SEGMENTATION OF TORSO CT IMAGES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ONUR ALİ DEMİRKOL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

JULY 2006

Approval of the Graduate School of Natural and Applied Sciences.

Prof. Dr. Canan Özgen
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. İsmet Erkmen
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Asst. Prof. Dr. Yeşim SERİNAĞAOĞLU DOĞRUSÖZ
Supervisor

Examining Committee Members

Prof. Dr. Nevzat Güneri GENÇER (METU, EEE)

Asst. Prof. Dr. Yeşim SERİNAĞAOĞLU DOĞRUSÖZ (METU, EEE)

Prof. Dr. Murat EYÜBOĞLU (METU, EEE)

Assoc. Prof. Dr. A. Aydın ALATAN (METU, EEE)

Assoc.Prof. Dr. Kemal ARDA (Ufuk Univ., Radiology Dept)

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name:

Signature :

ABSTRACT

SEGMENTATION OF TORSO CT IMAGES

DEMİRKOL, Onur Ali
M.S., Department of Electrical and Electronics Engineering
Supervisor: Assistant Prof. Dr. Yeşim SERINAĞAOĞLU DOĞRUSÖZ

June 2006, 193 pages

Medical imaging modalities provide effective information for anatomic or metabolic activity of tissues and organs in the body. Therefore, medical imaging technology is a critical component in diagnosis and treatment of various illnesses. Medical image segmentation plays an important role in converting medical images into anatomically, functionally or surgically identifiable structures, and is used in various applications. In this study, some of the major medical image segmentation methods are examined and applied to 2D CT images of upper torso for segmentation of heart, lungs, bones, and muscle and fat tissues. The implemented medical image segmentation methods are thresholding, region growing, watershed transformation, deformable models and a hybrid method; watershed transformation and region merging. Moreover, a comparative analysis is performed among these methods to obtain the most efficient segmentation method for each tissue and organ in torso.

Some improvements are proposed for increasing accuracy of some image segmentation methods.

Keywords: Medical image segmentation, torso, watershed transformation, region growing, deformable models.

ÖZ

GÖVDENİN BİLGİSAYARLI TOMOGRAFİ GÖRÜNTÜLERİNİN BÖLÜTLENMESİ

DEMİRKOL, Onur Ali

Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü

Tez Yöneticisi: Yard.Doç.Dr. Yeşim SERİNAĞAOĞLU DOĞRUSÖZ

Temmuz 2006,193 sayfa

Tıbbi görüntüleme modaliteleri vücuttaki dokuların ve organların anatomik veya metabolik aktiviteleri hakkında etkin bilgiler sağlamaktadır. Bu yüzden tıbbi görüntüleme teknolojileri çeşitli hastalıkların teşhis ve tedavisinde kritik rol oynayan bir bileşendir. Tıbbi imge bölütleme, tıbbi görüntülerin anatomik, fonksiyonel ve cerrahi olarak tanımlanabilir yapılara dönüştürülmesinde önemli bir rol oynamaktadır. Bu çalışmada, başlıca tıbbi imge bölütleme metotlarından bazıları incelenmiş ve üst gövdede yer alan kalp, akciğerler, kemikler, kas ve yağ dokusunun bölütlenmesi için iki boyutlu bilgisayarlı tomografi (BT) görüntülerine uygulanmıştır. Tıbbi imge bölütleme metotları olarak eşikleme, alan büyütme, akaçlama havzası dönüşümü, deforme olabilen modeller ve karma metot akaçlama havzası dönüşümü ve alan birleştirme uygulanmıştır. Buna ek olarak, uygulanan metotlar arasında gövdede yer alan her doku ve organ için en etkin bölütleme metodunu bulmak için karşılaştırmalı analiz yapılmıştır. Bazı imge bölütleme metotlarının doğruluğunu arttırmak için birtakım iyileştirmeler önerilmiştir.

Anahtar Kelimeler: Tıbbi imge bölütleme, Gövde, Akaçlama havzası dönüşümü, Alan büyütme, Deforme olabilen modeller

ACKNOWLEDGEMENTS

Firstly, I would like thank to my advisor Assistant Prof. Dr. Yeşim Serinağaoğlu Doğrusöz for her guidance and suggestions throughout this research. Also I would like to thank to Assoc. Prof. Dr. Kemal Arda for his precious technical support on analyzing, and providing CT images. Special thanks to my parents and business friends for their supports.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

SRG	: Seeded Region-Growing
CT	: Computed Tomography
WTRM	: Watershed Transformation and Region Merging
NR	: Negative Rate metric
MRI	: Magnetic Resonance Imaging
PET	: Positron Emission Tomography

CHAPTER 1

INTRODUCTION

Medical imaging modalities such as Computerized Tomography (CT), Magnetic Resonance Imaging (MRI), and Positron Emission Tomography (PET) etc. provide effective information on anatomic or metabolic activity of tissues in the body. Therefore, medical imaging technology is a critical component in diagnosis and treatment activities.

Medical image segmentation plays an important role in converting medical images into anatomically, functionally or surgically identifiable structures. Medical image segmentation is used in various biomedical applications such as virtual endoscopy, to assist surgical planning using 3D reconstruction of a patient's anatomy, treatment planning, diagnosis, measuring tissue volumes, anatomical studies and for building 3D realistic models that correctly represent the electrical and geometrical properties of the region of interest for electromagnetic source localization studies.

The choice of image segmentation method depends on the specific application, imaging modality and some other factors. For example, requirements for segmentation of lungs might be different from that of fat or muscle tissue. There is currently no single segmentation method that yields acceptable results for every tissue or every medical image. There exist more general methods that can be applied to a variety of data. However, methods that are specialized to particular applications can often achieve better performances. Selection of an appropriate approach to a segmentation problem can therefore become a difficult dilemma [1].

In this study we performed segmentation of main tissues and organs in torso using 2D CT images. Some of the fundamental medical image segmentation methods and a hybrid method are examined and their performances are compared to obtain the most efficient segmentation method for each tissue and organ.

1.1. Definition of Image Segmentation Problem

Image segmentation can be defined as partitioning of an image into non-overlapping, consistent regions, which are homogeneous with respect to some characteristics such as intensity or texture. For a given image *I*, segmentation results in *k* subsets S_k ($S_k \subset I$), union of S_k results in original image *I*. Then sets S_k must satisfy

$$I = \bigcup_{i=1}^{k} S_k$$

where $S_k \cap S_j = \phi$ for $j \neq k$, and S_k are connected. Ideally, the sets obtained after segmentation correspond to distinct anatomical structures or regions of interest [2].

Image segmentation is one of the most challenging problems of digital image processing and many different approaches and methods have been proposed in the literature. Since image segmentation utilizes different approaches, a single standard method of image segmentation has not emerged. Rather, there are a collection of methods that have received some degree of popularity in different applications. Haralick and Shapiro [3] stated that for a good image segmentation, regions of the segmented image should be uniform with respect to some characteristics, such as gray tone. Region interiors of the segmented regions should be smooth and without many holes. The adjacent segmented regions should have a significant contrast with respect to the characteristic on which they are uniform. Moreover, the boundaries of the segmented regions should not be ragged and spatially accurate.

1.1.1. Overview of Medical Image Segmentation Methods

There are three main approaches to image segmentation,

- **Manual segmentation**

  In manual segmentation, pixels or regions are manually grouped depending on a feature like texture or intensity range. It is obvious that this is a time consuming method, especially for medical applications where an image dataset is composed of several 2D slices.

- **Automatic Segmentation**

  Automatic segmentation is a difficult process because of the complexity and variation of images. Automatic segmentation methods generally need a prior knowledge to lead acceptable results. Some atlas-guided segmentation methods are used for automatic segmentation, where a prior knowledge is given by a standard atlas or template.

- **Semiautomatic Segmentation**

  Semiautomatic segmentation methods combine the benefits of both manual and automatic segmentation. Generally, semiautomatic segmentation is performed by incorporating prior information required for automatic segmentation by manual interaction. The trade-off between manual interaction and performance is an important consideration in any segmentation application. Manual interaction can improve performance with knowledge of an operator. On the other hand, for large datasets, this can be time consuming.

There are a collection of methods that have received some degree of popularity in different medical image segmentation applications. These methods can be divided into 4 main categories: pixel-based, region based, deformable models, and atlas-guided methods. Although each method is described separately, multiple techniques are generally used together to obtain more accurate results. In the following sections, brief descriptions of these methods are given.

1.1.1.1. Pixel based methods

Pixel based segmentation methods takes only the gray level value of the pixel into account when dividing the image into different classes. Thresholding, clustering and Markov random field approaches can be classified as pixel based methods.

- **Thresholding**

Thresholding approaches are used to segment images by creating binary partitioning of the image intensities [2]. Thresholding can be done as bi-level thresholding or multi-level thresholding. In bi-level thresholding, segmentation is performed by grouping all the pixels with intensity greater than the threshold into one class, and all the other pixels into another class. Multi-level thresholding procedure is similar but with multiple threshold values. The threshold value or values can be provided manually by the user, obtained from the image histogram or from other features. Since thresholding approach does not take into account the spatial characteristics of the image, it is very sensitive to noise and intensity inhomogenities. Thresholding is not used as a segmentation method alone, rather it is generally used as an initial step for hybrid algorithms. Detailed information for thresholding is given in Section 3.1.

- **Clustering**

Clustering may be defined as the process of organizing objects into groups whose members are similar in some way. Three commonly used clustering algorithms in medical image segmentation are the K-means or ISODATA algorithm [4], the fuzzy C-means algorithm [5], and the expectation-maximization (EM) algorithm [6]. These algorithms do not use training data. Thus, they are termed unsupervised methods. In order to compensate for the lack of training data, clustering methods iterate between segmenting the image and characterizing the properties of each class. In a sense, clustering methods train themselves using the available data. Clustering algorithms do not directly incorporate spatial modeling and can therefore be sensitive to noise and intensity inhomogenities. On the other hand, clustering based methods are significantly fast [2].

- **Markov Random Field Models**

Markov random field (MRF) modeling is not a segmentation method itself, rather a statistical model that is widely used in medical image segmentation methods for modeling the spatial interactions between neighboring or nearby pixels. In medical image segmentation applications, MRF models are used to take into account the fact that most pixels belong to the same class as their neighboring pixels [2]. MRF models are often incorporated into clustering segmentation algorithms, and also used in modeling intensity inhomogenities that can occur in MR images and texture properties.

1.1.1.2. Region Based Methods

Region-based methods try to isolate areas of images that are homogeneous according to a given set of characteristics. Region growing and watershed transformation can be classified as region based methods.

- **Region Growing**

Seeded region growing is one of the most simple and popular region-based segmentation algorithms. Region growing algorithm takes seeds, which can be individual points or connected components, to mark each of the regions to be segmented [2]. Seeds can be manually selected by an operator, or can be selected automatically. Starting from the seed regions, region growing algorithm grows the regions around seed regions based upon a certain homogeneity criteria. Detailed information for seeded region growing is given in Section-3.2.

- **Watershed Transformation**

Watershed Transformation can be classified as a region-based segmentation approach, and it is coming from the field of mathematical morphology [7]. It is widely used in many fields of image processing including medical image segmentation, since it produces a complete division of the image in separated regions. The watershed transform is computed on the gradient image, and after the transformation, the image is divided into two classes: the catchment basins and the watershed lines. *Catchment basins* correspond to the homogeneous grey level regions. The boundaries of catchment basins, which are located at high gradient points, correspond to *watershed lines*. Detailed information for watershed transformation is given in Section-3.3.

1.1.1.3. Deformable models

Deformable models are widely used in medical image segmentation applications [2]. These models are physically motivated techniques for delineating region boundaries using closed curves or surfaces that deform to minimize an energy function. Detailed information on deformable models is given in Section-3.4.

1.1.1.4. Atlas-guided methods

Atlas-guided approaches are a powerful tool for medical image segmentation when a standard atlas or template is available [2]. The atlas is generated by compiling information on the anatomy that requires segmenting. This atlas is then used as a reference frame for segmenting new images. The standard atlas-guided approach treats segmentation as a registration problem. It first finds a one-to-one transformation that maps a pre-segmented atlas image to the target image that requires segmenting. Because the atlas is already segmented, all structural information is transferred to the target image.

1.2. Previous Work

In the field of medical image segmentation many studies have been carried out by using various methods. In the field of segmentation of the brain, Akalın and Gencer [8] proposed a semi-automatic hybrid algorithm that makes use of snakes, morphological operations, thresholding and region growing for segmentation of main tissues in head from the multimodal MR images. Using the results of segmentation and mesh generation, a realistic head model is developed. In this study realistic head model is used for solving the forward and inverse problem in Electro Magnetic Source

Localization (EMSI) studies. Ferrant et all [9] used a deformable atlas for the automatic localization and identification of brain structures in MR images. The atlas is built as a multi-object set of 3D triangulated closed surfaces, each representing a given brain structure, and sharing its faces with neighboring structures. They also proposed an active surface model to refine the local shape of the globally registered structures. Zhang and Smith [10] proposed a 3-D fully automatic approach for brain MR image segmentation. The proposed method is based on a Hidden Markov Random Field-Expectation Maximization (HMRF-EM) framework, which is a combination of the HMRF model and the associated MRF-MAP estimation and the EM fitting procedures. In this study segmentation of gray matter (GM), white matter (WM) and cerebrospinal fluid (CSF) were performed. Baillard and Barillot [11,12] used the level set formulation to obtain deformable surfaces and performed automatic segmentation of brain in MR images. In this study, an initial deformable surface was located inside the brain and the brain surface was extracted iteratively.

Many studies have been carried out for segmentation of organs or tissues in the torso, using different methods ranging from model-based methods to traditional methods. Robison and Ghita used a hybrid method which integrates region growing and unsupervised clustering methods for segmentation of fat tissue in the whole body from MRI data [13]. This study is used for quantification of total body fat volume, and medical analysis of obesity. Zhou and Hara [14] proposed an automatic segmentation method for segmentation of skeleton, fat and skin from torso CT images. In this method, the human body is divided into four disjoint regions: skeleton, fat, muscle and air. First, a bi-level thresholding is applied to remove the background, and following thresholding a multi-level thresholding is applied to obtain the skeleton regions. Following multi-level thresholding, a 3D region growing is applied to refine the results of segmentation for the skeleton. Segmentation of the skin is done by detecting candidate regions whose distance to body surface is less than 5mm, and the gray level value is in a certain range. Fat tissue is extracted using gray-level thresholding from the CT image. Sachse and Werner [15] used region growing and split-merge methods for segmentation of muscle, bones and fat in the Visible Human dataset. Martin and Bordes [16] used region growing method to extract liver, tumors and blood vessels from CT images of lower torso. A very comprehensive hybrid method for segmentation of thin structures in CT images is proposed by Gazit and Kimmel [17]. In this method, researchers integrated 3D edge detection, geodesic active surfaces and geometric active contour models for segmentation of blood vessels in 3D. Burton and Isaacs used thresholding, morphological operations and region growing for segmentation of lungs from MR images [18]. In this study, researchers generated a human lung boundary model with the goal of creating a framework for anatomically realistic models of the human airway network. Heuberger and Geissbuhler used thresholding for separating lungs from other low density tissues fat and background [19]. Following thresholding, background is removed and undesired holes are filled using morphological operations for obtaining a satisfactory segmentation.

In the field of segmentation of heart, since the heart is a dynamic organ that deforms during pumping blood to body, generally 4D images are used. These images are obtained during one cardiac cycle. Moreover, segmentation of heart mostly focuses on segmentation of intra-organ structures like Left Ventricle (LV), Right Ventricle (RV), and heart muscle. Redwood and Camp [20] proposed a semiautomatic method for segmentation of heart from CT images using traditional methods like thresholding, morphological operations and region growing. Thresholding and some morphological operations are applied to separate the heart from surrounding tissues. Following thresholding process, region growing is applied in 3D for segmentation of vessels or the chambers of the heart. Deformable models were widely used for segmentation of heart. Metaxas and Chen proposed a new class of deformable model that integrates both shape/edge energy terms and intensity/texture energy terms for segmentation of the heart [21]. Moreover, McInerney and Terzopoulos used dynamic deformable models for segmentation of LV from 4D CT images [22]. Neubauer and Wegenkittl [23] developed a new skeleton-based segmentation method for segmentation of LV and heart muscle using 4D-CT images. Atlas based methods are also widely used for segmentation of the heart. Ortiz and Rueckert proposed an automatic segmentation method for segmentation of heart muscle and left and right ventricles (LV and RV) using 4D MR images [24]. In this method, first a cardiac atlas is obtained from 14 normal subjects, and then using registration techniques, the cardiac atlas is aligned to a subject specific atlas.

In our study, we performed segmentation of the heart as an organ in stead of trying to obtain-intra organ structures like LV, RV, or heart muscle separately.

1.3. The Image Dataset

Medical images like CT or MRI normally have 12-16 bits information in gray scale. In CT images, anatomical tissues may occupy significantly different dynamic ranges on display due to difference of X-ray attenuation.

The data used in this thesis work is obtained with CT medical imaging modality. The data set is composed of 45 CT images of upper torso in DICOM format, with properties given Table-1.1.

One of the images from the data set, in original form, is in Figure-1.1. It is obvious that the contrast between organs or tissues to be segmented is low. Due to the limitation of the human visual perception, a contrast enhancement is usually needed for more significant contrast information. Chest CT contains three significant anatomical tissues: bone, lung and soft tissue (heart, fat and muscle tissue). Lung, bone and soft tissue enhanced images are given in Figure-1.2, Figure-1.3 and Figure-1.4 respectively.

Table 1. 1: Properties of the image data set

Photometric interpretation	Grayscale
Size	512x512
X Voxel Width	0.75 mm
Y Voxel Width	0.75 mm
Slice Thickness	5 mm
Bits Allocated	12 Bit

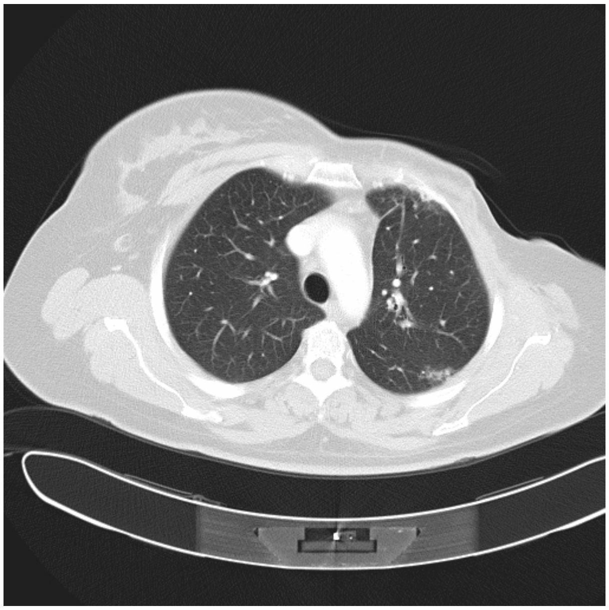


Figure-1.1: Original image

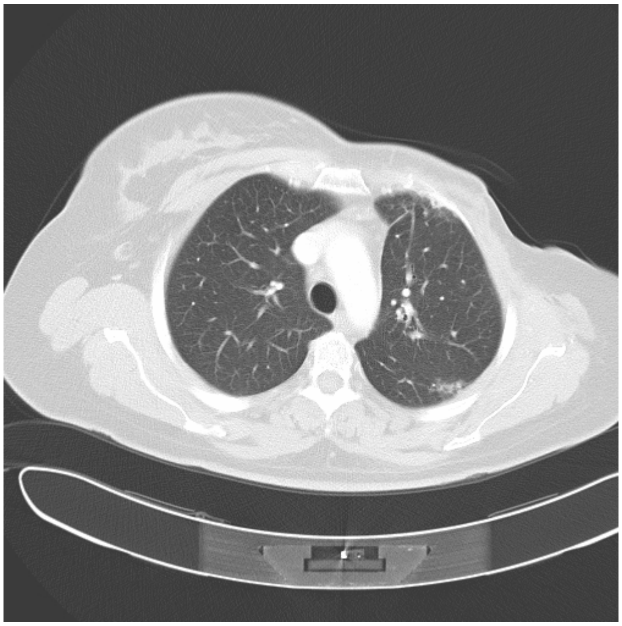


Figure-1.2: Lung enhanced image

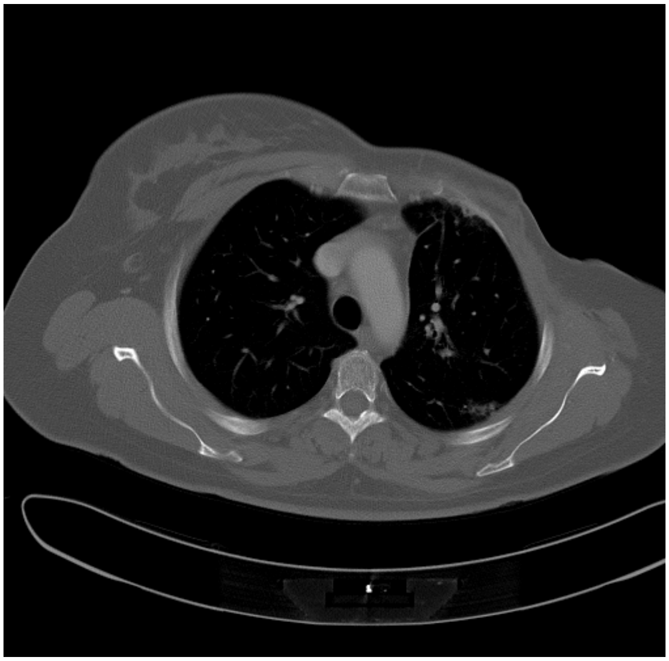


Figure-1.3: Bone enhanced image

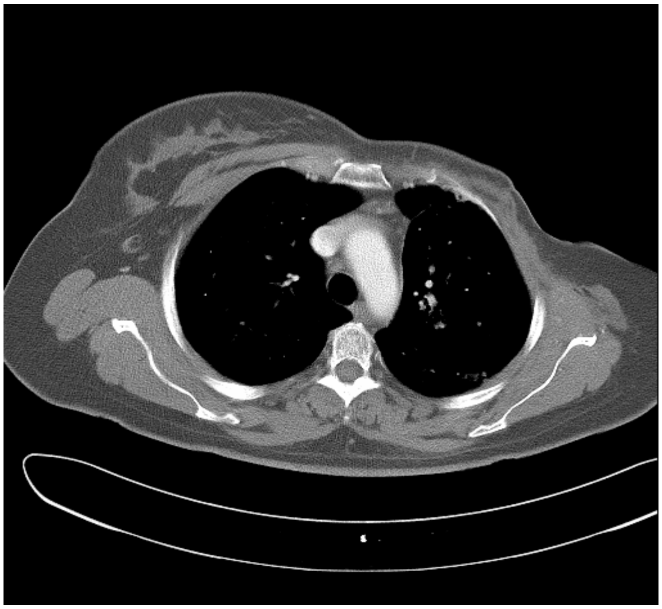


Figure-1.4: Soft tissue enhanced image

The gray level of original images is in the range of (0, 4096), and the histogram of the image is given in Figure-1.5.

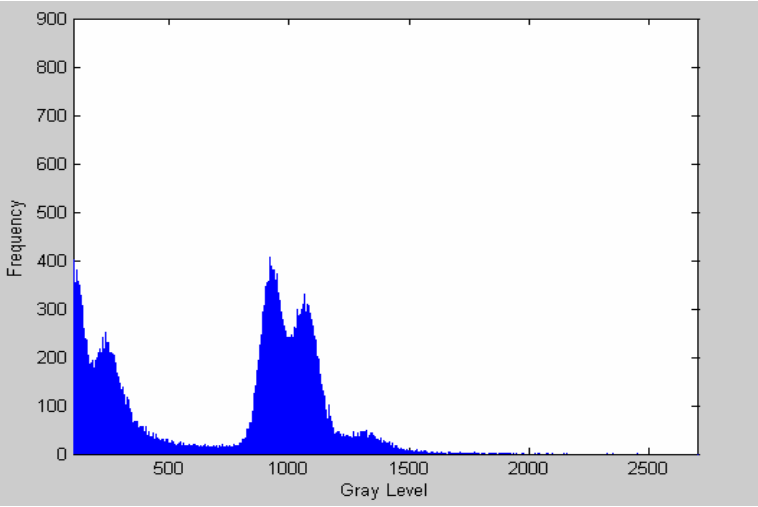


Figure-1.5: Histogram of the original image

The contrast enhancement for the original image can be achieved by using any medical image viewing software. Images in Figure-1.2 to Figure-1.4 are obtained by predefined window center and window width values [25].

It is known that human visual system can perceive less than 100 different gray levels [26], and therefore there is no difference between 7-bit or 12-bit coded images for human eye. However, this is not the case in image processing, and contrast enhancement causes loss of data, therefore we studied on original DICOM images without contrast enhancement.

1.4. The Base Image

In order to quantify the performance of a segmentation method, validation experiments are necessary [27]. The most straightforward approach to validation is by comparing the result of segmentation with a manually obtained segmentation result. This approach does not guarantee a perfect truth model since it completely depends on the performance of the operator. Nevertheless, in order to quantify the performance of segmentation methods, we formed a base image by manual segmentation (see Figure-1.6). Base image is obtained by manual segmentation using contrast enhanced CT images given in Figure-1.2 to Figure-1.4. In the base image, ideal segmentation masks of every tissue are represented by corresponding mean gray level values. Tissues or organs to be segmented are shown in Figure-1.6. Numerical properties of the base image are given in Table-1.2.

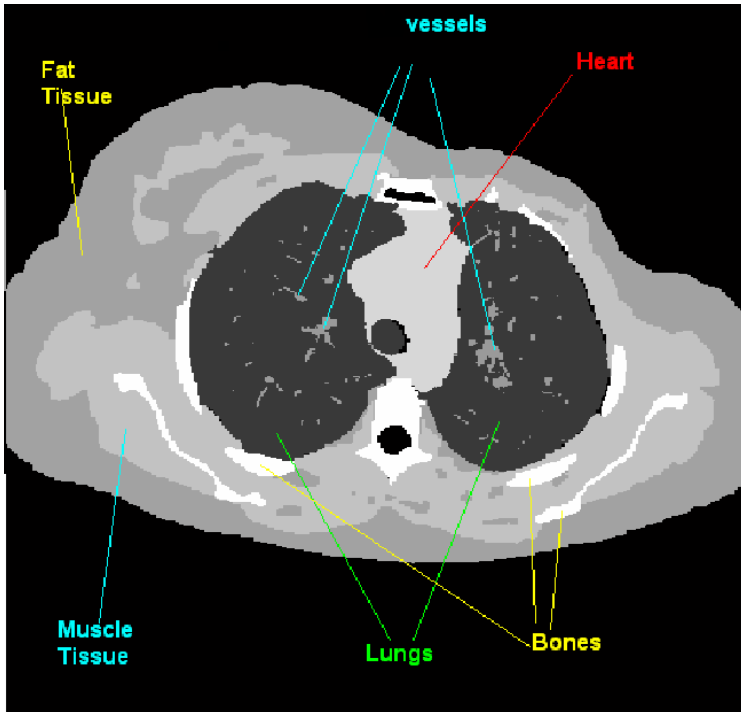


Figure-1.6: Base image with explanatory labels

Table 1. 2: Numerical properties of the base image

Tissues/Organs	**Number of pixels**	**Mean value (in Gray-Level)**	**Variance (in Gray-Level)**
Heart	4962	1175.6	10145
Bones	5915	1394.2	7659
Muscle	31559	1064.2	1951
Fat	37975	886.09	3059
Lungs	30236	272.08	3452.4

1.5. Scope of the Thesis

This thesis study aims comparative analysis of fundamental medical image segmentation methods for segmentation of main tissues and organs in CT images of torso, as well as proposing some improvements for increasing accuracy of some methods. Some of the fundamental medical image segmentation methods, namely thresholding, watershed transformation, region growing, deformable models, and hybrid methods are implemented and applied to 2D CT images of torso for segmentation of heart, lungs, bones (skeleton), muscle and fat tissues. The hybrid method, watershed transformation and region merging (WTRM) is implemented and improved by integrating with K-means clustering for faster and more accurate results. Results of segmentation obtained by each method is compared and evaluated for obtaining the most accurate and efficient segmentation method among the implemented methods for each tissue and organ in the torso. All of the methods are implemented in 2D. Segmentation of heart and lungs are performed for obtaining organ boundaries, segmentation of inter-organ structure is not included.

1.6. Organization of the Thesis

This thesis consists of five chapters. In Chapter 2, some of the major image segmentation algorithms are examined and experimental results on some test images and the base image are given. In Chapter 3, hybrid image segmentation method WTRM is explained. In addition, this chapter explains how the efficiency of WTRM is improved by integrating with K-means clustering. In Chapter 4, comparison among the segmentation methods given in Chapter 2 and Chapter 3 for segmentation of each tissue and organ is given. In addition, for visual inspection of the segmentation results, 3D images obtained using the results of segmentation of each tissue and organ are shown in this chapter. Finally, in Chapter 5 some concluding remarks are stated on the performance of segmentation algorithms and future work is given. Anisotropic diffusion filtering is explained in Appendix A. Implementation of the algorithms and discretization of deformable models are explained in appendices B and C respectively.

CHAPTER 2

MEDICAL IMAGE SEGMENTATION METHODS

In this chapter, some of the fundamental medical image segmentation methods are examined, and applied to the CT images for segmentation of main tissues and organs in the torso. Implemented image segmentation methods are thresholding, region growing, watershed transformation and deformable models. Theory and detailed information about implementation of these methods are given in sub-sections of this chapter.

We also evaluated segmentation methods using qualitative and quantitative metrics. In qualitative evaluation, we evaluated methods in the degree of user interaction required and practicality for segmentation of related tissue or organ. In quantitative evaluation, we used the basic segmentation evaluation metric, namely **Negative Rate Metric (NR)***,* for obtaining how much the segmentation result differs from the ideal segmentation mask or the base image [28]. Negative rate metric evaluates a false negative rate *(NR_{fn})* and false positive rate *(NR_{fp})* as shown in equations 2.1 to 2.3. This metric is based on a pixel-wise mismatches between the base image given in Figure-1.6 and the segmented image.

$$NR = \frac{1}{2}(NR_{fn} + NR_{fp}) \qquad \textbf{(2. 1)}$$

where

$$NR_{fn} = \frac{N_{fn}}{N_{tp} + N_{fn}} \qquad \textbf{(2.2)}$$

$$NR_{fp} = \frac{N_{fp}}{N_{fp} + N_{tn}} \qquad \textbf{(2.3)}$$

N_{fp} *(false positive)* : Number of incorrectly detected pixels

N_{fn} *(false negative)* : Number of undetected pixels that belong to object of interest

N_{tp} *(true positive)* : Number of correctly detected pixels that belong to object of interest

N_{tn} *(true negative)* : Number of correctly rejected pixels

For the negative rate metric, the lower the score the better the algorithm is at correctly segmenting the object of interest that matches the ideal segmentation mask in the base image. In the ideal case, NR is equal to zero.

As a final evaluation criterion, , we have also taken into account the elapsed time in the performance evaluation.

The algorithms are implemented on MATLAB 6.5 and run on a Pentium-III 550 MHz computer with 256 MB DDR RAM.

2.1. Thresholding

Thresholding is a pixel or intensity-based segmentation method. It is one of the oldest, simple and popular techniques for image segmentation. A bi-level thresholding procedure for gray-level images determines an intensity value, called *threshold* that separates pixels belonging to foreground from background. In the case of multi-level thresholding, pixels of the image are classified in more than two classes with more than two threshold values T_1,T_2,... depending on the number of classes desired. Thresholding is used in many applications ranging from document image analysis, map processing, thermal images, CT images, MR images, to ultrasonic images and spatio-temporal video images.

The resultant image after thresholding process is a binary image, in which one of the states indicates the foreground objects and the complementary state indicates background. Thresholding can be done based on global information (e.g. gray level histogram of the entire image) or it can be done using local information (e.g. co-occurrence matrix) [29]. If only one threshold is used for the entire gray-level image, it is called global thresholding. On the other hand, if the image is divided into sub-regions and a gray-level threshold value is obtained for each region, it is called local or adaptive thresholding [28]. In our study we used global thresholding.

Thresholding techniques can be classified in six different categories according to information they exploit [29]. These categories are;

1. *Histogram shape-based methods*: This category of methods achieves thresholding based on the shape properties of the histogram. The shape properties come into play in different forms like the peaks, valleys and curvatures of the histogram.
2. *Clustering-Based Thresholding Methods*: In this class of algorithms, the gray-level data undergoes a clustering analysis with the number of clusters being set always to two, and the gray-level samples are clustered into two parts as background and foreground object.
3. *Entropy-Based Thresholding Methods*: This class of algorithms exploits the entropy of the distribution of the gray levels in a scene that use the entropy of the foreground and background regions, the cross-entropy between the original and binarized image.
4. *Thresholding Based on Attribute Similarity*: These algorithms select the threshold value based on some attribute quality or similarity measure

between the original image and the binarized version of the image such as edge matching, compactness, gray-level moments, connectivity, texture, or stability of segmented objects.

5. *Spatial Thresholding Methods*: This class of algorithms utilizes not only gray value distribution but also dependency of pixels in a neighborhood in the form of context probabilities, correlation functions, co-occurrence probabilities, local linear dependence models of pixels, 2-D entropy and etc.
6. *Locally Adaptive Thresholding*: In this class of algorithms, a threshold is calculated at each pixel, which depends on some local statistics like range, variance, or surface-fitting parameters of the pixel neighborhood.

For more information on various thresholding techniques under these categories, we refer the reader to [29].

In our study we used a multi-level thresholding method, which is an extended version of Otsu's method [30]. Otsu's method is a type of clustering-based thresholding method, and mostly this method is preferred in medical image segmentation studies [31-32]. Before proceeding to multi-level thresholding method, Otsu's method for bi-level thresholding is given in the following section.

2.1.1. OTSU's Method For Image Thresholding

This thresholding method chooses a threshold value that maximizes the *between-class variances*, essentially by this way also minimizes *within-class variances*.

Let us assume that a gray level image has N pixels, and the gray level value ranges from 1 to L. The number of pixels with gray level *i* ($1 \le i \le L$) is given with f_i and the probability of having gray level *i* in image is given as:

$$P_i = \frac{f_i}{N} \qquad \textbf{(2.1.1)}$$

After bi-level thresholding, the pixels are divided into two classes, and the obtained classes are:

C$_1$: set of pixels with gray level value i, $1 \le i \le T$, where T is the threshold value.
C$_2$: set of pixels with gray level value i, $T+1 \le i \le L$,

The corresponding probability distribution functions for classes C$_1$ and C$_2$ are

C$_1$: $\frac{P_1}{\omega_1}, \ldots\ldots \frac{P_T}{\omega_1}$ and,

C$_2$:. $\frac{P_{T+1}}{\omega_2}, \ldots\ldots \frac{P_L}{\omega_2}$,

where ω_1 and ω_2 are zeroth-order cumulative moments of C$_1$ and C$_2$ respectively, and given as,

$$\omega_1 = \sum_{i=1}^{T} p_i \qquad \textbf{(2.1.2)}$$

$$\omega_2 = \sum_{i=T+1}^{L} p_i \qquad \textbf{(2.1. 3)}$$

the mean value of classes C$_1$ and C$_2$ are given as,

$$\mu_1 = \sum_{i=1}^{T} \frac{ip_i}{\omega_1} \qquad \textbf{(2.1. 4)}$$

$$\mu_2 = \sum_{i=t+1}^{L} \frac{ip_i}{\omega_2} \qquad \textbf{(2.1. 5)}$$

and the mean value of the whole image is given as,

$$\mu_T = \omega_1 \mu_1 + \omega_2 \mu_2 \qquad \textbf{(2.1. 6)}$$

Otsu defined the between-class variance of the thresholded image as [33],

$$\sigma_B^2 = \omega_1 (\mu_1 - \mu_T)^2 + \omega_2 (\mu_2 - \mu_T)^2 \qquad \textbf{(2.1. 7)}$$

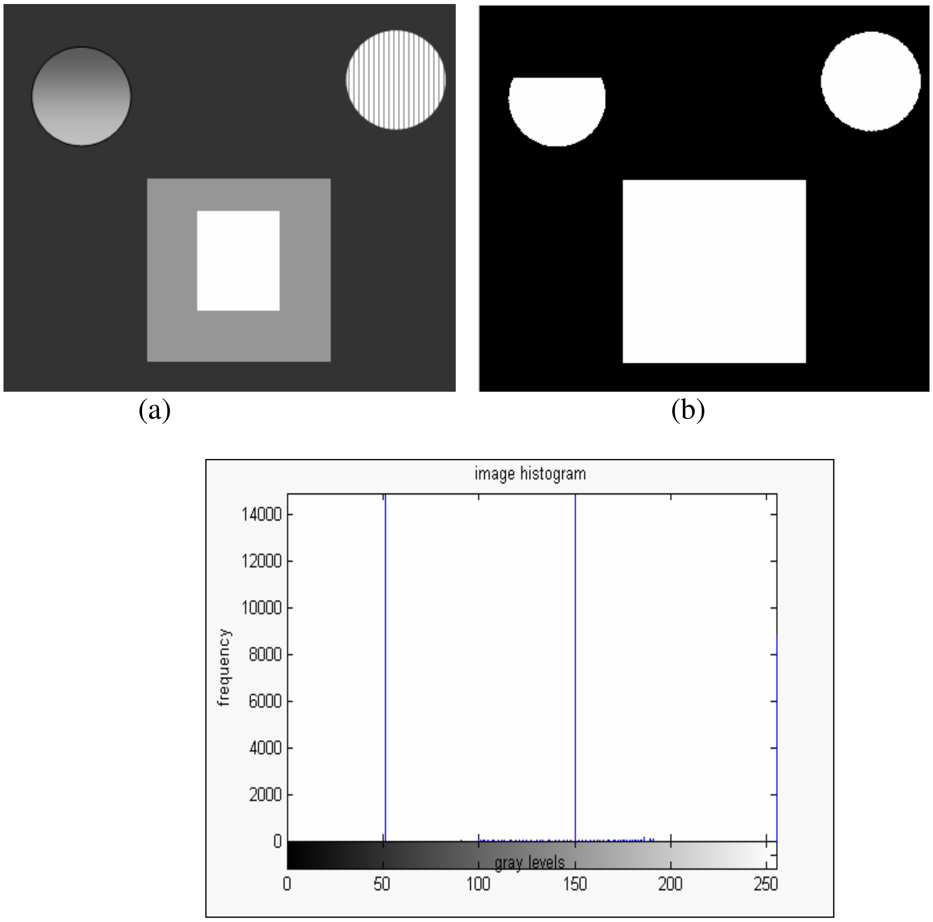
for bi-level thresholding, Otsu verified that the optimal threshold value T is chosen such that the between-class variance σ_B^2 is maximized. That is, we implement (2.1.7) for every gray level value, and chose the T value that maximizes σ_B^2:

$$\widehat{T} = \operatorname{argmax}\{\sigma_{\mathrm{B}}^2\} \qquad \textbf{(2.1. 8)}$$

We applied Otsu's thresholding method to a test image (Figure-2.1.1(a)) that contains an unevenly illuminated object. The gray-level range of pixels of test image is between (0-255), which can be seen on the image histogram Figure-2.1.1(c). The result of segmentation is in Figure-2.1.1(b). The obtained threshold value is $\widehat{T}$ = 117, pixels with gray level value below $\widehat{T}$ are classified as background and others as foreground. We can say that, because this method only takes into account the gray-level values of pixels and does not depends on some local statistics or neighborhood, it failed to detect the uneven illumination, and classified some part of the unevenly illuminated object as background.

To investigate the weaknesses of the thresholding method in the presence of noise, we applied thresholding to the same test image but with zero-mean Gaussian noise SNR = 39dB. We can see that the histogram of the noisy image (Figure-2.1.2(c)) and the result of thresholding (Figure-2.1.2(b)) has changed significantly. The estimated threshold value $\widehat{T}$ is 116. It is obvious that some pixels are wrongly classified, and the result of thresholding is unsatisfactory.

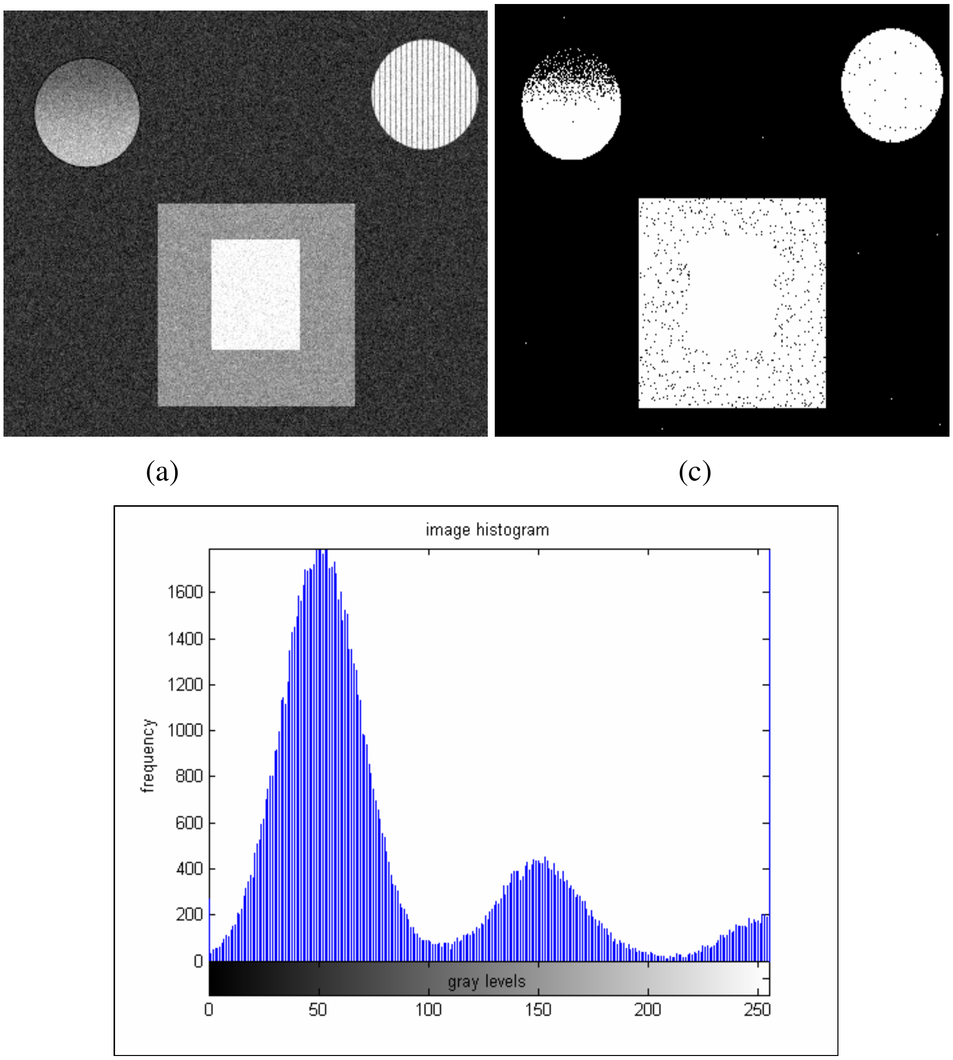
We also applied thresholding to the CT image (12-bit and gray-level range is (0-4095)) shown in Figure-2.1.3(a). The estimated threshold value is $\widehat{T}$ =574. From the resultant image, we can see that pixels that belong to muscle, fat, vessels, heart and bones are classified as foreground.



(a) (b)

(c)

Figure 2.1.1: Result of thresholding,(a) original test image, (b) result of thresholding, (c) histogram of the test image



(a) (c)

(c)

Figure 2.1.2: Result of thresholding for noisy image, (a) noisy test image, (b) result of thresholding, (c) histogram of the noisy image.

