.0 &
          sl_x[index_min[2]]-sl_x[index_min[1]] == 0.0) line_flag = 1;

      /*Check for straight line*/
      if (line_flag == 0)
      {
        grad1 = (sl_y[index_min[1]]-sl_y[index_min[0]])/
                 (sl_x[index_min[1]]-sl_x[index_min[0]]);
        grad2 = (sl_y[index_min[2]]-sl_y[index_min[1]])/
                 (sl_x[index_min[2]]-sl_x[index_min[1]]);
        if ((grad2 < 1.1 * grad1 & grad2 > 0.9 * grad1) ||
            (grad2 > 1.1 * grad1 & grad2 < 0.9 * grad1)) line_flag = 1;
      }
    }
  }
}
/*If the points are indeed co-linear we need to pick a third point which
is nearest to both of the closest two (but not the third co-linear
point!!!). sl_*[index_min[0]] is closest, sl_*[index_min[1]] is next and
sl_*[index_min[2]] is furthest of the three points away*/
if (line_flag == 1)
{
  /*printf("THE THREE POINTS LIE ON A STRAIGHT LINE!!!!!!!!!!!!!!!!!");*/
  /*Search arrays sl_x and sl_y for the next nearest point*/

  /*Reset line_flag*/
  line_flag = 0;

  ave_dist_sml = 10000.0;
  for (k=0; k<=192; k++)
  {
    /*Reject current 3 points and also any which don't have a corresponding
    point in distorted space (i.e. where peak_x[k] = 1000.0). Also reject
    any further co-linear points!!!*/
    if (k != index_min[0] && k != index_min[1] && k != index_min[2] &&
        peak_x[k] != 1000.0)
    {
      /*Check that current point is not co-linear with the first three*/
      /*Check for infinite slope*/
      if ((grad2 < 1.1 * grad1 &&& grad2 > 0.9 * grad1) ||
          (grad2 > 1.1 * grad1 &&& grad2 < 0.9 * grad1)) line_flag = 1;

      if (line_flag == 0)
      {
        grad1 = (sl_y[k]-sl_y[index_min[0]])/
            (sl_x[k]-sl_x[index_min[0]]);
        grad2 = (sl_y[index_min[2]]-sl_y[index_min[1]])/
            (sl_x[index_min[2]]-sl_x[index_min[1]]);
        if ((grad2 < 1.1 * grad1 &&& grad2 > 0.9 * grad1) ||
            (grad2 > 1.1 * grad1 &&& grad2 < 0.9 * grad1)) line_flag = 1;
      }
      if (line_flag == 0)
      {
        dist1 = sqrt(pow((sl_x[index_min[0]]-sl_x[k],2)+
                        pow((sl_y[index_min[0]]-sl_y[k],2);
        dist2 = sqrt(pow((sl_x[index_min[1]]-sl_x[k],2)+
                        pow((sl_y[index_min[1]]-sl_y[k],2);
        if (dist1 + dist2/2 < ave_dist_sml)
        {
          ave_dist_sml = (dist1 + dist2)/2;
          index_sml = k;
        }
      }
      /*Reset (sl_x[index_min[2]],sl_y[index_min[2]]) to (sl_x[k],sl_y[k])*/
      index_min[2] = index_sml;
      /*printf("THE NEW POINT IS (%f,%f), sl_x[index_min[2]],sl_y[index_min[2]]");*/
    }
    /*printf("%d\n", f, k=%i", dist_min[0],index_min[0]);*/
    /*printf("%d\n", f, k=%i", dist_min[1],index_min[1]);*/
    /*printf("%d\n", f, k=%i", dist_min[2],index_min[2]);*/
    /*Set up system of linear equations to describe the distortion in the
area of the three nearest points identified in the previous code
the equations are of the form:

\[ x_1' = \alpha(x) \cdot x_1 + \beta(x) \cdot y_1 + \eta(x) \]
\[ x_2' = \alpha(x) \cdot x_2 + \beta(x) \cdot y_2 + \eta(x) \]
\[ x_3' = \alpha(x) \cdot x_3 + \beta(x) \cdot y_3 + \eta(x) \]

\[ y_1' = \alpha(y) \cdot x_1 + \beta(y) \cdot y_1 + \eta(y) \]
\[ y_2' = \alpha(y) \cdot x_2 + \beta(y) \cdot y_2 + \eta(y) \]
\[ y_3' = \alpha(y) \cdot x_3 + \beta(y) \cdot y_3 + \eta(y) \]

where: \((x_1', y_1')\)
\((x_2', y_2')\)
\((x_3', y_3')\) are the sl points in uncorrected image space

\[ (x_1, y_1) \]
\[ (x_2, y_2) \]
\[ (x_3, y_3) \] are the sl points in corrected space */

