University of Alberta

Multiscale Boundary Identification for Medical Images

by

Zu Ying Wang

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science

Department of Electrical and Computer Engineering

Edmonton, Alberta
Spring 2007

Abstract

Segmentation of medical image data plays a crucial role in image processing. Manual tracing of the object boundaries by a specialist is a time-consuming task and is subject to operator variability. Edge information contains important characteristics of image content. Fully automatic segmentation procedures are still far from satisfactory in many real situations. In this thesis, the well-known multiscale wavelet-based techniques have been used for edge detection and investigated to enhance boundary identification of medical images. Directional edge detection and subpixel edge resolution are obtained to improve the edge detection performance. Wavelet modulus maxima chain connection through wavelet decay gives a measurement of scale depth of the edge. An edge quality analysis is carried out after using Lipschitz regularity theory to further identify true edges from noise. Experimental results have shown that the proposed multiscale boundary identification method provides superior performance in medical image segmentation.

Chapter 1

Introduction

Various medical imaging modalities such as the radiograph, computed tomography (CT), and magnetic resonance (MR) imaging are widely used in routine clinical practice. Anatomical information about the human body can be obtained from non-invasive medical imaging technique such as MR imaging. With full three-dimensional capabilities, excellent soft-tissue contrast and high spatial resolution, it provides a detailed image of living tissues for brain or any other part of the body. Data obtained from MR images can be used for detecting tissue deformities such as cancers and injures, and also for studies of brain pathology. Manual tracing of the object boundaries by a specialist is a very time-consuming task and is subject to operator variability, and hence automated image segmentation is preferable.

1.1 Motivation

Image segmentation is among the most common tasks in image processing, and it is of fundamental importance in such areas as computer vision and medical imaging. Image segmentation is a process of separating an image into several component parts. For example, using segmentation technique, a head image can be segmented into different tissue classes, such as gray matter, white matter, and cerebrospinal fluid. Image segmentation can be performed by identification of the tissue boundary for each tissue class, which is edge-based segmentation. It can also be performed by classification of the pixels for each tissue class, which is region-based segmentation. Many researchers have

been working to solve the medical image segmentation problem, however, it is still a difficult task as fully automatic segmentation procedures are far from satisfactory in many real situations.

Edge detection is a problem of fundamental importance in image analysis, and it plays an important role in segmentation. Edges can be considered as pixel intensity discontinuities within an image. An edge is a curve that follows a path of rapid change in image intensity. Basically, edge detection identifies edge pixels where rapidly changing intensity occurs. Since edges are often associated with object boundary in an image, one can link edge pixels to form the object boundary after edge detection.

Classical edge detection is based on convolution of difference operator, such as Sobel and Prewitt operators. Gradient methods such as Canny and Laplacian of Gaussian use a first and second derivative of the Gaussian filter respectively. Most of the edge detectors can perform well on a clean image, but in noisy images, it may miss edges or produce false edges caused by discontinuities in intensity levels due to noise.

Segmentation of MR images is not a trivial task as it involves three main difficulties. First of all, the gradient coils create an inhomogeneous magnetic field. The intensity is non-uniform, which means the average intensity level of pixels within a single tissue class varies over the whole image. Secondly, the noise in the MR image can alter the intensity of a pixel to make its classification become uncertain. Thirdly, there is a partial volume effect that exists in all medical imaging modalities. It means individual pixels in the image contain a mixture of tissue classes, so there may appear a finite width edge between two classes where the intensity of a pixel on the edge is not consistent with either class but lies somewhere between.

Medical image segmentation software mtrack [1] written in MATLAB [2] has been developed. This software contains several functions such as edge detection, edge feature extraction, track parameter initialization, edge tracing, and result display. The track contours can be stored for further use. It is an edge-based segmentation method.

1.2 Objective

For edge-based segmentation, usually edge detection is performed first to form an edge image, and then edge tracing is performed after object boundary is identified. Thus, edge detection, boundary identification, and edge tracing are the three main stages in edge-based image segmentation. In the edge-based segmentation process, the identification of a coherent boundary provides useful information for the purpose of segmentation, so that the edge tracing algorithm can trace the correct boundary.

A novel edge-based segmentation algorithm [1] has been developed in University of Alberta. The main limitations [1] of edge-based segmentation algorithms are: i) sensitivity to noise; ii) the potential for gaps in the boundaries that are formed; and iii) the potential for false edges to be included in the boundary. These have the effect of producing low robustness in the edge-based segmentation. Since noise is the main limitation in most of the MR image segmentation algorithms. Wavelet based edge detection technique has been shown to have advantages over other classical edge detection techniques as it is less sensitive to noise. The aim of this thesis is to study and investigate wavelet techniques, to develop a multiscale boundary identification algorithm for better edge-based segmentation of MR imaging.

1.3 Thesis Organization

The remainder of this thesis is organized as follows.

Chapter 2 of this thesis briefly reviews the recent multiscale research in edge detection and image segmentation area. The existing methods such as the wavelet technique, edge detection, the optimization method, fuzzy logic, and medical image segmentation software that will be used in this thesis research are described in detail.

Chapter 3 contains the proposed method developed based on the existing theory for multiscale edge detection and boundary identification, and demonstrates how they are exploited, improved, and integrated into the image segmentation software.

Chapter 4 presents the performance evaluation of the proposed method. The conclusions and future work are presented in chapter 5.

Chapter 2

Review of Related Work

Multiscale image representations have attracted much attention from researchers lately. The objective of this thesis is to investigate and develop efficient boundary identification method based on multiscale wavelet technique. In this chapter, we present a brief review of the related background work. The organization of this chapter is as follows. Wavelet theory will be reviewed briefly in section 2.1. Multiscale wavelet technique for edge detection will be described in section 2.2. Optimization method, fuzzy logic, and image segmentation software that will be used in this thesis research will also be described in sections 2.4, 2.5, and 2.6, respectively. These algorithms described in this chapter will be used in the proposed method for multiscale boundary identification.

2.1 Wavelet Transform

There are many transforms such as Fourier transform [3] and wavelet transform [4] that are used by engineers and mathematicians. Every transform has its own area of application with advantages and disadvantages. The Fourier transform is very popular in frequency based digital signal processing. However, it is not suitable for non-stationary signal analysis. Using Fourier transform, it is easy to determine frequency components present in a signal, but it is difficult to determine when those frequency components occur. In order to solve this problem, short time Fourier transform (STFT) has been developed, which uses a sliding window to map a signal into a 2-D time-frequency

representation. In STFT analysis, it is assumed that within a window the signal is stationary. A major drawback of the STFT is that the window size is fixed for all frequencies. This shortcoming of STFT is solved in wavelet transform (WT). The WT also maps a signal into a 2-D time-frequency representation. However, unlike the STFT, the window size of the WT is varied to enable a trade-off between time and frequency.

Since the advent of wavelet theory, it has been shown that the WT has many advantages over other traditional transforms. The WT decomposes a signal into shifted and scaled versions of the mother wavelet function. Compared to the STFT, the 'window' (i.e. the support of the wavelet functions) can be compressed or stretched by scaling. Through scaling and translation, the mother wavelet reproduces itself in certain ways, and hence it has the ability to perform analysis in both time and frequency domains.

Wavelet analysis is a powerful tool for time-frequency analysis. It has been applied successfully in image processing applications, such as image compression. It also shows high potential for detecting image edges. The multiresolution feature of WT allows us to 'zoom in' (at higher resolution) or 'zoom out' (at lower resolution) to look at signal components at different scales. Another advantage is that the discrete wavelet transform is computationally efficient. It is simple and fast to calculate the wavelet transform of a signal using filter bank.

2.1.1 Continuous Wavelet Transform

A wavelet function $\psi(x)$ is a localized function of zero mean [4], which satisfies the following admissibility condition.

$$\int_{-\infty}^{+\infty} \psi(x)dx = 0 \tag{2.1}$$

In practice, a wavelet function looks like a little wave with compact support, unlike the Fourier basis functions, which are infinitely long. Consider a signal $f(x) \in L^2(R)$, where $L^2(R)$ is the set of finite energy functions. The continuous wavelet transform (CWT) of $f(x)$ is defined as [4]

$$W(a,b) = \int_{-\infty}^{+\infty} f(x)\psi_{a,b}(x)dx \tag{2.2}$$

where $W(a,b)$ are the wavelet coefficients of the signal. $\psi_{a,b}(x)$ is the analyzing wavelet function with scale parameter a (> 0) and translation parameter b, defined as follows.

$$\psi_{a,b}(x) = \frac{1}{\sqrt{a}}\psi\left(\frac{x-b}{a}\right) \tag{2.3}$$

The CWT turns a signal into a function of two variables, scale and time, by adjusting the scale parameter (compress or stretch) and displacing the wavelet function over the signal. This technique can be very useful in recognizing certain characteristics of a signal, such as edges in an image. The CWT as defined by Eq. (2.2) is highly redundant. However, we can use a finite number of scales to represent a signal. This representation will be less redundant compared to the CWT definition.

An MR image is shown in Fig. 2.1 (a). This MR head image is obtained from the University of Alberta Hospital. Fig. 2.1 (a) shows one slice (512×384) of the transverse plane. There are total 176 slices in the transverse plane. As we see, different tissue classes correspond to different gray levels in this 2-D intensity image.

Fig. 2.1 (b) shows a 1-D intensity plot for row 120 of Fig. 2.1 (a). This 1-D data corresponds to the eye position. As we see, it is a noisy signal, which is typically the case for MR image data generated in an inhomogeneous magnetic field. The corresponding CWT coefficients computed for 64 scales are plotted in Fig. 2.2 (a).

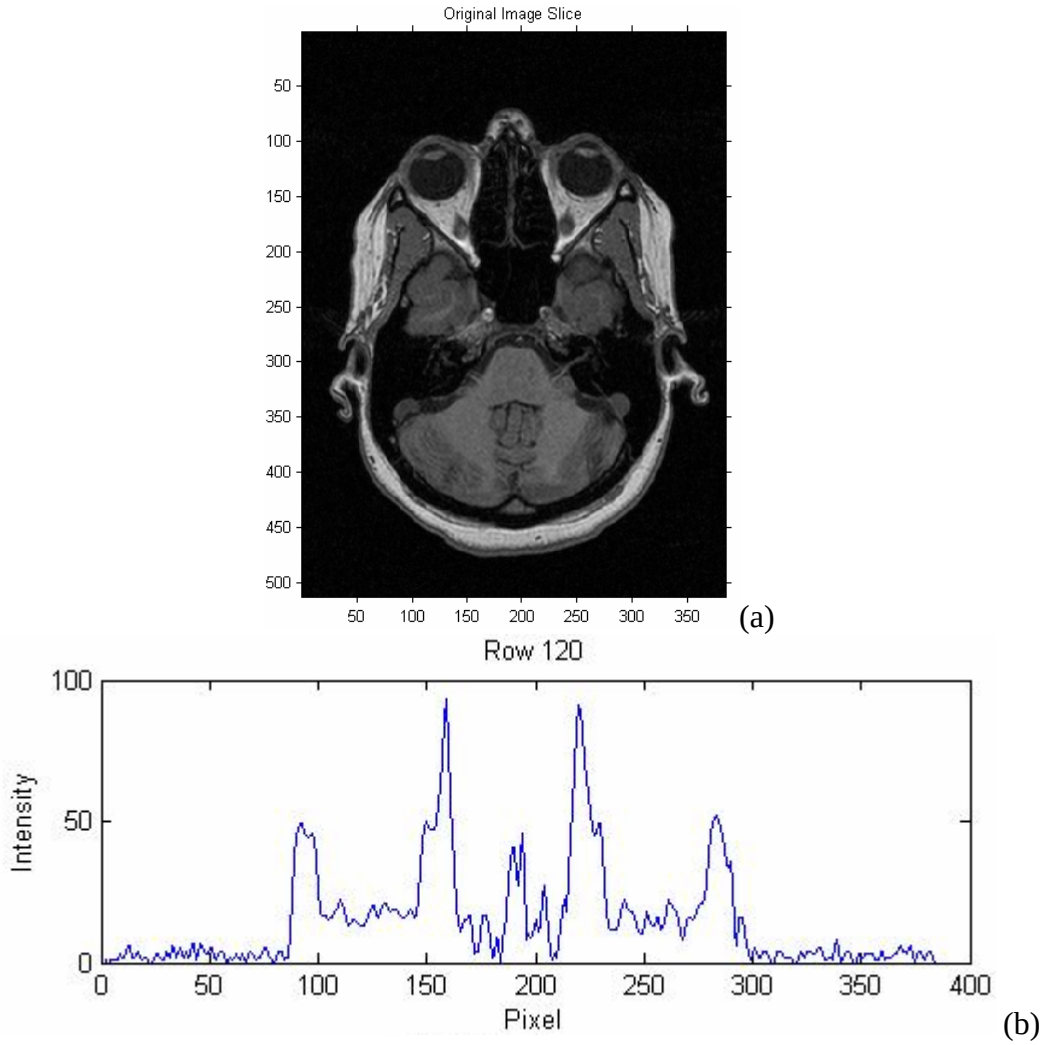


Figure 2.1: MR data.
a) One slice of MR data; b) Row 120 of Fig.2.1(a).

Fig. 2.2 (a) shows the absolute values of the wavelet coefficients for 64 scales. Brighter gray values represent large magnitude of the wavelet coefficient. Darker gray values represent small magnitude of the wavelet coefficient. The colorbar at the right side of the image represents the absolute value of the wavelet coefficients.

In Fig. 2.2 (b), the CWT coefficients are shown in 3-D mesh surface plot instead of 2-D spectrum visualization. Here we can see the beauty of the CWT. There are five mountains in the 3-D plot representing the five peaks in the 1-D intensity plot and these are the edges that we can see directly from the intensity plot.

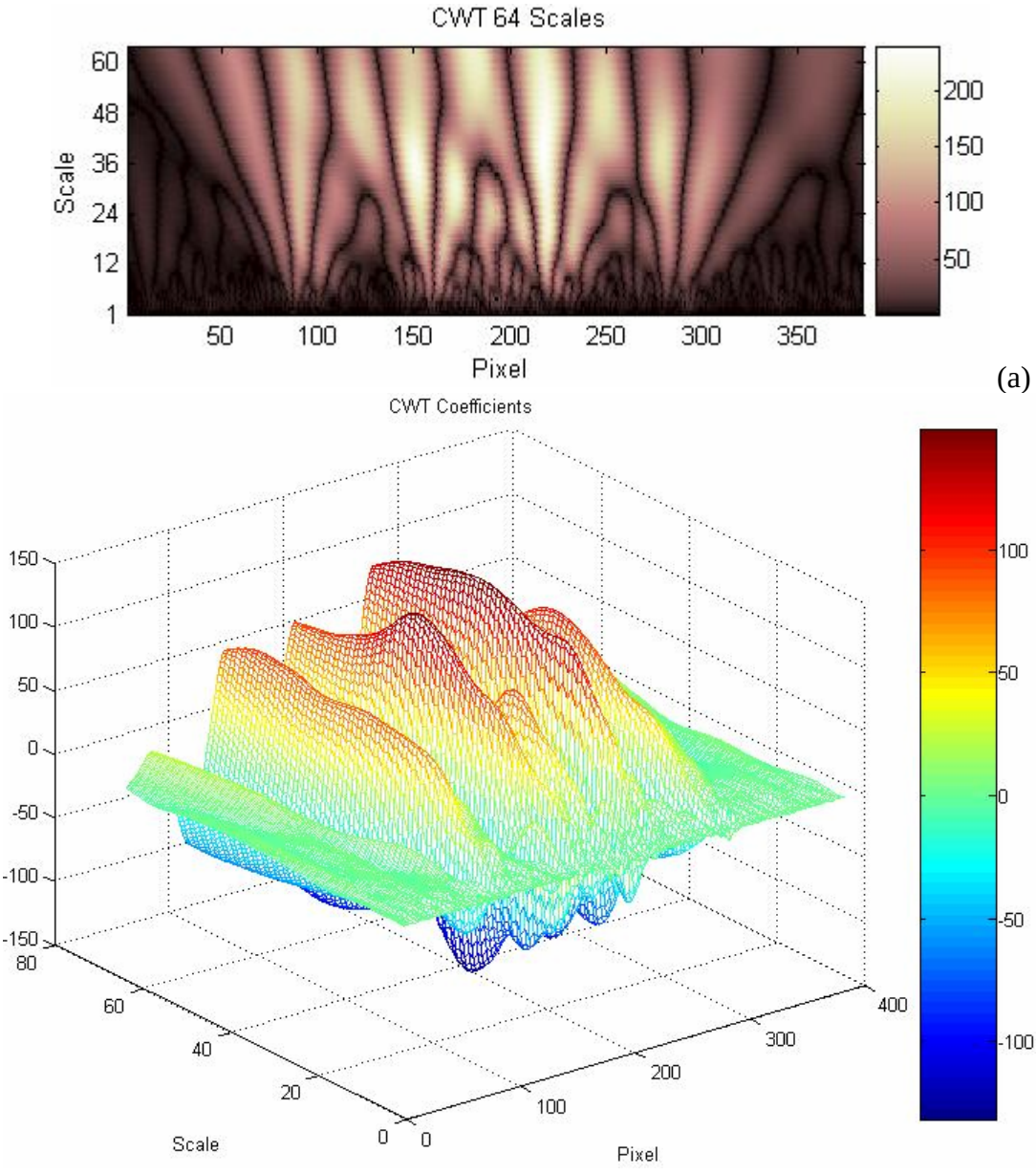


Figure 2.2: CWT coefficients of the MR data.
a) CWT coefficients of Fig.2.1 (b); b) 3D plot of Fig. 2.2 (a).

Note that for computer-based signal processing, the input data must be represented as a discrete signal, which means that the signal is measured in discrete time (or space). The 'continuous' about the CWT analysis is the set of scales and positions at which it operates. The CWT can operate at every scale, up to some maximum scale that we need. We determine this scale parameter by trading off the detail needed for the analysis and

the computational horsepower that are available for use. The CWT is also continuous in terms of translation during the computation. In other words, the analyzing wavelet function is shifted smoothly over the full length of the signal to be analyzed.

2.1.2 Discrete Wavelet Transform

If we could implement 2-D CWT for discrete image, calculating wavelet coefficients at every possible scale is a huge amount of work. It would generate a large amount of data, and would result in excessive computation time. For most applications, however, it is not necessary to keep a continuous scale parameter in the wavelet representation. Therefore, a discrete wavelet transform (DWT) has been developed where both scale parameters and the translation parameters are discreteized.

The DWT of a signal $f(x)$ can be computed by convolving the signal with wavelet function as follows [5]

$$W_s(x) = f * \psi_s(x) \tag{2.4}$$

where $W_s(x)$ and $\psi_s(x)$ are the wavelet coefficients and the wavelet function at scale *s*, respectively.

In 1989, Mallat proposed a fast wavelet decomposition and reconstruction algorithm [7]. In this algorithm, only a subset of scales is used to calculate the wavelet transform. This algorithm for DWT is widely used in signal and image processing, and is known as two-channel subband decomposition.

The two-channel DWT decomposition is obtained by convolving an image with a set of filters. The wavelet coefficients are computed through lowpass and highpass filtering. Fig. 2.3 shows the block schematic of one-stage 2-D DWT [2]. It shows decomposition of *j* level DWT coefficients into (*j*+1) level DWT coefficients. In Fig. 2.3, each block

represents an operation. The operation $\boxed{2\downarrow 1}$ downsamples the columns, where the even indexed columns are retained and the odd indexed columns are dropped. The operation $\boxed{1\downarrow 2}$ downsamples the rows, where the even indexed rows are retained and the odd indexed rows are dropped. The operation $\overset{rows}{\boxed{X}}$ means convolution of rows of the input image with filter X. The operation $\overset{columns}{\boxed{X}}$ means convolution of columns of the input image with filter X.

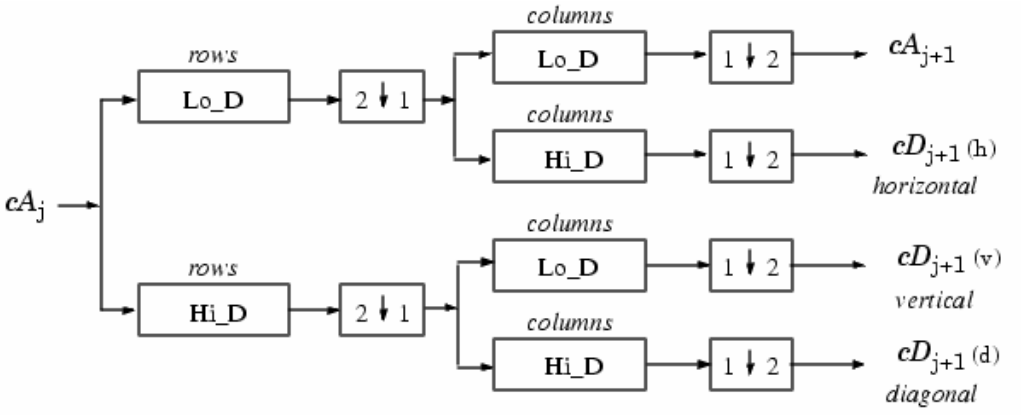


Figure 2.3: Two-dimensional DWT decomposition step.

For the first level DWT decomposition, the image pixel values I are considered to be $j = 0$ level wavelet coefficients. In other words, $cA_0 = I$. Convolving the rows and the columns of cA_0 with lowpass filter Lo_D, we can get approximation coefficients cA_1 after downsampling the rows and columns of the filter output. Similarly, convolving cA_0 appropriately with lowpass and highpass filter and after the downsampling process, we can get detail coefficients cD_1 of three directions: horizontal $cD_1(h)$, vertical $cD_1(v)$, and diagonal $cD_1(d)$.

The block schematic of one-stage 2-D DWT decomposition is shown in Fig. 2.4, where the coefficient cA_j are decomposed into approximation coefficient cA_{j+1}, and

detail coefficients $cD_{j+1}(h)$, $cD_{j+1}(v)$, and $cD_{j+1}(d)$. Since DWT calculation involves downsampling by a factor of two in both horizontal and vertical directions, after each decomposition step, it will produce four images each with one-quarter size.

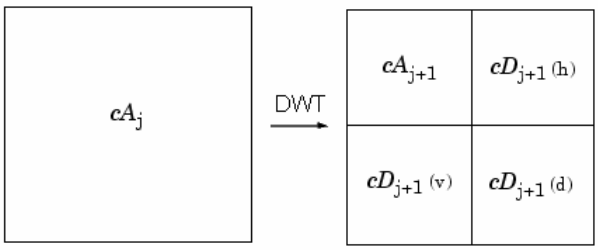
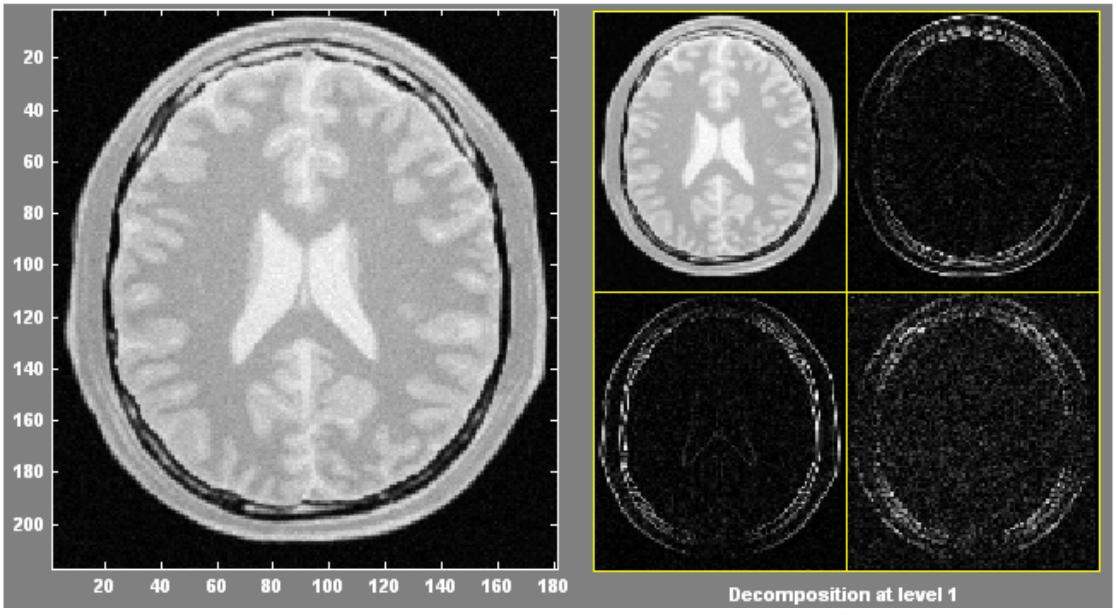


Figure 2.4: Block representation of image DWT decomposition.

The DWT decomposition of an MR image for one level is shown in Fig. 2.5. The original image is shown in Fig. 2.5 (a). After one-level DWT decomposition, the approximation coefficients cA_1 after lowpass filtering are shown as the top-left image in Fig. 2.5 (b). The detail coefficients $cD_1(h)$, $cD_1(v)$, and $cD_1(d)$ are shown as the top-right, bottom-left, and bottom-right image in Fig. 2.5 (b), respectively.



(a) (b)

Figure 2.5: DWT two-channel subband coder demonstration.
a) Original MR image; b) One-level DWT coefficients.