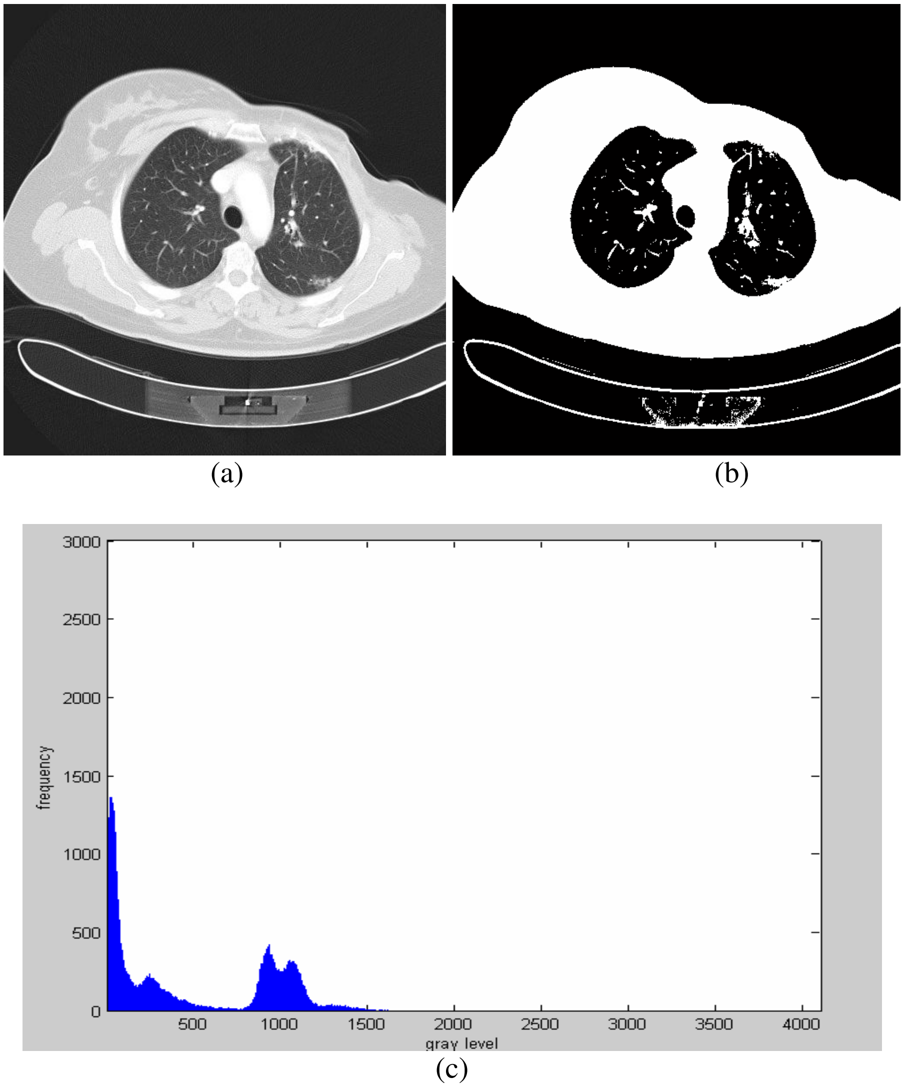

(a)
(b)
(c)

Figure 2.1 3: Result of thresholding for the CT image, (a) Original image, (b) Result of thresholding, (c) histogram of the original image.

2.1.2.Multi-Level Thresholding

We used an extended version of the Otsu's method for multi-level thresholding [33]. In multi-level thresholding, previous formulation is extended by assuming that we have *M* classes instead of two, and therefore we should have *M-1* threshold values (T_1, T_2, $T_{3.....}$ T_{M-1}). The pixels of the image are divided in to *M* different classes :

C$_1$: set of pixels with gray level value i, $1 \le i \le T_1$, where T_1 is the first threshold value.

C$_2$: set of pixels with gray level value i, $T_1 + 1 \le i \le T_2$,

$\vdots$

C$_M$: set of pixels with gray level value i, $T_{M-1} + 1 \le i \le L$,

The corresponding probability distribution functions for each class are computed similarly. Then the zeroth-order cumulative moments are :

$$\omega_k = \sum_{i \in C_k} p_i \quad \mathbf{,}\, k{=}1{,}2{,}....m \tag{2.1.9}$$

and the first-order cumulative moment of the kth class is computed as:

$$\mu_k = \sum_{i \in C_k} i p_i \,/\, \omega_k \ . \tag{2.1.10}$$

The mean intensity of the whole image is then:

$$\mu_T = \sum_{i=1}^{M} \mu_i \omega_i \qquad . \tag{2.1.11}$$

Finally, using equations 2.1.9 - 2.9.11, we can write the between-class variance as:

$$\sigma_B^2 = \sum_{i=1}^{M} (\mu_i - \mu_T)^2 \omega_i \qquad . \tag{2.1.12}$$

optimum threshold values ($\widehat{T}_1 ,\widehat{T}_2 ,\widehat{T}_3 ....\widehat{T}_{M-1}$) are chosen by maximizing σ_B^2 :

$$\left\{\widehat{T}_1 ,\widehat{T}_2 ,\widehat{T}_3 ....\widehat{T}_{M-1}\right\} = \arg\max\left(\sigma_B^2 (T_1 ,T_2 ,T_3 ....T_{M-1})\right) \tag{2.1.13}$$

We applied multi-level thresholding method to the same test image shown in Figure-2.1.1(a), and we set the number of classes to *M=4*. The obtained threshold values are $\widehat{T}_1$ *=88,* $\widehat{T}_2$ *=127,* $\widehat{T}_3$ *=189.* Pixels with gray level below or equal to $\widehat{T}_1$ are classified as C$_1$, pixels with gray level value between $\widehat{T}_1$ and $\widehat{T}_2$ are classified as C$_2$, pixels with gray level value between $\widehat{T}_2$ and $\widehat{T}_3$ are classified as C_3, and pixels with gray level value above T_3 are classified as C_4. We can see from Figure-2.1.4 that the upper part of the unevenly illuminated circular object in the upper left corner is classified in C_2, where it was classified as background in the previous case. Obtained classes can be seen in Figure-2.1.5.

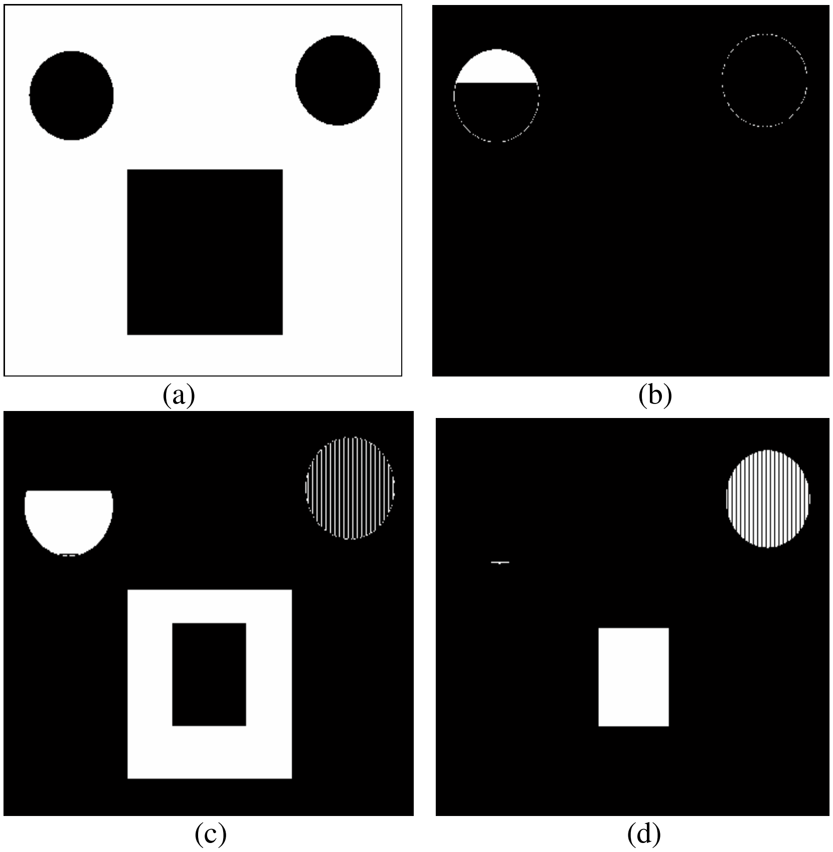


Figure 2.1 4: Obtained classes after multi-level thresholding (pixels that belong to a class are shown in white) , (a) Class-1 (C_1) background, (b) Class-2 (C_2), (c) Class-3 (C_3), (d) Class-4 (C_4).

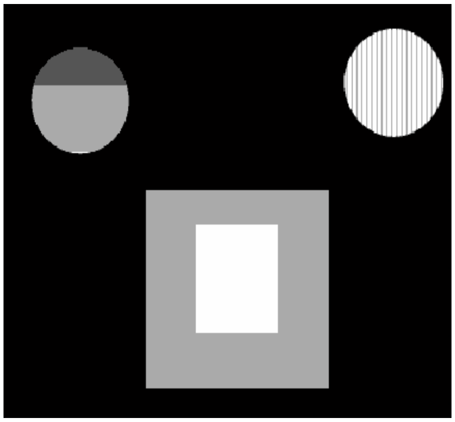


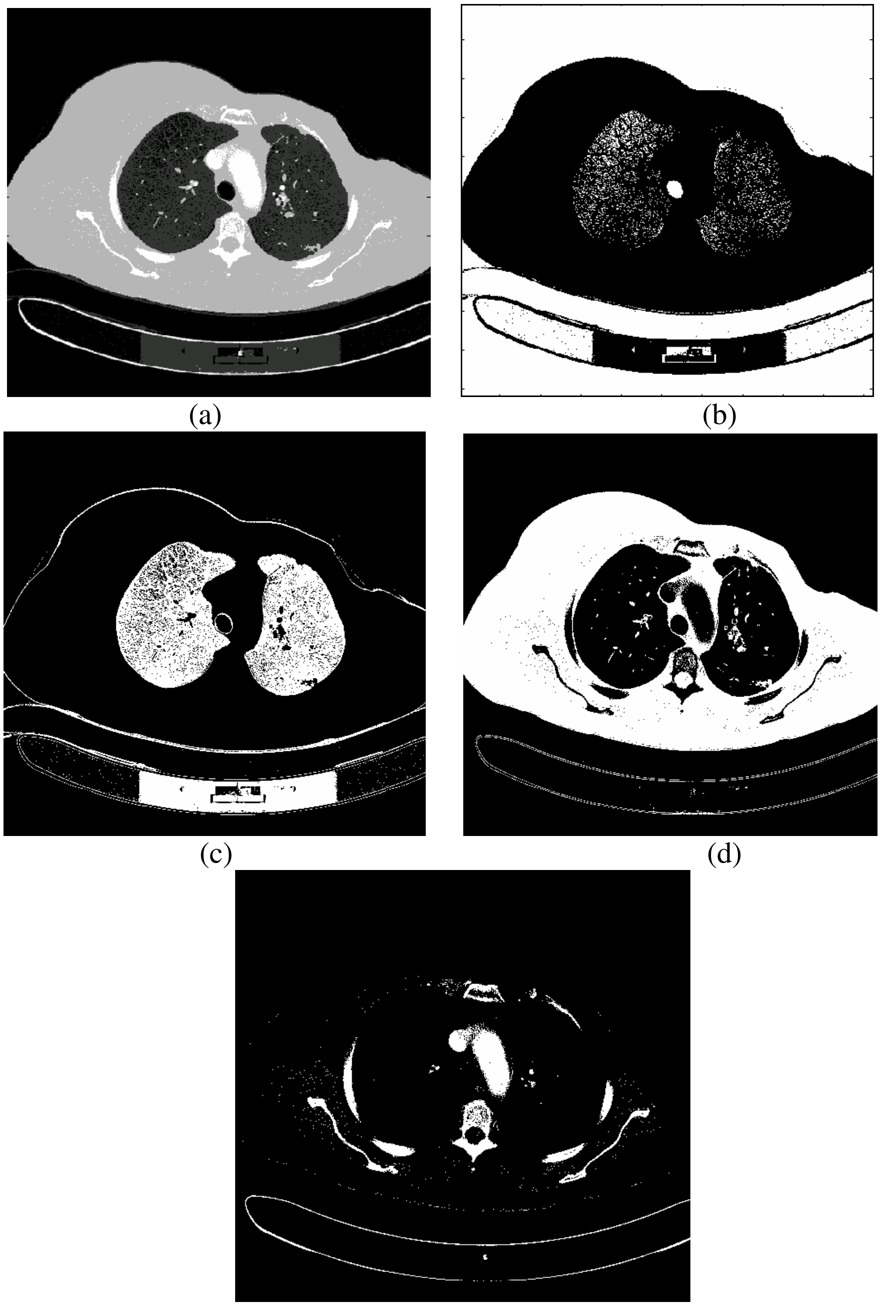
Figure 2.1.5*:* Result of multi-level thresholding

Result of multi-level thresholding for the CT image in Figure-2.1.3(a) is given in Figure-2.1.6. The number of classes is chosen as 4. We can state that we cannot perform segmentation of heart, muscle and fat tissues separately when we use three different threshold values, because pixels that belong to these organs & tissues are classified in the Class-3 image (Figure-2.1.6 (d)). On the other hand, we can see that Class-4 image consists of pixels that belong to bones, heart and some scattered pixels belong to muscle tissue. We can perform segmentation of bones by removing scattered pixels with median filtering, and manually selecting connected regions that belong to bones. Similarly , by using the Class-2 image given in Figure-2.4.6 (c), we can perform segmentation of lungs. Results obtained for segmentation of bones and lungs are given in Table-2.1.1, Figure-2.1.7 and Figure-2.1.8. We can state that multi-level thresholding produces accurate results for segmentation of bones with a very low NR value, and produces acceptable results for segmentation of lungs. However, because every possible combination of threshold values are analyzed to obtain the optimum, the elapsed time is very large for three threshold values.

Since the bit depth in DICOM images is generally 12 Bit or 16 Bit, and the gray level value range is between (0,4095) in our dataset, and therefore applying multi-level thresholding requires considerable amount of time. In the case of three threshold values, it takes about 5 minutes to obtain threshold values. However, for 4 threshold values it takes more than 4 hours for one image where 4 nested *'for-loops'* are iterated from 1 to 4095 to obtain the maximum of the class variance values. Therefore, multi-level thresholding with more than 3 threshold values was not taken into account. Moreover, since this thresholding technique does not take into account the spatial characteristics of the image, resultant images may contain scattered and disconnected pixels. Furthermore, due to lack of spatial information, this method is very sensitive to noise and intensity inhomogenities.

Table 2.1.1: Numerical Results for segmentation using multi-level thresholding with three threshold values

Segmented Tissue/Organ	Negative Rate Metric (NR)	Elapsed Time (sec)
Bones	0.0077	253
Lungs	0.1218	



(a)

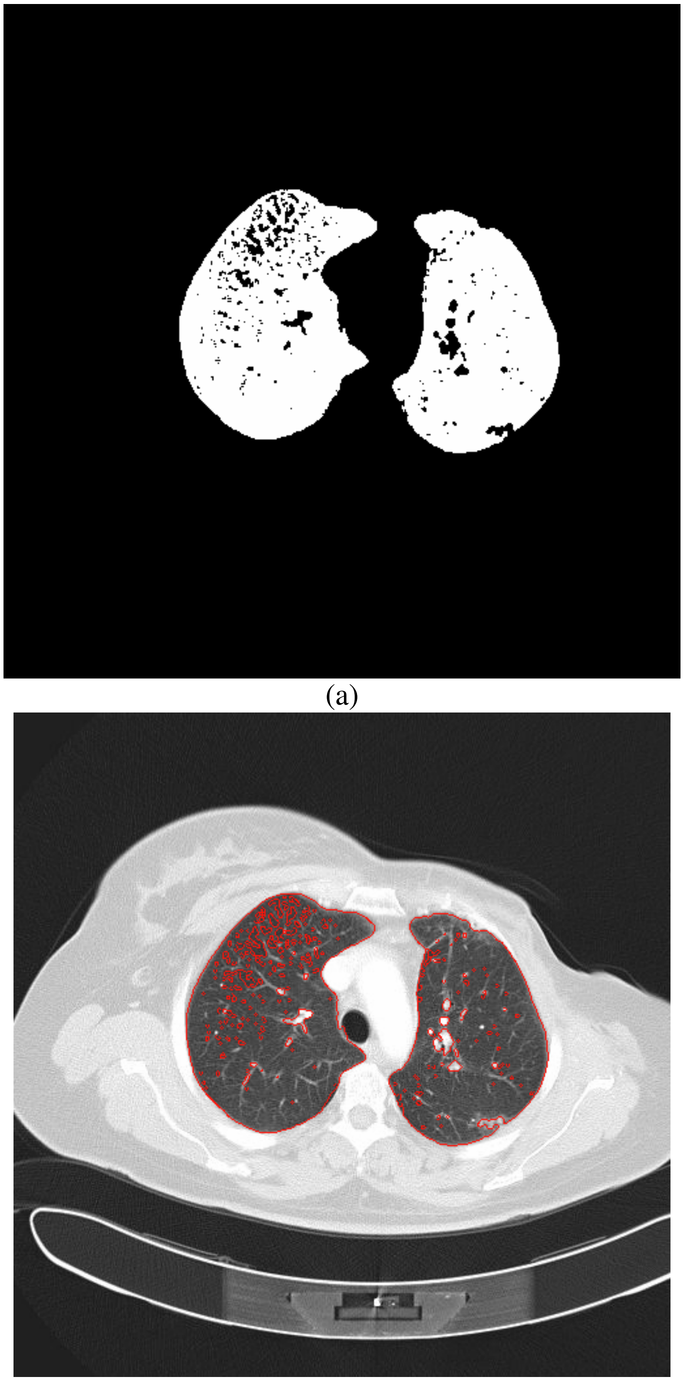
(b)

(c)

(d)

(e)

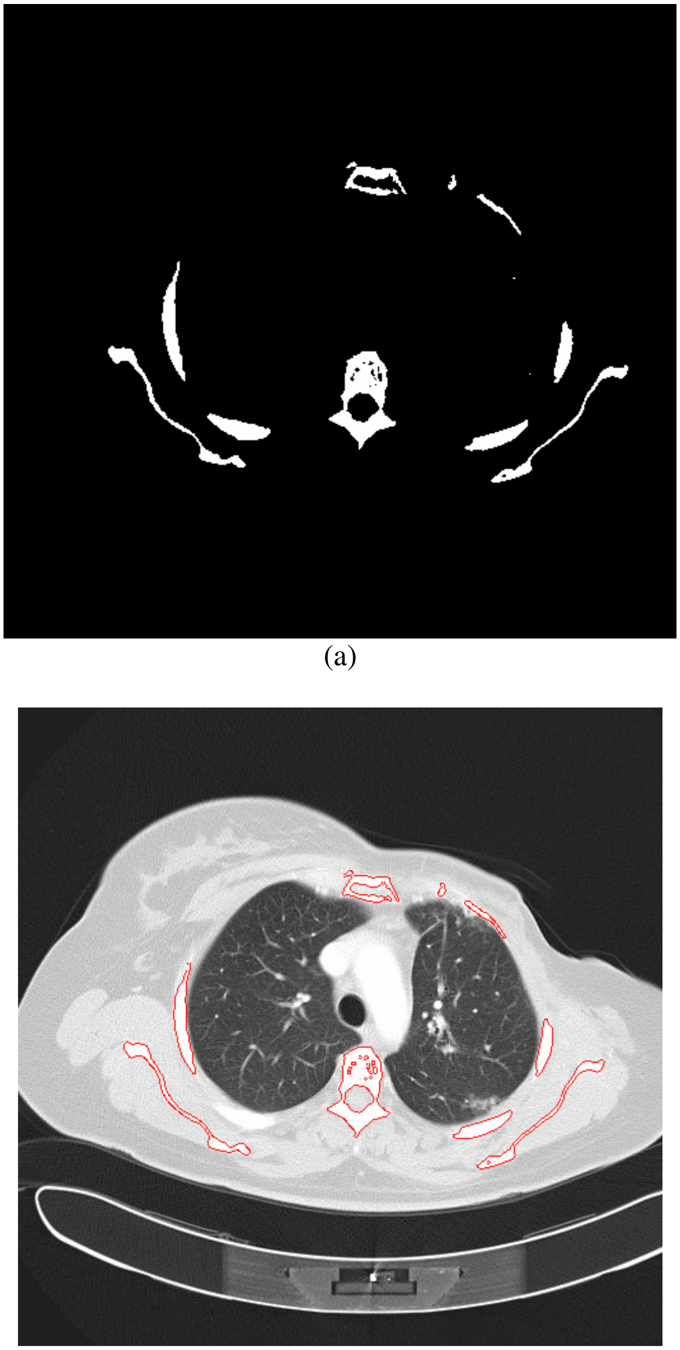
Figure 2.1.6: Result of multi-level thresholding, (a) thresholded image, (b) Class-1, (c) Class-2, (d)Class-3, (e)Class-4 (pixels that belong to a class are shown in white).



(a)

(b)

Figure 2.1.7: Result of segmentation for lungs, (a) resultant binary image, (b) original image overlapped with borders of lungs



(a)

(b)

Figure 2.1.8: Result of segmentation for bones (a) resultant binary image, (b) original image overlapped with borders of bones.

2.2. Region Growing

Region Growing is a region-based segmentation method which can be used for extracting a connected region of the image based on some predefined criteria. These criteria can be based on intensity information and/or edges in the image. Region growing algorithms take seeds, which can be individual points or connected components, to mark each of the regions to be segmented. Seeds can be manually selected by an operator, or automatically selected. Starting from the seed regions, region growing algorithm grows the regions around seed regions based upon a certain homogeneity criterion. If the adjoining pixels are similar to the seed, they are merged within a single region. The process continues until there is no neighboring pixel left to be merged.

In our study we applied an improved version of seeded region growing proposed by Adams and Bischop [1, 3], and integrated this algorithm with edge-detection as given in [34,35].

2.2.1. Seeded Region Growing Integrated with Edge-Detection

Seeded Region Growing (SRG) is initialized by choosing a usually small number of pixels, known as seeds. Each seed region is a connected component and represented by A_i, where i=1,2,...n, and n is the number of seed regions. The set of unallocated pixels, which border at least one of the regions A_i is given by *T,* i.e.

$$T = \left\{ x \notin \bigcup_{i=1}^{n} A_i \mid N(x) \bigcap \bigcup_{i=1}^{n} A_i \neq 0 \right\}$$

where *N(x)* is the a neighboring pixel of the pixel x. In our case we consider 8-connected neighborhood (i.e. for each pixel all its 8 neighbors). At each step in the algorithm, one neighboring pixel from the set *T* is taken and examined. The examination is carried out between the neighboring pixel *N(x)*, and the region it intersects $A_{i,}$ using a dissimilarity test. The dissimilarity test is applied by using a dissimilarity function, $\delta(x)$, and the edge information. Depending on the result of dissimilarity test, *N(x)* is added to the region A_i, by labeling it with the same label of A_i . When *N(x)* intersects with more than one region, then the dissimilarity function $\delta(x)$ is computed for each region, and *N(x)* is taken into the dissimilarity test with the region which produces smallest $\delta(x)$ value [1].

In order to regulate the order of applying dissimilarity test to the neighboring pixels *N(x),* a dynamic queue is formed. This dynamic queue holds the addresses of the pixels *N(x)* in a descending order with respect to the corresponding $\delta(x)$ values. Namely, a neighboring pixel *N(x)* is inserted to the queue with respect to its $\delta(x)$ value. The neighboring pixel *N(x)* with the smallest $\delta(x)$ value is inserted to the top, and only this pixel can be removed from the queue. Therefore, only one of the neighboring pixels *N(x)* is processed just once [3].

2.2.1.1. Dissimilarity Test

The dissimilarity test is used to decide whether a neighboring pixel should be added to the growing region or not, by using the information obtained from edge detection and dissimilarity function.

Dissimilarity test consist of two parts, the dissimilarity function and edge map test.

In the dissimilarity function part we used the dissimilarity function proposed by Adams and Bischop [1], which is also known as mean-luminance difference :

$$\delta(x) = \left|N(x) - \mu(R_i)\right| \qquad \textbf{(2.2.1)}$$

Where *g(x)* is the gray-level value of the neighboring pixel x, and $\mu(R_i)$ is mean value of i^{th} seed region R_i.

In the edge-map part we computed the edge-map of the smoothed version of the given image using canny edge detector. The obtained edge-map is a binary image which gives significant edges.

Using edge detection or dissimilarity function alone in dissimilarity test suffers from following problems;

- Because edge detection does not always result in closed contours, growing the seed region until reaching to an edge pixel can cause leaking out of borders.
- Using the dissimilarity function alone in dissimilarity test can lead false results by growing the region through salient edges.

Therefore we integrated edge detection and dissimilarity test to stop growing at salient image edges using the edge information.

The neighboring pixel *N(x)* is taken from the top of dynamic queue and included into the region R_i it intersects if it passes the following test;

- If neighboring pixel *N(x)* is not an edge pixel, and the value of the dissimilarity function $\delta(x)$ is smaller than or equal to a threshold value (one tenth of the mean value of the region of interest).

  $$\delta(x) \leq 0.1 * \mu(R_i),$$

- Else if neighboring pixel *N(x)* is an edge pixel, then result of dissimilarity function $\delta(x)$ is smaller than or equal to a threshold value (0.05 times the mean value of the region of interest) .

  $$\delta(x) \leq 0.05 * \mu(R_i),$$

then the neighboring pixel N(x) is included in region R_i and labeled with the same label as R_i. After each pixel is added to the region, the mean value of the region is recomputed. The pseudo code of the algorithm is given in Appendix-B.1.

The flowchart of the region growing algorithm is given in Figure-2.2.1. We can see that before applying region growing, a noise reduction process is performed. This is because region growing is very sensitive to noise, which causes false edges and extracted regions that have holes. In noise reduction process, anisotropic diffusion filtering is utilized (see Appendix-A) for smoothing the image, and reducing the effect of noise without loosing the significant edges. The performance and sensitivity of the region growing algorithm to noise are discussed in the next section.

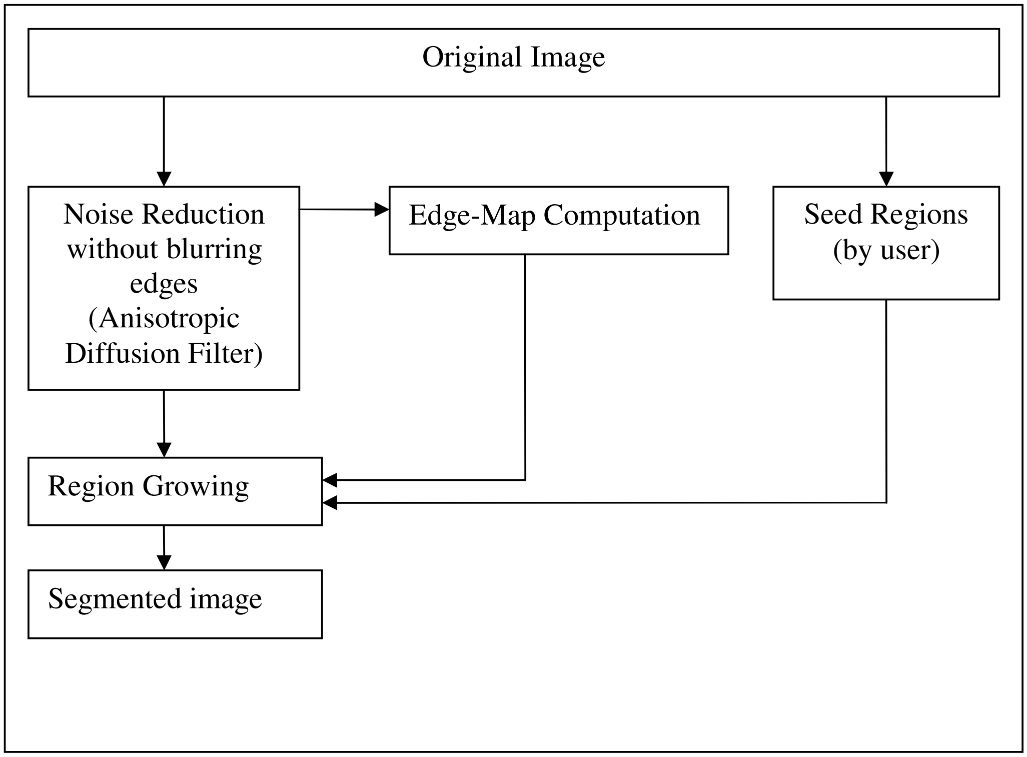
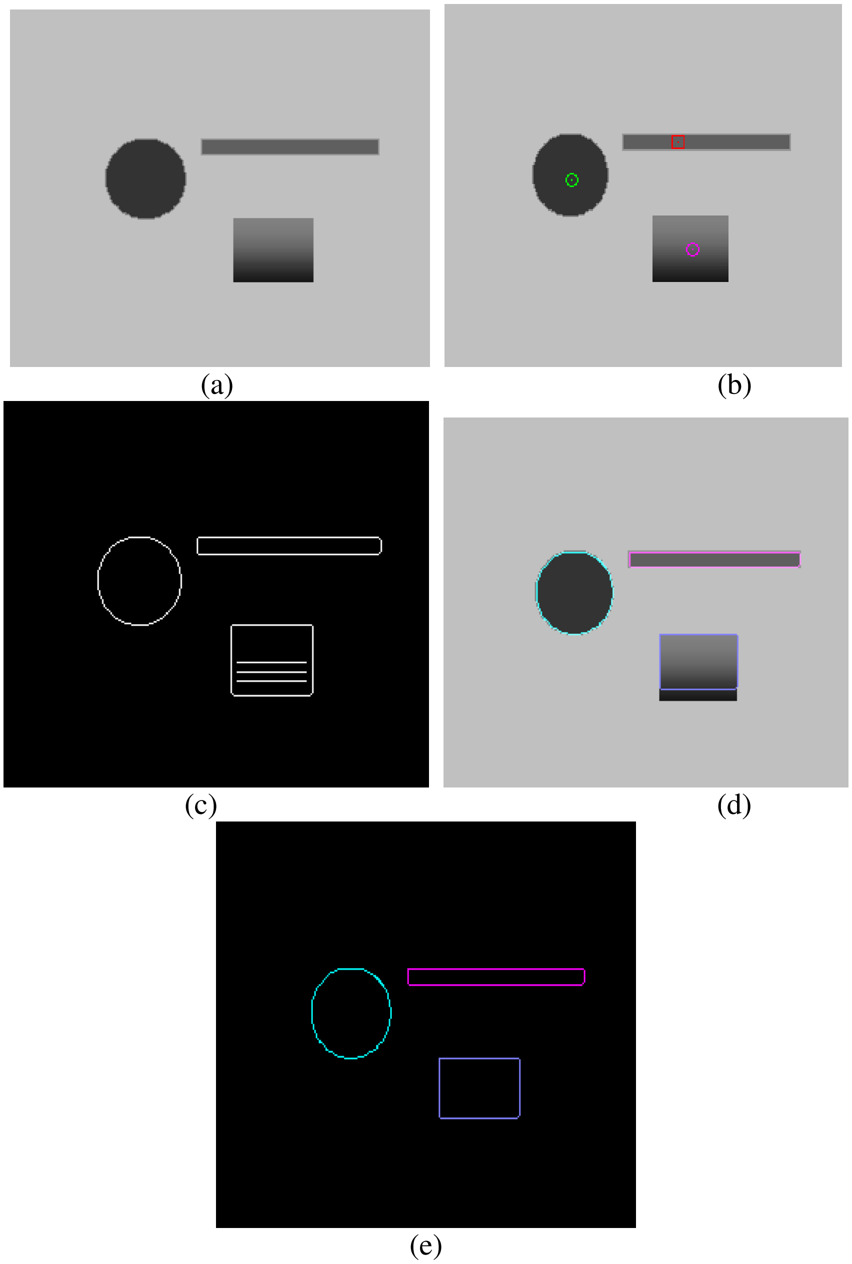


Figure 2.2.1: Stages of Region Growing

2.2.1.2.Experimental Results and Conclusion

We first applied region growing to the test image in Figure-2.2.2(a). The chosen seed regions are shown with different colors in this manuscript in Figure-2.2.2(b). The computed edge map is shown in Figure-2.2.2(c), and the result of region growing can be seen in Figure-2.2.2(d). The borders of segmented objects after region growing are shown in Figure-2.2.2(e). We can see from Figure-2.2.2(d) that the circular and rectangular objects are perfectly segmented using region growing. However, for the unevenly illuminated square object, region growing failed to obtain a satisfactory result.

We also applied region growing to the same test image but with a zero-mean Gaussian noise where the SNR is 43 dB. The noisy image is on Figure-2.2.3(a), and the filtered image with anisotropic diffusion filter is on Figure-2.2.3(b). In order to show the weakness of region growing in the presence of noise, result of segmentation without applying anisotropic filtering is given in Figure-2.2.3 (f). It is obvious that obtained result for segmentation of the test image is unacceptable. However, by using anisotropic diffusion filtering we can enhance the segmentation as shown in Figure-2.3.3 (e). Anisotropic filtering is applied with 30 iterations, diffusion constant K = 30 and λ=0.25.

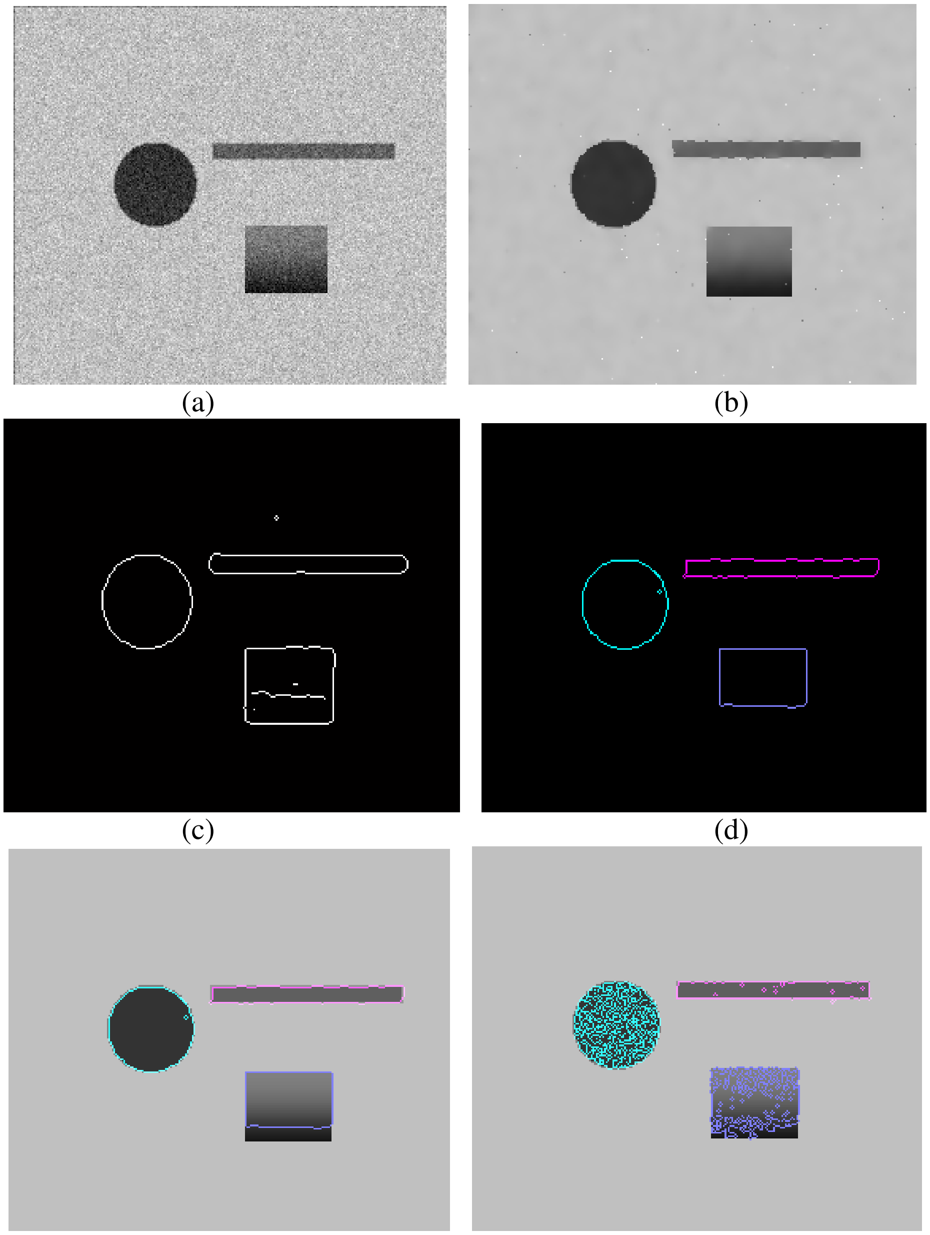


(a) (b)

(c) (d)

(e)

Figure 2.2.2*:* Result of region growing, (a) original image, (b) original image with seeds, (c) obtained edge-map, (d) result of segmentation, (e) obtained borders of segmented regions



(a) (b)

(c) (d)

(e) (f)

Figure 2.2.3: Result of region growing under noise, (a) noisy image, (b) smoothed image, (c) obtained edge-map, (d) obtained borders of segmented regions, (e) result of segmentation, (f) result of segmentation without applying anisotropic diffusion filtering.

In order to assess the performance of region growing algorithm for segmentation of heart, lungs, bones, fat and muscle tissues, we utilized the CT image in Figure-2.2.4(a), and the base image given in Figure-1.6. Results are shown in Figure-2.2.4 to Figure-2.2.8. In Table-2.2.1, numerical results for segmentation of heart, lungs, bones, fat and muscle tissues are given. NR is used to obtain how much the segmentation results differs from the ideal segmentation masks of the base image

We can see that NR values for segmentation of bones and lungs are very low, and elapsed times are acceptable. Therefore we can state that region growing produces very accurate segmentation results for lungs and bones. In addition, NR values for segmentation of fat and muscle tissue are acceptable, and produce visually satisfactory results (see Figure-2.2.4). Therefore, we can state that region growing produces accurate results for segmentation of bones, lungs, muscle, and fat tissues.

Table 2.2. 1: Numerical Results for Region Growing

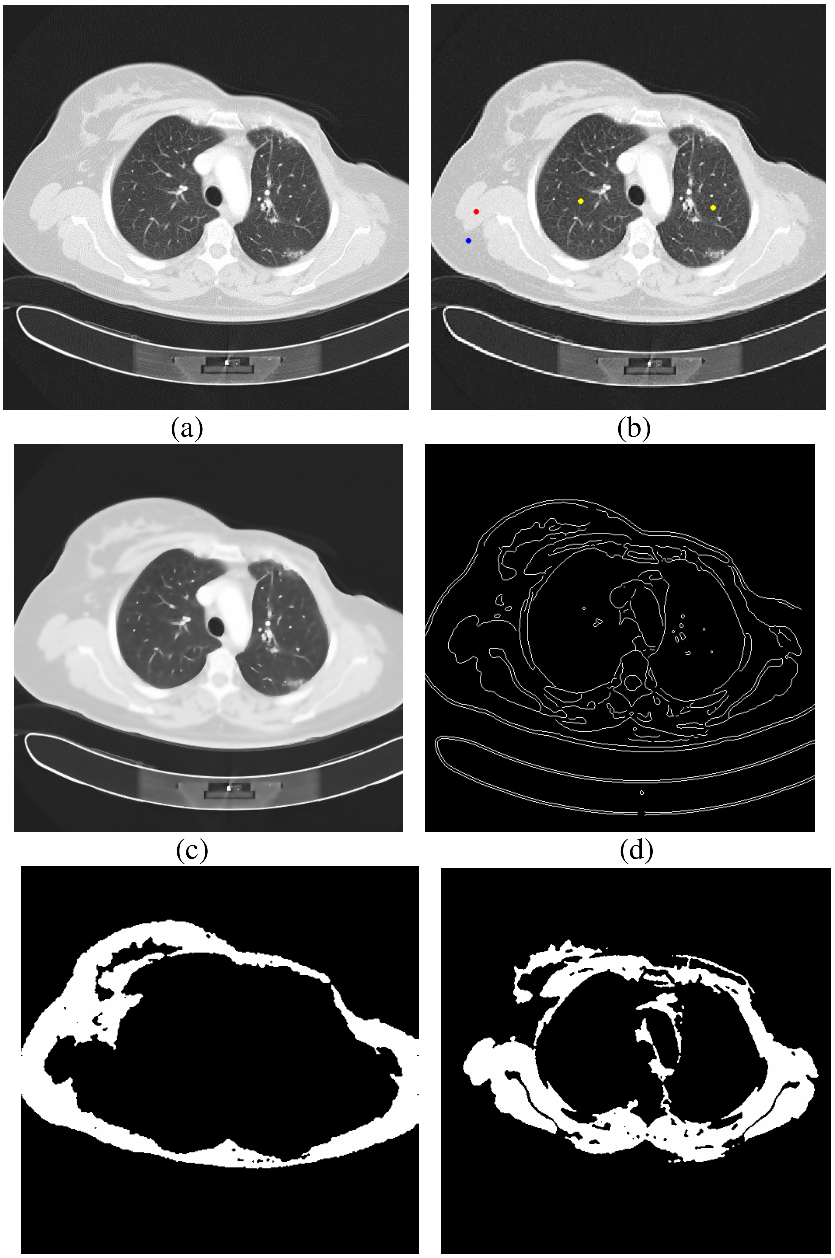
Segmented Tissue/Organ	Negative Rate Metric (NR)	Elapsed Time (sec)
Heart	0.22029	28.4
Bones	0.00037	37.4
Muscle	0.03921	76.8
Fat	0.06504	48.6
Lungs	0.00011	52.4

On the other hand, region growing does not produce acceptable results for segmentation of heart. Moreover, we can not obtain better results for segmentation of heart because of its high variance, and low-contrast between its neighboring tissues (bones and muscle tissues). Increasing the number of seed points or increasing the threshold value over $0.1*\mu(R_i)$ in dissimilarity test leads to even worse results. Therefore, using region growing as a segmentation tool for heart is not a very good choice.

For segmentation of bones, although they have high variance too, we can get rid of this problem by increasing the number of seeds, especially at inhomogeneous parts of bones in gray-level, because the contrast between bones and its neighboring tissues is high. However, despite the very low NR value for segmentation bones, the main drawback is that the user must select at least 10 seed points to obtain a good result, where selecting 2 seed points are enough for an accurate segmentation of lungs, muscle or fat tissues. Moreover, selecting 10 seed points for bones is the case for our base image, and this value can increase if we consider the whole CT image dataset. Therefore, segmentation of bones is not very practical using region growing.

In the software we developed, one can apply region growing algorithm for all tissues with the desired number of seeds at once, but it takes to much time even for one image. In addition to this drawback, its primary disadvantage is that it requires manual interaction to obtain the seed point. Thus, for each region that needs to be extracted, a seed must be planted.

Automatic selection of seed points is also possible for different applications. For example, the brightest pixel infra-red image can serve as a seed pixel, or can also be determined from peaks in an image histogram. However, because we want to segment various tissues with different gray-levels in a CT image of torso, it is not straightforward for us to select seed points automatically. Therefore in our study we preferred selecting seed points manually.