/*This solves for the x coordinate for resampling*/
/*Use Gauss Elimination & Back-substitution method to solve the systems*/
/*See notes for variable definitions*/
/*printf("x1'=\%f,x2'=\%f,x3'=\%f",peak_x[index_min[0]],peak_x[index_min[1]],peak_x[index_min[2]]);*/
/*printf("y1'=\%f,y2'=\%f,y3'=\%f",peak_y[index_min[0]],peak_y[index_min[1]],peak_y[index_min[2]]);*/
/*printf("x1'=\%f,x2'=\%f,x3'=\%f",sl_x[index_min[0]],sl_x[index_min[1]],sl_x[index_min[2]]);*/
/*printf("y1'=\%f,y2'=\%f,y3'=\%f",sl_y[index_min[0]],sl_y[index_min[1]],sl_y[index_min[2]]);*/

/*Need to ensure that equ. (1) != 0. If it is then swap (1) and (2)*/
if(sl_x[index_min[0]] == 0.0 || sl_y[index_min[0]] == 0.0)
{
   index_min_dum = index_min[0];
   index_min[0] = index_min[1];
   index_min[1] = index_min_dum;
}

/*Determine Elimination Multipliers*/
m2 = sl_x[index_min[1]] / sl_x[index_min[0]];
m3 = sl_x[index_min[2]] / sl_x[index_min[0]];
Y2 = (sl_y[index_min[1]] - (m2 * sl_y[index_min[0]]));
Z2 = (1.00 - (m2 * 1.00));
X2p = (peak_x[index_min[1]] - (m2 * peak_x[index_min[0]]));
Y3 = (sl_y[index_min[2]] - (m3 * sl_y[index_min[0]]));
Z3 = (1.00 - (m3 * 1.00));
X3p = (peak_x[index_min[2]] - (m3 * peak_x[index_min[0]]));

/*Check that Y2 is not 0.0 to avoid zero divide. If it is then interchange equn. (4) and (5)*/
if ( Y2 < 0.00001 && Y2 > -0.00001 )
{
   Y2_dum = Y2;
   Y2 = Y3;
   Y3 = Y2_dum;
   Z2_dum = Z2;
   Z2 = Z3;
   Z3 = Z2_dum;
   X2p_dum = X2p;

X2p = X3p;
X3p = X2p_dum;

/*printf("nm2=%f10,sl_y1=%f10",m2,sl_y[index_min[0]],sl_y[index_min[1]]);*/
/*printf("nY3=%f20",Y3);*/
/*printf("nY2=%f20",Y2);*/
m3p = Y3 / Y2;

Zeta3 = (Z3 - (m3p * Z2));
Chi3p = (X3p - (m3p * X2p));

/*Perform Back-substitution*/
eta_x = Chi3p / Zeta3;
beta_x = (X2p - (eta_x * Z2)) / Y2;
alpha_x = (peak_x[index_min[0]] - (beta_x * sl_x[index_min[0]]) - (eta_x * sl_x[index_min[0]]));

/*This solves for the y coordinate for resampling*/
/*Use Gauss Elimination & Back-substitution method to solve the systems*/
/*See notes for variable definitions*/
/*printf("nx=%f,y=%f",peak_x[index_min[0]],peak_x[index_min[1]],peak_x[index_min[2]]);*/
/*printf("ny1=%f,y2=%f",peak_y[index_min[0]],peak_y[index_min[1]],peak_y[index_min[2]]);*/
/*printf("nx1=%f,x2=%f",sl_x[index_min[0]],sl_x[index_min[1]],sl_x[index_min[2]]);*/
/*printf("ny1=%f,y2=%f",sl_y[index_min[0]],sl_y[index_min[1]],sl_y[index_min[2]]);*/

if(sl_x[index_min[0]] == 0.0 || sl_y[index_min[0]] == 0.0)
{
    index_min_dum = index_min[0];
    index_min[0] = index_min[1];
    index_min[1] = index_min_dum;
}
/* Determine Elimination Multipliers */
m2 = sl_x[index_min[1]] / sl_x[index_min[0]];  
m3 = sl_x[index_min[2]] / sl_x[index_min[0]];  

Y2 = (sl_y[index_min[1]] - (m2 * sl_y[index_min[0]]));  
Z2 = (1.00 - (m2 * 1.00));  
X2p = (peak_y[index_min[1]] - (m2 * peak_y[index_min[0]]));

Y3 = (sl_y[index_min[2]] - (m3 * sl_y[index_min[0]]));  
Z3 = (1.00 - (m3 * 1.00));  
X3p = (peak_y[index_min[2]] - (m3 * peak_y[index_min[0]]));