2.1.3 Undecimated Dyadic Wavelet Transform

Fig. 2.5 shows one-level of DWT decomposition. In image analysis, generally the lowpass subimage is further decomposed. This recursive decomposition of the lowpass subimage is known as dyadic wavelet transform. As the image size gets reduced (to one-quarter for each stage), the computational complexity gets reduced significantly. A dyadic wavelet transform developed by Mallat is based on the classical wavelet decomposition scheme [7].

Typically, a dyadic DWT uses downsampling (by a factor of 2 in both directions) in order to keep the total number of DWT coefficients same as the number of pixels in the input image. However, as the level of decomposition increases, the size of the lowpass subimages decreases. It is difficult to use these small subimages in image analysis. Therefore, in many imaging applications, the DWT coefficients are not decimated. In this thesis, we have used an undecimated dyadic wavelet transform (UDWT) for edge detection.

The UDWT for multiscale edge detection can be implemented using a filter bank algorithm [8] (known as *algorithme à trous* in French). The forward and inverse UDWT can be computed via a cascade of separable convolutions using the filters h, g, $\tilde{h}$, $\tilde{g}$ at each resolution (corresponding to scale $s = 2^j$).

Forward: $$a_{j+1}[n] = a_j * h_j[n] \tag{2.5}$$

$$d_{j+1}[n] = a_j * g_j[n] \tag{2.6}$$

Inverse: $$a_j[n] = \frac{1}{2}(a_{j+1} * \tilde{h}_j[n] + d_{j+1} * \tilde{g}_j[n]) \tag{2.7}$$

where $a[n]$ are the approximation wavelet coefficients, $d[n]$ are the detail wavelet coefficients. Note that Eq. (2.7) is for reconstruction (i.e. inverse wavelet transform), and hence it will not be used for edge detection. The h and g are the impulse response of a lowpass and highpass filter, respectively. Each time the dilated filter is obtained by inserting $2^j - 1$ zeros between each sample of $h[n]$ or $g[n]$. Inserting zeros in filters to create holes (*trous* in French) to obtain the dilated wavelet functions is the essence of this fast filter bank implementation.

This dyadic wavelet decomposition process is computed by cascading convolutions with dilated filters step by step (see Fig. 2.6). We can compute the wavelet coefficients at each scale by this filter bank algorithm [8]. For signal of N samples, the dyadic wavelet transform can be decomposed up to $\log_2 N$ scales.

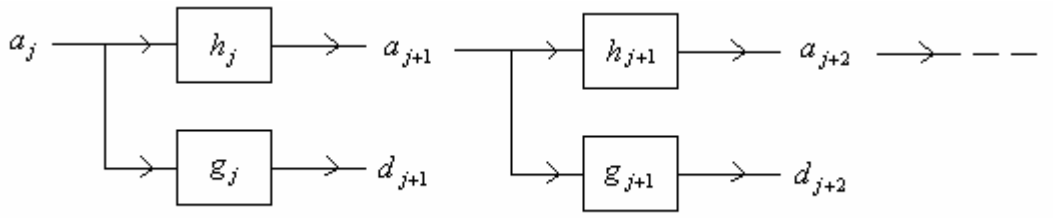


Figure 2.6: Cascade block diagram of undecimated dyadic wavelet decomposition.

Note that the main difference between the UDWT and other DWT implementations (e.g. the one described in section 2.1.2) is that the UDWT does not using downsampling after each filtering step. Because downsampling removes every other coefficients at each decomposition level, it makes the computation of the wavelet transform faster and more compact in terms of storage space. In spite of the downsampling, the original signal can be perfectly reconstructed from the DWT coefficients.

As there is no downsampling, the numbers of the wavelet coefficients do not shrink between the decomposition levels. This is very useful for better analysis and

understanding of the signal properties. For instance, in order to perform multiscale edge detection and analysis of the multiscale edges, we need to keep the wavelet coefficients at each decomposition level the same size as the original. Hence, no downsampling is more suitable for edge detection and signal characterization.

2.1.4 Wavelet Filter Design

Unlike traditional transforms such as Fourier, the DWT does not have a fixed set of basis functions. An appropriate set of DWT filters can be designed to suit an application. In the following, we present a wavelet filter that provides good performance for edge detection.

According to Mallat's wavelet design theory [8], let h and g be a pair of finite impulse response filters, where h is a lowpass filter and g is a highpass filter. Suppose that the transfer function $\hat{h}(\omega)$ of the lowpass filter satisfies $\hat{h}(0) = \sqrt{2}$. In the orthogonal wavelet basis case, we can construct a scaling function $\phi(t)$ whose Fourier transform is defined by

$$\hat{\phi}(\omega) = \frac{1}{\sqrt{2}} \hat{h}\left(\frac{\omega}{2}\right) \hat{\phi}\left(\frac{\omega}{2}\right) \tag{2.8}$$

Suppose this Fourier transform $\hat{\phi}(\omega)$ is a finite energy function, i.e., $\phi \in L^2(R)$. The corresponding wavelet function $\psi(t)$ has a Fourier transform defined by

$$\hat{\psi}(\omega) = \frac{1}{\sqrt{2}} \hat{g}\left(\frac{\omega}{2}\right) \hat{\phi}\left(\frac{\omega}{2}\right) \tag{2.9}$$

The Fourier transform $\hat{g}(\omega)$ of the highpass filter g is given by

$$\hat{g}(\omega) = -i\sqrt{2}\sin\left(\frac{\omega}{2}\right)\exp\left(\frac{-i\omega}{2}\right) \tag{2.10}$$

From Eq. (2.10), we see that $\hat{g}(\omega)$ is the Fourier transform of a finite difference filter, and it has a discrete approximation of a derivative.

$$g[n] = \begin{cases} -1/\sqrt{2} & n = 0 \\ 1/\sqrt{2} & n = 1 \\ 0 & otherwise \end{cases} \tag{2.11}$$

Both scaling function $\phi(t)$ and wavelet function $\psi(t)$ are compact support (i.e., non-zero in a finite time interval, energy concentrated mostly at low frequencies), since *h* and *g* are the filters with a finite number of non-zero coefficients.

The quadratic spline wavelet is suggested in Mallat's paper [5] for fast computation of the dyadic wavelet transform. The Fourier transform of a box spline of degree *m* is [8]

$$\hat{\phi}(\omega) = \left(\frac{\sin(\omega/2)}{\omega/2}\right)^{m+1}\exp\left(\frac{-i\varepsilon\omega}{2}\right) \quad \text{with} \quad \varepsilon = \begin{cases} 1 & m \ is \ even \\ 0 & m \ is \ odd \end{cases} \tag{2.12}$$

The spline is centered at $n = 1/2$ if *m* is even and at $n = 0$ if *m* is odd.

$$\hat{h}(\omega) = \sqrt{2}\left(\cos\left(\frac{\omega}{2}\right)\right)^{m+1}\exp\left(\frac{-i\varepsilon\omega}{2}\right) \tag{2.13}$$

The coefficients of the filter *h* and *g* can be calculated from their transfer functions defined in Eq. (2.13) and (2.10). Table 2.1 shows an example of the quadratic spline wavelet filter coefficients [8] computed for $m = 2$.

Table 2.1: Quadratic spline wavelet filter coefficients.

n	-1	0	1	2
h[*n*]	0.1768	0.5303	0.5303	0.1768
g[*n*]		-0.7071	0.7071	

2.1.5 Regularity Analysis & Lipschitz Exponent

In this section, the relationship between the Lipschitz exponent and wavelet transform will be described. The wavelet transform can localize signal structures with a zooming procedure by adjusting the scale parameter. The local signal regularity is characterized by the decay of the wavelet transform across scales, and hence the singularities (edges) can be detected by following the wavelet transform local maxima across scales [6].

The Lipschitz exponents provide uniform regularity measurements over sample intervals. If a signal has a singularity (edge) at a point *v*, the Lipschitz exponent at that point characterizes the singular behavior.

The Lipschitz exponent is the mathematical foundation of regularity analysis. A wavelet transform has a sequence of local maxima, and the Lipschitz regularity can be calculated from the decay of the modulus maxima [5, 6]. It is possible to precisely quantify the local regularity of a signal to characterize singular structures.

For a signal $f(x)$, the Lipschitz exponent α satisfies the following [5].

$$|W_s(x)| \le As^{\alpha} \tag{2.14}$$

where $W_s(x)$ are wavelet coefficients at scale *s*, A ($A > 0$) is some constant (i.e., A is not a function of *s*). Eq. (2.14) states that the modulus of the wavelet coefficients of a signal vary with the scale *s* according to the Lipschitz regularity of that signal.

If the Lipschitz exponent α is positive, Eq. (2.14) implies that the amplitude of the wavelet modulus $|W_s(x)|$ should increase when the scale *s* increases. On the contrary, if

the Lipschitz exponent α is negative that the amplitude of the wavelet modulus $|W_s(x)|$ should decrease when the scale *s* increases.

The characterization of edges by Lipschitz exponent and wavelet transform has been explained by Mallat et al. [5, 6]. It has been shown that positive Lipschitz exponent α corresponds to edge structure (i.e. step edge) and negative Lipschitz exponent α corresponds to Dirac structure (i.e. spike noise).

2.2 Multiscale Edge Detection

Edge detection is generally used to determine an object boundary. Many edge detection operators, such as Sobel, Prewitt, and Canny, will miss edge or produce false edges due to the noise in the image. It has been shown recently that wavelet techniques can provide better edge detection performance as is more immune to noise compared to other techniques. In the following we present a brief review of wavelet based multiscale edge detection.

In computer vision, Marr and Hildreth [9] first introduced the concept of multiscale edge detection for detecting the contours of small structures as well as the boundaries of larger objects in an image. The image is smoothed with a Gaussian filter before edge detection. The scale defines the size of the neighborhood where the signal changes are computed.

Mallat and Zhong [5] demonstrated that multiscale edge detection can be performed in a wavelet framework and can provide a deeper understanding of these multiscale algorithms. They studied the properties of multiscale edges through wavelet theory, and showed that the evolution of wavelet local maxima across different scales characterizes

the local shape of irregular structures. In addition, the authors produced a compact coding algorithm to reconstruct signals from their multiscale edges.

Park et al. [11] proposed a multiresolution edge detection technique for noisy images contaminated by additive Gaussian noise or multiplicative noise. The technique automatically determines the resolution by defining a discontinuity measurement. The mode of the discontinuity measurement is calculated over a window to determine the resolution of the edge detector.

Zhang and Bao [12] proposed a wavelet based edge detection scheme based on scale multiplication. The wavelet transforms at two adjacent scales are multiplied to magnify the edge structures and suppress the noise. Unlike other algorithms, it determines the edges as the local maxima directly in the scale product after an efficient thresholding. It achieves better results on the localization performance in natural images, but requires a properly determined threshold to suppress the noise maxima.

Li [13] interpreted classical edge detectors from the view point of wavelet transform, and developed a multiscale wavelet model for noisy image edge detection.

Tang et al. [14] studied the characterization of edges with wavelet transform and Lipschitz exponents. Different edge structures are identified, and an algorithm with the modulus-angle-separated wavelet was developed to extract step-structure edges from a multi-structured edges.

Berkner and Wells [15] proposed an approximation of a CWT. The approximate CWT was developed based on a hierarchical scheme, which also preserves the characteristics of singularities, and leads to applications in multiscale edge detection.

Wavelet based edge detection has been shown to have less sensitivity to noise than

other techniques. Most of the techniques (e.g. [11-15]) are primarily based on the classical multiscale edge detection method proposed by Mallat [5, 6]. In this thesis, our proposed method is also based on Mallat's algorithm. Therefore, we present the details of Mallat's edge detection technique in the following.

2.2.1 Edge Detection using Local Maxima

The thesis work is developed based on Mallat's multiscale edge detection technique. Therefore, in this section, we will review Mallat's technique [5] in detail. We will explain the procedure to find multiscale edges from the local maxima of the wavelet transform.

Based on wavelet theory and Canny [10] edge detection theory, Mallat and Zhong [5] developed a multiscale edge detection theory. It has been shown that multiscale edge detection can be implemented by smoothing a signal with a convolution kernel at various scales, and detecting sharp variation points as edges. Let us first choose a smoothing function $\theta(x)$ that satisfies the following two conditions.

$$\lim_{|x|\to\infty} \theta(x) = 0 \tag{2.15}$$

$$\int_{-\infty}^{+\infty} \theta(x)dx = 1 \tag{2.16}$$

A classic example of a smoothing function is the Gaussian function. We can generate a wavelet function by calculating the derivative of $\theta(x)$ as follows.

$$\psi(x) = \frac{d\theta(x)}{dx} \tag{2.17}$$

Note that $\psi(x)$ can be considered as a wavelet, because its integral is equal to zero, and therefore it satisfies the wavelet condition given by Eq. (2.1). Now we generate a wavelet function $\psi_s(x)$ from $\psi(x)$ at scale *s* as follows.

$$\psi_s(x) = \frac{1}{s}\psi\left(\frac{x}{s}\right) \tag{2.18}$$

Similarly, we can also denote a smoothing function $\theta_s(x)$ at scale *s* as follows.

$$\theta_s(x) = \frac{1}{s}\theta\left(\frac{x}{s}\right) \tag{2.19}$$

Thus, from Eq. (2.17) and (2.19), it can be shown that

$$\psi_s(x) = s\frac{d\theta_s(x)}{dx} \tag{2.20}$$

As shown in Eq. (2.2), the wavelet transform is computed by convolving a signal with dilated wavelet functions. Therefore, we can define the wavelet transform of a signal $f(x)$ at scale *s* and position *x* with respect to wavelet function $\psi_s(x)$ as follows.

$$W_s(x) = f * \psi_s(x) \tag{2.21}$$

Thus, from Eq. (2.20) and (2.21), we can derive that

$$W_s(x) = f * \psi_s(x) = f * \left(s\frac{d\theta_s}{dx}\right)(x) = s\frac{d}{dx}(f * \theta_s)(x) \tag{2.22}$$

Similar concept can be extended to 2-D image analysis. An image $f(x,y)$ can be smoothed by convolution with some smoothing function, $\theta_s(x,y)$, at different scales, *s*. The convolution processing is similar to computing the gradient vector, $\vec{\nabla}(f * \theta_s)(x,y)$. The direction of the gradient vector at a point (x_0, y_0) indicates the direction in the image plane (x,y) along which the directional derivative of $f(x,y)$ has the largest absolute value. We can define an edge as a collection of points where the modulus of the gradient vector is maximum.

This gradient calculation part is similar to other edge detectors, but note that we are

computing the modulus of the gradient vectors at each scale, *s*, as the convolution kernel $\theta_s(x)$ is dilated at each scale. This type of multiscale gradient calculation is not done by other edge detectors, such as Canny operator where the convolution is done only once with a fixed convolution kernel.

We can relate this modulus computation processing to the 2-D wavelet transform. Similar to the Eq. (2.21) and (2.22), the wavelet transform of image $f(x, y)$ at the scale *s* has two components that are defined as

$$W_s^1(x, y) = f * \psi_s^1(x, y) \tag{2.23}$$

$$W_s^2(x, y) = f * \psi_s^2(x, y) \tag{2.24}$$

and can be expressed in the form as follows.

$$\begin{pmatrix} W_s^1(x, y) \\ W_s^2(x, y) \end{pmatrix} = s \begin{pmatrix} \frac{\partial}{\partial x}(f * \theta_s)(x, y) \\ \frac{\partial}{\partial y}(f * \theta_s)(x, y) \end{pmatrix} = s \vec{\nabla}(f * \theta_s)(x, y) \tag{2.25}$$

Note that the two wavelet transform components in Eq. (2.25) are proportional to the gradient vector of the image smoothed by $\theta_s(x, y)$. Edge points can be located from these two components. At each scale *s*, the modulus of the gradient vector is called the *wavelet transform modulus*, and it is defined as

$$M_s(x, y) = \sqrt{\left|W_s^1(x, y)\right|^2 + \left|W_s^2(x, y)\right|^2} \tag{2.26}$$

and the angle of the wavelet gradient vector for each scale is

$$A_s(x, y) = \tan^{-1}\left(\frac{W_s^2(x, y)}{W_s^1(x, y)}\right) \tag{2.27}$$

The local extrema of the wavelet coefficients at each scale corresponds to the

inflection point of the convolution. We know that an inflection point of $f * \theta_s(x, y)$ can either be a maximum or a minimum value of the derivative, and hence the local maxima are the sharp variation points of $f * \theta_s(x, y)$. We now discuss a method to find edges where the wavelet transform modulus is local maximum along the angle direction.

Wavelet Transform Modulus Maxima

Consider a 1-D signal $f(x)$, and assume that the wavelet modulus maximum occurs at a point x_0, i.e., $|W_s(x)|$ is locally maximum at $x = x_0$. This implies that

$$\frac{dW_s(x_0)}{dx} = 0 \tag{2.28}$$

Note that we can find the local maximum (edge) in either the left or right neighborhood of x_0, and it should be a strict local maximum that avoids any local maxima when $|W_s(x)|$ is constant [8]. Therefore, by comparing the left or right neighborhood of each modulus point, we can decide whether this point is a local maximum or not. If the point is a local modulus maximum in the one-dimensional neighborhood of the left or right modulus points, it is an edge point of that scale.

For example, a 1-D signal $f(x)$ has wavelet modulus $|W_{s_1}(x)|$ at scale s_1. The $|W_{s_1}(x)|$ at a point x is local maximum, if and only if $|W_{s_1}(x)| > |W_{s_1}(x-1)|$ and $|W_{s_1}(x)| > |W_{s_1}(x+1)|$.

For a 2-D image, the wavelet transform modulus maxima (WTMM) are located at the points where wavelet transform modulus (defined in Eq. (2.26)) is larger than its two neighbors in one-dimensional neighborhoods along the angle direction. Note that in the 2-D case, there are eight pixels around one pixel in the neighborhood. Thus we divide the

local neighborhood into four one-dimensional directions (quantize the angle into 0°, 45°, 90°, and 135°). For example, see the pixel grid shown in Fig. 2.7. In order to decide the 2-D WTMM, we need to first check the wavelet angle of each pixel and then check if the pixel is a local maximum in that angle direction. To better explain WTMM finding procedure, we give the pseudo code as follows.

$M_s(x, y)$ *is a WTMM if any of the following conditions is true:*

$A_s(x, y) \approx 0$ & $\{M_s(x, y-1) < M_s(x, y) > M_s(x, y+1)\}$;

$A_s(x, y) \approx \pi/4$ & $\{M_s(x+1, y-1) < M_s(x, y) > M_s(x-1, y+1)\}$;

$A_s(x, y) \approx \pi/2$ & $\{M_s(x-1, y) < M_s(x, y) > M_s(x+1, y)\}$;

$A_s(x, y) \approx 3\pi/4$ & $\{M_s(x-1, y-1) < M_s(x, y) > M_s(x+1, y+1)\}$.

1 (x-1,y-1)	**2** (x-1,y)	**3** (x-1,y+1)
4 (x,y-1)	**5** (x,y)	**6** (x,y+1)
7 (x+1,y-1)	**8** (x+1,y)	**9** (x+1,y+1)

Figure 2.7: Nine pixel grid.
(pixel X-Y coordinates are shown in brackets)

For example (see Fig. 2.7), if we want to decide whether pixel 5 is a local maximum or not, suppose the angle at pixel 5 is 0° or nearest to 0°, then check the wavelet modulus of pixel 4, 5 and 6. If the modulus of pixel 5 is larger than the modulus of two other pixels, we keep pixel 5 as an edge pixel.

Theoretically, the 2-D WTMM are determined by finding the pixels where the modulus is larger than its two neighbors in one-dimensional neighborhood along the angle direction of that pixel.

1-D WTMM

We can connect all modulus maxima points across scales in the scale-space plane to obtain the maxima curves. Any connected curve along which all points are modulus maxima is called a maxima line [8]. Using the WTMM theory presented above, we can show all the individual wavelet modulus maxima across scales to form a maxima curve that follows an edge through scales. Let us consider an example of 1-D WTMM in the real world (still using the 1-D signal from Fig. 2.1 (b) for demonstration). Fig. 2.8 shows the WTMM of the 1-D signal.

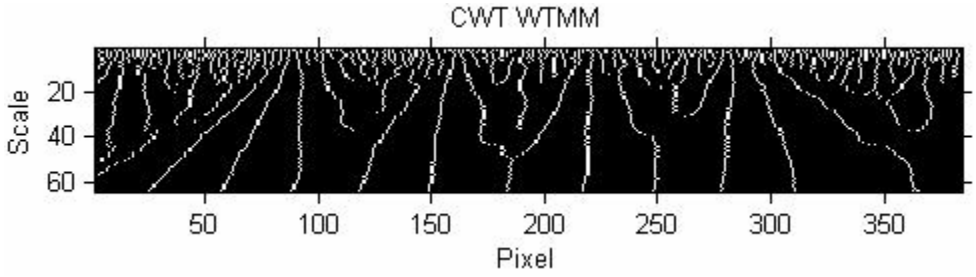


Figure 2.8: 1-D WTMM.

Fig. 2.8 shows the WTMM calculated from the CWT coefficients for all 64 scales. The WTMM are found among the adjacent columns (left and right neighborhood), where the middle pixel is the maxima among the three pixels. The WTMM is shown as white pixels in the plot. As we see, there are more maxima points at the lower scales. This is because lower scales correspond to higher resolution, higher scales correspond to lower resolution. Lower scales will result in lots of noise and edges. Higher scales will result in less noise and fewer edges.

Instinctively, we know there are connections (correlation) between wavelet scales, and hence it is necessary to chain the maxima points from the maxima curve across the scales to indicate where the true edge is.

UDWT for Edge Detection

A discrete dyadic wavelet transform proposed by Mallat for multiscale edge detection (see section 2.1.3 and 2.1.4) assumes that the scale varies only along the dyadic sequence $(2^j)_{j\in Z}$, where *Z* is the set of integers. Similar to Eq. (2.18), a wavelet function whose average is zero, with dilation by a factor 2^j is denoted by $\psi_{2^j}(x)$.

$$\psi_{2^j}(x) = \frac{1}{2^j}\psi\left(\frac{x}{2^j}\right) \tag{2.29}$$

The wavelet transform of $f(x)$ at scale 2^j and at the position *x* is defined by the convolution product.

$$W_{2^j}(x) = f * \psi_{2^j}(x) \tag{2.30}$$

We can simply rewrite Eq. (2.18-2.28) from above sections for multiscale edge detection for dyadic wavelet transform using scale $s = 2^j$. The multiscale edge detection can be implemented by the filter bank algorithm described in section 2.1.3.

2.3 Multiscale Image Segmentation

A large number of techniques have been proposed in the literature to segment medical images. A simple method to segment an image is to apply threshold and divide regions of different intensity. Region growing can be applied starting from a seed in the image, checking the pixels against a predefined rule to allow the region to grow, and then define the object region. Region split/merge can be operated in a recursive way, check for homogeneity of pixel intensities, if the pixels are not all of similar intensity then the region splits into equal-sized subsections. A merge step is performed to aggregate the subregions that have similar intensity levels.

A major difficulty of image segmentation is the presence of noise in the image. The wavelet technique has been shown to be less sensitive to noise. Therefore, edge-based and region-based segmentation have been extended to include multiscale technique for superior performance.

Raman et al. [16] proposed an algorithm for computationally focusing the tissue boundaries in MR images at different scales. The point-based, scale-space search is performed from low to high resolution. A new edge detector is proposed which performs better than the commonly used Laplacian of Gaussian operator. A similarity function is used to determine the scale space step size. It is the function of the edge operator to predict edge behavior at one scale to give knowledge of the edge behavior at another scale.

Vincken et al. [17] proposed a multiscale segmentation method with a probabilistic linking approach. A multiscale data structure is constructed by convolving the original image with a Gaussian kernel of increasing width. A child-parent linkage scheme is established between pixels at adjacent scales. In the resulting tree-like data structure, roots are formed to indicate locations in scale space where segments in the image are represented. The final segmentation is obtained by tracing back the linkages for all roots.

Rezaee et al. [18] presented an unsupervised image segmentation technique, which combines pyramidal image segmentation with fuzzy c-means clustering algorithm. A pyramid layer is split into a number of regions by a root labeling technique, and fuzzy c-means technique is used to merge the regions of the layer. A cluster validity function is used to automatically find the optimal number of objects.

Grau et al. [19] presented a general image segmentation system for hierarchical analysis. The system uses a tree representation of the original image, where the image regions are assigned to tree nodes. A correspondence process with a model tree is followed, which embeds a priori knowledge about the images. It performs the minimization of an error function that quantifies the difference between the input image tree and the model tree. It also automatically calculates the model tree from a set of manually segmented images.

Edge-based segmentation (i.e. [16]) and region-based segmentation (i.e. [17-19]) based on multiscale information are both active research areas in medical imaging. There are many papers published in image segmentation area, but all have their own limitations (e.g. operator involvement) to achieve fully satisfactory automatic segmentation.

2.4 Auction Algorithm

In this thesis, we will use an optimization method to efficiently determine 2-D WTMM chain. We choose an auction algorithm to do this. Here, we give a brief description of the auction algorithm.

The auction algorithm [20] introduced by Bertsekas is a method for optimally solving the assignment problem. Suppose there are *n* persons and *n* objects that we want to match on a one-to-one basis. There is a benefit or value a_{ij} for matching person *i* with object *j*, and we need to assign persons to objects to maximize the total benefit.

