2005

Efficient automatic correction and segmentation based 3D visualization of magnetic resonance images

Mikhail V. Milchenko
Louisiana State University and Agricultural and Mechanical College, mmiltc1@lsu.edu

Follow this and additional works at: https://digitalcommons.lsu.edu/gradschool_dissertations

Part of the Computer Sciences Commons

Recommended Citation
https://digitalcommons.lsu.edu/gradschool_dissertations/1461

This Dissertation is brought to you for free and open access by the Graduate School at LSU Digital Commons. It has been accepted for inclusion in LSU Doctoral Dissertations by an authorized graduate school editor of LSU Digital Commons. For more information, please contact gradetd@lsu.edu.
EFFICIENT AUTOMATIC CORRECTION AND SEGMENTATION BASED 3D VISUALIZATION OF MAGNETIC RESONANCE IMAGES

A Dissertation
Submitted to the Graduate Faculty of the Louisiana State University and Agricultural and Mechanical College in partial fulfillment of the requirements for the degree of Doctor of Philosophy

in

The Department of Computer Science

by
Mikhail V. Milchenko
Diploma Moscow State University, 1999
December 2005
Acknowledgements

I wish to thank my advisor, Dr. John M. Tyler, for his unceasing energetic support and priceless guidance of my research.

I wish to thank also Dr. Oleg S. Pianykh for his numerous and valuable scientific and technical advice, incredible responsiveness and support that helped to improve the quality of this research.

I wish to thank Dr. Warren Waggenspack, Jr. for the pleasure of working under his guidance in order to fulfill my minor requirement. A large part of what I know about 3D visualization and modeling I learned from him, and large part of the work that I performed under his guidance served the foundation for this research.

I wish to thank my committee, Dr. Bijaya B. Karki, Dr. Claire D. Advokat and Dr. Jianhua Chen, for their valuable comments and critiques.

I wish to thank all the professors and instructors of Department of Mechanics and Mathematics, Moscow State University, whose tireless educational effort and exceptional scientific merit allowed me to obtain the excellent mathematical background necessary for doctoral research.

I wish to thank my mother country, Russia, and its people who provided me with opportunity to receive a mathematical education of highest quality for free.

I wish to thank my parents for things we can receive only from parents.

And, I wish to thank my fiancée, my beloved Jennie, for being my endless source of inspiration and true knowledge about the world.
# Table of Contents

ACKNOWLEDGEMENTS ........................................................................................................... ii

ABSTRACT ............................................................................................................................... iv

INTRODUCTION ..................................................................................................................... 1

CHAPTER 1 MAGNETIC RESONANCE INHOMOGENEITY CORRECTION ............. 3
  1.1 Magnetic Resonance Image Acquisition Process ................................................. 3
  1.2 Inhomogeneity in MR Images ............................................................................ 6
  1.3 Non-uniformity Artifact Model ......................................................................... 8
  1.4 MR Inhomogeneity Correction Methods .......................................................... 9
  1.5 MR Non-uniformity Correction: Why Another Method? ............................... 18
  1.6 New Method Derivation ................................................................................... 19
  1.7 The Algorithm .................................................................................................... 27
  1.8 Basic Evaluation of \( Dsf \) ................................................................................ 50
  1.9 Comparison with Selected Published Methods ................................................ 53
  1.10 Discussion and Conclusions .......................................................................... 65

CHAPTER 2 AUTOMATED MEDICAL IMAGE VOLUME RENDERING ............ 68
  2.1 Introduction ........................................................................................................... 68
  2.2 Medical Image Volumes .................................................................................... 69
  2.3 Surface Rendering ............................................................................................. 70
  2.4 Surface Rendering and Isosurface Representation .......................................... 71
  2.5 Volume Rendering ............................................................................................. 73
  2.6 Rendering Based on Boundary Voxels of a Segmented Volume .................... 75
  2.7 The Problem of Efficient Rendering ............................................................... 76
  2.8 Pre-segmentation of a Medical Image Volume ................................................ 77
  2.9 Gravitational Shading Algorithm ...................................................................... 86
  2.10 Evaluation of \( Gs \) ............................................................................................ 102
  2.11 Discussion and Conclusions .......................................................................... 108

CHAPTER 3 SUMMARY ...................................................................................................... 111
  3.1 Results ................................................................................................................ 111
  3.2 Conclusions ....................................................................................................... 113
  3.3 Future Research ................................................................................................. 114

REFERENCES ....................................................................................................................... 116

APPENDIX: \( DSF \) ALGORITHM PSEUDOCODE ......................................................... 120

VITA ..................................................................................................................................... 126
Abstract

In the recent years, the demand for automated processing techniques for digital medical image volumes has increased substantially. Existing algorithms, however, still often require manual interaction, and newly developed automated techniques are often intended for a narrow segment of processing needs.

The goal of this research was to develop algorithms suitable for fast and effective correction and advanced visualization of digital MR image volumes with minimal human operator interaction. This research has resulted in a number of techniques for automated processing of MR image volumes, including a novel MR inhomogeneity correction algorithm derivative surface fitting (dsf), automatic tissue detection algorithm (atd), and a new fast technique for interactive 3D visualization of segmented volumes called gravitational shading (gs).

These newly developed algorithms provided a foundation for the automated MR processing pipeline incorporated into the UniViewer medical imaging software developed in our group and available to the public. This allowed the extensive testing and evaluation of the proposed techniques.

Dsf was compared with two previously published methods on 17 digital image volumes. Dsf demonstrated faster correction speeds and uniform image quality improvement in this comparison. Dsf was the only algorithm that did not remove anatomic detail. Gs was compared with the previously published algorithm fsvr and produced rendering quality improvement while preserving real-time frame-rates.

These results show that the automated pipeline design principles used in this dissertation provide necessary tools for development of a fast and effective system for the automated correction and visualization of digital MR image volumes.
Introduction

In the last two decades, digital imaging has been replacing conventional film in hospitals and other medical institutions. Digital is becoming a de facto standard for medical image storage and communication. Currently, all modern medical image acquisition devices produce a digital format for this output. Consequently, the dissemination of these digital images has increased the demand for the computer-based processing and visualization of digital image volumes in medicine, and many new digital processing algorithms emerged. The earlier processing methods were designed to help clinicians manage medical imaging data in a new format and transfer established analysis into this new environment. While the elements of automatic computer-based processing provided new capabilities, the original “medical” methods took a very conservative approach to the data handling, which required a larger than necessary substantial manual labor component.

The constantly fast evolution of computer technology has produced enormous volumes of digital images, i.e. almost all radiology clinics process images on the order of thousands daily. The current trend is to increase the image throughput with the same number of radiologists, increasing their “efficiency”. To obtain this increased image throughput requires many new automated processing methods. For this reason, many recent medical image processing and visualization methods are more automated and tend to reduce all the required human operator interactions.

Our research continues this trend. We concentrated our effort on magnetic resonance acquisitions, which are non-invasive and possess good spatial resolution and soft tissue contrast. For these reasons, magnetic resonance is widely used in the diagnosis of many diseases, e.g. multiple sclerosis, atrophies, infarction, tumors, visual/hearing disturbances, traumas, etc.
We addressed two major problems in the acquisition and analysis of MR images: inhomogeneity artifact correction and visualization of internal structures. Our goal was to develop a system of data correction and subsequent visualization that requires minimal operator interaction in the processing for high volumes of images. Additionally, such a system should use inexpensive generally available (PC) hardware and require no additional third-party software libraries to encourage a broader range of possible applications. Finally, processing times must require only few seconds to be of use in the real-time clinical applications.

Based on these requirements, we developed a new novel real time digital MR image volume processing system with two major modules:

1. Automatic inhomogeneity correction presented in Chapter 1;
Chapter 1 Magnetic Resonance Inhomogeneity Correction

In the first half of this chapter, we discuss the magnetic resonance image acquisition process, artifacts associated with it and methods developed for correction of resulting medical images. In the second half, we will present a novel method for automatic MR inhomogeneity correction, details of our implementation and results of our method’s performance evaluation.

1.1 Magnetic Resonance Image Acquisition Process

Any nucleus with odd atomic number, i.e. odd number of neutrons and protons, has a non-zero spin called its magnetic moment. In a normal state, all nuclei in the tissue are randomly oriented, and their net magnetization is zero. In the strong external magnetic field $B_0$, however, nuclei start “precessing” about an axis parallel to the direction of that field vector, and the tissue becomes “magnetized” (see Figure 1.1). The main tissue magnetization characteristics are net magnetization vector $M_0$ aligned along $B_0$ with the precession frequency $\omega_0$. Precession frequency depends upon the particular tissue type and serves for the subsequent distinction between different tissues. Precession frequency is determined from the Larmor equation:

$$\omega_0 = \frac{\gamma B_0}{2\pi}$$

(1)

where $\gamma$ is a scalar constant. Since the energy levels of “magnetized” nuclei differ from the non-excited state, excited nuclei can interact with the external electromagnetic pulses (refer to [1] for further details on underlying physics).

The magnetic resonance experiment involves the application of radiofrequency spectrum pulses (RF pulses) to the volume to be imaged. After the application of an electromagnetic pulse, nuclei gain additional “magnetization” if the RF pulse contains the frequency close to their own (resonant) frequency, and the net magnetization vector $M_0$
moves to a transverse (xy) plane. During the subsequent lapsed time, nuclei re-emit at the same frequency, and $M_0$ returns to its original orientation. This process is called free induction decay (FID) and is illustrated on Figure 1.2.

**Figure 1.1.** Magnetized nuclei spin vector components. $B_0$ is an external constant magnetic field, $\omega_0$ is the precession frequency.

If a loop of wire (the receiver) is placed in the transverse plane (the plane perpendicular to the magnetic field and xy plane on Figure 1.1), an alternating electric current is induced in the receiver. This current is called the MR signal and is characterized by amplitude, frequency and phase relative to the phase of the transmitter.

T1 is the time required for the z component of $M$ to recover 63% of its original magnitude, and T2 is the time required for the transverse (xy) component of $M$ to decay to 37% of its value immediately after the RF pulse. T1 and T2 (called FID relaxation times) are often used as a measure of resulting pixel intensity on images produced using T1 or T2 weighting.
Figure 1.2. Free induction decay. (a) Net magnetization response to an RF pulse; (b) its Fourier transform. \( \omega_{NQ} \) is Nyquist frequency, \( \omega_{NQ} = (\text{Total number of data points}) / [2^*(\text{Sampling time})] \), \( \omega_{TR} \) is the transmitter reference frequency.

To obtain MR images for a three-dimensional volume, the MR signal needs to be localized for every sample point in the volume. For this, small perturbations are applied to the main magnetic field \( B_0 \) in short time intervals. These perturbations are called gradient pulses and depend linearly upon the x, y and z coordinates of the magnetic field. So it is possible to “decode” the point position in space from the resulting signal described by the expanded form of Larmor equation (1):

\[
\omega_i = \gamma \left( B_0 + \vec{V} \times \vec{r}_i \right),
\]

where \( \omega_i \) is the proton frequency at position \( \vec{r}_i \) and \( \vec{V} \) is the gradient vector.

The combination of gradient pulses, RF pulses and data sampling is called a pulse sequence. Several techniques characterized by specific pulse sequences for the RF and
magnetic gradient field arrangements are used to obtain a good spatial resolution within a reasonable time scan. These techniques have the following characteristics:

1. **2D multislice imaging** utilizes the excitation of several 2D slices in the same time scan;
2. **Sequential slice technique** implements a slice-by-slice sequential excitation;
3. **3D volume acquisition** uses double-phase encoding for one-time excitation of small volumes;
4. **Half-acquisition/half-Fourier technique** “takes advantage of the intrinsic symmetry of the raw data to reduce the scan time [1].”

### 1.2 Inhomogeneity in MR Images

Magnetic resonance imaging is based on the resonant frequencies and relaxation times for different tissues being different enough to produce contrast images. The underlying physical theory suggests that within one uniform tissue, the MR signal would have only insignificant deviation from the mean value that characterizes this tissue. The only nonuniformity that the pixel intensities can have naturally, is due to the tissue microstructure ([2], [3]), although other authors ([4], [5]) assume the “ideal” pixel intensity variation within a single tissue to be zero.

This model implies the ideal conditions for a MR experiment: uniform external magnetic field, absence of random noise, no correlation between the signals obtained from neighboring sample points, etc. The actual MR scanners produce contrast but non-uniform images. MR artifacts are classified by Brown et al. [1] into three groups according to the cause of signal misinterpretation:

1. Artifacts caused by patient motion during acquisition;
2. Artifacts due to measurement technique parameters;
3. Artifacts generated by scanner or the source external to both patient and scanner.
This classification implies that the source MR signal can be interpreted to produce correct depictions of anatomic detail in all 3 cases. In this work, we are more interested in the artifacts that can be corrected after image acquisition (i.e. when they do not remove some anatomical information from the image). Therefore, we limit our consideration to artifacts, of which the most important are the following:

1. Stationary gradients, where the main magnetic field \( B \) nonuniformity affects the characteristics of received signal. According to [1], imperfections in the magnet from manufacturing, as well as the presence of metal objects in the vicinity of MR scanner that distort the magnetic field. The RF receiver coil may also be a source for smooth signal variation [3]. This inhomogeneity is constant during the acquisition time and results in continuous gradients in soft tissues. The amount of stationary non-uniformity depends upon magnetic field characteristics and can sometimes be significant (as can be seen on the linear profile of the soft tissue on Figure 1.3).

2. Differences in the magnetization in adjacent tissues may introduce a distortion into local magnetic field near the interface between the tissues.

3. Since the technique used for localization of received signal implements imaging gradients, the local temporary magnetic field inhomogeneity induces proton dephasing. This may result in repeated intensity fluctuations within the same tissue with a noticeable structure. Figure 1.4 shows a magnified area of white matter with pixel contrast enhanced to observe the inhomogeneity microstructures.

The stationary gradient is the major artifact that interferes with both human and automatic processing of MR data, and it does not necessarily remove anatomical information from the resulting signal. For this reason, it is theoretically possible to apply a correction procedure and reduce MR image non-uniformity. In further sections, only stationary gradient non-uniformity is discussed, and is called the MR inhomogeneity or non-uniformity artifact.
In the next section, we will consider more closely the structure of this artifact.

**Figure 1.3.** Inhomogeneity within a tissue. Position of the profiling line within the same tissue on the MR image (left), intensity along this profile (right)

**Figure 1.4.** Contrast optimized local inhomogeneity induced by the gradient-driven eddy currents

### 1.3 Non-uniformity Artifact Model

In many images, MR inhomogeneity gradients are not visible. This is due to the eye’s high capacity for accommodation, as well as the fact that our brain often adapts the distorted image from the optics of an eye, correcting it toward more “interpretable” results. For this
reason, it is hard to design an automatic correction method without describing the
inhomogeneity artifact by analytical model first.

It follows from (2) that MR bias field may be considered three-dimensional for 3D
image volumes. For simplicity, the derivation of mathematical foundation for the correction
methods will be performed for a two-dimensional case; it is implied that it can be expanded
to 3D by addition of a third coordinate and appropriate change in notation.

It follows from the MR acquisition process that “the observed image is the product of
the spin density distribution in the tissue (…) and sensitivity profile of the surface coil [6].”
The multiplicative nature of MR non-uniformity can be described for two-dimensional image
by the following equation:

\[ I(x, y) = f(x, y)I'(x, y) + n(x, y) , \quad (3) \]

where \( I \) represents actually obtained two-dimensional image, \( I' \) the non-distorted “ideal”
image, \( f(x, y) \) a multiplicative bias field and \( n(x, y) \) an additive noise. The latter is present in
almost any MR image and should be taken into account before any transformation of model
(3) since it can be the source of significant computation error as discussed in Section 1.6.1.
The bias field model (3) is the foundation of a majority of correction methods discussed in
the next section.

1.4 MR Inhomogeneity Correction Methods

This section contains a general review of existing MR correction techniques. Since the
existing techniques are numerous and diverse, any classification would be formal and
incomplete. We do not intend to form our classifications based on any specific formal rule
since we believe it is more informative to view existing correction methods as having natural
trends that can be combined by a certain quality. Therefore, some methods mentioned below
fall into more than one category.
1.4.1 Phantom Methods

The earlier methods for correction of MR non-uniformity emerged in the early 1980’s, when commercial MR scanners spread. Brown and Semelka et al. [1] mention that increasing sampling frequency bandwidth of the MR scanner leads to a reduction of non-uniformity in observed images, but this also leads to a lower signal-to-noise ratio and the loss of anatomic information [7]. The more efficient early methods use a phantom image to compute \( f(x, y) \) in (3). Phantoms are simple objects filled with a uniform substance which can be imaged using MR. After acquisition of phantom image, the bias field \( f(x, y) \) for MR scanner \( S \) can be extracted using (3). An example of this procedure is described in [7]. One of the difficulties associated with this method is the difficulty determining a phantom that exactly fits the scanner’s three-dimensional field of view and separating the object from its background when necessary. Due to non-uniformity in the phantom image, manual processing may be required (see Figure 1.5 for illustration of a phantom image with ambiguous edge definition).

Furthermore, it is not always possible to access the actual scanner where the images were acquired. Since the development of broadband computer networks, many clinicians and researches view medical data remotely, in which case the application of a phantom-based method may not even be possible. This lack of universal applicability and high manual labor requirements led to development of other correction techniques that do not require a preliminary MR experiment to estimate the bias field \( f(x, y) \). These techniques are often called post-acquisition.

1.4.2 Registration Based Methods

The problem of bias estimation is closely related to segmentation of MR data. We have already shown that the estimation of bias field in phantom images requires
Figure 1.5. Distorted image of cylindrical phantom. Notice the loss of edge definition at the top.

segmentation into object and background. A similar approach used on brain images is based on registration. The task of registration is to find the transformation between the original brain image volume and a known model volume for which the bias field and tissue distributions are established. After the registration transformation operator $\Theta$ is found, applying its inverse $\Theta^{-1}$ to original MR image and comparing this result with the model volume allows computing the bias field based on (3). The model image for registration can come from different sources. For instance, Lai and Fang [8] suggest using an additional low resolution image acquired from the same spatial position as the original image (this approach resembles phantom methods). The uniformity of this small additional image (acquired using a body-coil instead of a surface-coil pulse sequence, Lai and Fang [8]), allows the estimate of the bias field $f(x, y)$ of original image after registration. Many other authors as Studholme et al. [9], Christensen et al. [10] and Collins et al. [11] use reference MRI intensity templates for registration. The principle difficulty with registration based methods is to determine the operator $\Theta$. Most use an iterative approach for this and therefore may require an extended
time to complete. The final error in $\Theta$ depends upon the correctness of the registration. Significant non-uniformity can interfere with the registration, so these methods work better with a relatively small bias field. Additionally, constructing a tissue model requires substantial preliminary work that may or may not be reproducible. Finally, a registration based method’s applicability is limited to a specific body part.

1.4.3 Statistical Methods

From a statistical point of view, the MR image can be considered a mixture of several probability distributions; in this case $f(x, y)$ is considered to be a probability density function. In the statistical view, the segmentation problem consists of finding the unknown soft tissue distribution of the “ideal” image. Then a subsequent analysis of (3) would calculate $f(x, y)$.

Many statistical methods develop two expressions $E_1$ and $E_2$: $E_1$ for estimating the bias field $f(x, y)$ and $E_2$ to compute a tissue distribution map. The calculation of either $E_1$ or $E_2$ depends on the other. $E_1$ and $E_2$ are estimated iteratively using the Expectation-Maximization (EM) algorithm described originally in Dempster et al. [12]. Various authors derived the estimates $E_1$ and $E_2$ using Bayesian statistics [2], [13], Markov process theory [14], deconvolution filter based on Fourier transform [15] and other techniques. The difficulty with the correction methods based on the EM algorithm is that $E_1$ and $E_2$ must be initialized for the first iteration. Therefore, some prior knowledge must be available about the tissue distribution in the original image. The resulting error with the statistical method based on EM depends upon the correctness of the prior distribution. It is easier to model soft tissues, but any irregular anatomy such as edges and/or fine detail does not fit well in statistical models. To exclude such areas in the MR image, Guillemaud et al. [2] introduce an additional tissue class called ‘other’, and all irregular anatomy is assumed to belong to ‘other’.

Depending on the design of $E_1$ and $E_2$, the local minimum using EM may not be a good approximation of the bias field and tissue distribution (see our results with the method
described in [16] later in this chapter). To achieve smoothness for the estimated bias field $f(x, y)$, blurring in $E_1$ or $E_2$ is used between EM iterations; the bias field determined by EM-based statistical methods often looks like a fuzzy original image (see Figure 1.6). This result suggests a computation error and may remove some anatomical information after the correction has been applied.

Wells et al. [13] and other authors have reported good correction results; however, it is hard to estimate the performance of many statistical methods, since the published results are often based upon unique training data and sometimes achieved after many algorithm parameter adjustments, e.g., Wells et al. [13] mention two years of training data analysis. The statistical methods required tissue distribution models, usually developed for one specific body part. The statistical methods are usually applied to brain images.

Figure 1.6. Correction of a phantom brain image using Wells et al. [13]. Uniform image (top left), biased image (top right), its bias field $f(x,y)$ found by Wells algorithm (bottom left) and correction (bottom right).

1.4.4 Histogram Analysis Methods

Large areas of uniform tissue in MR images correspond to histogram peaks (see Figure 1.7). This property is widely used in intensity-based segmentation of MR images to
determine the number of tissue classes, their means and other distribution characteristics. Histograms of uniform MR images usually have well-defined peaks which are easy to separate.

The non-uniformity introduces additional irregularities into the histogram: the peaks in such histograms are lower and tissue distributions may significantly overlap, posing an additional difficulty for subsequent separation (see Figure 1.8 an example of a MR brain image with an irregular histogram).

![Image](image_url)

**Figure 1.7. Effect of non-uniformity on image histogram.** Top row: non-uniform phantom image of the brain with 40% non-uniformity (left), its multiplicative bias (right); bottom row: histogram of biased image (left) and original uniform image (right).

The MR non-uniformity correction method should improve the image histogram. This is widely used by the methods that employ histogram analysis. These methods form a separate class of MR artifact correction techniques and their analysis can include a combination of local and global histograms. For instance, Brinkmann et al. [17] compare local and global histogram mean and median ratios; Christensen [18] uses histogram...
derivative analysis; DeCarli et al. [19] utilize an intensity-based segmentation using a global histogram and then estimate the bias field $f(x, y)$ using local histogram analysis.

Figure 1.8. A case where histogram peak identification is difficult. MR sagittal brain image (left) and its histogram (right).