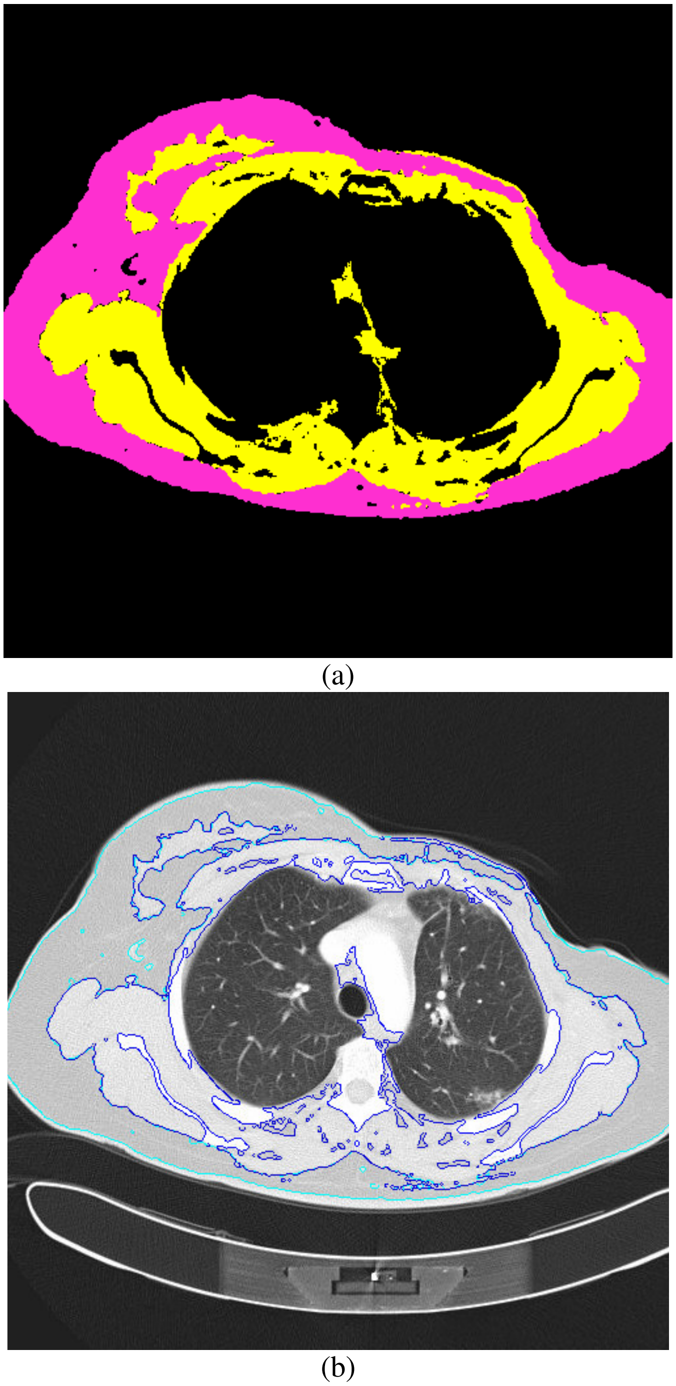


(a) (b)

(c) (d)

(e) (f)

Figure 2.2.4: Result of region growing, (a) Original image, (b) Original image with seeds (for lung, fat and muscle tissues), (c) smoothed image with anisotropic diffusion filtering, (d) obtained edge-map, (e) result of segmentation for fat tissue (extracted fat tissue given in white), (f) result of segmentation for muscle tissue.



(a)

(b)

Figure 2.2.5: Result of segmentation for fat and muscle tissues, (a) resulting regions (pink for fat, yellow for muscle), (b) Original image with the resulting contours

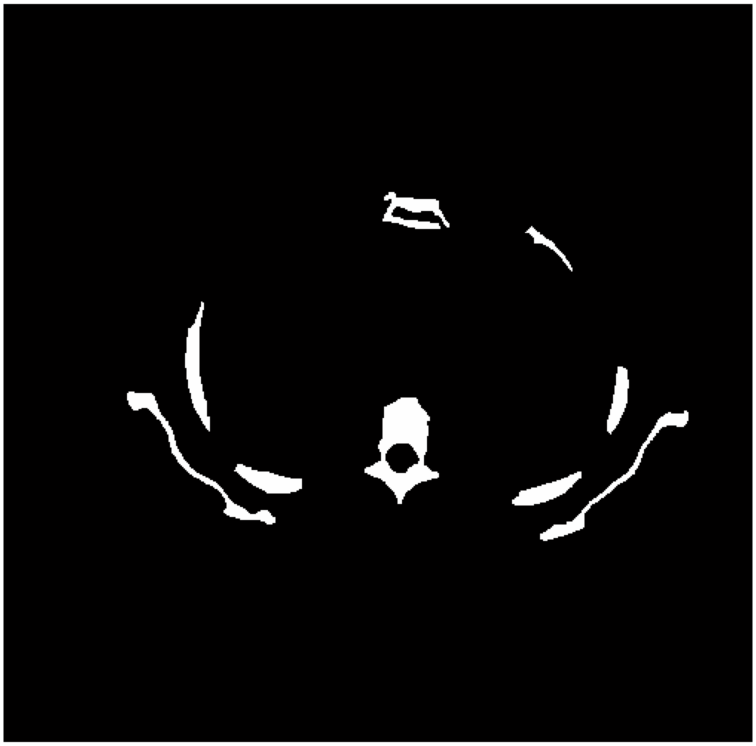
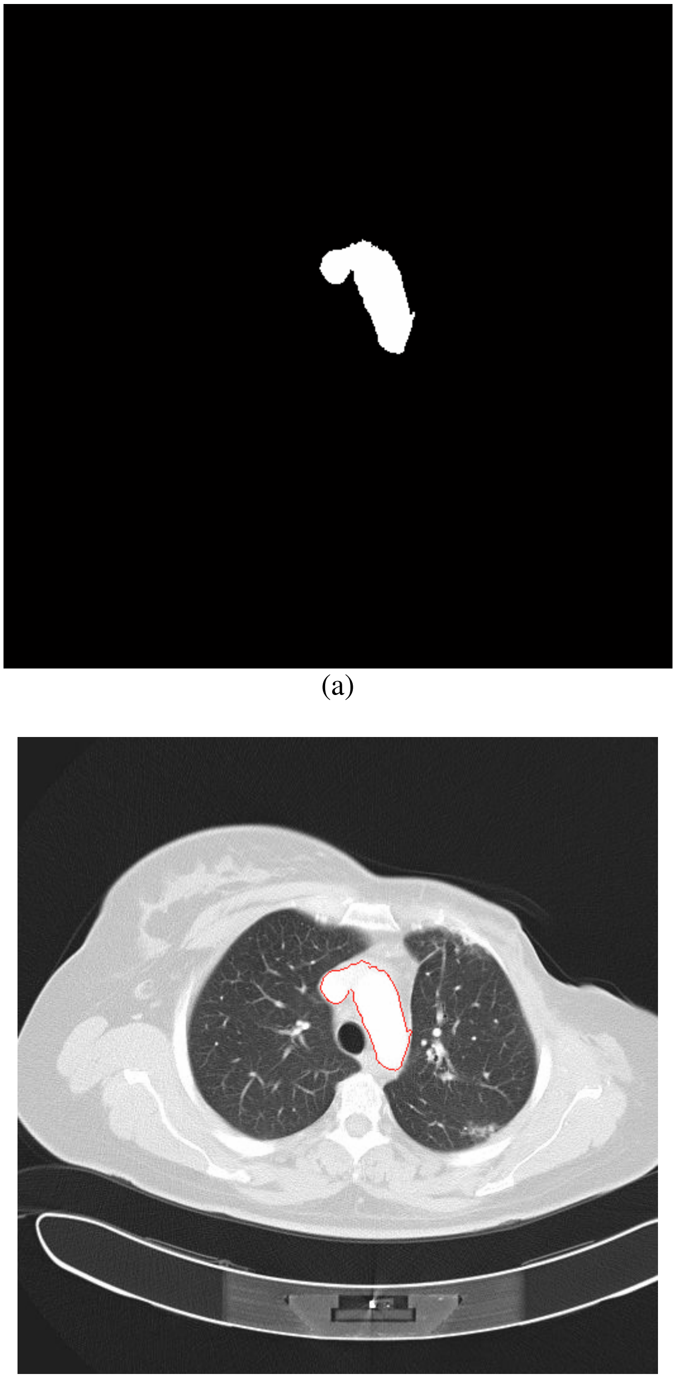


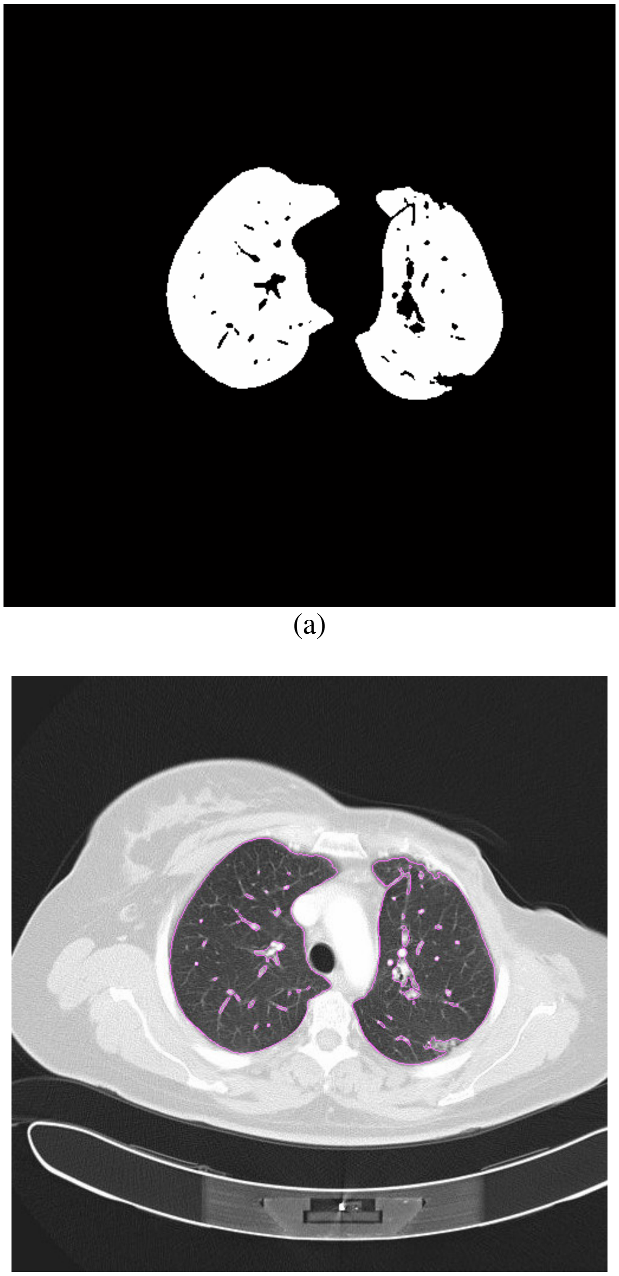
Figure 2.2.6: Result of segmentation for bones



(a)

(b)

Figure 2.2.7: Result of segmentation for heart (a) resulting contour (b) original image with resulting contour



(a)

(b)

Figure 2.2.8*:* Result of segmentation for lungs (a) obtained region after segmentation (b) original image with resulting contour

2.3. The Watershed Transformation

Watershed transformation can be classified as a region-based segmentation approach, and it is from the field of mathematical morphology. Watershed transformation separates an image into two different classes, namely catchment basins and watershed lines. Consider a gray-scale image as a topographic surface, in which the elevation of each point in the surface is directly related to the corresponding pixels gray level, and consider rain is gradually falling on the terrain. By accumulation of water "lakes" (called *catchment basins)* are formed, and lines that separate lakes are called watersheds lines [36].

Generally, the watershed transform is computed on the gradient of the original image, so the catchment basins correspond to the homogeneous grey level regions, and the catchment basin boundaries, which are located at high gradient points, correspond to *watershed lines*.

The watershed transformation is widely used in many fields of image processing, including medical image segmentation, due to the number of advantages it possesses. One of the superior part of watershed transformation over other segmentation methods is that it produces a complete division of the image in separated regions even if the contrast is poor. None of the other segmentation methods divides the image to complete separated regions. The watershed transformation is not effected by lower-contrast edges formed because of noise, the watershed lines always correspond to the most significant edges between catchment basins. Finally, even if there are no strong edges between the local minima, the watershed transform always detects a contour in the area [36].

Some important drawbacks also exist, and they have been widely treated in the related literature. Among the most important are as follows [36]:

- **Oversegmentation**: When the watershed transform extract catchment basins from the gradient of the image, the result of the watershed transform contains too many small regions, which makes this result hardly useful. Several methods were used to get rid of over segmentation problem. A marker image can be used to reduce the number of minima and also the number of regions [37,38]. Moreover, using different filters, like anisotropic diffusion filters the number of minima can be reduced [39].

- **Sensitivity to noise:** Local variations of the image can dramatically change the results. This affect is worsened by the use of high pass filters to estimate the gradient, which amplify the noise. Anisotropic filters have been used to minimize this problem [40].

- **Poor detection of thin structures:** The watershed transformation is generally applied on the gradient image. The smoothing associated with gradient estimation causes spreading of edges, and gradient values are stored only at the image pixel positions in the absence of sub-pixel accuracy. The result is poor detection of thin structures, which is critical for successful segmentation of medical images [36].

In our study, as preprocessing, anisotropic diffusion filtering was applied to the gradient image to remove the noise and to smooth the image to get rid of oversegmentation. Also different strategies such as marker-controlled segmentation can be used to overcome this problem [37,38]. However, poor detection of thin structures problem cannot be solved completely. This problem was mostly encountered when we were extracting the bones, and it can be solved by decreasing the smoothing applied at the gradient estimation process. On the other hand, decreasing the smoothing leads to over segmentation that cannot be solved by anisotropic diffusion filtering anymore. Finally, we decided to apply median filtering instead of Gaussian smoothing to the original image before gradient estimation, which reduces the noise in image while mostly preserving edges.

Applying median filtering was useful but cannot solve the problem completely. The chart of the method is given in Figure-2.3.1.

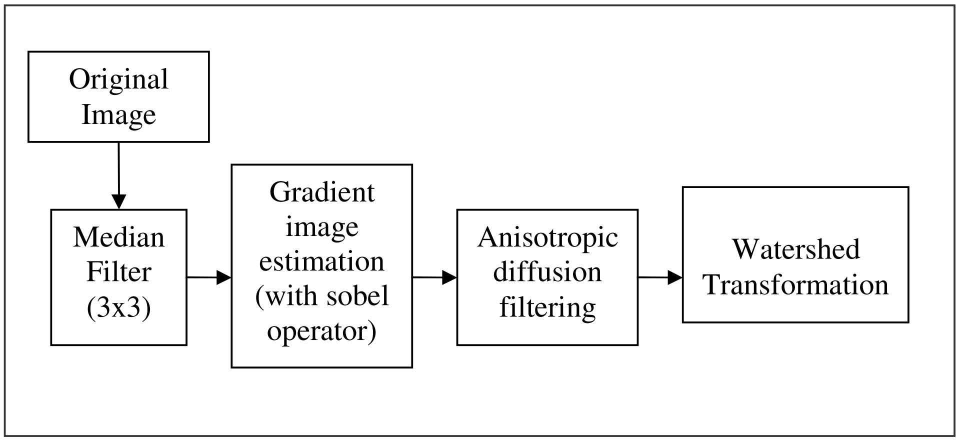

Original Image
Median Filter (3x3)
Gradient image estimation (with sobel operator)
Anisotropic diffusion filtering
Watershed Transformation

Figure 2.3.1*:* The chart of the watershed transformation method

2.3.1. Definition and Computation of Watersheds

Consider a two-dimensional gray scale image *I*, defined in $D_I \subset Z^2$. The image *I* can take gray scale values between the range (0, N), G is the digital grid, and $N_G(p)$ is the set of the neighbors of a pixel p with respect to G [41].

A minimum point in the image is defined as follows:

Defn 1: A minimum M at altitude *h* is a connected plateau of pixels with the value *h* from which it is impossible to reach a point of lower altitude without having to climb. A minimum is thus a connected and iso-intensive area where the gray level is strictly darker than on the neighboring pixels [41].

Catchment basins and watershed lines can be defined as follows:

Defn 2: The catchment basin, C(M) associated with a minimum M, is the set of pixels p of D_I such that water drop falling at p flows down along the relief, following a certain descending path called the *downstream* of p and eventually reaching M. The lines which separate different catchment basins are called watersheds of *I* (see Figure-2.3.2). The catchment basins of an image correspond to the *influence zones* of its minima [41,42].

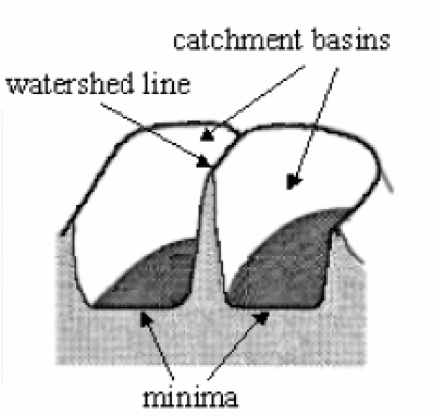


Figure 2.3 2: Watershed lines and catchment basins [41] .

2.3.1.1. Algorithmic Definition By Immersion

We can also define watersheds in an algorithmic approach, which is more convenient for practical application. Let,

I : grayscale image,

I(p) : gray level value of pixel p,

$h_{\min}$ and $h_{\max}$: smallest and largest gray level values in D_I respectively.

$T_h(I)$: set of image pixels whose gray level value is equal to or less than *h :*

$$T_h(I) = \{p \in D_I, I(p) \leq h\} \qquad \textbf{(2.3.1)}$$

$C(M)$: catchment basin associated with the minimum M,

$C_h(M)$: subset of $C(M)$ that includes points with altitude value smaller than or equal to h :

$$C_h(M) = \{p \in C(M), I(p) \leq h\} = C(M) \bigcap T_h(I) \qquad \textbf{(2.3.2)}$$

Defn 3: Let A be a set of connected pixels, and path P_{xy} is the path joining two pixels in A, then we can define the geodesic distance $d_A(x, y)$ as the smallest length of the path which join pixels x and y in A (see Figure-2.3.3).

$$d_A(x, y) = Length(P_{xy})$$

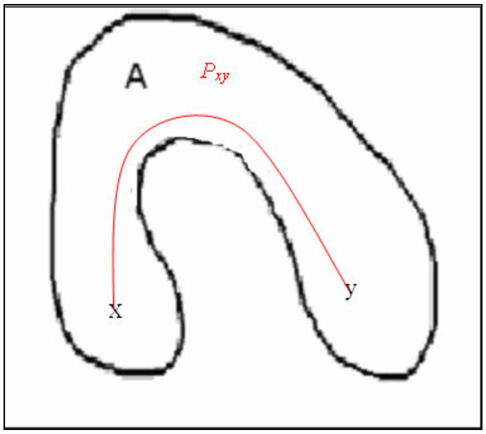


Figure 2.3.3: Geodesic distance

Suppose that A contains a set B composed of n connected components $B_1, B_2 ... B_n$

Defn 4: The geodesic influence zone $iz_A(B_i)$ $(1 \leq i \leq n)$, is the set of points of A that whose geodesic distance to B_i is smaller than their geodesic distance to any other component of B (see Figure-2.3.4).

$$iz_A(B_i) = \{p \in A, \forall j \in [1....n]/\{i\} : d_A(p_i, B_i) < d_A(p_i, B_j)\} \qquad \textbf{(2.3.3)}$$

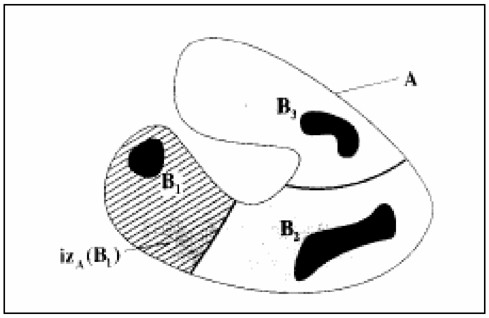


Figure 2.3.4: Geodesic influence zones [41].

The set $IZ_A(B_i)$ is the union of the geodesic influence zones of the connected components B_i

$$IZ_A(B) = \bigcup_{i=1}^{n} iz_A(B_i) \qquad \textbf{(2.3.4)}$$

The set points of that do not belong to any influence zone $iz_A(B_i)$, constitute the *skeleton by influence zones* (SKIZ) of B inside A, $SKIZ_A(B)$ (see Figure-2.3.5).

$$SKIZ_A(B) = A - IZ_A(B\ \ ) \qquad \textbf{(2.3.5)}$$

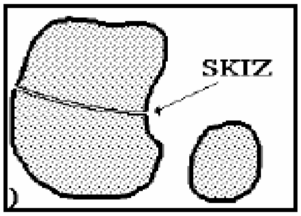


Figure 2.3 5: Skeleton by influence zones [41].

Immersion Approach: Imagine that a hole is drilled in each minimum of the surface, and we flood water into different catchment basins from the holes. If the water of different catchment basins is likely to merge due to further immersion, a dam is built to prevent the merging. This flooding process will eventually reach a stage when only the top of the dam (the watershed lines) is visible above the water line. In Figure-2.3.6 1D immersion is illustrated.

In order to simulate the immersion process, a recursion process is defined. In the recursion process, the grey level h increasing from h_{min} to $h_{max,}$ and the basins associated with the minima of I are expanded. Let the set of points that have the gray level h or less than h is given as $T_h(I)$, and X_h be the union of the set of basins computed at level h. Then at gray level $h+1$, a connected component of the threshold set $T_{h+1}(I)$ can be either a new minimum, or an extension of a basin in gray level h, namely X_h. If the connected component of the threshold set $T_{h+1}(I)$ is an extension of X_h, then X_{h+1} is updated by computing the geodesic influence zone of X_h in $T_{h+1}(I)$ [42] .

The set of catchment basins for gray level value $h_{\max}$, can be computed using the following recursion,

$$X_{h_{\min}} = T_{h_{\min}}(I)$$

$$X_{h+1} = \min_{h+1} \bigcup IZ_{T_{h+1}(I)}(X_h) \qquad h \in [h_{\min}, h_{\max}] \qquad \textbf{(2.3.6)}$$

where $\min_{h+1}$ is the union of all regional minima at altitude $h+1$.

The watersheds lines of I correspond to the set of pixels in D_I that do not belong to any catchment basin (see Figure-2.3.7).

$$Wshed(I) = D \setminus X_{h_{\max}} \qquad \textbf{(2.3.7)}$$

For better understanding the watershed transform according to recurrence in (2.3.6), an example is given in Figure-2.3.8. A, B, C and D are labels of basins and W is watershed pixels. Immersion is applied on the 8-connected grid. Minima pixels and labels are shown with bold text [43].

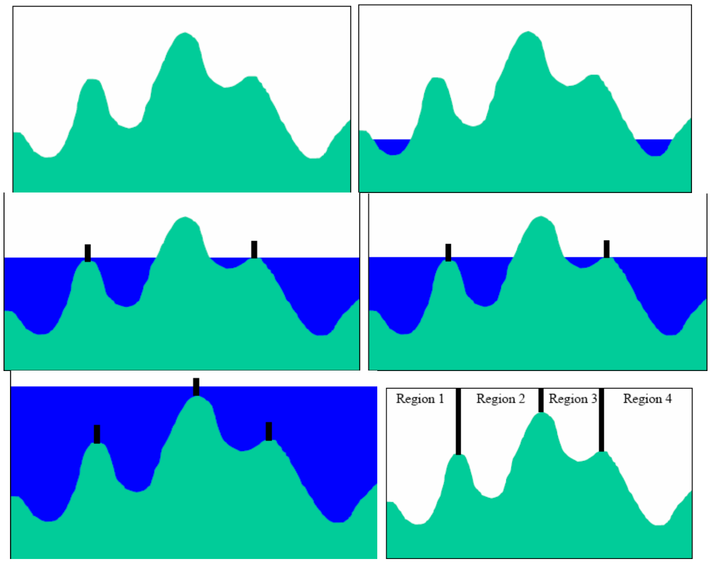


Figure 2.3.6: 1D Immersion Process

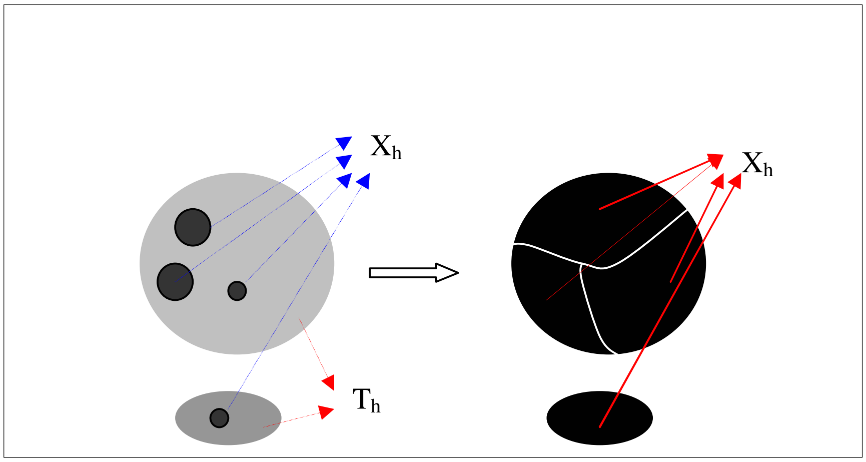


Figure 2.3.7: Watershed Lines

1	2	**0**
2	2	1
3	4	3
0	3	1

(a) original image

1	2	**B**
2	2	1
3	4	3
A	3	1

(b) h=0

C	2	**B**
2	2	**B**
3	4	3
A	3	**D**

(c) h =1

C	**W**	**B**
C	**W**	**B**
3	4	3
A	3	**D**

(d) h=2

C	**W**	**B**
C	**W**	**B**
W	**4**	**W**
A	**W**	**D**

(e) h=3

C	**W**	**B**
C	**W**	**B**
W	**W**	**W**
A	**W**	**D**

(f) h=4

Figure 2.3.8: Watershed Transform by immersion on the 8-connected grid, (a) original image, (b-e) labeling step

In this example h_{min} and h_{max} are 0 and 4 respectively. For h=0, there are two local minima (the zeroes), so there will be two basins whose pixels are labeled A and B (see Figure-2.3.8(b)). When the current gray level is $h=1$, then the 1s in Figure-2.3.8(b) are candidates to get assigned to a catchment basin. Because the pixel in 2nd row, 3rd column is in the influence zone of minima labeled with B, it is labeled as B. Pixels in the 4th row, 3rd column and 1st row, 1st column are new minima therefore labeled as D and C respectively. When the current gray level is $h=2$, then the pixels with gray level two in Figure-2.3.8(c) are candidates to get assigned to a catchment basin or labeled as W. Because the pixel in 2nd row, 1st column is in the influence zone of minima labeled with C, it is labeled as C. Pixels in the 2nd row, 2nd column and 1st row, 2nd column are labeled as watershed pixel because they are equidistant to basins C and B (see Figure-2.3.8(d)). Similarly, labeling according to equation 2.3.6 , for $h=3$ and $h=4$ are shown in Figure- 2.3.8(e) and Figure- 2.3.8(f) [42]. Further explanation of the algorithm is given in Appendix-B.2.

2.3.2. Experimental Results and Conclusion:

- **Simulation studies : Test image Results**

Let us observe the results of watershed transformation on a test image given in Figure-2.3.9. The test image contains three different objects and the gray level value is in the range of (21-192). The square object is unevenly illuminated and its gray level value is in the range of (21-124). The circular and rectangular objects haves the gray level values of 51 and 95 respectively.

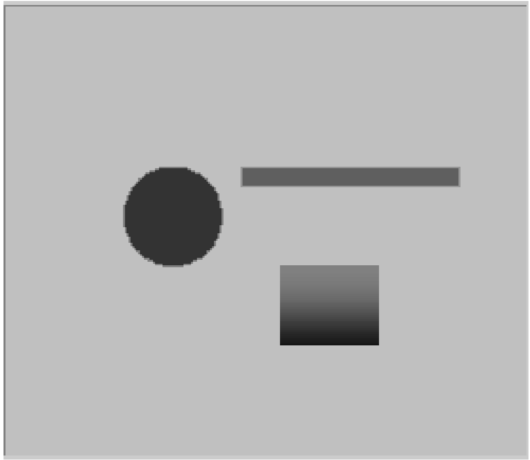


Figure 2.3.9*:* Original Test image

- **Preprocessing :**

In the preprocessing part we apply 3x3 median filtering for smoothing the image. However, for this artificial image, applying median filtering is not very important because the image is noise free and smooth.

After median filtering we get the image in Figure 2.3.10.

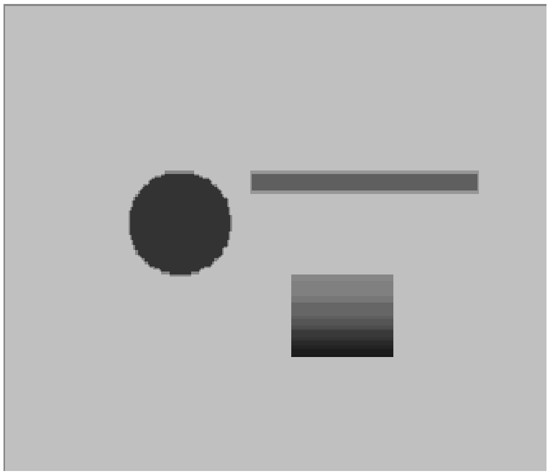


Figure 2.3.10: Test image after median filtering

- **Gradient image Computation:**

The next step is computing the gradient image. We used sobel operator for computing the gradient image [44]. The resultant gradient image is in Figure -2.3.11 (a).

- **Watershed Transformation**

After computing the gradient image given in Figure-2.3.11(a), we applied watershed transformation to the gradient image. In this study, we used the Vincent-Soille algorithm, explained in Appendix-B.2 [42]. For different algorithms see [42,45]. The resultant segmented image with 11 sub-regions is given in Figure-2.3.11 (b). In this image watershed lines are given in black, and catchment basins are labeled with different colors. In Figure-2.3.11(c) original image overlapped with the watershed lines is given, where watershed lines are in white.

We can observe that circular and rectangular objects are perfectly segmented by watershed transformation. In addition for the unevenly illuminated square object, watershed transformation detected several catchment basins. Each catchment basin consists of almost homogeneous regions, and divided by watershed lines.

Because the test image in Figure-2.3.9 is smooth and noise free, it did not suffer from oversegmentation. However, for non-smooth or noisy images watershed transformation generally leads to oversegmentation. In Figure-2.3.12, watershed transformation is applied to the noisy image, where the SNR is 33 dB. The noisy image is given in Figure-2.3.12 (a) and smoothed image is given in Figure-2.3.12 (b).

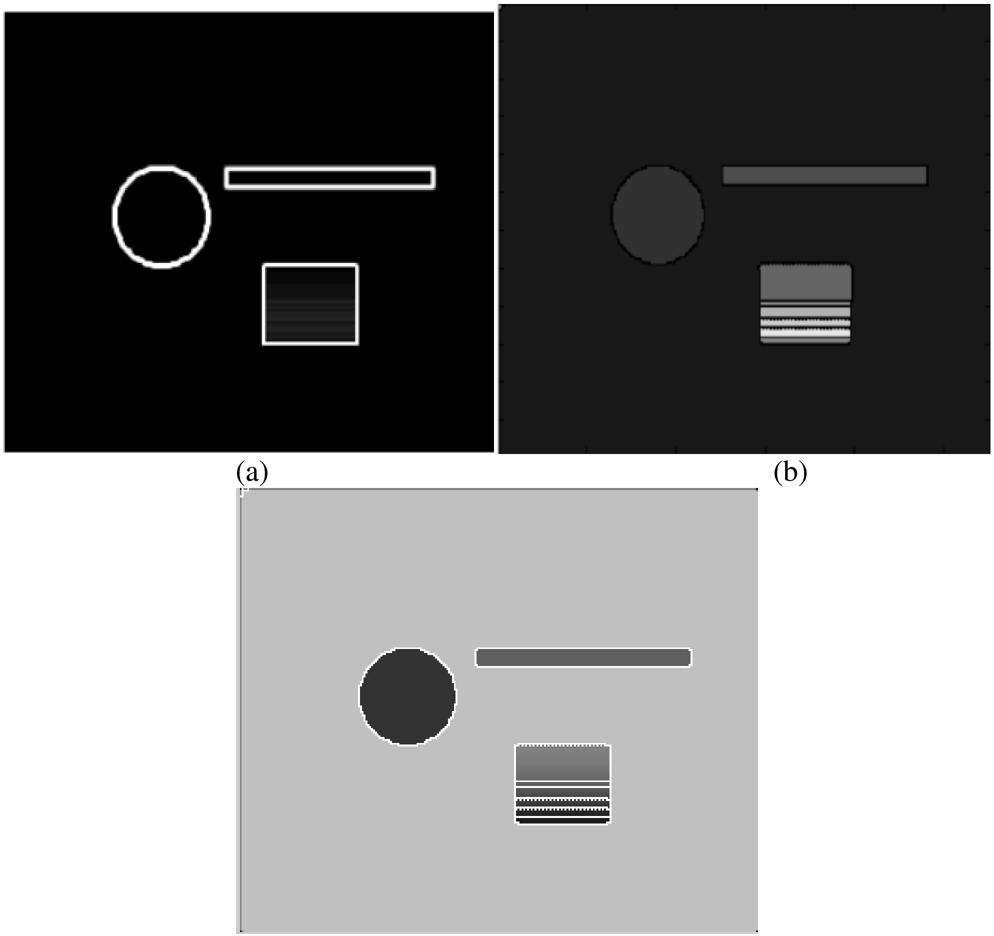
Firstly, we applied watershed transformation directly to noisy gradient image given in Figure-2.3.12(c). The result of watershed transformation is given in Figure-2.3.12(d), where the watershed lines are given in black and catchment basins are given in white, and total number of sub-regions is 6142. It is obvious that watershed transformation is very sensitive to noise, which causes oversegmentation. Therefore to get rid of oversegmentation problem, we applied anisotropic diffusion filtering to gradient image in Figure-2.3.12(c).

In Figure-2.3.13(a), the resultant gradient image after anisotropic diffusion filtering (Appendix-A) is given. Filter specifications are:

Number of Iterations: 30
Diffusion Const (K) : 30
Lambda :0.25

The result of watershed transformation using anisotropic diffusion filtering is given in Figure-2.3.13, where the number of sub-regions (catchment basins) is 147. Original image overlapped with the watershed lines is given in Figure-2.3.13(c).

We can see that by applying anisotropic diffusion filtering to the gradient image before watershed transformation, the result of segmentation is refined by decreasing number of sub-regions is from 6142 to 147. On the other hand, due to the smoothing effect of anisotropic diffusion filtering, the square object with gradual change in gray level can not be segmented accurately.



(a) (b)

(c)

Figure 2.3.11: Result of segmentation for test image (a) gradient image, (b) watershed transformed image, (c) original image overlapped with watershed lines.

- **Segmentation of CT images**

Next, we applied watershed transformation on the CT image, given in Figure-2.3.14(a).

- **Preprocessing :**

In the preprocessing step original image is smoothed using a 3x3 median filter to avoid detection of false edges due to noise, and decrease the number of weak edge pixels. Smoothed image is given in Figure-2.3.14(b). Because we used median filter while smoothing the original image most of the edge pixels are preserved.

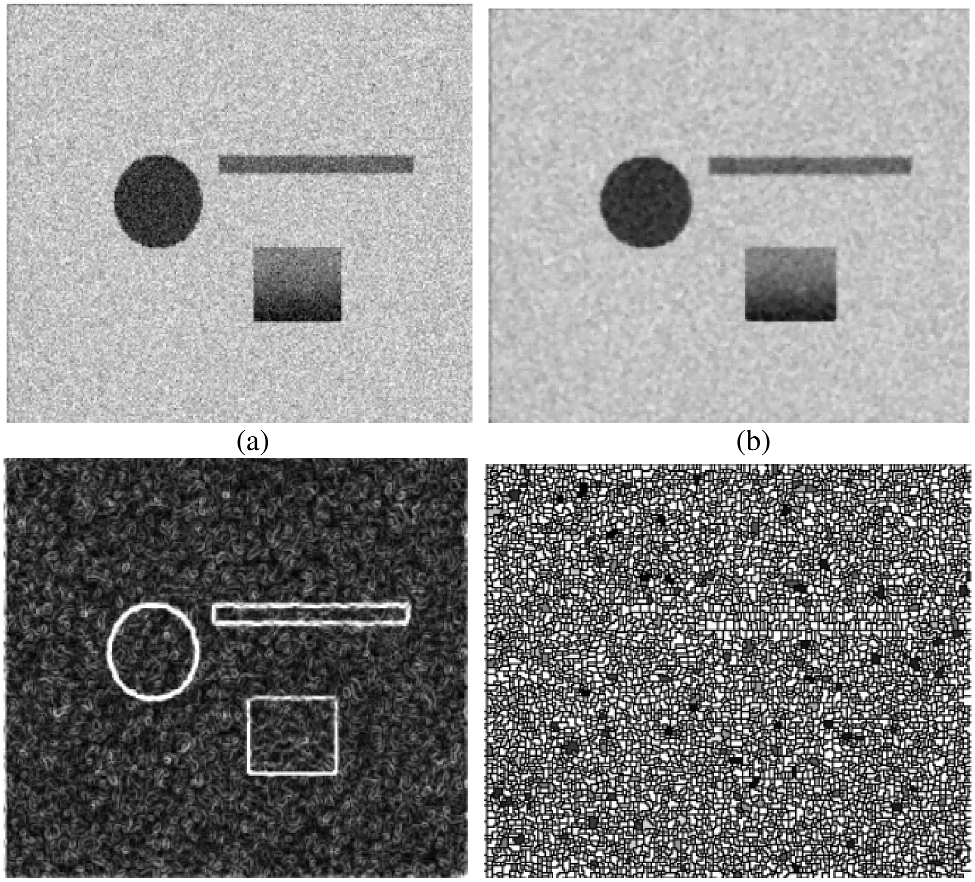


Figure 2.3.12: **:** Result of segmentation for the noisy test image (a) noisy image, (b) smoothed noisy image , (c) gradient of smoothed noisy image (d) result of watershed transformation

- **Gradient image Computation:**

After smoothing process we computed the gradient image using the sobel operator. Computed gradient image is given in Figure-2.3.14(c).

- **Watershed Transformation**

After computing the gradient image, we applied the watershed transformation to the gradient image and obtained the watershed transformed image, which is given in Figure-2.3.15.

It is obvious that the number of regions is extremely high (21834), and as in the first example we can apply anisotropic diffusion filtering on the gradient image to overcome over-segmentation with the following parameters:

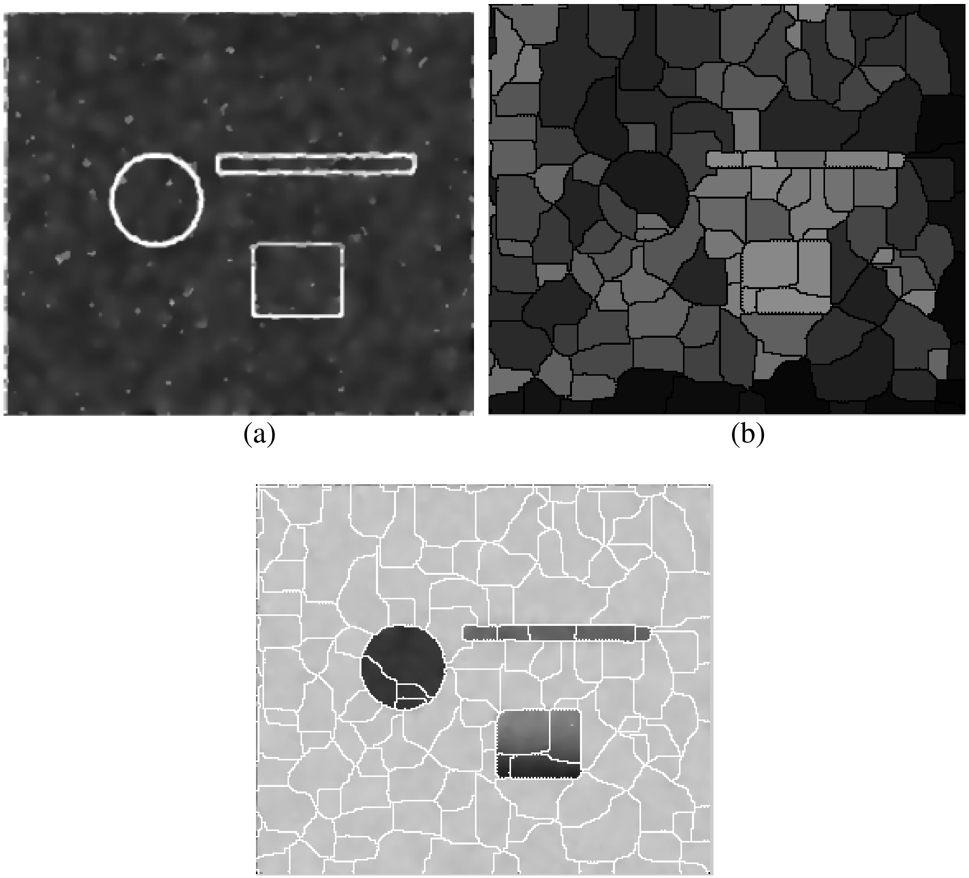
Number of Iterations: 30

Diffusion Const. (K): 25

Lambda=0.25;

$$c(x,y,t)=\exp\left(-\left(\frac{\left|\nabla I(x,y,t)\right|}{K}\right)^2\right)$$

The resulting smoothed gradient image is shown in Figure-2.3.16. Result of watershed transformation obtained by using the smoothed gradient image is in Figure-2.3.17(a), where catchment basins are given in white and watershed lines are given in black, and the number of sub-regions (catchment basins) is 7843. In Figure-2.3.17(b), original image overlapped with watershed lines is given, where the watershed lines are in red.