/* Check that Y2 is not 0.0 to avoid zero divide below. If it is then interchange eqn. (4) and (5) */
if ( Y2 < 0.00001 & & Y2 > -0.00001 )  
{  
   Y2_dum = Y2;  
   Y2 = Y3;  
   Y3 = Y2_dum;  
   Z2_dum = Z2;  
   Z2 = Z3;  
   Z3 = Z2_dum;  
   X2p_dum = X2p;  
   X2p = X3p;  
   X3p = X2p_dum;  
}

/*print("\nm2=",m2,sl_y0=",sl-y1=",sl-y[index_min[0]],sl-y[index_min[1]]);*/
/*print("\nY3=",Y3);*/
/*print("\nY2=",Y2);*/

m3p = Y3 / Y2;  

Zeta3 = (Z3 - (m3p * Z2));  

Chi3p = (X3p - (m3p * X2p));

/*print("\nm3p=%g",m3p);*/
/*print("\nY2=",Y2);*/
/*print("\nY3=",Y3);*/
/*print("\nZeta3=",Zeta3);*/
/*print("\nChi3p=",Chi3p);*/

/* Perform Back-substitution */
eta_y = Chi3p / Zeta3;  
beta_y = (X2p - (eta_y * Z2)) / Y2;  
alpha_y = (peak_y[index_min[0]])-  
   (beta_y * sl_y[index_min[0]])-(eta_y*1)/  
   sl_x[index_min[0]];  

/*print("\nalpha_y=%f, beta_y=%f, eta_y=%f",alpha_y,beta_y,eta_y);*/
/*print("\norigin_x=%f, origin_y=%f",origin_x,origin_y, scale_fact);*/

/* Compute the value of the y pixel coordinate sampling point in distorted */  
space and store this in the array trans_y(n,h)*
y_cart_samp = (alpha_y * x_corr) + (beta_y * y_corr) + eta_y;  
y_pix_samp = origin_y - ((float)scale_fact*100 * y_cart_samp);
trans_y(j,i) = y_pix_samp;
/*printf("nx_corr=%f, y_corr=%f, y_cart_samp=%f",x_corr,y_corr,y_cart_samp);*/
/*printf("nxpix = %i ypix = %i y_pix_samp = %f",i,j,y_pix_samp);*/
}

/*Load Scale Factor & Arrays trans_x and trans_y to output file*/

/*Output Scale Factor*/
if (fwrite(&scale_fact,sizeof(int),1,outfile) == 0)
{
printf("Binary write to output file of scale_fact failed !!!");
exit(1);
}

/*Loop to cover all pixels (0,0) (255,255)*/
for (j=0; j < y_dim; j++)
{
for (i=0; i < x_dim; i++)
{
int_trans_x = (int)(trans_x(j,i)*100);
if (fwrite(&int_trans_x,sizeof(int),1,outfile) == 0)
{
printf("Binary write to output file of x pixel coord failed !!!");
exit(1);
}

int_trans_y = (int)(trans_y(j,i)*100);
if (fwrite(&int_trans_y,sizeof(int),1,outfile) == 0)
{
printf("Binary write to output file of y pixel coord failed !!!");
exit(1);
}
Appendix G: Distortion Correction (Resampling) Program

/* resample.c

This program resamples an MRI image with previously determined
resampling coordinates to produce a geometrically correct image.

W. Beckham
June 1994

***************************************************************************/

#include <stdio.h>
#include <stdlib.h>
#include <dos.h>
#include <alloc.h>
#include <math.h>
#define matrix ima(n,m) matrix (((unsigned)m + n * 256))
#define matrix-cor(n,m) mmatrix (((unsigned)m + n * 256))
#define x-sam-coord(n,m) nmatrix (((unsigned)m + n * 256))
#define y-sam-coord(n,m) omatrix (((unsigned)m + n * 256))
#define TRUE 1
#define FALSE 0

FILE *infile1;
FILE *infile2;
FILE *outfile;
void pause(void);
void initialise(void);
void resample(void);
void output-mri-file(void);

unsigned short huge *matrix;
unsigned short huge *mmatrix;
int huge *nmatrix;
int huge *omatrix;

char input[12],input2[12],output[12],out_ans[5],buffer[4096];
unsigned scale_fact_ima,scale_fact_sam;
unsigned x_dim,y_dim;
int x,y;
main()
{
    /*Initialise input and output streams*/
    initialise();

    /*Resample the distorted image*/
    resample();

    /*Rebuild a corrected MRI file*/
    output-mri_file();
    printf("\nBACK IN MAIN!!!\n");