Histogram based algorithms normally need only a few image related computations to complete and therefore are fast; they do not require any training data. However, the following reasoning may lead to questions about their direct applicability to bias correction. An image histogram represents the distribution of different intensity levels in an image. It is a mapping of two-dimensional (or three-dimensional with appropriate change in notation) images $I(x, y)$ to a one-dimensional histogram graph $H(i)$, where

$$H(i) = \sum_{I(x,y)=i} 1$$

and $i$ ranges between 0 and $\max_{x,y}[I(x,y)]$. Equation (4) illustrates that the transformation $H(i)$ is degenerate and an inverse transformation does not exist. This means that the histogram does not uniquely identify the image. A purely histogram based approach reconstructs a certain global image characteristic (bias field) from the local characteristic (histogram). Since the histogram transformation is degenerate, some information for the reconstruction of its global characteristic may be missing. For example, smoothness or even piecewise continuity of a bias field is not guaranteed; therefore, additional assumptions are almost always used. Histogram-based methods should require more empirical adjustments than the other methods.
being discussed. To summarize, histogram based methods are fast, simple to implement, but their accuracy can be insufficient.

1.4.5 Reintegration Methods

The image gradient is another important characteristic from which the bias field estimation can be obtained. Image gradient analysis methods assume that in the areas where the tissue is uniform the gradient vector \( \nabla I = \left( \frac{\partial}{\partial x}, \frac{\partial}{\partial y} \right) \) at each point \((x, y)\) is roughly equal to the gradient vector of a bias field:

\[
D^{i.i} I(x, y) \approx D^{i.i} f(x, y).
\]  

Formally, it follows from (3) that the bias field can be obtained by an application based on the re-integration of the resulting gradient field. Assumption (5) and (3) alone are not sufficient for the bias field estimation, e.g. the additive noise and small anatomical detail would introduce an unacceptable error. The models derived with reintegration methods are designed to suppress the noise and exclude the “bad” areas of an image from consideration. Then the application of a selected bias field reconstruction is performed. Vokurka et al. [5], for instance, designed a special filter to be applied during the gradient field computation. They report that the resulting gradient field is smooth and regular enough to allow the bias field re-integration. However, the original paper, does not contain sufficient experimental data (only three datasets were analyzed) to determine the efficiency of this approach and since re-integration is associated with high computation error, extended experiments are needed to show the stability of a particular re-integration. In another re-integration method Lai and Fang [8] minimize the error by performing re-integration on a finite element grid, so that the error can be minimized on each finite element separately.
1.4.6 Surface Fitting Methods

All the previously covered correction methods are based on one particular foundation for the specific design of the correction. There are numerous methods that may use different models but are very similar in one formal characteristic; they model the bias field as a smooth and slow varying function that can be approximated with a finite basis. This approach is the defining characteristic of surface fitting; many of the previously mentioned techniques belong in this category. In general, surface fitting methods approximate the bias field $f(x, y)$ with a finite basis:

$$f_b(x, y) = \sum_{i=1}^{N} c_i B_i(x, y),$$

where $f_b(x, y)$ is the bias approximation, $B_i(x, y)$ are basis functions and $c_i$ are the resultant basis coefficients. Using (6), the problem of finding the bias field is to determine the coefficients $c_i$. This has the obvious advantages, i.e. instead of searching for the bias field value at every point $(x, y)$ (e.g. done by statistical methods Wells et al. [13]), only a few coefficients need to be found. The resultant smoothness is obtained automatically from the properties of basis functions.

Surface fitting can employ any of the techniques mentioned, and contain features of their different algorithms. For example, some methods ([20]-[22]) construct an error functional using the bias model (6) and an iterative minimization method to determine the basis coefficients. The newer fuzzy clustering methods ([16], [23], and [24]) use a fuzzy-set approach with statistical methods plus the EM algorithm to determine the basis coefficients. Surface fitting methods commonly use an iterative minimization algorithm, where the number of iterations is not always known in advance. For example, the spm method [25] (Section 1.9.7) required from 5 to 90 iterations on different datasets, sometimes running for as long as twenty minutes.
1.5 MR Non-uniformity Correction: Why Another Method?

In this research, a new method for correction of MR non-uniformity is developed. In Section 1.4, several methods are presented to solve this problem. What is our rationale for creating another method?

To answer this question, consider the complex structure of the various MR artifacts described in Sections 1.2 and 1.3 which suggest that any efficient correction method must provide simplification where its bias field model only approximates the real bias field. The applicability of any correction method depends upon its model properties and the scope of this model. In a correction method design, a compromise between the generality and accuracy of the method’s performance on actual images of immediate importance must be made. The current trend is to apply MR correction methods to brain images in a pre-processing step to do segmentation, volume calculation and 3D rendering. A majority of the publications referenced addressed the correction of or verified the performance of their proposed method with brain images. In addition, recent studies show that even the most elaborate methods cannot entirely remove MR non-uniformity: “none of the algorithms that we evaluated performed ideally under all circumstances,” Arnold et al. [26] comparing six bias correction algorithms. Thus it is clear that a general method that performs equally well on a broad range of MR images would be of interest.

Established correction methods commonly perform several iterations on their input. As the size of data increases, their resultant execution time increases several times, on the order of the cube of the linear image dimension. Clinicians demand fast execution times for almost all image processing, and any undefined time is undesirable. For these reasons, a method with a fixed short execution time would be preferable for use in clinical environments.
To summarize, we consider a correction method generality, robustness and short processing time to be an optimal combination for clinical and research environments with the usual demand from MR image processing. In the sections that follow, we will present our new correction method named derivative surface fitting (dsf) with these properties and evaluate it on volumes of MR image data.

1.6 New Method Derivation

This section contains the mathematical foundation of our proposed correction algorithm. To justify our decisions, we will provide a minimal mathematical background and prove several useful facts and relations.

1.6.1 Non-uniformity Model Used in Dsf

Although the multiplicative MR bias model described (3) is the foundation for the majority of existing correction methods, it has never been used in this form for computation. The reason is obvious: any method that determines \( f(x, y) \) from the product \( I(x, y) \) \( f(x, y) \) from (3) is very likely to perform numerous divisions, which are not computationally efficient. The log-transformation is routinely performed to convert the multiplicative form into an additive form:

\[
\log[I(x, y)] = \log[f(x, y)I'(x, y) + n(x, y)].
\]  

(7)

Various authors ([8], [13]) often declare that the noise term is small enough to be neglected. This is not obvious from (7), and a more precise estimation of resulting error is necessary. Let’s denote \( g_0 = n(x, y) \) and \( g = f(x, y)I'(x, y) \).

**Lemma 1.1.** Let \( g, g_0 > 0 \). For natural logarithm, the inequality
\[ \log g \leq \log(g_0 + g) \leq \log g_0 + \log g \]  
holds when

\[ 2 < g_0 \leq g. \]  

**Proof:** Suppose \( g_0 \) is fixed. Consider the function

\[ G(g) = \log g_0 + \log g - \log(g_0 + g). \]

We have

\[ G'(g) = \frac{1}{g} - \frac{1}{g_0 + g} = \frac{g_0}{g(g_0 + g)}, \]

so it increases by \( g \). \( G(g) = 0 \) only when \( g = \frac{g_0}{g_0 - 1} \) and therefore \( G(g) \geq 0 \) when \( g = \frac{g_0}{g_0 - 1} \).

The function \( \frac{g_0}{g_0 - 1} \) decreases by \( g_0 \) when \( g_0 > 1, g_0 \rightarrow +\infty \). Therefore, \( G(g) \geq 0 \) when

\[ 1 < g_0 \leq g \]  
(see Figure 1.9), and from increasing of the log function follows (8).

---

**Figure 1.9.** Area of safe log-transform. The graph of \( g = \frac{g_0}{g_0 - 1} \) illustrates the proof of Lemma 1.1.
Denoting: \( I_{\log}(x, y) = \log I(x, y) \), \( f_{\log}(x, y) = \log f(x, y) \), \( I'_{\log}(x, y) = \log I(x, y) \) and \( n_{\log}(x, y) = \log n(x, y) \), from (7) and (8) it follows that
\[
\left| I_{\log}(x, y) - f_{\log}(x, y) - I'_{\log}(x, y) \right| \leq n_{\log}(x, y),
\]
when the inequality (9) is met. We will use this important relation to improve the robustness of our model.

To reduce \( n(x, y) \) and therefore the error defined by (10), an edge preserving smoothing filter similar to those described in [27], [28] will be used. With \( n_{\log}(x, y) \equiv 0 \) (7) becomes
\[
I_{\log}(x, y) = f_{\log}(x, y) + I'_{\log}(x, y).
\]
After the multiplicative bias model has been converted into an additive one, we can better design a numerical method to estimate \( f_{\log} \).

### 1.6.2 MR Image and Bias Field Modeling

As shown in Sections 1.2 and 1.3, the bias field \( f_{\log}(x, y) \) may be considered smooth and slow varying based on the nature of actual MR artifact. \( f_{\log}(x, y) \) can be approximated by a finite set of basis functions \( B_i(x, y) \) (6):
\[
f_{\log}^a(x, y) = \sum_{i=1}^{N} c_i B_i(x, y).
\]
Since we are approximating a smooth function, smooth basis elements \( B_i(x, y) \) such as polynomials should be used.

Since homogeneous tissues are represented by pixels of similar intensity, we model an unbiased MR image \( I(x, y) \) as a piecewise constant function, where areas of minimal intensity variation correspond to a single tissue. A regular MR image contains some of the following: representations of large organs, vessels, bones, and smaller organs. Typically, large organs and background occupy most of the image space, and since they represent uniform objects
from MR imaging point of view, we can assume that the areas of low intensity variation prevail in a regular MR image.

The areas of an image with fine structures or edges need to be excluded from the model to minimize the computation error. This can be done by introducing a pixel weight function \( w(x, y) \) which determines the influence of local image characteristics of each pixel on the final result. The detailed substantiation of our choice of \( w(x, y) \) for use in dsf is provided in Section 1.7.5.

1.6.3 Computation of Basis Coefficients

To determine \( f_{\log} \) from (11), it is necessary that we extract an ideal image \( I'_{\log} \) first. For example, Brechbühler et al. [21] evaluates the difference between \( I_{\log} \) and pre-defined tissue intensities for this purpose. In our view, an original image may be too irregular and such an operation would inevitably introduce additional error, and smoothing the original image suggested in [30] would lead to a loss in edge definition and an increased error in the areas of irregular anatomy. Our approach is based on the property of \( I_{\log} \) illustrated in previous sections to be piecewise constant for the most of the image.

The only characteristic of interest for detection of a slow varying bias field is the low frequency variation throughout the image. The partial derivative operator applied to an image produces a gradient field in the derivative direction and can be used as a natural measure of slow variation. We can apply a mixed partial derivative operator

\[
D_\alpha = \{ \frac{\partial^i \partial^j}{\partial x^i \partial y^j} \}; \quad \alpha = (i, j); \quad i + j > 0; \quad i, j = 0, 1, \ldots, K
\]

\( D_\alpha \) to both the image and the bias field modeled by a polynomial bias. A partial derivative of a constant is zero, and \( D_\alpha (I_{\log}) \) can be considered zero everywhere except the tissue boundaries and fine anatomical structures removed from calculations by the use of the weight
function $w^\alpha(x, y)$. Applying $D_\alpha$ to both sides of (11) and omitting $(x, y)$ arguments for simplicity, we obtain using (12):

$$D_\alpha (I_{\log}) = I_{\log}^\alpha = f_{\log}^{B\alpha} + D_\alpha (I_{\log}') \equiv f_{\log}^{B\alpha} = \sum_{i=1}^N c_i B_i^\alpha,$$

where $f_{\log}^{B\alpha} = D_\alpha (f_{\log}^\alpha)$ and $B_i^\alpha = D_\alpha (B_i)$. The set of functions $\{B_i^\alpha, \ i = 1, \ldots, N\}$, however, may contain linearly dependent elements and no longer be suitable for approximation.

**Lemma 1.2.** Let real differentiable functions $f_1(x), \ldots, f_N(x)$ be linearly independent, with at least one $f_j \equiv const \neq 0$. If $\exists (c_1, \ldots, c_N), \exists k, c_k \neq 0$, so that

$$\sum_{i=1}^N c_i f_i'(x) \equiv 0,$$

then $f_k \equiv const$.

**Proof:** First, it should be noticed that only one constant can be contained in a linearly independent set of functions. Taking the indefinite integral from both parts of (14), we obtain

$$\int \sum_{i=1}^N c_i f_i'(x) \equiv 0 dx \Rightarrow \sum_{i=1}^N c_i \int f_i'(x) \equiv C_1.$$

From the definition of indefinite integral it follows that

$$\sum_{i=1}^N c_i f_i(x) + C_2 = C_1 \Rightarrow \sum_{i=1}^N c_i f_i(x) = C_1 - C_2 = C.$$

Suppose $f_j \equiv const \neq f_k$. Let $\tilde{c}_j = c_j - \frac{C}{f_j}$. Using (15), we have

$$\sum_{i \neq j} c_i f_i(x) + \tilde{c}_j f_j(x) = \sum_{i \neq j} c_i f_i(x) + c_j f_j(x) - C = 0.$$

Since $c_k \neq 0$ and by our supposition $j \neq k$, the left hand side of (16) is a non-trivial linear combination of $f_1(x), \ldots, f_N(x)$ and we have a contradiction.

It follows from Lemma 1.2 that removing constants from the set of basis functions guarantees the linear independence of their derivatives. After necessary index changes, we
can assume without limiting the generality the linear independence of $\{B_i^\alpha, \ i = 1, \cdots, M\}$, $M = N$ or $M = N - 1$.

The problem of finding the bias field can now be reformulated as finding the best approximation $\sum_{i=1}^{M} c_i B_i^\alpha$ of a known function $f = I_\log$ using the basis $\{B_i^\alpha, \ i = 1, \cdots, M\}$. Suppose $f$ belongs to the normal linear space. Finding the best approximation means that we need to find an element $\sum_{i=1}^{M} c_i^0 B_i^\alpha$ such that

$$\left\| f - \sum_{i=1}^{M} c_i^0 B_i^\alpha \right\| = \inf_{c_1, \cdots, c_M} \left\| f - \sum_{i=1}^{M} c_i B_i^\alpha \right\|.$$  \hspace{1cm} (17)

If such an element exists, it is called the element of the best approximation. Bakhvalov et al. [29] shows that the element of the best approximation exists. To find it in our case, it is convenient to consider the norm and scalar product

$$\|f\|_2 = \left( \iint_{\Omega} f(x, y)^2 q(x, y) dx dy \right)^{1/2},$$

$$(f, g) = \iint_{\Omega} f(x, y) g(x, y) q(x, y) dx dy,$$  \hspace{1cm} (18)

where $q(x, y) \geq 0$ and $\Omega = [0 \cdots w] \times [0 \cdots h]$, $w$ and $h$ are the linear dimensions of an image.

The norm $\|f\|_2$ defines a Hilbert space, for which the element of best approximation is unique. The proof of this can also be found in [29].

From (17), coefficients $\{c_1, \cdots, c_M\}$ of the element of best approximation provide a minimum for the expression

$$\Phi(c_1, \cdots, c_M) = \left\| f - \sum_{i=1}^{M} c_i B_i^\alpha \right\|^2 = \left( f - \sum_{i=1}^{M} c_i B_i^\alpha, f - \sum_{i=1}^{M} c_i B_i^\alpha \right).$$

This expression reaches its minimum when conditions $\frac{\partial \Phi}{\partial c_k} = 0$ are satisfied. We have
\[ \frac{\partial \Phi}{\partial c_k} = \left( -B_k^\alpha, f - \sum_{i=1}^M c_i B_i^\alpha \right) + \left( f - \sum_{i=1}^M c_i B_i^\alpha, -B_k^\alpha \right) = -2 \left( f - \sum_{i=1}^M c_i B_i^\alpha, B_k^\alpha \right) = 0. \]

From this we obtain the system of linear equations with unknowns \( \{c_1, \ldots, c_M\} \) that correspond to local minimum of \( \Phi \):

\[ \sum_{i=1}^M c_i (B_i^\alpha, B_k^\alpha) = (f, B_k^\alpha), \quad k = 1, \ldots, M. \quad (19) \]

One of the solutions of this system corresponds to the element of best approximation, so we need to know how many solutions the system has. Since the elements \( \{B_i^\alpha, \quad i = 1, \ldots, M\} \) are linearly independent due to their selection based on Lemma 1.2, the matrix \( B_M = [B_i^\alpha, B_j^\alpha] \) is positively defined [29], i.e. from \( (B_M g, g) = 0 \) it follows that \( g \equiv 0 \). Since \( B_M \) is positively defined, its determinant cannot be zero and the system (19) has a unique solution which defines, due to its uniqueness, the element of best approximation.

For discrete digital images, integrals in (18) are replaced by summations:

\[ \|f\|_d = \sum_{x,y} f^2(x,y) q(x,y) \]
\[ (f, g)_d = \sum_{x,y} f(x,y) g(x,y) q(x,y). \quad (20) \]

Using \( q(x, y) = w^\alpha(x, y) \) and combining (13), (19) and (20), we obtain:

\[ \sum_{i=1}^M c_i \left\{ \sum_{x,y} w^\alpha(x, y) B_i^\alpha(x, y) B_k^\alpha(x, y) \right\} = \sum_{x,y} w^\alpha(x, y) I_{log}^\alpha(x, y) B_k^\alpha(x, y), \quad k = 1, \ldots, M \]

(21)

For every \( \alpha \), (21) has a unique solution \( C^\alpha = \{c_1^\alpha, \cdots, c_M^\alpha\} \) which can be obtained directly using Gaussian elimination.

**Lemma 1.3.** Let \( x \in L \), where \( L \) is a normal linear space with norm \( \|\cdot\|_L \). Let \( x_1, \ldots, x_M \in L \) be
the elements that approximate \( x \) on \( L \): \( x = x_i + \delta_i, \quad i = 1, \ldots, M \), \( \delta_i \in L \) being the approximation errors. Then if \( \bar{x} = \frac{1}{M} \sum_{i=1}^{M} x_i \), the inequality

\[
\|x - \bar{x}\|_L \leq \max_i \|\delta_i\|_L
\]  

(22)

holds.

**Proof.** We can represent \( x \) as \( \frac{1}{M} \sum_{i=1}^{M} (x_i + \delta_i) \), so that

\[
\|x - \bar{x}\|_L = \left\| \frac{1}{M} \sum_{i=1}^{M} (x_i + \delta_i) - \frac{1}{M} \sum_{i=1}^{M} x_i \right\|_L = \\
\frac{1}{M} \left\| \sum_{i=1}^{M} \delta_i \right\|_L \leq \frac{1}{M} \sum_{i=1}^{M} \|\delta_i\|_L \leq \frac{1}{M} \sum_{i=1}^{M} \max_k \|\delta_k\|_L = \max_k \|\delta_k\|_L,
\]

which proves (22). It should be observed that the estimate \( \bar{x} \) cannot be improved: in the case when \( \delta_1 = \delta_2 = \cdots = \delta_M \) (22) is an equality.

The solution \( C^\alpha \) of system (21) provides an approximation to the true bias coefficient vector \( C = (c_1, \cdots, c_M) \). Let \( A \) be the set of all partial derivatives for which the solutions are obtained. Combining solutions for all \( \alpha \in A \), we can develop an approximation of \( C \)

\[
\tilde{C} = \frac{1}{|A|} \sum_{\alpha \in A} C^\alpha
\]

(23)

From Lemma 1.3, the error of this approximation \( \tilde{\delta} = \|C - \tilde{C}\|_L \) is guaranteed to be smaller than or equal to the error of every \( C^\alpha \tilde{\delta}^\alpha = \|C - C^\alpha\|_L \). If the distribution of \( \delta^\alpha \) is symmetric about zero, the proof of Lemma 1.3 suggests that the error reduction may be significant, which is important for practical computations.
1.7 The Algorithm

1.7.1 A General Scheme of Dsf Algorithm

Based on Sections 1.6.1-1.6.3, we can now develop a general form of the dsf algorithm for the MR bias correction. In this section, the major steps involved are presented. The general algorithm steps and data flow are shown on Figure 1.10.

1. **Initialization.** The input includes an original $m \times n$ image matrix $I(x, y)$, set $A$ containing the orders of partial derivatives and the basis parameter set $\Omega$. The latter depends on type of the basis discussed in the next section.

2. **Edge preserving smoothing of** $I(x, y)$. This step reduces the computational error on further steps.

3. **Log-transform of** $I(x, y)$. We can assume $I(x, y) \geq 0$ for all $x, y$. To minimize the error associated with the additive noise component $n(x, y)$ in (7), the log-transform is performed to hold (10) true in accordance with (9):

\[
I_{\log}(x, y) = \log[I(x, y) + 2].
\]

4. **Calculation of image partial derivative matrices** $I_{\log}^\alpha$, $\alpha \in A$. This is done using the partial derivative scheme described in Section 1.7.3.

5. **Generation of basis.** Based on $\Omega$, a set of $m \times n$ basis matrices

$\{B_i(x, y)\}, i = 1, 2, \ldots, N$ is generated. For every $\alpha \in A$, matrices

$\{B_i^\alpha(x, y)\}, i = 1, 2, \ldots, M$ are constructed using an analytical expression for each corresponding $D^\alpha$. When the analytical form of $B_i$ is not available, we can use the finite difference scheme used to calculate $I_{\log}^\alpha$.

6. **Calculation of the weight matrices** $w^\alpha(x, y), \alpha \in A$. The weights are used to remove from consideration the inter-tissue areas and fine detail which cannot be described by the piecewise constant model. Justification of our approach for calculation of $w^\alpha$ is
Figure 1.10. The general steps of the \textit{dsf} algorithm. Dashed arrows indicate optional data flow.
presented in Section 1.7.5.

7. **Construction of linear system** $S_\alpha$ **for every** $\alpha \in A$ **and solving** $S_\alpha$ **with Gaussian elimination.** In this step, the $S_\alpha$ ’s are defined using summations in (21) and, since they are positively defined, they can be solved with Gaussian elimination.

8. **Construction of a final solution.** In this step, pooling of the solutions obtained on Step 7 using (23) is done to reduce the resulting error.

9. **Inverse log-transform and scaling.** After the final solution for the bias field $\tilde{f}_{\log} = \sum_{i=1}^{M} \tilde{c}_i B_i$ is obtained, the inverse log-transform $\log^{-1}$ is applied to the corrected image:

$$
\tilde{I} = \log^{-1}(I_{\log} - \tilde{f}_{\log})
$$

Finally, the scaling transformation

$$
T_S (\tilde{I}) = \left(\tilde{I} - \min_{x,y} \tilde{I}\right) \frac{\max_{x,y} I - \min_{x,y} I}{\max_{x,y} \tilde{I} - \min_{x,y} \tilde{I}} + \min_{x,y} I
$$

is applied to preserve the intensity range of original image. The transform $T_S (\tilde{I})$ produces the final form of the corrected image. There exists a problem of original and corrected image histogram mismatch. We discuss the way to improve $T_S (\tilde{I})$ in Section 1.7.6.

1.7.2 **Edge Preserving Smoothing**

The random additive noise reduction is a fast pre-processing step which improves the robustness of the results of further processing. According to (10), the approximation error by the finite basis in the log-domain depends directly on the magnitude of additive noise component $n(x, y)$. At the same time, a denoising filter should not remove or blur the edges in
an image, since that would affect the values of the partial derivatives and thus interfere with our *dsf* algorithm.