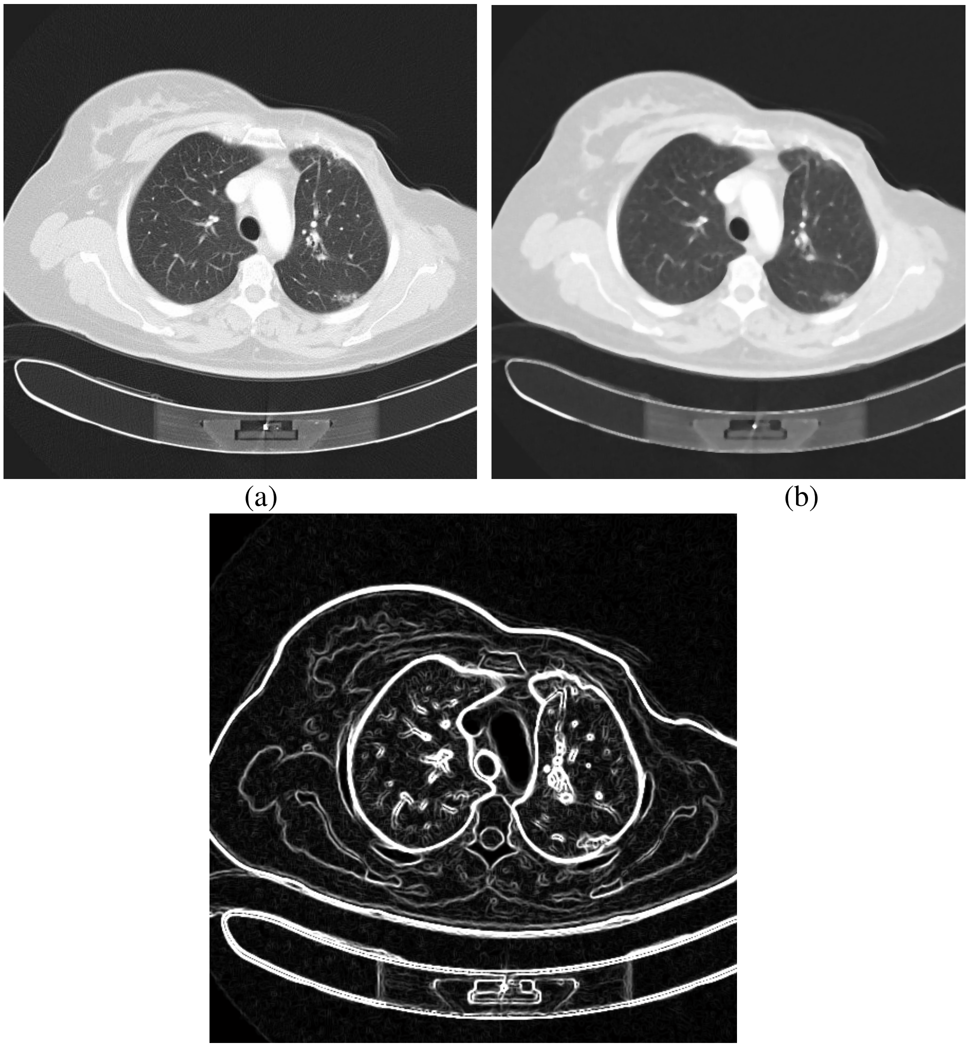


(a)

(b)

(c)

Figure 2.3.13: Result of segmentation with anisotropic diffusion filtering (a) smoothed gradient image, (b) Watershed Transformed image (watershed lines are black and each catchment basin is given with different gray tone), (c) original image overlapped with the watershed lines



(a) (b)

(c)

Figure 2.3 14: Preprocessing results (a) Original image, (b) smoothed image with median filtering , (c) computed gradient image

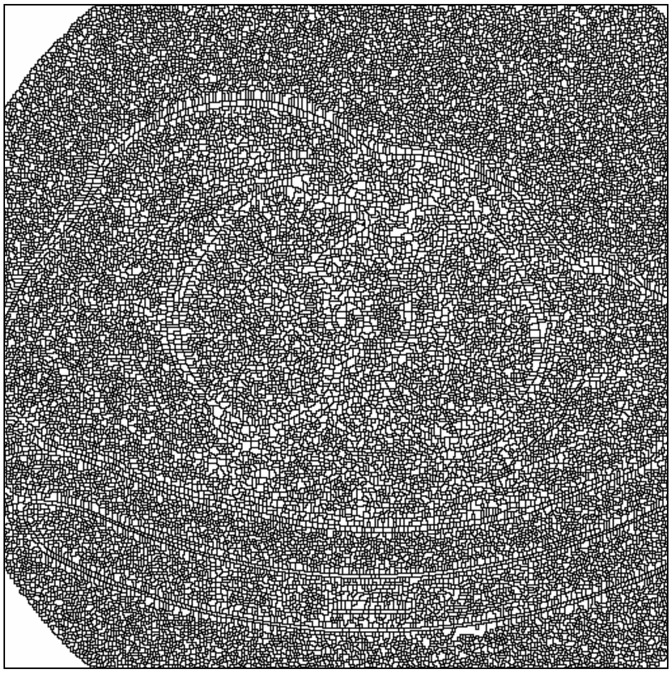


Figure 2.3.15: Result of watershed transformation (# of regions: 21834)

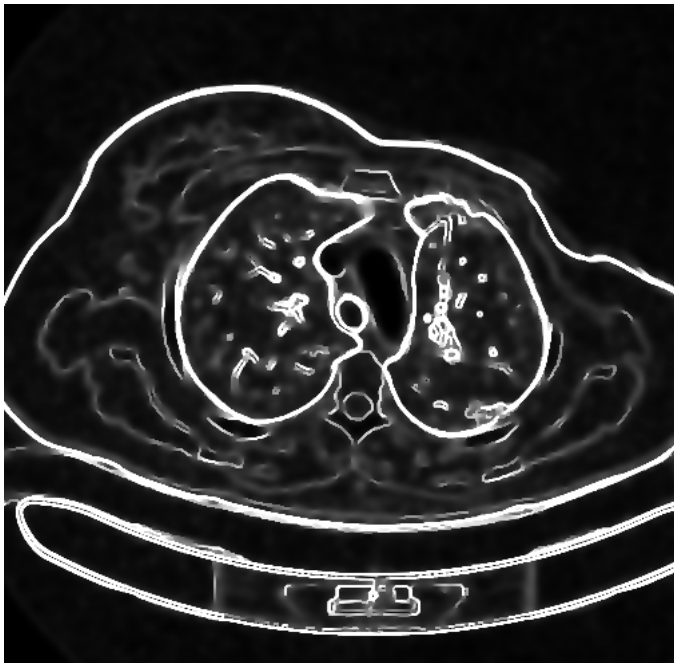
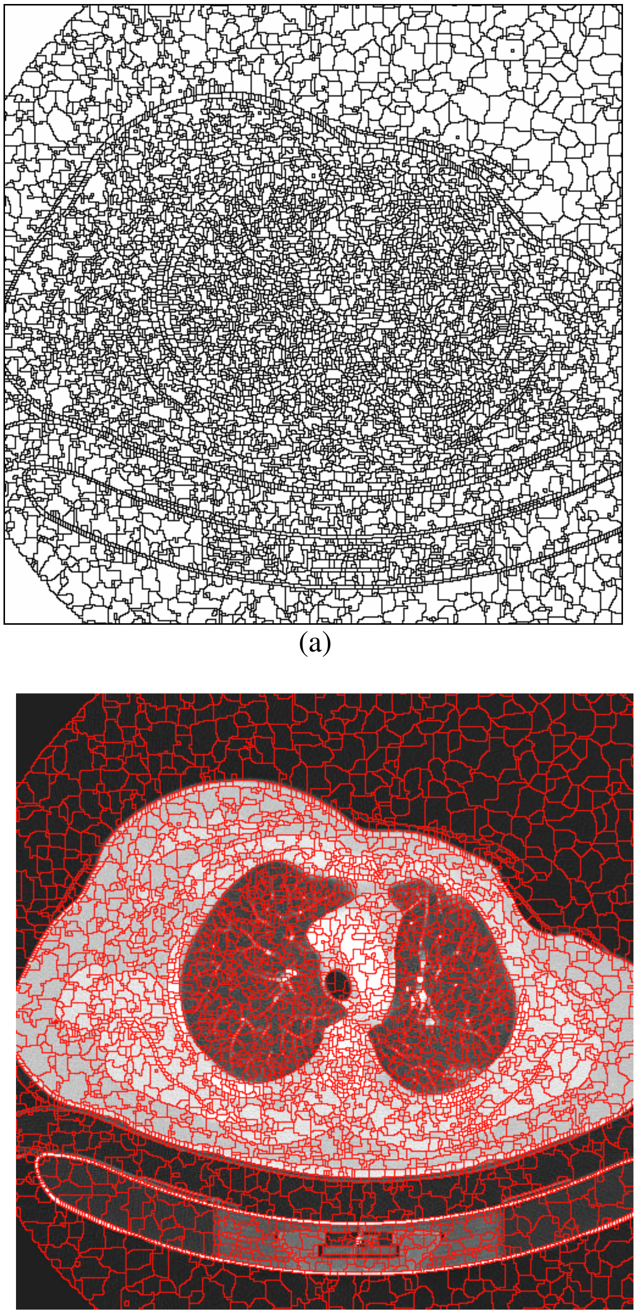


Figure 2.3.16: Smoothed gradient image



(a)

(b)

Figure 2.3.17: Result of watershed transformation for the CT image after anisotropic filtering, (a) watershed segmented image, (b) original image overlapped with watershed lines.

In our study, we used anisotropic diffusion filtering to handle the oversegmentation problem. However, we can not overcome this problem completely.

The most superior part of watershed transformation is that it performs a complete division of the image in homogeneous separated sub-regions. However, in general a sub-region does not contain the whole object to be segmented. By observing the result of watershed segmentation in Figure-2.3.17, we can state that, none of the sub-regions includes all the pixels belonging to tissues or organs to be segmented. Instead, tissues or organs to be segmented consist of several sub-regions. Due to the fact that it needs post-processing for complete segmentation of bones, lungs, heart, fat and muscle tissues, we called watershed transformed image as *primitively segmented image.* In order to perform segmentation of these tissues or organs; a hybrid method is used in Section-3.1, where the sub-regions of primitively segmented image are merged with respect to a merging criterion. In this hybrid method, the segmentation of all tissues or organs can be performed simultaneously and automatically.

2.4. Deformable Models

2.4.1. Introduction

There are two general types of deformable models: **parametric deformable models** and **geometric deformable models**. Parametric deformable models represent curves and surfaces explicitly in their parametric forms during deformation. This representation allows direct interaction with the model. However, adaptation of the model topology, such as splitting or merging parts during the deformation can be difficult using parametric models. Geometric deformable models, on the other hand, can handle topological changes naturally. These models, based on the theory of curve evolution and the level set method, represent curves and surfaces implicitly as a level set of a higher-dimensional scalar function.

In our study we implemented three different deformable model; traditional snake model, snake with gradient vector flow (GVF), and level set methods. Traditional snake model and snake with GVF are classified under parametric deformable models, and Level set methods are classified as geometric deformable models.

2.4.2. Parametric deformable models

Active contours models, also called snakes, were proposed by Kaas as a solution to the task of finding salient contours like edges and lines in digitized images [46]. An active contour model represents an object boundary or some other salient image feature (e.g. shape) as a parametric curve that is allowed to deform from some initial shape towards the desired final shape. The problem of finding this final contour is an energy minimization problem with the intention that the final contour yields a local minimum of an associated energy function. The energy function of the contour is defined such that the energy of the contour attains a local minimum when the contour is spatially aligned with the shape or object boundary of interest in the image. The energy functional is thus based upon the spatial features of the image under inspection that we want to detect. In this study our goal was to obtain the boundaries of the objects.

Snakes move under the effect of *internal forces*, which are defined within the curve itself, and *external forces*, which are computed from the image data. The internal forces are designed to keep the model smooth during deformation. The external forces are defined to move the model toward an object boundary or other desired features within an image.

2.4.2.1. Traditional Snake Model

In parametric active contours**,** the curves are drawn toward the edges by external forces, which are defined as the negative gradient of a potential function.

There are also internal forces designed to hold the curve together (elasticity forces) and to keep it from bending too much (bending forces) [47].

A traditional snake is a curve, *X*(s) =[*x*(s), *y*(s)], where *s* a is parameter between [0, 1], that increases as we move around the contour and is related to arc length. This curve moves through the spatial domain of an image to minimize the following energy function:

$$E(X) = S(X) + P(X) \qquad \textbf{(2.4.1)}$$

Having specified the contour as *X*(s)*,* our model is defined as a sum of energy terms in the continuous spatial domain. The energy terms can be categorized as internal energy and external energy.

2.4.2.1.1. Internal Energy (S(X))

Internal energy is a function of the contour *X*(s) itself and it specifies the tension and smoothness of the curve. It therefore depends on the internal properties of the snake. It is the sum of elastic energy and bending energy.

$$S(X) = E_{elastic} + E_{bending} \qquad \textbf{(2.4.2)}$$

Elastic Energy**:** makes the snake act like a membrane that is treated as an elastic rubber band possessing elastic potential energy. It discourages stretching by introducing tension [48]. Elastic energy is defined as:

$$E_{Elastic} = \frac{1}{2}\int_0^1 \left[\alpha(s) \left| \frac{\partial X}{\partial s} \right|^2 \right] ds \qquad \textbf{(2.4.3)}$$

where α is a measure of the elasticity of the snake (generally selected as a constant [49]). This weighting term allows us to control elastic energy along different parts of the contour. It is responsible for shrinking of the contour.

Bending Energy**:** makes the snake behave like a thin metal strip; it discourages bending and makes the model behave like a rigid rod. It is defined as sum of squared curvature of the contour [48].

$$E_{Bending} = \frac{1}{2}\int_0^1 \left[\beta(s)\left|\frac{\partial^2 X}{\partial s^2}\right|^2 \right] ds \qquad \textbf{(2.4.4)}$$

where β is a measure of the stiffness of the snake that controls the rigidity of the snake (similar to α, generally selected as a constant). This weighting term allows us to control the contribution of the bending energy to the total snake energy.

$$S(X) = \frac{1}{2}\int_0^1 \left[\alpha(s)\left|\frac{\partial X}{\partial s}\right|^2 + \beta(s)\left|\frac{\partial^2 X}{\partial s^2}\right|^2 \right] ds \qquad \textbf{(2.4.5)}$$

The first term in (2.4.3) corresponds to the elastic energy ($E_{elastic}$) and the second term bending energy ($E_{bending}$) respectively.

Adjusting the weights α(s) and β(s) controls the relative importance of the elastic and bending energy terms and therefore the internal energy of the contours. We consider them to be simply constants α and β.

2.4.2.1.2. External Energy (P(X))

The external energy function *P(X)* is computed from the image itself. A function P*(X) = E_{image}(X)* must be defined to take smaller values at the features of interest, such as at the boundaries. External image energy of the whole contour ($E_{EXT} = P(X)$) is then defined as

$$E_{EXT} = \int_0^1 \left[E_{image}(X(s)) \right] ds \qquad \textbf{(2.4.6)}$$

Because the gradient of an image takes higher values at the boundaries we can select an image function $E_{image}(X)$ related with the negative of the gradient. Some possible E_{EXT} definitions to pull an active contour toward step edges are:

$$E_{EXT}(x, y) = -\left|\nabla I(x, y)\right|^2 \qquad \textbf{(2.4.7)}$$

$$E_{EXT}(x,y) = -\left|\nabla\left[G_{\sigma}(x,y) * I(x,y)\right]\right|^{2} \qquad \textbf{(2.4 8)}$$

Where *I(x,y)* is the original image and $G_{\sigma}(x,y)$ is the two-dimensional Gaussian function with standard deviation σ, and ∇ is the gradient operator. Using a large σ for $G_{\sigma}(x,y)$ makes the image more blurry and also causes spreading of the boundaries. On the other hand this increases the capture range of the snake.

2.4.2.1.3. Energy and force equations:

The problem of finding a curve that minimizes the energy functional (2.4.1) is known as a variational problem. It has been shown that the curve that minimizes the energy function must satisfy the following Euler-Lagrange equation [48]:

$$\frac{\partial}{\partial s}\left(\alpha\frac{\partial X}{\partial s}\right) - \frac{\partial^{2}}{\partial s^{2}}\left(\beta\frac{\partial^{2} X}{\partial s^{2}}\right) - \nabla E_{EXT} = 0 \qquad \textbf{(2.4 9)}$$

which can be rewritten as,

$$\alpha X^{''}(s) - \beta X^{''''}(s) - \nabla E_{EXT} = 0 \qquad \textbf{(2.4 10)}$$

This equation can be interpreted as a force balance equation, that is composed of internal and external forces.

$$F_{\text{int}} + F_{ext} = 0 \qquad \textbf{(2.4.11)}$$

Internal force (F_{int}): discourages stretching and bending of the snake and it is composed of elastic and bending force.

$$F_{\text{int}} = F_{elastic} + F_{bending} \qquad \textbf{(2.4.12)}$$

$F_{elastic}$: causes the contour to expand until the concavity is eliminated. The other function of elastic forces is to shrink the curve and it is responsible for collapsing the contour to a single point in the absence of any opposing force. Elastic force corresponds to the first term in equation (2.4.10):

$$F_{elastic} = \alpha X^{''}(s) \qquad \textbf{(2.4.13)}$$

The figure below shows the action of elastic force on the curve. The initial contour was allowed to deform freely under the influence of only elastic force. Notice how the concavities are eliminated and the entire curve is shrinking. If we would allow the curve to move freely for a few more iterations, it will eventually collapse to a point.

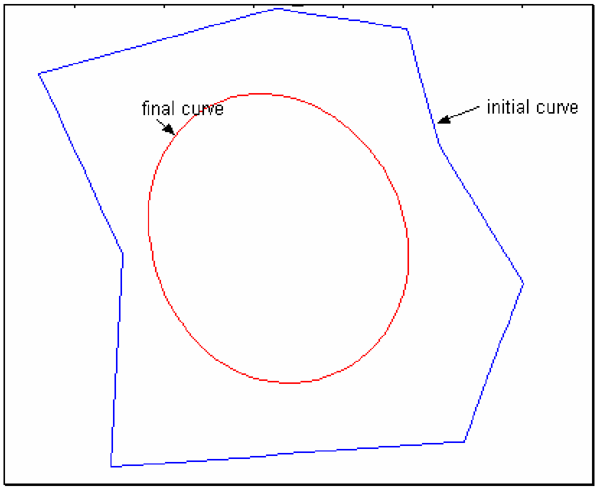


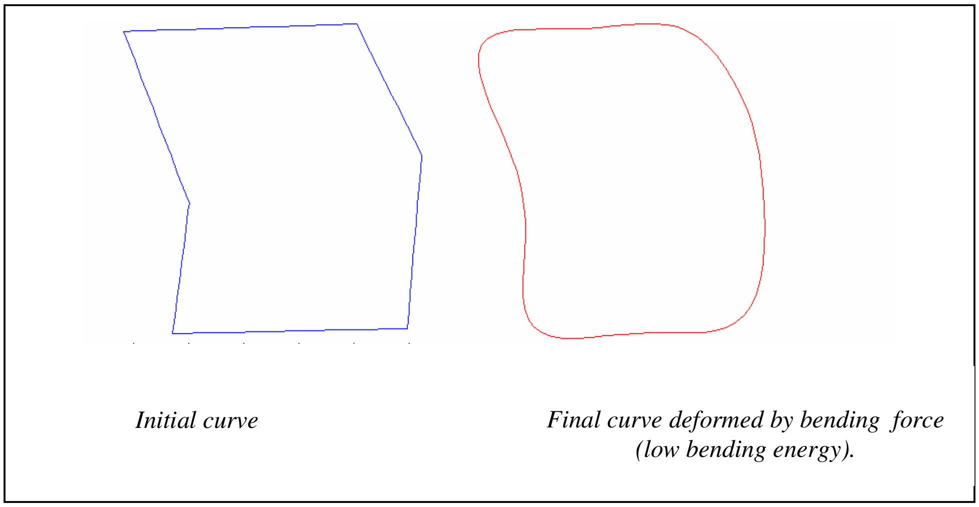
Figure 2.4.1 : Effect of the elastic force.

$F_{bending}$: Bending force acts on the curve due to the bending energy of the contour and it corresponds to the second term in (2.4.10):

$$F_{bending} = \beta X^{''''}(s) \qquad \textbf{(2.4 14)}$$

Bending force is strong at the corners, i.e. at the points with very high curvature and it tries to eliminate these sharp corners thereby smoothing the contour.

Figure-2.4.2 helps us understand the effect of the bending force. The initial curve was allowed to deform only under the influence of the bending force results in the final curve. Notice how all the corners were smoothed out thereby reducing the bending energy of the contour. If we continue this for more number of iterations the bending force will eventually seek to turn the curve into circle because a circle has the lowest bending energy.



Initial curve

Final curve deformed by bending force (low bending energy).

Figure 2.4. 2: Effect of Bending Force

External Force:

External Force (F_{ext}) acts in the direction so as to minimize the energy computed from image E_{EXT} , and it corresponds to the last term in (2.4.10):

$$F_{ext} = -\nabla E_{EXT} \qquad \textbf{(2.4.15)}$$

The choice of F_{ext} has a big effect on both the implementation and the behavior of a snake. External forces are divided into two classes, static and dynamic external forces. Static external forces are computed once for an image and do not change when snake deforms, on the other hand dynamic external forces changes as the snake deforms [47,50]. Because using dynamic external forces requires a huge computational effort, we used static external forces in our study. We used two different static external force models, the traditional model, which is the subject of this sub-chapter, and GVF, as explained in section 2.4.2.2.

For example, the external force computed from the original image in Figure-2.4.3(a) using the energy function given by equation 2.4.15, is shown in Figure-2.4.3 (b). We can see that around the edges of the original image, the external force is non-zero, which makes the snake to move towards the edges. On the other hand, at smooth regions the external range is about to be zero, and only the internal force dominates.

We can see that around smooth regions of the original image, the range of the external force around zero, and only the internal force dominates. As explained previously, when only the internal force dominates, snake is smoothed due to the bending force and collapses to a point due to elastic force. Thereby, the force balance equation is satisfied. On the other hand, around edge regions the external force is non-zero, and therefore snake must deform to minimize the force balance equation.

2.4.2.1.4. Discretization and Implementation of Traditional Snakes:

In practice, one does not study the contour at continious points. Instead, the contour is represented by a vector X(s) of control points. The control points must not be separated by more than a few pixels to prevent the contour from bypassing attractive but small areas in the image. Each control point has a position, given by (x, y) coordinates in the image, and a snake is entirely specified by the number and coordinates of control points. The adjustment of the snake (deformation) is obtained by moving the control points individually. The energy terms, like forces are all converted into the discrete form with the derivatives substituted by finite differences and the integrals substituted by summations. The forces acting along the contour are calculated at each of these control points separately and the entire curve is declared to achieve a minima when each of these control points attain local minima resulting in an energy minimization.

To find a solution for (2.4.9), the deformable contour is made dynamic by defining it as a function of time *t* and *s*. The partial derivative of *X* with respect to *t* is then set equal to the left-hand side of (2.4.9) as follows [47]:

$$\gamma\frac{\partial X}{\partial t}=\frac{\partial}{\partial s}\left(\alpha\frac{\partial X}{\partial s}\right)-\frac{\partial^2}{\partial s^2}\left(\beta\frac{\partial^2 X}{\partial s^2}\right)-\nabla P(X) \qquad \textbf{(2.4 16)}$$

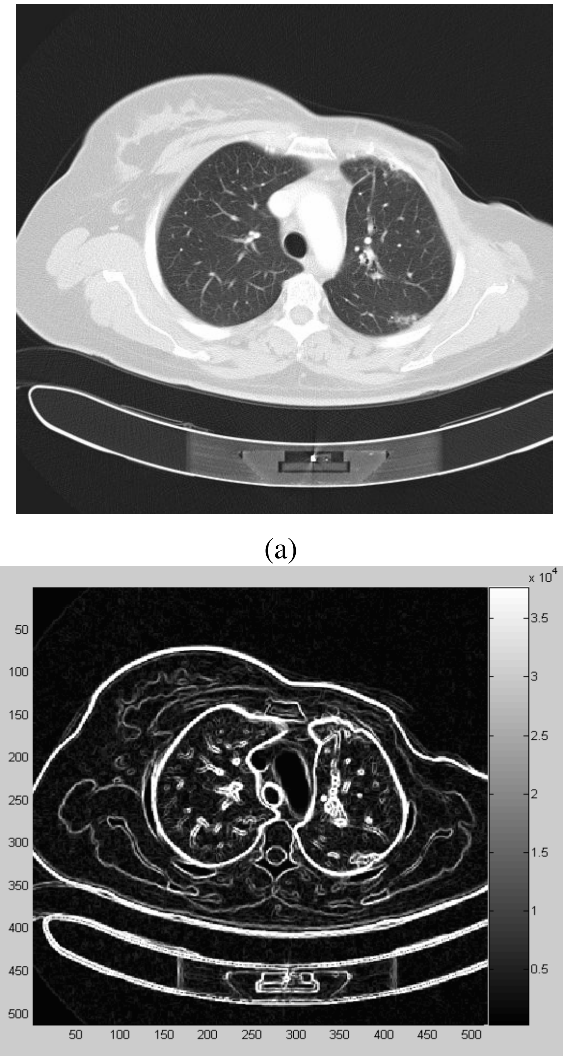
The damping coefficient γ is introduced to make the units on the left side consistent with the right side. When the solution stabilizes, the left side vanishes and we achieve a solution as: .

$$A_i^n=\frac{1}{h^2}\left[\alpha_{i+1}(\overline{X}_{i+1}^n-\overline{X}_i^n)-\alpha_i(\overline{X}_i^n-\overline{X}_{i-1}^n)\right]$$

$$B_i^n=\frac{1}{h^4}\beta_{i-1}(\overline{X}_{i-2}^n-2\overline{X}_{i-1}^n+\overline{X}_i^n)-2\beta_i(\overline{X}_{i-1}^n-2\overline{X}_i^n+\overline{X}_{i+1}^n)+\beta_{i+1}(\overline{X}_i^n-2\overline{X}_{i+1}^n+\overline{X}_{i+2}^n)$$

$$\gamma\frac{\overline{X}_i^n-\overline{X}_i^{n-1}}{\Delta t}=A_i^n-B_i^n-F_{ext}(\overline{X}_i^{n-1}) \qquad \textbf{(2.4.17)}$$

See Appendix-C.1. for derivation and definition of equation 2.4.17. Equation 2.4.17 is solved iteratively, by giving each control point a displacement (push) in the *x* and *y* direction proportional to the sum of forces acting on it. After certain number of iterations we converge to the final contour and convergence is guaranteed. Since we do not have any stopping condition, user has to specify the number of iterations to run. But the problem with this method is that we do not know how many iterations we need to converge to the result.



(a)

(b)

Figure-2.4.3: (a) Original image, (c) computed external force

Implementation Steps

- **Step 1: Suggesting an initial contour**

Snake does not find a desired image's contour automatically. To make the contour attract to the shape which one wants to fit the contour, an initial suggestion should be given. Snake can converge to false edges unless it is positioned close to desired edges.

- **Step 2: Calculating the External Force**

The original image has to be smoothed to create the external force according to (2.4.8) and the image is then filtered with a 3x3 gradient filter (such as sobel operator) to create the image energy. Finally by applying (2.4.15) we get the external force as in Figure-2.4.3 (c) .

- **Step 3: Iteration**

The suggested contour is iterated according to (2.4.17). The iteration will proceed until the desired number of iterations is reached. Finding the true number of iterations to get an eventually stable contour is an ad-hoc method.

In Figure 2.4.4 we see the initial contour given by the user, which is composed of reference points positioned inside the heart. By using the computed external force Figure 2.4.3 (c), and (2.4.17) with 200 iterations, the step size in time h=1, elasticity coefficient α=0.25, bending coefficient β=0.005 and damping coefficient 1, we obtained the resulting contour in Figure 2.4.5.

In Figure 2.4.5, we can see that due to the small capture range of the external force F_{ext} , contour can not taken the same shape of the heart. We also increased the elasticity value of the contour to α= 0.45 and the number of iterations to 300 but the resultant contour could not obtain the desired concavity. This is because of small capture range of the traditional external force.

2.4.2.1.5. Drawbacks of traditional snake model:

Although the model proposed by Kass et al although gave a new approach to the process of image segmentation and seems very promising, it has many weaknesses :

- **Extremely sensitive to parameters:**

There are many different parameters that must be decided (α,β, γ, σ, and distance between control points while discretizing). The performance snake relies heavily on the proper assignment of these parameters by the user. Unfortunately these parameters depend heavily on the underlying image and object of interest.

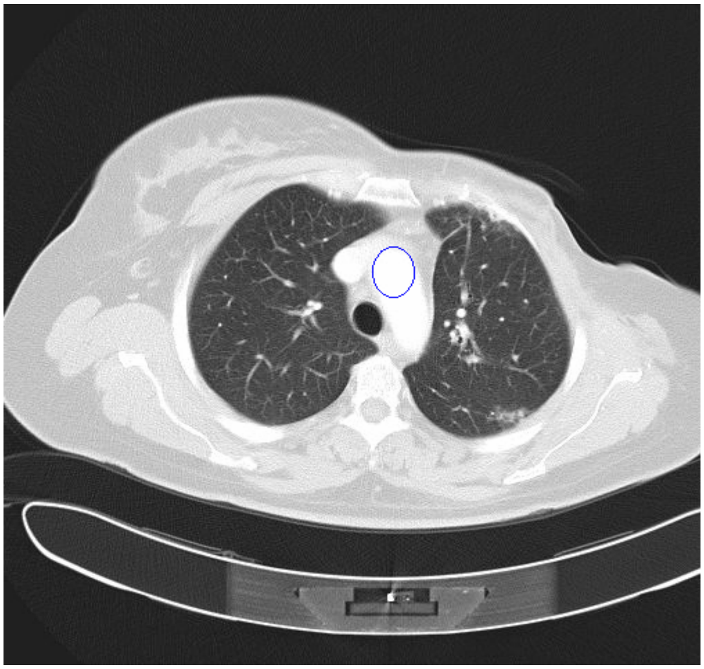


Figure 2.4.4: Original image with initial contour

- **Small capture range:**

For images with very low gradient or with gradient value of 0 for smooth areas, the external force acting on a control point of snake in this region is then 0. This can be observed in Figure 2.4.3 (c) in black areas. Therefore traditional snakes have a very small capture range and they should be initialized close to the object boundary we want to detect. This is a very demanding condition. The capture range of the snake can be increased by applying Gaussian low pass filter with a very high variance to smooth out the edges, but by doing this we will have less accurate results and loose out the concave regions completely and therefore fail to detect boundary concavities.

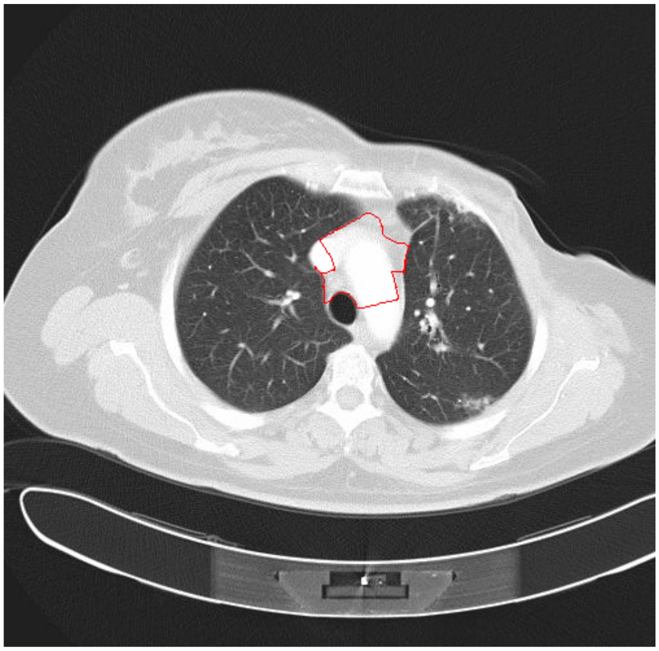


Figure 2.4.5: Result of deformation, image with final contour.

- **Poor detection of boundary concavities**

If object in the image has an edge which shows a concavity then the traditional snake will have a problem in tracing the edge at these concaved parts. One reason is that the capture range at these parts of edges is too far from the snake. Also even if they are not far from snake, the gradient force on each wall of concavity is effectively canceling each other and leaving snake with no gradient force to work with and hence no deformation can take place.

2.4.2.2. Snake with GVF

In order to overcome the small capture range and initialization problems in traditional snakes, Xu proposed a new method by keeping the desirable property of the traditional snake (internal forces) and seeking another solution for external force [51]. The main property of this new external force is; it has a large capture range without blurring the edges of the gradient image and it proposes a solution for the problem with depicting concaved edges. Also, this force solves some initialization problems because with the increase of the capture range, we can now place the snake further away from the object's edge.

In this model everything is the same as the traditional snake model, except the used external force. We used the GVF as our external force.

GVF is defined as the vector field *V(x,y)*=(u(*x,y*), v(*x,y*)),where x and y are coordinates in the image domain, *u(x,y)* and *v(x,y)* are the GVF along horizontal and vertical directions respectively. *V(x,y)* substitutes the external vector force F_{ext} of the traditional snake. The internal forces are defined similar to the original model consisting of elastic and bending forces.

To model *V(x,y), we* first define a gradient image *f(x,y)* of the image *I(x,y)* as:

$$f(x,y) = \left|\nabla(G_\sigma(x,y) * I(x,y))\right|^2 \qquad \textbf{(2.4.18)}$$

V(x,y) is then defined to be a vector field which minimizes the following energy function.

$$E = \iint \left[\mu(u_x^2 + u_y^2 + v_x^2 + v_y^2) + |\nabla f|^2 |V - \nabla f|^2\right] dxdy \qquad \textbf{(2.4 19)}$$

where u_x, u_y, v_x, v_y are the partial derivatives of *u(x,y)* and *v(x,y)* in the *x* and *y* directions respectively, and μ is a regularization parameter.

The ∇f *term* will be very high near the object boundary. In homogenous regions where *I(x,y)* is nearly constant, ∇f is nearly zero .

From equation (2.4.19) we can clearly see that in a non-homogeneous region, where the gradient is strong, the second term dominates (i.e. $V < \nabla f$). Therefore if we want to minimize E, we set $V = \nabla f$. That is exactly what we want because we are happy with the effect ∇f has on snake near the boundary [47]. In contrast, in a homogeneous region the second term is very small (i.e. $\nabla f=0$) and first term, which creates force in this region, dominates.

GVF field can be found by solving the Euler-Lagrange differential equations;

$$\mu\nabla^2 u - (u - f_x)(f_x^2 + f_y^2) = 0$$

$$\mu\nabla^2 v - (v - f_y)(f_x^2 + f_y^2) = 0 \qquad \textbf{(2.4.20)}$$

The desired vector field *V(x,y)* is then the one that satisfies the above Euler equations. In smooth areas, f_x and f_y are zeros and so the vector fields *u(x,y)* and *v(x,y)* are defined by the first term. ∇^2 is the Laplacian operator, therefore the resulting vector field is obtained by diffusing the boundary vectors into smooth areas using Laplacian equation [51].

The greater the distance between a point and an edge, the lesser the force acting on it due to diffusion. The resulting force is the sum of these forces. This represents a competition between all the boundary vectors.

Equation (2.4.20) can be solved by treating *u* and *v* as functions of time, as given in equation 2.4.21. For numerical implementation of equation 2.4.21 see [51].

$$\mu\nabla^2 u(x,y,t) - \left[u(x,y,t) - f_x(x,y)\right].\left[f_x(x,y)^2 + f_y(x,y)^2\right] = u_t(x,y,t)$$

$$\mu\nabla^2 v(x,y,t) - \left[v(x,y,t) - f_y(x,y)\right].\left[f_x(x,y)^2 + f_y(x,y)^2\right] = v_t(x,y,t) \qquad \textbf{(2.4.21)}$$

Let us continue with the example in Figure-2.4.6, where we can see the computed GVF external force for the original image given in Figure-2.4.3(a). In Figure-2.4.6(a), *u(x,y)* gives the force acting on the points in the horizontal direction whereas *v(x,y)* gives the force acting on the point in the vertical direction and the direction of these orthogonal vector components is towards the edges in the concave region and even inside the object and far away from the object. We can see from Figure-2.4.6 that even at the regions where the image is homogeneous, or very far to salient edges, the computed external force is non-zero. However in Figure-2.4.3(c) the external force is non-zero only at salient edges. By comparing the results obtained for external forces in Figure-2.4.6 and Figure-2.4.3 (b), we can simply say that the problem of small capture range in traditional snake model can be solved by using GVF as the external force.

Implementation of the Algorithm

Implementation of snakes with GVF is the same as the traditional snakes except the external force. We again solved Equation-2.4.17 iteratively, the only difference is that we changed the external force with the GVF external force *V(x,y)*. Similar to the traditional model we do not have a stopping condition, and user specifies the number of iterations.

Step-1 and step-3 implementation steps of the traditional model is also valid for GVF Snakes, however in step-2, calculation of the external force is different. In step-2, we first smoothed the image with a 3x3 Gaussian smoothing filter, and then solved equation 2.4.21 iteratively to obtain GVF external force along the horizontal direction *u(x,y),* and GVF external force along the vertical direction *v(x,y).*

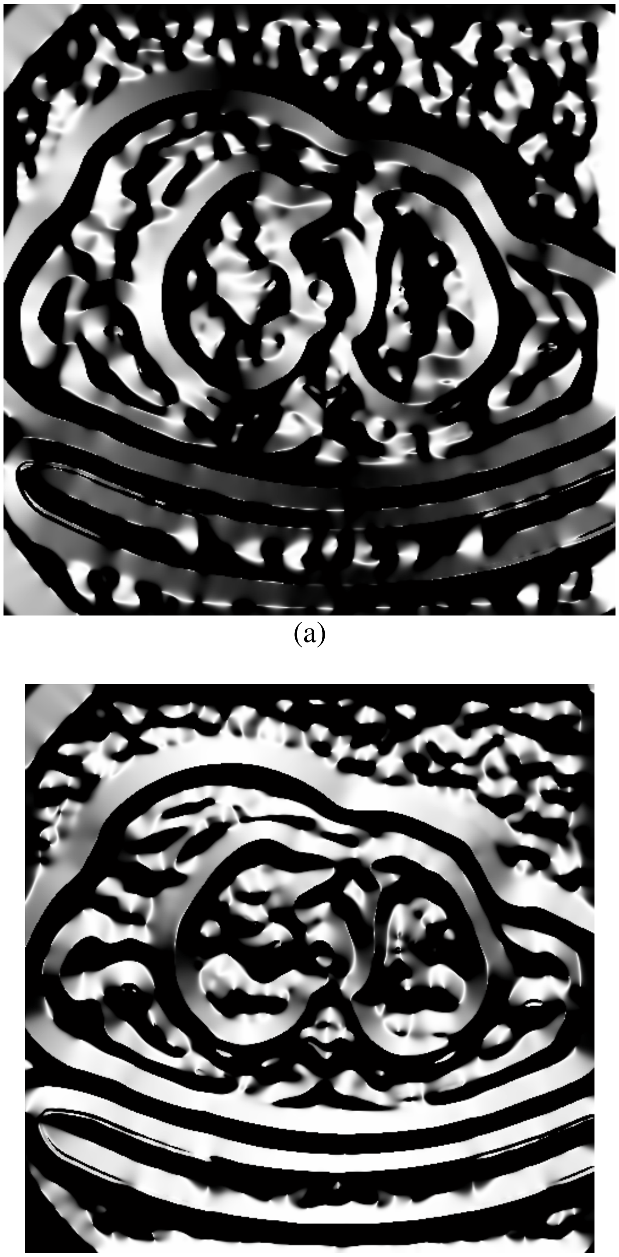
Results for GVF snakes

We applied GVF snakes for the segmentation of heart and lungs. In Figure-2.4.7 (a), we see the initial snake planted inside the heart. The parameters are selected as α =0.2, β=5, γ=1, and the number of iterations is 100. In Figure -2.4.7 (b)), we can see that the resultant snake deformed towards the heart by taking its shape.

While applying GVF snakes for segmentation of lungs, because lungs are not connected to each other, we used two different snakes, and located them initially as in Figure 2.4.8(a). The parameters are selected for both snakes as α =0.2, β=5, γ=1, and the number of iterations is 100. The resultant snakes are in Figure-2.4.8(b). By observing Table-2.4.1 and Figure-2.4.8, we can state that GVF Snakes is not a very practical tool for segmentation of lungs; better results can be obtained by using region growing or hybrid methods. Elapsed time given in Table-2.4.1 is for 100 iterations, therefore it is same for both lungs and heart.

Table 2.4. 1*:* Numerical Results for GVF Snakes

Segmented Tissues	Negative Rate Metric (NR)	Elapsed Time (sec)
Heart	0.09819	85.6
Lungs	0.26234	85.6



(a)

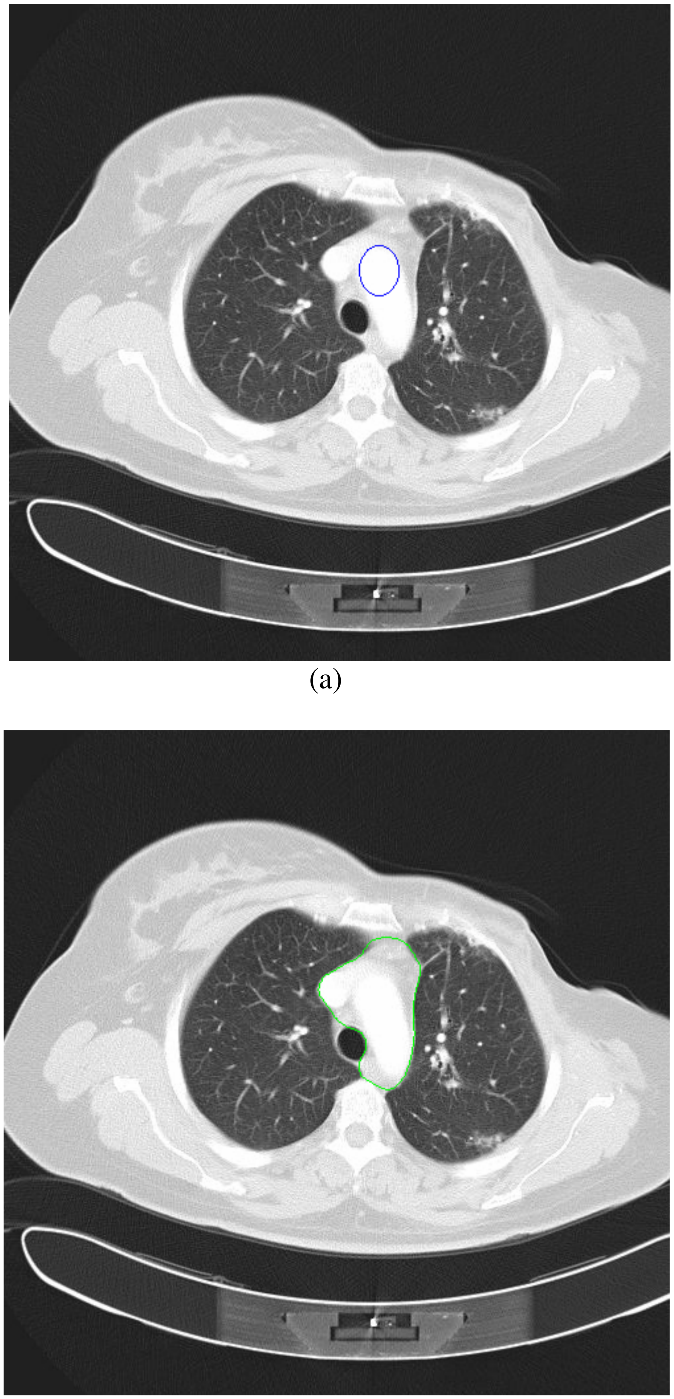
(b)

Figure 2.4.6: GVF external force, (a) GVF external force along the horizontal direction *u(x,y)* (b) GVF external force along the vertical direction *v(x,y)*

2.4.2.2.1. Drawbacks of Parametric Snake Models:

Using parametric snake models (traditional and GVF) for segmentation of bones, muscle and fat tissue is not practical. In CT torso images, bones are located separately in the image and because the snake can not split or merge during deformation, segmenting all the regions of bones is not possible with one snake. This can be achieved by using many snakes, but the initialization of these snakes is a very complex task, in which the user should locate each snake very carefully to avoid converging to the wrong edges. On the other hand, the bones can be segmented much more easily and efficiently using other segmentation methods.

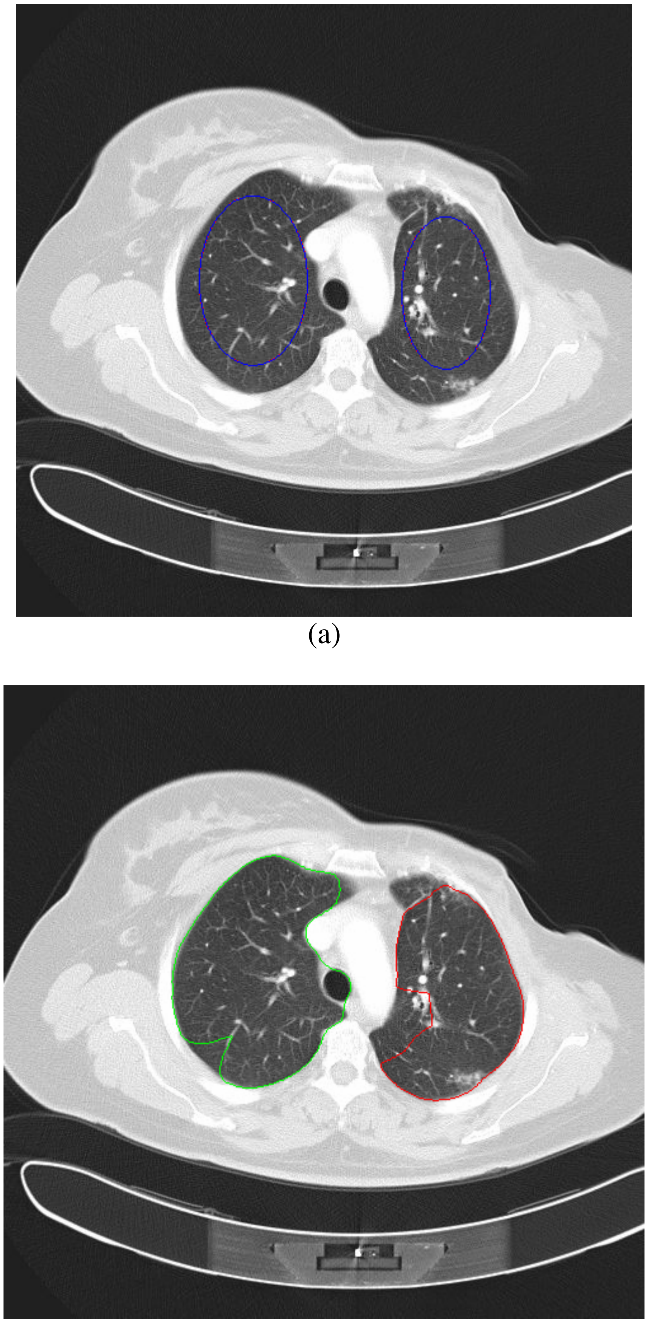
Similarly using parametric snakes for segmentation of fat or muscle tissue is also impractical. In the case of muscle tissue, one must insert many snakes around related regions. Strictly talking for the image in Figure-2.4.3(a), one must insert at least the contours given in Figure-2.4.9, to have a solution. Segmentation of muscle and fat tissues can be done much more easily with region growing or with hybrid methods that we will discuss in Chapter-3.