    /*Close input and output files and free up array space*/
    free((unsigned short far *)matrix);
    printf("\n1");
}
/*farfree((unsigned short * )mmatrix);*/
printf("\n2");
/*farfree((int * )mmatrix);*/
printf("\n3");
/*farfree((int * )omatrix);*/

printf("\n4");
printf("\n5");
printf("\n6");
printf("\n7");
printf("\n8");

void pause()
{
    printf("Hit return to continue ... ");
    getchar();
    getchar();
}

void initialise()
{
/*User input of uncorrected image filename*/
printf("\nInput uncorrected image filename --> ");
scanf("%s",input1);

/*Opens file for read and exits if not found*/
if ((infile1 = fopen(input1,"rb")) == NULL)
{
    printf("\nCould not open file %s",input1);
    exit(1);
}

/*User input of resampling coord filename*/
printf("\nInput resampling coord filename --> ");
scanf("%s",input2);

/*Opens file for read and exits if not found*/
if ((infile2 = fopen(input2,"rb")) == NULL)
{
    printf("\nCould not open file %s",input2);
    exit(1);
}

/*Opens output file*/
/*Prompt user for output file name*/
printf("\nEnter name of corrected image output file --> ");
scanf("%s",output);
if((outfile = fopen(output,"wb")) == NULL)
{
    printf("\nCould not open output file %s",output);
    exit(1);
}

/*Sends file pointer to position of x matrix dimension*/
fseek(infile1,650,0);
fwrite(&x_dim,sizeof(unsigned short),1,infile1);
/*Repeat for Y dimension*/

fread(&y_dim,sizeof(unsigned short),1,infile1);

/*Test that matrix size is 256 x 256, if not then exit*/

if (x_dim != 256 || y_dim != 256)
{
    printf("FILE PROBLEM: Matrix is %u x %u, not 256 x 256",x_dim,y_dim);
    exit(1);
}

/*Extract the number of pixels for 10.0 cm from the image file and check
that this agrees with the scale factor from the coord sampling file*/

/*Get factor from image file*/

fseek(infile1,694,0);

fread(&scale_fact_ima,sizeof(unsigned short),1,infile1);

/*Get factor from sampling file*/

fread(&scale_fact_sam,sizeof(int),1,infile2);

if ((int)scale_fact_ima != scale_fact_sam)
{
    printf("Problem!!! --> Scale Factors for the two files do not match");
    exit(1);
}

/*Allocate memory space for array "matrix_ima"*/

printf("%lu bytes left before array space allocated",farc coreleft());

if ((matrix =(unsigned short huge *)

falloc((unsigned long)x_dim*y_dim,(unsigned long)sizeof (unsigned short))) == NULL)
{
    printf("Insufficient space for pixel array");
    exit(1);
}

printf("%lu bytes left after matrix_ima array space allocated",farc coreleft());

printf("Starting address of pixel array = %P",matrix);

/*Fill array "matrix_ima" with pixel data*/

/*Set pointer at first pixel location in input file*/

printf("Loading pixel array with pixel data...");

fseek(infile1,4096,0);

for (y=0; y < y_dim; y++)
{
    for (x=0; x < x_dim; x++)
    {
        fread((void *)&matrix_ima(y,x),sizeof(unsigned short),1,infile1);
    }
}

fclose(infile1); /*Close infile1*/

fclose(infile1); /*fclose(infile1);*/

printf("Filled pixel array");

/*Test if array loaded correctly*/

for (x=200; x < 256; x++)
    printf("(%u,255) = %u",x,matrix_ima(255,x));
/* Allocate memory space for corrected image array matrix_cor */
printf("\n\n%lu bytes left before space allocated",farcoreleft());
if (((mmatrix = (unsigned short huge *))
farcalloc((unsigned long)x_dim*y_dim,(unsigned long)sizeof (unsigned short))) == NULL)
{
    printf("\nInsufficient space for corrected image array");
    exit(1);
}
printf("\n\n%lu bytes left after matrix_cor array space allocated",farcoreleft());
printf("\n\nStarting address of matrix_cor = %P",mmatrix);

/* Allocate memory space for resampling coordinate arrays */
if (((nmatrix = (int huge *))
farcalloc((unsigned long)x_dim*y_dim,(unsigned long)sizeof (int))) == NULL)
{
    printf("\nInsufficient space for x coord sampling array");
    exit(1);
}
if (((omatrix = (int huge *))
farcalloc((unsigned long)x_dim*y_dim,(unsigned long)sizeof (int))) == NULL)
{
    printf("\nInsufficient space for y coord sampling array");
    exit(1);
}
/* Load resampling coord arrays */
for (y=0; y < y_dim; y++)
{
    for (x=0; x < x_dim; x++)
    {
        fread((void *)&x_sam_coord(y,x),sizeof(int),1,infile2);
        fread((void *)&y_sam_coord(y,x),sizeof(int),1,infile2);
    }
}
/* Close infile2 */
close(infile2);