We have chosen to implement the edge preserving filter with the following convolution formula:

\[
\hat{I} = \sum_{i,j} k^{\beta(i,j) - I(0,0)} - \beta (i^2 + j^2) - I(0,0)
\]

where \( \hat{I} \) is the resulting intensity, the summations are done over the filter core dimensions, and \( k, \alpha, \beta, \sigma \) are coefficients that determine the strength of the filter and contributions of its various components. Edge preservation is achieved by exponentiation of the contribution weights, which are based on the intensity difference and distance from the central point.

Figure 1.11 illustrates the application of this filter to a brain image.

### 1.7.3 Partial Derivative Estimation

Since our solutions of resultant linear systems (21) depend on the error in calculation of partial derivatives, their estimates should be robust and the error minimal. In this section, we will examine various partial derivative schemes and their ability to produce satisfactory estimates. Let the function \( I(x, y) \) be defined on Cartesian grid \( I_{i,j} \), \( I_{i,j} = I(ih, jh) \). Without limiting generality, we will consider only the first-order partial derivatives in detail and provide necessary remarks about higher orders.

For two-point schemes, *symmetric* finite difference schemes produce better approximations.

For \( I_{i,j} \), \( \frac{\partial I}{\partial x} \) and \( \frac{\partial I}{\partial y} \) are approximated:
\[
\begin{align*}
\frac{\partial I}{\partial x} \bigg|_{x=ih, y=jh} &\sim L_{2,x} = \frac{I_{i+1,j} - I_{i-1,j}}{2h}, \\
\frac{\partial I}{\partial y} \bigg|_{x=ih, y=jh} &\sim L_{2,y} = \frac{I_{i,j+1} - I_{i,j-1}}{2h},
\end{align*}
\]
with residual errors,

\[
R_x = -\frac{h^2}{6} I_{xxx}(\xi, y), \quad \xi \in [(i-1)h, (i+1)h],
\]
\[
R_y = -\frac{h^2}{6} I_{yyy}(x, \xi), \quad \xi \in [(j-1)h, (j+1)h],
\]

**Figure 1.11.** Effect of edge preserving smoothing on a brain image. Contrast optimized magnified area in the source (left) and in the denoised MR image (right). The intra-tissue intensity variation becomes smoother while edges are preserved.

The inference is based on Taylor series expansions and Rolle's theorem [29], p. 79. Although the approximations (27) have residual errors \(O(h^2)\), the exact values of the function on the grid knots are not known because images contain random noise. So in the calculation of partial derivatives (27) the approximate value \(\tilde{I}_{i,j} = I_{i,j} + n_{i,j}\) is used instead of the exact value \(I_{i,j}\). The maximum residual is \(R_{\max} = \frac{1}{6} \max (|I_{xxx}(\xi, y)|, |I_{yyy}(x, \xi)|)\) and the random noise magnitude is \(E = \max_{i,j} |n_{i,j}|\). From (27) and (28), we obtain for \(\frac{\partial I}{\partial x}\) (similarly for \(\frac{\partial I}{\partial y}\) )
\[
\frac{I_{i+1,j} - I_{i-1,j}}{2h} = \frac{I_{i+1,j} - I_{i-1,j}}{2h} + \frac{n_{i+1,j} - n_{i-1,j}}{2h} = \frac{\partial f}{\partial x_{x=ih, y=jh}} + r, \quad r \leq R_{\text{max}} h^2 + \frac{E}{h}.
\]

The term \(O\left(\frac{M}{h}\right)\) increases as \(h\) decreases, so decreasing \(h\) leads to increases in the resulting error. The minimum of the expression \(R_{\text{max}} h^2 + \frac{E}{h}\) is when \(h = \frac{\sqrt{E}}{\sqrt{2} R_{\text{max}}}\). From experience, \(R_{\text{max}} \sim \frac{1}{12} \max I(x, y)\) for the typical MR image and the random noise varies within

\[
0.01 \max_{x,y} I(x, y) \leq E \leq 0.09 \max_{x,y} I(x, y),
\]

from which

\[
0.4 \leq h \leq 0.8
\]

is the optimal interval grid step. The minimum digitization step in a digital image \(I(x, y)\) is 1, and our estimate shows that \(h = 1\) is acceptable for use in a finite difference scheme for the calculation of partial derivatives. Formally, it is possible to use \(h < 1\) by interpolating the values between the pixels. However, any such interpolation would be based on the discrete values with the same sampling points, which leads to a finite difference scheme based on more points, and the resulting error will not be reduced. To maintain the same error with higher order partial derivatives, the number of grid points in the appropriate scheme should be increased.

What will happen if we add more grid points to (27)? Suppose \(N\) knots are used in a finite difference approximating a first order partial derivative:

\[
D^n I(ih, jh) = \frac{1}{K} \sum_{t=1}^{N} a_t \tilde{I}_{i^t, j^t} = \frac{1}{K} \sum_{t=1}^{N} a_t I_{i^t, j^t} + \frac{1}{K} \sum_{t=1}^{N} a_t n_{i^t, j^t},
\]

with

\[
K = \sum_{t=1}^{N} |a_t|.
\]
The noise $n_{ij}$ at point $(i, j)$ can be modeled as a random variable with Gaussian distribution with $\mu = 0$ and $\sigma^2 = M^2$. The sum

$$\frac{1}{K} \sum_{t=1}^{N} a_{t} n_{i+t, j+t},$$

also has a Gaussian distribution with $\mu = 0$ and

$$\sigma^2 = \frac{M^2}{K^2} \sum_{t=1}^{N} a_{t}^2.$$  

Since

$$\frac{1}{K^2} \sum_{t=1}^{N} a_{t}^2 = \frac{\sum_{t=1}^{N} a_{t}^2}{\left(\sum_{t=1}^{N} a_{t}\right)^2} \to 0, \quad N \to +\infty,$$  

(31)

$\sigma^2 \to 0$ as $N \to +\infty$. Because $\sigma^2$ determines the variation of error in this finite difference scheme, the error that results from inaccurate measurements at image sampling points will be reduced by the factor defined in (31).

It is desirable to use a small neighborhood window for a partial derivative scheme. The design of our algorithm requires accurate detection of tissue interface areas to exclude them from consideration, and some borders in an image may have very sharp definition. For this reason, the introduction of the points located far from the point of interest in the calculation of a local derivative would reduce the accuracy of such calculation. In a nine-pixel neighborhood centering in a point of interest, consider the patterns depicted on Figure 1.12. In the initial first order partial derivative approximation (27), the points shown as 2 on Figure 1.12 are used. The points shown as 1 and 3 can also produce a symmetric derivative estimate.

The derivation for $\frac{\partial I}{\partial x}$ using the points labeled 2 on Figure 1.12 will be presented, and $\frac{\partial I}{\partial y}$ can be derived similarly.
Consider the Taylor series expansions using $\frac{\partial I}{\partial y}$:

$$I_{i\pm h_j^\pm} = I_{ij} \pm h \frac{\partial I_{ij}}{\partial y} + \frac{h^2}{2} \frac{\partial^2 I_{ij}}{\partial y^2} + \frac{h^3}{6} \frac{\partial^3 I_{ij}}{\partial y^3} + \frac{h^4}{24} \frac{\partial^4 I_{ij}}{\partial y^4} + \cdots,$$

from which we obtain

$$I_{i\pm h_j} = \frac{1}{2}(I_{i\pm h_j-1} + I_{i\pm h_j+1}) \pm \frac{h^2}{4} \frac{\partial^2 I_{ij}}{\partial y^2} \pm \frac{h^3}{12} \frac{\partial^3 I_{ij}}{\partial y^3} \pm \frac{h^4}{48} \frac{\partial^4 I_{ij}}{\partial y^4} + \cdots. \quad (32)$$

Now we can also consider an expansion for $I_{i\pm h_j}$:

$$I_{i\pm h_j} = I_{ij} \pm h \frac{\partial I_{ij}}{\partial x} + \frac{h^2}{2} \frac{\partial^2 I_{ij}}{\partial x^2} \pm \frac{h^3}{6} \frac{\partial^3 I_{ij}}{\partial x^3} \pm \frac{h^4}{24} \frac{\partial^4 I_{ij}}{\partial x^4} + \cdots,$$

from which

$$\frac{\partial I_{ij}}{\partial x} = \frac{I_{i+1,j} - I_{i-1,j}}{2h} + \frac{h^2}{6} \frac{\partial^2 I_{ij}}{\partial x^2} + O(h^4).$$

Combining this with (32), the final form of the four-point finite difference approximation $L_{4x}$ for $\frac{\partial I}{\partial x}$ follows:
\[
\frac{\partial I_{i,j}}{\partial x} = \frac{1}{2h} \left( \frac{1}{2} (I_{i+1,j-1} + I_{i+1,j+1}) - \frac{1}{2} (I_{i-1,j-1} + I_{i-1,j+1}) + \frac{h^2}{4} \frac{\partial^2 I_{i+1,j}}{\partial y^2} \right) + \\
h^2 \frac{\partial^3 I_{i,j}}{\partial x^3} + O(h^4) = \frac{I_{i+1,j-1} - I_{i-1,j-1} + I_{i+1,j+1} - I_{i-1,j+1}}{4h} + O(h) = L_{4,x} + O(h)
\]

Thus we obtain the diagonal difference scheme approximation for the partial derivative \( \frac{\partial I}{\partial x} \).

To obtain a six-point linear approximation, combine this with (27):
\[
\frac{\partial I_{i,j}}{\partial x} = \frac{1}{2} (L_{2,x} + L_{4,x}) + O(h^2) = L_{6,x} + O(h^2)
\]

The analogous expression for \( \frac{\partial I}{\partial y} \) is
\[
L_{6,y} = \frac{I_{i-1,j+1} - I_{i-1,j-1} + 2I_{i+1,j+1} - 2I_{i+1,j-1} + I_{i+1,j+1} - I_{i+1,j-1}}{8h}
\]

**Figure 1.13.** Evaluation of the six-point and two-point approaches. The histogram of corrected image using \( L_2 \) (curve 2) and \( L_6 \) (curve 1).

The variance of random error for (33) and (34) is \( \frac{1^2 + 2^2 + 2^2 + 2^2 + 1^2}{8^2} = \frac{1}{8} \) of the original random noise variance \( E \). Since \( L_{6,x} \) and \( L_{6,y} \) are two-point approximations, (29)
is also valid for these and using $h = 1$ allows this inaccurate measurement error to be close to a theoretical minimum. However this reduces the approximation accuracy for the derivative from $O(h^2)$ in $L_2$ to $O(h)$ in $L_6$.

Even so, $L_6$ produced more suitable derivative estimates for $dsf$. We compared $dsf$ performance using $L_2$ and $L_6$. $L_6$ resulted in higher peaks in the histogram of corrected images (see Figure 1.13), and thus provided better distinction between different soft tissues.

Calculation of higher order partial derivatives is based on similar considerations.

### 1.7.4 Selection of Approximation Functions

The existing methods for a one-dimensional approximation are well developed and standardized. The performance of current computers allows most one-dimensional problems to be solved with standard methods developed from theoretical research.

The complexity of these problems rises sharply as their dimensionality increases, and multi-dimensional methods usually do not provide the same level of accuracy as one-dimensional methods. For this reason, approximation functions are usually selected for each particular problem. Even if a set of approximation functions seems suitable, their use requires theoretical substantiation.

In MR images, the approximation functions $B_i(x, y)$ should be both smooth and slow varying, which means that higher order derivatives of $B_i(x, y)$ should be approximately zero. In the one-dimensional problem, using either a polynomial or trigonometric basis for the smooth function approximation, one could expect to use this same approach for two- and three-dimensional medical image problems. Multi-dimensional approximation functions can be obtained from the single-dimensional function set using a Cartesian product: from the approximation functions of one variable $B_i(x)$ we can generate functions of two variables, i.e. $B_{i,j}(x, y) = B_i(x)B_j(y)$, $i, j = 1, \cdots, N$. If the function set $B_1(x), \cdots, B_N(x)$ is linearly
independent, then the 2D function set \( B_{1,1}(x, y), \cdots, B_{N,N}(x, y) \) (without the constant function, see Section 1.6.3) is also linearly independent. Indeed, suppose the opposite: in such, there should exist a non-trivial linear combination

\[
\sum_{i=1}^{N} \sum_{j=1}^{N} c_{ij} B_i(x) B_j(y) = 0.
\]

This means

\[
\sum_{i=1}^{N} B_i(x) \sum_{j=1}^{N} c_{ij} B_j(y) = 0
\]

for all values of \( x, y \). Let \( y_0 \) be some fixed value for which some

\[
\tilde{c}_i = \sum_{j=1}^{N} c_{ij} B_j(y_0)
\]

is not zero. Such value exists since \( B_1(x), \cdots, B_N(x) \) are linearly independent, and we have a contradiction:

\[
\sum_{i=1}^{N} \tilde{c}_i B_i(x) = 0.
\]

Therefore, the Cartesian product generates a set of linearly independent basis functions.

This constructed two-dimensional basis is \((N+1)^2\). To reduce its complexity, we can limit the number of its higher-order members: \( i + j \leq N \) for all \( B_{i,j}(x, y) \). In this case, the basis size is reduced in two dimensions to

\[
\frac{(N + 1)(N + 2)}{2} - 1.
\]

We considered the two types of functions that are most frequently used in one-dimensional approximation problems: sets of polynomials \( P_n(x) \) of degree \( n \) and trigonometric functions

\[
T_n(x) = \exp(2\pi in x).
\]
Figure 1.14. Comparison of basis functions. Polynomial basis functions (left) vs. trigonometric (right).

Figure 1.14 shows the examples of polynomial and trigonometric two-dimensional basis functions, where $N = 3$. Both $P_n(x)$ and $T_n(x)$, may not approximate the bias field ideally. The problem with polynomial functions is their unlimited growth on borders, which can potentially lead to a loss in accuracy. It may also be useful to consider other systems of polynomials with special properties. In particular, orthogonal polynomials are commonly used as interpolation functions because they have many useful properties. For example, zeros of orthogonal polynomials cannot be multiple, and are distributed asymptotically uniformly on any given line segment [29].

With $dsf$ we used two systems of orthogonal polynomials: Legendre polynomials and Hermite polynomials. Legendre polynomials

$$L_n(x) = \frac{1}{2^n n!} \frac{d^n}{dx^n} (x^2 - 1)^n$$

have a norm

$$\|L_n\| = \sqrt{2/(2n+1)}$$

and their coefficients are computed using the recurrence relation

$$(n+1)L_{n+1}(x) - (2n+1)xL_n(x) + nL_{n-1}(x) = 0.$$ 

Hermite polynomials
\[ H_n(x) = (-1)^n e^{x^2} \frac{d^n}{dx^n}(e^{-x^2}) \]

have a norm
\[ \|H_n\| = \sqrt{2^n \cdot n! \sqrt{\pi}} \]

and their coefficients are computed from the relation
\[ H_{n+1}(x) - 2xH_n(x) + 2nH_{n-1}(x) = 0. \]

Trigonometric functions have limited growth, but are more complex computationally than polynomials. Additionally,
\[ T'_n(x) = 2\pi i\exp(2\piinx) \]

would not approximate a constant uniformly. This would reduce accuracy when the bias field is small.

To select the set of functions for approximation, it is useful to estimate the error of solution in each case. Suppose we are solving a linear system
\[ Ax = b \]
and its coefficients are known only approximately, and ideally the system
\[ A_1x = b_1, \quad A_i = A + \Delta, \quad b_1 = b + \eta \]
should be solved. Let \( X \) be the solution of (35), \( X^* \) the solution of (36), and \( X - X^* = r \). The following estimate of \( r \) is correct (29):
\[ \|r\| \leq \frac{\|A^{-1}\|\|h\| + \|\Delta\|\|X\|}{1 - \|A^{-1}\|\|\Delta\|}. \]

In (21), the left hand side is known precisely, in this case \( \Delta = 0 \) and we can write
\[ \|r\| \leq \|A^{-1}\|\|h\|. \]

The quantity
\[ \tau = \sup_{\eta} \left( \frac{\|r\|}{\|X\| \|\eta\|} \right) = \frac{\|b\|}{\|X\|} \sup_{\eta} \frac{\|r\|}{\|\eta\|} = \frac{\|b\|}{\|X\|} \|A^{-1}\| \]

expresses the connection between relative errors of the right hand side and the solution:

\[ \frac{\|r\|}{\|X\|} \leq \tau \frac{\|h\|}{\|b\|} \quad \text{(37)} \]

and is called the condition measure of a system. We can also consider the characteristic of a system

\[ \nu(A) = \sup_{b} \tau \]

based on the left-hand side of a system only. It is called the condition number of a matrix \( A \), so (37) can be rewritten as

\[ \frac{\|r\|}{\|X\|} \leq \nu(A) \frac{\|h\|}{\|b\|} . \]

It is clear that \( \nu(A) \geq 1 \) and its magnitude is proportional to the relative error. To compute \( \nu(A) \), since

\[ \sup_{b} \frac{\|b\|}{\|X\|} = \sup_{x} \frac{\|Ax\|}{\|x\|} = \|A\| , \]

we have

\[ \nu(A) = \frac{\|A\|}{\|A^{-1}\|} . \]

### Table 1.1. Effect of basis selection on condition number of resulting system. Condition numbers of the system (21) for different basis functions are shown.

<table>
<thead>
<tr>
<th>( P_n(x) )</th>
<th>( L_n(x) )</th>
<th>( H_n(x) )</th>
<th>( T_n(x) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \nu(A) )</td>
<td>( 10^7 )</td>
<td>( 10^4 )</td>
<td>( 10^3 )</td>
</tr>
</tbody>
</table>

To select basis functions with the lowest error, we randomly chose ten MR images and evaluated the condition number of (21) for each MR image using different basis functions.
These results are summarized in Table 1. Based on these results, $P_n(x)$ was chosen as the basis resulting in a linear system with the smallest condition number.

1.7.5 Weight Functions $w_\alpha(x, y)$

Some points in an image may have values that produce errors in the calculation of the coefficients for (21). Weight functions $w_\alpha(x, y)$ are used to prevent any negative effect such points may have upon resulting error. There are two possible sources of singularity: a very low signal and a high magnitude of the gradient

$$\nabla I = \left( \frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right).$$

Without limiting generality, we can define the points with a very low signal $\Lambda$ as those where the image has an intensity with magnitude smaller than a small $\epsilon_\Lambda$:

$$(x, y) \in \Lambda \iff I(x, y) \leq \epsilon_\Lambda.$$

Suppose $n(x, y) \leq E$. If $E - \epsilon$, the condition (9) of Lemma 1.1 will not hold for $\Lambda$, and we cannot consider the noise component in (7) small enough to perform a log-transformation. Since random noise usually varies between 1% and 7% of the intensity range, it is sufficient to exclude points belonging to $\Lambda$ from (21), where

$$\epsilon_\Lambda = 0.1I_{\text{max}}.$$

Accordingly, we can define the weight function for a very low signal as

$$w_{u,\Lambda}(x, y) = \begin{cases} 1, & I(x, y) \geq 0.1I_{\text{max}} \\ 0, & \text{otherwise} \end{cases}.$$  \hspace{1cm} (38)

Similarly, we can define a set $\Gamma$ with a high gradient magnitude by specifying the upper bound:

$$(x, y) \in \Gamma \iff \nabla I(x, y) \leq \epsilon_\Gamma.$$
Notice that the weight function is zero when $\nabla I(x, y) > \varepsilon_\Gamma$. In practice, the distribution of gradient values is not known in advance, and calculating $\varepsilon_\Gamma$ for every image is more accurate than using a fixed value for all images. For that, we can select $\varepsilon_\Gamma = \varepsilon_\beta$ based on a fixed $\beta$:

$$\varepsilon_\beta = \nabla I(x_\beta, y_\beta),$$

where the point $(x_\beta, y_\beta)$ is such that if

$$S_1 = \{(x, y) : \|\nabla I(x, y)\| \leq \|\nabla I(x_\beta, y_\beta)\|, \quad x, y \in [1 \cdots m] \times [1 \cdots n]\},$$

then

$$\frac{1}{mn} |S_1| = \beta.$$

From statistics, $\nabla I(x_\beta, y_\beta)$ represents a $\beta$-percentile of $S$. To approximately compute $\varepsilon_\beta$, it is sufficient to determine a sequence $\{x_i, y_i\}$ of all image points ordered by gradient magnitude. Since the number of points is $mn$,

$$\varepsilon_\beta = \nabla I(x_\beta, y_\beta) \approx a_{[mn\beta+0.5]}.$$

This has the same complexity as the quick sort algorithm for the one-dimensional array consisting of image elements. If $m > n$, its complexity is $O(mn \log m)$. The corresponding weight function is defined as

$$w_{a,\Gamma}(x, y) = \begin{cases} 1, & \|\nabla I(x, y)\| \leq \|\nabla I(x_\beta, y_\beta)\| \\ 0, & \text{otherwise} \end{cases}$$

(39)

We have only discussed weight functions that can be either 0 or 1. It is also possible to develop a continuous $w_{a,\Gamma}(x, y)$, for example

$$\hat{w}_{a,\Gamma}(x, y) = \frac{1}{e - 1} \left[ \exp\left(1 - \frac{\|\nabla I(x, y)\|}{\max_{x,y} \|\nabla I(x, y)\|} \right) - 1 \right].$$

Clearly, $\hat{w}_{a,\Gamma}(x, y)$ decreases exponentially as
\[ r_0 = \frac{\|\nabla I(x, y)\|}{\max_{x,y} \|\nabla I(x, y)\|} \]

increases. In points where the gradient is low \((r_0 \sim 0)\), the weight function is close to one \((\tilde{w}_{a,r}(x, y) \sim 1)\), and when the gradient is high \((r_0 \sim 1)\), then \(\tilde{w}_{a,r}(x, y) \sim 0\). Since this continuous weight function may provide additional sources of error, a comparison of \(dssf\) using a discrete and a continuous weight function was necessary. We tested \(dssf\) using \(w_{a,T}(x, y)\) and \(\tilde{w}_{a,T}(x, y)\) with every examined image in our “random” set. The result showed that the correction with \(\tilde{w}_{a,T}(x, y)\) depended significantly on the of edges in an image and their spatial distribution; this effect was not observed using \(w_{a,T}(x, y)\). This may be due to the continuous weights introducing more random factors into the system (21), which result in a less predictable outcome. For this reason, we used \(w_{a,T}(x, y)\) in the final version of \(dssf\) to obtain the more robust solution.

The weight function that allows excluding or significantly reducing both instability factors is obtained from the combination of (38) and (39):

\[ w(x, y) = w_{a,T}(x, y) \cdot w_{a,A}(x, y). \]

1.7.6 Resultant Image Scaling

The inverse log-transform (24) produces the image \(\tilde{I}(x, y)\). This resultant image often has an intensity distribution different than the original image. The post-acquisition correction methods tend to shrink the histogram of the original image [26], which can complicate a proper intensity based tissue identification (see Figure 1.15). For this reason, a more detailed analysis of the scaling transformation is needed.