The process to find an equilibrium assignment is in repeating a sequence of iterations until each person has an assigned object. This process can be viewed as an auction where at each iteration the bidder *i* raises the price of a preferred object by the

bidding increment r_i. This bidding increment cannot be negative, and hence the object price can increase. Also it cannot be zero, because if r_i is zero when two or more objects offer maximum value for the bidder *i*, a situation may be created where several persons contest a smaller number of equally desirable objects without raising their prices, thereby creating a never ending cycle. Thus, learn from real auctions where each bid for an object must raise its price by a minimum positive increment, and bidders must on occasion take risks to win their preferred objects, therefore a positive scalar ε (minimum positive increment) for the bidding increment r_i is fixed.

The object *j* has a price p_j means the person who receives that object must pay the price p_j. The value of object *j* for person *i* is $a_{ij} - p_j$, and each person *i* would logically want to be assigned to an object j_i with maximal value. Suppose that a person *i* is almost happy with an assignment, and a set of prices if the value of its assigned object j_i is within ε of being maximal, that is

$$a_{ij_i} - p_{j_i} \geq \max_{j=1,\ldots n}\left\{a_{ij} - p_j\right\} - \varepsilon \tag{2.31}$$

where $a_{ij_i} - p_{j_i}$ means one of the value of the object j_i for that person *i*. We will say that an assignment and a set of prices are almost at equilibrium when all persons are almost happy, if we can satisfy the condition (Eq. (2.31). This condition is known as ε-complementary slackness. Note that the bidding increment is always at least equal to ε.

At each iteration, there is a price for each object. One or more unassigned persons bid simultaneously for their best objects, which is the one offering maximum benefit minus price. The object is given to the highest bidder, thereby raising the corresponding prices. A sequence of iterations is performed until each person is assigned and the

algorithm terminates. Auction algorithm has been shown to have better performance than other competing algorithms [20]. If the number of objects *n* is larger than the number of persons *m*, it is an asymmetric assignment problem. We still want one person to assign one object to maximize the total benefit. A variation of the auction algorithm from Bertsekas can solve this type of assignment problem [21].

2.5 Fuzzy Logic

In this thesis, we will use fuzzy logic to give a quality evaluation of the Lipschitz regularity analysis result. Therefore, here we briefly present the fuzzy logic.

Zadeh [22] first introduced the concepts of fuzzy sets and fuzzy logic. Fuzzy sets are extension of classical set theory. Fuzzy logic is a superset of conventional (Boolean) logic. It is derived from fuzzy set theory to deal with approximate rather than precise logic [23]. Fuzzy logic has been used to handle the concept of partial truth between 'completely true' and 'completely false'.

A fuzzy subset *F* of a set *S* can be defined as a set of ordered pairs, each with a first element that is an element of the set *S*, and a second element is a value in the interval [0, 1], with exactly one ordered pair present for each element of *S*. This defines a mapping between elements of the set *S* and values in interval [0, 1]. The value 0 represents complete non-membership (completely false), whereas the value 1 represents complete membership (completely true). The values in between represent intermediate degrees of membership (partial true). The mapping process is described by a function called the membership function of *F*. A membership function is a curve that defines how each element in the set *S* is mapped to a membership value between 0 and 1. The set *S* is often referred to as the universe of discourse [24].

The standard definition of fuzzy logic operators are introduced by Zadeh [23]:

$$A \,\&\, B \Leftrightarrow \min(A, B) \tag{2.32}$$

$$A \mid B \Leftrightarrow \max(A, B) \tag{2.33}$$

$$\overline{A} \Leftrightarrow 1 - A \tag{2.34}$$

where '&' is the AND operator, ' | ' is the OR operator, ' $\overline{\ }$ ' is the NOT operator, and ' $\Leftrightarrow$ ' means equivalence. The left side of the equations is Boolean logic and the right side is fuzzy logic. There are more fuzzy logic operators such as those derived by Dombi [25].

There are many types of membership functions. The graphs of the membership functions may have different shapes. The simplest one is formed using straight lines, for example, the linear function (see Fig. 2.9 (a)). A more complex function use triangular shape (see Fig. 2.9 (b)).

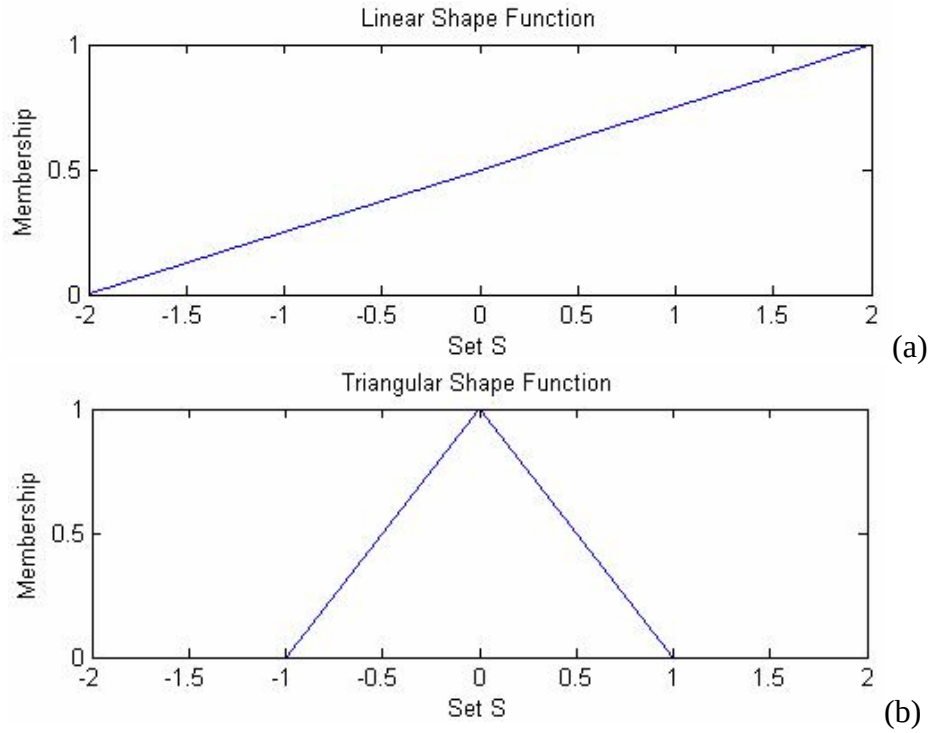


Figure 2.9: Examples of membership functions.
a) Linear shape function; b) Triangular shape function.

A fuzzy logic system involves a process of formulating the mapping from a given input and combines fuzzy rules to produce an output (see Fig. 2.10). Basically, the input and output of the fuzzy logic system can range from $-\infty$ to $+\infty$, but the intermediate results after membership functions and fuzzy logic operators should all be in [0, 1].

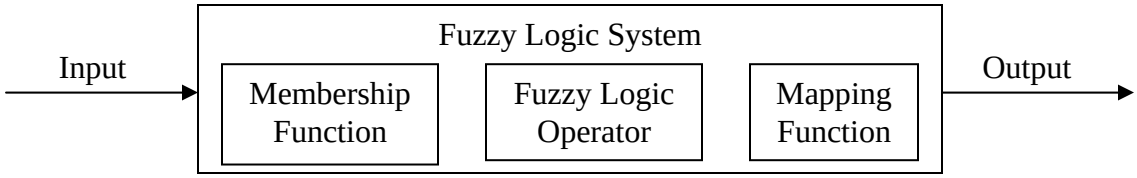


Figure 2.10: Block diagram of a basic fuzzy logic system.

2.6 mtrack Algorithm

The mtrack software [1] has been developed at the University of Alberta for medical image segmentation, especially for MR images of the head. As shown in Fig. 2.11, it includes three major modules: edge detection, edge feature extraction, and edge tracing. The three modules are explained below.

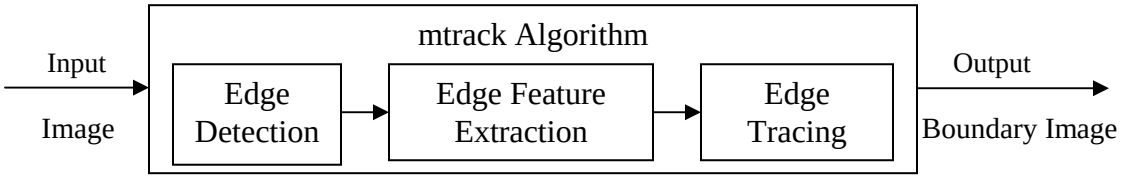


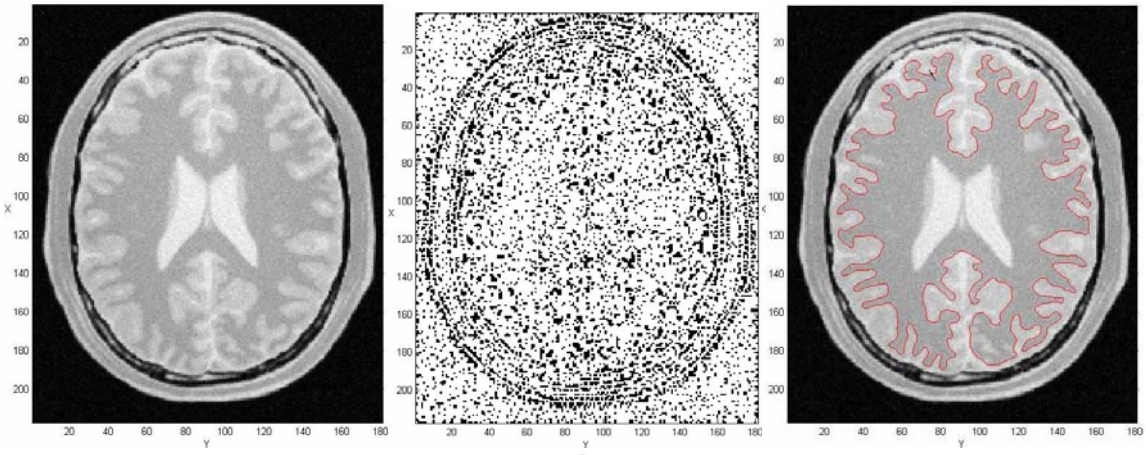
Figure 2.11: Block diagram of mtrack algorithm.

1. Edge detection: In this module, the edge of an image is obtained by Canny edge detector. The detector is applied for six operator directions. Subpixel edge resolution is obtained by linearly interpolating the zero crossings in the second derivative of a Gaussian filtered image.
2. Edge Feature Extraction: Edge features are extracted from the detected edges. For each edge point along the operator direction, the top and bottom intensity points of that edge slope are obtained by examining the 1^{st} and 2^{nd} derivatives curves. The

coordinates of the subpixel edge points and the top and bottom intensity on each side of the edge points are combined as four-dimensional edge information.

3. Edge Tracing: A target tracking algorithm is used to link the edge points to form an object boundary. The edge information is assumed to be position information of a hypothetical object (target) that moves along the boundary. Target tracking starts from a starting point, follows the path of the target (edge) until no further edge point can be founded on the track or the starting point is revisited. Thus, the boundary can be drawn by linking all the edge points that are found in the path of the target, including the starting point.

An example of segmentation using mtrack software is shown in Fig. 2.12 for edge tracing on the gray matter /white matter boundary in a head MR image. Fig. 2.12 (a) is the input MR image to the mtrack software. Fig. 2.12 (b) is the binary edge image after edge detection. Fig. 2.13 (c) is the output boundary image after tracing the edge from a starting point. The boundary is plotted on the MR image that separates the white matter (inner side of the boundary) and the gray matter (outer side of the boundary).



(a) (b) (c)

Figure 2.12: Example of mtrack software.
a) Original input MR image; b) Edge image;
c) Output image with gray/white boundary.

This edge-based image segmentation method can perform automatic tracking and can also manually cut and link tracks to form the desired contours. However, some problems remain: First, the tracking performance depends on the starting point. Second, it depends on the track direction. The tracking algorithm also involves the possibility of self intersection, the selection of appropriate parameter values, and the possibility that a closed contour is not formed in all cases. The main limitation of this algorithm (as we have discussed in section 1.2) is that it is sensitive to noise.

This thesis work is focused on multiscale boundary identification to achieve better edge-based image segmentation. The edge quality obtained from the multiscale technique will help the edge tracing algorithm to give better tracking performance.

2.7 Summary

In this chapter, we presented a review of background work related to this thesis. We briefly reviewed the recent advances in multiscale edge detection and segmentation. Both edge-based segmentation and region-based segmentation have their own limitations for practical applications. The research will continue in the multiscale boundary identification direction for better edge-based segmentation. Wavelet theory has been generally reviewed in section 2.1. Mallat's multiscale technique is discussed in section 2.2.1 for edge detection. Other work related to this research, such as the auction algorithm, fuzzy logic and medical image segmentation software that will be used in this research have been also described in section 2.4-2.6. The proposed method for multiscale boundary identification will be developed based on these algorithms.

Chapter 3

Proposed Method

We presented a brief review of the recent advances in multiscale edge detection and segmentation in chapter 2. Wavelet theory and multiscale techniques for edge detection and signal characterization were reviewed. The research is focused on the multiscale edge detection and boundary identification for achieving better edge-based segmentation. In this chapter, we propose a multiscale method for boundary identification in MR images.

The organization of the chapter is as follows. Section 3.1 provides the schematic of the proposed work. Section 3.2 presents the multiscale edge detection method and section 3.3 presents boundary identification method. The integration of the proposed work into the medical image segmentation software is presented in section 3.4. Summary is given in section 3.5.

3.1 Scheme of Proposed Work

The proposed method is developed based on the classical wavelet theory for multiscale edge detection and boundary identification. It has been applied to the mtrack software to see if the proposed work is useful for edge-based image segmentation. The schematic of the method integrated into the mtrack software is shown in Fig. 3.1.

A comparison between Fig. 3.1 and Fig. 2.11 reveals that the new block diagram replaces Canny edge detection with multiscale edge detection, and adds multiscale boundary identification before edge tracing. The two new modules in Fig. 3.1 are explained below.

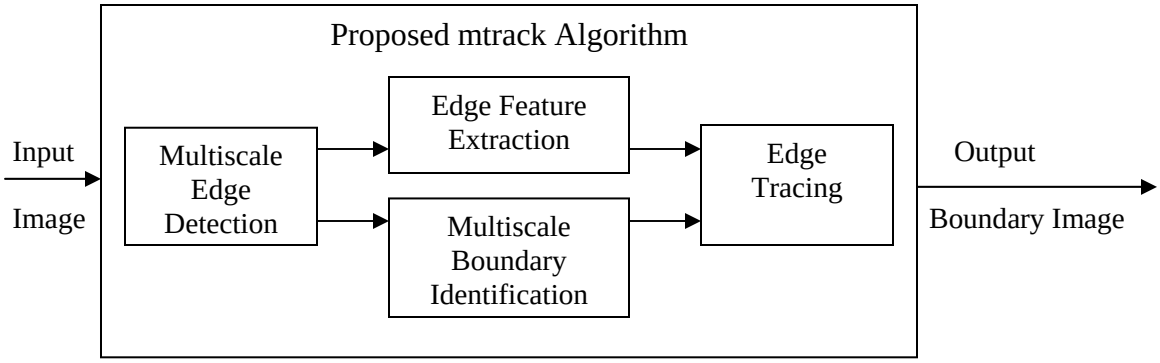


Figure 3.1: Block diagram of the proposed mtrack algorithm.

1. Multiscale Edge Detection: Mutliscale technique based on Mallat's classical wavelet theory will be used for edge detection. Directional multiscale edge detection will be used, together with subpixel edge resolution.
2. Multiscale Boundary Identification: From multiscale representation, the wavelet transform modulus maxima chain will be obtained. The chain connection represents how deep the edge is connected through the wavelet decay. Lipschitz regularity analysis will be used to enhance the boundary identification performance by further separating the noise from the true edge. Fuzzy logic will be used to perform edge quality evaluation from the results of Lipschitz regularity analysis.

In the following sections, we present the new modules in detail.

3.2 Multiscale Edge Detection

The multiscale edge detection algorithm proposed by Mallat et al. [5] was reviewed in section 2.2.1. In this thesis, we use this algorithm to calculate multiscale edges. In the following subsections (3.2.1-3.2.3), we present the implementation details of the multiscale edge detection algorithm. Sections 3.2.4 and 3.2.5 present the method to obtain directional multiscale edges with subpixel resolution.

3.2.1 UDWT

The first step of multiscale edge detection is to calculate the wavelet transform. We use the UDWT (reviewed in section 2.1.3) for this purpose. The block diagram of UDWT decomposition for two stages is shown in Fig. 3.2.

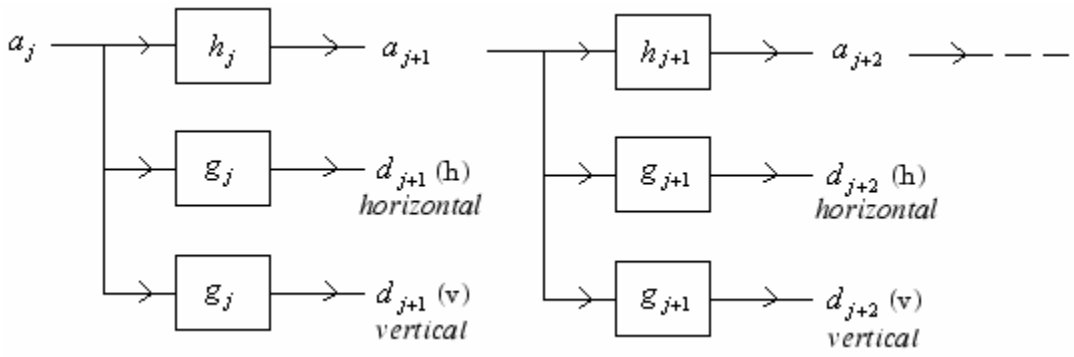


Figure 3.2: UDWT decomposition steps.

The dyadic wavelet coefficients are computed through lowpass and highpass filtering. In Fig. 3.2, each block represents an operation. Note the (h) and (v) in Fig. 3.2 indicate the highpass filtering for horizontal and vertical coefficients, respectively. For the first level decomposition, the image I is used as the initial set of wavelet coefficient a_j, i.e., $a_0 = I$ for $j = 0$. Convolving a_0 with a 2-D lowpass filter h_1, we obtain the approximation coefficients a_1. Convolving a_0 with highpass filter g_1, we obtain the detail coefficients $d_1(h)$ in vertical direction and $d_1(v)$ in horizontal direction. The approximation coefficients a_1 obtained in first level decomposition becomes the input of the second level decomposition.

The lowpass and highpass filter coefficients h_1 and g_1 for the dyadic wavelet decomposition are adopted from Table 2.1. The 2-D convolution mask of the lowpass filter h_1 for computing the approximation wavelet coefficients is calculated as

$$h_1[n,m] = \begin{bmatrix} 0.1768 \\ 0.5303 \\ 0.5303 \\ 0.1768 \end{bmatrix} \times \begin{bmatrix} 0.1768 & 0.5303 & 0.5303 & 0.1768 \end{bmatrix}$$
$$= \begin{bmatrix} 0.0313 & 0.0938 & 0.0938 & 0.0313 \\ 0.0938 & 0.2813 & 0.2813 & 0.0938 \\ 0.0938 & 0.2813 & 0.2813 & 0.0938 \\ 0.0313 & 0.0938 & 030938 & 0.0313 \end{bmatrix} \tag{3.1}$$

The 1-D convolution masks of the highpass filter g_1 for computing the detail coefficients $d_1(h)$ and $d_1(v)$ are

$$g_1^1[n] = \begin{bmatrix} -0.7071 \\ 0.7071 \end{bmatrix} \tag{3.2}$$

$$g_1^2[m] = \begin{bmatrix} -0.7071 & 0.7071 \end{bmatrix} \tag{3.3}$$

The one-level decomposition of an MR head image for the first scale is shown in Fig. 3.3. The original image is shown in Fig. 3.3 (a). Fig. 3.3 (b) shows the wavelet coefficients obtained by applying the 2-D convolution mask defined in Eq. (3.1) to the original image. Fig. 3.3 (c) shows the wavelet coefficients obtained by applying the 1-D convolution mask defined in Eq. (3.2) to the original image. Fig. 3.3 (d) shows the wavelet coefficients obtained by applying the 1-D convolution mask defined in Eq. (3.3) to the original image.

Note that the main difference between the UDWT decomposition shown in Fig. 3.3 and the DWT decomposition shown in Fig. 2.5 is that the UDWT does not use downsampling after each filtering process. Hence the UDWT is suitable for edge detection purpose, since the multiscale edge image obtained from these UDWT coefficients will be of the same size as the original image.

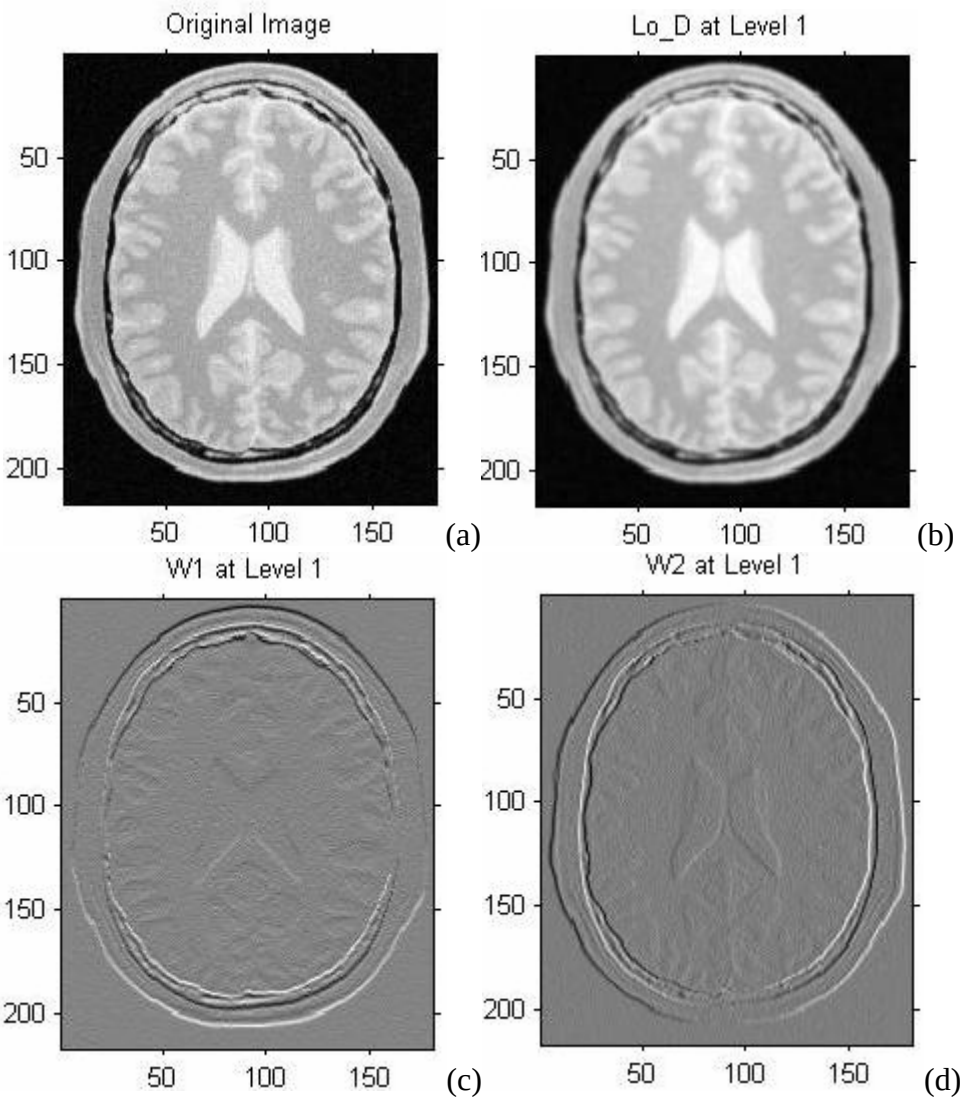


Figure 3.3: UDWT one-level decomposition demonstration.
a) Original MR image a_0; b) Lowpass filtered image a_1;
c) Highpass filtered image for wavelet transform in vertical direction $d_1(h)$;
d) Highpass filtered image for wavelet transform in horizontal direction $d_1(v)$.

3.2.2 Wavelet Modulus & Angle

In this thesis, we use four-level UDWT decomposition for multiscale edge detection. Fig. 3.4 shows the UDWT results of the original MR image shown in Fig. 3.3 (a). There are four images for wavelet components W_s^1 defined in Eq. (2.23), which are shown in Fig. 3.4 (a). Four images for wavelet components W_s^2 defined in Eq. (2.24), which are

shown in Fig. 3.4 (b). Four images for wavelet transform modulus M_s defined in Eq. (2.26) are shown in Fig. 3.4 (c). Four images for angle A_s defined in Eq. (2.27) are shown in Fig. 3.4 (d).

There are sixteen images shown in Fig. 3.4, four rows (labeled as 1, 2, 3, 4) and four columns (labeled as a, b, c, d). Each row of the images shows one level UDWT decomposition result that corresponds to each scale. Each column of the images shows one type of UDWT decomposition result that corresponds to one wavelet calculation.