/* Test if array loaded correctly */
for (x=95; x < 105; x++)
printf("\n\n%u,%u)x=%i,y=%i",x,x_sam_coord(100,x),y_sam_coord(100,x));
printf("\n\n%lu bytes after all array space allocated",farcoreleft());
}

void resample()
{
    int l_bound_x,u_bound_x,l_bound_y,u_bound_y;
    int ly_uux_bit,ly_ux_bit,uy_ux_bit,uy_lex_bit,bit_flag;
    float x_wt_fact,y_wt_fact,grey_x1,grey_x2,grey_fin;

    /* Loop over entire pixel space of corrected image */
    for (y=0; y < y_dim; y++)

for (x=0; x < x_dim; x++)
{
  /*Make sure that resampling coord is inside the image bounds*/
  if (x_resam_coord(y,x)/100.0 >= 0.0 && y_resam_coord(y,x)/100.0 >= 0.0 &&
      x_resam_coord(y,x)/100.0 <= 256.0 && y_resam_coord(y,x)/100.0 <= 256.0)
    {
      /*Print "nxsam = %f, ysam = %f", x_resam_coord(y,x)/100.0, y_resam_coord(y,x)/100.0;*/
      /*Perform 4-way linear interpolation to determine sampling point pixel magnitude*/
      /*Establish upper & lower integer bounds for the sampling pixel coord*/
      l_bound_x = (int)(x_resam_coord(y,x)/100.0);
      l_bound_y = (int)(y_resam_coord(y,x)/100.0);
      u_bound_x = l_bound_x + 1;
      u_bound_y = l_bound_y + 1;
      /*Print "ul_bound_x=%i, l_bound_x=%i, l_bound_y=%i, u_bound_y=%i", l_bound_x, u_bound_x, l_bound_y, u_bound_y;*/
      /*Establish the relative weight factors dependant on the distance of the resampling coord from the bounding pixel centres*/
      x_wt_fact = x_resam_coord(y,x)/100.0 - (float)l_bound_x;
      y_wt_fact = y_resam_coord(y,x)/100.0 - (float)l_bound_y;
      /*Print "%f, y_wt_fact = %f", x_wt_fact, y_wt_fact;*/
      /*Print "%g1=%%g2=%%gu", matrix ima(l_bound_y, l_bound_x),
         matrix ima(l_bound_y, u_bound_x);*/
      /*Print "%g3=%%g4=%%gu", matrix ima(u_bound_y, l_bound_x),
         matrix ima(u_bound_y, u_bound_x);*/
      /*Need to take into account if ROI bits are set i.e. bits 13,14,15,16*/
      lyux_bit=0;
      lylx_bit=0;
      uyux_bit=0;
      uylx_bit=0;
      /*Bit flag set if any ROI bit is set in pixels being interpolated*/
      bit_flag = 0;
      if (((unsigned short)(matrix ima(l_bound_y, u_bound_x) & 32768) != 0))
        {
          lyux_bit = 16;
          matrix ima(l_bound_y, u_bound_x) =
            (unsigned short)(matrix ima(l_bound_y, u_bound_x) ^ 32768);
          bit_flag = 1;
          /*Print "lu=16;*/
        }
      if (((unsigned short)((matrix ima(l_bound_y, u_bound_x) & 16384) != 0))
        {
          lyux_bit = 15;
          matrix ima(l_bound_y, u_bound_x) =
            (unsigned short)(matrix ima(l_bound_y, u_bound_x) ^ 16384);
          bit_flag = 1;
          /*Print "lu=15;*/
        }
      if (((unsigned short)((matrix ima(l_bound_y, u_bound_x) & 8192) != 0))
        {
          }
ly_uX_bit = 14;
matrix_ima(l_bound_y,u_bound_x) =
(unsigned short)(matrix_ima(l_bound_y,u_bound_x) ^ 8192);
bit_flag = 1;
/*printf("\nlu=14");*/
}

if (((unsigned short)(matrix_ima(l_bound_y,u_bound_x) & 4096) != 0))
{
    ly_uX_bit = 13;
matrix_ima(l_bound_y,u_bound_x) =
(unsigned short)(matrix_ima(l_bound_y,u_bound_x) ^ 4096);
bit_flag = 1;
/*printf("\nlu=13");*/
}