In this work, only linear image transformations are considered, although only the exact intensity registration of original and corrected images would provide an ideal match.
However, registration is a complex problem beyond the scope of this study. In this section, an improvement to the matches between the original and corrected image histograms using only linear transformations are discussed.

Let \( \tilde{I} \) be the image obtained by inverse log transformation at Step 8 of the \( dsf \) algorithm. We seek scalars \( a, b \) such that the final output image

\[
\hat{I} = a \tilde{I} + b E,
\]

where \( E \) is \( m \times n \) matrix and \( E(i, j) = 1 \), would provide the best intensity range match with the original image \( I \). Transformation (40) preserves piecewise continuous functions, and therefore the result of non-uniformity correction is also preserved. However, the intensity range of the result may not match the intensity range of original image

\[
\begin{bmatrix}
I_{\text{min}} = \min_{x,y} I(x, y),
I_{\text{max}} = \max_{x,y} I(x, y)
\end{bmatrix}.
\]

Requiring this leads to the following conditions on \( a \) and \( b \):

\[
a = \frac{I_{\text{max}} - \tilde{I}_{\text{min}}}{I_{\text{max}} - \tilde{I}_{\text{min}}}, \quad b = \frac{\tilde{I}_{\text{min}} - I_{\text{max}}}{I_{\text{max}} - \tilde{I}_{\text{min}}}
\]

and (40) becomes (25). Clinicians often require preserving the original points of the image, so transformation (25) is only available as an option in the \( dsf \) algorithm. Since additive random noise is always present, the maximum intensity observed in an image can be considered a random variable and we can determine its distribution. That is, if \( |n(x, y)| \leq E \), we can consider the brightest tissue \( T_{\text{high}} \) in an image to have intensities in the range

\[
[I_{\text{high}} - E, I_{\text{high}} + E],
\]

where \( I_{\text{high}} \) is the tissue average. All the points belonging to \( T_{\text{high}} \)

\( P_1, \cdots P_k \) represent a sample of size \( k \) from the normal distribution with unknown parameters.

The order statistics \( P_{(1)} \leq \cdots \leq P_{(k)} \) can be obtained from this sample. Their distribution density function is described by the expression [31]
\[ P_{(i)} \sim p_{(i)}(x) = \frac{k!}{(i-1)!(k-i)!} [F(x)]^{i-1} [1 - F(x)]^{k-i} f(x) , \]

Figure 1.15. Spm correction. Histogram of original image \( I(x, y) \) shown on top and correction \( I_{\text{spm}}(x, y) \) produced by spm algorithm [25] shown on bottom. The average intensity \( \bar{I}(x, y) = 40.3 \), whereas \( \bar{I}_{\text{spm}}(x, y) = 50.6 \). where \( f(x) \) is normal probability density function (pdf):

\[
f(x) = \frac{1}{\sigma \sqrt{2\pi}} \exp \left[ -\frac{(x - \mu)^2}{2\sigma^2} \right].
\]

and \( F(x) \) is normal cumulative distribution function (cdf):

\[
F(x) = \int_{-\infty}^{x} f(t) dt .
\]

The maximum of this sample \( P_{(k)} \) is distributed as

\[
P_{(k)} \sim p_{(k)}(x) = k[F(x)]^{k-1} f(x) . \tag{41}
\]

Accordingly, the probability the observed intensity maximum is within \([I_{\text{high}} - E, I_{\text{high}} + E]\) is
\[
\Pr_{\text{max}} = \int_{I_{\text{max}} - E}^{I_{\text{max}} + E} p(t) \, dt.
\]

Using the sample mean and variance for the brightest tissue, we can evaluate this integral numerically. We selected \( E = 0.02 I_{\text{max}} \), and computed the confidence interval limits

\[
\Pr(I_{\text{max}} - t_{\alpha} \leq P(t) \leq I_{\text{max}} + t_{\alpha}) = \alpha
\]

for ten 256×256 8-bit grayscale MR images with \( \alpha = 0.95 \). \( t_{\alpha} \) varied between 0.06\( I_{\text{max}} \) and 0.15\( I_{\text{max}} \) throughout this study. This is the error in the intensity range that may result from (25) when the correction transformation was assumed to be linear. Since it is not linear, the actual error may be higher.

Because of image intensity randomness in any given point, averaging estimators provide a more robust landmark for the accurate estimation of transformation parameters in (40). It is convenient to consider three estimators:

1. The average intensity of the set \( \Lambda \) from Section 1.7.5

\[
I_{\text{low}} = \frac{1}{|\Lambda|} \sum_{x,y \in \Lambda} I(x, y),
\]

2. the global image average intensity \( I_{\text{avg}} \),

3. \( I_{\text{high}} \).

Using \( I_{\text{low}} \) is not desirable since \( \Lambda \) is an area of low signal-to-noise ratio. We discussed this in a previous section that the inference based on the points from this area is error-prone. Since we only need two parameters to define a linear transformation, we choose \( I_{\text{avg}} \) and \( I_{\text{high}} \). Hence:

\[
I_{\text{avg}} = a\tilde{I}_{\text{avg}} + bE, \quad I_{\text{high}} = a\tilde{I}_{\text{high}} + bE,
\]

from which
The transformation defined by (42) permits an adequate match of the histograms. When the non-uniformity in the original image is not significant, it is possible to find \( a, b \) more accurately using a least squares minimization:

\[
\| I - \tilde{I} \|^2 = \sum_{x,y} [I(x,y) - \tilde{I}(x,y)]^2 = \sum_{x,y} [I(x,y) - (a\tilde{I}(x,y) + b)]^2 \rightarrow \min.
\]

Setting the derivatives by \( a, b \) to zero, we obtain the following linear system:

\[
\begin{align*}
\sum_{x,y} \tilde{I}(x,y)[I(x,y) &- (a\tilde{I}(x,y) + b)] = 0 \\
\sum_{x,y} [I(x,y) &- (a\tilde{I}(x,y) + b)] = 0
\end{align*}
\]

Noticing that \( \sum_{x,y} 1 = mn \), we can solve this system as follows:

\[
S_I = \sum_{x,y} I(x,y), \quad S_{II} = \sum_{x,y} I(x,y)\tilde{I}(x,y), \quad S_{I\tilde{I}} = \sum_{x,y} \tilde{I}^2(x,y), \quad S_{\tilde{I}} = \sum_{x,y} \tilde{I}(x,y)
\]

lead to the following expressions for \( a, b \):

\[
a = \frac{mnS_{II} - S_I^2}{mnS_{I\tilde{I}} - S_I S_{\tilde{I}}}, \quad b = \frac{1}{mn}(S_I - aS_{\tilde{I}}).
\] (43)

These coefficients can be used for an accurate least squares solution.

In summary, our original goal was to improve the match between histograms of the original and the corrected images. We discussed image characteristics that were not directly related to its histogram to obtain the linear transformation. To define this relation, recall that the histogram is obtained by intensity summations (4), and therefore the linear transformation of an image produces the histogram

\[
H(a\tilde{I} + bE) = aH(\tilde{I}) + b.
\]

Therefore, the desirable properties of final image are also reflected in its histogram.
1.7.7 Implementation of Dsf

Originally, \( \text{dsf} \) was implemented in MATLAB 6, the environment suitable for quick (although sometimes inefficient) development and testing of numerical algorithms. Since MATLAB is available for major operating systems, verification of \( \text{dsf} \) performance is possible both in Windows and UNIX based operating systems.

In development of \( \text{dsf} \) pseudocode, we defined the following procedures:

- **Main**: the main control function which takes the image and algorithm parameters as an input and outputs the corrected image;

- **Generate\_basis\_matrices** – takes basis parameters as an input and outputs the matrices of basis functions and analytically calculated partial derivatives of basis functions;

- **Generate\_derivative\_matrices** - takes log-transformed image as an input, produces a set of partial derivative matrices of this image as an output;

- **Partial\_derivative** – takes a matrix and partial derivative order as an input, returns partial derivative matrix obtained by convolution with a finite difference scheme approximating this partial derivative.

See Appendix A for complete \( \text{dsf} \) pseudocode listing.

For extensive testing and use, an image processing algorithm should be implemented efficiently. The efficient implementation is usually based on selecting a specific operating system and using platform dependent development tools. In many research environments, platforms from the UNIX family are preferred for development and testing of new software. This choice in many cases is defined historically by existing infrastructure and availability of low cost software for research purposes. For example, the majority of free medical image processing tools are developed under Unix-like platforms. Of several available implementations of MR inhomogeneity correction algorithms referred in this work, only one
could run under the Windows platform, and only because the code was written in MATLAB.

Figure 1.16. UniViewer main window.

Despite the seeming attractiveness of the UNIX platform, Windows was chosen for implementation of this algorithm. The reason for this is that since the extensive testing was required to validate dsf performance, it was desirable to run it in many different locations on a diverse input. Therefore, our purpose was to develop the software for work in most clinical environments, and the Windows platform is more suitable for this purpose. Previously our group had developed the DICOM (Digital Communications in Medicine) PACS (Picture Archiving and Communication System). Part of it was a Windows-based DICOM viewer called UniViewer capable of displaying and manipulating images in all medical imaging modalities as they come from the scanner (see Figure 1.16 for the main view of the UniViewer). We incorporated dsf in UniViewer, which made it widely available for review, testing and use. C++ implementation of dsf is currently a part of UniViewer, which is available from: http://www.unipacs.com/en/uniView.html.
1.8 Basic Evaluation of Dsf

Numerical algorithm evaluation should achieve four goals:

1. Validation of implementation;
2. Testing performance on datasets with known “ideal” output;
3. Extensive testing of performance on large amounts of real data;
4. Comparison to other methods.

In this section, we describe the basic tests performed to verify that dsf decreases non-uniformity in MR images.

1.8.1 Synthetic Images

According to theoretical results in computation theory proven first by Alan Turing in 1936, the halting problem is in general unsolvable, so it is impossible to design a procedure that determines whether a given algorithm halts on some arbitrary given input or not. For this reason, the validity of a particular algorithm implementation is in general impossible to prove mathematically. The only possibility here is to verify that the implementation output is consistent with theoretical algorithm output through a series of experiments. The major procedure is:

1. One or several experiments that expose the key algorithm features;
2. Theoretical output of the algorithm is calculated and the experiments on its implementation are carried out;
3. The output is compared with theoretical estimates, and based on their match the conclusion about implementation correctness is drawn.

In the case for dsf, it is necessary to find a non-trivial image which can be corrected with predictable result. To do that, a piecewise constant image with non-uniformity described
by (3) can be used. We emulated a piecewise constant function as the monochrome “chessboard” image

\[
\Theta(x, y) = C_0 \cdot (-1)^{\left\lfloor \frac{x}{c_x} \right\rfloor \left\lfloor \frac{y}{c_y} \right\rfloor},
\]

where square brackets denote an integral part of a positive number, \( C_0 > 0 \) is a fixed constant, \( c_x, c_y \) – constants representing cell sizes.

To emulate the multiplicative bias field, we used the parabolic function. The biased image is defined as:

\[
I(x, y) = \Theta(x, y) \ast \left\{ \frac{x(w-x)}{w^2} + \frac{y(h-y)}{h^2} \right\}, \quad x \in [0 \ w], \ y \in [0 \ h].
\]

The result of correction by \( dsf \) is shown on Figure 1.17 (right). Comparison with \( \Theta(x, y) \) showed that the variation of intensity within any single class does not exceed 0.1%, so we can conclude that our implementation is consistent with the \( dsf \) algorithm.

Figure 1.17. Model image correction. Artificially distorted image (left), found bias field (center), corrected image (right)

1.8.2 Phantom Images

To evaluate \( dsf \) on data with a known bias field, we used simulated MRI brain image volumes available from McGill University [32]. MR image volumes were chosen with 1mm slice thickness and 40% non-uniformity. Six emulated volumes consisting of 181 slices each
were tested, three of normal brains with T1, T2 and proton density weighting and three with lesion brains. Correction produced results similar to those shown on Figure 1.18.

**Figure 1.18.** Phantom image correction. In the top row: original distorted image (left), true bias field (middle), biased image histogram (right); in the bottom row: corrected image (left), bias field found by algorithm (middle), corrected image histogram (right).

**Figure 1.19.** Axial image correction. Original image (top left), its histogram (top right), bias field found (bottom left) and the histogram of corrected image (bottom right).
1.8.3 Histogram Evaluation

It follows from Section 1.4.4 that histogram visual analysis can show whether an image was improved after correction. Height and width of histogram peaks provide information about the variance observed in different soft tissues of MR image, and efficient correction method should reduce the variance and increase the peak heights. The actual experiments with phantom images confirm that for dsf (Figure 1.18). We also compared the histograms of real MR images with corrections produced by dsf. The comparison was performed on two complete image volumes and a number of separate brain images from different sources and of different quality. In all cases, the histogram peaks increased in height after correction. The example of histogram comparison is shown on Figure 1.19.

1.9 Comparison with Selected Published Methods

In this section, we describe the extended analysis of dsf performance on a large volume of phantom and real images in comparison with selected previously published MR non-uniformity correction methods.

1.9.1 MR Artifact Correction Methods Chosen for Comparison with Dsf

Arnold et al. [26] divided the existing MR artifact correction methods into two groups: non-locally adaptive, where the parameters of the bias field at a particular point are determined using global image information, and locally adaptive, where the bias field at a given point is determined from local neighboring points. For comparison with dsf, we selected two previously published methods representing each of these groups: spm and bcfcm.

Spm is a non-locally adaptive method developed by Ashburner and Friston [25]. It uses pre-segmentation of the brain image to extract the white matter as the first approximation. After that, the bias field is iteratively approximated by EM using maximum log-likelihood. MATLAB implementation of spm is incorporated into freely available SPM.
software package [33] developed in the Department of Imaging Neuroscience, University College London (UCL).

**Bcfcm** (Bias-corrected Fuzzy C-Means) algorithm is a recently published by Ahmed, Yamany et al. [16] locally adaptive method. The idea of fuzzy C-means is to determine the tissue prototype cluster (median tissue intensity) for every soft tissue in an image and define an objective function for partitioning into $c$ clusters:

$$J = \sum_{i=1}^{c} \sum_{x,y} u_{i,x,y} \| f(x,y) - v_i \|^2,$$

(44)

where $v_i$ are prototype clusters and $u_{i,x,y} \in [0,1]$ determine fuzzy membership of the point $(x, y)$ in the $i$th cluster. **Bcfcm** extends (44) by introducing the bias field adjustment parameters; the resulting optimization problem is solved iteratively by sequential approximations. We implemented **bcfcm** in MATLAB. To validate our implementation, we chose the same BrainWeb phantom images [32] used in original **bcfcm** paper and compared our output with results reported by Ahmed and Yamany et al [16].

### 1.9.2 Testing Criteria

In section 1.2, the two main reasons for developing MR non-uniformity correction methods were discussed:

1. improving of visual quality

2. improving the intensity uniformity within a single soft tissue for subsequent automated processing.

Therefore, non-uniformity method testing should answer the question whether these two goals are achieved.

A visual comparison is inevitably subjective; therefore, conclusions about the correction method’s visual performance will be made after comparing corrections on large volumes of data from different subjects.
The uniformity within a single tissue is a more subtle characteristic and its accurate
visual detection is difficult, so a numerical evaluation is used. If each tissue \( s \) is modeled as a
random variable, the natural measure of its non-uniformity can be derived from sample
variance:

\[
\sigma^2(s) = \frac{1}{|s| - 1} \sum_{x,y \in s} [I(x, y) - \mu(s)]^2,
\]

where \(|s|\) is the number of points in \( s \) and

\[
\mu(s) = \frac{1}{|s|} \sum_{x,y \in s} I(x, y)
\]
is the sample mean. The actual magnitude of \( \sigma(s) \) depends on the amount of variation and on
the image intensity range, which does not allow comparing \( \sigma(s) \) for different images. To
avoid this, the normalized version of \( \sigma(s) \), called the coefficient of variation, will be used as
the tissue intensity measure of uniformity:

\[
cv(s) = \frac{\sigma(s)}{\mu(s)}.
\]
The coefficient of variation is invariant to a uniform scaling intensity transformation: since

\[
\mu(\lambda s) = \frac{1}{|s|} \sum_{x,y \in s} [\lambda I(x, y)] = \lambda \frac{1}{|s|} \sum_{x,y \in s} I(x, y) = \lambda \mu(s)
\]
and

\[
\sigma^2(\lambda s) = \frac{1}{|s| - 1} \sum_{x,y \in s} [\lambda I(x, y) - \mu(\lambda s)]^2 = \frac{1}{|s| - 1} \sum_{x,y \in s} \lambda^2 [I(x, y) - \mu(s)]^2 = \lambda^2 \sigma^2(s),
\]

\[
cv(\lambda s) = \frac{\sigma(\lambda s)}{\mu(\lambda s)} = \frac{\sqrt{\lambda^2 \sigma(s)}}{\lambda \mu(s)} = cv(s).
\]

However, \( cv(s) \) is not invariant to a uniform additive intensity transformation:

\[
cv(s + \lambda) = \frac{\sigma(s + \lambda)}{\mu(s + \lambda)} = \frac{\sigma(s)}{\mu(s) + \lambda} \neq cv(s)
\]
and therefore coefficients of variation cannot be compared for different tissues. Since the majority of MR correction algorithms were evaluated in the past with brain images, we chose to evaluate the coefficient of variation on the principal soft tissues of the brain (white matter (WM) and grey matter (GM)).

As discussed in Section 1.7.6, MR correction methods may significantly modify the original soft tissue means. In some cases, imaged tissues become harder to separate, i.e. for two soft tissues $s_1, s_2$ the quantity

$$\Delta \mu(s_1, s_2) = |\mu(s_1) - \mu(s_2)|$$

may decrease after correction. This would mean degrading the image quality instead of improving it, and Arnold et al. [26] describe this as a common problem in MR correction algorithms. Likar et al. [20] suggested a measure to estimate the overlap between two tissues $s_1, s_2$, called coefficient of joint variation, as

$$cjv(s_1, s_2) = \frac{\sigma(s_1) + \sigma(s_2)}{|\mu(s_1) - \mu(s_2)|}.$$

The coefficient of joint variation reflects the relation between $\Delta \mu(s_1, s_2)$ and the variance of $s_1, s_2$. Clearly, $cjv$ is small for well-separated tissues, and increases as $\Delta \mu(s_1, s_2)$ decreases. $Cjv(s_1, s_2)$ can be shown (using expressions similar to (45), (46)) to be invariant to both scalar multiplicative and additive intensity transformation and as such, can efficiently characterize the soft tissue overlap.

To evaluate $cv$ and $cjv$, preliminary classifications of soft tissues are desirable. Since this is a very time-consuming process, we used a combination of sources to obtain these classifications. These sources are provided in the next section.

Apart from image quality enhancement criteria described above, several other characteristics of MR correction algorithms examined in this study can be compared. As discussed in Section 1.5, our goal is to develop a fast and robust algorithm, so the comparison
criteria must also reflect this. To compare robustness, we tested dsf, spm and bcfcm on phantom images with different noise levels (Section 1.9.5). To determine comparative speeds, we measured their MATLAB code execution time (Section 1.9.7).

1.9.3 Testing Datasets

In this section, the MR datasets used in this study for the numerical evaluation of dsf, spm and bcfcm are described. These do not include over 1,000 datasets corrected by dsf that were evaluated visually.

1. Six phantom image volumes from the BrainWeb simulator [32] were used. These include T1, T2 and proton density (PD) weighted variations of normal and multiple sclerosis lesion 3D brain images. For all these images, the resolution is $181 \times 217 \times 181$ with a 1 mm slice thickness. Intensity of each pixel in these images is represented using 12 bits, providing 4096 shades of gray. A simulated multiplicative non-uniformity bias field $f(x, y)$ was chosen with a 40% variation, which means

$$0.8 \leq |f(x, y)| \leq 1.2.$$  

The random noise level defined for tissue $s$

$$\sqrt{\frac{1}{|s|} \sum_{(x,y) \in s} [n(x,y)]^2}$$

was fixed at 3%. In further references, these datasets are named T1N, T1L, T2N, T2L, PDN, PDl, where capital letters represent the pulse sequence and indexes N and L – normal and lesion brains accordingly (see Table 1.2). Additionally, five variations of the T1 normal brains with noise levels 0 %, 1 %, 3 %, 5 % and 7 % were used to compare the noise sensitivity of the correction algorithms. The soft tissue segmentations were extracted from original piecewise constant images using an intensity range match.
2. Six real T1 MR normal brain image sets (Sets 1-6 Table 1.2) and their manual segmentations provided by the Center for Morphometric Analysis at Massachusetts General Hospital, available through http://www.cma.mgh.harvard.edu/ibsr/. The spatial resolution for these sets ranges between 256 × 256 × 55 and 256 × 256 × 128, 8 bits per pixel.

3. Four real T1 MR brain image sets from different sources (Sets 7-10 Table 1.2), for which we performed the soft tissue segmentation manually.

4. Six MR and one RF image set of various body parts from different sources (Sets 11-17 Table 1.2) were also included.

1.9.4 Coefficient of Variation Evaluation

The coefficients of variation obtained in this study are shown in Table 1.2. To obtain a graphical interpretation of these results, we defined the normalized gradient for the coefficient of variation of tissue $s$:

$$ dcv(s) = \frac{cv(s_{cor}) - cv(s_{orig})}{cv(s_{orig})} \cdot 100\% , $$ (47)

where $s_{orig}$ represents tissue $s$ in the original image and $s_{cor}$ in the output image of correction algorithm. Using (47), we obtained the scatter plot of $dcv(WM)$ versus $dcv(GM)$ for $spm$, $dsf$ and $bcfcm$ (Figure 1.20).

We can also define the normalized gradient for the coefficient of the joint variation of two tissues $s_1$, $s_2$:

$$ dcjv(s_1, s_2) = \frac{cjv(s_{1,cor}, s_{2,cor}) - cjv(s_{1,orig}, s_{2,orig})}{cjv(s_{1,orig}, s_{2,orig})} \cdot 100\% . $$ (48)

The relative change in coefficients of joint variation resulting from the non-uniformity correction for sets 1-10 is plotted on Figure 1.21.

$Ds$ and $spm$ reduced the WM and GM coefficients of variation for all simulated sets.
Table 1.2. Datasets used for numerical comparison of dsf, spm and bcfcm.