For example, the first level UDWT decomposition gives wavelet components $W_{s_1}^1$ and $W_{s_1}^2$ for the first scale s_1 (where $s_1 = 2^0 = 1$), which shown in Fig. 3.4 (a.1) and (b.1). The wavelet modulus M_{s_1} shown in Fig. 3.4 (c.1) are calculated from $W_{s_1}^1$ and $W_{s_1}^2$ using Eq. (2.26). The angle A_{s_1} shown in Fig. 3.4 (d.1) are calculated from $W_{s_1}^1$ and $W_{s_1}^2$ using Eq. (2.27). The rest of wavelet components W_s^1 and W_s^2, wavelet modulus M_s, and angle A_s for scale s_2, s_3, s_4 are calculated as the same way as for s_1. (where $s_2 = 2^1 = 2$, $s_3 = 2^2 = 4$, and $s_4 = 2^3 = 8$).

Note that all the images shown in Fig. 3.4 as grayscale intensity image. The lowest value of the image displays as black. The highest value of the image displays as white. The value in between displays as intermediate shades of gray. For wavelet components W_s^1 and W_s^2, black, grey and white pixels correspond respectively to negative, zero and large coefficient values. For the wavelet modulus M_s, black and white pixels correspond respectively to zero and large modulus values. For angle A_s, black, grey and white pixels correspond respectively to negative, zero and large angle values (i.e., $[-\pi, \pi]$).

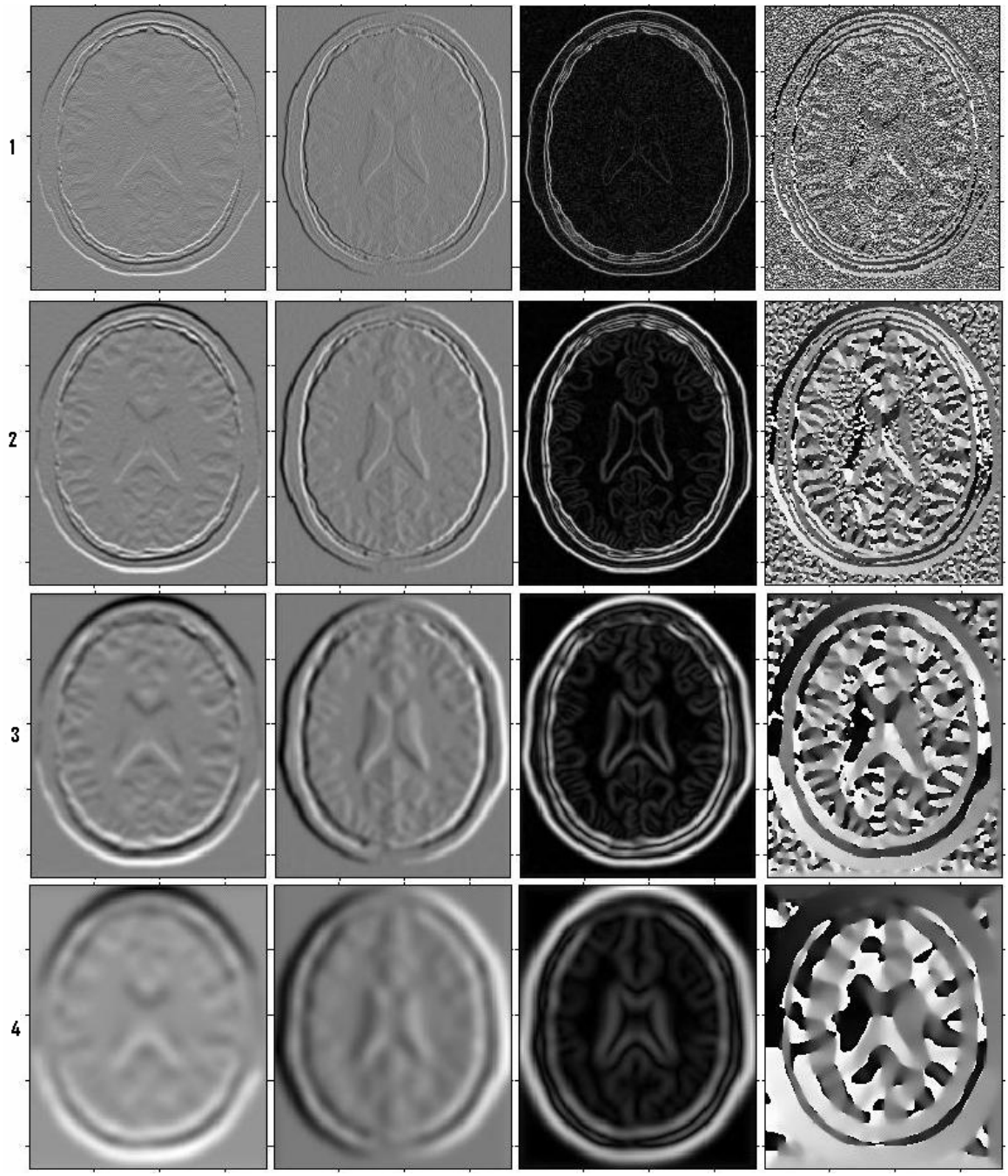


Figure 3.4: UDWT results of four-level decomposition.
a) Wavelet component W_s^1; b) Wavelet component W_s^2;
c) Wavelet modulus M_s; d) Wavelet angle A_s.

As we see from Fig. 3.4, the four-level wavelet decomposition allows us to view the image components at different resolutions. Lower scale corresponds to higher resolution.

Higher scale corresponds to lower resolution.

3.2.3 WTMM

Modulus maxima points typically represent the edge points in an image. We can determine the modulus maxima for each scale from the wavelet modulus and angle using the method defined in section 2.2.1. The WTMM points at each decomposition level are shown in Fig. 3.5 as white pixels in the binary image. The WTMM for scale s_1 shown in Fig.3.5 (a) is found by using M_{s_1} and A_{s_1} (shown in Fig. 3.4 (c.1) and (d.1)). The WTMM points at scale s_2, s_3, and s_4 are found in the same way as for scale s_1.

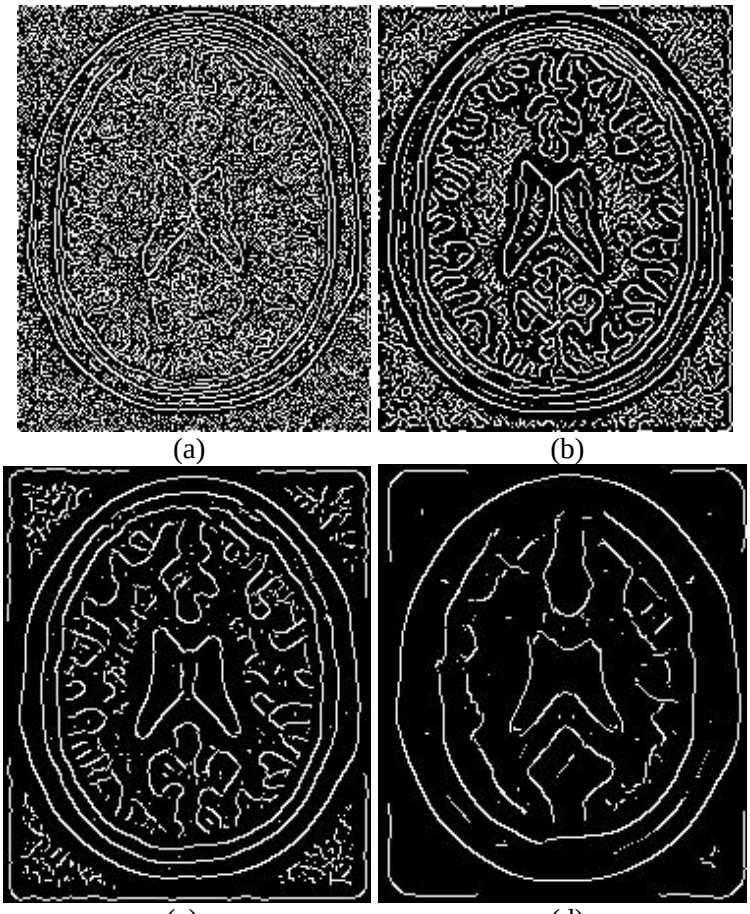


Figure 3.5: WTMM at four scales.
a) WTMM of scale s_1; c) WTMM of scale s_2;
c) WTMM of scale s_3; d) WTMM of scale s_4.

It is shown in Fig. 3.5 that when the scale increases, the details and the noise effect decrease quickly. As a result, the edge locations may change in higher scales. We use the WTMM of the first scale (i.e., the highest resolution) as the edge image. By looking at the WTMM at different scales, we find that edges of higher significance (i.e., stronger edges) are more likely to be kept by the wavelet transform across scales. Edges of lower significance (i.e., weaker edges) are more likely to disappear at higher scales.

3.2.4 Directional Multiscale Edge detection

It was presented in section 2.2.1 that WTMM can be found along the gradient angle. But practically, we found that only using one-dimensional neighborhood along the angle direction to decide an edge pixel is not sufficient. Because a pixel can be part of one edge in one angle direction and be part of another edge in another angle direction. Thus only checking one direction for each pixel to determine the WTMM will lose information if that pixel is also WTMM in other directions. Therefore, we check all the eight pixels around one pixel in the centre for four different directions to decide whether the centre pixel is a WTMM in each direction.

For example (see Fig. 2.7), at 0 degree we compare the modulus of the central pixel 5 with the left and right pixels (pixel 4 and 6). At 45 degree we compare it with the left-down and right-up pixels (pixel 3 and 7). At 90 degree we compare it with the up and down pixels (pixel 2 and 8). Finally, at 135 degree we compare it with the left-up and right-down pixels (pixel 1 and 9). In this way, we not only locate the WTMM of the pixel, but also we know in which direction it is a local maximum. This multi-directional way to find the modulus maxima is very helpful to keep all possible edge pixels in

different directions. Fig. 3.6 shows the directional edges detected for the first scale. The edges are shown as white pixels in the binary image.

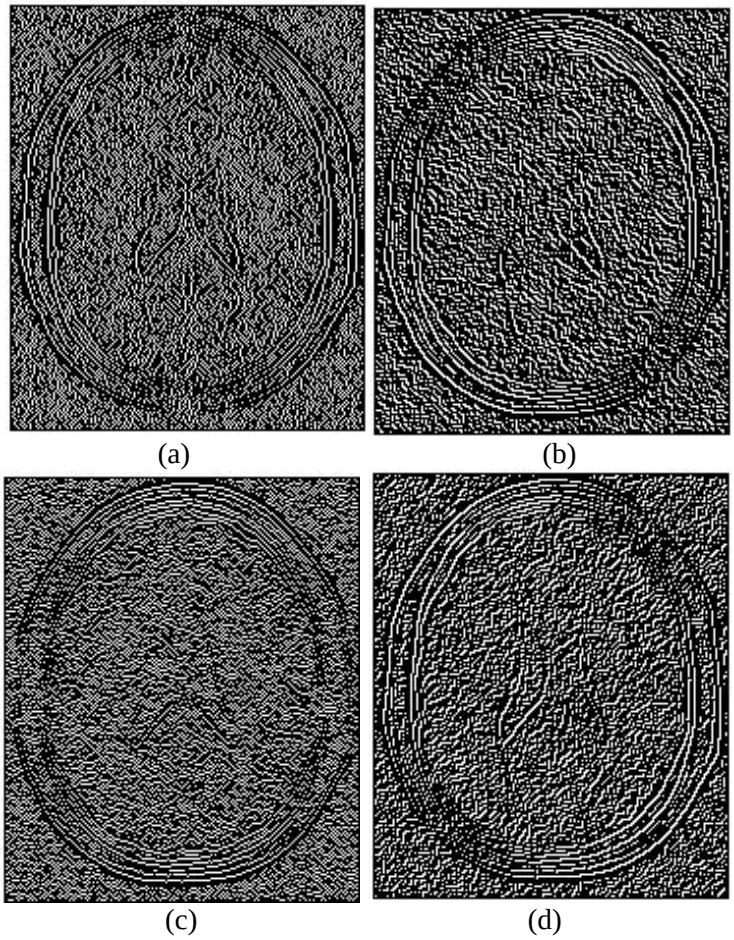


Figure 3.6: Directional edges for scale s_1.
a) 0° edges; b) 45° edges; c) 90° edges; d) 135° edges.

3.2.5 Subpixel Edge Resolution

In order to achieve more accurate edge pixel location, subpixel edge coordinates need to be obtained. As explained in section 3.2.4, we check each pixel in its local neighborhood to determine the WTMM in different directions. If it is the local maximum, then we record it as an edge pixel in that direction. Basically, we can interpolate the edge pixel that we found in the WTMM process, and fit the pixel coordinates in a polynomial curve to get subpixel resolution.

Consider a quadratic function

$$P(x) = ax^2 + bx + c \tag{3.4}$$

The peak of this quadratic function can be computed at

$$x_{peak} = -\frac{b}{2a} \tag{3.5}$$

The coefficients of this quadratic function can be found by polynomial curve fitting in the least squares sense [2]. An example of edge pixel interpolation is shown in Fig. 3.7. In 0 degree, there are three pixels whose X coordinates are the same, but Y coordinates are 24, 25, 26. We found that the middle one is the modulus maxima among those three pixels in 0 degree direction, and hence we record 25 as an edge pixel Y coordinate. Now we fit the three pixels in a polynomial curve and the peak value from the quadratic function come out as 24.8791. This subpixel coordinate is more precise than 25.

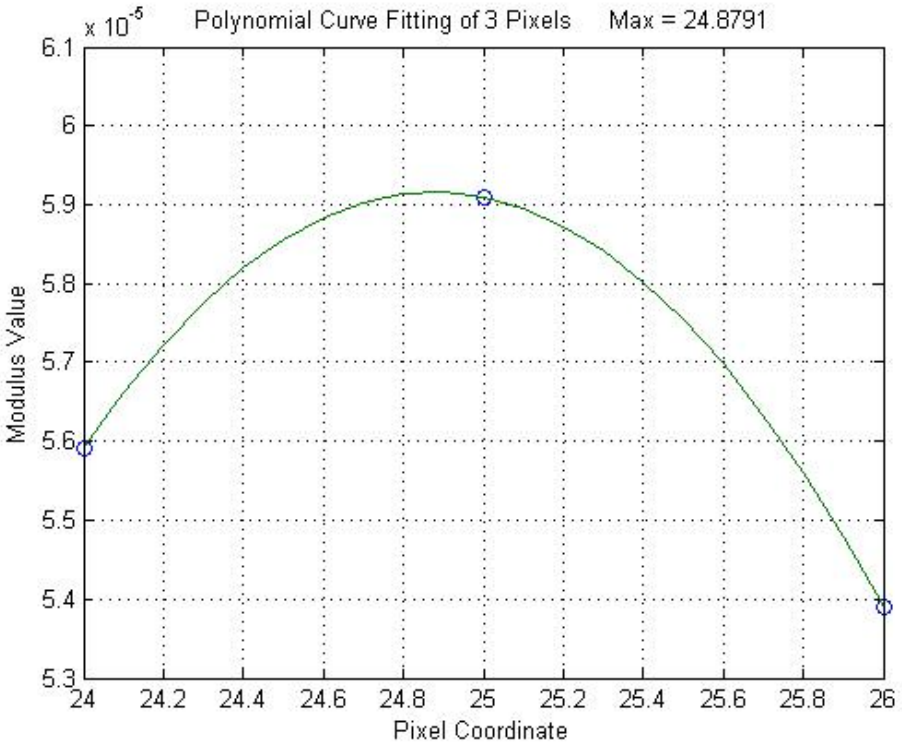


Figure 3.7: Edge pixel interpolation.

In other directions, such as 45 or 135 degree, we can simply rotate the pixel coordinates to 0 degree and fit in polynomial to get the peak coordinates, and then rotate back to the original angle to get the coordinates in that direction.

3.3 Multiscale Boundary Identification

We have described how to obtain the edges from wavelet modulus maxima in section 3.2. In this section, we will present a method to analyze the multiscale edge information obtained through the WTMM chain process. This will be very helpful for identifying the boundary.

3.3.1 2-D WTMM Chain

As we have seen in Fig. 3.5, the WTMM at different scales represent different edges through wavelet decay. After we obtain the WTMM for each scale, we need to chain those maxima across scales to indicate where the true edge is. If it is the true edge, it should not disappear during the wavelet decay, thus we start the WTMM chaining process from the highest scale down to the lowest scale. We need to link the maxima points between two adjacent scales and record the WTMM chain.

We do a four-level decomposition and have four scales with WTMM found for each scale. We start at the highest scale (i.e., scale s_4), and for each maxima pixel, we search in the previous scale (i.e., scale s_3) within a small region (a square window).

For example (see Fig. 3.8) for the chaining process diagram. Between scale *j* and scale $j-1$, suppose we want to chain a modulus maxima pixel *z* in scale *j*, whose X-Y coordinates are (z_r, z_c) in the plane. The centre of this search window matches the maxima pixel location in the higher scale. Thus the centre pixel coordinate of this search

window in scale $j-1$ is also (z_r, z_c). Then a qualification test is performed to check if there is one maxima pixel among all these pixels in the window that has strongest connection to pixel *z*. If the strongest connection is between these two maxima pixels, we link these two maxima pixels together. The chaining process is repeated up to the lowest scale, so that the maxima line can be found across the scales and all of the maxima are chained together.

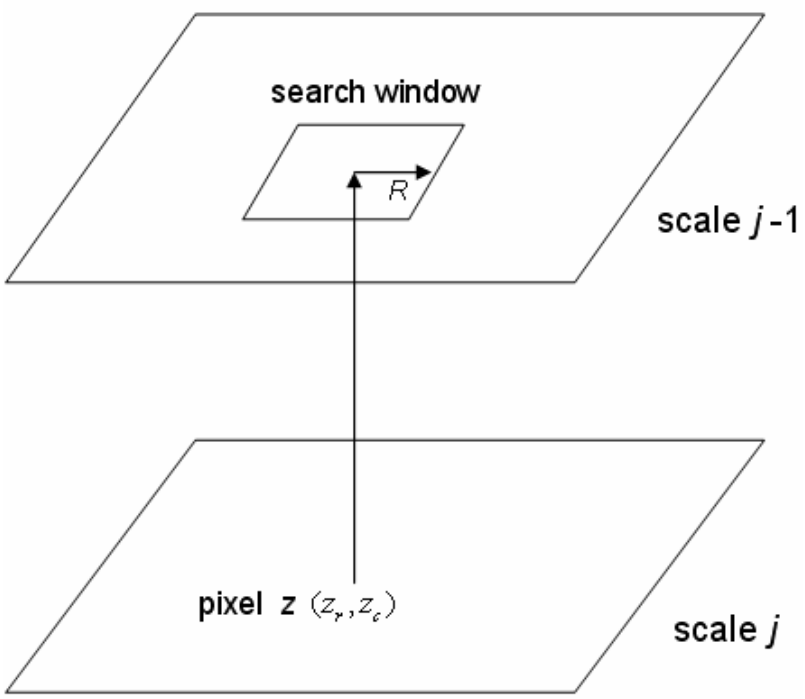


Figure 3.8: Image WTMM chain process across scales.

We need to define a qualification test to determine one of the maxima in the search window in scale $j-1$ that has the strongest connection to the maxima in scale *j*. Since the WTMM are calculated from wavelet modulus/angle and also involve with pixel location in the searching window, the qualification test considers a distance measurement that involves calculation of the modulus distance, angle distance and location distance of the maxima pixels between two adjacent scales. All three distances are normalized to

have a dynamic range of $[0,1]$. A weighted average of the three normalized distance is the final qualification result.

The modulus distance m_1 of all the modulus maxima pixels in the searching window is calculated by

$$m_1 = 1 - \frac{M_x}{M_{\max}} \tag{3.6}$$

where M_x is the wavelet modulus of one of the WTMM pixel, *x*, whose X-Y coordinates are (x_r, x_c) in the search window. $M_{\max}$ is the maximum value of all WTMM in scale $j-1$.

The angle distance m_2 of all the modulus maxima pixels in the searching window is calculated by

$$m_2 = \frac{\min(2\pi - d, d)}{\pi} \tag{3.7}$$

$$d = \left| A_x - A_z \right| \tag{3.8}$$

where A_x is the wavelet angle of pixel *x* in the searching window and A_z is the wavelet angle of pixel *z* in scale *j*. Note that the wavelet angle is between $-\pi$ and π.

The location distance m_3 of all the modulus maxima pixels in the searching window is calculated by

$$m_3 = \frac{\sqrt{(x_r - z_r)^2 + (x_c - z_c)^2}}{L_{\max}} \tag{3.9}$$

$$L_{\max} = \sqrt{R^2 + R^2} = R\sqrt{2} \tag{3.10}$$

where R is the half size of the square search window, and $L_{\max}$ is the maximum distance value between the centre pixel and any of the pixels in the searching window.

The qualification test is the combination of the three distances test defined in Eq. (3.6), (3.7), and (3.9) by

$$m = \frac{a_1 m_1 + a_2 m_2 + a_3 m_3}{a_1 + a_2 + a_3} \tag{3.11}$$

where a_1, a_2, and a_3 are weighting factors. In this thesis, the values for those weighting factors are 1, 5, and 1, respectively.

The pixel that has the maximum m value in a search window has the strongest connection to pixel z. Therefore, these two pixels should link together. For four scales, we need to do three chaining processes between the adjacent scales. The chaining process goes from the highest scale $j = 4$ to the lowest scale $j = 1$. As we see from the WTMM of all scales in Fig. 3.5, the edge location is shifted more as the scale increases. Thus, the search window size ($(2R+1)^2$) can be decreased at each step of the chaining process. Here, we choose $R = 3, 2, 1$ for scale $j = 3, 2, 1$ so that a larger window size is used for higher scales and a smaller window size is used for lower scales.

We want to link one maxima point in a higher scale to one maxima point from the search window in a lower scale. Since the search windows overlap, it is possible to have one pixel with maximum m value for more than one maxima point at a higher scale. For example, we may find two pixels in scale j that could connect to the same maxima pixel in scale $j-1$. However, we would like to have a one-to-one match to the lower scale maxima pixel for this chaining process. In order to prevent this competing situation, auction algorithm [21] is used to optimally perform one-to-one maxima pixel matching.

The benefit (cost) for the auction algorithm (reviewed in section 2.4) is defined by the qualification test in Eq. (3.11).

In order to use the 2-D WTMM chain for further analysis, a chaining record (defined in Eq. (3.12)) is kept for all the WTMM pixels in the lowest scale. For the WTMM pixel chains that pass through all four scales, we assign 4 for that pixel in the WTMM chaining record. For the WTMM pixel presented only in the lowest scale with no connection to the next scale, we assign 1 for that pixel. Fig. 3.9 shows the WTMM chaining record diagram. The small square in the scales indicates WTMM. The arrows indicate WTMM linking between two adjacent scales.

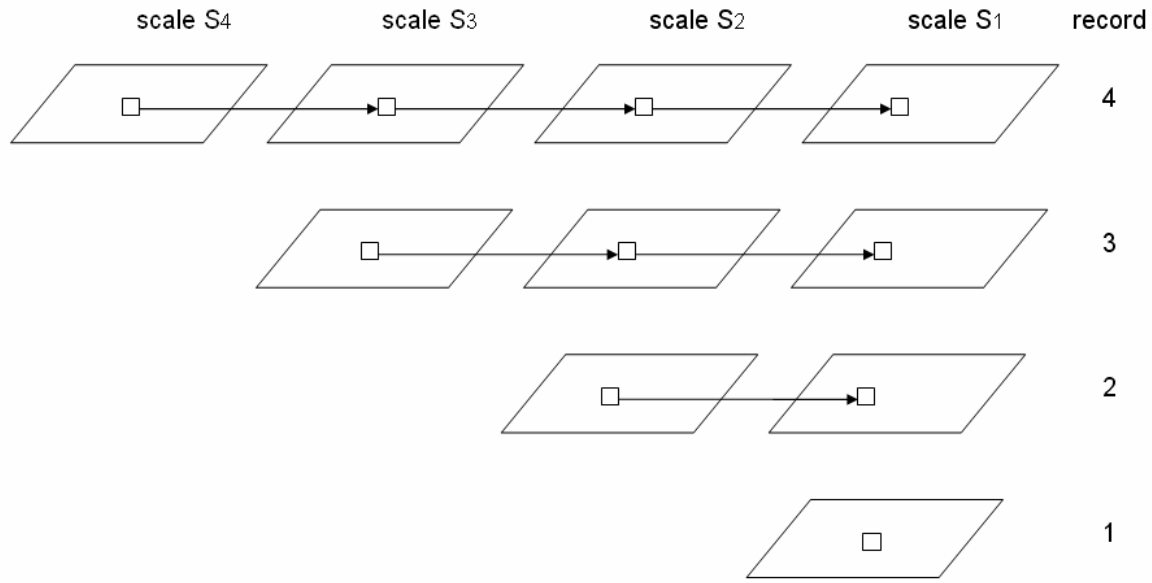


Figure 3.9: Illustration of WTMM chaining record.

The WTMM chaining record for four scales is defined as

$$WTMM_{record} = \begin{cases} 4 & if \ \ chaining \ \ through \ \ scale \ \ s_4 \ \ to \ \ s_1 \\ 3 & if \ \ chaining \ \ through \ \ scale \ \ s_3 \ \ to \ \ s_1 \\ 2 & if \ \ chaining \ \ through \ \ scale \ \ s_2 \ \ to \ \ s_1 \\ 1 & the \ \ rest \ \ of \ \ WTMM \ \ at \ \ scale \ \ s_1 \end{cases} \tag{3.12}$$

In the chaining record, the higher score corresponds to a stronger edge pixel since it survives along the wavelet decay, whereas the lower score corresponds to a weaker edge pixel. We define our 2-D WTMM chaining record by ‘Scale Depth’, as it demonstrates how deep the WTMM pixel is connected through scales.