(a)

(b)

Figure 2.4.7*:* GVF Snake result for Heart, (a) initial snake, (b) resultant snake



(a)

(b)

Figure 2.4.8: GVF Snake result for lungs, (a) initial snakes,(b) resultant snakes

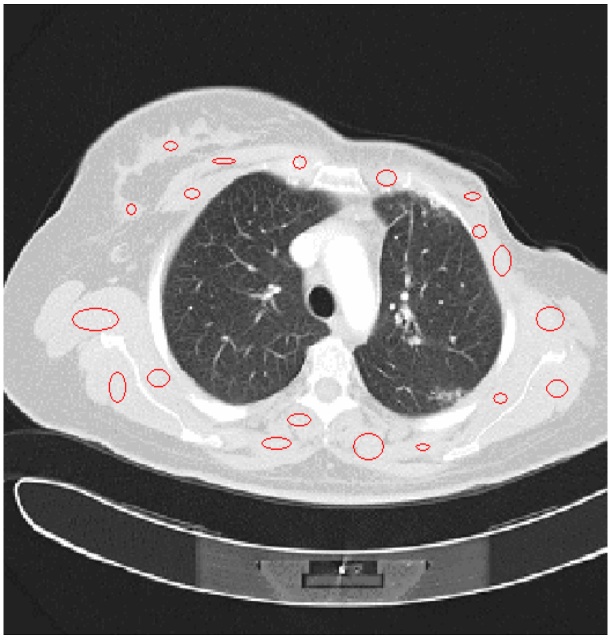


Figure 2.4.9: Initial contours for segmentation of muscle tissue

2.4.2.2.1.1. Drawbacks of GVF Snakes:

The small capture range and poor detection of boundary concavities problems of traditional snakes is solved by using GVF, however GVF snakes still suffers from the following drawbacks:

1. **Sensitivity to parameters:** As in the case of traditional snakes, there is not a rule for choosing the parametersα, β and γ.
2. **Speed:** Solving the Euler Equation (2.4.21) to find the GVF field components is computationally a very expensive process. Therefore it becomes almost impossible to implement GVF in real time applications.

The external forces computed in both the traditional snake model and the GVF snake model depend on the image gradient, and they are classified as edge-stopping approaches, because they make the snake to move towards image edges. Since the external forces depend on the image gradient, only objects with edges, defined by the gradient image can be segmented.

2.4.3. Geometric Deformable Models

Geometric deformable models proposed by Caselles [52] and Malladi [53] provide a solution to address the primary limitations of parametric deformable models. These models are based on curve evolution theory [54,55] and the level set method [56,57]. In geometric deformable models, curves are deforming only because of geometric measures, resulting in a deformation that is independent of the parameterization. Similar to parametric deformable models, the evolution is coupled with the image data to recover object boundaries. Since the deformation of curves is independent of the parameterization, the deforming curves can be represented implicitly as a level set of a higher-dimensional function. As a result, topology changes can be handled automatically [58].

2.4.3.1.Level Set Methods

In this sub-section, basic terminology of the level set methods is given. In the level set method, the curve is represented implicitly as a level set of a 2D scalar function, which is defined on the same domain as the image. The level set is defined as the set of points that have the same function value. Figure-2.4.10 shows an example of embedding a curve as a zero-level set [58]. The zero-level set function is the level set function that has value of zero.

In level-set methods the deformation of the curve is done by updating the level set function at fixed coordinates through time. The propagating contour can automatically change topology in geometric models (merge or split) while the level set function still remains a valid function, which is not possible in parametric models.

Given a level set function $\phi(x, y, t)$ with a zero level set *X(s,t),* where *s=(x,y)* and t is the time*,* we can write the following equation,

$$\phi(X(s,t),t) = 0 \qquad \textbf{(2.4.22)}$$

we assume that the level set function and the zero level set function has the following relationship which is also demonstrated in Figure-2.4.11.

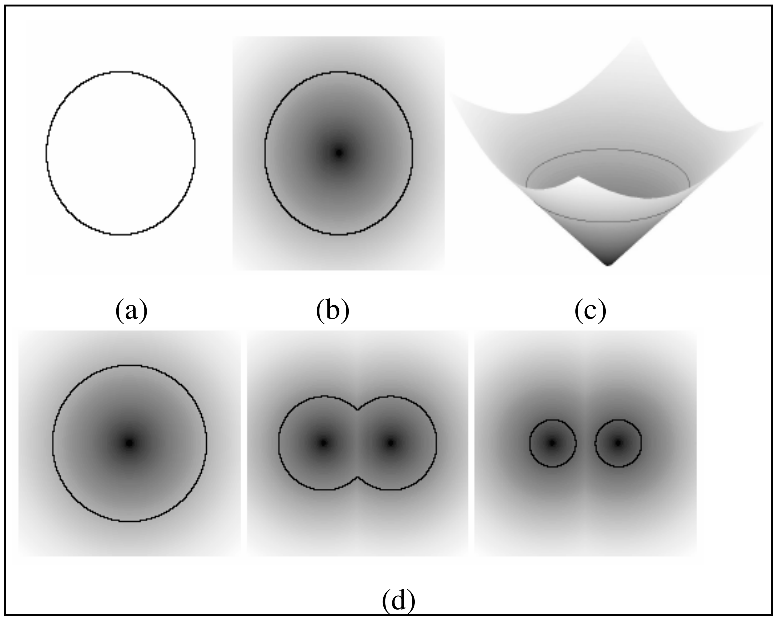


Figure 2.4.10: Embedding a zero-level curve (a) A single curve. (b) The level set function where the curve is embedded as the zero level set (in black). (c) The height map of the level set function with its zero level set depicted in black.(d) From left to right, the zero level set splits into two curves while the level set function still remains a valid function (Figure-2.4.2 is taken from [58]).

$$\phi(x, y, t) \quad \left(\begin{array}{l} < 0, \text{if } \ (x,y) \text{ outside the } \ X(s,t) \\ = \ 0, \text{if } \ (x,y) \text{ on the } \ X(s,t) \\ > 0, \text{if } \ (x,y) \text{ inside the } \ X(s,t) \end{array} \right)$$

The outward unit normal to the level set curve is given by,

$$N = -\frac{\nabla \phi}{|\nabla \phi|} \qquad \textbf{(2.4.23)}$$

and the curvature equals,

$$\kappa = -\nabla . N = \nabla . \frac{\nabla \phi}{|\nabla \phi|} \quad \textbf{(2.4.24)}$$

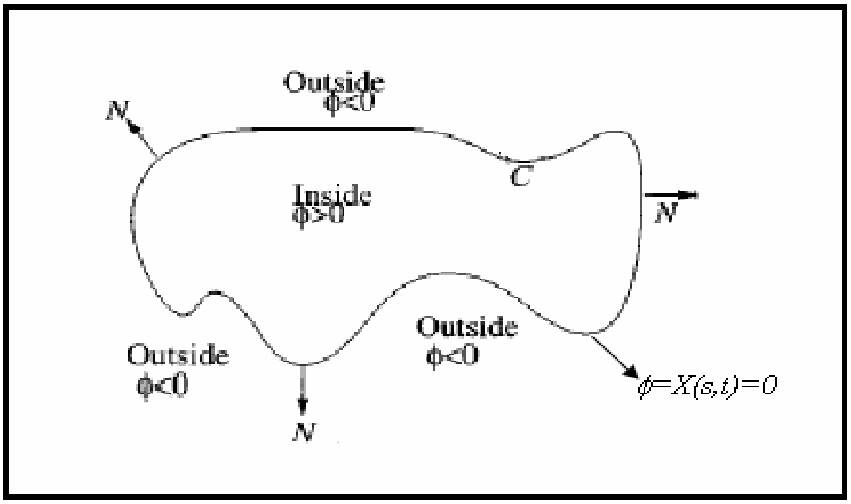


Figure 2.4.11: The image domain divided into two regions by the zero level-set curve X, on which ϕ=0.

2.4.3.2.Active Contours without Edges

Because all the active contour models we have studied so far rely on the image gradient $|\nabla I_0|$, where the original image is given as I_0, these models can detect only object boundaries defined by the image gradient. In practice, the discrete gradients are bounded and the gradient based stopping function never reaches zero on the edges. Therefore the curve may pass through the boundaries.

In order to handle these problems a different active contour model is proposed, which does not require a stopping function that depends on the image gradient. We used this active contour model, namely active contours without edges method, where the stopping term is based on Mumford–Shah segmentation techniques [59].

Active contours without edges is a geometric active contour model that can detect objects whose boundaries are not necessarily defined by a gradient. Similar to previously defined active contour models, an energy function is minimized. However, in this case the stopping term does not depend on the gradient of the image, as in the classical active contour models, but is instead related to a particular segmentation of the image.

Suppose an evolving curve C is in image domain Ω, as the boundary of an open subset ω of Ω $(\omega \subset \Omega)$. Because the C is the boundary of the open subset ω (C=$\partial\omega$), $\Omega \setminus \omega$ is the outside of curve C and ω denotes inside of curve C [59].

This model depends on the minimization of energy, and the basic idea is as follows, assume an image $I_0(x, y)$ that has two regions with approximately constant intensities with values of, c$_1$, and c$_2$ as in Figure-2.4.13. The object to be detected in $I_0(x, y)$ is represented by the region with the value c$_1$, and its boundary is given with C_0 .In the below equation, c$_1$ and c$_2$ are the mean values of $I_0(x, y)$ inside and outside the C respectively, and C_0 is the boundary of the object,

$$F_1(C) + F_2(C) = \int_{inside(C)} \left| I_0(x, y) - c_1 \right|^2 dxdy + \int_{outside(C)} \left| I_0(x, y) - c_2 \right|^2 dxdy \qquad \textbf{(2.4.25)}$$

where $F_1(C)$ and $F_2(C)$ are the first and second terms in the integrand respectively. We can see all possible results of equation 2.4.25 in Figure-2.4.12. The case where the curve C exactly fit on the boundary of the object C$_0$ is the case where the mean values of $I_0(x, y)$ inside and outside the C are zero (see Figure-2.4.13(d) and equation 2.4.26.).

$$F_1(C_0) + F_2(C_0) = 0 \qquad \textbf{(2.4.26)}$$

In this active contour model, additional regularizing terms are added, like the length of the curve, and (or) the area of the region inside, and the new the energy function to be minimized is as below,

$$F(c_1,c_2,C)=\mu.L(C)+v.A(C)+\lambda_1\int_{in(C)}\left|I_0(x,y)-c_1\right|^2 dxdy+\lambda_2\int_{out(C)}\left|I_0(x,y)-c_2\right|^2 dxdy \quad \textbf{(2.4.27)}$$

where, *L(C)* is the length of the curve C, *A(C)* is the area inside C, and weighting terms $\lambda_1,\lambda_2 \geq 0$ and v, μ, >0, generally $\lambda_1,\lambda_2 = 1$, $\mu = 0.1$ and v=0 [59].

Level Set Formulation of the Model

In the level set method the curve C, $C \subset \Omega$, is represented by the zero level set of a Lipschitz function ϕ, such that [59] :

$$\begin{aligned}&C = \partial w = \{(x, y) \in \Omega : \phi(x, y) = 0\}\\&inside(C) = w = \{(x, y) \in \Omega : \phi(x, y) > 0\}\\&outside(C) = \Omega / w = \{(x, y) \in \Omega : \phi(x, y) < 0\}\end{aligned}$$

Using the Heaviside (unit-step function) function H, and the one-dimensional Dirac function δ_0, is defined as below,

$$H(z) = \langle \begin{matrix} 0 \text{ if } z < 0 \\ 1 \text{ if } z \geq 0 \end{matrix} \quad \textbf{(2.4.28)}$$

$$\delta_0(z) = \frac{d}{dz}H(z) \quad \textbf{(2.4.29)}$$

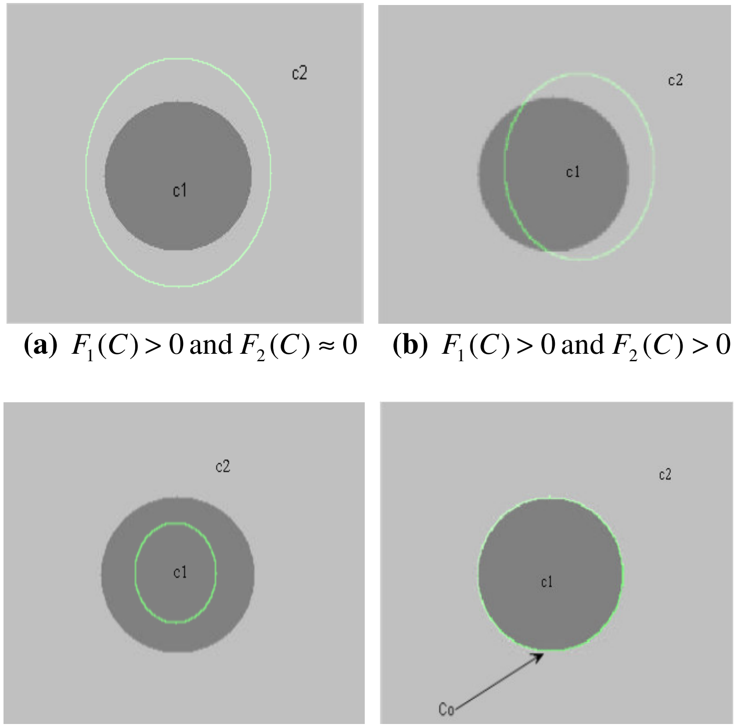
We can define the length of C and the area inside C in 2.4.28 as,

$$L(C) = Length(\phi = 0) = \int_{\Omega} \nabla\left|H(\varphi(x, y))\right|dxdy = \int_{\Omega} \delta_0(\phi(x, y))\left|\nabla\phi(x, y)\right|dxdy \quad \textbf{(2.4.30)}$$

$$A(C) = Area(\phi > 0) = \int_{\Omega} H(\varphi(x, y))dxdy \quad \textbf{(2.4.31)}$$

and $F_1(C)$ and $F_2(C)$ can be rewritten using Heaviside functions as,

$$\begin{aligned} F_1(C) &= \int_{inside(C)} \left|I_0(x,y)-c_1\right|^2 dxdy = \int_{\Omega} \left|I_0(x,y)-c_1\right|^2 H(\phi(x,y)dxdy. \\ F_2(C) &= \int_{outisde(C)} \left|I_0(x,y)-c_1\right|^2 dxdy = \int_{\Omega} \left|I_0(x,y)-c_2\right|^2 (1-H(\phi(x,y))dxdy. \end{aligned} \qquad \textbf{(2.4.32)}$$



(a) $F_1(C) > 0$ and $F_2(C) \approx 0$ **(b)** $F_1(C) > 0$ and $F_2(C) > 0$

(c) $F_1(C) \approx 0$ and $F_2(C) > 0$ **(d)** $F_1(C) = 0$ and $F_2(C) = 0$

Figure 2.4.12: All possible cases in the position of the curve

then the energy function given in equation 2.4.27 can be rewritten as,

$$\begin{aligned} F(c_1, c_1, C) &= \mu.\int_{\Omega} \delta_0(\phi(x,y))\left|\nabla\phi(x,y)\right| dxdy + v.\int_{\Omega} H(\varphi(x,y))dxdy + \\ &\lambda_1 \int_{\Omega} \left|I_0(x,y)-c_1\right|^2 H(\phi(x,y))dxdy + \lambda_2 \int_{\Omega} \left|I_0(x,y)-c_2\right|^2 (1-H(\phi(x,y))dxdy. \end{aligned} \qquad \textbf{(2.4.33)}$$

The solution image *I* with two distinct regions can be rewritten as

$$I(x,y) = c_1 H(\phi(x,y)) + c_2(1 - H(\phi(x,y)). \qquad \textbf{(2.4.34)}$$

We look for the best approximation *I* of *I₀*, as a function taking only two values inside and outside the curve. The constants c$_1$ and c$_2$ are the averages of I_0 in $\phi \geq 0$ and $\phi < 0$ respectively, and they are given as [59]:

$$c_1(\phi) = \frac{\int_\Omega I_0(x,y) H(\phi(x,y) dxdy}{\int_\Omega H(\phi(x,y) dxdy}.$$

$$c_2(\phi) = \frac{\int_\Omega I_0(x,y)(1 - H(\phi(x,y)) dxdy}{\int_\Omega (1 - H(\phi(x,y)) dxdy}. \qquad \textbf{(2.4.35)}$$

Finally we can say that,

$$c_1(\phi) = mean(I_0)\ in\ \phi \geq 0$$
$$c_2(\phi) = mean(I_0)\ in\ \phi < 0$$

By keeping c_1 and c_2 fixed and minimizing F with respect to ϕ, using an artificial time ,$t \geq 0$ Euler–Lagrange equation for ϕ is obtained as follows:

$$\frac{\partial \phi}{\partial t} = \delta_\varepsilon(\phi)\left[\mu \nabla . \left(\frac{\nabla \phi}{|\nabla \phi|}\right) - v - \lambda_1 (I_0 - c_1)^2 + \lambda_2 (I_0 - c_2)^2\right] = 0 \qquad \textbf{(2.4.36)}$$

Because the Dirac delta function and Heaviside function are not implementable, we used approximations for the $\delta_\varepsilon, H_\varepsilon$ respectively [59]:

$$\delta_\varepsilon(\phi) = \frac{1}{\pi}\frac{\varepsilon}{\varepsilon^2+\phi^2}$$

$$H_\varepsilon(\phi) = \frac{1}{2}\left(1+\frac{2}{\pi}\arctan(\frac{\phi}{\varepsilon})\right) \qquad \textbf{(2.4.37)}$$

where zero level-set or the initial contour is $\phi(t=0,x,y)=\phi_0(x,y)$. The discretization of (2.4.36) is given in Appendix-C.2.

Results

- **Simulation Results**

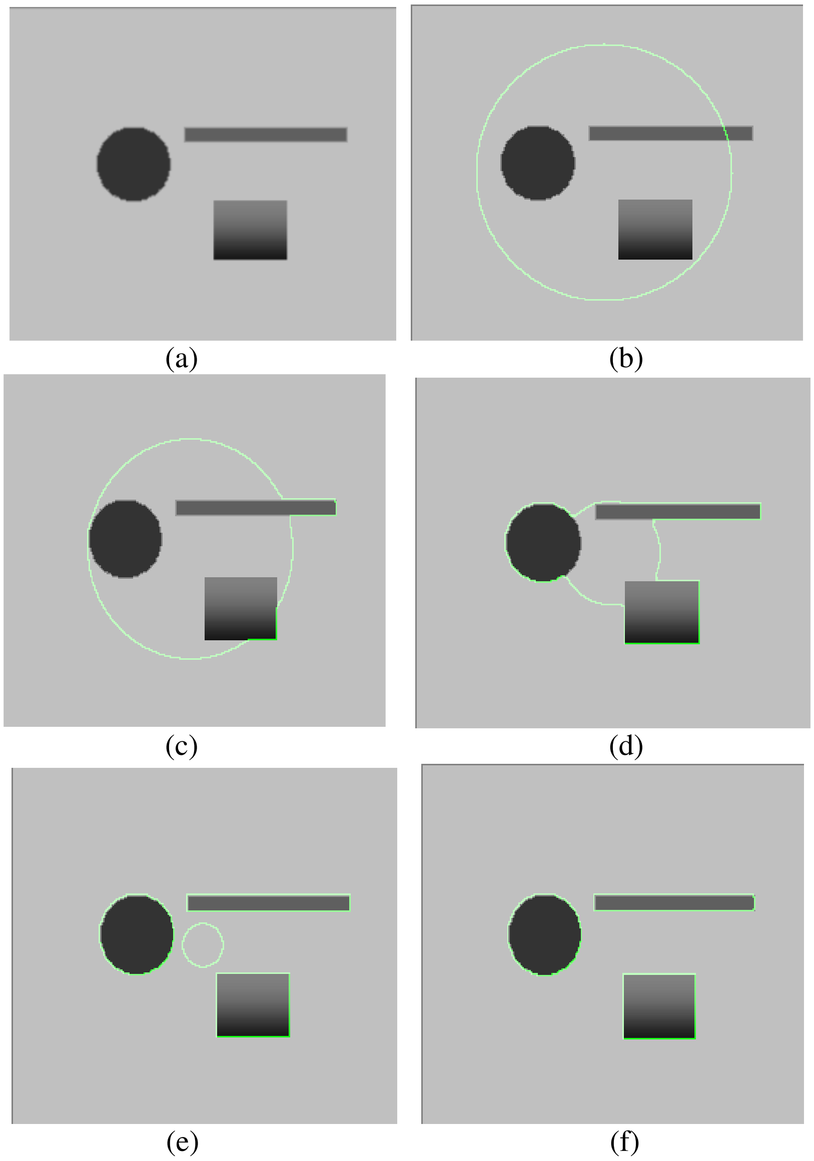
We first applied active contours without edges method to a test image in Figure-2.4.16(a), with parameters, $\lambda_1, \lambda_2 = 1$, step size in time $\Delta t = 0.1$, step size in space *h*=1, and μ =0.01*255^2. If as many objects as possible have to be detected, the value of weight parameter μ must be chosen small (0.0001*255^2) , on the other hand for images in which we only wish to segment larger objects, μ must be chosen high (0.01*255^2).

We can see from Figure-2.4.13 that using active contours without edges, one can perfectly perform segmentation of unconnected objects with one initial contour. Because parametric active contours can not split, or two parametric snakes can not merge, obtaining the same result (Figure-2.4.14(h)) by using one parametric active contour (snakes with GVF, or with traditional snakes) is not possible. In Figure-2.4.15 , we applied GVF snakes to the same test image, with 200 iterations, step size in time *h*=1, elasticity coefficient α=0.5, bending coefficient β=0.005 and damping coefficient γ=1. We obtained the resulting contour in Figure-2.4.15 (b). It is obvious that the result is not comparable with Figure-2.4.14(h). A comparable result can be obtained by using three parametric snakes located very carefully. On the other hand, because geometric active contours without edges can propagate in the image domain without the effect of an external force, they can split and merge, location of the initial contour is not very critical.

- **CT Image Results**

We applied active contours without edges to the CT image of torso shown in Figure-2.4.3(a) with an initial contour in Figure-2.4.16(a). It can be seen from Figure 2.4.16(c) that the resultant contour separates the image into two different disjoint regions, where the interior part of the contour contains pixels belong to both muscle, fat, lungs, heart and bones, and the exterior part of the contour is the background. Result of segmentation by using one level set function is similar to bi-level thresholding, where the foreground is separated from the background. The original idea in the level set method is to use the sign of a given function to separate the given domain into two disjoint regions, and the use the continuity of the level set function near its zero to define the boundary of these disjoint regions.

It is obvious that it is not possible to perform segmentation of fat, muscle, heart, bones or lungs separately using one level set function because the image is separated to only two disjoint regions. On the other hand, by using two or more level set functions, the image can be divided four or more disjoint regions.

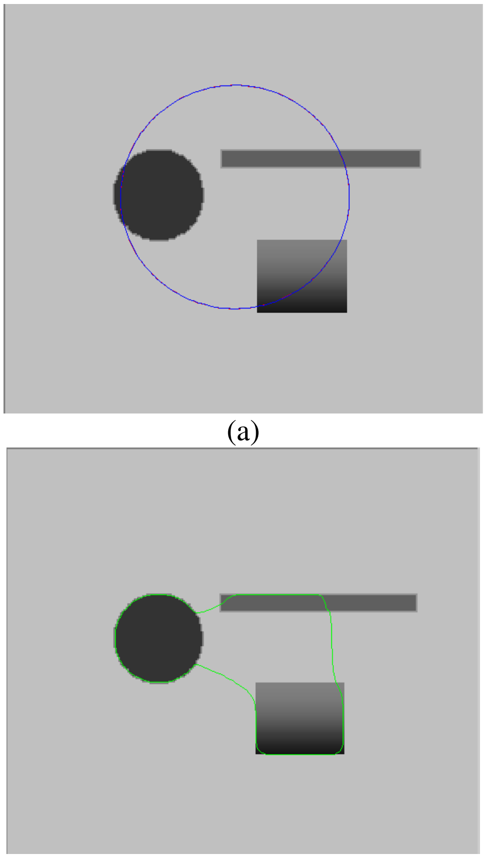


(a) (b)

(c) (d)

(e) (f)

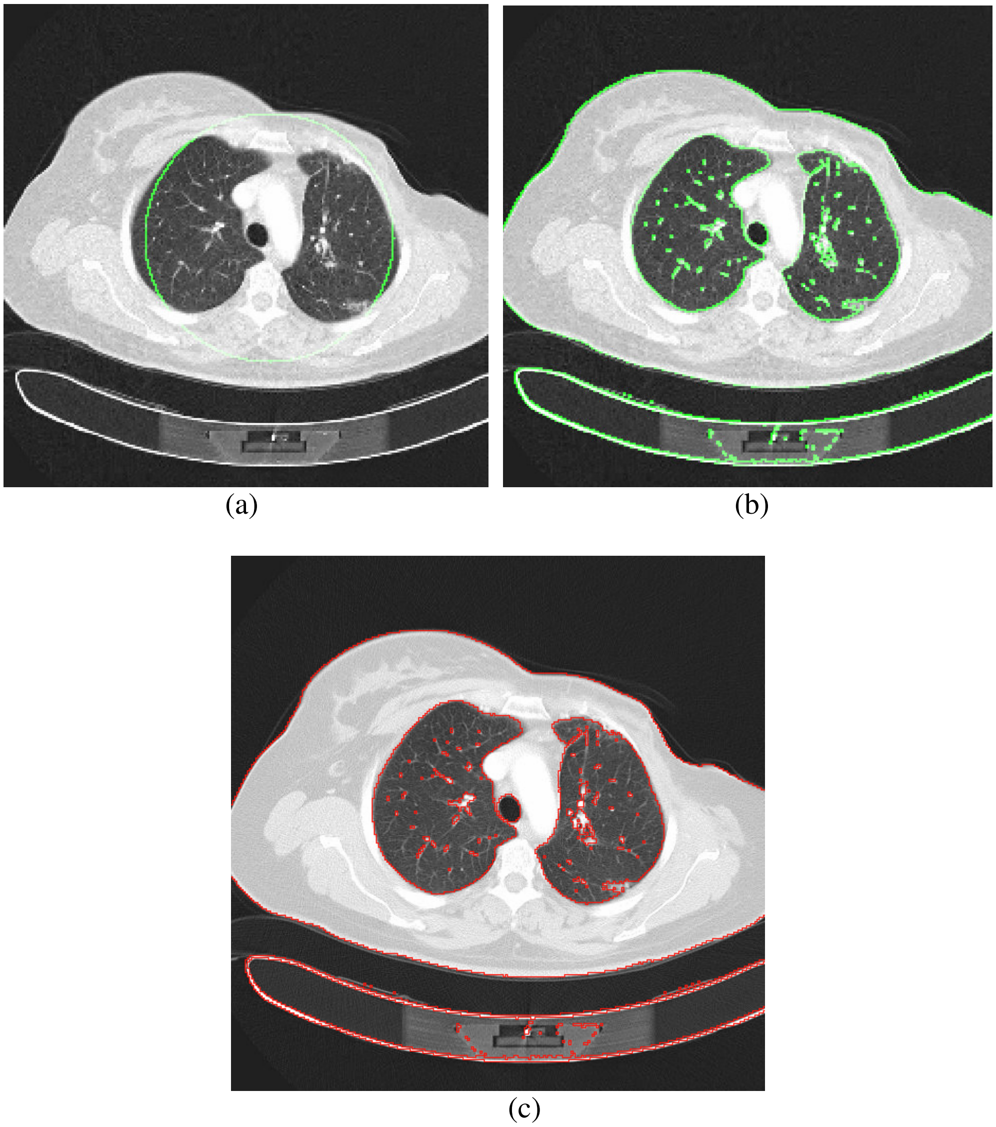
Figure 2.4.13: Segmentation of the test image with active contours without edges, (a) original image, (b) the initial contour, (c) result of segmentation after 100 iterations , (d) result of segmentation after 300 iterations,(e) result of segmentation after 500 iterations, (f) result of segmentation after 600 iterations,



(a)

(b)

Figure 2.4.14: Segmentation of the test image using Snakes with GVF, (a)initial active contour, (b) resultant active contour.


(a)

(b)

(c)

Figure 2.4.15: Deformation of the contour, (a) smoothed image with initial contour, (b) result after 50 iterations, (c) result of segmentation after 100 iterations

Segmentation with two Level Set Functions:

Level set method can be extended to multiple phase segmentation by using multiple level set functions. N level set functions can be used for representing up to 2^N phases.

For example, with two level set functions, it is possible to locate four distinct regions, where the four distinct region can be defined as;

$$\Omega_{11} = \{x, y : \phi_1 \geq 0 \, \text{and} \, \phi_2 \geq 0\}$$
$$\Omega_{10} = \{x, y : \phi_1 \geq 0 \, \text{and} \, \phi_2 < 0\}$$
$$\Omega_{01} = \{x, y : \phi_1 < 0 \, \text{and} \, \phi_2 \geq 0\}$$
$$\Omega_{00} = \{x, y : \phi_1 < 0 \, \text{and} \, \phi_2 < 0\}$$

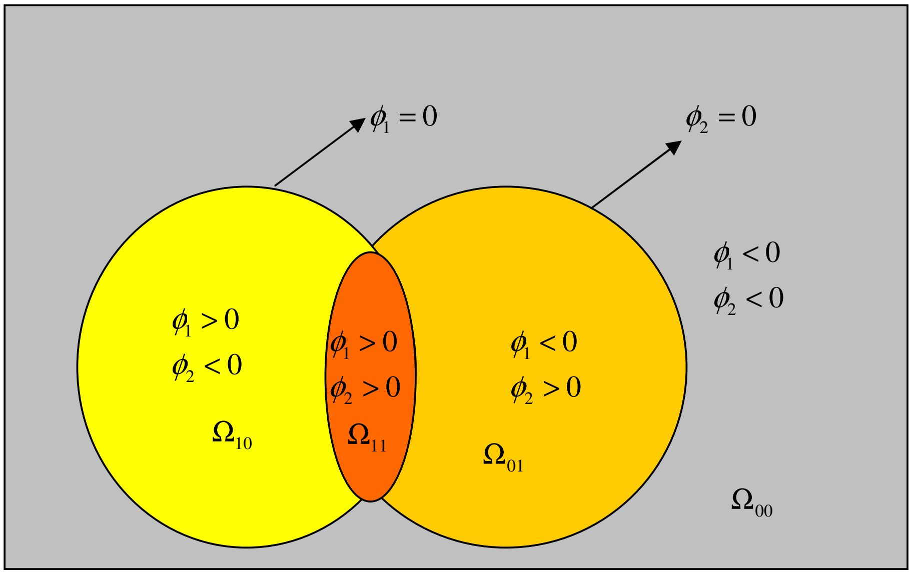


Figure 2.4.16: Four distinct regions can be detected using two level set functions.

where the image domain $\Omega = \Omega_{11} \cup \Omega_{10} \cup \Omega_{01} \cup \Omega_{00}$, and $X_1$and $X_2$are the curves where $\phi_1 = 0$ and $\phi_2 = 0$. The resultant regions after segmentation using multi-level sets will not be overlap and each pixel in the image will belong to only one phase. The energy function to be minimized is given as:

$$\begin{aligned} F(\phi_1,\phi_2,c_{ij}) = {} & \mu_1\int_\Omega \delta_0(\phi_1(x,y))\left|\nabla\phi_1(x,y)\right|dxdy + \mu_2\int_\Omega \delta_0(\phi_2(x,y))\left|\nabla\phi_2(x,y)\right|dxdy \\ & + v_1\int_\Omega H(\varphi_1(x,y))dxdy + v_2\int_\Omega H(\varphi_2(x,y))dxdy \\ & + \lambda\int_\Omega \left|I_0(x,y)-c_{11}\right|^2 H(\phi_1(x,y))H(\phi_2(x,y))dxdy \\ & + \lambda\int_\Omega \left|I_0(x,y)-c_{10}\right|^2 H(\phi_1(x,y))(1-H(\phi_2(x,y))dxdy \\ & + \lambda\int_\Omega \left|I_0(x,y)-c_{01}\right|^2 (1-H(\phi_1(x,y))H(\phi_2(x,y))dxdy \\ & + \lambda\int_\Omega \left|I_0(x,y)-c_{00}\right|^2 (1-H(\phi_1(x,y))(1-H(\phi_2(x,y)). \end{aligned} \tag{2.4.38}$$

Where $\mu_1,\mu_{2,}v_1,v_2,\lambda$ are constants, $c_{11,}c_{01},c_{10},$and c_{00} are averages of I_0 in regions $\Omega_{11},\Omega_{10},\Omega_{01}$and Ω_{00} respectively,and can be defined as:

$$c_{11} = \frac{\int_\Omega I_0(x,y)H(\phi_1(x,y)H(\phi_2(x,y)dxdy}{\int_\Omega H(\phi_1(x,y))H(\phi_2(x,y))dxdy}.$$

$$c_{10} = \frac{\int_\Omega I_0(x,y)H(\phi_1(x,y)(1-H(\phi_2(x,y))dxdy}{\int_\Omega H(\phi_1(x,y)(1-H(\phi_2(x,y))dxdy}$$

$$\begin{aligned} c_{01} &= \frac{\int_\Omega I_0(x,y)(1-H(\phi_1(x,y))H(\phi_2(x,y))dxdy}{\int_\Omega (1-H(\phi_1(x,y))H(\phi_2(x,y))dxdy} \\ c_{00} &= \frac{\int_\Omega I_0(x,y)(1-H(\phi_1(x,y))(1-H(\phi_2(x,y)))dxdy}{\int_\Omega (1-H(\phi_1(x,y))(1-H(\phi_2(x,y)))dxdy} \end{aligned} \tag{2.4.39}$$

After segmentation with two level set functions, the segmented image is approximated by a piecewise constant function which can take four value.

$$I(x,y) = c_{11}H(\phi_1(x,y))H(\phi_2(x,y)) + c_{10}H(\phi_1(x,y))(1-H(\phi_2(x,y)))$$
$$+ c_{01}(1-H(\phi_1(x,y)))H(\phi_2(x,y)) + c_{00}(1-H(\phi_1(x,y)))(1-H(\phi_2(x,y))) \quad \textbf{(2.4.40)}$$

By keeping $c_{11,}c_{01}, c_{10}$, and c_{00} fixed, and minimizing F with respect to ϕ_1 and ϕ_2 , with an artificial time $t \geq 0$ Euler–Lagrange equations for ϕ_1 and ϕ_2 are obtained as follows:

$$\frac{\partial \phi_1}{\partial t} = \delta_\varepsilon(\phi_1)\mu_1 \nabla . \left(\frac{\nabla \phi_1}{|\nabla \phi_1|} \right) - \delta_\varepsilon(\phi_1)v_1 - \lambda \delta_\varepsilon(\phi_1)\left[(I_0 - c_{11})^2 - (I_0 - c_{01})^2 \right] H(\phi_2) \quad \textbf{(2.4.41)}$$
$$- \lambda \delta_\varepsilon(\phi_1)\left[(I_0 - c_{10})^2 - (I_0 - c_{00})^2 \right](1 - H(\phi_2)) = 0$$

$$\frac{\partial \phi_2}{\partial t} = \delta_\varepsilon(\phi_2)\mu_2 \nabla . \left(\frac{\nabla \phi_2}{|\nabla \phi_2|} \right) - \delta_\varepsilon(\phi_2)v_2 - \lambda \delta_\varepsilon(\phi_2)\left[(I_0 - c_{11})^2 - (I_0 - c_{10})^2 \right] H(\phi_1) \quad \textbf{(2.4.42)}$$
$$- \lambda \delta_\varepsilon(\phi_2)\left[(I_0 - c_{01})^2 - (I_0 - c_{00})^2 \right](1 - H(\phi_1)) = 0$$

Numerical implementation of equations 2.3.41 and 2.4.42 are similar to 2.4.36.

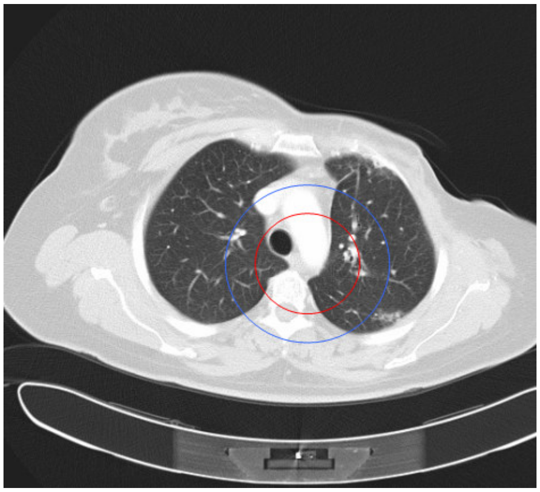
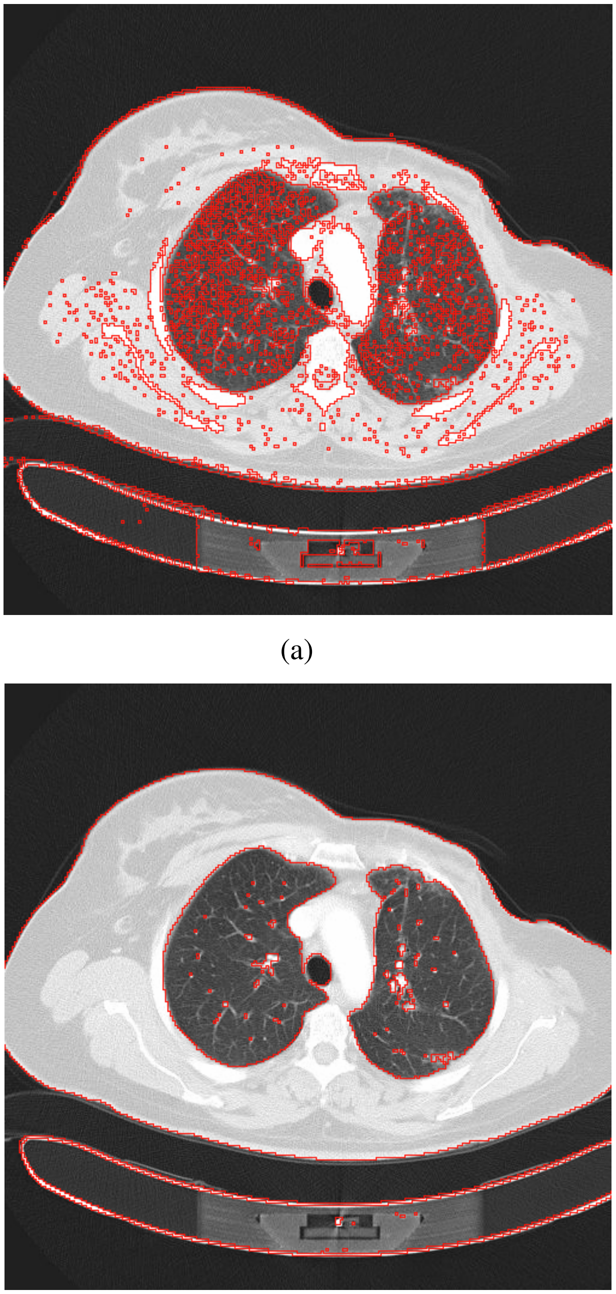


Figure 2.4.17: Initial positions of two level set functions



(a)

(b)

Figure 2.4.18: (a) resultant zero level set function ϕ_1 (b) resultant zero level set function ϕ_2

We applied two level set functions to the original image given in Figure-2.4.15(a) , initial positions of two level set functions are given in Figure-2.4.17 and the resultant zero level set functions after 400 iterations is given in Figure-2.4.18.

The four disjoint regions obtained from resultant level set functions are given in Figure-2.4.20, where each of the four disjoint regions are represented by a different color.

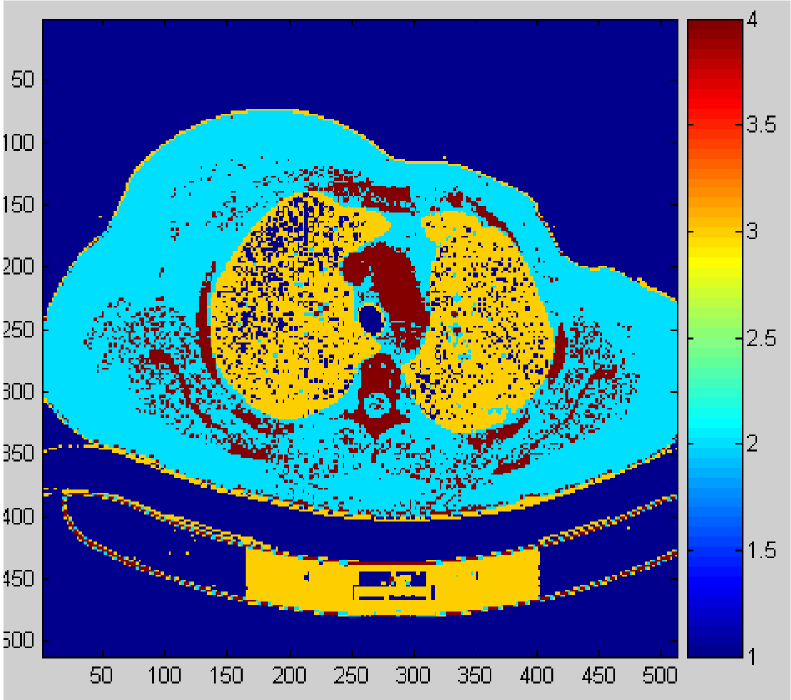


Figure 2.4.19: Result of segmentation with two level set functions after 400 iterations.

In Figure-2.4.20 to Figure-2.4.22, four phases or disjoint regions are given as binary images, where white pixels represent segmented pixels and black ones represent background. We can see from Figure-2.4.19 and Figure-2.4.21 that by using two level set functions, we can not perform segmentation of heart, fat and muscle tissue separately because pixels belonging to these tissues are classified in phase-2 image. Pixels belonging to lungs are classified in both phase-1 and phase-3 images. Finally, phase-4 image consist of pixels belong to bones, heart and scattered pixels belong to muscle tissue.

By using three level set functions we can divide the image into eight disjoint regions. Segmentation with three level set functions is given in next section.

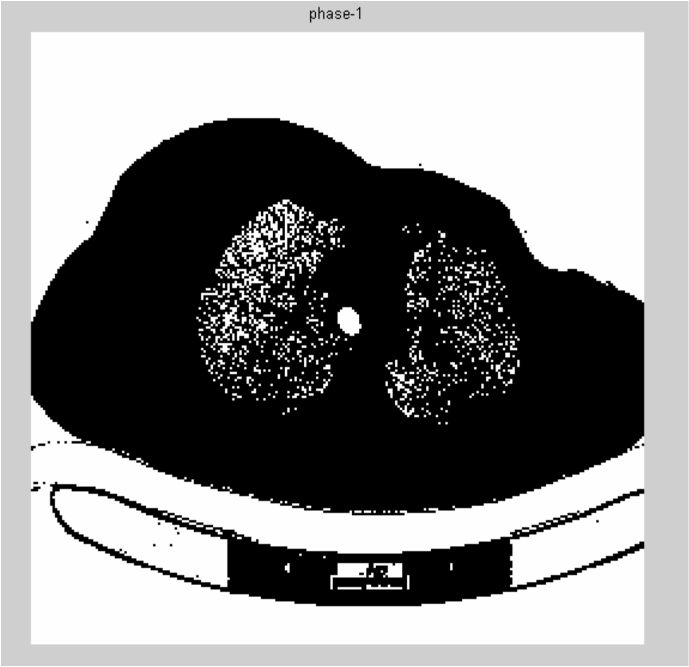


Figure 2.4.20: Phase-1 image obtained two level set functions after 400 iterations.