<table>
<thead>
<tr>
<th>Set</th>
<th>Modality</th>
<th>Body part</th>
<th>Bits per pixel</th>
<th>Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1n</td>
<td>MR, T1</td>
<td>Normal brain</td>
<td>12</td>
<td>181 × 217 × 181</td>
</tr>
<tr>
<td>T1l</td>
<td>MR, T1</td>
<td>Lesion brain</td>
<td>12</td>
<td>181 × 217 × 181</td>
</tr>
<tr>
<td>T2n</td>
<td>MR, T2</td>
<td>Normal brain</td>
<td>12</td>
<td>181 × 217 × 181</td>
</tr>
<tr>
<td>T2l</td>
<td>MR, T2</td>
<td>Lesion brain</td>
<td>12</td>
<td>181 × 217 × 181</td>
</tr>
<tr>
<td>PDn</td>
<td>MR, PD</td>
<td>Normal brain</td>
<td>12</td>
<td>181 × 217 × 181</td>
</tr>
<tr>
<td>PDl</td>
<td>MR, PD</td>
<td>Normal brain</td>
<td>12</td>
<td>181 × 217 × 181</td>
</tr>
<tr>
<td>Set 1</td>
<td>MR, T1</td>
<td>Brain</td>
<td>8</td>
<td>256 × 256 × 63</td>
</tr>
<tr>
<td>Set 2</td>
<td>MR, T1</td>
<td>Brain</td>
<td>8</td>
<td>256 × 256 × 63</td>
</tr>
<tr>
<td>Set 3</td>
<td>MR, T1</td>
<td>Brain</td>
<td>8</td>
<td>256 × 256 × 59</td>
</tr>
<tr>
<td>Set 4</td>
<td>MR, T1</td>
<td>Brain</td>
<td>8</td>
<td>256 × 256 × 58</td>
</tr>
<tr>
<td>Set 5</td>
<td>MR, T1</td>
<td>Brain</td>
<td>8</td>
<td>256 × 256 × 61</td>
</tr>
<tr>
<td>Set 6</td>
<td>MR, T1</td>
<td>Brain</td>
<td>8</td>
<td>256 × 256 × 58</td>
</tr>
<tr>
<td>Set 7</td>
<td>MR, T1</td>
<td>Brain</td>
<td>8</td>
<td>256 × 256 × 109</td>
</tr>
<tr>
<td>Set 8</td>
<td>MR, T1</td>
<td>Brain</td>
<td>8</td>
<td>446 × 348 × 94</td>
</tr>
<tr>
<td>Set 9</td>
<td>MR, T1</td>
<td>Brain</td>
<td>8</td>
<td>255 × 223 × 108</td>
</tr>
<tr>
<td>Set 10</td>
<td>MR, T1</td>
<td>Brain</td>
<td>16</td>
<td>378 × 378 × 13</td>
</tr>
<tr>
<td>Set 11</td>
<td>MR</td>
<td>Chest</td>
<td>8</td>
<td>256 × 256 × 11</td>
</tr>
<tr>
<td>Set 12</td>
<td>MR</td>
<td>Abdomen</td>
<td>16</td>
<td>512 × 512 × 16</td>
</tr>
<tr>
<td>Set 13</td>
<td>MR</td>
<td>Chest</td>
<td>8</td>
<td>256 × 256 × 9</td>
</tr>
<tr>
<td>Set 14</td>
<td>MR</td>
<td>Heart</td>
<td>16</td>
<td>256 × 256 × 1</td>
</tr>
<tr>
<td>Set 15</td>
<td>RF</td>
<td>Knee</td>
<td>8</td>
<td>1024 × 1024 × 1</td>
</tr>
<tr>
<td>Set 16</td>
<td>MR</td>
<td>Lumbar spine</td>
<td>16</td>
<td>256 × 256 × 16</td>
</tr>
<tr>
<td>Set 17</td>
<td>MR</td>
<td>Shoulder</td>
<td>16</td>
<td>256 × 256 × 14</td>
</tr>
</tbody>
</table>

Figure 1.20. Performance comparison I. Scatter plot of $dcv$(GM) using (47) versus $dcv$(WM) for sets 1-10. One point for bcfcm in the left bottom corner is not shown to preserve the scale.
Table 1.3. Coefficients of variation of corrections performed by algorithms being compared. Each characteristic (GM coefficient of variation $cv(GM)$, WM coefficient variation $cv(WM)$, WM and GM coefficient of joint variation $cjv(WM, GM)$) measured for uncorrected volumes with $(src)$, $dsf$, $spm$ and $bcfcm$.

<table>
<thead>
<tr>
<th>Set</th>
<th>$cv(GM)$</th>
<th>$cv(WM)$</th>
<th>$cjv(WM, GM)$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>src</td>
<td>dsf</td>
<td>spm</td>
</tr>
<tr>
<td>T1$_N$</td>
<td>10.5</td>
<td>9.1</td>
<td>9.4</td>
</tr>
<tr>
<td>T1$_L$</td>
<td>11.0</td>
<td>9.5</td>
<td>9.8</td>
</tr>
<tr>
<td>T2$_N$</td>
<td>12.8</td>
<td>9.9</td>
<td>11.7</td>
</tr>
<tr>
<td>T2$_L$</td>
<td>18.1</td>
<td>12.8</td>
<td>9.6</td>
</tr>
<tr>
<td>PD$_N$</td>
<td>7.0</td>
<td>5.3</td>
<td>3.4</td>
</tr>
<tr>
<td>PD$_L$</td>
<td>7.5</td>
<td>5.2</td>
<td>3.5</td>
</tr>
<tr>
<td>Set 1</td>
<td>17.3</td>
<td>14.1</td>
<td>13.1</td>
</tr>
<tr>
<td>Set 2</td>
<td>28.8</td>
<td>28.8</td>
<td>29.4</td>
</tr>
<tr>
<td>Set 3</td>
<td>19.8</td>
<td>19.6</td>
<td>20.2</td>
</tr>
<tr>
<td>Set 4</td>
<td>57.1</td>
<td>55.8</td>
<td>56.9</td>
</tr>
<tr>
<td>Set 5</td>
<td>38.8</td>
<td>38.7</td>
<td>41.4</td>
</tr>
<tr>
<td>Set 6</td>
<td>15.2</td>
<td>15.1</td>
<td>15.6</td>
</tr>
<tr>
<td>Set 7</td>
<td>12.3</td>
<td>11.5</td>
<td>10.9</td>
</tr>
<tr>
<td>Set 8</td>
<td>14.0</td>
<td>13.5</td>
<td>13.1</td>
</tr>
<tr>
<td>Set 9</td>
<td>21.5</td>
<td>21.5</td>
<td>23.2</td>
</tr>
<tr>
<td>Set 10</td>
<td>12.9</td>
<td>12.0</td>
<td>13.1</td>
</tr>
</tbody>
</table>

On authentic datasets, $dsf$ reduced or produced the same coefficient of variation for the GM in all cases and for the WM in 80% of the cases. The other two algorithms, $spm$ and $bcfcm$,
reduced or produced the same GM coefficient of variation in 40% and 90% respectively for authentic datasets, and reduced the WM coefficient of variation in 70% and 90%.

Since *spm* does a pre-segmentation of the brain image volume for white matter as a first approximation, the results of the bias field extrapolation from white matter to the entire image must be very accurate to produce a consistent correction. *Spm* can be expected to achieve good results in correcting white matter inhomogeneity, but the improvement for the entire image depends on error of WM bias field estimation, extrapolation method used and the correctness of an assumption that the bias can be extrapolated from the white matter to the entire image volume. *Spm*’s GM correction in 70% of authentic cases decreased modestly and sometimes even increased the GM variation coefficient. With the same authentic data, the WM coefficient of variation was reduced in most cases, which points to an incoherency in estimating the total bias field.

The *bcfcm* algorithm produced corrections with a significantly decreased contrast between white and grey matter distributions in 94% of the cases, which raises a question of whether it really improved those datasets. *Dsf* improved T1 phantoms as well as *spm*, although its resulting bias corrections for T2 and PD were somewhat smaller. However, it uniformly improved both WM and GM coefficients of variation by not introducing any additional non-uniformity. *Spm* was particularly unstable on both the high and very low non-uniformity image datasets and required a large number of iterations (up to 60) on Sets 3, 6, and 9, requiring up to 20 minutes, which is not acceptable in actual use.

1.9.5 Sensitivity to Noise

With authentic MR images, the signal-to-noise ratio can vary from scanner to scanner and it depends on the acquisition pulse sequence and any factors that may be present. It is, therefore, important to evaluate the performance of any correction algorithm on images with randomly changing noise parameters. For such an evaluation, we applied *spm*, *dsf* and *bcfcm*
to the simulated normal T1 image volume with 20% non-uniformity and five different levels of random noise (Section 1.9.3). We compared the difference between the GM coefficient of variation for the corrected volume and original volume; these results are shown on Figure 1.22.

![Figure 1.22](image)

**Figure 1.22.** Sensitivity to noise. Solid curve: $cv_b(GM) - cv_0(GM)$; dash-and-dot curve: $cv_{spm} - cv_0(GM)$; dashed curve: $cv_{dsf}(GM) - cv_0(GM)$. $cv_0(GM)$ represents GM coefficient of variation of an unbiased noisy source, $cv_b(GM)$ of a biased noisy source, $cv_{dsf}(GM)$ and $cv_{spm}(GM)$ – of the correction produced by dsf and by spm, respectively.

Bcfcm results are not shown on Figure 1.22 because they were too erratic and would interfere with the graph scale used. As shown, dsf correction was stable even with high levels of random noise. This is due to the smooth bias field model that is not sensitive to a signal of higher frequency, and to the use of noise-canceling six-point partial derivative approximations (33), (34).

### 1.9.6 Other Body Parts

As discussed in Section 1.5, many MR correction algorithms were designed specifically for brain images, and the majority of evaluations in the literature has been performed on brain images. Since our goal was to develop a more general correction
Figure 1.23. Visual evaluation of corrections. Comparative correction results for sets 11-16 are shown: a – original image, b – corrected with our method, c – corrected with *spm*, d – corrected with *bcfcm*. 
algorithm, dsf was designed for any MR image, regardless of the body part. Dsf even produces good results on non-MR images if they have the same slow varying multiplicative pattern of inhomogeneity and areas of well defined homogeneous tissue. These include, for example, many CT chest images.

Since UniViewer software was installed in a number of locations, we were able to apply dsf to over 1000 images (of MR and other modalities) from different sources, and observed a visual improvement on most of these images. For analysis, we randomly chose several image datasets of various body parts corresponding to sets 11-17 in Table 1.2.

A comparison of visual results for the three methods evaluated is shown on Figure 1.23. On these image datasets, spm removed some non-uniformity in sets 13 and 16, but introduced additional non-uniformity in sets 14 and 15. Spm also tended to reduce the higher intensities in the image, which in combination with high output contrast result in a loss of anatomical information. Bcfcm produced a visible decrease in the tissue contrast and removed some anatomical detail, which was typical for all datasets corrected by this algorithm in this study. Dsf reduced non-uniformity in all the image datasets, although an extra algorithm pass might rarely be required; it did not result in any visual loss of anatomical detail.

1.9.7 Execution Times

Since dsf has been incorporated into UniViewer, it performs the correction of $256 \times 256 \times 200$ MR images in a “volume” in 17 seconds with a fixed correction time. Dsf clearly is considered to execute fast enough to process high volumes of data. However, since fast implementations of bcfcm and spm were not available, we cannot make any conclusions about their performance. To compare the execution times of these three algorithms, we ran them in MATLAB on Pentium IV 3.06 MHz PC. The relation of the execution times using a more efficient implementation may vary.
We randomly chose four image datasets and measured the execution times of each of the three correction algorithms (Table 1.4). *Dsf* was designed to use a single pass, *spm* and *bcfcm* had to do several passes (iterations). *Spm*'s iterations varied in different image datasets between 5 and 80.

**Table 1.4. Running times for evaluated algorithms on four datasets.**

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Execution time in MATLAB, min:sec</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><em>dsf</em></td>
</tr>
<tr>
<td>Set 1</td>
<td>1:30</td>
</tr>
<tr>
<td>Set 12</td>
<td>2:40</td>
</tr>
<tr>
<td>Set 14</td>
<td>2:55</td>
</tr>
<tr>
<td>Set 15</td>
<td>0:56</td>
</tr>
</tbody>
</table>

1.10 Discussion and Conclusions

The method presented in this chapter is based on the assumption that the MR bias field is multiplicative, smooth and slow varying and the partial derivatives of an MR image can be approximated by its corresponding partial derivatives of the modeled bias field. Surface fitting of the bias field using a polynomial basis guarantees smoothness and stability of a modeled bias signal; the selection of basis functions was justified by their computational properties. A similar basis model described in [22] uses Legendre polynomials; however, our method was more stable using a simpler polynomial basis.

Widely used statistical methods based on the EM algorithm, such as in [13] and [2], use intermediate low-pass filtering after each iteration to reduce computation error and produce a smoother bias field. Due to the nature of the bias model, *dsf* produces a bias field estimate that is always smooth, noise insensitive and not affected by local image distortions. Our model also does not require prior knowledge of the tissue intensity distribution. The only input parameter for our algorithm is the percentile value $b$ that characterizes the high frequencies in the input MR signal. This parameter was found to be different for the images
of different body parts and modalities, but \( dsf \) performed consistently with fixed \( \beta \) on the images of the same body part obtained from different sources.

For this study, we applied \( dsf \) and two previously published methods, \( spm \) and \( bcfcm \), to synthetic images, six simulated image volumes from BrainWeb [32], ten authentic brain image volumes from Massachusetts General Hospital Center for Morphometric Analysis [34] and other sources, and six datasets of other body parts. Several parameters were estimated: variation coefficients for WM and GM, visual quality of correction, sensitivity to random noise in data and running times. \( Dsf \) decreased the WM and GM coefficients of variation for most of the brain datasets and was robust. \( Spm \) in several cases decreased the WM coefficient of variation more than \( dsf \), but was also less stable and introduced additional non-uniformity in several cases, especially for grey matter. Both \( dsf \) and \( spm \) performance was not affected by increasing the random noise in simulated datasets. Our algorithm visually improved 100% of six image datasets (52 images in total) of other body parts (one of them was not a MR modality); both \( spm \) and \( bcfcm \) were less stable on these and appeared to remove anatomical structures from original data. Due to its non-iterative design, \( dsf \)’s running time depends only on the image data size, whereas \( spm \)’s number of iterations ranged between 5 and 80 and was hard to predict. The \( bcfcm \) algorithm appeared to significantly reduce soft tissue contrast and remove anatomical detail in practically all examined image datasets.

The speed of our algorithm makes it particularly useful for real clinical applications. Many accurate MR correction algorithms usually use iterations to approximate a solution, and each step often requires advanced computation. Long computation times make it difficult to use many current MR correction algorithms in clinical software. Our method uses a single iteration (although it can be applied repeatedly) and was initially designed to be computationally efficient. Using the implementation described in Section 1.7.7, the correction of 256x256x200 MR brain image volume takes about 17 seconds on a current PC. Due to
short running times, our algorithm can also be used as a quick image “fix” and/or a pre-
processing for volume segmentation or rendering procedures.
Chapter 2  Automated Medical Image Volume Rendering

2.1  Introduction

The visualization of medical image volumes has become a large growing research area. It has attracted many researchers throughout the world. Visualization has its roots in the 1960’s and 1970’s, when the theoretical foundation for 2D image processing was established. However, intensive investigation of volumetric image generation from medical data did not begin in earnest until the second half of the 1980’s. One reason for this increase was a demand to find efficient methods for 3D image processing and visualization. The rapid development of medical imaging equipment produced large three-dimensional volumes of digital medical images. Traditional methods of the manual scan-by-scan examination by radiologists are tedious on the sets containing hundreds of high resolution images. Automation is required to rapidly extract useful features from this ocean of data. On the other hand, the automatic processing of volumetric data requires computational power that is proportional to the cube of the volume’s linear dimensions. In any case, there is a minimum time threshold for volume image processing, which when not achieved, automatic volumetric visualization remains mainly within the bounds of pure theory, i.e. it will not be used in a production environment. Fortunately, this threshold has been achieved in recent years, and this allowed real world clinical applications using 3D-visualization. The rapid growth of medical image volume visualization has given rise to numerous methods for volume rendering, which can be broadly classified into surface and direct volume rendering. Surface rendering generates and displays the object’s boundary surface and need to be generated first. Direct volume rendering uses every volume element in the generation of the 3D scene.
2.2 Medical Image Volumes

There are two different approaches to the representation of the smallest data element in a discretized image volume. This is due to the duality of a discrete input signal:

(i) it can be considered a set of samples along a certain grid from the continuous object;

(ii) each sample can be considered an average value over a certain area.

This duality is analogous to one in the physics of quantum theory: according to Heisenberg’s uncertainty principle, either the exact position of a particle or the exact time in this position can be measured precisely, but not both simultaneously. The “uncertainty principle for a discrete signal” may be formulated: signal magnitude (interpreted as intensity for images) and its exact spatial position (the corresponding point) in a volume cannot both be determined precisely at the same time.

A sample from a medical image volume is frequently interpreted in terms of (ii), as an averaged signal over a small subvolume. This is justified by the nature of the medical image acquisition process (described for MR images in sections 1.1 – 1.3). In accordance with (ii), medical image volume $V$ is a set of three-dimensional elements (voxels) $v_i$ that can be interpreted as material points each having a color $c(v_i)$. The domain of a voxel’s color function $c(v)$ depends on the input signal interpretation used to represent the volumetric data. Sometimes in volume rendering, it is necessary to treat voxels according to (i), to calculate local image characteristics such as its gradient. With (i), $v_i$’s are understood to be samples of a volumetric function $F(x, y, z)$ on a regular grid (knots on a regular grid in the Cartesian coordinate system are spaced evenly along the coordinate axes). The majority of medical image data is grayscale with rare exceptions (such as certain ultrasound images), and in this research we only consider grayscale images. For such images, $c(v)$ has an intensity value $0 \leq c(v) \leq 2^n - 1$ for $n$ bits per pixel.
The isosurface $\partial V|_c$ for a given volume $V$, intensity $c_0$ and fixed small $\epsilon_0$, it is understood to be a two dimensional surface having non-empty intersections with all voxels with a given intensity range:

$$v_i \in \partial V|_c \iff v_i \cap \partial V|_c \neq \emptyset \text{ and } |c(v_i) - c_0| < \epsilon_0.$$  \hfill (49)

This isosurface represents the “layer” in the volumetric object that contains the elements with intensities close to a fixed value $c_0$. It follows from this definition that there are an infinite number of ways to construct the isosurface for a fixed volume $V$ and color $c_0$, so it is better to consider a class of isosurfaces defined by $c_0$. In terms of a functional $F$, this isosurface is a representation of its surface value $F(x, y, z) = c_0$, and any uncertainty in its definition follows from the missing information related to the values of $F$ between the grid knots.

### 2.3 Surface Rendering

One of the two approaches used in surface rendering of 3D volumes is isosurfacing. The classical method of isosurface extraction, the Marching Cubes Algorithm, was proposed by Lorensen and Cline [43] and Wyvil et al. [44]. Each voxel is considered the topological equivalent of a cube, and the planar approximation of the isosurface $F(x, y, z) = c$ within this cube is sought. To calculate intersections of $\partial V|_c$ with the cube edges, voxel trilinear interpolation is used. “Each vertex of a cube can be either greater than or less than the threshold value, giving 256 different scenarios [40].” Each of these 256 configurations represents one or more triangles constituting the isosurface within the voxel of interest, and every configuration is stored in a look-up table for subsequent fast access. This analysis of each voxel in the volume produces a triangulated isosurface. To create the resultant 3D image, every triangle for each voxel is rendered according to the selected lighting model. After this triangulation is constructed, rendering can be fast enough (especially using
graphics hardware to render triangles) to provide acceptable interactive rates of rotation and scaling operations for the resultant 3D image.

2.4 Surface Rendering and Isosurface Representation

How does the isosurface correlate with the medical image volume visualization? In other words, what conditions should the original data satisfy for the extracted isosurface to represent meaningful “structures” to the clinician? To answer this question, let’s consider MRI data, which is a good example for the typical volume rendering source data. Under certain pulse sequences, the intensities of soft tissues in MR images can be statistically separated to model the MR image as a piecewise constant function multiplied by a slowly varying bias field (see Chapter 1). The application of a threshold which is close to one of the soft tissue intensities would lead to a noisy and sometimes meaningless set of partially disjoint voxels, with some of the soft tissue voxels belonging to the isosurface and some not (Figure 2.1).

Figure 2.1. Intensity thresholding. 2D MR image (left) and the part that would be part of the isosurface if intensity thresholding is applied (right).
From this example, it is intuitively clear that to represent a boundary for a subvolume $V_0 \subset V$, all voxels $v_i^0 \in V_0$ should also have intensities either higher or lower than the specified threshold $c_0$. Therefore additional voxel intensities that do not belong to $V_0$ should be on the higher/lower side of $c_0$, i.e. for all $i, j, v_i^0 \in V_0, v_j \in V \setminus V_0$, one of inequalities

$$\begin{align*}
v_j < c_0 & \leq v_i^0, \\
v_j > c_0 & \geq v_i^0
\end{align*}$$

(50)

should always hold. This partitioning is often possible for the task of separating the object from its background, and the isosurface determination should allow creation of the entire object outline, such as the MR head isosurface presented on Figure 2.2. Other examples of such separations are provided in Section 2.8.2.

![Figure 2.2. MR head source data (left) and the isosurface constructed by applying a background threshold (right)](image)

Since clinicians are often interested in internal organs and structures, the direct applicability of this isosurfacing is not possible in all cases, and other considerations involving additional implicit knowledge about the input must also be used to construct boundary surfaces. This requires additional preprocessing to convert the original volume to a form where the construction of an isosurface using (50) is applicable. Detailed classification of such cases is beyond the scope of this research.
2.5 Volume Rendering

Volume rendering methods can be essentially classified into two groups:

1. **Image order** and
2. **Object order**.

“In the image order approach (also called backward rendering), processing is done from the image plane to the volume. In the object order approach (also called forward rendering), processing is done from volume to image [40]”. The classical image order approach (described in Levoy et al. [38]), considers the volume as a “cloud” of particles, each of which absorbs a certain amount of the light that goes through it. The density \( \mu \) of particles varies throughout the volume, and the amount of light received by the image plane is obtained by the following integral (according to notation used in [42]):

\[
I = \int_0^L C(s)\mu(s) e^{-\int_0^s \mu(t)dt} ds
\]

(51)

where \( L \) is the length of the light ray and \( C(s) \) is the amount of light reflected at location \( s \) along the ray. In many applications, (51) is approximated by a discrete integral sum, and every method that involves calculation of \( I \) has a compromise between higher accuracy of the 3D representation of an object and the greater speed of rendering achieved by using less samples along the ray.

The typical object order approach, using the “splatting” technique, was described by Westover [45, 46]. Voxels in the volume are essentially projected onto a viewing plane, forming splats, for further composition in the image plane. The splatting algorithm orders the voxels in the target volume in such a way that for a given scene, the voxels nearest the observer are always processed first. Then each voxel is projected into a viewing plane using a smoothing filter to determine its image space occupation, and this is blended with previously projected voxels using transparent color blending. The object order methods convert voxels
directly to geometric objects on viewing plane, and for that reason are also called direct methods.