Fig. 3.10 shows the scale depth map illustrated in a color image with a normalized colorbar at the right side of the image. The chaining score 1, 2, 3, and 4 are normalized to be 0.25, 0.5, 0.75, and 1, respectively. It is the scale depth map of the MR head image shown in Fig. 3.3 (a). The first four colors represent four scores for the WTMM chain through scales. Obviously, the higher color scores in the colorbar are the stronger edges in the image, and the lower color scores are the weaker edges in the image.

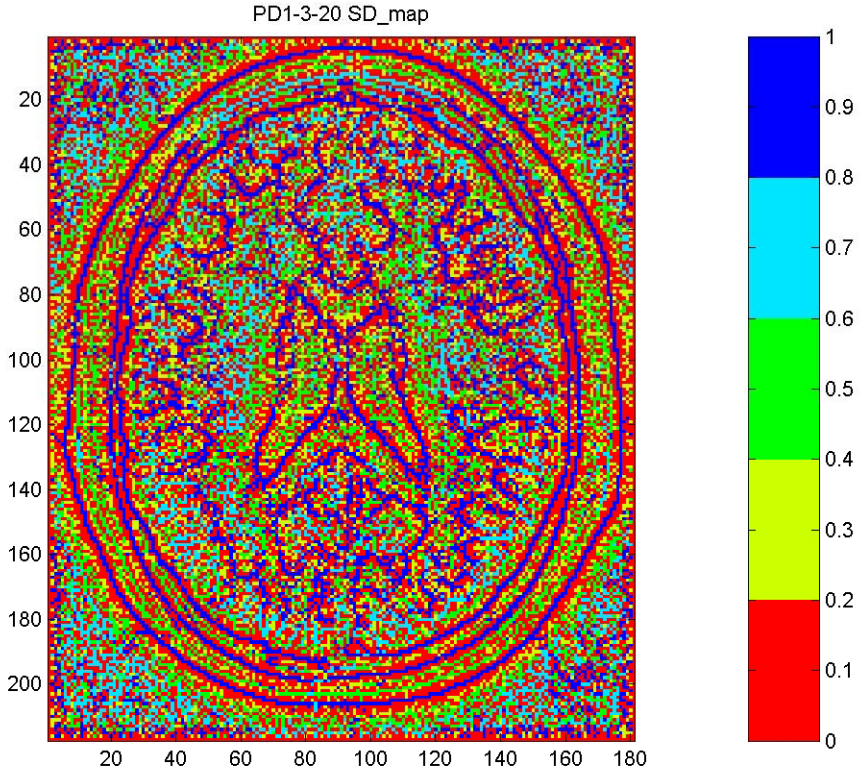


Figure 3.10: Scale depth map.

3.3.2 Lipschitz Regularity Analysis

It has been shown that the local Lipschitz regularity of a signal $f(x)$ at point v depends on the decay of wavelet modulus $|W_s(x)|$ in the neighborhood of v [8]. The Lipschitz exponent at that point characterizes the singular behavior. This allows us to distinguish step and spike edges. Thus we can use WTMM from wavelet decay to do Lipschitz analysis.

As we discussed in section 2.1.5, the singularities (edges) can be detected from the WTMM at different scales, since the evolution of wavelet local maxima across scales characterizes the local shape of the edge structures.

Continuing from section 2.1.5, we have

$$|W_s(x)| \le As^{\alpha} \tag{3.13}$$

We can take logarithms of Eq. (3.13),

$$\log_2 |W_s(x)| \le \log_2 A + \alpha \log_2 s \tag{3.14}$$

It appears from Eq. (3.14), the Lipschitz exponent α is the maximum slope of $\log_2 |W_s(x)|$ as a function of $\log_2 s$ along the maxima lines. It is clear that finding the slope of the linear relationship of Eq. (3.14) can produce an estimate of Lipschitz exponent.

It is difficult to estimate A in Eq. (3.13) accurately. In addition, the value of Lipschitz exponent α varies from scale to scale, thus we can find the linear relationship between the adjacent scales to estimate α. For example, Eq. (3.14) can be rewritten for scale s_1 and s_2 as follows.

$$\log_2 |W_{s_1}(x)| \le \log_2 A + \alpha \log_2 s_1 \tag{3.15}$$

$$\log_2 \left|W_{s_2}(x)\right| \le \log_2 A + \alpha \log_2 s_2 \tag{3.16}$$

Then, the Lipschitz exponent corresponding to scale s_1 and s_2 must satisfy the following.

$$\frac{\log_2 \left|W_{s_2}(x)\right| - \log_2 \left|W_{s_1}(x)\right|}{\log_2 s_2 - \log_2 s_1} \le \alpha \tag{3.17}$$

As we see from Eq. (3.17), the Lipschitz exponent is controlled by the log of wavelet modulus across scales. We want to use this Lipschitz information for our multiscale boundary identification, as it can allow us to distinguish step edge and spike noise. Thus, instead of estimating the Lipschitz exponent, we will examine the log of wavelet modulus across scales.

Note that for 2-D image, the wavelet modulus $\left|W_s(x)\right|$ in Eq. (3.13) should change to be $M_s(x)$ as we defined in Eq. (2.26). The $\log_2 M_s(x)$ values for one real edge point in the MR image at each scale are shown in Fig. 3.11 (a), and for one background noise point is shown in Fig 3.11 (b).

We use four-level wavelet decomposition to obtain multiscale edges. Fig. 3.11 plots the $\log_2 M_s(x)$ values for four scales, where $s_1 \le s \le s_4$. The X-axis of Fig. 3.11 is scale *s*, and Y-axis is $\log_2 M_s(x)$ value. The dots in Fig. 3.11 indicate the $\log_2 M_s(x)$ values corresponding to the scales. Apparently, as the scale increasing from s_1 to s_4, the $\log_2 M_s(x)$ of an edge point is increasing, but for a noise point it is decreasing. The $\log_2 M_s(x)$ through scales of an edge point has positive slope, and of a noise point it has negative slope. Also note that, the four $\log_2 M_s(x)$ values are quite different between the

edge and noise. The $\log_2 M_s(x)$ values for an edge point are between 6 and 10, and the values for noise one are between 0.5 and 2.5.

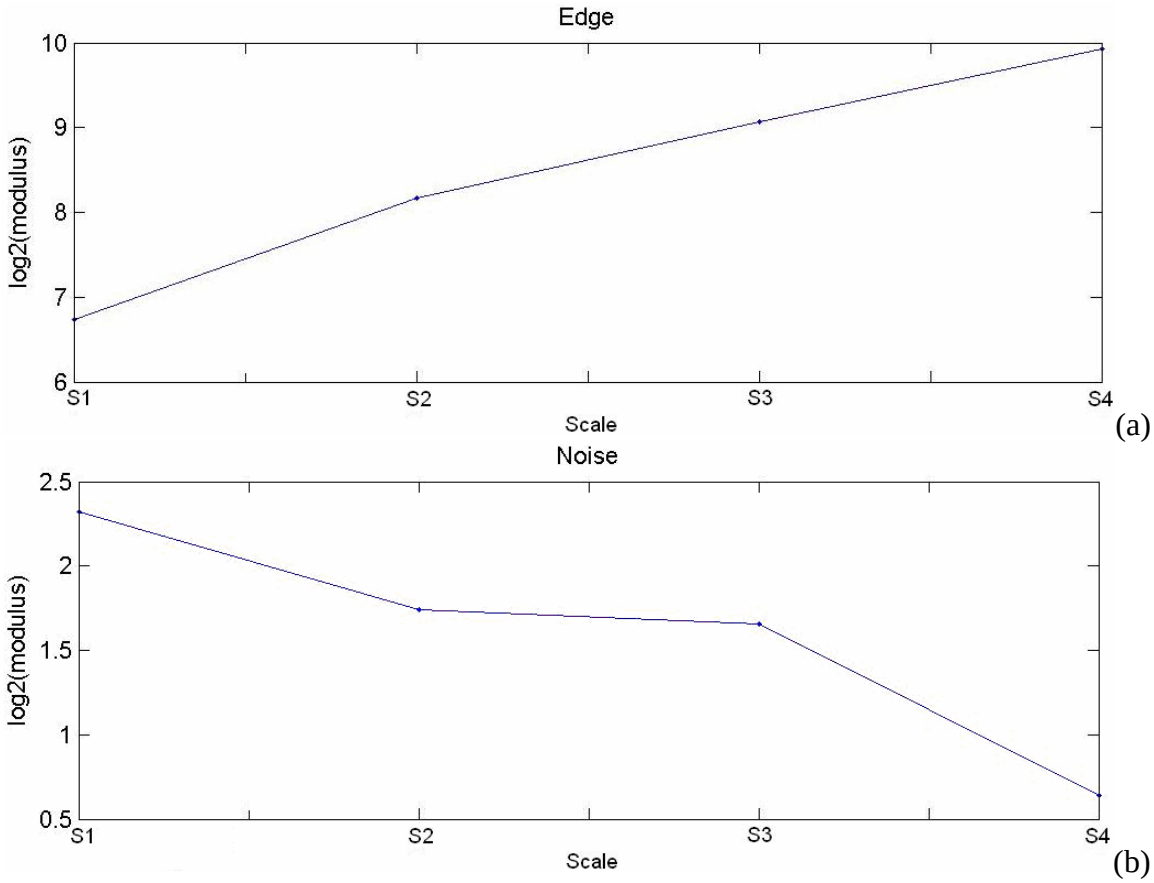


Figure 3.11: $\log_2 M_s(x)$ values for two points in the MR image of four scales. a) $\log_2 M_s(x)$ values of a edge point; b) $\log_2 M_s(x)$ values of a noise point.

We have discussed that the Lipschitz exponent characterizes different edge structures, and we have seen from Fig. 3.11 that the true edge and noise can be separated in two different ways: $\log_2 M_s(x)$ value (amplitude) and $\log_2 M_s(x)$ difference (slope). Larger amplitude indicates stronger edge or noise. Also the more positive slope indicates truer edge, and the more negative slope indicates truer noise. We also know that weaker edge or noise, of course, will have less $\log_2 M_s(x)$ value and less $\log_2 M_s(x)$ difference.

3.3.3 Edge Quality Analysis

We want to utilize the two pieces of information observed (demonstrated in Fig. 3.11) to better separating true edge from noise. Therefore, an edge quality test is formed to combine the amplitude and slope information of the WTMM chain through scales to give the strength of each point.

Edge quality analysis is implemented using fuzzy logic (described in section 2.5). One WTMM point (shown in Fig. 3.10) can have up to four $\log_2 M_s(x)$ values, since we have four scales. Let us denote *A* as the values of the $\log_2 M_s(x)$ amplitude, and *B* as the values of the $\log_2 M_s(x)$ difference. The amplitude information *A* and the slope information *B* are the input of the fuzzy logic system.

The output of the fuzzy logic system *EQ* is the edge quality of each WTMM point. The fuzzy logic system maps the input *A* and *B*, and after the fuzzy logic operations, we have the output *EQ* defined as

$$EQ = \max\left(\min(A'), \min(B')\right) \tag{3.18}$$

where A' is the mapping result using linear membership function shown in Fig. 3.12 (a), B' is the mapping result using sigmoid membership function shown in Fig. 3.12 (b).

The equation of the sigmoid function for mapping *B* is given by

$$B' = \frac{1}{1 + e^{-\lambda \cdot slope}} \tag{3.19}$$

where $\lambda = 3$ and the slope is calculated as

$$slope = diff\left(\log_2 M_s(x)\right) \tag{3.20}$$

where *diff* is the difference operator.

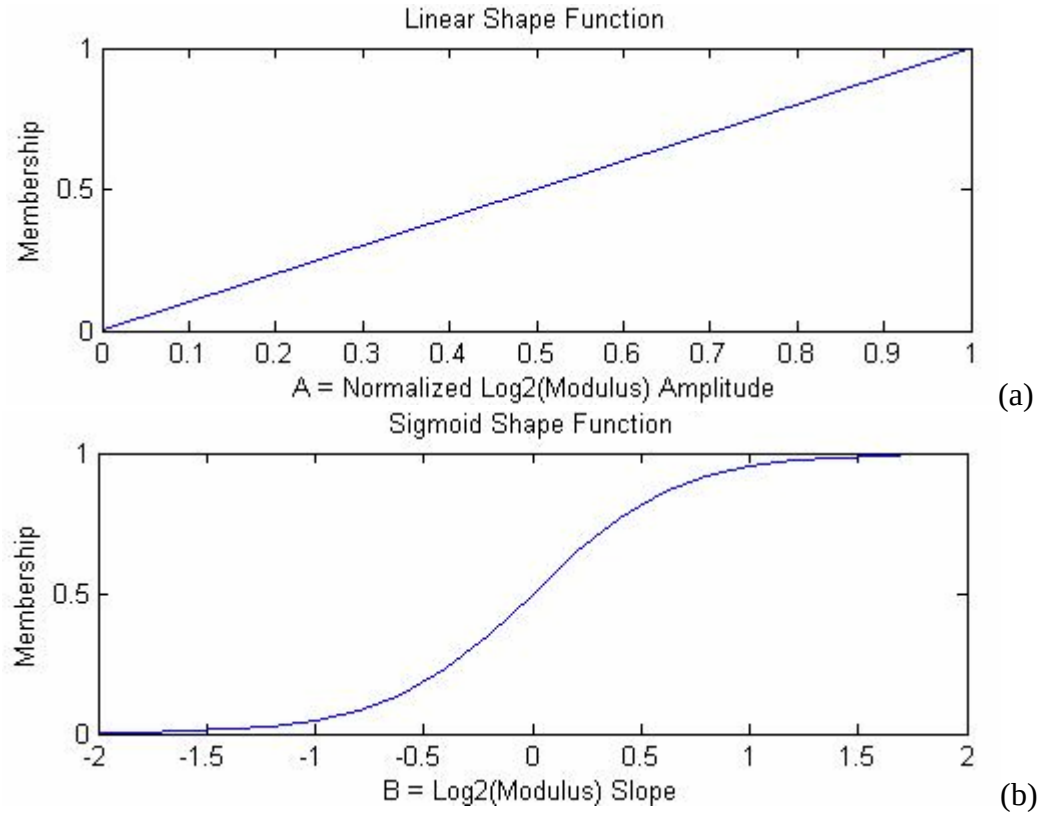


Figure 3.12: Membership functions for edge quality test.
a) Linear membership function; b) Sigmoid membership function.

The block diagram of the fuzzy logic system implemented for the edge quality analysis is shown in Fig. 3.13.

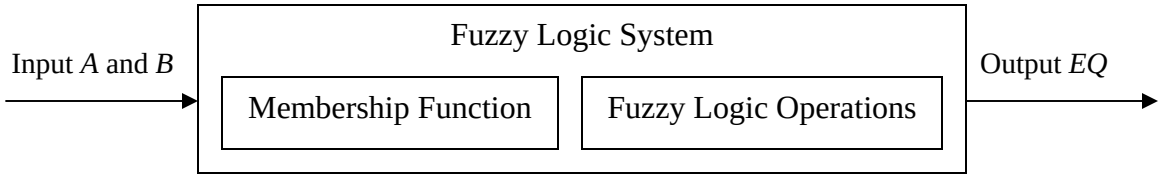


Figure 3.13: Fuzzy logic system for edge quality test.

Note that our fuzzy logic system (shown in Fig. 3.13) does not have the third part compared to Fig. 2.10, because we need to have the fuzzy logic output value in [0, 1] for our edge quality test. We called the result after edge quality test 'Edge Quality'.

Fig. 3.14 shows the edge quality map illustrated in color image with a normalized colorbar at the right side of the image. Obviously, the higher scores in the colorbar

indicate the stronger edges in the image, the lower scores indicate the weaker edges in the image.

Apparently, comparing this edge quality map and the scale depth map shown in Fig. 3.10, small edges of the gray/white matter boundary shown in the scale depth map that has lower scale depth value, now shown in the edge quality map has higher quality value, which means adding the edge quality analysis gives more valuable information about the small edges.

Through the edge quality test, we can tell the true edge apart from the noise, even it's a very small weak edge. Thus, it gives a measurement for the quality of the edge, which helps to better identify the edges from noise.

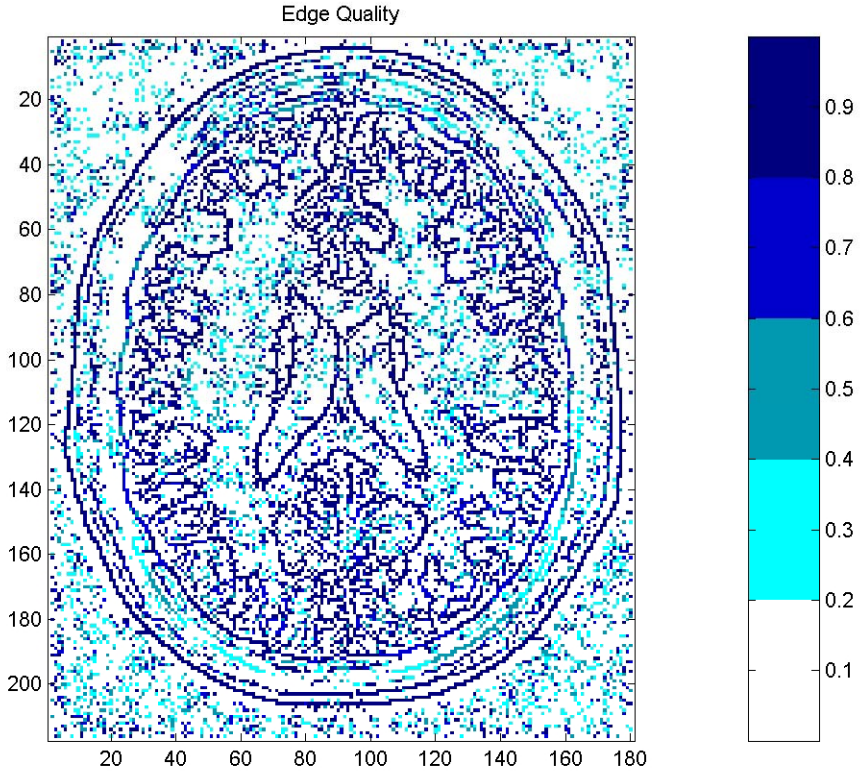


Figure 3.14: Edge quality map.

3.4 mtrack Integration

The multiscale edge detection and boundary identification method developed have been integrated into the mtrack software. In this section, we present how this integration is performed.

First of all (see Fig. 3.1), the directional multiscale edge detection replaces the Canny edge detection in mtrack. The multiscale subpixel edge resolutions are also included. The edge feature extraction works in the same functions as the old one to extract the edge features.

There is an objective function (Eq. 3.21) in the data association process of the edge tracing algorithm. The objective function is used to evaluate the candidate data points for further tracing processing.

$$\varphi_i = A_i \cdot B_i \cdot C_i \tag{3.21}$$

This objective function is a product of elements, and all the elements are normalized between 0 and 1. Thus, our edge quality value after the fuzzy logic system can be easily included in the product as one additional factor *EQ*.

$$\varphi_i = A_i \cdot B_i \cdot C_i \cdot EQ_i \tag{3.22}$$

Hence, multiscale edge quality information can be used in mtrack.

3.5 Summary

In this chapter, the proposed framework for multiscale boundary identification has been presented. We explained how to implement the multiscale edge detection, and how to perform the boundary identification from the multiscale edges. This framework mainly contains four stages: multiscale edge detection, WTMM chain, Lipschitz regularity

analysis, and edge quality analysis. Directional multiscale edge detection with subpixel edge resolution provides more precise edge points. WTMM chain connection through wavelet decay gives scale depth measurement. Lipschitz regularity analysis identifies true edge from noise. Edge quality analysis performs a quality measurement for detected edges. In addition, the proposed work is integrated into the medical image segmentation software to achieve better edge tracing result.

Chapter 4

Performance Evaluation

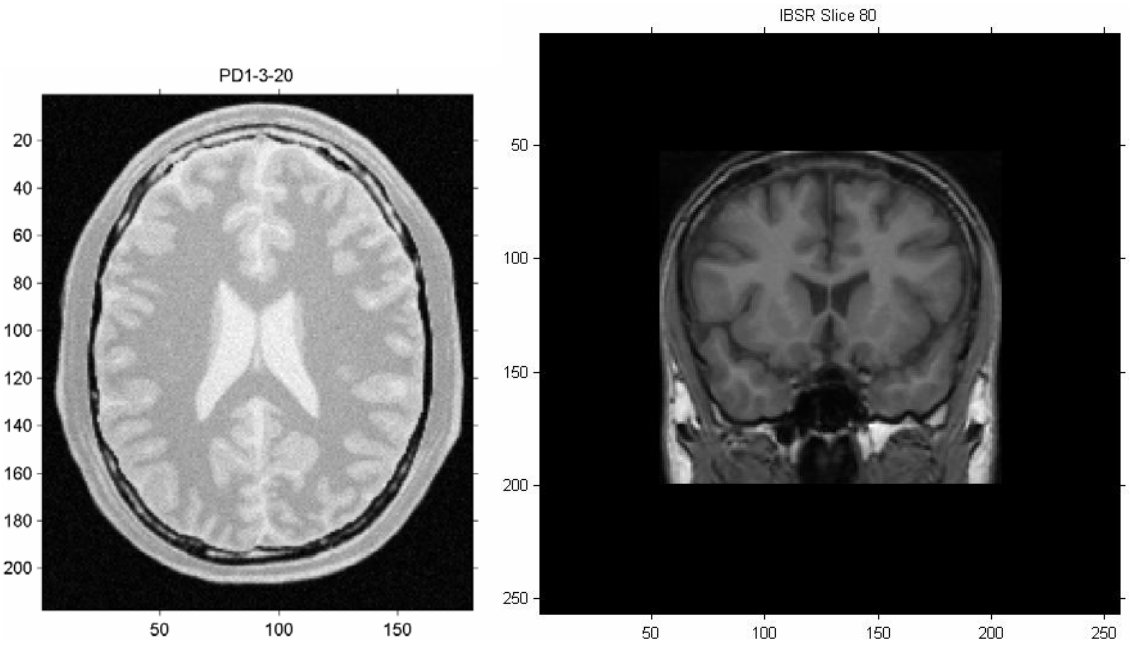
We proposed a technique for multiscale edge detection and boundary identification in chapter 3. In this chapter, we evaluate the performance of the proposed technique. In order to do this, the proposed technique has been integrated with the mtrack image segmentation software. The comparison of MR image edge tracing results are presented to show that the multiscale boundary identification method improves the edge tracing performance.

The organization of the chapter is as follows. Section 4.1 presents the synthetic and real MR data used for evaluation. Section 4.2 presents the reference boundary obtained from the MR data. Section 4.3 presents the evaluation criteria for testing. Section 4.4 presents performance of synthetic MR data. Section 4.5 presents performance of real MR data. Summary is given in section 4.6.

4.1 MR Test Images

There are two types of MR images used for performance evaluation. One is synthetic MR images obtained from Montreal Neurological Institute (MNI) [26]. The MR simulator can generate the ‘true’ images with added noise and non-uniformity. Basically, the MR image has nine different tissue classes of the brain. Each tissue class has its own intensity level. The synthetic images are formed by passing the manually segmented data through an MR simulator. The MR simulator adds a specific level of noise and non-

uniformity onto the true image data, and hence we can test various noise and non-uniformity levels. These MNI data has 8 bits per pixel resolution. The data is obtained as 1mm slice thickness. It is 217×181 pixels per slice. The slices are oriented in the transverse plane. There are total 181 slices. We use synthetic data with 1%, 3%, 5%, 7% and 9% noise, and 0%, 20% non-uniformity. In this thesis, we have used slice 95 of the MNI data to evaluate the performance. This slice is shown in Fig. 4.1 (a) as a proton density image, and it has 3% of noise and 20% of non-uniformity in it.



(a) (b)

Figure 4.1: MNI and IBSR data.
a) Slice 95 of MNI data with 3% of noise and 20% of non-uniformity;
b) Slice 80 of IBSR data.

The second image data used in the performance evaluation is real MR images adopted from Internet Brain Segmentation Repository (IBSR) [27]. This real MR data has low noise and has been pre-processed for non-uniformity correction. The IBSR data is obtained as 1.5 mm slice thickness. It is 256×256 pixels per slice. There are total of 128 slices oriented in the coronal plane. In this thesis, we have used slice 80 of the IBSR data

to evaluate the performance, as it is more difficult to obtain boundaries in this slice than others. This slice is shown in Fig. 4.1 (b).

4.2 Reference Boundary

In order to do the performance evaluation, we first need to obtain the true boundary for both synthetic and real MR image data. In the true boundary image, the exact location of the boundary is known on the image. Thus, after edge tracing, we can compare the tracked boundary with the true boundary to see how many tracking points are on the true boundary for evaluation. The performance evaluation testing is focused on the gray/white matter boundary. Since the gray/white mater boundary is usually a low contrast boundary, it is difficult to determine this boundary with high accuracy.

We need to obtain the gray/white matter reference boundary for our evaluation. For the synthetic MNI data slice 95, there is a gray matter image (shown in Fig. 4.2 (a)) and a white matter image (shown in Fig. 4.2 (b)) in the database. We can combine the gray matter image and the white matter image to obtain a gray/white matter boundary to be considered as a reference boundary.