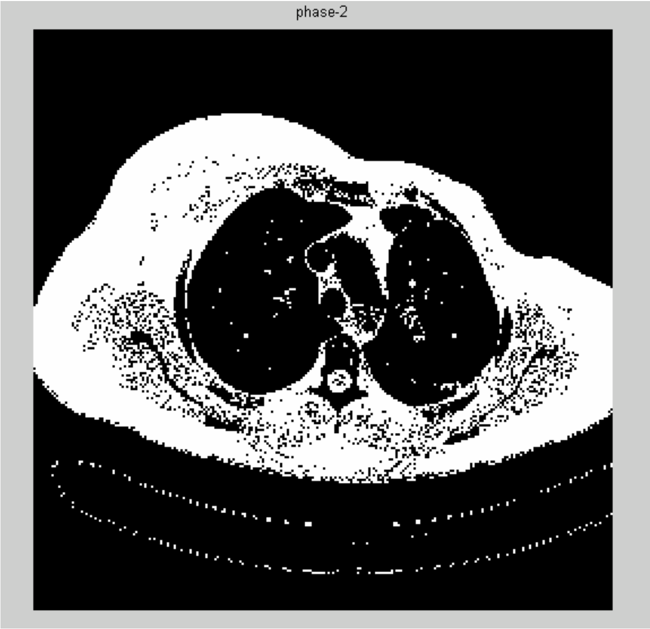


Figure 2.4.21: Phase-2 image obtained two level set functions after 400 iterations

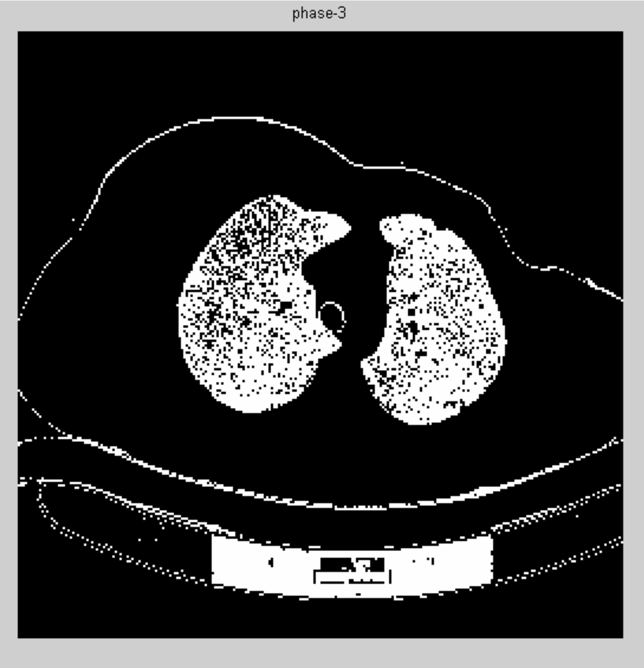


Figure 2.4.22: Phase-3 image obtained two level set functions after 400 iterations

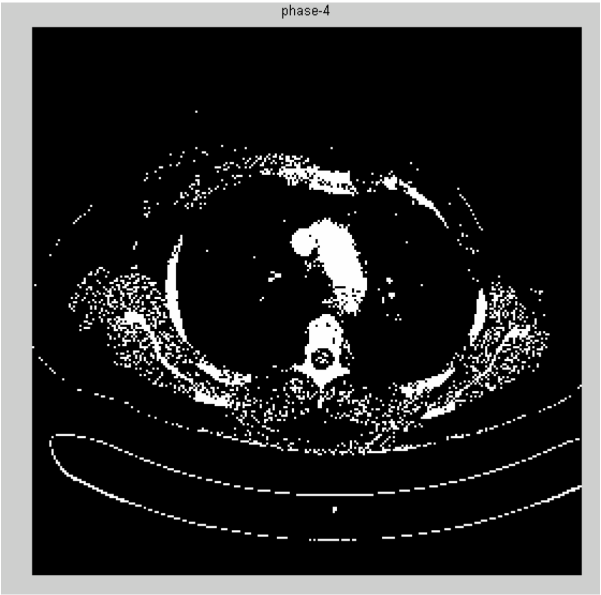


Figure 2.4.23: Phase-4 image obtained two level set functions after 400 iterations

Segmentation with three Level Set Functions:

By using three level set functions, it is possible to locate eight distinct regions.

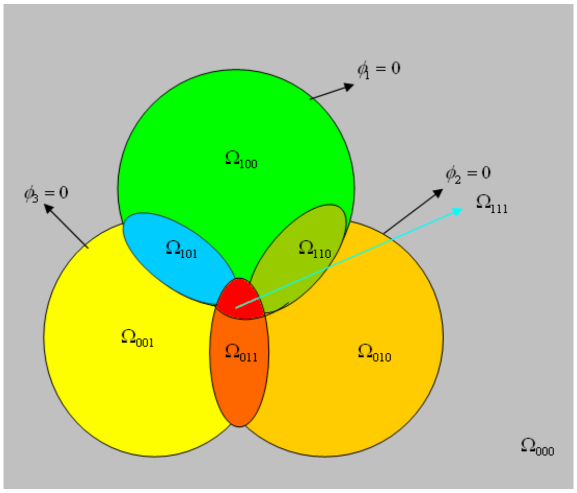


Figure 2.4.24: Eight distinct regions can be detected using three level set functions.

where the eight distinct region can be defined as;

$$
\begin{aligned}
\Omega_{111} &= \{x, y : \phi_1 \geq 0 \,\mathrm{and}\, \phi_2 \geq 0 \,\mathrm{and}\, \phi_3 \geq 0\} \\
\Omega_{110} &= \{x, y : \phi_1 \geq 0 \,\mathrm{and}\, \phi_2 \geq 0 \,\mathrm{and}\, \phi_3 < 0\} \\
\Omega_{101} &= \{x, y : \phi_1 \geq 0 \,\mathrm{and}\, \phi_2 < 0 \,\mathrm{and}\, \phi_3 \geq 0\} \\
\Omega_{100} &= \{x, y : \phi_1 \geq 0 \,\mathrm{and}\, \phi_2 < 0 \,\mathrm{and}\, \phi_3 < 0\} \\
\Omega_{011} &= \{x, y : \phi_1 < 0 \,\mathrm{and}\, \phi_2 \geq 0 \,\mathrm{and}\, \phi_3 \geq 0\} \\
\Omega_{010} &= \{x, y : \phi_1 < 0 \,\mathrm{and}\, \phi_2 \geq 0 \,\mathrm{and}\, \phi_3 < 0\} \\
\Omega_{001} &= \{x, y : \phi_1 < 0 \,\mathrm{and}\, \phi_2 < 0 \,\mathrm{and}\, \phi_3 \geq 0\} \\
\Omega_{000} &= \{x, y : \phi_1 < 0 \,\mathrm{and}\, \phi_2 < 0 \,\mathrm{and}\, \phi_3 < 0\}
\end{aligned}
$$

The energy function to be minimized is given as:

$$
\begin{aligned}
& F(\phi_1, \phi_2, c_{ij}) = \mu_1 \int_\Omega \delta_0(\phi_1(x, y)) \left|\nabla \phi_1(x, y)\right| dxdy + \mu_2 \int_\Omega \delta_0(\phi_2(x, y)) \left|\nabla \phi_2(x, y)\right| dxdy \\
& + \mu_3 \int_\Omega \delta_0(\phi_3(x, y)) \left|\nabla \phi_3(x, y)\right| dxdy + v_1 \int_\Omega H(\varphi_1(x, y)) dxdy + v_2 \int_\Omega H(\varphi_2(x, y)) dxdy \\
& + v_3 \int_\Omega H(\varphi_3(x, y)) dxdy \\
& + \lambda \int_\Omega \left|I_0(x, y) - c_{111}\right|^2 H(\phi_1(x, y)) H(\phi_2(x, y)) H(\phi_3(x, y)) dxdy \\
& + \lambda \int_\Omega \left|I_0(x, y) - c_{110}\right|^2 H(\phi_1(x, y)) H(\phi_2(x, y)) (1 - H(\phi_3(x, y))) dxdy \\
& + \lambda \int_\Omega \left|I_0(x, y) - c_{101}\right|^2 H(\phi_1(x, y)) (1 - H(\phi_2(x, y))) H(\phi_3(x, y)) dxdy \\
& + \lambda \int_\Omega \left|I_0(x, y) - c_{100}\right|^2 H(\phi_1(x, y)) (1 - H(\phi_2(x, y)) (1 - H(\phi_3(x, y))) dxdy \\
& + \lambda \int_\Omega \left|I_0(x, y) - c_{011}\right|^2 (1 - H(\phi_1(x, y)) H(\phi_2(x, y)) H(\phi_3(x, y)) dxdy \\
& + \lambda \int_\Omega \left|I_0(x, y) - c_{010}\right|^2 (1 - H(\phi_1(x, y)) H(\phi_2(x, y)) (1 - H(\phi_3(x, y))) dxdy \\
& + \lambda \int_\Omega \left|I_0(x, y) - c_{001}\right|^2 (1 - H(\phi_1(x, y)) (1 - H(\phi_2(x, y))) H(\phi_3(x, y)) dxdy \\
& + \lambda \int_\Omega \left|I_0(x, y) - c_{000}\right|^2 (1 - H(\phi_1(x, y))) (1 - H(\phi_2(x, y)) (1 - H(\phi_3(x, y))) dxdy
\end{aligned}
\tag{2.4.43}
$$

where $\mu_1, \mu_{2,} \mu_{3,} v_1, v_2, v_3 \lambda$ are constants.

After segmentation with three level set functions, the segmented image is approximated by a piecewise constant function which can take eight value.

$$\begin{aligned} I(x,y) = {} & c_{111}H(\phi_1(x,y))H(\phi_2(x,y))H(\phi_3(x,y)) \\ & + c_{110}H(\phi_1(x,y))H(\phi_2(x,y))(1-H(\phi_3(x,y))) + \\ & + c_{101}H(\phi_1(x,y))(1-H(\phi_3(x,y)))H(\phi_3(x,y)) \\ & + c_{100}H(\phi_1(x,y))(1-H(\phi_2(x,y)))(1-H(\phi_3(x,y))) \\ & + c_{011}(1-H(\phi_1(x,y))H(\phi_2(x,y))H(\phi_3(x,y)) \\ & + c_{010}(1-H(\phi_1(x,y))H(\phi_2(x,y))(1-H(\phi_3(x,y))) \\ & + c_{001}(1-H(\phi_1(x,y)))(1-H(\phi_2(x,y)))H(\phi_3(x,y)) \\ & + c_{000}(1-H(\phi_1(x,y)))(1-H(\phi_2(x,y)))(1-H(\phi_3(x,y))) \end{aligned} \qquad \textbf{(2.4.44)}$$

where $c_{111,}c_{110}, c_{101}, c_{100,}c_{011}, c_{010}, c_{001}$, and c_{000} are averages of $\Omega_{111,}\Omega_{110}, \Omega_{101}, \Omega_{100,}$ $\Omega_{011}, \Omega_{010}, \Omega_{001}$, and Ω_{000} regions respectively, and can be defined as follows:

$$c_{111} = \frac{\int_\Omega I_0(x,y)H(\phi_1(x,y))H(\phi_2(x,y))H(\phi_3(x,y))dxdy}{\int_\Omega H(\phi_1(x,y))H(\phi_2(x,y))H(\phi_3(x,y))}.$$

$$c_{110} = \frac{\int_\Omega I_0(x,y)H(\phi_1(x,y))H(\phi_2(x,y))(1-H(\phi_3(x,y)))dxdy}{\int_\Omega H(\phi_1(x,y))H(\phi_2(x,y))(1-H(\phi_3(x,y))dxdy}$$

$$c_{101} = \frac{\int_\Omega I_0(x,y)H(\phi_1(x,y))(1-H(\phi_2(x,y)))H(\phi_3(x,y))dxdy}{\int_\Omega H(\phi_1(x,y))(1-H(\phi_2(x,y)))H(\phi_3(x,y))dxdy}$$

$$c_{011} = \frac{\int_\Omega I_0(x,y)(1-H(\phi_1(x,y)))H(\phi_2(x,y))H(\phi_3(x,y))dxdy}{\int_\Omega (1-H(\phi_1(x,y)))H(\phi_2(x,y))H(\phi_3(x,y))dxdy}$$

$$\begin{aligned}
c_{010} &= \frac{\int_{\Omega} I_0(x,y)(1-H(\phi_1(x,y)))H(\phi_2(x,y))(1-H(\phi_3(x,y)))dxdy}{\int_{\Omega}(1-H(\phi_1(x,y)))H(\phi_2(x,y))(1-H(\phi_3(x,y)))dxdy} \\
c_{001} &= \frac{\int_{\Omega} I_0(x,y)(1-H(\phi_1(x,y)))(1-H(\phi_2(x,y)))H(\phi_3(x,y))dxdy}{\int_{\Omega}(1-H(\phi_1(x,y)))(1-H(\phi_2(x,y)))H(\phi_3(x,y))dxdy} \\
c_{000} &= \frac{\int_{\Omega} I_0(x,y)(1-H(\phi_1(x,y)))(1-H(\phi_2(x,y)))(1-H(\phi_3(x,y)))dxdy}{\int_{\Omega}(1-H(\phi_1(x,y)))(1-H(\phi_2(x,y)))(1-H(\phi_3(x,y)))dxdy}
\end{aligned} \tag{2.4.45}$$

By keeping $c_{111,}\, c_{110}, c_{101}, c_{100,}\, c_{011}, c_{010}, c_{001},$ and c_{000} fixed and minimizing F with respect to ϕ_1, ϕ_2 and ϕ_3 with an artificial time $t \ge 0$, Euler–Lagrange equations for ϕ_1, ϕ_2 and ϕ_3 are obtained as follows:

$$\begin{aligned}
\frac{\partial \phi_1}{\partial t} &= \delta_\varepsilon(\phi_1)\mu_1 \nabla.\left(\frac{\nabla \phi_1}{|\nabla \phi_1|}\right) - \delta_\varepsilon(\phi_1)v_1 - \lambda\delta_\varepsilon(\phi_1)\left[(I_0 - c_{111})^2 - (I_0 - c_{011})^2\right]H(\phi_2)H(\phi_3) \\
&\quad - \lambda\delta_\varepsilon(\phi_1)\left[(I_0 - c_{110})^2 - (I_0 - c_{010})^2\right]H(\phi_2)(1-H(\phi_3)) \\
&\quad - \lambda\delta_\varepsilon(\phi_1)\left[(I_0 - c_{101})^2 - (I_0 - c_{001})^2\right](1-H(\phi_2))H(\phi_3) \\
&\quad - \lambda\delta_\varepsilon(\phi_1)\left[(I_0 - c_{100})^2 - (I_0 - c_{000})^2\right](1-H(\phi_2))(1-H(\phi_3)) = 0
\end{aligned} \tag{2.4.46}$$

$$\begin{aligned}
\frac{\partial \phi_2}{\partial t} &= \delta_\varepsilon(\phi_2)\mu_2 \nabla.\left(\frac{\nabla \phi_2}{|\nabla \phi_2|}\right) - \delta_\varepsilon(\phi_2)v_2 - \lambda\delta_\varepsilon(\phi_2)\left[(I_0 - c_{111})^2 - (I_0 - c_{101})^2\right]H(\phi_1)H(\phi_3) \\
&\quad - \lambda\delta_\varepsilon(\phi_2)\left[(I_0 - c_{110})^2 - (I_0 - c_{100})^2\right]H(\phi_1)(1-H(\phi_3)) \\
&\quad - \lambda\delta_\varepsilon(\phi_2)\left[(I_0 - c_{011})^2 - (I_0 - c_{001})^2\right](1-H(\phi_1))H(\phi_3) \\
&\quad - \lambda\delta_\varepsilon(\phi_2)\left[(I_0 - c_{010})^2 - (I_0 - c_{000})^2\right](1-H(\phi_1))(1-H(\phi_3)) = 0
\end{aligned} \tag{2.4.47}$$

$$\begin{aligned}
\frac{\partial \phi_3}{\partial t} &= \delta_\varepsilon(\phi_3)\mu_3 \nabla.\left(\frac{\nabla \phi_3}{|\nabla \phi_3|}\right) - \delta_\varepsilon(\phi_3)v_3 - \lambda\delta_\varepsilon(\phi_3)\left[(I_0 - c_{111})^2 - (I_0 - c_{110})^2\right]H(\phi_1)H(\phi_2) \\
&\quad - \lambda\delta_\varepsilon(\phi_3)\left[(I_0 - c_{101})^2 - (I_0 - c_{100})^2\right]H(\phi_1)(1-H(\phi_2)) \\
&\quad - \lambda\delta_\varepsilon(\phi_3)\left[(I_0 - c_{011})^2 - (I_0 - c_{010})^2\right](1-H(\phi_1))H(\phi_2) \\
&\quad - \lambda\delta_\varepsilon(\phi_3)\left[(I_0 - c_{001})^2 - (I_0 - c_{000})^2\right](1-H(\phi_1))(1-H(\phi_2)) = 0
\end{aligned} \tag{2.4.48}$$

Numerical implementation of equations 2.3.46, 2.4.47, and 2.4.48 are similar to equation 2.4.36.

We applied three level set functions to the original image given in Figure-2.4.15(a) , where the initial positions of the contours are given in Figure-2.4.25. The zero level set functions after 400 iterations are given in Figures-2.4.26, Figure-2.4.27 and Figure-2.4.28.

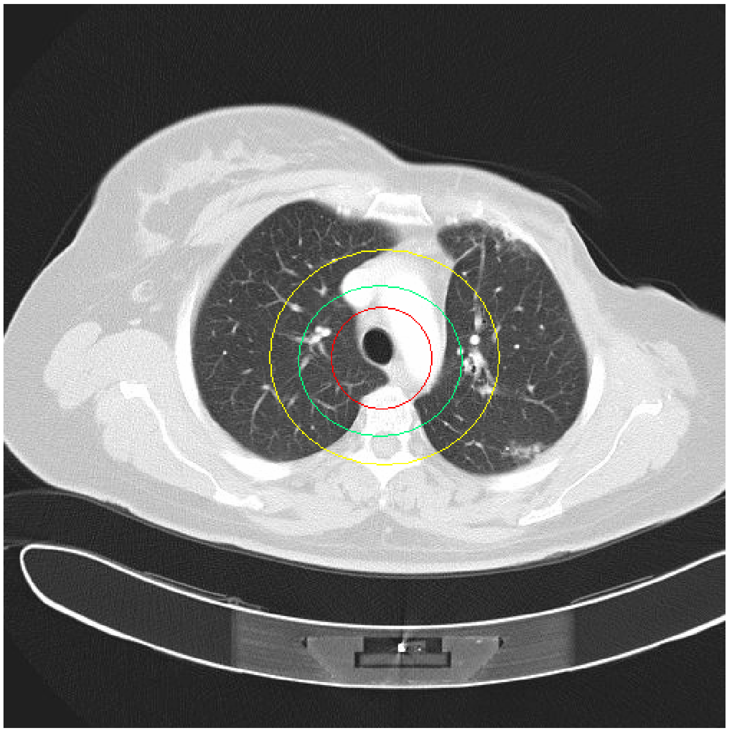


Figure 2.4.25: Initial positions of the three level set functions.

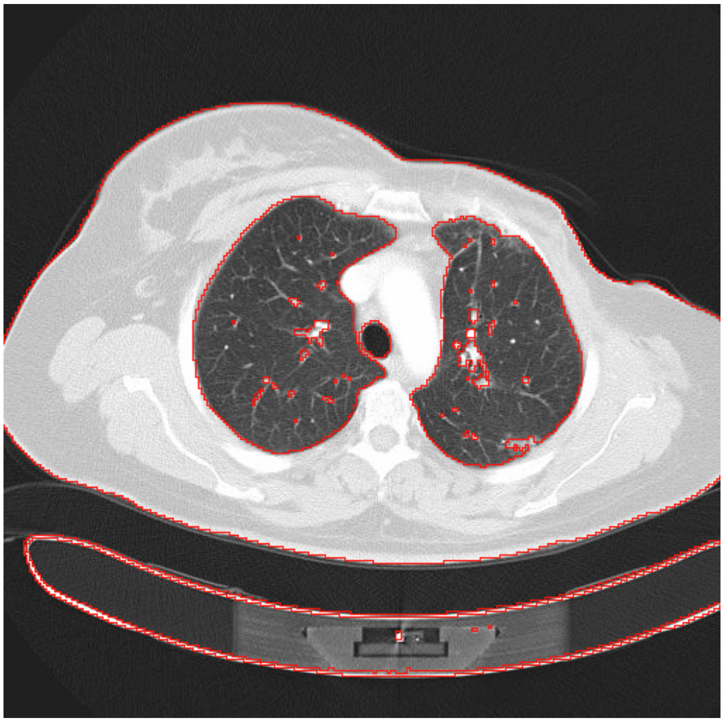


Figure 2.4.26: Resultant zero level set function ϕ_1

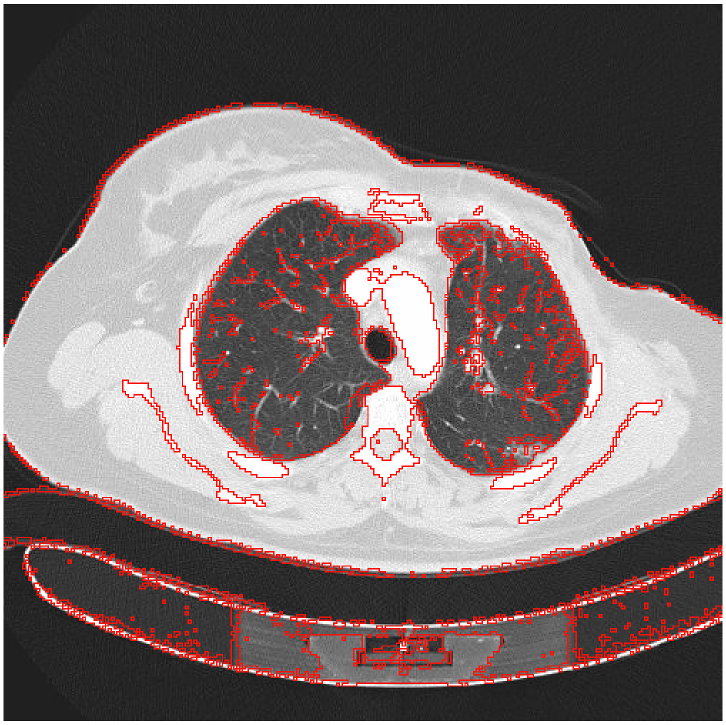


Figure 2.4.27: Resultant zero level set function ϕ_2

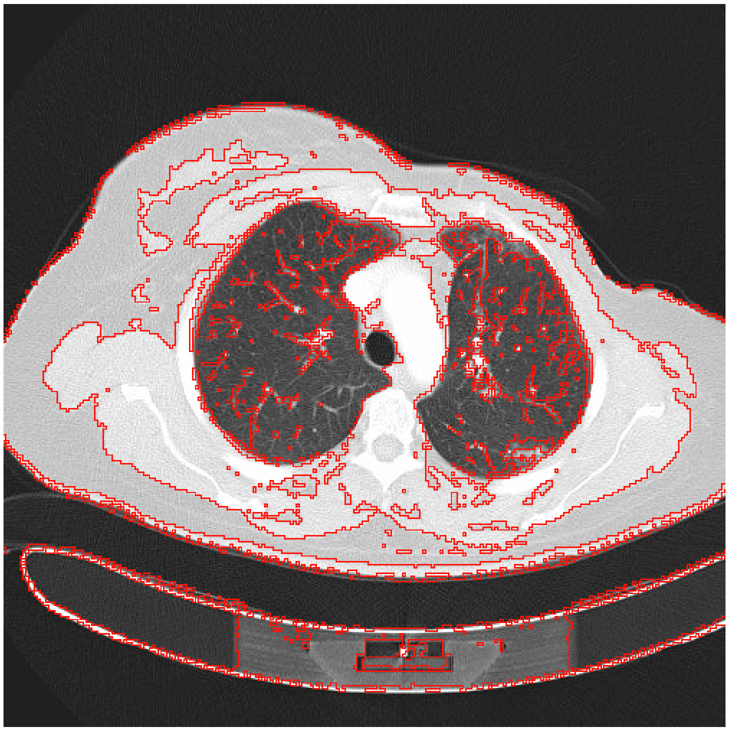


Figure 2.4.28: Resultant zero level set function ϕ_3

The eight disjoint regions obtained from resultant level set functions are given in Figure-2.4.29, where each of the eight disjoint regions are given in different colors. In Figure-2.4.30 to Figure-2.4.37, eight phases or disjoint regions are given as binary images, where white pixels shows result of segmentation and black is the background for the corresponding phase image. We can see that pixels belonging to bones and some of the pixels belonging to heart are classified in Phase-1 image. Pixels belong to fat and muscle tissue, which are illustrated with red and yellow in Figure-2.4.29, respectively, are classified in Phase-5 and Phase-7 images respectively. Most of the pixels belong to lungs are classified in Phase-4 image and the vessels are classified in Phase-2 and Phase-6 images.

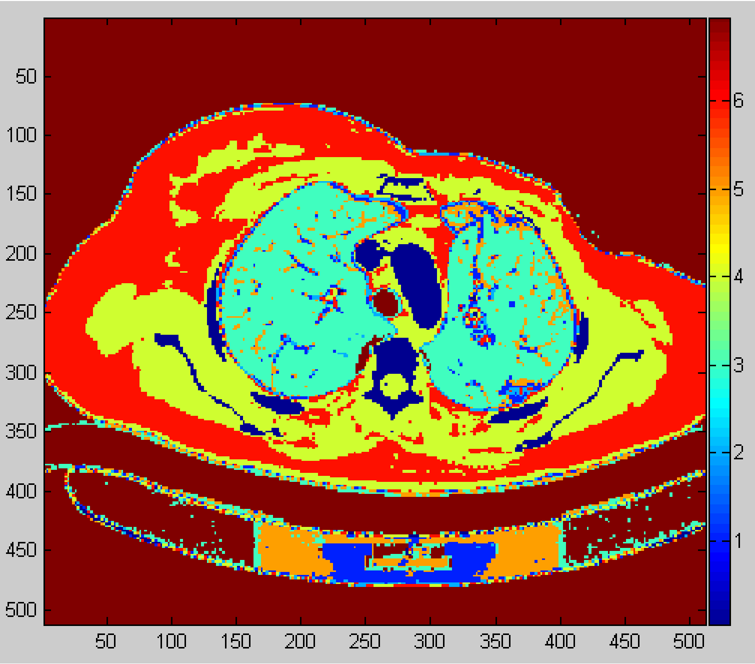


Figure 2.4.29: Result of segmentation with three level set functions after 400 iterations.

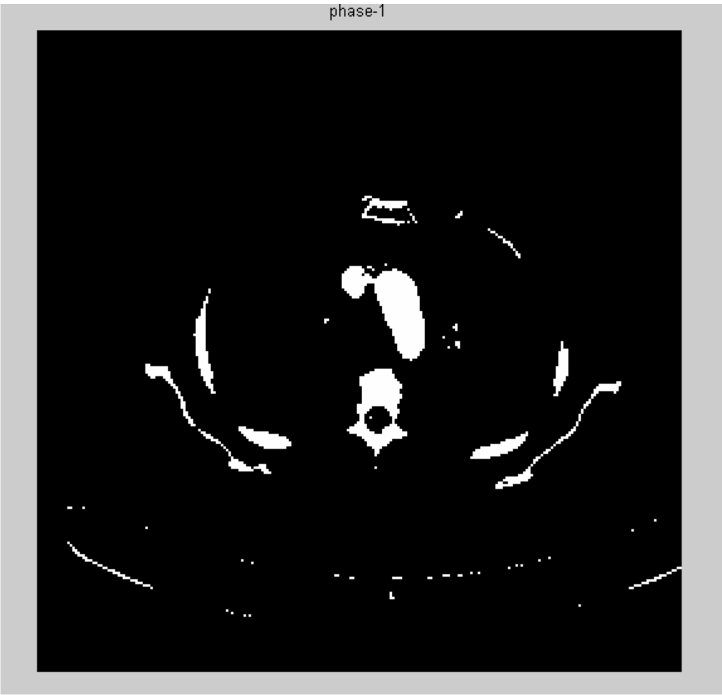


Figure 2.4.30: Phase-1 image obtained three level set functions after 400 iterations

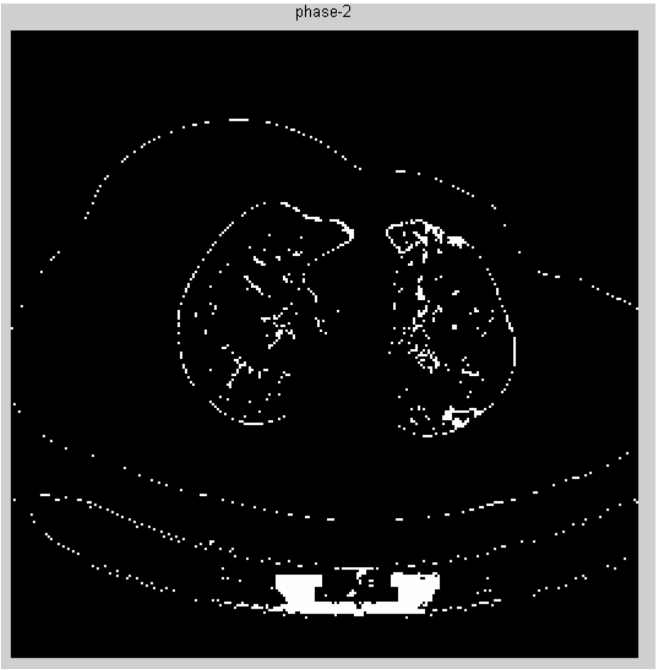


Figure 2.4.31: Phase-2 image obtained three level set functions after 400 iterations

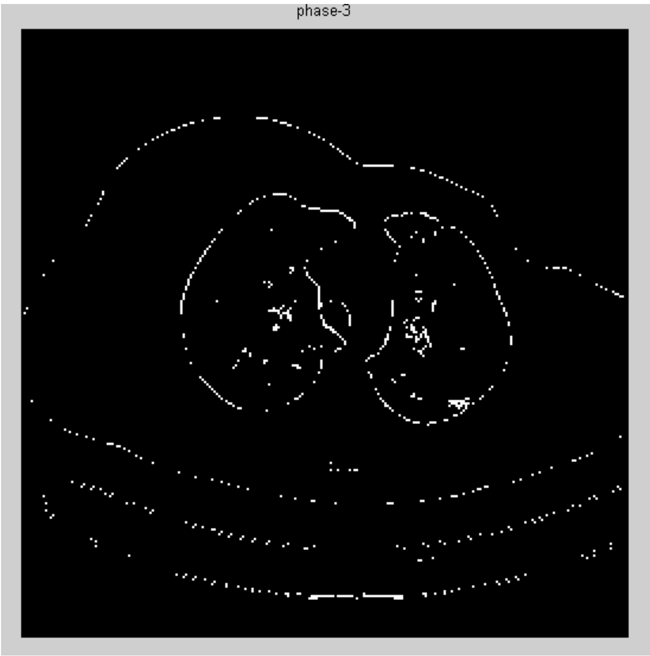


Figure 2.4.32: Phase-3 image obtained three level set fuctions after 400 iterations

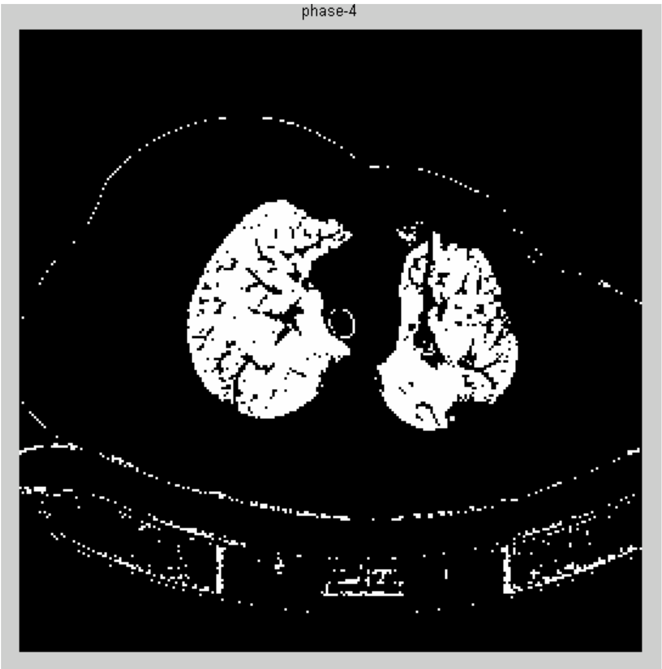


Figure 2.4.33: Phase-4 image obtained three level set functions after 400 iterations

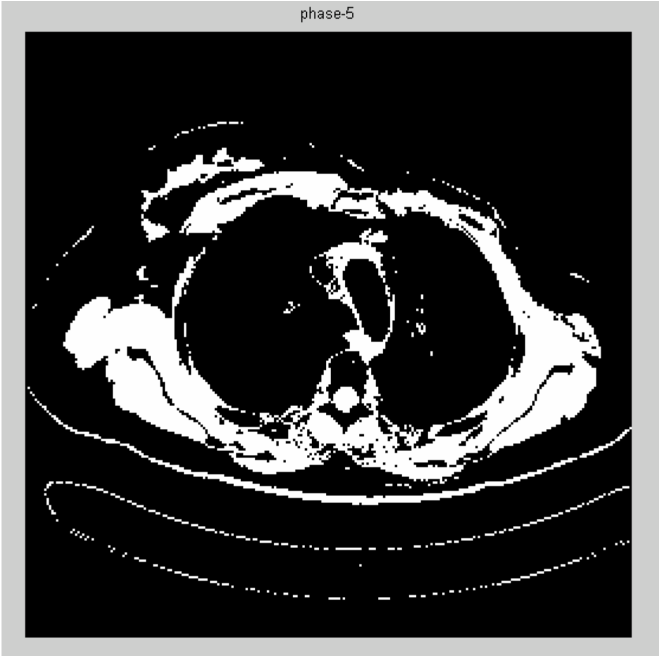


Figure 2.4.34: Phase-5 image obtained three level set functions after 400 iterations

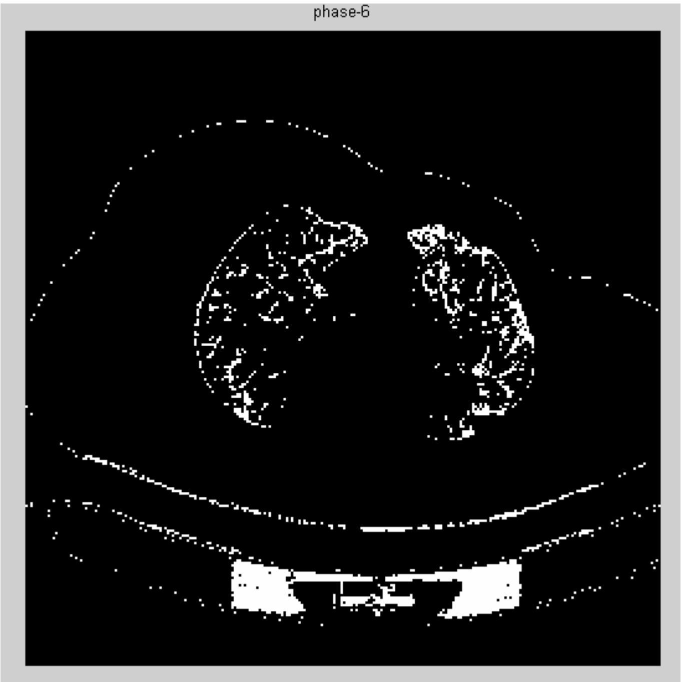


Figure 2.4.35: Phase-6 image obtained three level set functions after 400 iterations

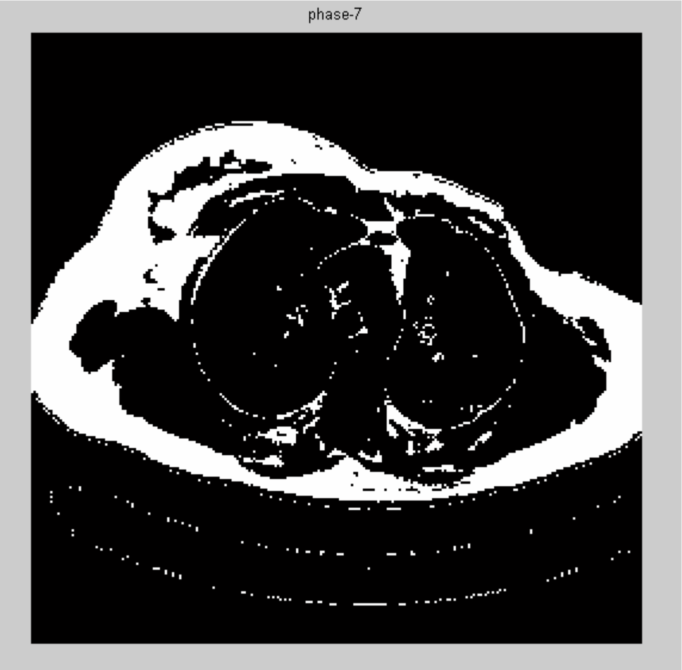


Figure 2.4.36: Phase-7 image obtained three level set functions after 400 iterations

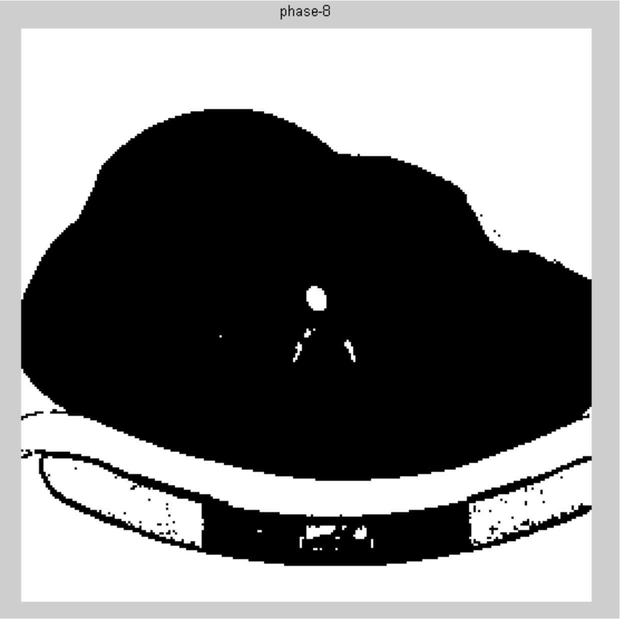


Figure 2.4.37: Phase-8 image obtained three level set functions after 400 iterations

After obtaining the image given in Figure-2.4.29, and manually selecting disconnected segmented regions from this image, we obtained the segmented images given in Figure-2.4.38 to Figure-2.4.42. Results of segmentation for muscle, fat, bones, heart and lungs, by using the obtained eight phase images are given in Table-2.4.2 and Figure-2.4.38 to Figure-2.4.42.

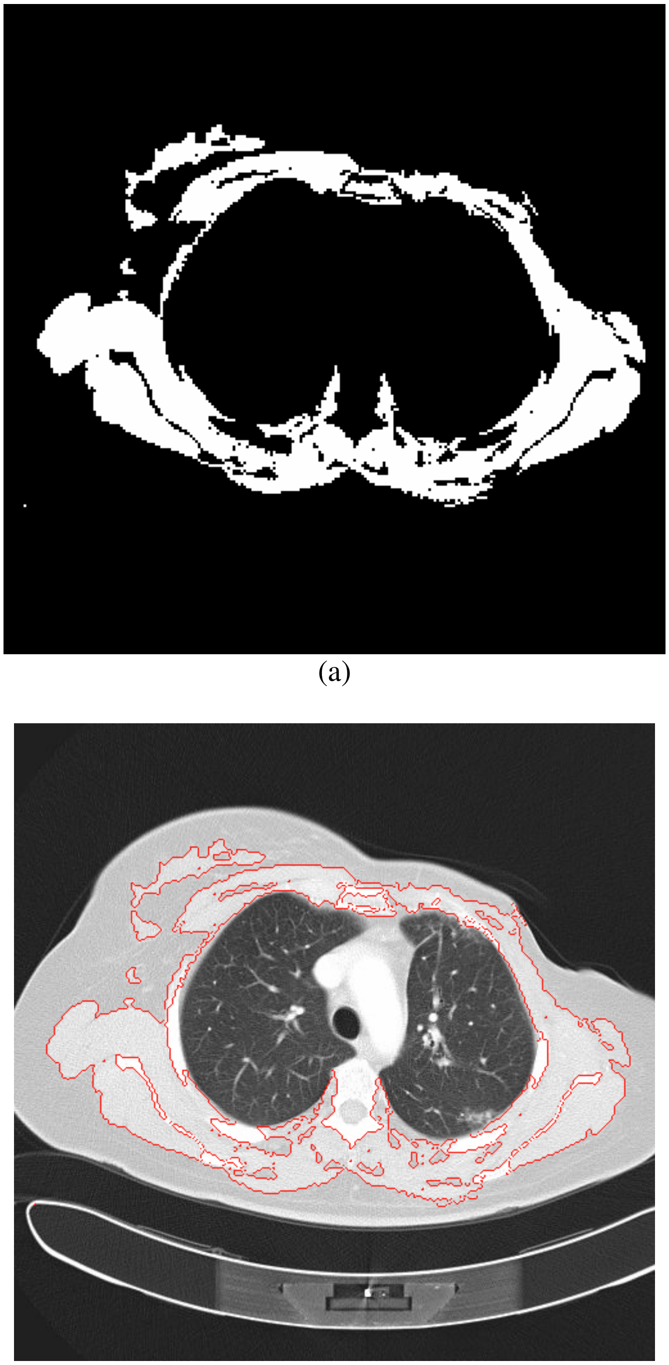
Table 2.4.2*:* Numerical Results for segmentation with three level sets

Tissues/Organs	Negative Rate Metric (NR)	Elapsed Time (sec)
Heart	0.00892	101.2
Lungs	0.1021	
Bones	0.14380	
Fat Tissue	0.00740	
Muscle Tissue	0.04550	

By evaluating Table-2.4.2, Figure-2.4.38 and Figure-2.4.39, we can state that segmentation of the muscle and fat tissues can be performed very accurately, with NR values of 0.0455 and 0.0074 respectively. Result for segmentation of lungs is given in Figure-2.4.42, with NR value of 0.1021, and we can state that the obtained result is acceptable. On the other hand, by looking at Figure-2.4.41 and NR value in Table-2.4.2, we can state that segmentation of bones can not be performed very accurately by using active contours without edges with three level set functions.

Because the regions belonging to heart are participated in different phase images (phase-1, phase-5 and phase-7), segmentation of heart must be done manually by selecting sub-regions belonging to the heart. The resultant image for segmentation of heart, given in Figure 2.4.39, and the NR value given in Table-2.4.2 are obtained after user interaction.