In direct rendering methods, an isosurface is determined by “rendering opaquely all voxels with values greater than some threshold [38].” The voxels are converted directly to geometric objects on a viewing plane without intermediate steps. When a surface is created, abrupt thresholding sometimes creates the surface with gaps or holes which obviously affects the visual quality and introduces artifacts. To make the computer generated 3D image look more natural, transferring between the imageable voxels \( v \in V \) and the remaining voxels \( r \in R \) is accomplished by limiting the value of the intensity gradient within the locality \( N(\partial V) \) of the boundary isosurface \( \partial V \). This technique is sometimes called fuzzy or shell thresholding [35]. This should not be confused with the volume rendering techniques which employ in-depth volume analysis to obtain semi-transparent images based on transfer functions [39]. Another problem with direct methods is their increased computational complexity, since every element in the volume is being processed for every new 3D scene generation. For instance, volume rendering [35] requires \( 6m^2n^4 \) calculations for \( m \) samples in each dimension of \( V \) and \( n^2 \) pixels in the resultant image \( I \).

Recent advances in direct methods, as well as increased processor speeds, have reduced the generation time for one 3D image frame from several minutes to fractions of a second. Software rendering algorithms have not yet achieved a combination of high quality 3D images with really usable interactive rates (the “holy grail” of volume rendering research). On the other hand, hardware graphics cards supporting different rendering approaches (such as VolumePro ray casting hardware described in a landmark paper by Pfister et al. [41]) have advanced interactive frame rates. Hardware rendering, however, inherently has the drawback of its low flexibility in its choice of rendering algorithms and the advantages they may offer.
2.6 Rendering Based on Boundary Voxels of a Segmented Volume

The problem of fast 3D object rendering has been addressed in [36]. If the 3D image is segmented into two voxel sets: a volume of interest $V$ and the remaining volume $R$, it is possible to compute the set of “boundary voxels” $B$, such that for every voxel $v$ belonging to $B$, its neighborhood $N(v)$ contains voxels from both $V$ and $R$. The elements from $V$ without its boundary (i.e. internal voxels, denoted $V\setminus B$) will be obscured by voxels from B. Thus, the problem of rendering the entire volume $V$ is reduced to the rendering of its boundary voxels $B$.

In order to construct the 3D image, voxels from $R$ are projected onto a view plane and rendered using one of three solutions [36]:

1. computing a voxel projection $p(v)$ onto a view plane and rendering the polygon obtained; this can result in aliasing, gaps between different polygons, “black holes”, etc.;
2. “generous” area fill such as circular [37];
3. using a scale factor small enough to fit every voxel projection into one pixel. This method, though, cannot be used on small datasets, which poses the major challenge to render.

Bullitt et al.[37] suggest representing each voxel $b \in B$ as a sphere $s_b$ for drawing purposes. If $s_b$ is such that $b \cap s_b = b$, its projection on the viewing plane is a circle and can be easily rendered. Two neighboring voxels are projected into partially overlapped circles. In this way, the computationally expensive calculation of a cubic voxel projection is avoided. The final scene is produced using a Z-buffer: each pixel inside the circle is assigned a “depth,” which is compared to the “depth” of the previously drawn at that position, and the resulting intensity is chosen from the pixel with minimal “depth”.
2.7 The Problem of Efficient Rendering

Volume and surface rendering methods are numerous, and the scope of this research does not allow an exhaustive review. For those interested in elaborate and detailed classification, the article by Ken Brodlie and Jason Wood [40] can be of help. Our purpose here was to illustrate two major approaches to volume visualization; based on this illustration, justification of our new rendering method can be presented.

Our final goal is to automate the process of medical image volume visualization and develop an efficient tool for its actual real-time use. We have seen that volume visualization can be performed in many ways, and thus a more specific definition of efficiency is needed. Efficient medical image volume visualization will be henceforth understood to possess the combination of the following properties:

1. Adequate isosurface rendering quality: This includes smoothness, minimal artifacts, and clear lighting.

2. Real time rendering speed: The rendering of the complete 3D scene should not take more (and preferably much less) than 1 second on a current “off-the-shelf” PC. The efficient visualization provides both acceptable rendering speed and rendering quality, even though the tradeoff between them must always be made.

3. Flexibility: Since many internal structures with different intensity characteristics are of interest to the clinician, either all volume parts should be easily identifiable or exposing each part should not require clinically unacceptable processing times. Therefore, visualizing any isosurface defined within the source 3D volume should not require too much overhead.

Under this efficiency requirement, the choice of basic algorithms to produce visualization is simplified. Rendering the entire volume is (in general) slower than surface rendering due to the number of computations (51), and therefore an isosurface method is
preferable. However, obtaining the isosurface involves substantial pre-processing (e.g. “marching cubes” [43] analyzes from 16 to 256 configurations for each voxel) and changing an intensity threshold may also require substantial time, thus compromising our flexibility requirement. The hybrid approach, based on volume segmentation [35-37] can produce a new frame for rendering quickly \(O(n^3)\), where \(n\) is the maximum linear dimension of source volume). This direct rendering approach involves only a small fraction of voxels that constitute the integral (51), and is therefore much faster. Using these advantages, we will develop an efficient automatic medical image volume visualization algorithm.

2.8 Pre-segmentation of a Medical Image Volume

As discussed in Section 2.6, before a fast rendering can be applied to a volume, the area of interest should be segmented. For each particular body part (e.g. organ), numerous algorithms have been developed to detect a particular organ and/or tissue. For instance, Ray et al. [47] implement active contours for MRI lung segmentation, Liew et al. [23], Zhang [14] et al. and Wells et al. [13] use fuzzy clustering and statistical methods to segment brain images into white matter, grey matter and cerebrospinal fluid. The main principle for this variety of methods is to obtain an initial intensity range and spatial position of the organ of interest. Most efficient current methods usually require some manual processing and may not be used directly in any automated visualization system. However, the isosurface (49) that corresponds to a selected intensity range may serve as a good approximation for certain structures. In the following sections, we will demonstrate that such visualization can be completely automated.

2.8.1 Isosurface Range Selection

In Chapter 1, an MR image representation was developed as a function which is piecewise constant on most of its domain and behaves otherwise on remaining fraction of
domain. For brain images, large piecewise constant areas correspond to white matter, grey matter and cerebrospinal fluid tissues; some other structures like bone in computed tomography (CT) chest images can be also identified using intensity thresholding. In this section, we present a method for an automated detection of homogeneous tissue ranges from the image histogram.

As discussed in Section 1.8.3, homogeneous tissue produces a peak in the image histogram. It follows directly from the histogram definition (4) from Section 1.4.4 that the peak height for the image intensity \( i H(i) \) depends linearly on the area of homogeneous tissue. Let \( N_I \) be the number of tissue classes \( S_1, \ldots, S_{N_I} \) in the image \( I(x, y) \), and the mean intensities of these classes \( I(S_i) \) are known to form an increasing sequence:

\[
I(S_1) < I(S_2) < \cdots < I(S_{N_I}).
\]  

Then, the problem of detecting the mean intensities \( I(S_i) \) for \( N_I \) classes in image \( I \) takes the form of finding the \( N_I \) highest local maximums of the image histogram. Normally, the peaks corresponding to white matter and grey matter are visually identifiable. However, the accurate automatic detection of these peaks is difficult due to their lack of definition, and a filtering scheme needs to be developed. As a first step, we suggest removing a resonant frequency (“palisade”) effect. This effect is produced by the acquisition pulse sequence: certain intensities are more frequent than neighboring ones, which results in a “palisade” clearly visible on Figure 2.3, left. To overcome this, we can use a smoothing edge preserving filter defined by (26). This introduces smoother intensity transitions for neighboring points, thus reducing the “resonant” frequencies significantly (Figure 2.3, right). After Gaussian histogram filtering is finished, peak frequency extraction can be done.

Examining numerous MR image histograms has led us to an observation that in many real applications the histogram peaks have a shape that resembles a reversed parabola. If a
sufficiently large local subsegment is selected, a parabola approximation of $H$ can be produced using least squares. The reverse parabola indicates a local maximum; the parabola’s curvature in the point of the maximum depends solely on its closeness to the maximum point and therefore can serve as a characteristic of the maximum. Illustration of relation between parabola curvature (denoted as $a_2$) and maximum points on the approximation segment are presented on Figure 2.4.

After this informal description, we will now develop this idea mathematically.

Consider the local subsegments

$$s_j = \left[ j \quad j + \delta \right] \in [I_{\min}, I_{\max}], \quad j = I_{\min}, \ldots , I_{\max} \quad (53)$$

Without limiting generality, we can assume $I_{\min} = 0$. To approximate $H(x)$ on $s_j$ with a parabola $P_j$

$$P_j(x) = a_2 x^2 + a_1 x + a_0,$$

the least squares sum function $F$ is composed:

$$F(x_j) = \sum_{x=j}^{j+\delta} \left[ H(x) - \left( a_2 x^2 + a_1 x + a_0 \right) \right]^2.$$

The equations for local minimum are constructed by setting derivatives with respect to $a_2$, $a_1$ and $a_0$ to zero:
\[
\frac{\partial F}{\partial a_2} = -2\sum_{x=j}^{i+8} [H(x) - a_2 x^2 - a_1 x - a_0] x^2,
\]
\[
\frac{\partial F}{\partial a_1} = -2\sum_{x=j}^{i+8} [H(x) - a_2 x^2 - a_1 x - a_0] x,
\]
\[
\frac{\partial F}{\partial a_0} = -2\sum_{x=j}^{i+8} [H(x) - a_2 x^2 - a_1 x - a_0]
\]

Figure 2.4. Approximation of a histogram segment by parabola. Approximation segments (top row) and corresponding parabola fits (bottom row). The negative curvature of parabola is insignificant at the absence of local maximum (left, \(a_2 = 0.07\)) and increases sharply in the vicinity of the point of maximum (right, \(a_2 = -100\)).

This can be rewritten in a matrix form

\[
\begin{bmatrix}
X_j^4 & X_j^3 & X_j^2
\end{bmatrix} \begin{bmatrix}
a_2
\end{bmatrix} =
\begin{bmatrix}
H_j X_j^7
\end{bmatrix},
\]
\[
\begin{bmatrix}
X_j^3 & X_j^2 & X_j^1
\end{bmatrix} \begin{bmatrix}
a_1
\end{bmatrix} =
\begin{bmatrix}
H_j X_j^1
\end{bmatrix},
\]
\[
\begin{bmatrix}
X_j^2 & X_j^1 & X_j^0
\end{bmatrix} \begin{bmatrix}
a_0
\end{bmatrix} =
\begin{bmatrix}
H_j X_j^0
\end{bmatrix}
\]

where

\[
X_j^p \equiv \sum_{x=j}^{i+8} x^p
\]
and

$$H_jX^p \equiv \sum_{x=j}^{j+\delta} H(x)x^p. \quad (56)$$

The unknowns $a_2 = a_2(j), a_1 = a_1(j), a_0 = a_0(j)$ from (54) define the best in a least squares sense, parabolic approximation $\hat{P}_j(x)$ of $H(x)$ on the segment $s_j$.

Sweeping through the discrete domain of $H(x)$ we obtain a family of parabolas

$$\mathcal{P} = \hat{P}_0(x), \hat{P}_1(x), \ldots, \hat{P}_{\text{max}}.$$ 

In the areas where $j > I_{\text{max}} - \delta$ or $j < 0$, extrapolation of $H(x)$ is used to obtain the values beyond its domain. For increased accuracy, quadratic or cubic splines can be used with boundary conditions on $H(x)$ that include first and second order derivatives. In our experience, the extrapolation accuracy is not critical, and linear or even constant extrapolation produces satisfactory results.

**Figure 2.5.** Result of parabola histogram filtering. Smoothed MR image histogram (top) and resulting $A(j)$ (bottom).

To complete the construction of a robust maximum filter, it is sufficient that $a_2(j)$ equals the curvature of the parabola and the local maximum of its negative occurs when the
best fit of the parabola shape and local $H(x)$ is achieved. At the same time, the areas where $a_2(j) > 0$ do not contain a local histogram peak. Thus, the local maximums of the function

$$A(j) = \begin{cases} -a_2(j), & a_2(j) < 0 \\ 0, & \text{otherwise} \end{cases}$$  \tag{57}$$

provide an approximation to the histogram peaks. Since $A(j)$ is obtained from averaging segment $s_j$, local histogram irregularities do not affect it, and its local maximums can be identified automatically (details are provided in the algorithm description below). See Figure 2.5 for an example of histogram peak extraction.

With this mathematical foundation, it is possible to describe an algorithm for the extraction of histogram peaks.

**Input:** Number of classes in the image $N_I$.

**Initialization:** Calculate image histogram function $H(x)$ using (4). Calculate the length $\delta$ of the discrete segment $s_j$. The length of $s_j$ varies depending on the number of peaks in the histogram. On one hand, small $\delta$ leads to detection of minor peaks due to fluctuations in intensity within homogeneous tissue and therefore does not provide the accurate intensity mean value; on the other hand, larger $\delta$ leads to a lack in resolution of local maximums of $A(j)$. We can assume that the length of the sampling segment should be proportional to the intensity range and inversely proportional to the number of classes $N_I$. We achieved optimal quality in most cases with

$$\delta = \frac{I_{\max}}{2N_I}.$$  \tag{58}$$

**Step 1:** for all $j$ from 0 to $I_{\max}$, calculate $A(j)$ defined by (57). Parabolic coefficient $a_2$ is found with (54) using (55) and (56).

**Step 2:** for all $j$ from 0 to $I_{\max}$, detect connected segments $s_{\text{conn}}^1, \cdots, s_{\text{conn}}^m$ that constitute the support of $A(j)$.
\[ j \in \sup A(j) \Leftrightarrow A(j) > 0. \]

A one-dimensional point \( j \) belongs to a connected discrete segment \( s_{\text{conn}} \) where \( s_{\text{conn}} \) contains more than one point, if one of its immediate neighbors \((j + 1 \text{ or } j - 1)\) belongs to \( s_{\text{conn}} \).

Therefore, a simple nearest neighbor examination (trivial for one dimension) would allow generating all connected segments.

Each connected segment \( s^i_{\text{conn}} \in \sup A \) contains one local maximum corresponding to a large histogram peak. Indeed, there should be at least one maximum since \( A(j) = 0 \) on the boundaries of \( s^i_{\text{conn}} \) and \( A(j) > 0 \) inside \( s^i_{\text{conn}} \). On the other hand, if the peak is significant and corresponds to a large area in the image, it is separated from the other peak by the area where \( A(j) = 0 \) due to selection (58) of the approximation segment length. Figure 2.5 illustrates this proposition.

**Step 3:** Calculate the maximums on each connected segment:

\[ H^{\text{max}}_i = \max_{x \in s^i_{\text{conn}}} H(x), \quad i = 1, \ldots, m \]

and sort them to form the non-decreasing sequence:

\[ \hat{H}^{\text{max}}_1 \geq \hat{H}^{\text{max}}_2 \geq \cdots \geq \hat{H}^{\text{max}}_m. \]

The array obtained forms the sequence of histogram peaks that correspond to the intensity uniformity areas (tissue classes) \( S_1, \cdots, S_m \) ordered by occupancy area in the image. That is, \( S_1, \cdots, S_m \) correspond to tissue classes that are present in the image, from the most to least defined. Therefore, the first \( N_I \) members of this sequence match the \( N_I \) best defined tissue classes, as described by (52). Therefore, the final output of the algorithm is \( S_1, \cdots, S_{N_I} \).
2.8.2 Automated Selection of Intensity Ranges

To produce a consistent segmentation, the automatic selection of intensity ranges \([s_i^{\text{min}}, s_i^{\text{max}}]\) for each tissue \(S_i\) (52) is needed. Regardless of selection method, a portion of the intensities \(0 \leq \rho \leq 1\) between the two tissues \([\bar{I}(S_i), \bar{I}(S_{i+1})]\) used for classification should be defined. Without limiting generality, we assume this portion to be equal \(\rho\) for all \(S_i\). In the simplest case the limits of \(S_i\) are defined as:

\[
\begin{align*}
    s_i^{\text{min}} &= \bar{I}(S_i) - \frac{1}{2} \rho \left[ \bar{I}(S_i) - \bar{I}(S_{i-1}) \right] = \bar{I}(S_i) \left[ 1 - \frac{1}{2} \rho \right] + \bar{I}(S_{i-1}), \\
    s_i^{\text{max}} &= \bar{I}(S_i) \left[ 1 - \frac{1}{2} \rho \right] + \bar{I}(S_{i+1}).
\end{align*}
\]

(59)

Varying \(\rho\) allows the extraction of tissues with different mean intensity variations. We found \(0.6 \leq \rho \leq 1\) to produce best results for brain images. From Figure 2.4 and Figure 2.5 it is clear that different tissues may have different mean variations (the “widths” of histogram peaks), and therefore borders defined by (59) may not always be optimal. A simple observation that the length of connected support for \(A(j) s_i^{\text{com}}\) corresponding to \(S_i\) is proportional to the “width” of the histogram peak allows the adjustment of (59) to be

\[
\begin{align*}
    s_i^{\text{min}} &= \bar{I}(S_i) \left[ 1 - \frac{1}{2} \frac{s_i^{\text{com}}}{s_{i-1}^{\text{com}}} \rho \right] + \bar{I}(S_{i-1}), \\
    s_i^{\text{max}} &= \bar{I}(S_i) \left[ 1 - \frac{1}{2} \frac{s_i^{\text{com}}}{s_{i+1}^{\text{com}}} \rho \right] + \bar{I}(S_{i+1}),
\end{align*}
\]

(60)

which allows an adequate correspondence between “widths” of neighboring histogram peaks.

The resulting algorithm, named Automatic Tissue Detection (atd), was executed on a number of images to validate the proposed method. In all cases, filtered intensity peaks
Figure 2.6. Automatic brain image segmentation.

Figure 2.7. Automatic chest image segmentation.
showed a close match with histogram peaks. Figures Figure 2.6 and Figure 2.7 illustrate the automated segmentation of a brain image with \( N_I = 3 \), and a chest image with \( N_I = 4 \).

2.9 Gravitational Shading Algorithm

The automatic algorithm developed in the section 2.8, extracts homogeneous tissues \( S_1, \cdots, S_N \) from an image. These input data may now serve as a foundation for an efficient (in terms of Section 2.7) 3D rendering algorithm.

Since direct rendering methods provide more flexibility in displaying different intensity ranges, a direct rendering (with each voxel projected onto a viewing plane) is preferable. We discussed in Section 2.7 that direct calculation of the integral (51) is time-consuming, but the knowledge of pre-segmented structures in the volume permits the reduction of the number of calculations along each ray; described in Section 2.6. If each voxel is represented by a sphere, a fast 3D rendering can be performed using this scheme. However, the quality of such a rendering (obtained by overlapping circles over the viewing plane) seriously degrades at lower resolutions. To improve the display quality at low resolution, we developed a rendering technique based on the “gravitational” voxel invariants.

In the following sections, we provide details on a segmentation based 3D rendering.

2.9.1 Extraction of the Boundary Voxels

The rendering procedure developed in this research can include visualization of one or more of the tissues \( S_i \) by the use of an isosurface \( \partial S_i \). The voxels that constitute \( \partial S_i \) are potentially visible and can be extracted for each of \( S_i \) on the first step of the algorithm. For each voxel in three dimensions \( v_{x,y,z} = V(x, y, z) \), a set of voxels termed neighborhood \( N(v_{x,y,z}) \) is defined for this extraction. The minimal neighborhood includes \( v_{x,y,z} \) and its six closest voxels along the directions parallel to the coordinate axes. For simplicity, we assume the
voxels are evenly spaced along all coordinate axes with a unit interval. Hence, the minimal neighborhood is defined as

\[ N_6(x, y, z) = N_6(v_{x, y, z}) = \{v_{x \pm 1, y, z}, v_{x, y \pm 1, z}, v_{x, y, z \pm 1}\}. \]

Neighborhoods with more points \( N_{10} \) and \( N_{26} \) are similarly defined. \( N_{10} \) and \( N_{26} \) are more computationally expensive, but they produce more accurate results.

The output of the \textit{atd} algorithm written in terms of indicator functions \( \delta_x(v_{x, y, z}) \) is:

\[ \delta_i(v_{x, y, z}) = \delta_i(x, y, z) = \begin{cases} 1, & v_{x, y, z} \in S_i \\ 0, & \text{otherwise} \end{cases}. \]

Defining the neighborhood characteristic function:

\[ \Delta_p, i(x, y, z) = \delta_i(x, y, z) \sum_{v \in N_p(v_{x, y, z})} \delta_i(v), \]

for which it is convenient to define \( \partial S_i \) as:

\[ v_{x, y, z} \in \partial S_i \iff \Delta_p, i(x, y, z) \mod p \neq 0. \] (61)

Indeed, \( \Delta_p, i(x, y, z) = 0 \) when either the voxel does not belong to \( S_i \), or none of the neighboring voxels belong to \( S_i \). On the other hand, when \( \Delta_p, i(x, y, z) = p \), all voxels from \( N_p \) are internal to \( S_i \) and therefore \( v_{x, y, z} \not\in \partial S_i \). Thus, the set \( \partial S_i \) can be calculated for every \( i \) using (61) with \( O(pn^3) \) comparisons and additions. This operation eliminates 90 – 99 % of the voxels in the 3D image sets used in this research for 3D scene generation.

### 2.9.2 Viewing Plane Projection

Visualizing the tissue \( S_i \) is now reduced to visualizing the \( \partial S_i \) calculated at the previous step. The 3D scene is assumed to be generated using an orthographic projection. Every voxel is projected onto viewing plane \( P_{\text{view}} \) according to a viewing transformation \( VT \). The viewing transformation includes the object’s proper scaling, rotation, and translation combined with the automatic scaling and translation needed to fit the 3D object to the...
viewing area. A detailed overview of a 3D pipeline design can be found in Waggenspack [48] or F. S. Hill [49].

We previously discussed the term “voxel” in the meaning of “material point” (approach (ii) Section 2.2). This “material point”, however, has non-zero “dimensions”, which should be reflected in projections onto the viewing plane. Bulitt and Ayward [37] used circles as approximations of such projections, assuming the original voxel was sphere-shaped. To form an object’s visible surface from its voxel projections, a Z buffer approach [48], [49] is used. A Z buffer contains the entry for each pixel in the viewing plane with two types of information: intensity and depth. A Z buffer algorithm for rendering \( S_i \) follows:

1. Initialize the Z buffer to contain an empty entry with three fields for each pixel in the viewing area:

\[
Z(x, y) = \{I_Z(x, y), z_Z(x, y), i_Z(x, y)\},
\]

where \( I_Z(x, y) \) corresponds to intensity at the given point (for monochrome lighting), \( z_Z(x, y) \) is the “depth” indicator and \( i_Z(x, y) \) is the tissue index.

2. For every voxel \( v_j \in \partial S_i \), do steps 3 and 4;

3. Calculate the projection \( P(v_j) \) of \( v_j \) onto the viewing plane. Every pixel \( (x, y) \in P(v_j) \) is obtained from the original voxel using a viewing transformation; its third coordinate \( z \) is interpreted as the distance of imaged voxel from the observer. The shading technique named gravitational shading (gs) developed in this research is described in Section 2.9.7.

4. If \( Z(x, y) \) is empty or \( z \geq z_{Z(x,y)} \), proceed to the next pixel. Otherwise, assign the fields of \( Z(x, y) \) to the values corresponding to \( P(v_j) \).