Let us represent the pixel values of the gray matter and white matter image as $g(m,n)$ and $w(m,n)$. A boundary image *b* can be obtained by

$$b(m,n) = g(m,n) \,\&\, w(m,n) \tag{4.1}$$

where '&' represents logic AND operator. Fig. 4.2 (c) shows the boundary image *b* obtained. It is an binary image, where the boundary pixels are 1, shown as white pixels in Fig. 4.2 (c). Note that the center part of MR image shown in Fig 4.1 (a) is cerebrospinal fluid (CSF). Thus, the boundary image *b* shown in Fig. 4.2 (c) includes a CSF/gray/white

matter boundary. Therefore, we manually delete the CSF/gray/white matter boundary in the center of Fig. 4.2 (c) to obtain the gray/white boundary.

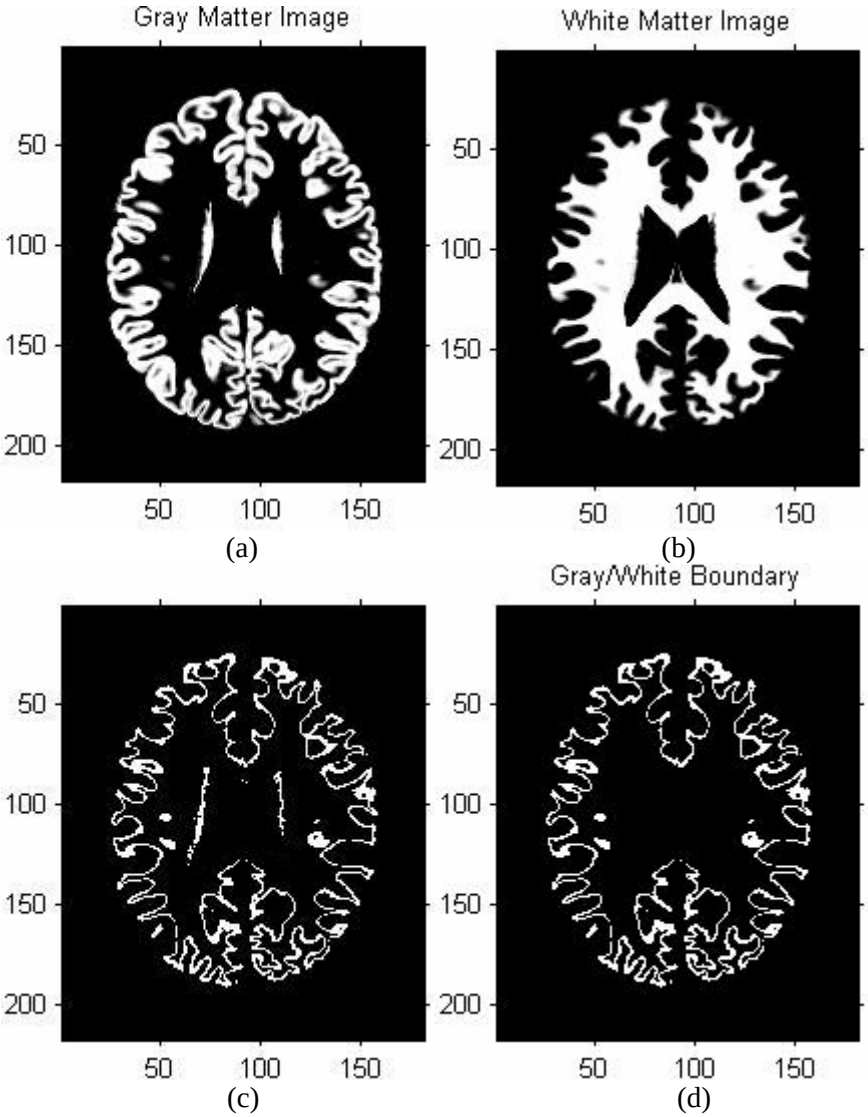


Figure 4.2: Illustration of the true gray/white boundary of MNI data obtained.
a) Gray matter image of slice 95; b) White matter image of slice 95;
c) Boundary image obtained from Fig. 4.2(a) and (b);
d) Gray/white boundary of slice 95.

Unlike the synthetic data, we do not know exactly where the gray/white boundary is in the real IBSR data. However, the database includes a manually-guided gray/white matter boundary for these real images. Thus, we consider the manual contour as the

reference boundary for testing. Fig. 4.3 shows the manual contour of gray/white matter boundary for IBSR slice 80. It is one-pixel wide contour.

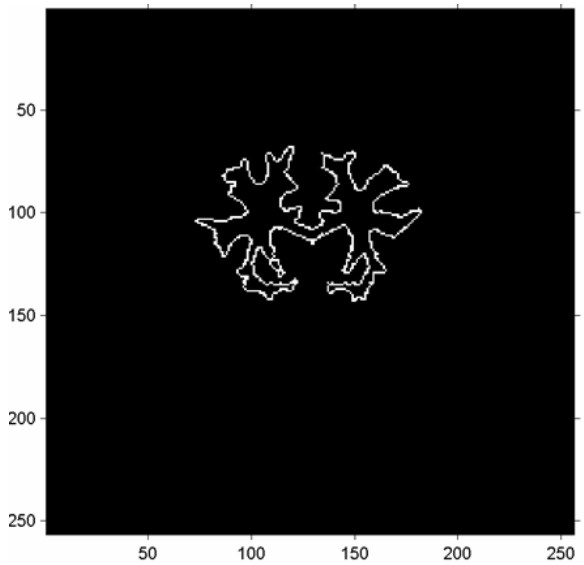


Figure 4.3: True gray/white boundary of IBSR data slice 80.
(white pixels represent boundary points)

4.3 Evaluation Criteria

We need to define the quality of the boundary image obtained for our performance evaluation. After running mtrack software, we obtain an output boundary image. The output boundary image is the result of the edge tracing algorithm. It represents tracked boundary as a binary image. Let the output tracked boundary image be denoted by *t*. The value of $t(m,n)=1$ if the pixel (m,n) is on the boundary, otherwise $t(m,n)=0$.

The total number of track points (i.e. the pixels on the output boundary) N_{TP} is obtained by the following equation.

$$N_{TP} = \sum_{(m,n)} t(m,n) \tag{4.2}$$

Note that not all the track points on the output boundary image *t* are the true boundary points. Thus, the good track point of the output boundary is obtained by

comparing it with the true boundary image *b*. The total number of good points N_{GP} on the output boundary image is obtained as follows.

$$N_{GP} = \sum_{(m,n)} t(m,n) \cdot b(m,n) \tag{4.3}$$

The track point ratio *R* is defined as

$$R = \frac{N_{GP}}{N_{TP}} \tag{4.4}$$

The parameter *R* represents the ratio of N_{GP} and N_{TP}. It provides useful information about the accuracy of the tracked boundary. Ideally, track point ratio *R* should be 1. However, because the edge tracing algorithm can trace on false boundary points, *R* is normally less than 1. The higher track point ratio corresponds to the better tracing result. Therefore, we use the track point ratio *R* to represent the performance of the output boundary image obtained.

As we mention in section 2.6, the edge tracing performance depends on the starting point. To acquire a statistical edge tracing result, for each input image, we set 30 starting points, approximately evenly spaced on the gray/white boundary of that image. Thus, we will have 30 tracks (output boundaries) for one input image. The mean, median, and standard deviation of all 30 tracks are calculated as follows.

The average of N_{GP}, N_{TP}, and *R* for 30 tracks are calculated as

$$mean_{GP} = \frac{1}{30} \sum_{i=1}^{30} N_{GP}(i) \tag{4.5}$$

$$mean_{TP} = \frac{1}{30} \sum_{i=1}^{30} N_{TP}(i) \tag{4.6}$$

$$mean_{R} = \frac{1}{30} \sum_{i=1}^{30} R(i) \tag{4.7}$$

where $N_{GP}(i)$, $N_{TP}(i)$, and $R(i)$ are the N_{GP}, N_{TP}, and R for track i.

The median value of N_{GP}, N_{TP}, and R for 30 tracks are calculated as

$$med_{GP} = median\left\{ N_{GP}(i),\ \ 1 \le i \le 30 \right\} \tag{4.8}$$

$$med_{TP} = median\left\{ N_{TP}(i),\ \ 1 \le i \le 30 \right\} \tag{4.9}$$

$$med_{R} = median\left\{ N_{R}(i),\ \ 1 \le i \le 30 \right\} \tag{4.10}$$

The standard deviation of N_{GP}, N_{TP}, and R for 30 tracks are calculated as

$$std_{GP} = \sqrt{\frac{1}{30}\sum_{i=1}^{30}\left(N_{GP}(i) - mean_{GP}\right)^2} \tag{4.11}$$

$$std_{TP} = \sqrt{\frac{1}{30}\sum_{i=1}^{30}\left(N_{TP}(i) - mean_{TP}\right)^2} \tag{4.12}$$

$$std_{R} = \sqrt{\frac{1}{30}\sum_{i=1}^{30}\left(R(i) - mean_{R}\right)^2} \tag{4.13}$$

For each input image, four different testing methods for the edge tracing algorithm are used for the evaluation comparison. We use the same edge image and the same starting points for each method. The mean, median, and standard deviation of 30 tracks are recorded for each method. Fig. 4.4 shows the block diagram of the four methods.

Method 1 shown in Fig. 4.4 (a) uses edge tracing with no constraint, and hence the edge tracing algorithm will trace based on the edge image obtained from multiscale edge detection. Method 2 shown in Fig. 4.4 (b) includes multiscale boundary identification prior to edge tracing, and hence the edge tracing algorithm will trace based on edge quality information obtained from multiscale boundary identification. Method 3 shown in

Fig. 4.4 (c) includes edge feature extraction prior to edge tracing, and hence the edge tracing algorithm will trace based on edge feature information. Method 4 shown in Fig. 4.4 (d) includes both edge feature extraction and multiscale boundary identification, so the edge tracing algorithm will trace based on both edge quality and edge feature information.

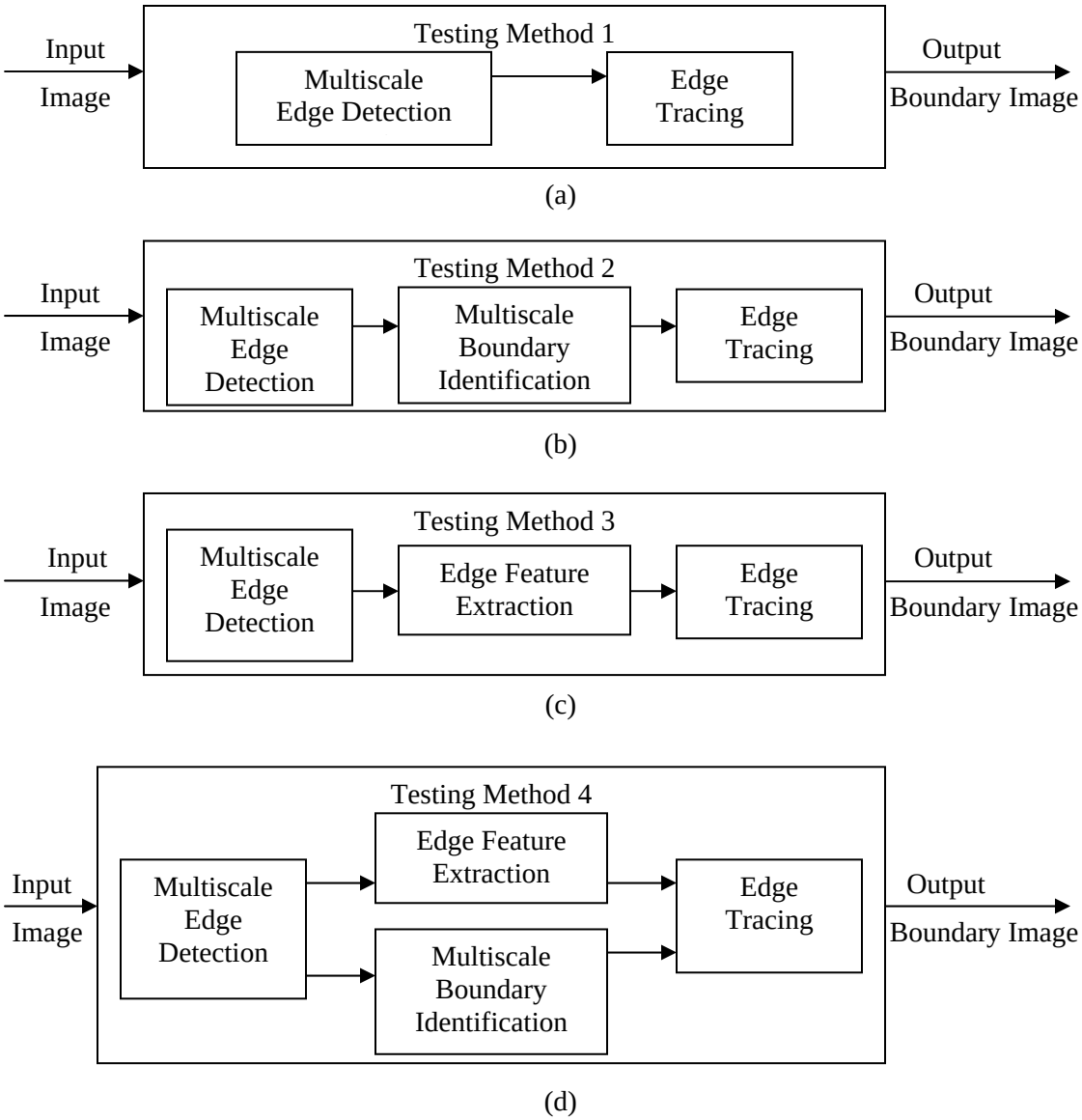


Figure 4.4: Block diagram of four testing methods.
a) Method 1; b) Method 2; c) Method 3; d) Method 4.

We will test all four methods shown in Fig.4.4 to evaluate how the multiscale boundary identification method can improve the edge tracing result. Note that Fig. 4.4 (d) is the proposed mtrack scheme shown in Fig. 3.1. The edge tracing algorithm involves different parameter settings. In our performance evaluation, the edge tracing parameters for MNI and IBSR data are the same as in [1], chapter 5.

4.4 Performance for MNI Data

This section presents the performance of synthetic MR data using the evaluation criteria and four testing methods defined in section 4.3. We first present the tracking results for image shown in Fig. 4.1 (a). We will then present the tracking performance for MNI image with different noise and non-uniformity levels.

Fig. 4.5 shows a gray/white boundary tracking result by four testing methods. The four methods use the same edge image obtained from multiscale edge detection, and the starting point chosen for edge tracing is in the same location of the edge image. We have 30 tracks per image for each method, Fig. 4.5 shows one track result out of the 30 tracks. Fig. 4.5 (a) shows the tracking result of Method 1. As there is no constraint, the track can go as long as there is an edge point available. When we add multiscale boundary identification (see Fig. 4.5 (b)), the tracking always selects the stronger edge points that have higher edge quality value, and hence it has been less affected by noise. Thus, between Fig. 4.5 (a) and (b), we can see that Method 2 is better than Method 1. When we add edge feature constraint (see Fig. 4.5 (c)), it will limit the track length, since it only accepts the edge points that satisfy the edge feature condition. Thus, between Fig. 4.5 (c) and (a), we can see that Method 3 is better than Method 1. When we use edge quality together with edge feature (see Fig. 4.5 (d)), it gives information on the edge

characteristic rather than just edge feature, and hence the tracking can select track points with higher edge quality than that with only edge feature. Between Fig. 4.5 (c) and (d), we can see that the small boundaries tracked in the left-bottom quarter in Fig. 4.5 (d) are more accurate than in Fig. 4.5 (c). Therefore, Method 4 is better than Method 3.

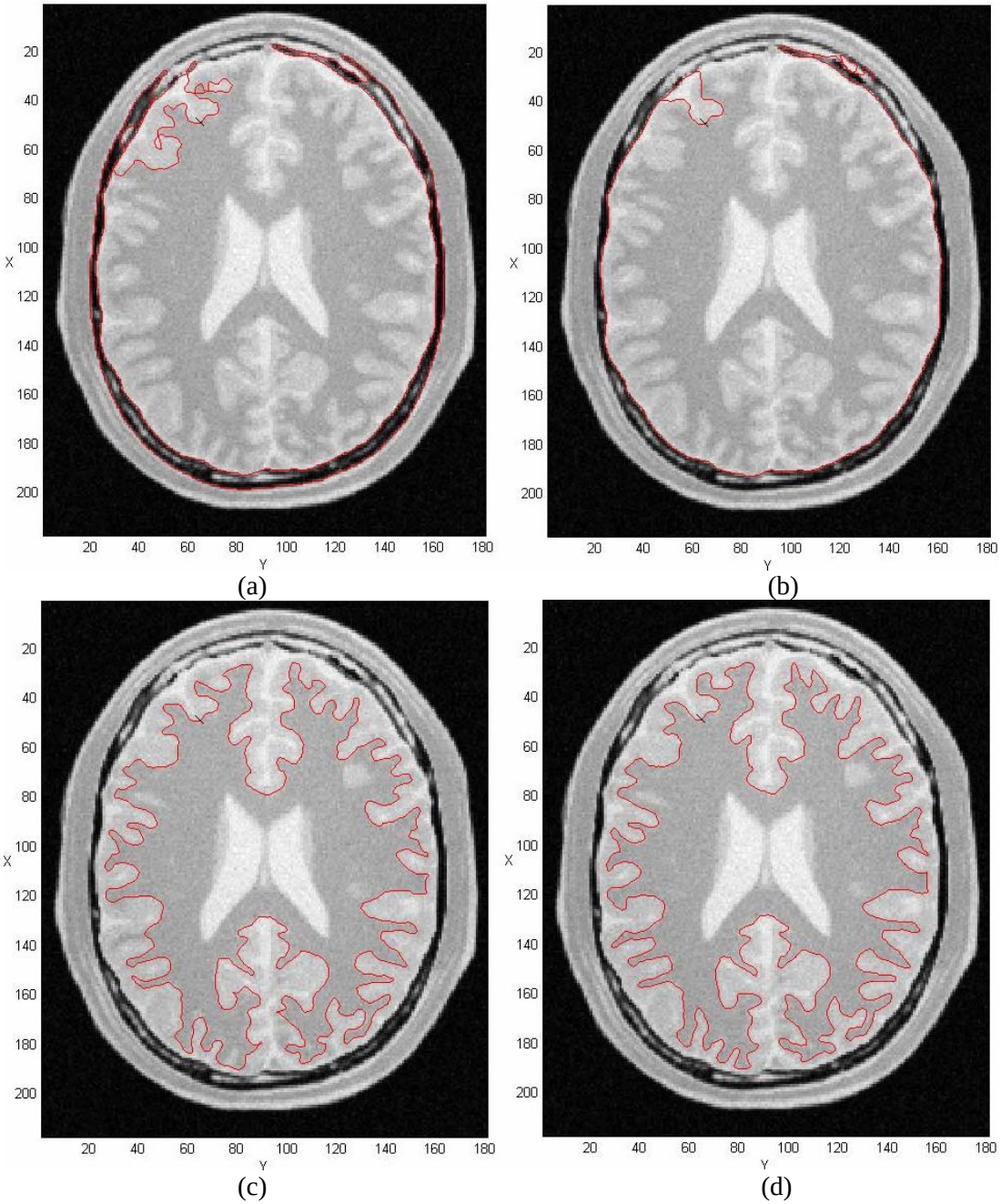


Figure 4.5: Gray/white boundary tracks for PD1_3_20 slice 95.
a) Edge tracing with Method 1; b) Edge tracing with Method 2;
c) Edge tracing with Method 3; d) Edge tracing with Method 4.

Tables 4.1-4.3 list the statistical results of 30 tracks of Fig. 4.1 (a) for each method. The mean, median, and standard deviation for N_{GP}, N_{TP}, and R are calculated using Eq. (4.5-4.13). Table 4.1 lists the number of good track points obtained. It is observed that Method 1 has higher med_{GP} and $mean_{GP}$ values than Method 2. But in Table 4.2 Method 1 has almost four times higher med_{GP} and $mean_{GP}$ values than Method 2. Thus, use the value of N_{GP} alone gives a false impression. Therefore, we should compare the two methods using the track point ratio listed in Table 4.3. We can see that Method 2 has higher track point ratio med_R and $mean_R$ than Method 1, which means Method 2 is better than Method 1.

Table 4.1: N_{GP} of PD1_3_20 slice 95.

	med_{GP}	$mean_{GP}$	std_{GP}
Method 1	201	209.4	132.8
Method 2	158.5	137.9	68.6
Method 3	1024	1007.8	449.1
Method 4	1234.5	1144.6	249.3

Table 4.2: N_{TP} of PD1_3_20 slice 95.

	med_{TP}	$mean_{TP}$	std_{TP}
Method 1	2032.5	1813.2	821.4
Method 2	590	533.2	217.7
Method 3	1623	1352.4	580.4
Method 4	1609.5	1500.4	308.9

Table 4.3: R of PD1_3_20 slice 95.

	med_R	$mean_R$	std_R
Method 1	0.126	0.155	0.141
Method 2	0.255	0.283	0.127
Method 3	0.745	0.704	0.103
Method 4	0.765	0.759	0.025

From the statistical results listed in Table 4.3, it is observed that the med_R and $mean_R$ of the 30 tracks are best when using Method 4, which listed in the last row. This means that Method 4 has the best edge tracing result than others. Also note that the standard deviation in the last row is the lowest among others (std_R of Method 4 is five times less than Method 3). This means that the edge tracing using Method 4 is less dependent of the choice of the starting point. This can be illustrated in Fig. 4.6. The X-axis is the number of tracks (from 1 to 30). The Y-axis is the track point ratio of different starting points. Apparently, the ratio R of Method 4 has less variation than Method 3. For example, see the 20th and the 21th track, Method 4 has $R \approx 0.75$, but Method 3 has $R \approx 0.45$. Therefore, from Fig. 4.6 we can see that Method 4 provides more robustness that Method 3, as the edge tracing is less dependent on the starting point in Method 4.

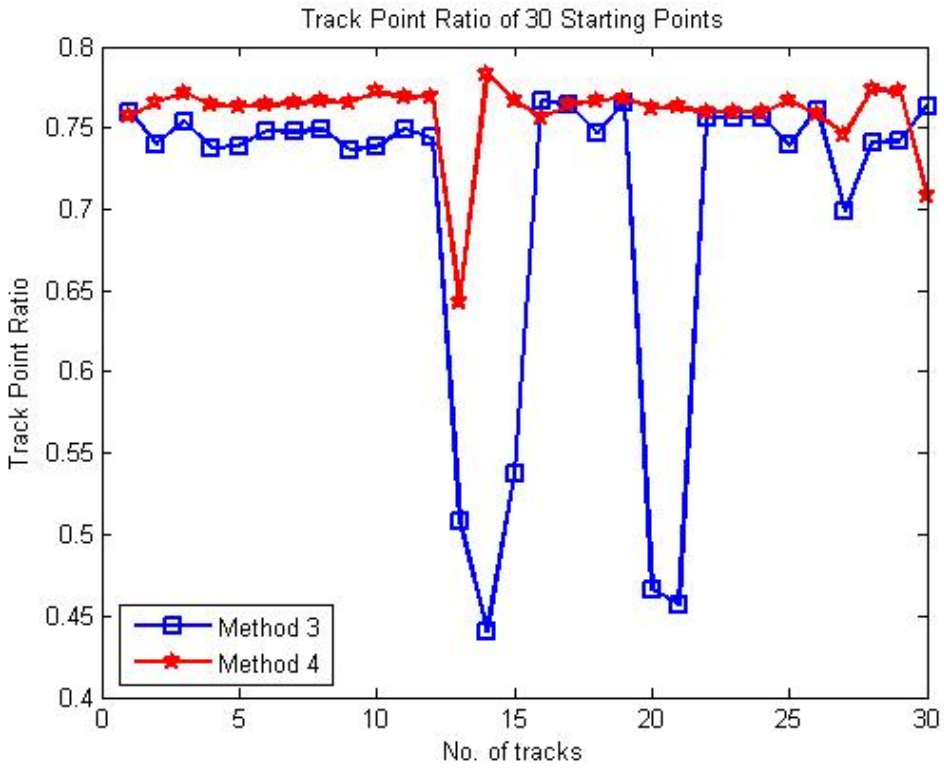


Figure 4.6: Track point ratio R of 30 tracks for PD1_3_20 slice 95.

In order to show more efficiently the statistical data collected in Table 4.1-4.3, we plot the median values of the four testing methods in Fig. 4.7. The X-axis is the total track points, and the Y-axis is the median value of the track point ratio. It is observed that in Fig. 4.7, Method 1 has the longest track length N_{TP} but the lowest track point ratio. Method 4 has a truer track length and the highest track point ratio than other methods.