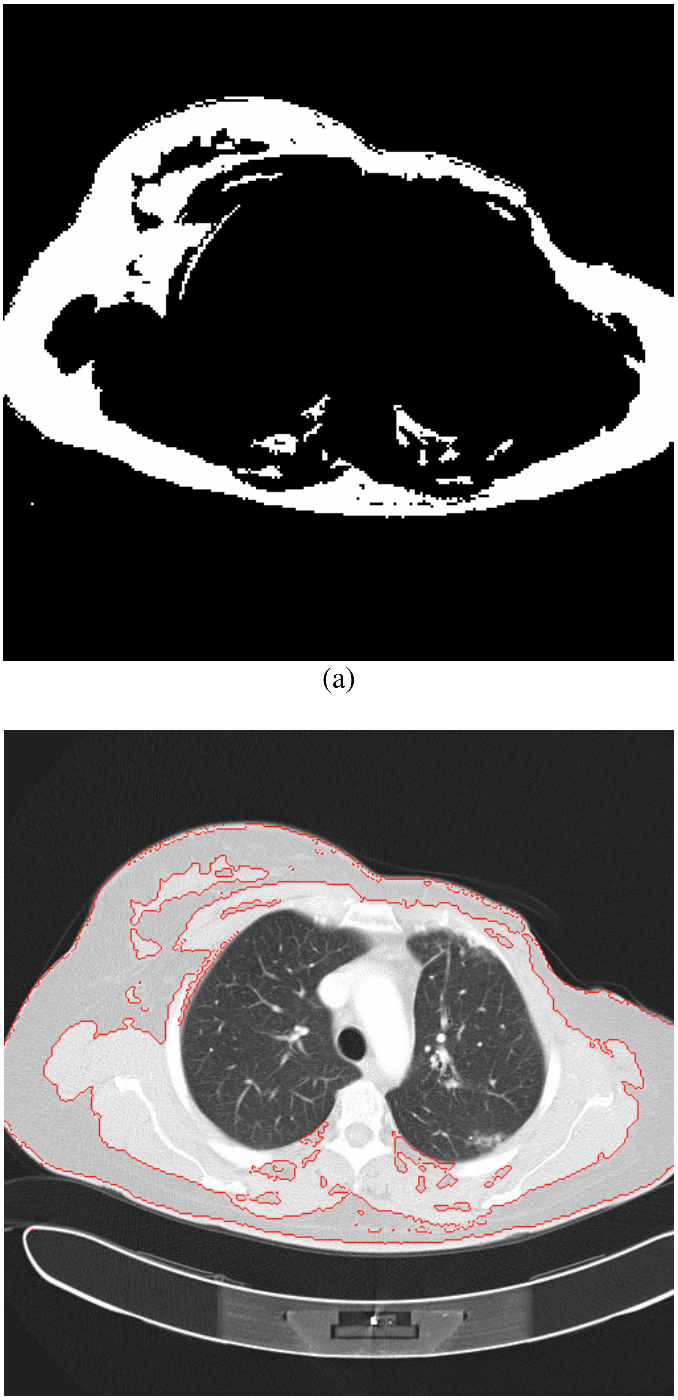
In conclusion we can state that active contours without edges algorithm with three level set functions performs segmentation of bones, lungs, fat, and muscle tissues automatically, and produces accurate results. For segmentation of the heart, user interaction is needed for merging the sub-regions belonging to heart. By using more than three level set functions, we can divide the image to more than eight disjoint regions and perform segmentation more accurately. On the other hand, using more than three level set functions requires more processing power



(a)

(b)

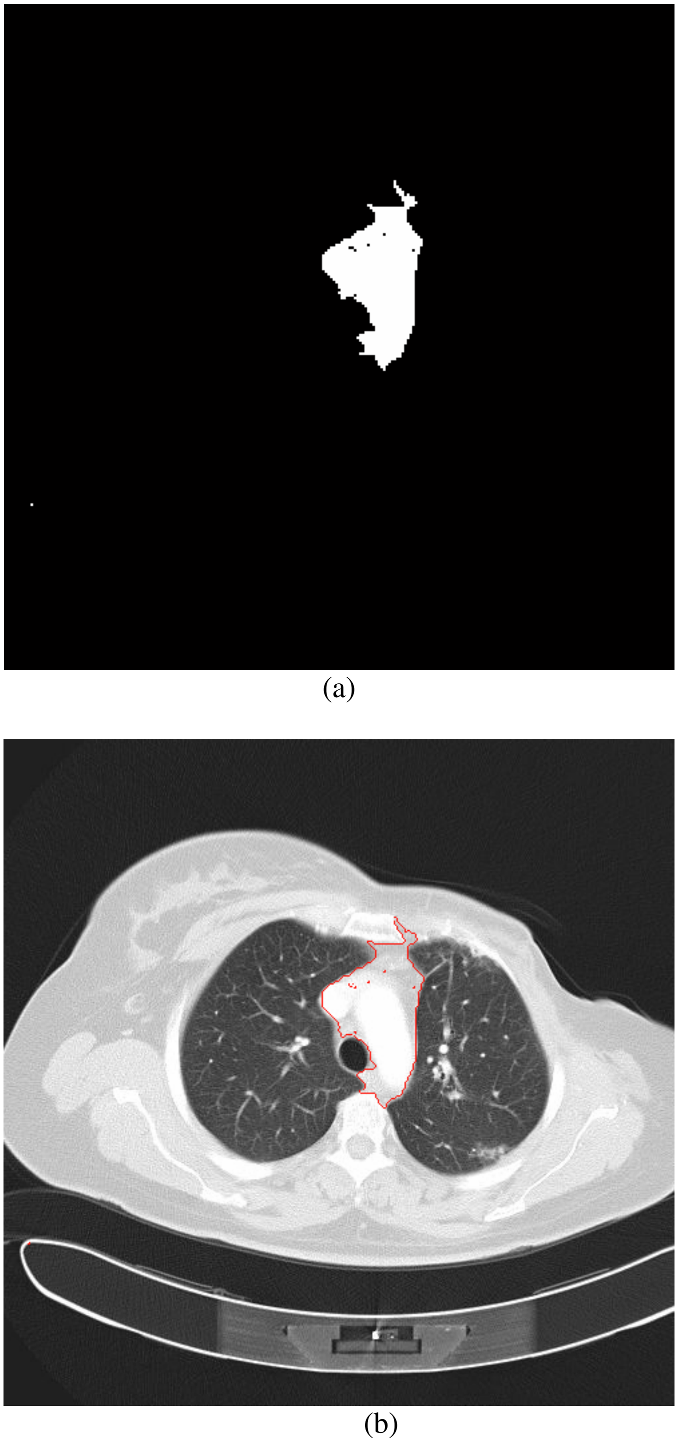
Figure 2.4.38: Result of segmentation for muscle tissue, (a) segmented binary image, (b) original image overlapped with the obtained boundaries



(a)

(b)

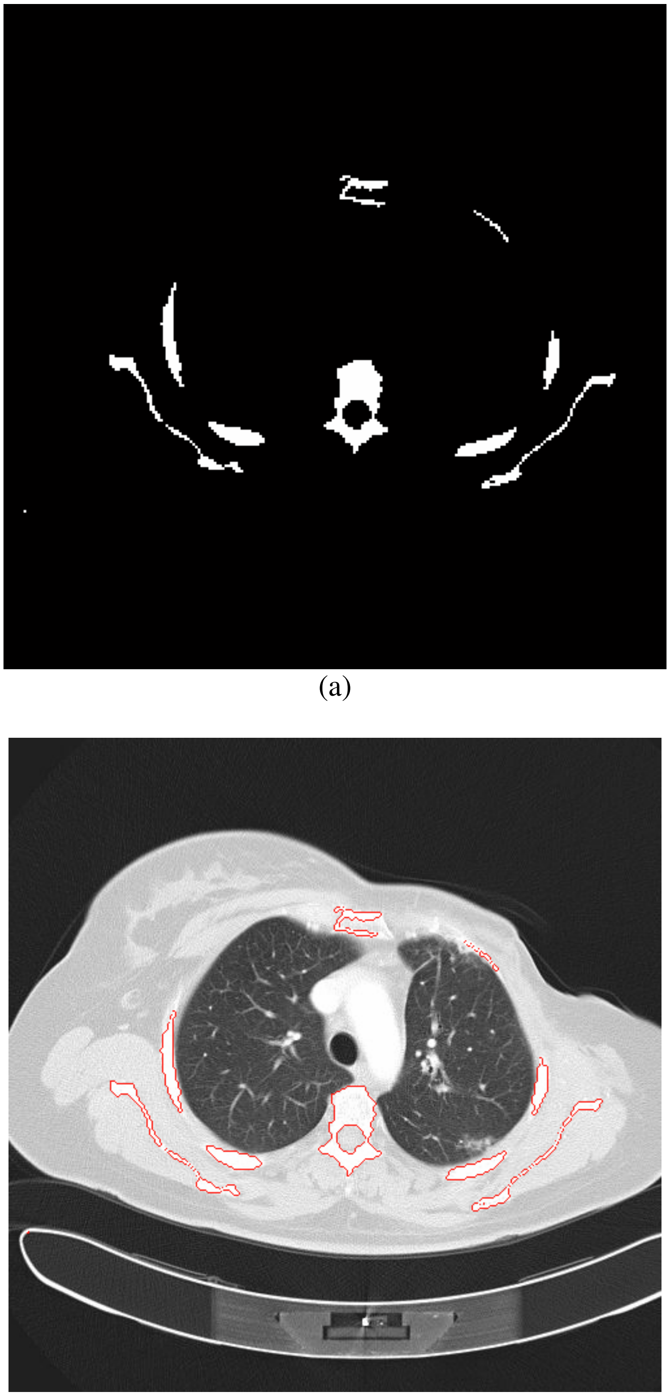
Figure 2.4.39: Result of segmentation for fat tissue, (a) segmented binary image, (b) original image overlapped with the obtained boundaries



(a)

(b)

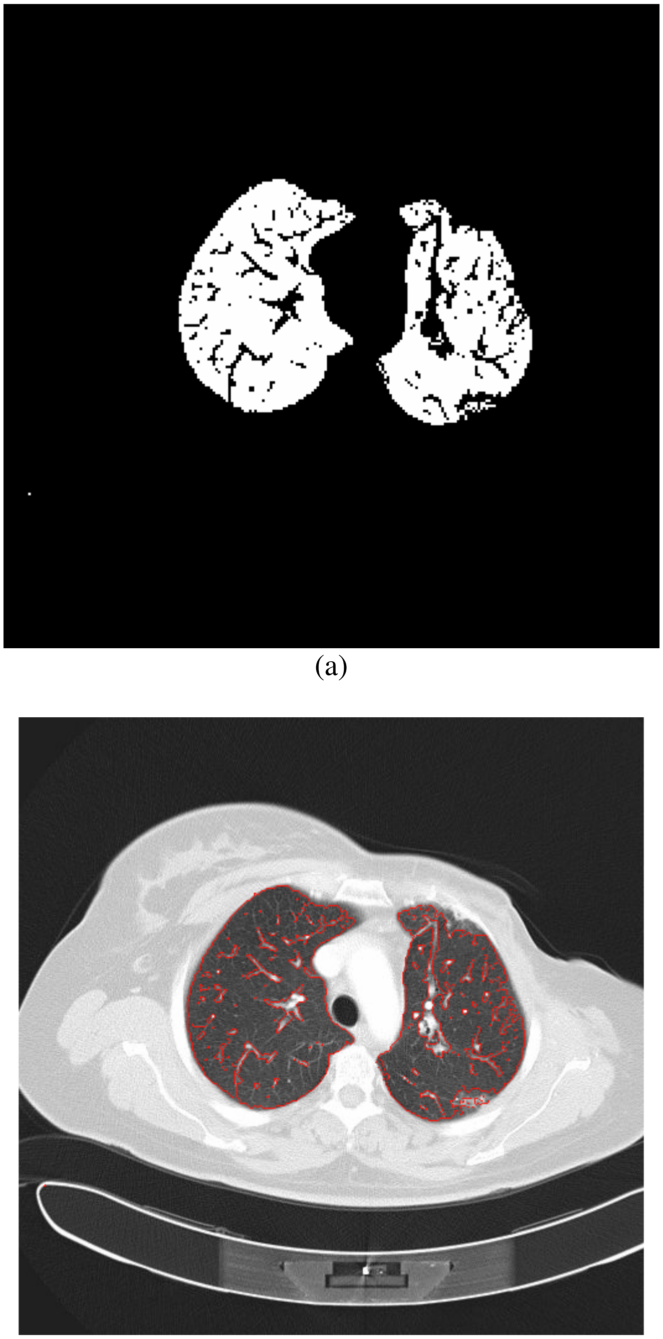
Figure 2.4.40: Result of segmentation for the heart, (a) segmented binary image, (b) original image overlapped with the obtained boundaries



(a)

(b)

Figure 2.4.41: Result of segmentation for bones, (a) segmented binary image, (b) original image overlapped with the obtained boundaries



(a)

(b)

Figure 2.4.42: Result of segmentation for lungs, (a) segmented binary image, (b) original image overlapped with the obtained boundaries

CHAPTER 3

HYBRID METHOD

Hybrid image segmentation methods combine the principles of two or more image segmentation techniques. As mentioned in previous chapters, both pixel based, region-based methods and deformable models have advantages and disadvantages. Hybrid method given in this chapter uses advantageous parts of different techniques. In this chapter, we have one hybrid method; watershed transformation and region merging (WTRM). In this method, we combined watershed transformation with region merging which is not a segmentation method alone but similar in idea to region growing. Moreover to increase the speed and the accuracy of this method we combined WTRM with K-means clustering.

3.1. Watershed Transformation and Region Merging

After initial partitioning of the image into primitive regions by applying the watershed transformation to the gradient image, further segmentation can be obtained by using a region merging process.

After watershed transformation, resultant image *I,* which is composed of *M* sub regions, R_k (k=1,2,...M), is:

$$I = \bigcup_{k=1}^{K} R_k$$

and for each region R_i, R_j :

$$R_i \cap R_j = \phi \quad \forall i, j \in [1,2,....M] \quad \text{for } i \neq j$$

$$R_i, \forall i \in [1,2,....M], \textit{connected} \tag{3.1}$$

$$I(p) = C_i, \text{ if pixel } p \in R_i, \forall i \in [1,2,....M]$$

where $I(p)$ is the gray level value of pixel p, which is a member of ith sub-region R_i, C_i is a constant (label of R_i) and $C_i \neq C_j$ if R_i and R_j are not adjacent regions.

The flowchart of the watershed transformation and region merging algorithm is given in Figure-3.1. Median filtering is applied to smooth the original image before gradient computation, and anisotropic filtering is applied to cope with the oversegmentation problem. Region merging process is performed by region algorithms, which uses a Region Adjacency Graph (RAG) that represents neighboring image regions (see Figure-3.2). RAG is can be obtained by finding all the neighboring regions of all the primitive regions obtained by watershed transformation. These algorithms are iterative, that is at each step the most similar pair of adjacent regions is detected using a cost function and merged. After merging adjacent regions, the RAG is updated.

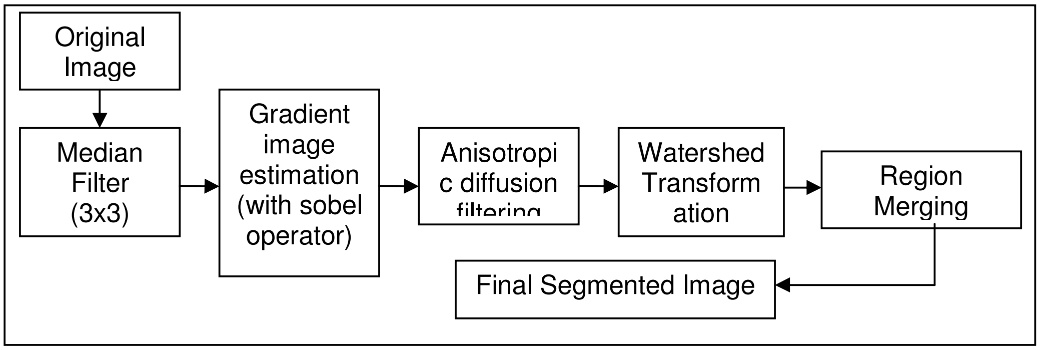

Original Image
Median Filter (3x3)
Gradient image estimation (with sobel operator)
Anisotropi c diffusion filtering
Watershed Transform ation
Region Merging
Final Segmented Image

Figure 3. 1: Stages of the WTRM

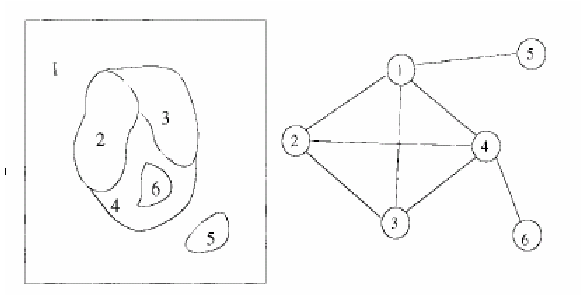


Figure 3. 2. Region Adjacency Graph (RAG)

Let ***W*** be the initial image partition produced by the watershed transformation, satisfying (3.1). The goal of the region merging process is to transform ***W*** to the final segmented image $\boldsymbol{I_s}$, using a sequence of region pair merges. Therefore, the problem is finding the optimal sequence of merges in the sense that its application to ***W*** produces $\boldsymbol{I_s}$.

3.1.1. Merging Criteria

A merging criterion consists of two parts: *a region model*, describing each image region with a set of features, and *a dissimilarity measure*, defining a metric on the features of the region model [60]. Some of possible region models are uniform luminance texture, shape or motion parameters. In our study the region model's feature is the mean luminance of each region. Dissimilarity measures are given the next sub-section.

3.1.1.1. Region Dissimilarity Functions

Let, I be the original image and

$R_M = \{R_M^1, R_M^2, R_M^3 ......R_M^M\}$ be M-partition of image I and

$R_M^k = \{p_1, p_1, ...p_k, N_k\}$ is the set of pixels belonging to the region R_M^k. Where k=1,2,..M, and N_k is the total number of pixels.

In the piecewise constant approximation to *I*, the image intensity in each region R_M^k , is approximated with one parameter which minimizes the overall square error. This parameter is nothing but the gray level mean value, therefore as a region model, we represent each region with its mean gray level value [61,62].

Mean value of ***I*** in R_M^k is,

$$\mu(R_M^k) = \frac{1}{N}\sum_{i=1}^{N} I(p_i) \tag{3. 2}$$

and the corresponding square error is,

$$E(R_M^k) = \sum_{i=1}^{N} \left(I(p_i) - \mu(R_M^k)\right)^2 \tag{3. 3}$$

If R_M is the optimal M-partition which minimizes the square error, then the optimal (M-1) partition is generated by merging the pair of regions R_M that minimizes the dissimilarity function.

Various region dissimilarity functions are used in this study to obtain the best one and some of the most efficient region dissimilarity functions are given below.

- **Mean Luminance Difference**

The simplest region model is to describe each region R_M^i by its mean luminance $\mu(R_M^i)$. A straightforward possibility to define a dissimilarity measure on this model is to use the squared difference. This dissimilarity function is one of the most basic and efficient dissimilarity functions [60].

$$\delta(R_M^i, R_M^j) = (\mu(R_M^i) - \mu(R_M^j))^2 \tag{3. 4}$$

If $\delta(R_M^i, R_M^j)$ is smaller than a threshold value (T), regions R_M^i, R_M^j are merged. Some choices of the threshold value T can be:

$*\quad T = n * \mu(R_M^i)$

$*\quad T = m * stdv(R_M^i)$

where *stdv* stands for standard deviation, m,n are positive and real. Obtaining the threshold value is an ad-hoc process.

- **Ward's Criterion**

This dissimilarity function operates on the Ward's criterion;

$$\delta(R_M^i, R_M^j) = \frac{N_i * N_j}{N_i + N_j}\left[\mu(R_M^i) - \mu(R_M^j)\right]^2 Q(i,j) \qquad \textbf{(3. 5)}$$

where *Q(i,j)* is 1 if R_M^i, R_M^j are adjacent, else $+\infty$.

If $\delta(R_M^i, R_M^j)$ is smaller than a threshold value, adjacent regions R_M^i, R_M^j are merged. Obtaining the threshold value is an ad-hoc process [60].

- **Border Criterion**

This dissimilarity function operates on the luminance values of pixels along the common boundary [60].

Let $B_{i,j} = \{(p_a, p_b)\}$be the set of pixels along the common boundary between regions R_M^i andR_M^j. Then the border criterion is defined as:

$$\delta(R_M^i, R_M^j) = \sum_{p_a, p_b \in B_{i,j}} (I(p_a) - I(p_b))^2 \qquad \textbf{(3. 6)}$$

If $\delta(R_M^i, R_M^j)$ is smaller than a threshold value, adjacent regions R_M^i, R_M^j are merged. Obtaining the threshold value is an ad-hoc process.

Within these Region Dissimilarity Functions, Ward's criterion is superior to others because the cost is calculated only using mean values (without standard deviation which can cause error in noisy images) and the number of pixels of regions is taken into account. Therefore we used Ward's criterion.

3.1.2. Region Merging Algorithms

Region merging algorithms defines the process of merging two adjacent regions using region dissimilarity functions. We used two different region merging techniques. The first one, Algorithm-1 is the widely used one, The second one is proposed by us to enhance the speed of Algorithm-1.

3.1.2.1. Algorithm 1

In this algorithm a RAG is computed, from the given watershed segmented image. This algorithm is applied to all of the sub-regions in the image. Starting from a sub-region and using the RAG, the dissimilarity function is applied for all of the neighboring regions. If the dissimilarity test for a neighboring region is satisfied, then regions are merged, and RAG is updated (see Figure 3.3). The merging process carries on until there is no neighboring region left to merge for the current sub-region. Algorithm-1 stops after all of the regions are processed. The pseudo code for Algorithm-1 is given in Appendix-B.3.

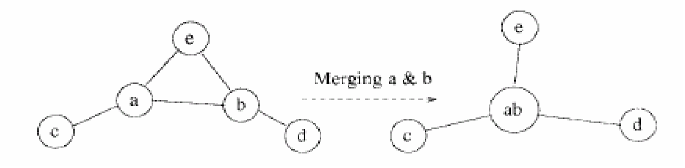


Figure 3. 3: Merging of two regions in RAG

It is obvious that the process takes too much time. Therefore, instead of processing for all the sub-regions, user can give a seed region to start and algorithm starts merging from similar adjacent regions until there are no neighboring regions left to merge.

- **Results of Algorithm-1**

  - **Simulation Results**

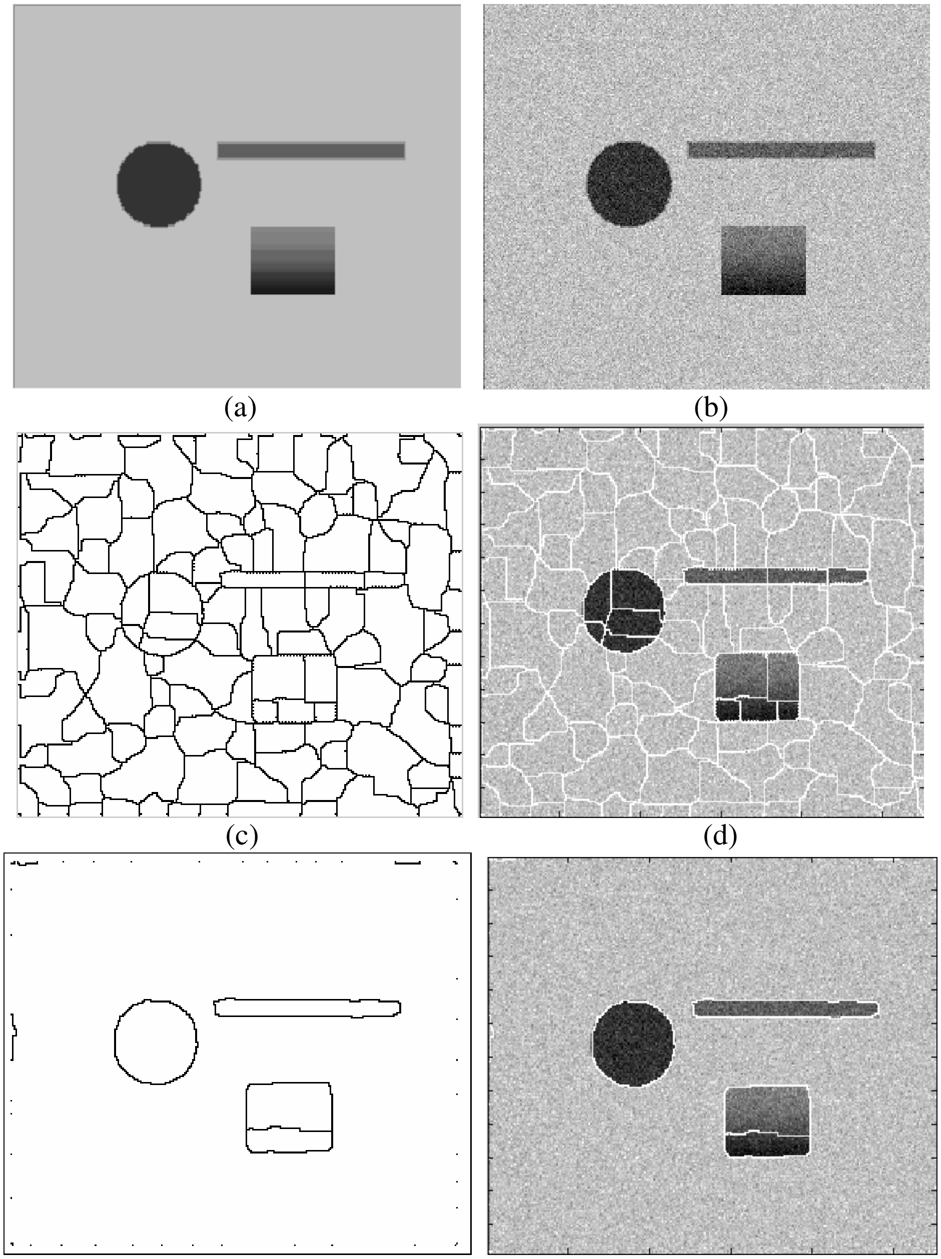
First, we applied Algorithm-1 to segment the test image test image shown in Figure 3.4. (b). This image is obtained by adding zero-mean Gaussian noise to the image in Figure-3.4 (a), where the SNR is 43 dB. Resulting watershed transformed image shown in Figure 3.4 (c) (with watershed lines black and catchment basins white) is composed of 129 sub-regions. After applying Algorithm-1 to the watershed transformed image with Ward's criterion dissimilarity function and we decreased the number of sub-regions to 12 as shown in Figure 3.4 (e). The threshold value for Ward's criterion is chosen as T=1.5*10^6.

Moreover, to compare the performances of dissimilarity functions, we applied Algorithm-1 using mean luminance dissimilarity function with threshold value, $T = 0.1 * \mu(R_M^i)$. The segmented image given in Figure 3.5(a) composed of 16 sub-regions. Result of segmentation with Border criterion dissimilarity function, is shown in Figure-3.5 (e), where the segmented image is composed of 22 sub-regions.

It can be seen from results obtained with different dissimilarity functions that, Ward's criterion produces the best result with 12 sub-regions. However we can say that the result obtained with mean-luminance dissimilarity function is comparable with Ward's criterion with 16 sub-regions. On the other hand, Border criterion dissimilarity function produces the worst result with 22 sub-regions. For more information on dissimilarity functions see [60].

- **CT Image Results**

We also examine the performance of Algorithm-1 for CT images. The original image is given in Figure 3.6 (a). Primitively segmented image given in Figure-3.6 (b) is obtained by applying watershed transformation to the original image, and consist of 7843 sub-regions. After applying region merging process with Ward's criterion, resultant image given in Figure 3.6 (e) is obtained, where the number of regions is decreased to 1005.



(a)

(b)

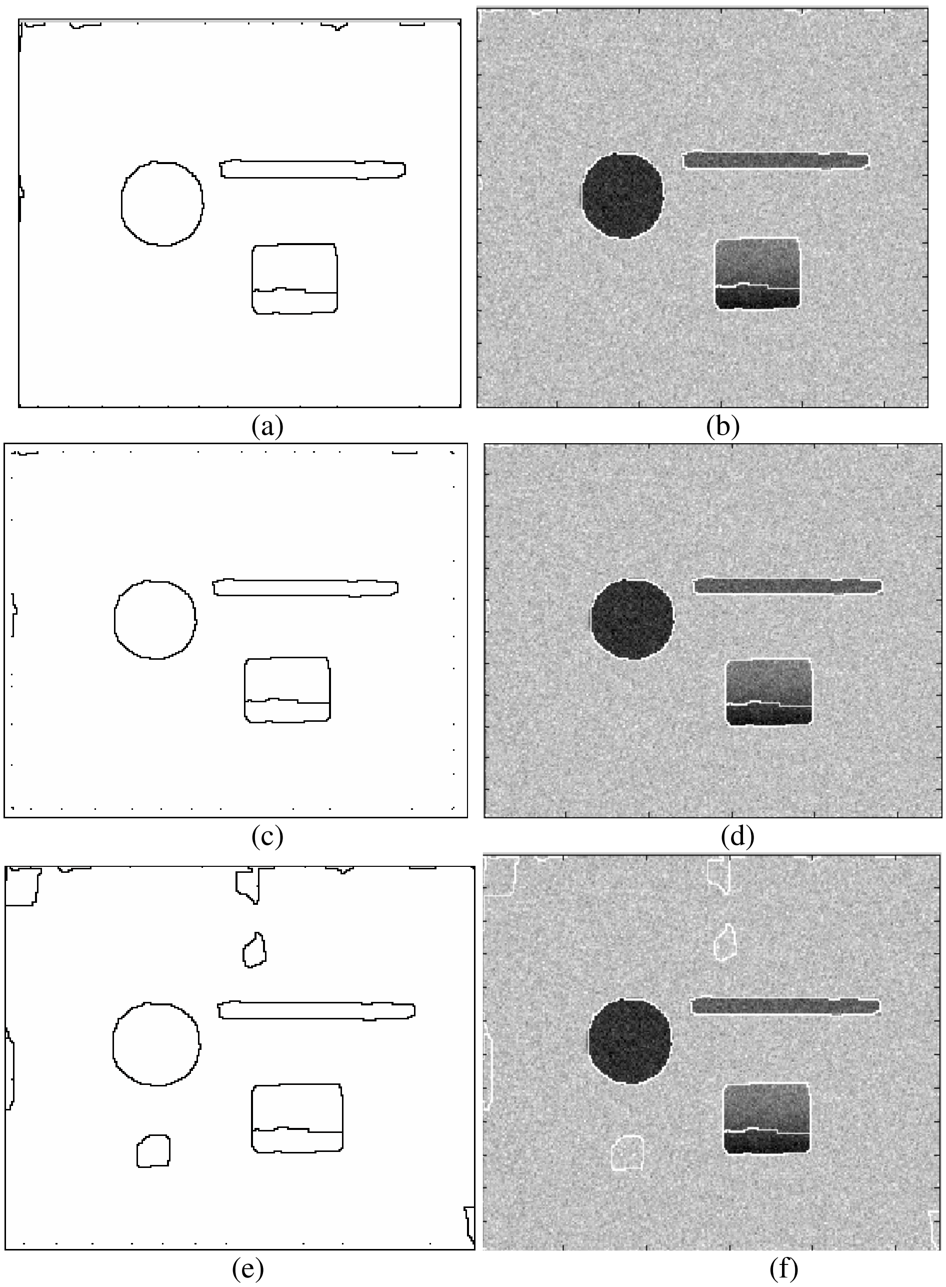
(c)

(d)

(e)

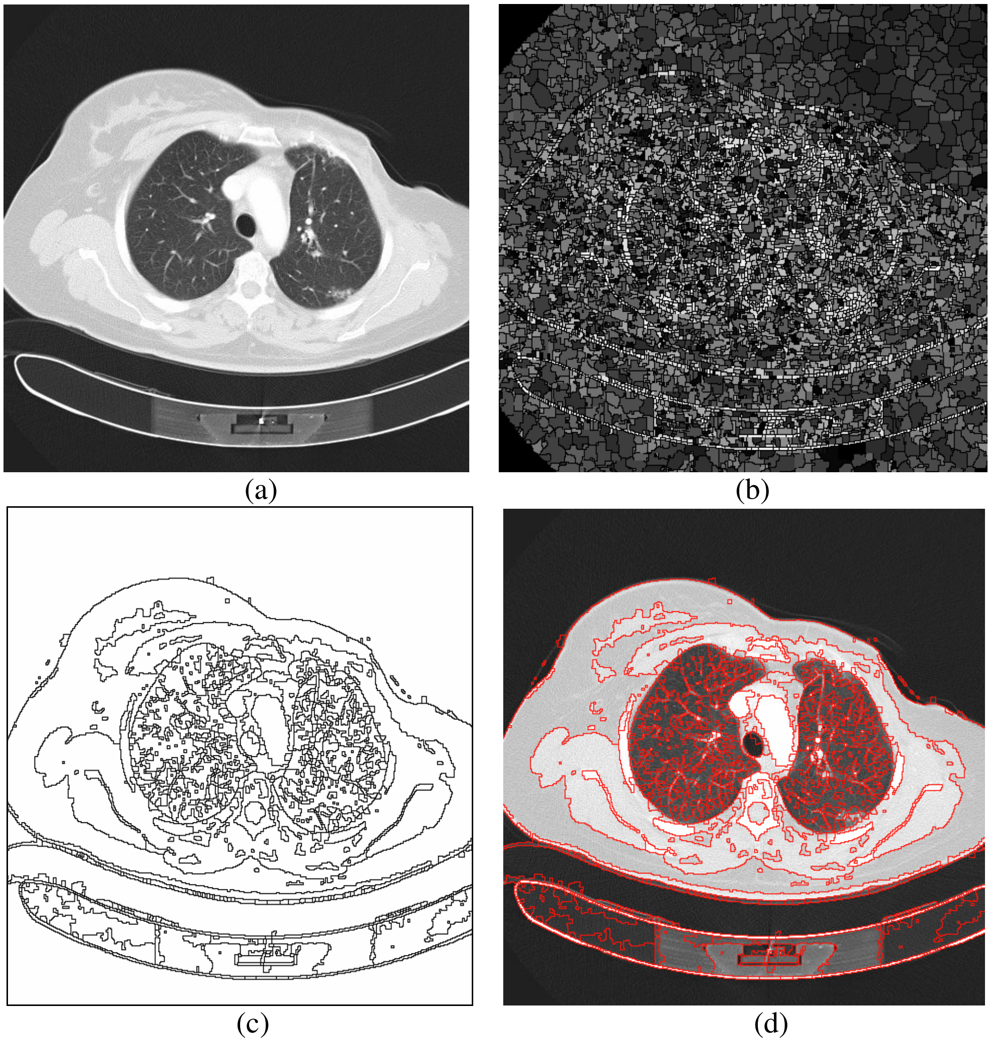
(f)

Figure 3. 4: Result of watershed transformation for the noisy test image (a) original y image, (b) noisy image, (c) result of watershed transformation (d) noisy image overlapped with the watershed lines, (e) result of region merging with Ward's criterion ,(f) noisy image overlapped with the watershed lines



(a) (b) (c) (d) (e) (f)

Figure 3. 5: (a) Result of region merging with mean luminance difference , (b) noisy image overlapped with the watershed lines, (c) result of region merging with Ward’s criterion (d) noisy image overlapped with the watershed lines,(e) result of region merging with border criterion, (f) noisy image overlapped with the watershed lines,



(a) (b) (c) (d)

Figure 3.6: Result of Algorithm-1 for CT image. (a) Original image, (b) watershed transformed image, (c) result of region merging with Ward’s Criterion (d) original image overlapped with the watershed lines

3.1.2.2. Algorithm 2

In Algorithm-2, we integrated K-means clustering in the region merging process to enhance the speed of the Algorithm-1. This algorithm consists of two stages: clustering and final merging.

- **Stage-1(Clustering):** In this stage, sub-regions obtained by watershed transformation are classified by K-means clustering algorithm [63] with respect to their mean values. We applied the mean values of the sub-regions in a feature vector X as input to the clustering process.

Input:

a. X: mean values of sub-regions obtained from Watershed transformation algorithm.

b. N: number of clusters (N=6)

we know that there must be 6 clusters because, we want to segment our image into 6 segments;

1. bones
2. muscle
3. fat
4. heart
5. background
6. lungs

Output: A vector (**Y**) which holds the cluster numbers of corresponding sub-regions.

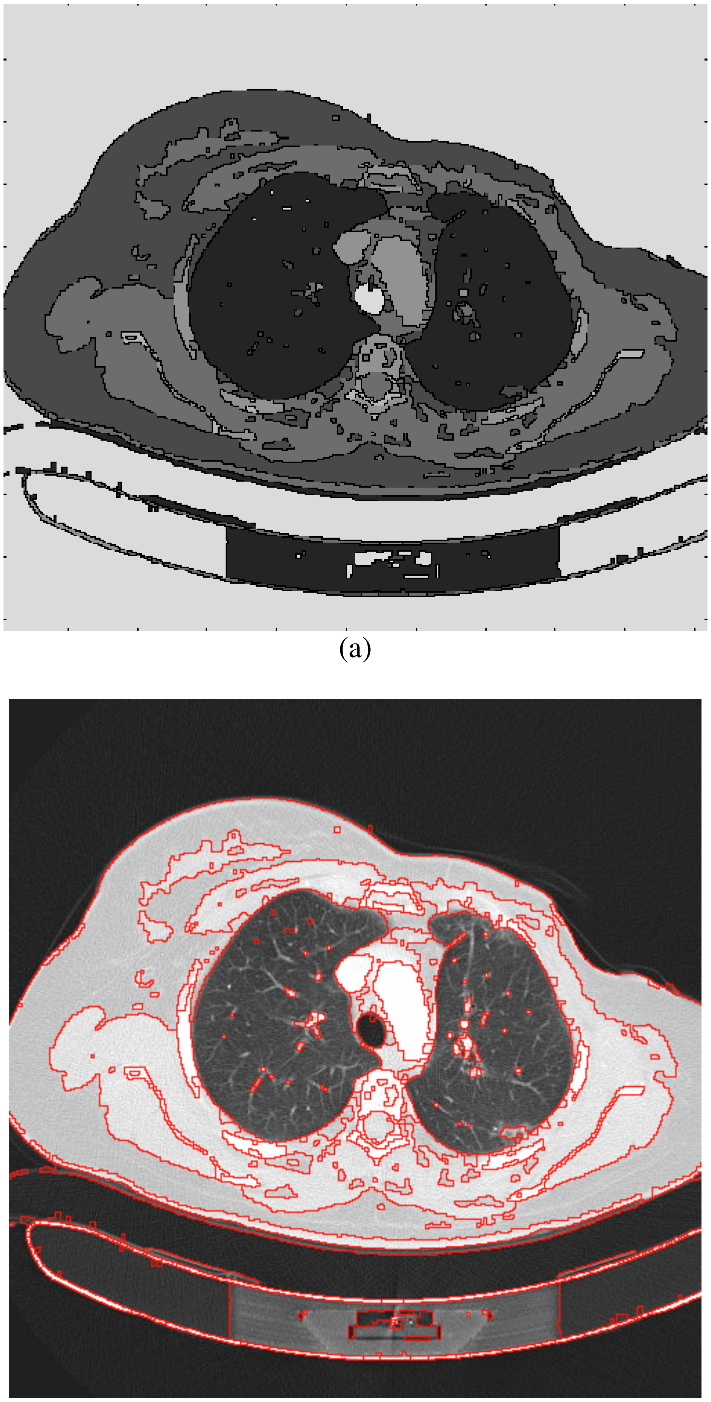
After K-means clustering process, each sub-region is classified in one of six clusters. The segmented image given in Figure-3.7 is then obtained by merging the regions in the same cluster. By observing the Figure-3.7, we can state that most of the similar sub-regions are grouped in the same cluster. Clusters 1 through 6 are shown in Figure-3.7 (a) in a decreasing order of gray-level. In Figure-3.7 (b), the original image overlapped with the resultant watershed lines is given. Numerical results for clusters such as cluster centorids and corresponding total sum of distances are given in Table-3.1.

We can see from Figure-3.7 and Table-3.1 that some regions of tissues are classified in more than one cluster except fat, muscle and lungs. Therefore segmentation of heart and bones can not be managed by clustering of sub-regions totally, therefore we applied a second stage where region merging is applied.

Table 3.1: Results for clustering

Cluster #	Centroid	Contains Regions Belonging to
1	32,08	Background and Lungs
2	989,52	Fat, vessels and Heart
3	1274,71	Muscle, vessels and Heart
4	1400,43	Bones and Heart
5	2113,30	Bones
6	4086	Background

• **Stage-2 (Merging):** In this stage, we applied Algorithm-1 to the results of the clustering process. A new region adjacency graph (RAG_N) is obtained using the results of the clustering process. Finally, we applied Algorithm-1 using the new region adjacency graph $RAG_{N.}$

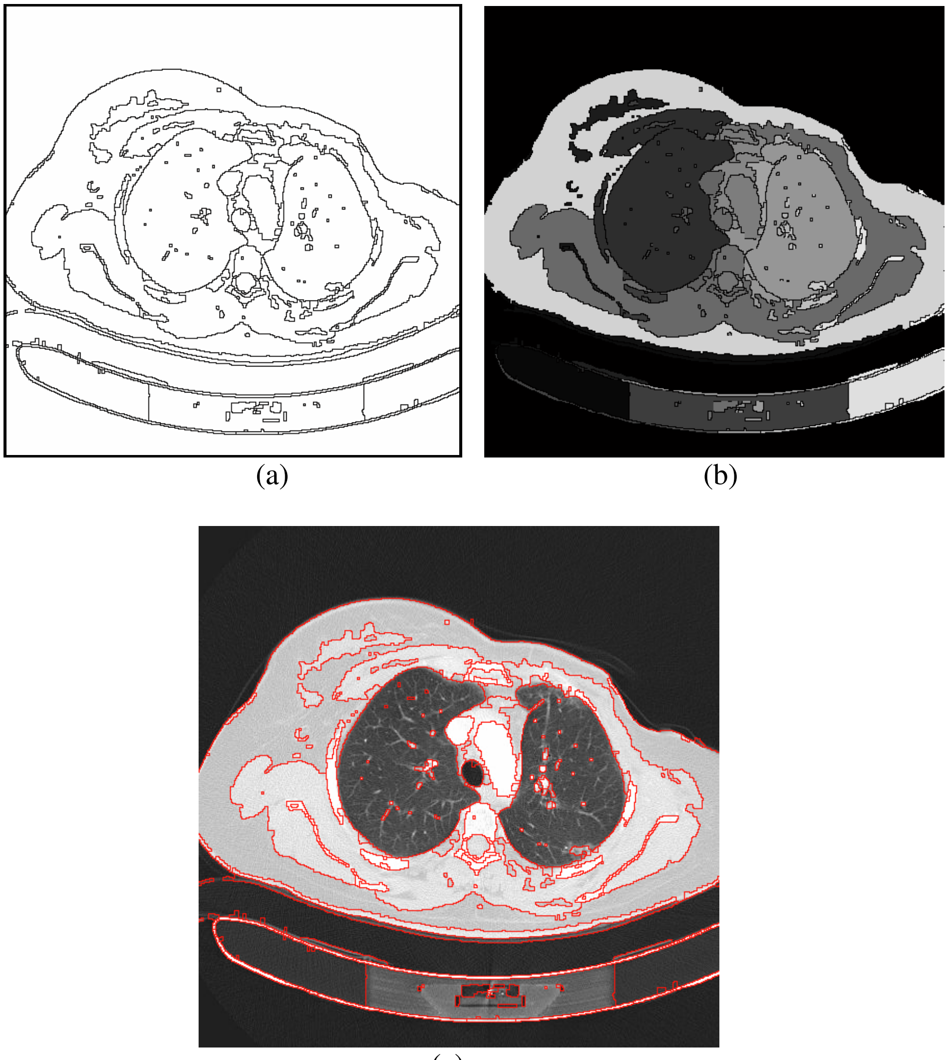


(a)

(b)

Figure 3.7: Result of segmentation image after clustering process (a) segmented image, (b) original image overlapped with resultant watershed lines

Resultant image after stage-2 is shown in Figure 3.8. The input to stage-2 is the watershed transformed image given in Figure 3.7 (b). After applying stage-2 with Ward’s criterion region dissimilarity function, we obtained the resultant image given in Figure 3.8 (b). The total number of sub-regions is decreased to 238, where it was 1005 in Algorithm-1.



(a)

(b)

(c)

Figure 3.8: Result of segmentation after stage-2 (a) resultant watershed lines, (b) segmented image, (c) original image overlapped with watershed lines

Table 3.2: Numerical results for segmentation with WTRM

Segmented Tissue/Organ	Negative Rate Metric (NR)	Total Elapsed Time (sec)
Heart	0.0029	8128
Bones	0.2177	
Muscle	0.06544	
Fat	0.04713	
Lungs	0.0068	

Because WTRM is a totally automatic segmentation algorithm which tries to perform segmentation of all tissues at once because performing segmentation for every tissue is not practical. Because WTRM performs segmentation of all tissues at once, the elapsed time is too long (8128).

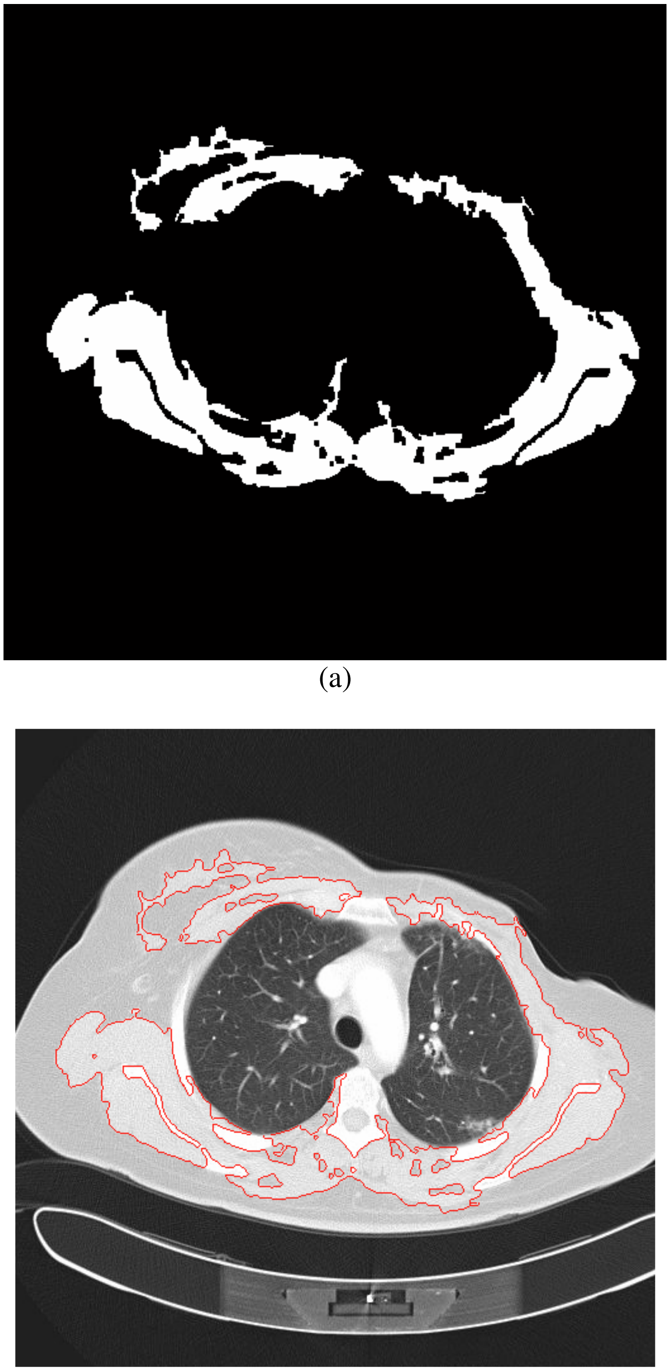
Results of segmentation for muscle and fat tissues, lungs, heart and bones obtained from the resultant image obtained by Algorithm-2 are given in Figures-3.9 to 3.13. After automatic segmentation, there is a user interaction phase which can be thought as a fine-tuning process. In this phase, disconnected regions belonging to the same tissue are classified in the same group. For example, regions belonging to lungs are segmented perfectly, however because they are disconnected, they have different labels. Therefore, regions belonging to lungs must be extracted from the image by user interaction.