5. Every pixel in the viewing plane is assigned an intensity \( I_{Z(x,y)} \).
2.9.3 Lighting Model and Trilinear Interpolation

To produce an image that actually looks three-dimensional, the intensity of each pixel needs to approximate the conditions of a naturally lighted 3D object. Routinely, the lighting intensity with one light source is defined by a combination of the three components: diffuse, specular and ambient:

\[ I = I_{\text{diff}} + I_{\text{spec}} + I_{\text{amb}}. \]

Since the diffuse component characterizes the directional reflections, an accurate determination of this component is essential to adequate 3D image generation. We will describe the calculation of only this component for our 3D rendering algorithm. For further detailed description of lighting models used in computer graphics, refer to [48], [49].

The diffuse component is determined from

\[ I_{\text{diff}} = I_i r_d \cos(\vec{r}, \vec{n}), \]  

where \( I_i \) is the intensity of light source, \( r_d \) - the diffuse reflectivity of the surface, \( \vec{r} \) - the vector from a point on surface to the light source, \( \vec{n} \) - the surface normal. Since \( I_i \) and \( r_d \) are constants and \( \vec{r} \) is known for each point, the problem of calculating \( I_{\text{diff}} \) is equivalent to the problem of finding the surface normal at a given point. To find an accurate approximation of \( \vec{n} \) for raw voxel data, we will use a trilinear interpolation scheme commonly used to find missing values inside a cube.

Consider the unit cube (Figure 2.8). Its corresponding 8 vertices are denoted

\[ V_{\text{cube}} = \{ V_{000}, V_{100}, \ldots, V_{111} \}, \]

and with trilinear interpolation, the value inside the cube at position \((x, y, z)\) can be determined from

\[ V_{\text{ycz}} = \sum_{V_{ijk} \in V_{\text{cube}}} V_{ijk} \left( (1-i)(1-j)(1-k) - x(yz) \right). \]  

(63)
Using (63), we can derive formulas for the intensity gradient at every point inside the cube:

\[
\nabla_{xyz} = \left( \frac{\partial V_{xyz}}{\partial x}, \frac{\partial V_{xyz}}{\partial y}, \frac{\partial V_{xyz}}{\partial z} \right)
\]

\[
\frac{\partial V_{xyz}}{\partial x} = \sum_{V_{ijk} \in \text{cube}} (-1)^{i+1} V_{ijk} \left(1 - j - (-1)^j y \right) \left(1 - k - (-1)^k z \right),
\]

\[
\frac{\partial V_{xyz}}{\partial y} = \sum_{V_{ijk} \in \text{cube}} (-1)^{j+1} V_{ijk} \left(1 - i - (-1)^i x \right) \left(1 - k - (-1)^k z \right),
\]

\[
\frac{\partial V_{xyz}}{\partial z} = \sum_{V_{ijk} \in \text{cube}} (-1)^{k+1} V_{ijk} \left(1 - i - (-1)^i x \right) \left(1 - j - (-1)^j y \right).
\]

(64)

To determine the resulting intensity of the rendered voxel, an averaged gradient is used. Since the trilinear interpolation formula is centrally symmetric, the gradient value \(\nabla_{ctr}\) at the center point

\[V_{ctr} = \frac{V_x}{2} + \frac{V_y}{2} + \frac{V_z}{2}\]

provides an accurate estimation of the normal. Using (64), we obtain \(\nabla_{ctr}\):}

\[
\frac{\partial V_{ctr}}{\partial x} = \frac{1}{4} \left[ 1 + \sum_{V_{ijk} \in \text{cube}} (-1)^i V_{ijk} \right],
\]

(65)

\[
\frac{\partial V_{ctr}}{\partial y} = \frac{1}{4} \left[ 1 + \sum_{V_{ijk} \in \text{cube}} (-1)^j V_{ijk} \right],
\]

\[
\frac{\partial V_{ctr}}{\partial z} = \frac{1}{4} \left[ 1 + \sum_{V_{ijk} \in \text{cube}} (-1)^k V_{ijk} \right].
\]
Formula (65) allows calculating the normal of a unit voxel. If the volume is anisotropic, a corresponding scaling transformation should also be performed.

### 2.9.4 3D Image Zooming Quality

A normal calculated using (62) and (65) provides adequate 3D image quality for high resolution volumes. At lower resolutions or with higher magnification, however, the rendering quality of this voxel-by-voxel scheme degrades, allowing separate voxel projections to become noticeable. Figure 2.9 presents a head visualization using uniform shading of circle voxel projections with normals calculated using (62) and (65). The source volumetric data was obtained from the public domain [50].

![Figure 2.9](image)

**Figure 2.9.** Head visualization using uniform voxel projection shading. Entire volume (left), magnified part of the volume (right).

There are two possibilities for improving the visualization quality of low-resolution data:

1. Using a higher order approximation in the normal calculation formula which includes more neighboring points. This approach allows smoother gradient transition and therefore could produce a more naturally looking image. However, this technique has limits within the current model: higher order approximations substantially increase
pre-processing time and make the visualization less flexible. In addition, as voxel projections become more distinct at higher zooms, the voxel borders can no longer be compensated for with a smoother gradient (See Figure 2.10 for illustration of the interface between two voxels at a high zoom).

2. Using a non-uniform voxel shading to reduce “grainy” effects permits voxel projections to be rendered using smooth transitions. Ideally, every pixel in the projection should be the result of a “micro ray casting” through the voxel. Practical implementation of this principle is not straightforward since the ray casting’s overhead is significant at high magnification.

![Figure 2.10. Two-voxel projection. The interface between two voxels at high zoom is hard to eliminate using uniform voxel shading.](image)

Hence, it is desirable to combine the higher order normal approximation with more “intelligent” voxel projection shading. In the following sections, a scheme to approximate a non-uniform voxel projection that is computationally efficient is derived.

### 2.9.5 Gravitational Concept

Any natural phenomenon can be considered from the point of view of several different theories. Existing physical theories often complement each other in providing a working model. For instance, corpuscular theory explains quantum effects observed in electricity, whereas wave representation of radiomagnetic pulses based on Maxwell equations allows accurate description of many other electromagnetic interactions. In 3D image
visualization, fundamental physical laws, such as those governing light reflection and absorption, form a foundation for rendering algorithms. However, in case of rendering with circle primitives these laws have limited application since the interaction of microstructures (i.e. voxels) produce a prevailing influence on the observed depiction of a 3D object. These effects are generally referred to as low-resolution artifacts, and may be considered the computer graphics analog of nanoparticle interactions that neither follow the patterns of the behavior of matter in a macro world nor the patterns at a submolecular level. For nanoparticles and nanomaterials, special laws govern interacting forces and movement. Analogically, we can consider special voxel invariants for more accurate rendering results.

Suppose we render the part of a digital volume $V$ bounded by two intensity values $c_0 < c_1$. Each rendered pixel is drawn with two values:

1. Its intensity using a lighting model described by (62) and (65);
2. Its opacity defining the result of semi-transparent blending.

The blending rule for drawing a pixel with intensity $i_2$ over the pixel with intensity $i_1$ and opacity $o_2$ is defined by [52]:

$$i_3 = i_1(1 - o_2) + i_2o_2.$$  

In the continuous case, opacity is obtained by integration along the ray. We accept that the background has zero opacity and the opacity of internal object’s voxel is 1. However, if the voxel on a boundary is rendered, we can assume that opacity in the point inside the voxel depends on its distance from the border. We consider opacity as an analog of the physical mass with a density function:

$$p(\bar{r}) = \min(p_0, p_1), \quad (66)$$

where $\bar{r}$ is the radius vector to the current point, and

$$p_0 = c(\bar{r}) - c_0,$$

$$p_1 = c_1 - c(\bar{r}).$$
The density function defined by (66) for point inside a boundary voxel characterizes its
closeness to the border of the segmented volume, and is related to the resulting opacity of the
voxel projection at point \((x, y)\):

\[
o_{xy} = \int_{R_{xy}} \rho(\vec{r}) d\vec{r},
\]

(67)

where \(R_{xy}\) is the ray cast through the voxel that ends on position \((x, y)\) in the view-plane.

However, this formula cannot be used for calculations since the direct application of (66) is
required at every 3D point along the ray. We may consider another approach calculating an
invariant that characterizes the distribution of the density inside the voxel \(v_i\), i.e. its center of
gravity \(C(v_i) = (x_i^c, y_i^c, z_i^c)\). The center of gravity for a three-dimensional volume \(v_i\) is
determined by [51]:

\[
\begin{align*}
x_i^c &= \frac{1}{m_i} \iiint_{v_i} x \rho dV, & y_i^c &= \frac{1}{m_i} \iiint_{v_i} y \rho dV, & z_i^c &= \frac{1}{m_i} \iiint_{v_i} z \rho dV,
\end{align*}
\]

(68)

where

\[
m_i = \iiint_{v_i} \rho dV.
\]

To develop a formula for the center of gravity, a trilinear interpolant (63) is used. An
analytical expression for the center of gravity cannot use the density (66) since this density
function is not defined by an analytical expression. To obtain the derived analytical
expression, we may assume that if the voxel is on the boundary, the density equals either \(\rho_0\)
or \(\rho_1\) throughout the entire voxel. That is, all points inside the voxel are closer to one border
of segmentation intensity band \([c_0, c_1]\) than to the other:

\[
\text{for all } (x, y, z) \in v_i \text{ either } \rho(x, y, z) = \rho_0 \text{ or } \rho(x, y, z) = \rho_1.
\]

(69)

If the intensity band is wide, this assumption produces a working model. With this
assumption, the center of gravity for \(\rho_0\) is:
\[ m_0 = \iiint_{V} \left\{ \sum_{v_{a e V}} V_{v_k} \left[ (1-i-(-1)^i x)(1-j-(-1)^j y)(1-k-(-1)^k z) \right] - c_0 \right\} dx dy dz = \frac{1}{8} \sum_{v_{a e V}} V_{v_k} - c_0, \]

\[ x^c_0 = \frac{1}{m_0} \iiint_{V} x \rho dv = \]

\[ \frac{1}{m_0} \iiint_{V} x \left\{ \sum_{v_{a e V}} V_{v_k} \left[ (1-i-(-1)^i x)(1-j-(-1)^j y)(1-k-(-1)^k z) \right] - c_0 \right\} dx dy dz = x^c - c_0, \]

where

\[ x^c = \frac{1}{24} \sum_{v_{a e V}} V_{v_k} (2-j). \]

Similarly, the expressions for \( y^c \) and \( z^c \) are:

\[ y^c = \frac{1}{24} \sum_{v_{a e V}} V_{v_k} (2-j), \quad z^c = \frac{1}{24} \sum_{v_{a e V}} V_{v_k} (2-k). \]

With this, the expressions for center of gravity are:

\[
\begin{align*}
x_0^c &= \frac{1}{m_0} (x^c - c_0) \\
y_0^c &= \frac{1}{m_0} (y^c - c_0) \\
z_0^c &= \frac{1}{m_0} (z^c - z_0)
\end{align*}
\]

\[
\begin{align*}
x_1^c &= \frac{1}{m_1} (c_1 - x^c) \\
y_1^c &= \frac{1}{m_1} (c_1 - y^c) \\
z_1^c &= \frac{1}{m_1} (c_1 - z^c)
\end{align*}
\]

**2.9.6 Other Forms of Gravitational Invariant**

(72) can be used only if (69) is valid; hence, it would be beneficial to have a more
general expression for the voxel center of gravity. For this purpose, instead of trilinear
interpolation of intensity values, a direct interpolation of the density function based on values
calculated at the vertices of the voxel can be used:

\[ \rho_{xyz} = \sum_{\rho_{v_{a e V}}} \rho_{v_k} \left[ (1-i-(-1)^i x)(1-j-(-1)^j y)(1-k-(-1)^k z) \right], \]

where, according to (66),

\[ \rho_{v_k} = \min(V_{v_k} - c_0, c_1 - V_{v_k}). \]

Using this interpolation,
\[ m = \frac{1}{8} \sum_{p_{\omega} \in V} \rho_{ijk} \]

and the coordinates of a gravitational invariant for the voxel are similar to (70), (71) with \( V_{ijk} \) replaced by \( \rho_{ijk} \):

\[
\begin{align*}
  x^c &= \frac{1}{m} \frac{1}{24} \sum_{p_{\omega} \in V} \rho_{ijk} (2 - j), \\
  y^c &= \frac{1}{m} \frac{1}{24} \sum_{p_{\omega} \in V} \rho_{ijk} (2 - j), \\
  z^c &= \frac{1}{m} \frac{1}{24} \sum_{p_{\omega} \in V} \rho_{ijk} (2 - k) 
\end{align*}
\]  

(73) allows direct computation of the gravitational invariant for the subsequent shading.

Invariants based on trilinear interpolation are computationally efficient first-order approximations. If higher accuracy is needed, a higher order interpolation involving a rapidly growing number of knots would result in a substantial increase in computational complexity. For this reason, we considered another approach to calculate the gravitational invariant.

Suppose that the average intensity values on the cube faces are known and denoted as \( V_{-1}^x, V_1^x \) for the faces \( \{x = 0\}, \{x = 1\} \) of unit cube (Figure 2.11), and \( V_{-1}^y, V_1^y, V_{-1}^z, V_1^z \) for remaining faces.

\[ \text{Figure 2.11. Unit cube with central axes.} \]

For a three-dimensional voxel \( v_{uvw} \), these correspond to
\[
V_i^x \sim c(v_{u+i,v,w}), \quad V_i^y \sim c(v_{u,v+i,w}), \quad V_i^z \sim c(v_{u,v,w+i}), \quad i = -1, 0, 1. \quad (74)
\]

The densities calculated with (66) \(\rho_i^x, \rho_i^y\) and \(\rho_i^z\) correspond to \(V_i^x, V_i^y\) and \(V_i^z\).

Suppose that the mass in the one-dimensional case is concentrated along the axes Ox, Oy, and Oz connecting the centers of opposite faces (Figure 2.11). Denoting density along these axes as \(\rho_x, \rho_y\) and \(\rho_z\), we can determine the one-dimensional centers of gravity along each of these axes as (68):

\[
\begin{align*}
x'^* &= \frac{\int_0^1 x \rho_x \, dx}{\int_0^1 \rho_x \, dx}, \quad y'^* = \frac{\int_0^1 y \rho_y \, dy}{\int_0^1 \rho_y \, dy}, \quad z'^* = \frac{\int_0^1 z \rho_z \, dz}{\int_0^1 \rho_z \, dz}.
\end{align*}
\]

(75)

If the voxel intensity, according to approach (ii) from Section 2.4, represents an average over the unit volume, then the selection of adjacent voxel values (74) is more appropriate than the model in the previous section, where the values of the cube vertices were discrete samples of the signal. Additionally, approach (ii) agrees with the physical process of medical image acquisition, and its realization is preferable and produces more accurate geometric calculations. Since \(\rho_x, \rho_y\) and \(\rho_z\) are one-dimensional functions of one variable, their interpolation requires fewer computations than the general three-dimensional problem. For a linear approximation, a two-point formula for \(\rho_x = a_x x + b\) is obtained from the system of equations

\[
\begin{align*}
\rho_i^x &= a_x \cdot 0 + b_x \Rightarrow a_x = \rho_i^x - \rho_{i-1}^x, \\
\rho_{i-1}^x &= a_x \cdot 1 + b_x \\
\rho_{i-1}^x &= \rho_i^x
\end{align*}
\]

and similarly for \(\rho_y\) and \(\rho_z\). Substituting these for \(\rho_x\) in (75):

\[
x'^* = \frac{\int_0^1 [(\rho_{i-1}^x - \rho_i^x) x^2 + \rho_i^x x] \, dx}{\int_0^1 [(\rho_{i-1}^x - \rho_i^x) x + \rho_i^x] \, dx} = \frac{2\rho_i^x + \rho_{i-1}^x}{6} = \frac{1}{3} \rho_i^x + \rho_{i-1}^x.
\]

(76)
The expressions for $y^c$ and $z^c$ are obtained by replacing the $x$ index by $y$ and $z$. The three-point formula can be similarly obtained with the quadratic approximation

$$\rho_x = a_x x^2 + b_x x + c_x.$$ 

Its coefficients are obtained from the following system of equations:

\[
\begin{align*}
\rho_{-1}^x &= a_x \cdot (-1)^2 + b_x \cdot (-1) + c \\
\rho_0^x &= a_x \cdot 0 + b_x \cdot 0 + c \\
\rho_1^x &= a_x \cdot 1 + b_x \cdot 1 + c
\end{align*}
\]

\[
\Rightarrow \begin{cases}
    a_x = \frac{1}{2} \left( \rho_{-1}^x + \rho_1^x - 2 \rho_0^x \right) \\
    b_x = \frac{1}{2} \left( \rho_{-1}^x + 2 \rho_0^x - \rho_1^x \right), \\
    c_x = \rho_0^x
\end{cases}
\]

and for $m_x$

\[
m_x = \frac{1}{3} \int_{-1}^{1} [a_x x^2 + b_x x + c_x] dx = \frac{2}{3} a_x = \frac{1}{3} \left( \rho_{-1}^x + \rho_1^x - 2 \rho_0^x \right).
\]

Combining these,

\[
x^c = \frac{1}{m_x} \int_{-1}^{1} [a_x x^3 + b_x x^2 + c_x x] dx = \frac{1}{m_x} \frac{2}{3} b_x = \frac{\rho_{-1}^x + 2 \rho_0^x - \rho_1^x}{\rho_{-1}^x + \rho_1^x - 2 \rho_0^x}.
\]

As before, expressions for $y^c$ and $z^c$ are obtained by replacing the $x$ index. A point within the voxel bounds obtained using (77) is termed a quasi center of gravity.

### 2.9.7 Efficient Shading of Semi-transparent Voxel Projection

When the voxel gravitational invariant (center of gravity (72) or (73), quasi center of gravity (77)) is calculated, it is possible to calculate voxel projection shading. To determine this shading, consider an arbitrary point inside the sphere $P_{xyz} = (x, y, z)$, where the distance between $P$ and an invariant point $C(v)$ is denoted $d(C(v), P_{xyz})$. Initially, we considered the relation between opacity and $d(C(v), P_{xyz})$ as:

\[
O(P_{xyz}) \sim 1 - \frac{d(C(v), P_{xyz})}{d(C(v), A)},
\]

(78)
Figure 2.12 presents the planar section $\beta$ of sphere $S(v)$ to represent voxel $v$; the plane $\beta$ is defined by three points: sphere center $O, P_{xyz}$ and $C(v)$. The point $A$ lies on the intersection of the ray $\overline{CP_{xyz}}$ and $S(v)$. The sphere is completely opaque in its center of gravity (78) and transparent on the borders to allow smooth transition into neighboring spheres.

![Figure 2.12. Sphere section illustrating the relation of opacity and distance from its gravitational invariant.](image)

Each point in the two-dimensional spherical voxel projection can be assigned a value that corresponds to the resultant opacity along the viewing ray (similar to (67)):

$$O_{xy} = \int_{(u,v,w) \in R_v} O(P_{uvw}) dudvdw.$$  

(79)

Unfortunately, (78) uses a ratio of Euclidean distances, and the integral (79) becomes too complex to be computationally-feasible. For this reason, a more practical approach that results in a similar visual effect was derived.

In order to simplify the results, it is convenient to perform the calculations in a coordinate system that has its origin at the center of $S(v)$ (Figure 2.13). To derive a working approximation for (78), the opacity can be modeled as being linearly dependent on the distance from the $\gamma$ plane on Figure 2.13, i.e.:

$$O(P_{xyz}) \sim \frac{1}{d(\gamma, P_{xyz})}.$$  

(80)
Figure 2.13. Spherical voxel representation and its coordinate system. Point \( C \) is the center of gravity, \( A \) lies on intersection of \( S \) and \( \overrightarrow{OC} \) ray. \( \gamma \) is the plane containing \( A \) and perpendicular to \( OC \), \( r \) is the radius of \( S \), \( r = \| OA \| \).

Using (80), the opacity increases linearly in the \( \overrightarrow{OC} \) direction. The magnitude of the opacity gradient throughout the sphere depends upon the ratio \( r_0 / r \), where \( r_0 = \| OC \| \). With this model, the opacity gradient perpendicular to the \( \gamma \) plane is close to zero when center of gravity is near the center of the sphere, and increases as the ratio \( r_0 / r \) increases.

The coordinates of point \( A \) on Figure 2.13 are:

\[
\begin{align*}
\begin{cases}
    x_A &= x^c t \\
y_A &= y^c t \\
z_A &= z^c t \\
x_A^2 + y_A^2 + z_A^2 &= r^2
\end{cases}
\Rightarrow t = \frac{r}{r_0}.
\end{align*}
\]

The equation for the \( \gamma \) plane is:

\[
x^c (x - x_A) + y^c (y - y_A) + z^c (z - z_A) = \left( x^c x + y^c y + z^c z \right) - \frac{r}{r_0} \left( [x^c]^2 + [y^c]^2 + [z^c]^2 \right) = x^c x + y^c y + z^c z - rr_0 = 0.
\]

Accordingly, the distance from the arbitrary point \( P(x, y, z) \) inside the sphere to \( \gamma \) is:
\[ d(P, \gamma) = rr_0 - \left( x^e x + y^e y + z^e z \right). \]

To obtain the result of a projection onto the point \((x_p, y_p)\) of viewing plane, integration similar to (79) along the line parallel to \(z\) axis within the sphere becomes:

\[
O(x_p, y_p) = \int_{-z_s}^{z_s} d(P, \gamma) dz = \int_{-z_s}^{z_s} (rr_0 - x^e x_p - y^e y_p - z^e z - rr_0) dz = (rr_0 - x_p x^e - y_p y^e) z - \frac{z^2}{2} \bigg|_{-z_s}^{z_s} = 2(rr_0 - x_p x^e - y_p y^e)Z_s,
\]

where \((x_p, y_p, \pm Z_s)\) are the points of intersection of the projection line \(\{x = x_p, y = y_p\}\) with the sphere \(S (x^2 + y^2 + z^2 = r^2)\). From these results, we obtain:

\[
O(x_p, y_p) = 2(rr_0 - x_p x^e - y_p y^e)\sqrt{r^2 - x_p^2 - y_p^2}. \tag{81}
\]

The desired spherical voxel projection shading was developed with (81), which we call gravitational shading \((gs)\). \(Gs\) is performed within step 3 of the Z buffer algorithm (Section 2.9.2) and it becomes part of our 3D graphics rendering pipeline. \(Gs\) requires that the center of gravity for every voxel must be determined; this calculation is performed only once, and can be carried out with either (73) or (82) in the initialization step. The remaining input for \(gs\) includes the center of each voxel \(v_i\) in the proper coordinate system, viewing transformation matrix \(VT\), the rotation/scaling matrix \(R\) of the viewing transformation, and the light source position \(L(x_L, y_L, z_L)\). Then the shading algorithm for every voxel \(v_i\) proceeds as follows:

1. Calculate the normal \(n(v_i)\) (65). Determine the voxel base intensity \(I_{diff}(v_i)\) with the lighting model (62).