if (((unsigned short)(matrix_ima(l_bound_y,l_bound_x) & 32768) != 0))
{
    ly_lx_bit = 16;
matrix_ima(l_bound_y,l_bound_x) =
(unsigned short)(matrix_ima(l_bound_y,l_bound_x) ^ 32768);
bit_flag = 1;
/*printf("\nll=16");*/
}

if (((unsigned short)(matrix_ima(l_bound_y,l_bound_x) & 16384) != 0))
{
    ly_lx_bit = 15;
matrix_ima(l_bound_y,l_bound_x) =
(unsigned short)(matrix_ima(l_bound_y,l_bound_x) ^ 16384);
bit_flag = 1;
/*printf("\nll=15");*/
}

if (((unsigned short)(matrix_ima(l_bound_y,l_bound_x) & 8192) != 0))
{
    ly_lx_bit = 14;
matrix_ima(l_bound_y,l_bound_x) =
(unsigned short)(matrix_ima(l_bound_y,l_bound_x) ^ 8192);
bit_flag = 1;
/*printf("\nll=14");*/
}

if (((unsigned short)(matrix_ima(l_bound_y,l_bound_x) & 4096) != 0))
{
    ly_lx_bit = 13;
matrix_ima(l_bound_y,l_bound_x) =
(unsigned short)(matrix_ima(l_bound_y,l_bound_x) ^ 4096);
bit_flag = 1;
/*printf("\nll=13");*/
}

if (((unsigned short)(matrix_ima(u_bound_y,u_bound_x) & 32768) != 0))
{
    uy_ux_bit = 16;
matrix_ima(u_bound_y,u_bound_x) =
(unsigned short)(matrix_ima(u_bound_y,u_bound_x) ^ 32768);
bit_flag = 1;
/*printf("\nuu=16");*/
}
if ((unsigned short)((matrix_ima(u_bound_y,u_bound_x) & 16384) != 0))
{
    uy_ux_bit = 15;
    matrix_ima(u_bound_y,u_bound_x) =
    (unsigned short)(matrix_ima(u_bound_y,u_bound_x) ^ 16384);
    bit_flag = 1;
    /*printf("nuu=15");*/
}

if ((unsigned short)((matrix_ima(u_bound_y,u_bound_x) & 8192) != 0))
{
    uy_ux_bit = 14;
    matrix_ima(u_bound_y,u_bound_x) =
    (unsigned short)(matrix_ima(u_bound_y,u_bound_x) ^ 8192);
    bit_flag = 1;
    /*printf("nuu=14");*/
}

if ((unsigned short)((matrix_ima(u_bound_y,u_bound_x) & 4096) != 0))
{
    uy_ux_bit = 13;
    matrix_ima(u_bound_y,u_bound_x) =
    (unsigned short)(matrix_ima(u_bound_y,u_bound_x) ^ 4096);
    bit_flag = 1;
    /*printf("nuu=13");*/
}

if ((unsigned short)((matrix_ima(u_bound_y,l_bound_x) & 32768) != 0))
{
    uy_lx_bit = 16;
    matrix_ima(u_bound_y,l_bound_x) =
    (unsigned short)(matrix_ima(u_bound_y,l_bound_x) ^ 32768);
    bit_flag = 1;
    /*printf("nul=16");*/
}

if ((unsigned short)((matrix_ima(u_bound_y,l_bound_x) & 16384) != 0))
{
    uy_lx_bit = 15;
    matrix_ima(u_bound_y,l_bound_x) =
    (unsigned short)(matrix_ima(u_bound_y,l_bound_x) ^ 16384);
    bit_flag = 1;
    /*printf("nul=15");*/
}

if ((unsigned short)((matrix_ima(u_bound_y,l_bound_x) & 8192) != 0))
{
    uy_lx_bit = 14;
    matrix_ima(u_bound_y,l_bound_x) =
    (unsigned short)(matrix_ima(u_bound_y,l_bound_x) ^ 8192);
    bit_flag = 1;
    /*printf("nul=14");*/
}

if ((unsigned short)((matrix_ima(u_bound_y,l_bound_x) & 4096) != 0))
{
    uy_lx_bit = 13;
    matrix_ima(u_bound_y,l_bound_x) =
    (unsigned short)(matrix_ima(u_bound_y,l_bound_x) ^ 4096);
    bit_flag = 1;
    /*printf("nul=13");*/
/*Interpolate in x direction first*/
grey_x1 = ((float)matrix_ima(l_bound_y,u_bound_x) -
    matrix_ima(l_bound_y,l_bound_x))*x_wt_fact +
    (float)matrix_ima(l_bound_y,l_bound_x);
grey_x2 = ((float)matrix_ima(u_bound_y,u_bound_x) -
    matrix_ima(u_bound_y,l_bound_x))*x_wt_fact +
    (float)matrix_ima(u_bound_y,l_bound_x);