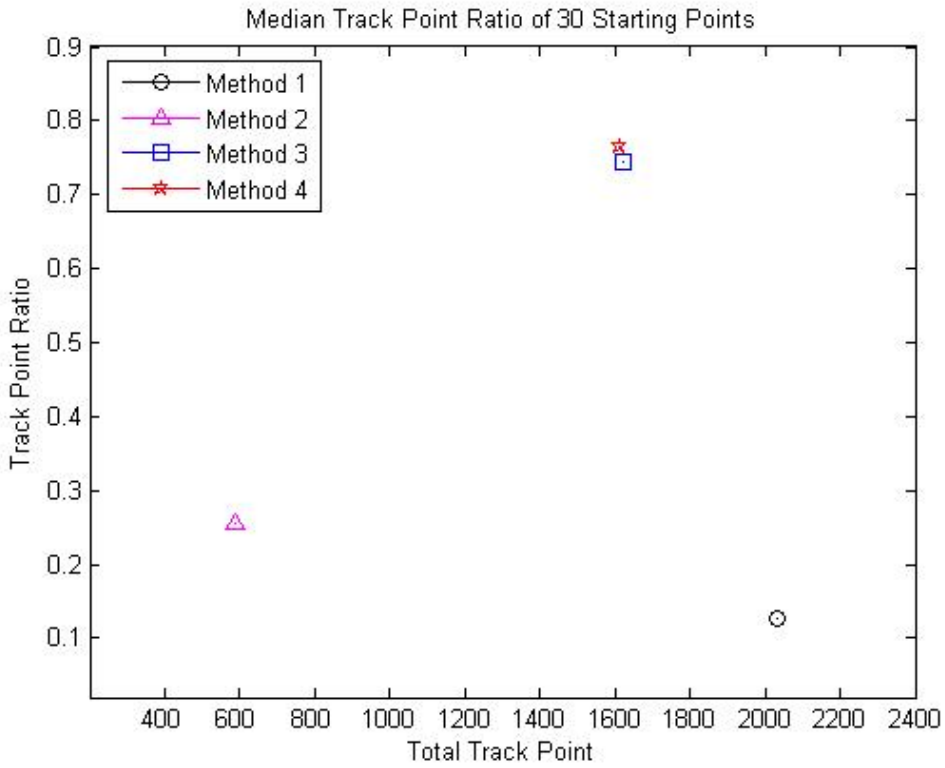


Figure 4.7: Median track point ratio of 30 tracks for PD1_3_20 slice 95.

Fig. 4.8 and Fig. 4.9, and Tables 4.4-4.6 list the tracking results for MR image with 5% noise and 20% non-uniformity in it. Similarly, we can see from Fig. 4.8 that the ratio *R* of Method 4 has less variation than Method 3, which means that edge tracing by Method 4 is less dependent on the starting point. Also from Fig. 4.9, we can see that Method 4 has the highest track point ratio among others, which means that edge tracing by Method 4 is the best among the other methods.

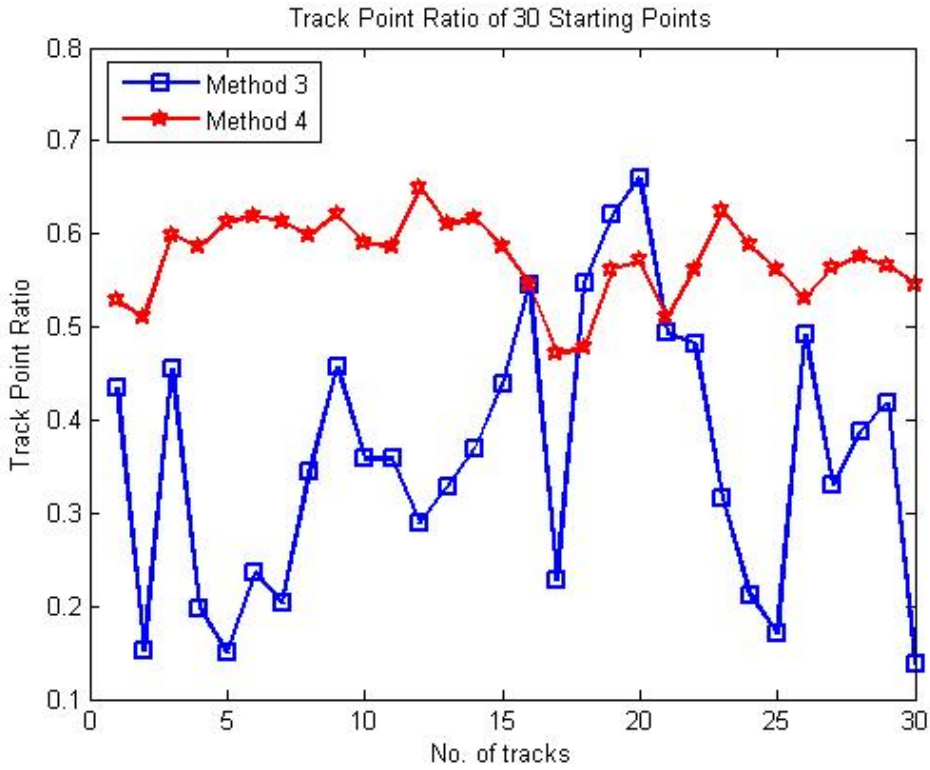


Figure 4.8: Track point ratio *R* of 30 tracks for PD1_5_20 slice 95.

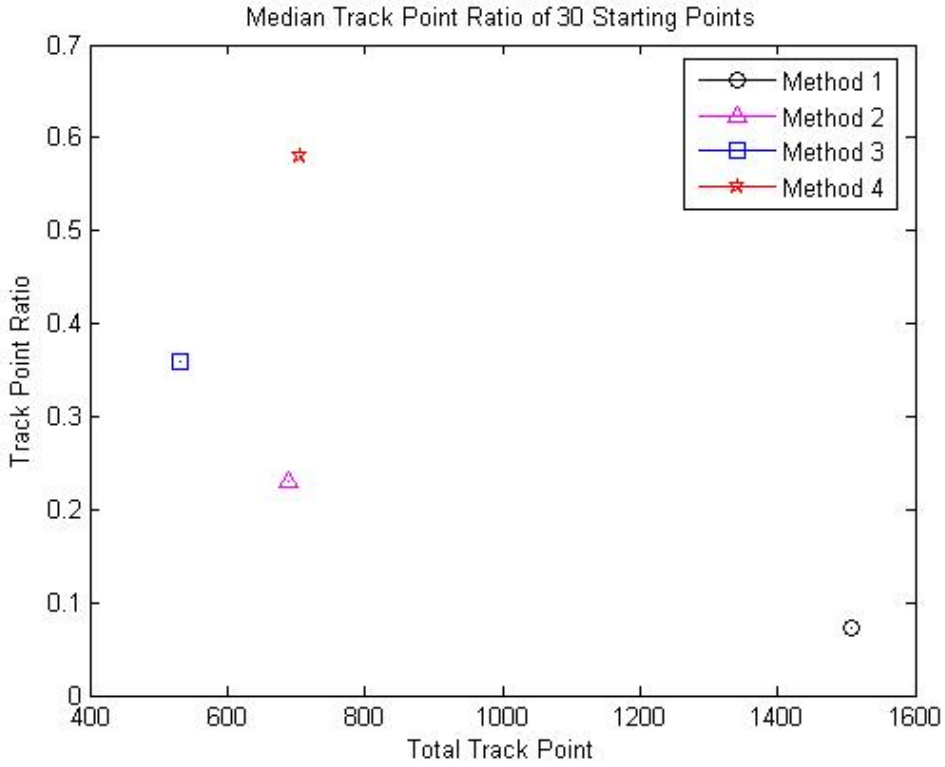


Figure 4.9: Median track point ratio of 30 tracks for PD1_5_20 slice 95.

Table 4.4: N_{GP} of PD1_5_20 slice 95.

	med_{GP}	$mean_{GP}$	std_{GP}
Method 1	99	114.9	90.6
Method 2	255.3	124	265.9
Method 3	152.5	216.9	192.1
Method 4	414	507	209.9

Table 4.5: N_{TP} of PD1_5_20 slice 95.

	med_{TP}	$mean_{TP}$	std_{TP}
Method 1	1057.5	1560.9	751.9
Method 2	689	993.5	868.9
Method 3	532.5	584.3	405.4
Method 4	706.5	898.8	394.1

Table 4.6: R of PD1_5_20 slice 95.

	med_R	$mean_R$	std_R
Method 1	0.072	0.117	0.135
Method 2	0.230	0.252	0.125
Method 3	0.356	0.360	0.144
Method 4	0.581	0.573	0.049

The performance evaluation is carried out with 1%, 3%, 5%, 7% and 9% of noise levels, and 0%, 20% non-uniformity levels in the synthetic MNI data. More evaluation results are shown in appendix A (see Table A.1 to A.24, and Fig. A.1 to A.8).

Table 4.7 lists the track point ratio of Method 3 and Method 4 on various noise levels (1%, 3%, 5%, 7% and 9%) and with 0% non-uniformity. As we see from Table 4.7, as the noise level increases, the track point ratio of both methods decreases. It is observed that at each noise level, Method 4 has higher med_R and $mean_R$ values than Method 3, which means the edge tracing by Method 4 is more accurate than Method 3. And also note that the std_R of Method 4 is always less than Method 3, which means the edge tracing by Method 4 is less dependent on the starting point.

Table 4.7: R of MNI data slice 95 with 0% non-uniformity.

Noise Level	Method 3			**Method 4**		
	med_R	$mean_R$	std_R	med_R	$mean_R$	std_R
1%	0.822	0.819	0.020	**0.823**	**0.822**	**0.008**
3%	0.631	0.595	0.138	**0.732**	**0.699**	**0.093**
5%	0.465	0.410	0.135	**0.623**	**0.623**	**0.022**
7%	0.307	0.308	0.134	**0.437**	**0.412**	**0.062**
9%	0.159	0.183	0.100	**0.228**	**0.262**	**0.087**

Table 4.8 lists the track point ratio of Method 3 and Method 4 on various noise levels (1%, 3%, 5%, 7% and 9%) and with 20% non-uniformity. Similarly, as the noise level increases, the track point ratio of both methods decreases. At each noise level, Method 4 performs better than Method 3, as it has higher med_R and $mean_R$ values and less std_R value, which means the edge tracing by Method 4 is more accurate and less dependent on the starting point.

Table 4.8: R of MNI data slice 95 with 20% non-uniformity.

Noise Level	Method 3			**Method 4**		
	med_R	$mean_R$	std_R	med_R	$mean_R$	std_R
1%	0.836	0.823	0.029	**0.817**	**0.819**	**0.011**
3%	0.745	0.704	0.103	**0.765**	**0.759**	**0.025**
5%	0.356	0.360	0.144	**0.581**	**0.573**	**0.049**
7%	0.255	0.260	0.117	**0.430**	**0.419**	**0.047**
9%	0.238	0.224	0.095	**0.305**	**0.303**	**0.068**

As we see the testing results shown in this section and appendix A, the edge tracing using Method 4 performs better than Method 3. This means that the multiscale boundary identification added into the edge tracing algorithm helps more accurately track the gray/white boundary in the synthetic MR images. Using edge quality from multiscale boundary identification, make the edge tracing less sensitive to noise, reduce the potential of tracing false edge of the boundary, and the most important thing, the edge tracing is less dependent on the choice of starting point.

4.5 Performance of IBSR data

In this section, we present the performance of the IBSR data using the evaluation criteria and four testing methods defined in section 4.3. Similar to the testing for the MNI data, we use 30 starting points on the gray/white boundary for four methods. We use the same edge image obtained from multiscale edge detection. The starting point choosing for edge tracing is in the same location of the edge image for the four methods. We have 30 tracks per image for each method. The testing results are shown in Fig. 4.10 and Fig. 4.11. The statistical testing results collected for 30 tracks are listed in Table 4.9-4.11 of four methods.

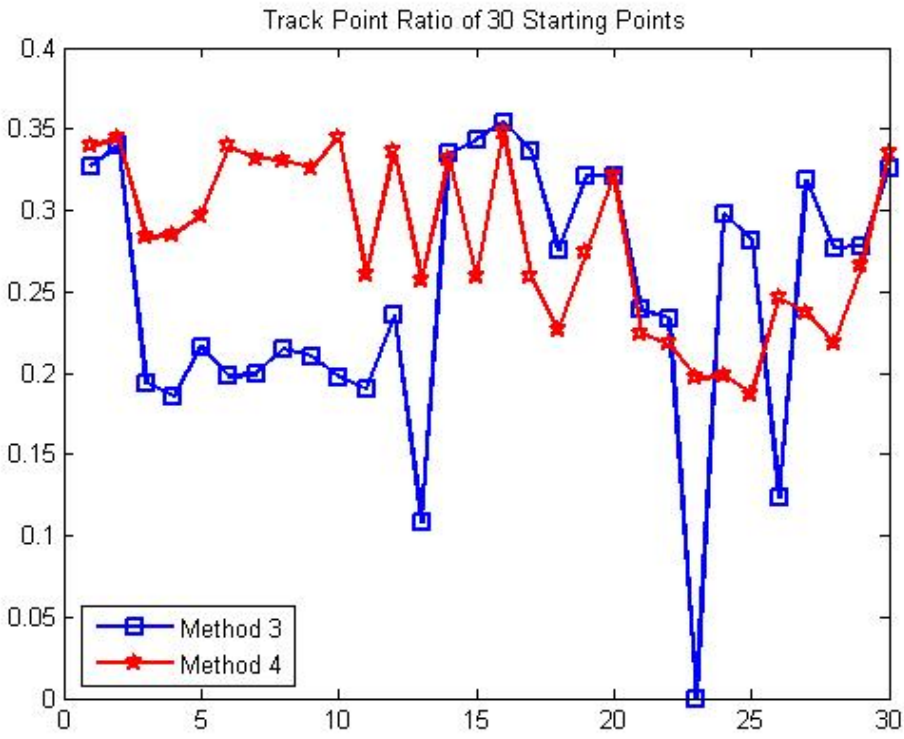


Figure 4.10: Track point ratio *R* of 30 tracks for IBSR slice 80.

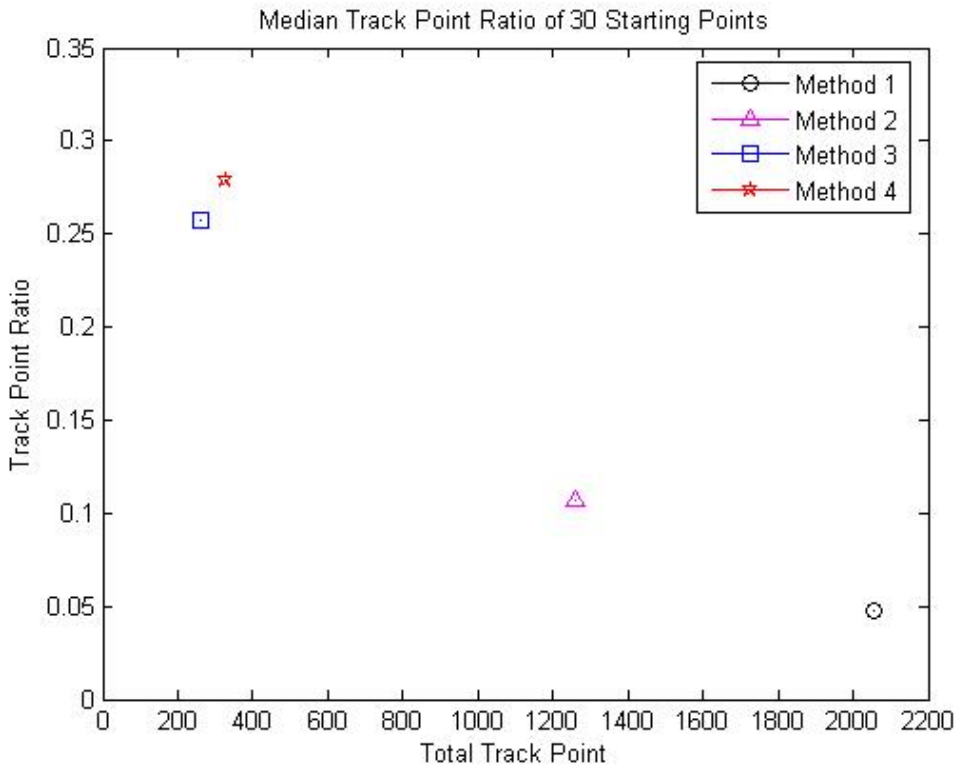


Figure 4.11: Median track point ratio of 30 tracks for IBSR slice 80.

Table 4.9: N_{GP} of IBSR slice 80.

	med_{GP}	$mean_{GP}$	std_{GP}
Method 1	89.5	78.4	27.8
Method 2	136	129	45.9
Method 3	60	62.9	40.2
Method 4	89	92.1	36.0

Table 4.10: N_{TP} of IBSR slice 80.

	med_{TP}	$mean_{TP}$	std_{TP}
Method 1	2055.5	1553.8	712.7
Method 2	1261	1214	400.5
Method 3	259	255.6	130.6
Method 4	325.5	329.9	116.3

Table 4.11: R of IBSR slice 80.

	med_R	$mean_R$	std_R
Method 1	0.048	0.079	0.084
Method 2	0.107	0.118	0.046
Method 3	0.257	0.249	0.082
Method 4	0.279	0.281	0.052

Similarly, we can see from Fig. 4.10 that the ratio *R* of Method 4 has less variation than Method 3, which means that edge tracing by Method 4 is less dependent on the starting point. Also from Fig. 4.11, we can see that Method 4 has the highest track point ratio among others, which means that edge tracing by Method 4 is better than other methods. From Table 4.9-4.11, Method 4 gives better edge tracing result than others as it has highest track point ratio, but it is not as efficient as what we have for the MNI data.

We do not have good edge tracing performance of this IBSR data, it is mainly because this real MR data has more partial volume effect compared to the MNI data, which makes the tissue boundary look very blurry (see Fig. 4.1 (b)). Note the most advantage of the multiscale boundary identification method that we proposed, is to provide the quality of the edge that can further separating the true edge from noise, so it is less sensitive to noise. That is why the proposed work improves the edge tracing result of the MNI data at different noise and non-uniformity levels.

As we mention in section 1.1, there are three main difficulties in segmentation of MR images: non-uniformity, noise, and partial volume effect. From the edge tracing performance of the MNI data and the IBSR data, we have demonstrated that the proposed method efficiently improves the edge tracing on the MR image of non-uniformity and noise, but it is not very efficient in the case of partial volume effect. However, note that the partial volume effect is not the main limitation of this edge-based segmentation software, it has been solved by adding another constraint in mtrack [1].

4.6 Summary

In this chapter, experiments have been carried out to quantitatively evaluate the performance of the proposed method. The results of these experiments show that the multiscale boundary identification method helps to improve the edge tracing result. The tracked boundary is more accurate, it is less dependent on the choice of starting point, and the edge tracing is less sensitivity to noise.

Chapter 5

Conclusions and Future Work

MR image segmentation is not a trivial task, as it involves three main difficulties: non-uniformity, noise, and partial volume effect. Thus, fully automatic segmentation is still far from satisfactory in many real situations. In this thesis, multiscale techniques have been investigated and exploited for better edge-based MR image segmentation, as the technique is less sensitive to noise. We have seen that wavelet transform modulus maxima is very useful in three ways: edge detection, WTMM chain, and edge characterization in the context of multiscale edges. First, the existence of local maxima marks the existence of a singularity (edge) in the image. In this way, the wavelet transform can be used for edge detection. Second, these modulus maxima form paths through the wavelet decay to assist in locating the edges in the original image. Third, from the modulus maxima chain we can characterize the edge structure via its regularity.

Compared to the traditional wavelet edge detection theory, we have obtained directional edge detection together with subpixel edge resolution to improve the edge detection performance. In multiscale WTMM representation, the auction algorithm has been applied for solving the chain connection problem to obtain the scale depth of the edge. Lipschitz regularity analysis gives the idea to further separate the true edge from noise through the wavelet decay. Thus, evaluating the Lipschitz regularity results using

the fuzzy logic system provides an analysis that gives the quality of the edge. Therefore, we can use edge quality to identify the boundary.

Experiments have been carried out on the proposed work in chapter 4. Performance has been evaluated in the mtrack framework for both synthetic and real MR image data with and without using multiscale boundary identification. It has been found that the boundary identification preformed by multiscale edge quality analysis improves the edge tracing significantly. Our observation can be summarized below.

1. The tracked boundary is more accurate.
2. The tracking is less dependent on the choice of starting point.
3. The edge tracing is less sensitive to noise.
4. The potential of tracing a false edge in the boundary is lower.

Overall, combining multiscale boundary identification and the edge tracing algorithm provides more robustness in the image segmentation process.

5.1 Future Work

Although the proposed work shows promising results on medical segmentation software, there are still several issues that need further research. They are listed below.

1. The multiscale technique adopted in this work is focused on edge information for boundary identification. There are multiscale segmentation methods, as mentioned in the literature review, which can give region information. These methods can be investigated as another direction for region-based image segmentation.
2. The objective function (mentioned in section 3.4) in the data association process of mtrack software is formulated as the product of elements. This objective function can be easily extended to include other valuable information as additional factors.

3. The edge quality information obtained in this work is very useful. It can be further evaluated by some techniques, such as clustering, to group different types of edge for classification, which gives different edge structure information that can be a benefit to the target tracking algorithm as well.

Bibliography

1. D. Withey, “Dynamic edge tracing: recursive methods for medical image segmentation”, Ph.D. Thesis, University of Alberta, Canada, 2006.
2. MATLAB, The Mathworks Inc.
3. A. V. Oppenheim, R. W. Schafer, and J. R. Buck, “Discrete-time signal processing”, Prentice Hall, 2nd Edition, 1999.
4. C. K. Chui, “An introduction to wavelet”, Academic Press, 1992.
5. S. Mallat and S. Zhong, “Characterization of signals from multiscale edges”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 14, No. 7, pp. 710-732, July 1992.
6. S. Mallat and W. Hwang, “Singularity detection and processing with wavelets”, *IEEE Transactions on Information Theory*, Vol. 38, No. 2, pp. 617-643, March 1992.
7. S. Mallat, “A theory for multiresolution signal decomposition: the wavelet representation”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 11, No. 7, July 1989.
8. S. Mallat, “A wavelet tour of signal processing”, Academic Press, 2nd Edition, 1999.
9. D. Marr and E. Hildreth, “Theory of edge detection”, *Proceedings of the Royal Society of London*, Vol. 207, pp. 187-217, 1980.
10. J. Canny, “A computational approach to edge detection”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 8, pp. 679-698, 1986.
11. D. J. Park, K. M. Nam, and R. H. Park, “Multiresolution edge detection techniques” *Pattern Recognition*, Vol. 28, No. 2, pp. 211-229, 1995.

12. L. Zhang and P. Bao, “Edge detection by scale multiplication in wavelet domain”, *Pattern Recognition Letters*, Vol. 23, pp. 1771-1784, 2002.
13. J. Li, “A wavelet approach to edge detection”, MSc. Thesis, Sam Houston State University, USA, 2003.
14. Y. Y. Tang, L. Yang, and L. Feng, “Characterization and detection of edges by Lipschitz exponents and MASW wavelet transform”, *Proceedings of the Fourteenth International Conference on Pattern Recognition*, Vol. 2, pp. 1572-1574, August. 1998.
15. K. Berkner and R. O. Wells, “ A new hierarchical scheme for approximating the continuous wavelet transform with applications to edge detection”, IEEE Signal Processing Letters, Vol. 6, No. 8, pp. 193-195, August 1999.
16. S. V. Raman, S. Sarkar, and K. L. Boyer, “Tissue boundary refinement in magnetic resonance images using contour-based scale space matching”, *IEEE Transactions on Medical Imagining*, Vol. 10, No. 2, pp. 109-121, June 1991.
17. K. L. Vincken, A. S. E. Koster, and M. A. Viergever, “Probabilistic multiscale image segmentation”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 2, pp. 109-120, February 1997.
18. M. R. Rezaee, P. M. J. Zwet, B. P. F. Lelieveldt, R. J. Geest, and J. H. C. Reiber, “A multiresolution image segmentation technique based on pyramidal segmentation and fuzzy clustering”, IEEE Transactions on Image Processing, Vol. 9, No. 7, pp. 1238-1248, July 2000.
19. V. Grau, M. A. Raya, C. Monserrat, M. C. Juan, and L. Marti-Bonmati, “Hierarchical image segmentation using a correspondence with a tree model”, *Pattern Recognition*,

Vol. 37, No. 1, pp. 47-59, 2004.

20. D. P. Bertsekas, "The auction algorithm for assignment and other network flow problems: a tutorial", *Interfaces*, Vol. 20, No. 4, pp. 133-149, July-August 1990.
21. D. P. Bertsekas and D. A. Castanon, "A forward/reverse auction algorithm for asymmetric assignment problems", *Computational Optimization and Applications*, Vol. 1, pp. 277-297, 1992.
22. L. A. Zadeh, "Fuzzy sets", *Information and Control*, Vol. 8, pp. 338-353, 1965.
23. L. A. Zadeh, "Fuzzy algorithms", *Information and Control*, Vol. 12, pp. 94-102, 1968.
24. W. Pedrycz and F. Gomide, "An introduction to fuzzy sets: analysis and design", MIT Press, 1998.
25. J. Dombi, "A general class of fuzzy operators, the DeMorgan class of fuzzy operators and fuzziness measures induced by fuzzy operators", *Fuzzy Sets and Systems*, Vol.8, pp. 149-163, 1982.
26. MNI data set, http://www.bic.mni.mcgill.ca/brainweb/, August 2005.
27. IBSR data set, http://www.cma.mgh.harvard.edu/ibsr/, August 2005.

Appendix A

In section 4.4, we have presented the performance of MNI slice 95 with 3% and 5% noise, and with 20% non-uniformity for the proposed method. In this appendix, we present additional performance evaluation results. These testing results are obtained using the following images (listed in Table A.1) with different noise and non-uniformity in it.