2. Determine the sphere radius \(r\):

\[
r = \frac{1}{2} \|R(1,1,1)\|. \tag{83}
\]
3. Obtain the viewing transformation result $VT(v_i)$, and change the voxel coordinate system to the one on Figure 2.13 by the translation of the center of the sphere to the origin.

4. Calculate $I_{\text{diff}}(v_i)$ and the opacity of every pixel inside the spherical projection (81) and pass these results to the Z buffer algorithm.

### 2.10 Evaluation of $Gs$

The fast rendering algorithm described in Sections 2.9.1-2.9.7 was implemented in C++ with a Windows XP platform without the use of any third-party graphical libraries. To validate the correctness of $gs$, an implementation of the voxel shading function was developed in MATLAB. This implementation correctly renders the three-dimensional sphere projections. Figure 2.14 shows the projection of a sphere with different locations for its center of gravity.

To assess the performance of $gs$, we used the properties of efficient 3D visualization from Section 2.7: adequate rendering quality, speed and flexibility.

![Figure 2.14. Test shading of sphere projection. Sphere of radius 100 is centered in origin. Centers of gravity, from left to right: (90, 0, 0), (90, 90, 0), (90, 90, 90), (20, 0, 0).](image)

### 2.10.1 Surface rendering quality

We compared $gs$ with a direct rendering method that uses uniform circular projection shading. We chose the fast segmented volume rendering (fsvr) based on its description from...
Bullitt and Aylward [37], complemented with the normal calculation (65) and the Z buffer algorithm (Section 2.9.2).

The first comparison used a 3D phantom dataset consisting of a $10 \times 10 \times 10$ cube. This size of rendered voxels permits an estimation of the subvoxel shading quality for both methods (Figure 2.15).

To compare the rendering quality with authentic medical image volumes, the following datasets were used:

1. A $256 \times 256 \times 109$ 16-bit MR sagittal images of the head with manually removed cerebral cortex [50]. Initial segmentation was obtained using background thresholding. Rendering results are shown on Figure 2.16;

2. A $256 \times 256 \times 200$ 10-bit MR axial images of the head. Initial segmentation was obtained with a method similar to Set 1. Rendering results are shown on Figure 2.17.

3. A $181 \times 217 \times 181$ 20% biased 12-bit sagittal phantom MR normal brain volumes from the BrainWeb simulator [32]. This set was pre-processed to demonstrate the capabilities of our automated segmentation and visualization pipeline using the algorithms developed in this research. Initially, it was corrected using $dsf$ (Chapter 1) to remove non-uniformity that could interfere with $atd$ (Sections 2.8.1-2.8.2), our segmentation algorithm. In the second stage, the cerebral cortex was removed using BET (Brain Extraction Tool) based on an extraction algorithm from S. M. Smith [53], which is incorporated into the MRIcro medical image analysis software [54]. Consecutively, $atd$ was applied to a stripped volume to detect the grey and white matter tissue intensity ranges. Based on this segmentation, visualization using $fsvr$ and $gs$ was performed separately for grey matter (Figure 2.18) and white matter (Figure 2.19).
Figure 2.15. Rendering of a cubical volume. Result produced by $fsvr$ (left) and $gs$ (right).

Figure 2.16. Rendering of Set 1. Results of $fsvr$ (left column) and $gs$ (right column).
Figure 2.17. Rendering of Set 2. Results of $f_{svr}$ (left column) and $g_s$ (right column).

Figure 2.18. Rendering of Set 3 (GM). Results of $f_{svr}$ (left column) and $g_s$ (right column).
Figure 2.19. Rendering of Set 3 (WM). Results of fsvr (left column) and gs (right column).

Figure 2.20. Rendering of Set 4. Result of fsvr (left) and gs (right).
4. A 512 × 512 × 50 10-bit CT axial chest image volume provided by the Bakoulev Center for Cardio-Vascular Surgery, Moscow, Russia. Its segmentation of soft tissue and bone was performed automatically with \textit{atd}, and the rendering results with \textit{fsvr} and \textit{gs} are shown on Figure 2.20.

Renderings of datasets 1-3 provide visual evidence that \textit{gs} produces more natural looking and smoother images than \textit{fsvr}. The \textit{gs} advantages are particularly visible on dataset 3, where the rendering of isosurfaces in a large area containing fine details was required. Here, gravitational semi-transparent shading provides smoother shapes with clearer fine details. In bone-like structures, however, many objects have several voxels of width, and the accuracy for the calculation of the normal decreases. For this reason, gravitational shading did not show significant improvement over \textit{fsvr} of dataset 4 (Figure 2.20).

\textbf{2.10.2 Rendering Speed}

The improvement in quality of shading often increases the time to process every frame of a 3D animation. Sometimes such an improvement is achieved with a cost in performance that may not be acceptable for real-time applications. If the resultant performance of a new method is inefficient in terms of the definition in Section 2.7, it will not be used. To verify that \textit{gs} does not significantly degrade the rendering speeds, we compared the average frame rates of \textit{fsvr} and \textit{gs} during rotation of an object about its principal axes. Since \textit{gs} performs non-uniform voxel projection shading, extended processing times can be expected when the size of voxel projections rendered becomes significant. For this reason, rendering frame rates with magnification up to 3 times the original frame were also examined.

Since \textbf{efficient} (Section 2.7) visualization should provide acceptable frame rates with a current “off-the-shelf” PC, a typical current PC workstation with Pentium IV 3.06 MHz processor and 1GB conventional memory was chosen to estimate the performance of \textit{gs} vs.
fsvr. The original viewing plane window was set to 400 × 400 pixels. The results of performance comparison of fsvr and gs are summarized in Table 2.1. The frame rates of gs have increased compared to fsvr, but remain acceptable for efficient rendering.

Table 2.1. Rendering speeds of fsvr and gs.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Rotation frame rate, seconds−1</th>
<th>Zoomed frame rate, seconds−1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>fsvr</td>
<td>gs</td>
</tr>
<tr>
<td>Set 1</td>
<td>4.2</td>
<td>2.9</td>
</tr>
<tr>
<td>Set 2</td>
<td>2.1</td>
<td>1.7</td>
</tr>
<tr>
<td>Set 3, GM</td>
<td>2.1</td>
<td>1.1</td>
</tr>
<tr>
<td>Set 3, WM</td>
<td>2.6</td>
<td>2.4</td>
</tr>
<tr>
<td>Set 4</td>
<td>3.0</td>
<td>1.7</td>
</tr>
</tbody>
</table>

2.10.3 Flexibility

Flexibility implies that the time required to prepare data for interactive 3D rendering should be “clinically” acceptable. For gs and fsvr, this is the time (T) spent pre-processing segmentation boundaries into the set of voxels to be rendered (Section 2.9.1). When the object to be rendered is redefined, T is the time required for gs to re-process the 3D volume. In Table 2.2, times T for all datasets are shown. Since T was only a few seconds, gs-based visualization is flexible and can be used in real clinical work.

Table 2.2. Pre-processing times.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Set 1</th>
<th>Set 2</th>
<th>Set 3, GM</th>
<th>Set 3, WM</th>
<th>Set 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>T, seconds</td>
<td>2.8</td>
<td>6.3</td>
<td>3.9</td>
<td>2.0</td>
<td>4.7</td>
</tr>
</tbody>
</table>

2.11 Discussion and Conclusions

The final goal of the research in this chapter was to develop stable and efficient algorithms for automatic 3D visualization of advanced structures in digital medical image volumes. These algorithms, excluding atd, were incorporated into UniViewer medical image viewing/analysis software (Figure 1.16) available from
One difficulty in visualization design is a trade-off between performance and visual quality. We intentionally avoided computationally complex techniques that require extended times for either pre-processing (surface rendering methods) or 3D image generation (a majority of volume rendering methods). When surface rendering is used, pre-processing decreases the flexibility of a rendering method, whereas 3D scene generation times are long and do not allow consideration as interactive tools for real-time use. We also excluded hardware dependent methods due to inherently low flexibility with their application. The most efficient methods developed are currently hybrids, which evolved from purely surface or volume rendering techniques into sophisticated conglomerates, and this research resulted in a hybrid based on previously proposed direct methods.

Direct rendering methods combined with volume pre-segmentation can produce fast visualizations ([35], [36], [37]). They treat the voxels as material points, and voxel projections are rendered using a geometric primitive, e.g. a circle with fixed radius and intensity. Due to the internal voxel removals based on segmentation, the rendering of a 3D image is very fast, although pre-existing object segmentation is required. To obtain an automated 3D processing pipeline, we developed a robust method of image histogram peak detection that provides segmentation ranges for all tissues represented in an image. This method (atd) was implemented in MATLAB and successfully applied to a number of brain and chest volumes. The output of atd serves as an input to the segmentation based 3D rendering for the full automation of 3D image generation.

One drawback of voxel-based rendering methods like fsvr [37], is a lack of visual quality, because fine objects with a size of several voxels cannot be adequately rendered with voxel-size primitives. A common method to improve fine detail quality is to use local surface
rendering based on smooth normal (such as splines) interpolation. This allows smoother surfaces at the cost of increased pre-processing time. To do 3D visualization in real-time, we used a different approach: before executing voxel-by-voxel projection, certain characteristics of the shading gradient for each voxel are determined. The further shading of the voxel projections uses advanced shading. We developed the “gravitational” concept that allowed expressions for the calculation of a shading gradient characteristic (gravitational invariant or (quasi) center of gravity), and semi-transparent voxel projection shading algorithm named gravitational shading. The visualization with these algorithms resulted in significant quality improvement compared with the conventional voxel-based fsvr algorithm, especially for large area surfaces with a considerable amount of fine detail, e.g. white and grey matter surfaces.

Although gs required more computation than conventional fsvr, it could still produce one or more frames per second for the datasets examined, providing the required interactive frame rates. Accordingly, gs can be used efficiently in clinical environments with large volumetric throughput.

The medical image volume processing pipeline consisting of atd, the gravitational invariant calculation and gs, does not require any additional input parameters with the original volumetric image data. It can quickly visualize the result of a 3D volume automated pre-segmentation without operator interaction, providing a strong tool for more efficient medical image volume analysis for diagnostic and research purposes.
Chapter 3 Summary

In this final chapter, we provide a brief description of results obtained in this research, conclusions reached and suggestions for future research.

3.1 Results

This research resulted in a number of new algorithms developed for automated correction, analysis and visualization of digital MR image volumes. In the first chapter, we developed an algorithm for the MR inhomogeneity correction. Our original work in developing this algorithm consisted of the following:

- overview and classification of existing correction methods;
- definition of a criteria for the new correction method;
- developing the conditions for the safe log-transform;
- developing an original MR bias correction concept;
- developing the apparatus to describe this concept and produce mathematical foundation for a correction algorithm;
- based on this foundation, developing the new inhomogeneity correction algorithm;
- development of a partial derivative scheme for use in the algorithm and the estimation of error;
- justification of basis function selection and error estimation for the chosen basis;
- development of the efficient weight functions;
- development of advanced image scaling procedure;
- implementation of the resulting algorithm derivative surface fitting ($dsf$) in software and making it available for public use and testing;
- evaluation of $dsf$ on synthetic and phantom images;
comparison of dsf with two previously published similar methods. The comparison by several criteria was performed on seventeen digital image volumes;
- visual evaluation of dsf on 1000+ images.

In the second part of this research, we have developed a sequence of new algorithms for automated analysis and visualization of digital MR image volumes. The original work relevant to this part included:
- overview of existing rendering techniques;
- mathematical definition of digital volume, its primary elements (voxels), isosurface and their relationship;
- definition of criteria for clinically efficient rendering;
- developing a fully automatic algorithm (atd) for detection of all sufficiently represented tissues in the MR image;
- verification of atd performance;
- development of formulas for the normal calculation based on a trilinear interpolation cube;
- developing the gravitational concept for improved voxel projection shading;
- developing mathematical foundation for calculation of three types of gravitational invariants used for improved voxel projection shading;
- developing an efficient shading algorithm of a spherical non-uniform voxel representation; this shading algorithm preceded by gravitational invariant calculation resulted in a general gravitational shading (gs) procedure;
- implementing gs-based 3D visualization in UniViewer software available as a download;
- evaluation of the automatic analysis and a visualization pipeline consisting of atd combined with gs on a number of digital image volumes. Comparison based on
efficiency criteria performed with the previously proposed fsvr visualization method [37].

3.2 Conclusions

Conceptually, the problem addressed in this research can be stated as: is it possible to create a fully automated system for correction, pre-segmentation and visualization of medical digital volumes that is stable, effective and fast enough to be used in clinical environments? Apparently, the scope of this problem is such that the effort of the entire research community is required for conclusive results.

This research has developed a number of approaches to produce elements of a such system. It has demonstrated that the efficient automated system for digital volume processing requires deep integration of the included algorithms. The pipeline consisting of MR correction, pre-segmentation and fast rendering developed here, is an example of such integration, since all its parts are closely related (for instance, gs requires atd or another segmentation algorithm to be run).

Another issue is the processing speed of all medical image processing pipeline components. For a clinician, it is often critical to obtain the final results in a short fixed time, after which they may not be useful. Even though the requirements in research environments are not as strict, efficient processing becomes a key issue on large volumes of image data used for validation of proposed algorithms. On the other hand, effectiveness of techniques developed cannot be below a certain established level of output data quality to be of interest for clinicians and researchers. In this research, we tried to incorporate these requirements at the initial algorithm design stage, estimating the complexity of each step and intentionally avoiding computationally expensive solutions. Since the total estimated error of the algorithm output cannot be below the highest error obtained at each step, error estimation at every stage of proposed technique avoids unnecessarily complex calculations with high precision, if this
precision significantly exceeds the maximum precision obtained at other steps. Consequently, this research has demonstrated the efficiency of these design considerations for developing the automated pipeline for digital MR image volume processing.

### 3.3 Future Research

To increase the effectiveness of MR non-uniformity correction algorithm developed in Chapter 1, several improvements can be suggested:

1. Modification of the algorithm to the iterative version by introducing a convergence condition. This would allow stronger corrections but can potentially lead to unstable results, so additional research is needed.
2. Research of locally adaptive weight functions to exclude high gradient areas.
3. Explore the possibilities to automate the β percentile selection used to determine the gradient threshold.

Further research related to the algorithms developed for the visualization pipeline in Chapter 2, is suggested to include:

1. Integration of *atd* into the *UniViewer* software;
2. Modification of existing graphics pipeline to display multiple segmented objects on a 3D image with different colors;
3. Improving the gravitational invariant calculation algorithm to exclude the degenerate cases when the center of gravity lies on the voxel’s boundary;
4. Developing a more accurate expression to determine the voxel radius (83) to compensate for potential interfacing artifacts between the voxels.

The general trend for the automation of various tasks of digital MR image volume processing should also continue in the future research. The pipeline developed in this research can be used as a foundation for the correction and visualization of specific organs and tissues, such as liver or brain matter.
Recent findings in neurology show the importance of measuring the volume of a brain structure called the hippocampus. It has been shown that the prediction and treatment of Alzheimer’s Disease (AD) and Mild Cognitive Impairment (MCI) can be improved significantly with hippocampal volume assessment. However, automated detection of hippocampus is complicated by a lack of a clear interface with other structures and natural subject variability. Among currently available automated hippocampus extraction methods, segmentations combined with hippocampus registration provide the results more consistent with manual segmentations than atlas-based. Therefore, the fully automated correction and segmentation developed in this research combined with the hippocampal registration will be a valuable contribution in the development of automated hippocampus extraction methods.
References


34. Manually segmented MR volume archive at the Center for Morphometric Analysis at Massachusetts General Hospital, http://www.cma.mgh.harvard.edu/ibsr/.


54. MRICro medical imaging and analysis software for Unix and Windows. [http://www.sph.sc.edu/comd/rorden/mricro.html](http://www.sph.sc.edu/comd/rorden/mricro.html)
Appendix:  *DsF* Algorithm Pseudocode

This appendix provides the *DsF* pseudocode, as described in Section 1.7.7. This version does not include the code for the edge-preserving Gaussian smoothing discussed in Section 1.7.2 and the advanced formula for inverse scaling discussed in Section 1.7.6. For weight functions, (39) is used. All input and output parameters of functions are *italic*; function name headers are *bold*.

**Main**

**Input**

- two-dimensional array *SRC* representing original image;
- its dimensions *m*, *n*;
- basis degree *N*;
- number of partial derivatives *nDerivs* (always 2 (*d / dx*, *d / dy*) for this version)
- persentile threshold *beta*.

**Output**

- two-dimensional array *RES* representing corrected image.

**Body**

REM in this version, *d / dx* and *d / dy* partial derivatives are used

\[ Nb2 := \left( \frac{(N + 2) * (N + 1)}{2} \right) - 1 \]

\[ \text{srcMin} = \min (SRC) \]

\[ \text{srcMax} = \max (SRC) \]

\[ \text{LOGSRC} := \log (SRC + 2) \]

\[ [B, dB] := \text{Generate\_basis\_matrices} (N, Nb2, m, n) \]

\[ [dA, W] := \text{Generate\_derivative\_matrices} (\text{LOGSRC}, m, n, nDerivs, beta) \]

REM Initialize *L* (left hand side of linear system)
L := Nb2 × Nb2 matrix
for all i, j L(i, j) := 0 End for

REM Initialize R (right hand side of linear system)
R := vector with Nb2 rows
for all i R(i) := 0 End for
for all i from 1 to Nb2
    for all j from 1 to Nb2
        for all k from 1 to nDerivs
            for all i1, j1
                L(i, j) := L(i, j) + W(i1, j1) * dB(i1, j1, i, k) * dB(i1, j1, j, k)
            end for
        end for
    end for
end for
for all k from 1 to nDerivs
    for all i1, j1
        R(i) := R(i) + W(i1, j1, k) * dB(i1, j1, i, k) * dA(i1, j1, k)
    end for
end for

REM Gaussian_solve returns a solution vector for L (left part of linear system) and R
C := Gaussian_solve (L, R)

REM initialize BIAS matrix
BIAS := m × n matrix
for all i

    for all i1, j1

        BIAS (i1, j1) := BIAS (i1, j1) + C(i) * B (i1, j1, i)

    end for

end for

REM initialize RES matrix

RES := m \times n matrix

RES := \exp (\text{LOGSRC} - \text{BIAS} - 2)

RES := (RES - \min (RES)) \times (\frac{\text{srcMax} - \text{srcMin}}{\max (RES) - \min (RES)}) + \text{srcMin}

output RES

end Main body

**Generate_basis_matrices**

Input

- basis degree \( N \);
- basis size \( Nb2 \);
- image matrix dimensions \( m, n \).

Output

- \( m \times n \times Nb2 \) array \( B \) containing basis elements
- \( m \times n \times Nb2 \times nDerivs \) array \( dB \) containing derivatives of all basis elements

Body

    Basis_Index := 0

    REM initialize \( B \) – output basis array and \( dB \) – basis derivative array

    B := m \times n \times Nb2

    dB := m \times n \times Nb2 \times nDerivs
for all i1, i2, i3 B (i1, i2, i3) := 0 End for

for all i from 0 to N
    for all j from 0 to N - 1
        if i = 0 and j = 0 continue to next j
        Basis_Index := Basis_Index + 1
        for all i1 from 1 to m
            for all j1 from 1 to n
                B (i1, j1, Basis_Index) := [(i1 / m) ^ i] * [(j1 / n) ^ j]
            end for
        end for
        for all k1 from 1 to nDerivs
            dB (i1, j1, Basis_Index, k1) := (i - 1) * (i1 / m) ^
                (i - 1) * (j - 1) * (j1 / m) ^ (j - 1)
        end for
    end for
end for

output B
output dB
end Generate_basis_matrices Body

Generate_derivative_matrices

Input
- log – transformed image LOGSRC;
- image dimensions m, n;
- number of derivatives nDerivs;
- percentile beta.
Output

- $m \times n \times n_{Derivs}$ array containing image partial derivatives.

Body

REM Initialize derivative output array $dA$

\[
dA := \text{array } m \times n \times n_{Derivs}
\]

REM Initialize weight output array $W$

\[
W := \text{array } m \times n \times n \times \text{Derivs}
\]

for all $i$ from 1 to $n_{Derivs}$

\[
dA (1 \to m, 1 \to n, i) := \text{Partial-Derivative (LOGSRC, } m, n, i)\]

REM percentile ($\beta$, $A$) function returns $\beta$-percentile of values

REM in matrix $A$

\[
\text{margin} := \text{percentile (} \beta, dA (1 \to m, 1 \to n, i))
\]

for all $j$ from 1 to $m$

for all $k$ from 1 to $n$

if $dA(j, k, i) < \text{margin}$ then $W (j, k, i) := 1$

else $W (j, k, i) := 0$

end for

end for

end for

output $dA$

output $W$

end Generate_derivative_matrices Body

Partial_derivative

Input

- Matrix $A$ on which to apply a convolution filter;
- matrix dimensions \( m, n \);
- derivative index \( iD \).

Output
- resulting \( m \times n \) matrix \( A_{\text{conv}} \)

Body
REM Initialize output array \( A_{\text{conv}} \)

\[ A_{\text{conv}} := \text{array } m \times n \]

for all \( i, j \) \( A_{\text{conv}} \)(i, j) := 0 end for

REM \( iD = 1 \) corresponds to \( d/dx \); \( iD = 2 \) corresponds to \( d/dy \).

REM initialize convolution matrix \( F_X \)

\[
\text{if } iD = 1 \text{ then } F_X = 
\begin{bmatrix}
-1/8 & 0 & 1/8 \\
-1/4 & 0 & 1/4 \\
-1/8 & 0 & 1/8 \\
\end{bmatrix}
\]

\[
\text{else } F_X = 
\begin{bmatrix}
1/8 & 1/4 & 1/8 \\
0 & 0 & 0 \\
-1/8 & -1/4 & -1/8 \\
\end{bmatrix}
\]

for all \( i, j \) from 1 to \( m, n \)

for all \( p, q \) from 0 to 2

\[ A_{\text{conv}} \)(i, j) := A_{\text{conv}} \)(i, j) + \( A \) (min (i + p, m), min (j + q, n)) \ast F_X \)(p, q) \]

end for

end for

output \( A_{\text{conv}} \)

end Partial_derivative Body
Vita

Mikhail Milchenko was born in 1977 in Obninsk, Kaluzhskaya Oblast, USSR. He graduated from the Gymnasium of Obninsk majoring in mathematics and physics, in 1994, and in the same year passed the entrance tests to the Lomonosov Moscow State University, Department of Mechanics and Mathematics. In 1998, he started working as a software engineer in Scientific Production Association “Typhoon” on meteorological visualization system. After graduating in 1999, he moved back to Obninsk and continued his work in “Typhoon” while teaching information technology in the Institute for Nuclear Power Engineering. In the fall of 2000, he started the doctoral program in computer science at Louisiana State University while working as a research assistant on the medical imaging project funded by LSU Medical School and hosted by the Department of Computer Science. The degree of Doctor of Philosophy will be awarded to Milchenko at the December 2005 Commencement.