/*Now interpolate in y direction*/
grey_fin = ((grey_x2 - grey_x1)*y_wt_fact) + grey_x1;

/*Condition to prevent rounding error in final grey scale value*/
if ((grey_fin - (int)grey_fin) >= 0.5)
    
    else matrix_cor(y,x) = (unsigned short)grey_fin;

    if (bit_flag == 1)
        
        /*printf("\nBit-flag set! x=%i, y=%oi",x,y);*/

        if (ly_ux_bit == 16) matrix_cor(y,x)=
            (unsigned short)(matrix_cor(y,x) | 32768);
        if (ly_ux_bit == 15) matrix_cor(y,x)=
            (unsigned short)(matrix_cor(y,x) | 16384);
        if (ly_ux_bit == 14) matrix_cor(y,x)=
            (unsigned short)(matrix_cor(y,x) | 8192);
        if (ly_ux_bit == 13) matrix_cor(y,x)=
            (unsigned short)(matrix_cor(y,x) | 4096);
        if (ly_lx_bit == 16) matrix_cor(y,x)=
            (unsigned short)(matrix_cor(y,x) | 32768);
        if (ly_lx_bit == 15) matrix_cor(y,x)=
            (unsigned short)(matrix_cor(y,x) | 16384);
        if (ly_lx_bit == 14) matrix_cor(y,x)=
            (unsigned short)(matrix_cor(y,x) | 8192);
        if (ly_lx_bit == 13) matrix_cor(y,x)=
            (unsigned short)(matrix_cor(y,x) | 4096);
        if (uy_ux_bit == 16) matrix_cor(y,x)=
            (unsigned short)(matrix_cor(y,x) | 32768);
        if (uy_ux_bit == 15) matrix_cor(y,x)=
            (unsigned short)(matrix_cor(y,x) | 16384);
        if (uy_ux_bit == 14) matrix_cor(y,x)=
            (unsigned short)(matrix_cor(y,x) | 8192);
        if (uy_ux_bit == 13) matrix_cor(y,x)=
            (unsigned short)(matrix_cor(y,x) | 4096);

        /*printf("\nmatrix_cor=%u",matrix_cor(y,x));*/
        pause();*/

        if (uy_lx_bit == 16) matrix_cor(y,x)=
            (unsigned short)(matrix_cor(y,x) | 32768);
        if (uy_lx_bit == 15) matrix_cor(y,x)=
            (unsigned short)(matrix_cor(y,x) | 16384);
        if (uy_lx_bit == 14) matrix_cor(y,x)=
            (unsigned short)(matrix_cor(y,x) | 8192);
        if (uy_lx_bit == 13) matrix_cor(y,x)=
            (unsigned short)(matrix_cor(y,x) | 4096);
```c
404         }
405     }
406     else matrix_cor(y,x) = 0;
407     /*printf("\nFinal Grey = %u",matrix_cor(y,x));*/
408     }
409     }
410     printf ("\nABOUT TO OUTPUT DATA TO FILE");
411     }
412     void output_mri_file()
413     {
414         for (y=0; y < y_dim; y++)
415             {
416                 for (x=0; x < x_dim; x++)
417                     {
418                         /*printf("\nmatrix_cor(%i,%i)=%u",y,x,matrix_cor(y,x));*/
419                     }
420                 }
421     /*Copy the header information from raw image file (infile1) to corrected
422     image file (outfile)*/
423     for (y=100; y < 101; y++)
424         {
425             for (x=95; x < 105; x++)
426                 {
427                     /*printf("\nmatrix_cor(%i,%i)=%u",y,x,matrix_cor(y,x));*/
428                 }
429             }
430     /*Read header into buffer from image input file*/
431     /*fseek(infile1,0,SEEK_SET);*/
432     printf("\nFile pointer at location %i before rewind",ftell(infile1));
433     rewind(infile1);
434     printf("\nFile pointer at location %i after rewind",ftell(infile1));
435     for (y=100; y < 101; y++)
436         {
437             for (x=95; x < 105; x++)
438                 {
439                     printf("\nmatrix_cor(%i,%i)=%u",y,x,matrix_cor(y,x));
440                 }
441             }
442     /*fread((void *)&buffer,sizeof(unsigned short),2048(infile1);*/
443     /*Loop Method of Read*/
444     for (x=0; x < 4096; x++)
445         {
446             fread((void *)&buffer[x],sizeof(char),1,infile1);
447         }
448     printf("\nFile pointer at location %i after fread",ftell(infile1));
449     for (y=100; y < 101; y++)
450         {
451             for (x=95; x < 105; x++)
207```
printf("\nmatrix-cor(x,y)=\n",x,matrix_cor(y,x));
}

/*Write buffer out to corrected image output file*/
for (x=0; x < 4096; x++)
{
    fwrite((void *)&buffer[x],sizeof(char),1,outfile);
}
/*printf("\nFile pointer at location %i after header fwrite",ftell(outfile));*/

/*Send file pointer to location 4096 bytes from beginning of file*/
/*fseek(outfile,4096L,0);*/
/*printf("\nFile pointer at location %i after fseek",ftell(outfile));*/

/*Write out the corrected image matrix to the file*/
/*fwrite((void *)&mmatrix,sizeof(unsigned short),65536,outfile);*/
for (y=100; y < 101; y++)
{
    for (x=95; x < 105; x++)
    {
        /*printf("\nmatrix-cor(x,y)=\n",x,matrix_cor(y,x));*/
    }
}
printf("\nABOUT TO DO FWRITE");
/*Try a loop method of fwrite instead*/
for (y=0; y < y_dim; y++)
{
    for (x=0; x < x_dim; x++)
    {
        /*printf("\nmatrix-cor(x,y)=\n",x,matrix_cor(y,x));*/
        fwrite((unsigned short *)&matrix_cor(y,x),sizeof(unsigned short),1,outfile);
    }
}
/*printf("\nFile pointer at location %i after pixel write",ftell(outfile));*/
fclose(infile1);
fclose(outfile);
Appendix H: Resampling Coordinate Interpolation Program

Note the distance weight factor referred to in the program header is calculated as described in section 4.4 of the text of this thesis.