Table A.1: Summary of testing results for MNI slice 95.

Image name	Noise	Non-uniformity	Results in
PD1_1_0	1 %	0 %	Table A.2-A.4 Figure A.1
PD1_1_20	1 %	20 %	Table A.5-A.7 Figure A.2
PD1_3_0	3 %	0 %	Table A.8-A.10 Figure A.3
PD1_5_0	5 %	0 %	Table A.11-A.13 Figure A.4
PD1_7_0	7 %	0 %	Table A.14-A.16 Figure A.5
PD1_7_20	7 %	20 %	Table A.15-A.19 Figure A.6
PD1_9_0	9 %	0 %	Table A.20-A.22 Figure A.7
PD1_9_20	9 %	20 %	Table A.23-A.25 Figure A.8

From the testing results, it is observed that the proposed method is better than other methods. This means that the multiscale boundary identification added into the edge tracing algorithm helps more accurately track the gray/white matter boundary in the synthetic MR images. Using edge quality from multiscale boundary identification, make the edge tracing less sensitive to noise, reduce the potential of tracing false edge of the boundary, and the most important thing, the edge tracing is less dependent on the choice of starting point.

Table A.2: N_{GP} of PD1_1_0 slice 95.

	med_{GP}	$mean_{GP}$	std_{GP}
Method 1	651	580.5	293.6
Method 2	506	973.2	213.7
Method 3	968	328.5	305.5
Method 4	976	1056.1	134.9

Table A.3: N_{TP} of PD1_1_0 slice 95.

	med_{TP}	$mean_{TP}$	std_{TP}
Method 1	1161	1377.5	738.1
Method 2	1171.5	1214.9	410.1
Method 3	1345.5	1181.9	365.1
Method 4	1182.5	1284.6	158.3

Table A.4: *R* of PD1_1_0 slice 95.

	med_R	$mean_R$	std_R
Method 1	0.455	0.440	0.156
Method 2	0.401	0.409	0.134
Method 3	0.822	0.819	0.020
Method 4	0.823	0.822	0.008

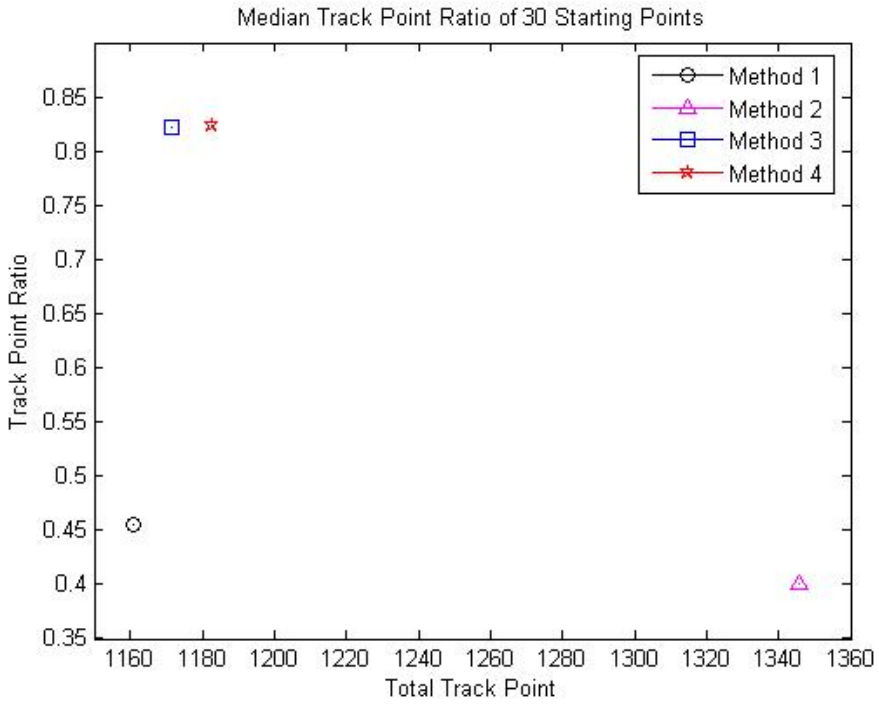


Figure A.1: Median track point ratio of 30 tracks for PD1_1_0 slice 95.

Table A.5: N_{GP} of PD1_1_20 slice 95.

	med_{GP}	$mean_{GP}$	std_{GP}
Method 1	694.5	634	261.4
Method 2	525	973.2	489.7
Method 3	525	571.9	224.8
Method 4	1305.5	969.1	401.4

Table A.6: N_{TP} of PD1_1_20 slice 95.

	med_{TP}	$mean_{TP}$	std_{TP}
Method 1	1451	1555.2	805.6
Method 2	1158	1388.5	678.7
Method 3	640	689.4	258.1
Method 4	1600	1186.1	496.7

Table A.7: R of PD1_1_20 slice 95.

	med_R	$mean_R$	std_R
Method 1	0.456	0.432	0.111
Method 2	0.508	0.449	0.137
Method 3	0.836	0.823	0.029
Method 4	0.817	0.819	0.011

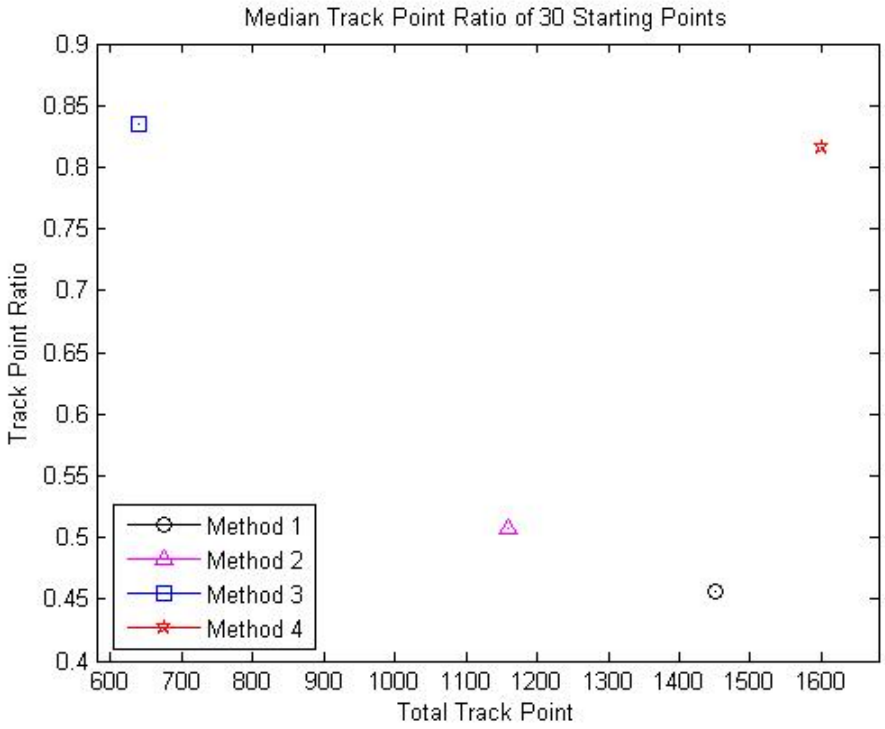


Figure A.2: Median track point ratio of 30 tracks for PD1_1_20 slice 95.

Table A.8: N_{GP} of PD1_3_0 slice 95.

	med_{GP}	$mean_{GP}$	std_{GP}
Method 1	195.5	245.1	174.4
Method 2	212	225.6	159.9
Method 3	382	328.5	192.2
Method 4	569.5	667.6	332.4

Table A.9: N_{TP} of PD1_3_0 slice 95.

	med_{TP}	$mean_{TP}$	std_{TP}
Method 1	2232	2085	513.3
Method 2	870.5	919.6	460.2
Method 3	590.5	534.8	272.2
Method 4	753	932.4	423.3

Table A.10: R of PD1_3_0 slice 95.

	med_R	$mean_R$	std_R
Method 1	0.096	0.125	0.090
Method 2	0.303	0.276	0.154
Method 3	0.631	0.595	0.138
Method 4	0.732	0.699	0.093

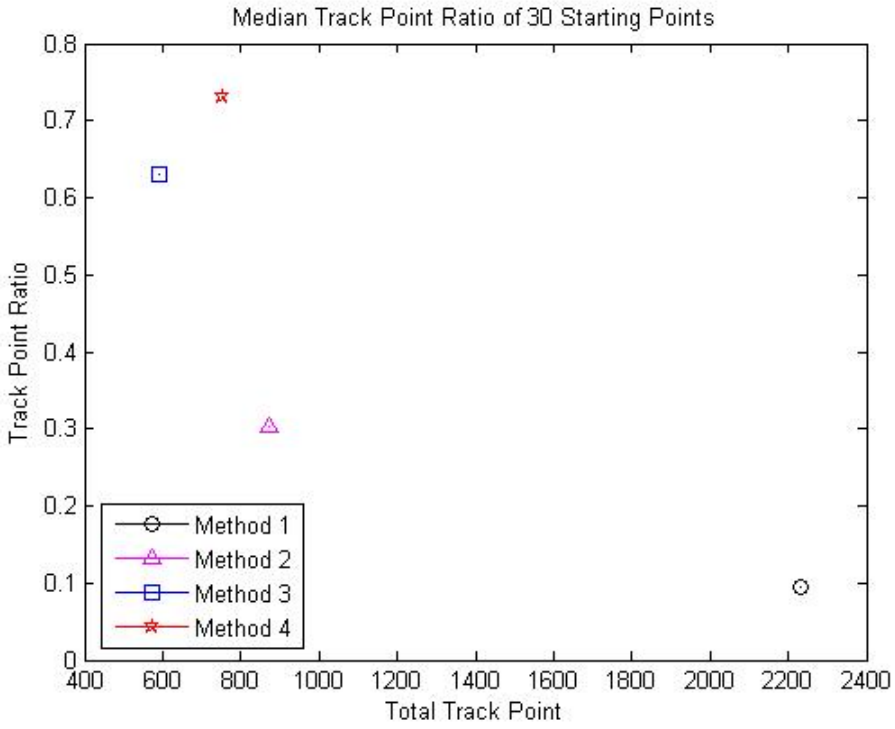


Figure A.3: Median track point ratio of 30 tracks for PD1_3_0 slice 95.

Table A.11: N_{GP} of PD1_5_0 slice 95.

	med_{GP}	$mean_{GP}$	std_{GP}
Method 1	78.5	87.9	63.2
Method 2	97	182.2	179.6
Method 3	435	424.9	322.3
Method 4	1023	998.6	90.9

Table A.12: N_{TP} of PD1_5_0 slice 95.

	med_{TP}	$mean_{R}$	std_{TP}
Method 1	555	638.2	466.8
Method 2	653	931.1	857.9
Method 3	926	907.6	600.5
Method 4	1622.5	1601.6	122.0

Table A.13: R of PD1_5_0 slice 95.

	med_{R}	$mean_{R}$	std_{R}
Method 1	0.146	0.196	0.139
Method 2	0.186	0.211	0.129
Method 3	0.465	0.410	0.135
Method 4	0.623	0.623	0.022

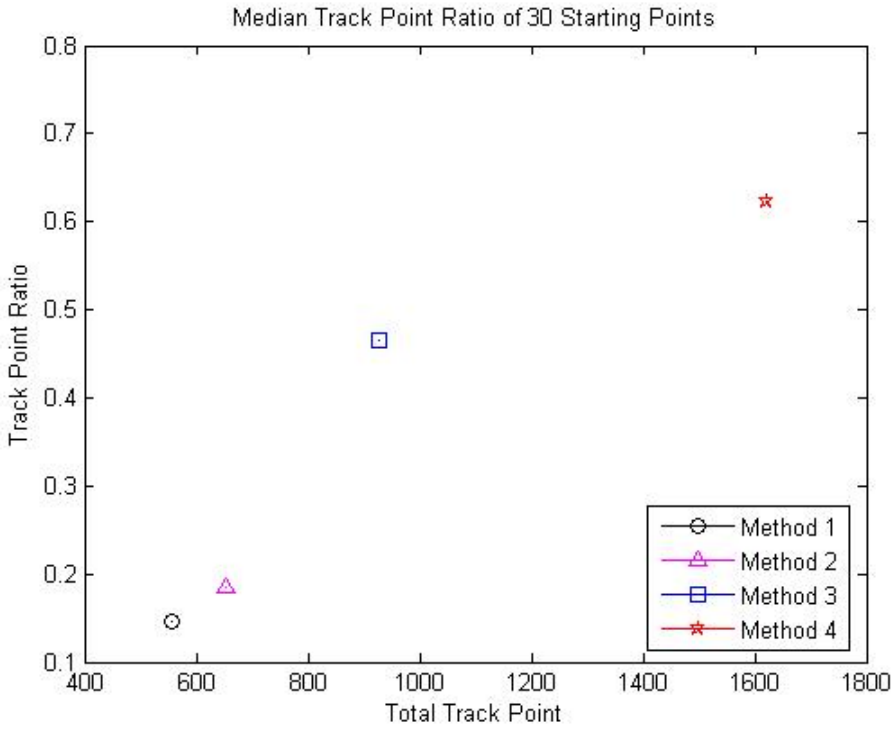


Figure A.4: Median track point ratio of 30 tracks for PD1_5_0 slice 95.

Table A.14: N_{GP} of PD1_7_0 slice 95.

	med_{GP}	$mean_{GP}$	std_{GP}
Method 1	74	78.3	44.7
Method 2	86	112.9	92.9
Method 3	97	113.7	96.8
Method 4	581.5	469.2	222.1

Table A.15: N_{TP} of PD1_7_0 slice 95.

	med_{TP}	$mean_{R}$	std_{TP}
Method 1	1455.5	1461.4	803.6
Method 2	754.5	734.3	435.1
Method 3	295.5	393.0	324.6
Method 4	1413.5	1177.2	569.8

Table A.16: R of PD1_7_0 slice 95.

	med_{R}	$mean_{R}$	std_{R}
Method 1	0.053	0.096	0.116
Method 2	0.190	0.197	0.153
Method 3	0.307	0.308	0.134
Method 4	0.437	0.412	0.062

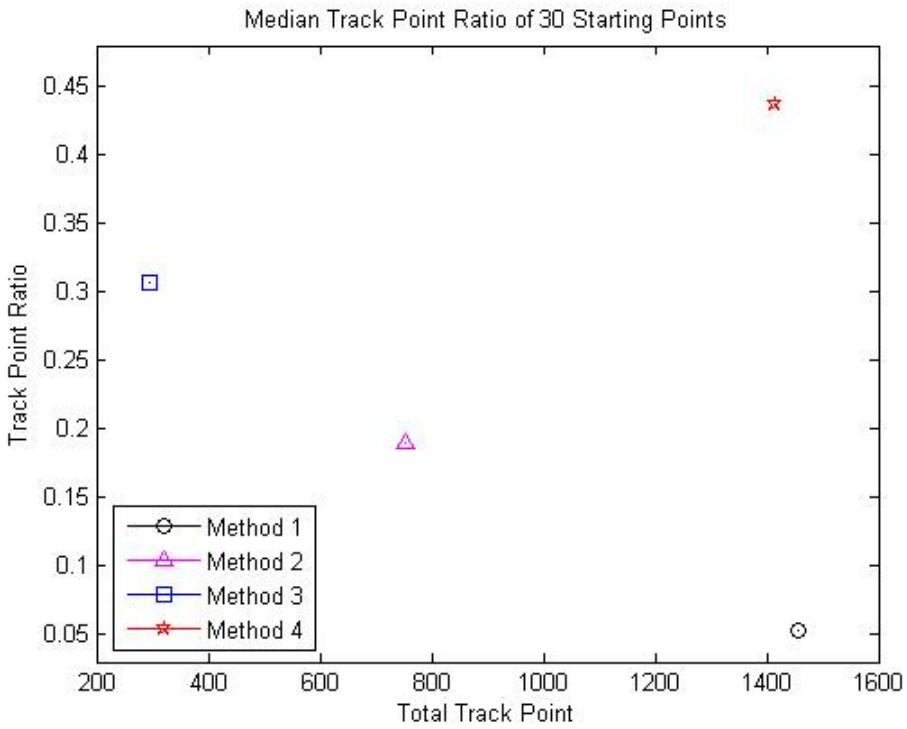


Figure A.5: Median track point ratio of 30 tracks for PD1_7_0 slice 95.

Table A.17: N_{GP} of PD1_7_20 slice 95.

	med_{GP}	$mean_{GP}$	std_{GP}
Method 1	33.5	54.4	51.2
Method 2	49.5	122.2	133.2
Method 3	36	52.8	54.7
Method 4	232.5	287.8	159.6

Table A.18: N_{TP} of PD1_7_20 slice 95.

	med_{TP}	$mean_{R}$	std_{TP}
Method 1	1117.5	972.8	839.9
Method 2	781	974.3	800.6
Method 3	155	231.7	219.1
Method 4	551	694.8	393.9

Table A.19: *R* of PD1_7_20 slice 95.

	med_{R}	$mean_{R}$	std_{R}
Method 1	0.079	0.141	0.175
Method 2	0.125	0.142	0.121
Method 3	0.255	0.260	0.117
Method 4	0.430	0.419	0.047

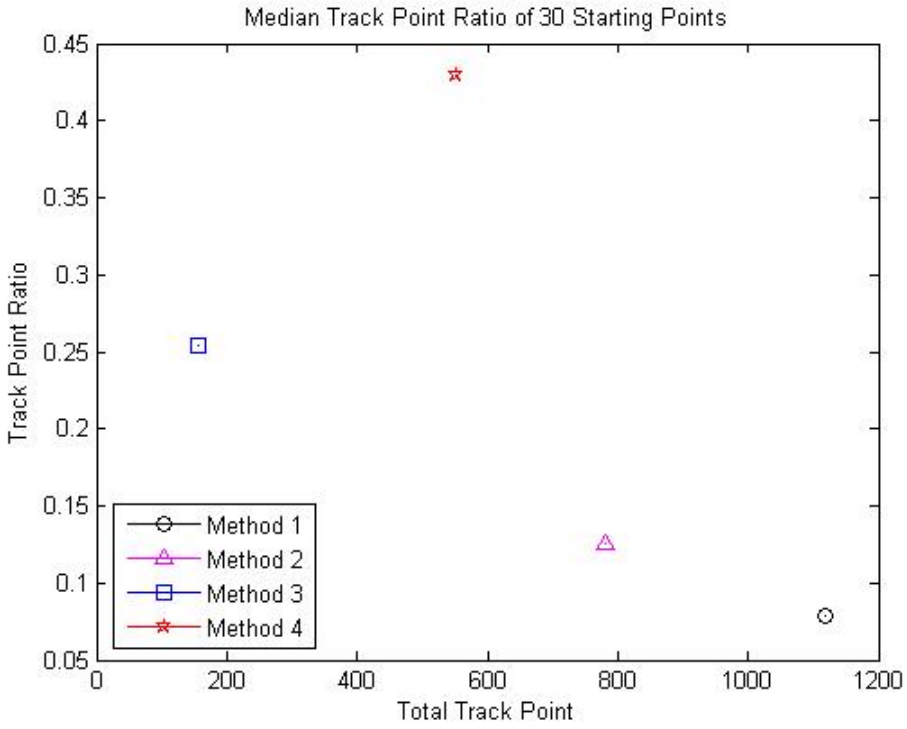


Figure A.6: Median track point ratio of 30 tracks for PD1_7_20 slice 95.

Table A.20: N_{GP} of PD1_9_0 slice 95.

	med_{GP}	$mean_{GP}$	std_{GP}
Method 1	38	38.7	23.4
Method 2	134.5	169.8	135.5
Method 3	64	82.9	79.6
Method 4	189	198.9	107.9

Table A.21: N_{TP} of PD1_9_0 slice 95.

	med_{TP}	$mean_{R}$	std_{TP}
Method 1	1408.5	1253.4	804.5
Method 2	793.5	1271	1085.9
Method 3	420	487.8	329.9
Method 4	828.5	852.6	522.0

Table A.22: R of PD1_9_0 slice 95.

	med_{R}	$mean_{R}$	std_{R}
Method 1	0.033	0.058	0.061
Method 2	0.133	0.149	0.074
Method 3	0.159	0.183	0.100
Method 4	0.228	0.262	0.087

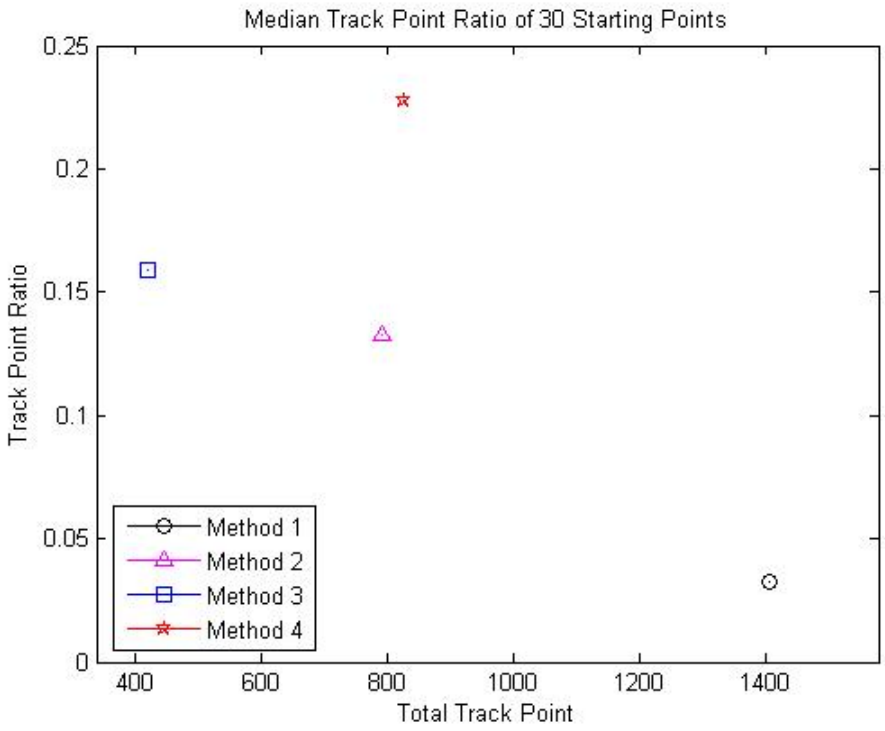


Figure A.7: Median track point ratio of 30 tracks for PD1_9_0 slice 95.

Table A.23: N_{GP} of PD1_9_20 slice 95.

	med_{GP}	$mean_{GP}$	std_{GP}
Method 1	89	85.3	61.0
Method 2	43	79.1	69.8
Method 3	77.5	83.2	69.8
Method 4	205	241.6	156.4

Table A.24: N_{TP} of PD1_9_20 slice 95.

	med_{TP}	$mean_{R}$	std_{TP}
Method 1	640.5	929.9	805.0
Method 2	1517.5	1219.4	854.0
Method 3	302.5	381.6	294.4
Method 4	708	857.1	609.0

Table A.25: *R* of PD1_9_20 slice 95.

	med_{R}	$mean_{R}$	std_{R}
Method 1	0.157	0.147	0.100
Method 2	0.081	0.155	0.172
Method 3	0.238	0.224	0.095
Method 4	0.305	0.303	0.068

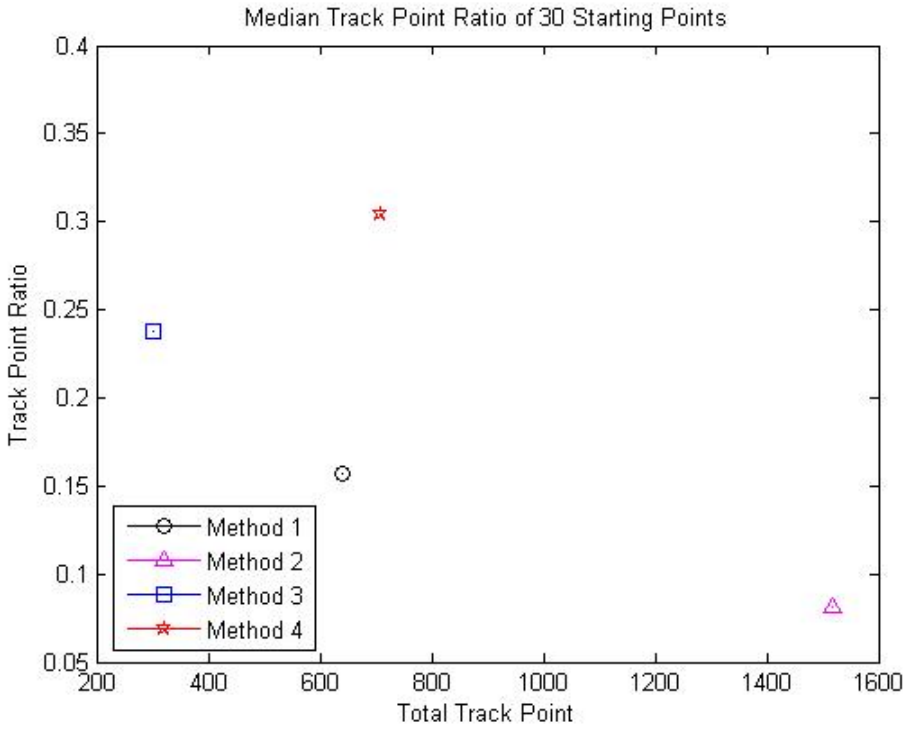


Figure A.8: Median track point ratio of 30 tracks for PD1_9_20 slice 95.