```c
#include <stdio.h>
#include <stdlib.h>
#include <dos.h>
#include <alloc.h>
#include <math.h>

FILE *infile1;
FILE *infile2;
FILE *outfile;

char input1[20], input2[20], output[20];
unsigned scale_fact_sam1, scale_fact_sam2;
int xcoord1, xcoord2, ycoord1, ycoord2, xcoord, ycoord;
long x;
float dist_wt_fact;

main()
{
    /*User input of first resampling coord filename*/
    printf("Enter distortion correction coordinate file 1 --> ");
    scanf("%s", input1);

    /*Opens file for read and exits if not found*/
    if ((infile1 = fopen(input1, "rb")) == NULL)
    {
        printf("Could not open file %s", input1);
        exit(1);
    }

    /*User input of second resampling coord filename*/
    printf("Enter distortion correction coordinate file 2 --> ");
    scanf("%s", input2);

    /*Opens file for read and exits if not found*/
    if ((infile2 = fopen(input2, "rb")) == NULL)
    {
        printf("Could not open file %s", input2);
        exit(1);
    }

    /*Open output file*/
    printf("Enter name of output resampling coord. file --> ");
    scanf("%s", output);
    if((outfile = fopen(output, "wb")) == NULL)
    {
        printf("Could not open output file %s", output);
    }
...
```

W. Beckham
October 1994
exit(1);

/*Prompt user for distance weight factor*/
printf("Dist wt fact = (req'd_posn - file1_posn)/(file1_posn - file2_posn)\n");
printf("Wt Factor = 0 all weight to file1, Wt Factor = 1 all weight to file2\n");
scanf("%f", &dist_wt_fact);

/*Get scale factor from input files and compare*/
fwrite(&scale_fact_sam1,sizeof(int), 1,infile1);
fread(&scale_fact_sam2,sizeof(int), 1,infile2);
if(scale_fact_sam1 != scale_fact_sam2)
{
printf("The two files have different scale factors\n");
exit(1);
}

/*Write the scale factor to the output file*/
fwrite(&scale_fact_sam1,sizeof(int), 1,outfile);

for (x=0; x < 65536; x++)
{
    fread(&xcoord1,sizeof(int), 1,infile1);
fread(&ycoord1,sizeof(int), 1,infile1);
fread(&xcoord2,sizeof(int), 1,infile2);
fread(&ycoord2,sizeof(int), 1,infile2);
    xcoord = (int)(((float)(xcoord2 - xcoord1)*dist_wt_fact)+(float)xcoord1);
    ycoord = (int)(((float)(ycoord2 - ycoord1)*dist_wt_fact)+(float)ycoord1);

    /*write the coords to the output file*/
fwrite(&xcoord,sizeof(int), 1,outfile);
fwrite(&ycoord,sizeof(int), 1,outfile);
}

fclose(infile1);
fclose(infile2);
fclose(outfile);