By observing Figure-3.9, Figure-3.10 and Figure-3.11 and Table-3.3, we can state that WTRM produces very accurate results for segmentation of lungs, fat and muscle tissues. Moreover, for segmentation of lungs, running only stage-1 in Algorithm-2 is enough.

As we know from the base image, the variance of the gray levels of pixels belonging to the heart is very large. Because of this, regions of heart are participated in different clusters (2, 3, 4) as given in Table-3.1. WTRM could not perform segmentation of heart by region merging completely, instead segmentation of heart is done manually by selecting sub-regions of heart (4 sub-regions for segmented image in Figure-3.8). The resultant image for segmentation of heart, given in Figure 3-12 and the NR value given in Table-3.3 are obtained after user interaction.

We can see from the resultant image given in Figure-3.13 and the corresponding NR value given in Table-3.3, WTRM does not produce accurate results for segmentation of bones. This is because of the weakness of watershed segmentation algorithm in detecting thin structures. Moreover, similar to region growing, user must select at least 11 disconnected sub-regions belonging to the bones for final segmentation, which is not very practical.

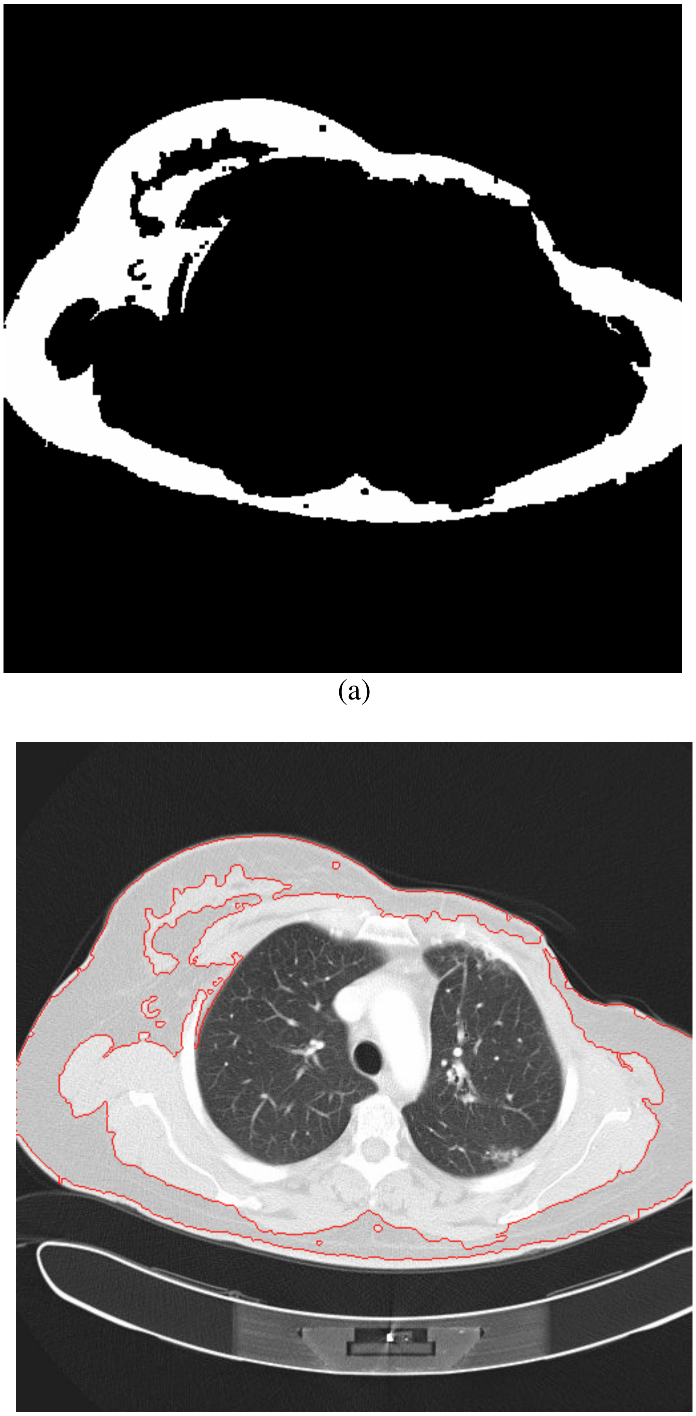
The main drawbacks of this algorithm are; first it needs too much user interaction for re-grouping of disconnected regions belonging to the same tissue. Second, the elapsed time is very long, especially in the region-merging process, and finally it is weak for segmentation of thin structures. On the other hand, segmentation of lungs, fat and muscle tissues can be performed perfectly using this algorithm, and except for the heart, segmentation is performed totally automatically.



(a)

(b)

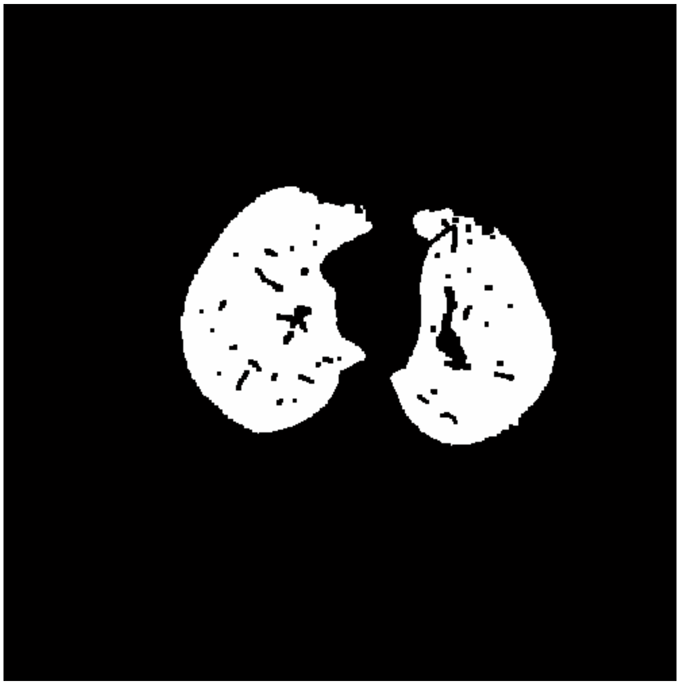
Figure 3. 9: Result of segmentation for muscle tissue (a) obtained binary mask , (b) original image with resultant contours



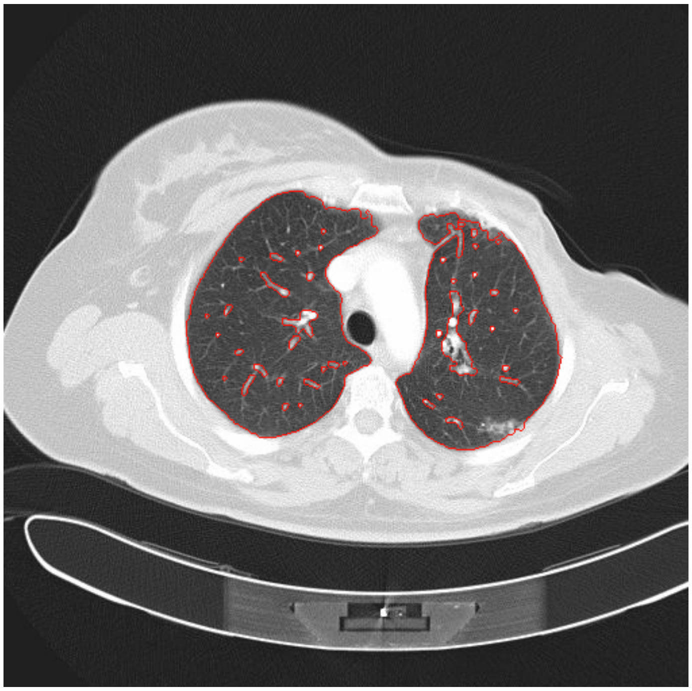
(a)

(b)

Figure 3. 10 : Result of segmentation for fat tissue (a) obtained binary mask, (b) original image with resultant contour for segmentation of fat.

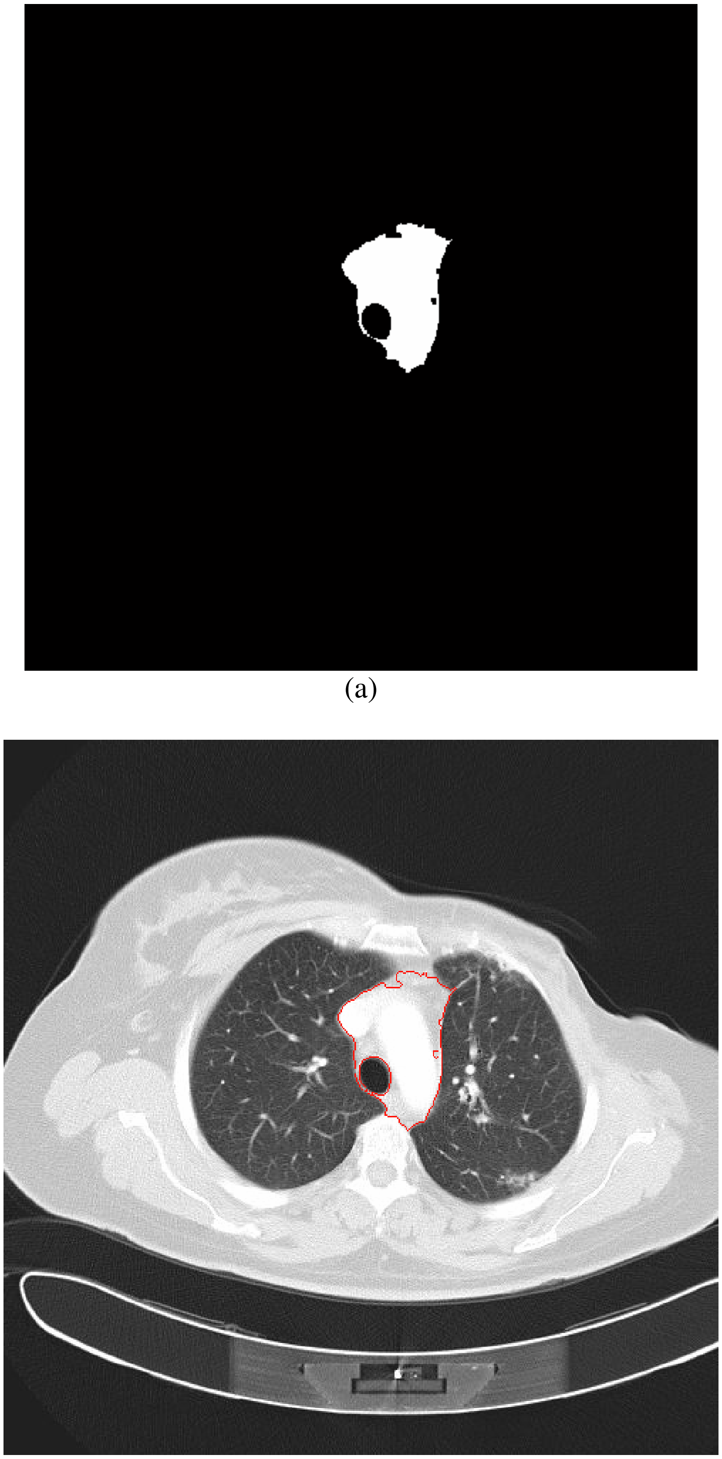


(a)



(b)

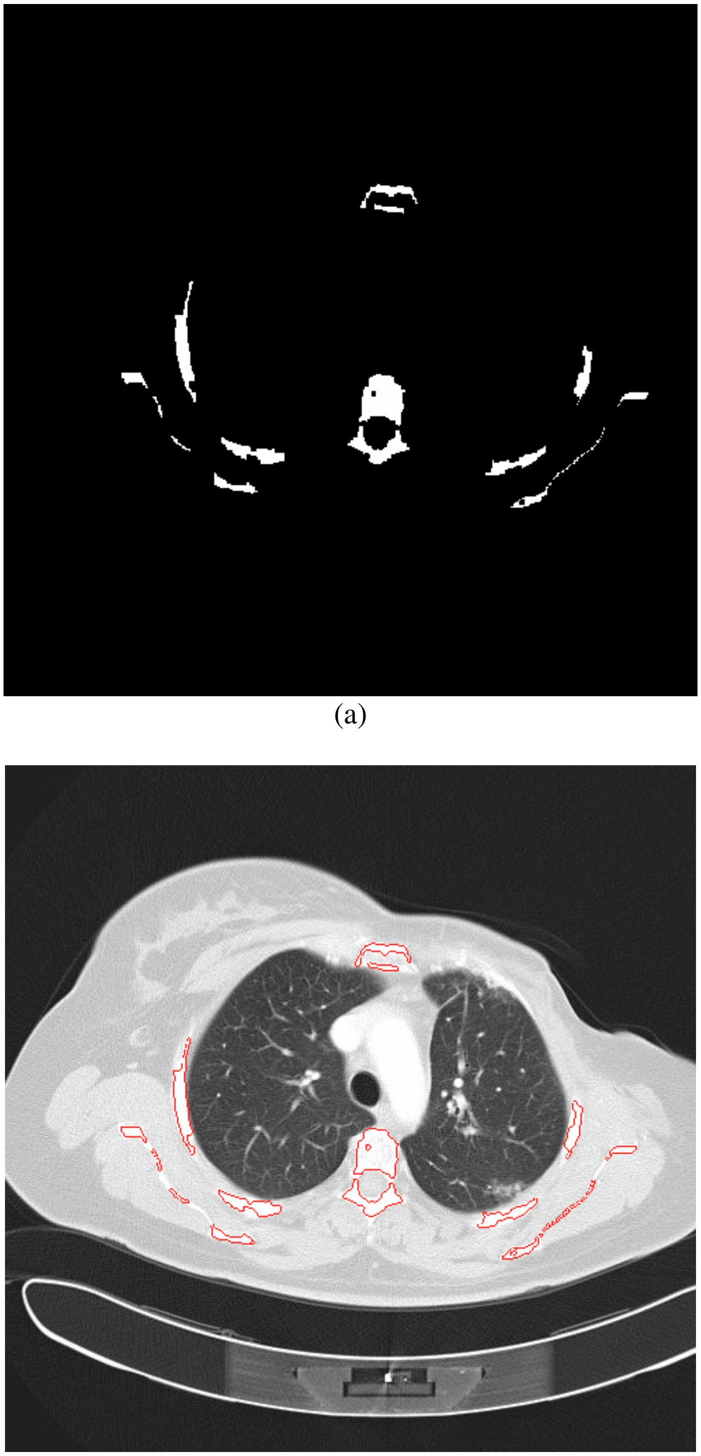
Figure 3. 11 : Result of segmentation for lungs (a) obtained binary mask, (b) original image with resultant contours



(a)

(b)

Figure 3.12: Result of segmentation for the heart (a) obtained binary mask, (b) original image with resultant contour



(a)

(b)

Figure 3.13: Result of segmentation for bones (a) obtained binary mask, (b) resultant contours

CHAPTER 4

COMPARISON OVER SEGMENTATION METHODS

In this chapter, we compared segmentation methods for segmentation of tissues, by utilizing NR and elapsed time as quantitative performance metrics. In addition we evaluated methods in degree of user interaction required and practicality. We compared the performances of segmentation methods for bones, heart, lungs, fat and muscle tissues under specific titles for each tissue, and only practically applicable methods are included for evaluation. By using the results obtained for segmentation of lungs, heart, bones, muscle and fat tissues with some of the practically applicable methods, we also generated 3D surface models for visual inspection of the performance. We used demo version of commercial software (3D DOCTOR) for generating 3D images [58].

4.1. Segmentation of Bones

From the results obtained in Chapter-2 and Chapter-3 we know that parametric deformable models are not practical for segmentation of bones in CT images. Practical methods for segmentation of bones are; watershed transformation and region merging (WTRM), region growing, multi-level thresholding and active contours without edges method. Comparison of these methods for segmentation of bones is given below.

- **Watershed Transformation And Region Merging**

By observing the result of segmentation for bones (Figure-3.13) by WTRM algorithm (Algorithm-2) we can state that segmentation of bones is not very accurate with this algorithm. This can also be seen from Table-4.1 where NR value is 0.2177, and it is very high compared to results of other algorithms. Some parts of bones could not be segmented accurately due to the weakness of watershed transformation in detection of thin structures. Moreover, because WTRM is a totally automatic segmentation algorithm which performs segmentation of all tissues at once, it takes too much time even for one image.

- **Region Growing**

By observing Figure-2.2.6 and Table-4.1, we can state that efficient results can be obtained for segmentation of bones using region growing. Moreover, the elapsed time is the shortest one. On the other hand, this method needs user interaction for planting seeds in the image.

- **Multi-level Thresholding**

By observing Figure-2.1.8 and Table-4.1, we can state that multi-level thresholding produces accurate results for segmentation of bones with a very low NR value. On the other hand it requires too much time even for one image.

- **Active Contours without Edges**

We can use active contours without edges method with three level set functions for segmentation of bones. However, by looking at Figure-2.4.42 and NR value in Table-4.1, we can state that segmentation of bones can not be performed with this method.

By observing Table-4.1, we can state that lowest NR value for segmentation of bones is given by region growing method. On the other hand, by taking the degree of user interaction required into account, we can state that this method is not very practical. Multi-level thresholding also produces accurate results for segmentation of bones without requiring user defined seed points, but the elapsed time is very long compared to region growing method.

We used region growing to all of the CT slices in our dataset and obtained a 3D model for the bones. Various views of this 3D model are shown in Figure 4.1 and Figure-4.2

Table 4. 1: Numerical Results for segmentation of bones

Method	Negative Rate Metric (NR)	Elapsed Time (sec)
Watershed Transformation and Region Merging	0.2177	8128
Region Growing	0.00037	37.4
Multi-level thresholding	0.0077	253
Active Contours without Edges (with three level set functions)	0.14380	101.2

4.2. Segmentation of Lungs

Practically applicable methods for segmentation of lungs are watershed transformation and region merging, parametric deformable models (GVF Snakes) region growing, multi-level thresholding, and active contours without edges method. Comparison of these methods for segmentation of lungs is given below.

- **Watershed Transformation And Region Merging**

By observing Figure-3.11 and Table-4.2, we can state that WTRM is a very efficient tool for segmentation of lungs. Moreover, because running only stage-1 in Algorithm-2 is enough for segmentation of the lungs the elapsed time is short compared to the other methods..

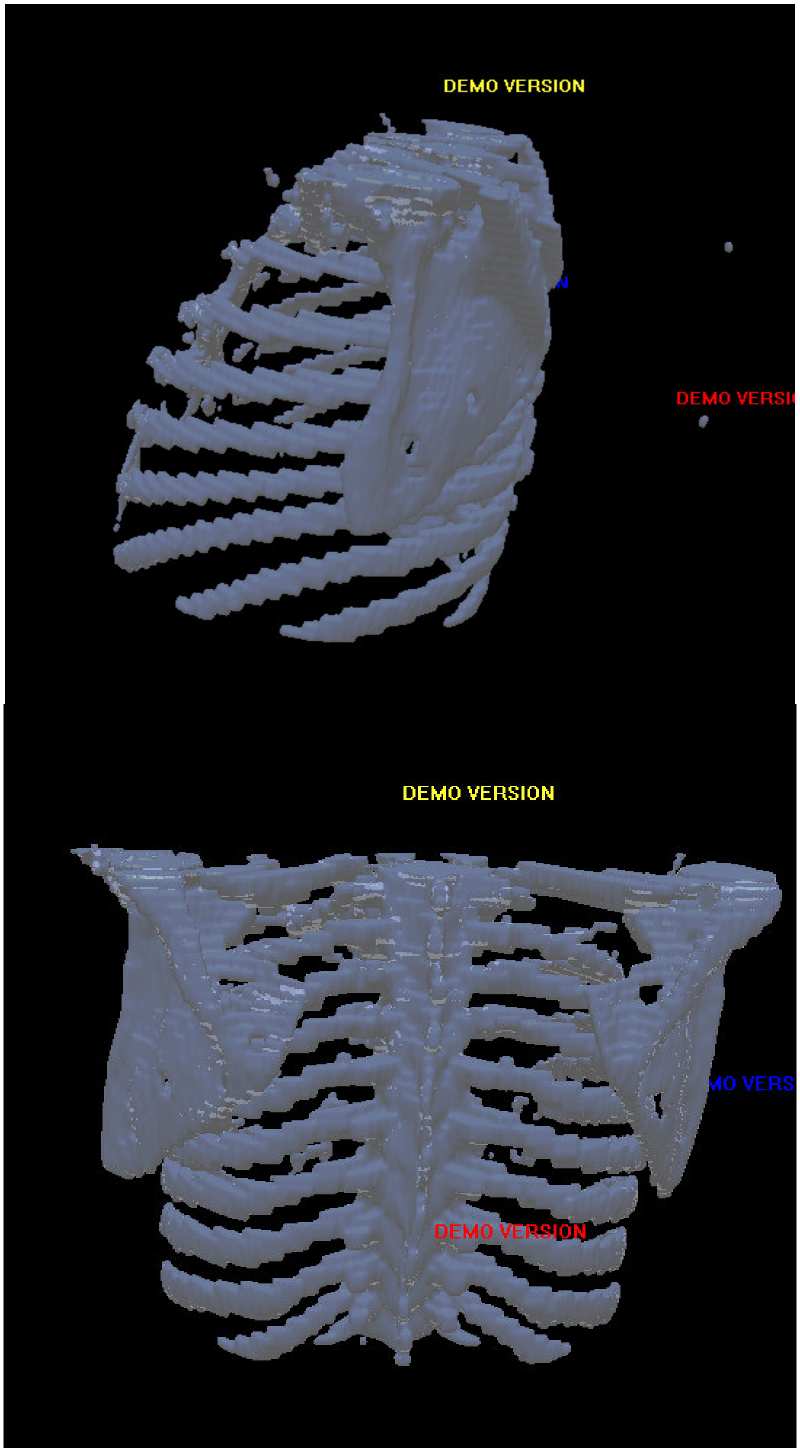


Figure-4.1: 3D surface model for bones-1

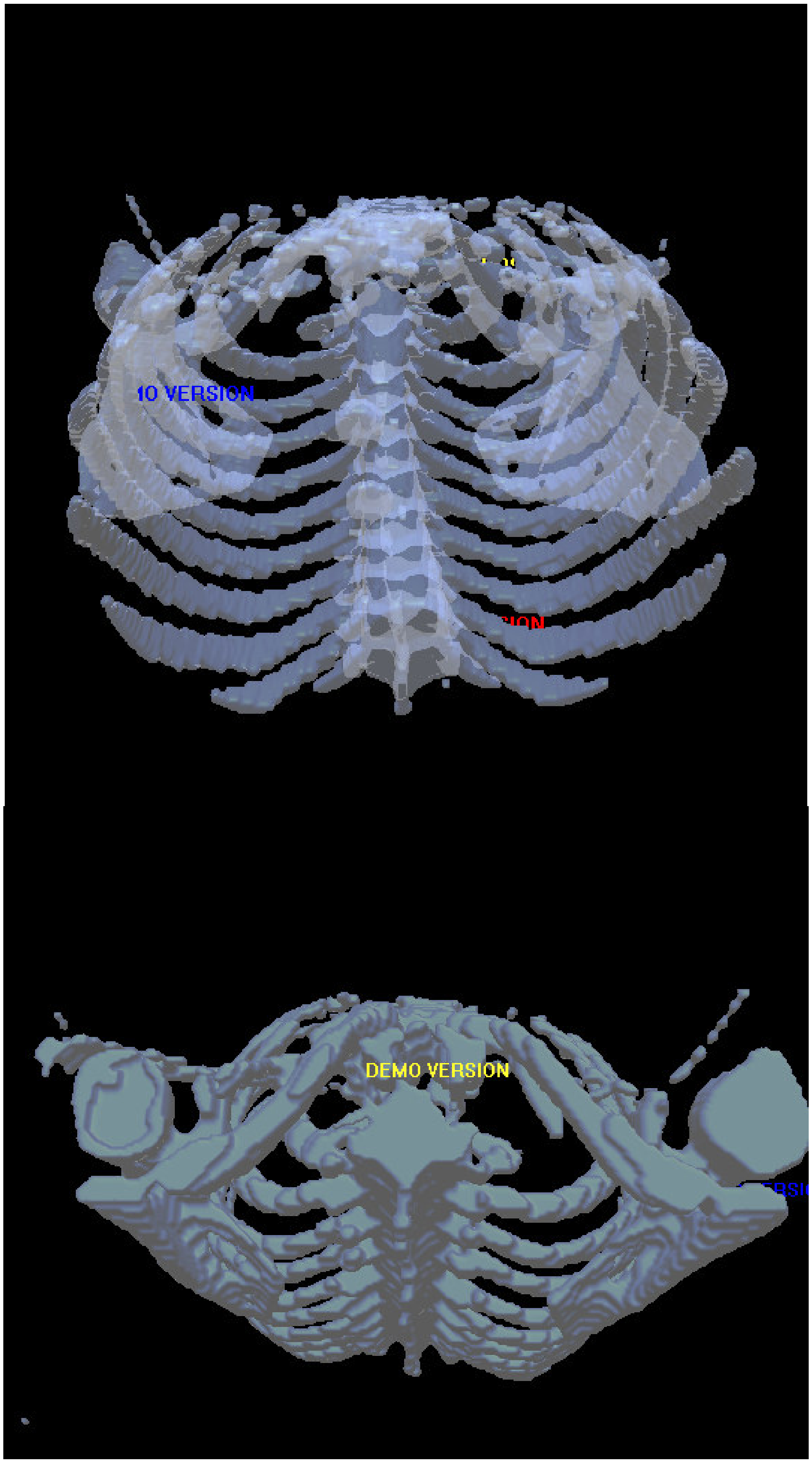


Figure-4.2: 3D surface model for bones-2

- **Parametric Deformable Models (GVF Snakes)**

By observing Figure-2.4.8(b) and Table-4.2, we can state that using GVF snakes for segmentation of lungs does not produce accurate results compared to other methods. Moreover, because right and left sides of the lungs are not connected, one must always use two different snakes, and locate each snake very carefully to avoid converging to wrong edges.

- **Region Growing**

By observing Figure-2.2.8 and Table-4.2, we can state that region growing is a very efficient segmentation method for segmentation of lungs, with the lowest NR value. It needs user interaction for planting seeds, but the number of seeds is only two for every image in the volume dataset. On the other hand, region growing is very sensitive to noise.

- **Active Contours without Edges**

By observing Figure-2.4.43 and Table-4.2, we can state that active contours without edges method with three level set functions produces acceptable results for segmentation of lungs within an acceptable time period. Because active contours without edges method with three level set functions can perform segmentation of lungs, bones, fat and muscle tissues automatically and fastly, one can use this method for segmentation of lungs while simultaneously performing segmentation for other tissues and organs.

- **Multi-level Thresholding**

By observing Figure-2.1.7 and Table-4.1, we can state that, multi-level thresholding produces acceptable results for segmentation of lungs, with a low NR value. On the other hand it requires too much time even for one image.

Results obtained for segmentation of lungs by WTRM and region growing algorithms are comparable, because WTRM does not require user defined seed points, we decided to use WTRM algorithm for segmentation of lungs. In Figure-4.3 and Figure-4.4, a 3D surface model for lungs with trachea is generated by using the segmented images obtained with WTRM method.

Table 4. 2: Numerical Results for segmentation of lungs

Method	Negative Rate Metric (NR)	Elapsed Time (sec)
Parametric Deformable models (GVF Snake)	0.26234	85.6
Region Growing	0.00011	52.4
Watershed Transformation and Region Merging	0.0068	42
Multi-level thresholding	0.1218	253
Active Contours without Edges (with three level set functions)	0.1021	101.2

4.3. Segmentation of Fat Tissue

Practically applicable methods for segmentation of fat tissue are watershed transformation, active contours without edges algorithm and region growing. Comparison of these methods for segmentation of fat tissue is given below.

- **Watershed Transformation And Region Merging**

By observing Figure-3.10 and Table-4.3 we can state that using WTRM produces accurate results for segmentation of fat tissue. On the other hand, because WTRM performs segmentation of all tissues and organs simultaneously the elapsed time is very long.

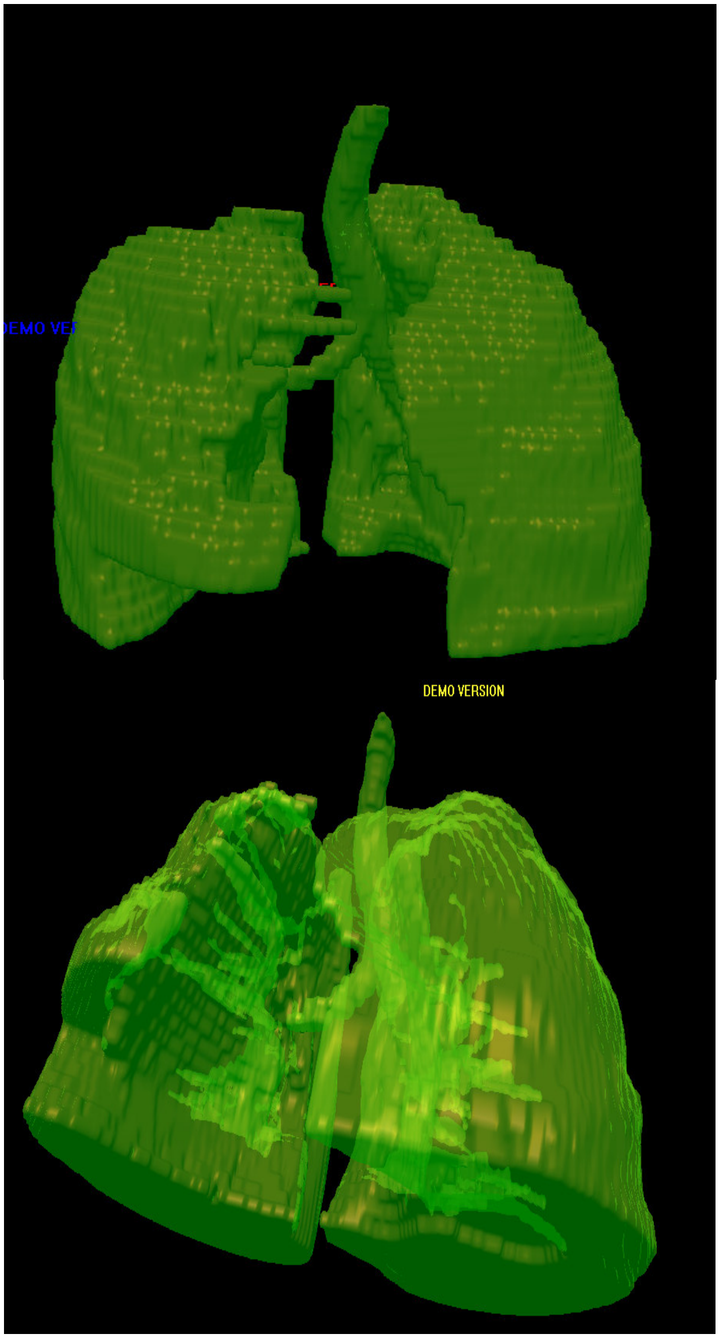


Figure-4.3: 3D surface model for lungs-1

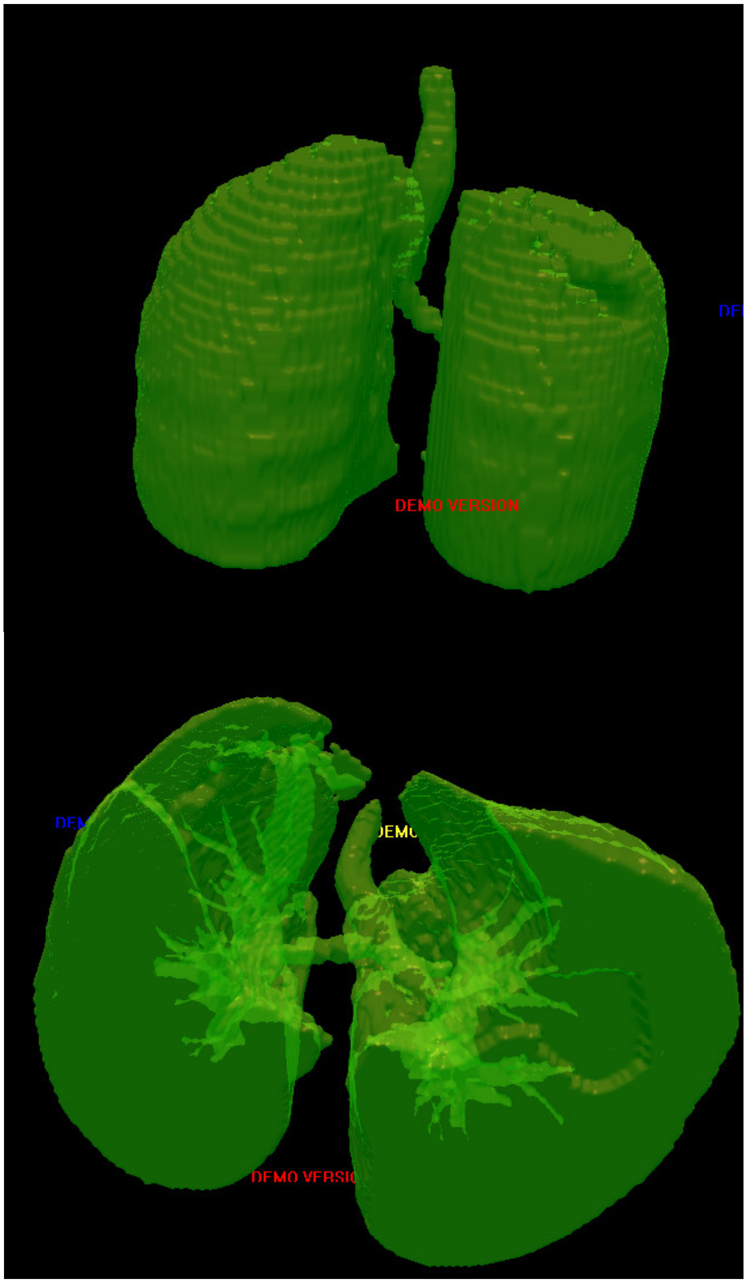


Figure-4.4: 3D surface model for lungs-2

- **Region Growing**

By observing Figure-2.2.4(e) and Table-4.2, we can state that region growing produces accurate results for segmentation of fat tissue in an acceptable time period.

On the other hand, region growing algorithm needs user interaction for planting seeds.

- **Active Contours without Edges**

By observing Figure-2.4.40 and Table-4.2, we can state that active contours without edges method with three level set functions is a very efficient tool segmentation of fat tissue, with the lowest NR value.

Because it does not require a seed point and produces the most accurate results in an acceptable time period, we used active contours without edges method for segmentation of fat tissue. The 3D surface model of fat tissue shown in Figure-4.5 and Figure-4.6, is generated by using the segmented images obtained with active contours without edges method.

Table 4.3: Numerical Results for segmentation of fat tissue

Method	Negative Rate Metric (NR)	Elapsed Time (sec)
Region Growing	0.10473	85.6
Watershed Transformation and Region Merging	0.04713	8128
Active Contours without Edges (with three level set functions)	0.00740	101.2

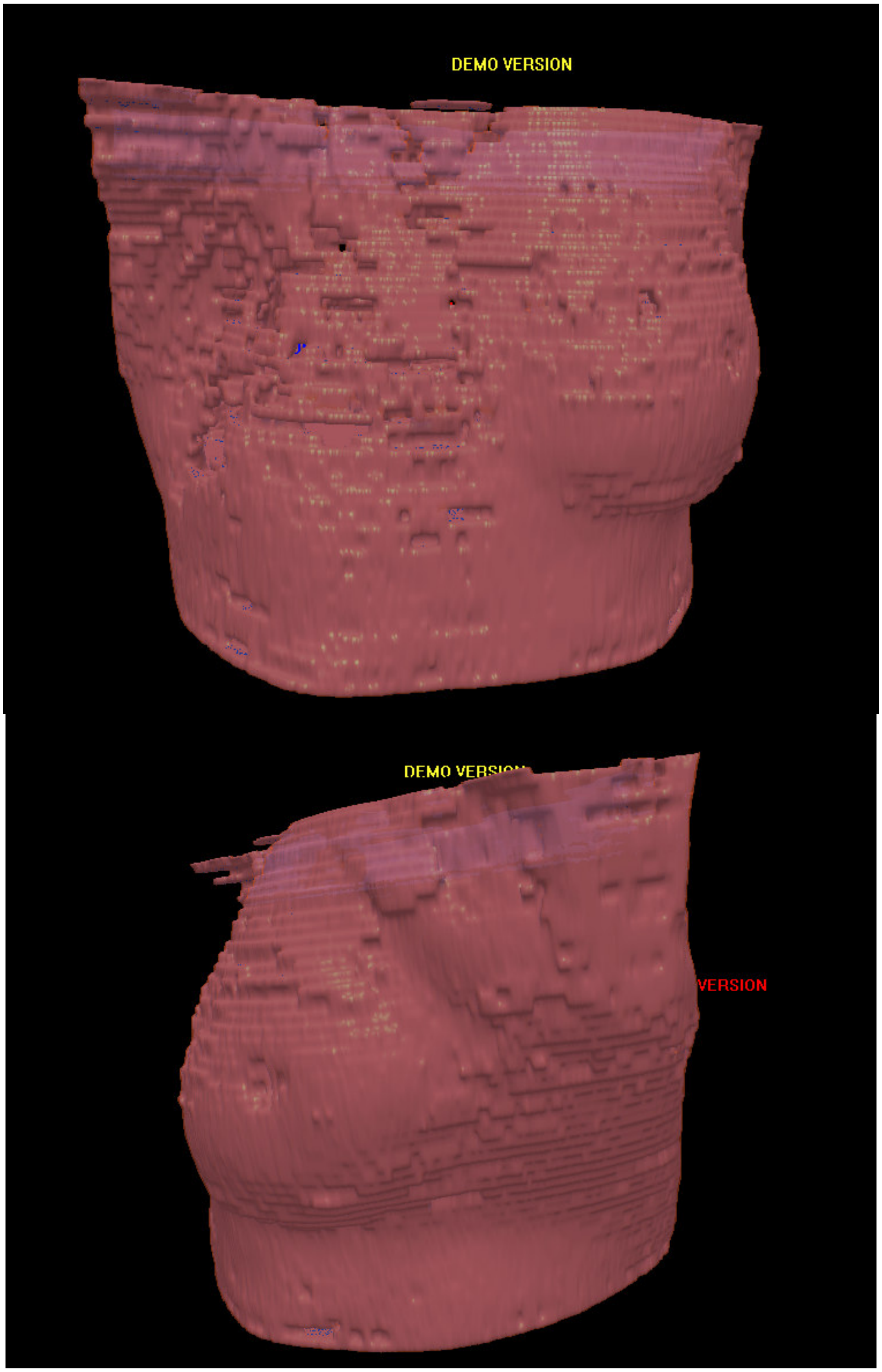


Figure-4.5: 3D surface model for fat tissue-1

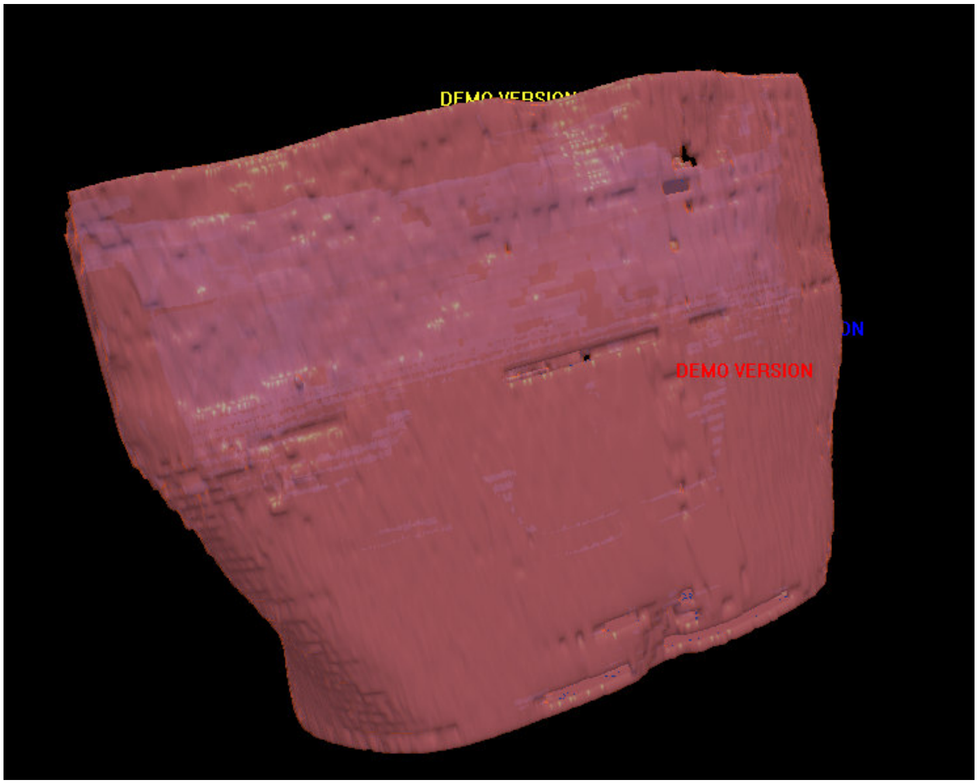


Figure-4.6: 3D surface model for fat tissue-2

4.4. Segmentation of Muscle Tissue

Applicable methods for segmentation of muscle tissue are; WTRM, region growing and active contours without edges algorithm. Numerical results for segmentation of muscle tissue with these methods are given in Table-4.4. Comparison of these methods for segmentation of muscle tissue is given below.

- **Watershed Transformation And Region Merging**

By observing the Figure-3.9 and Table-4.4, we can state WTRM produces accurate results for segmentation of muscle tissues. On the other hand, the elapsed time is very long compared to other methods.

- **Region Growing**

By observing the Figure-2.2.4(b) and Table-4.4, we can state that region growing is an efficient tool for segmentation of muscle tissues with the lowest NR value. On the other hand, because it needs user interaction for planting a lot of seeds in each frame, it is not very practical.

- **Active Contours without Edges**

By observing Figure-2.4.39 and Table-4.2, we can state that active contours without edges method with three level set functions is a very efficient tool for segmentation of muscle tissue with a very low NR value, and with an acceptable elapsed time with respect to WTRM.

Table 4.4: Numerical Results for segmentation of muscle tissue

Method	Negative Rate Metric (NR)	Elapsed Time (sec)
Region Growing	0.03921	76.8
Watershed Transformation and Region Merging	0.06544	8128
Active Contours without Edges with three level set functions	0.04550	101.2

We can easily state in the sense of NR values, results obtained with each method is comparable. However, because it does not require a user defined seed region and performs segmentation of muscle tissue in an acceptable time period, we preferred to use active contours without edges method to obtain the 3D surface model of muscle tissue shown in Figure-4.7 and Figure-4.8.

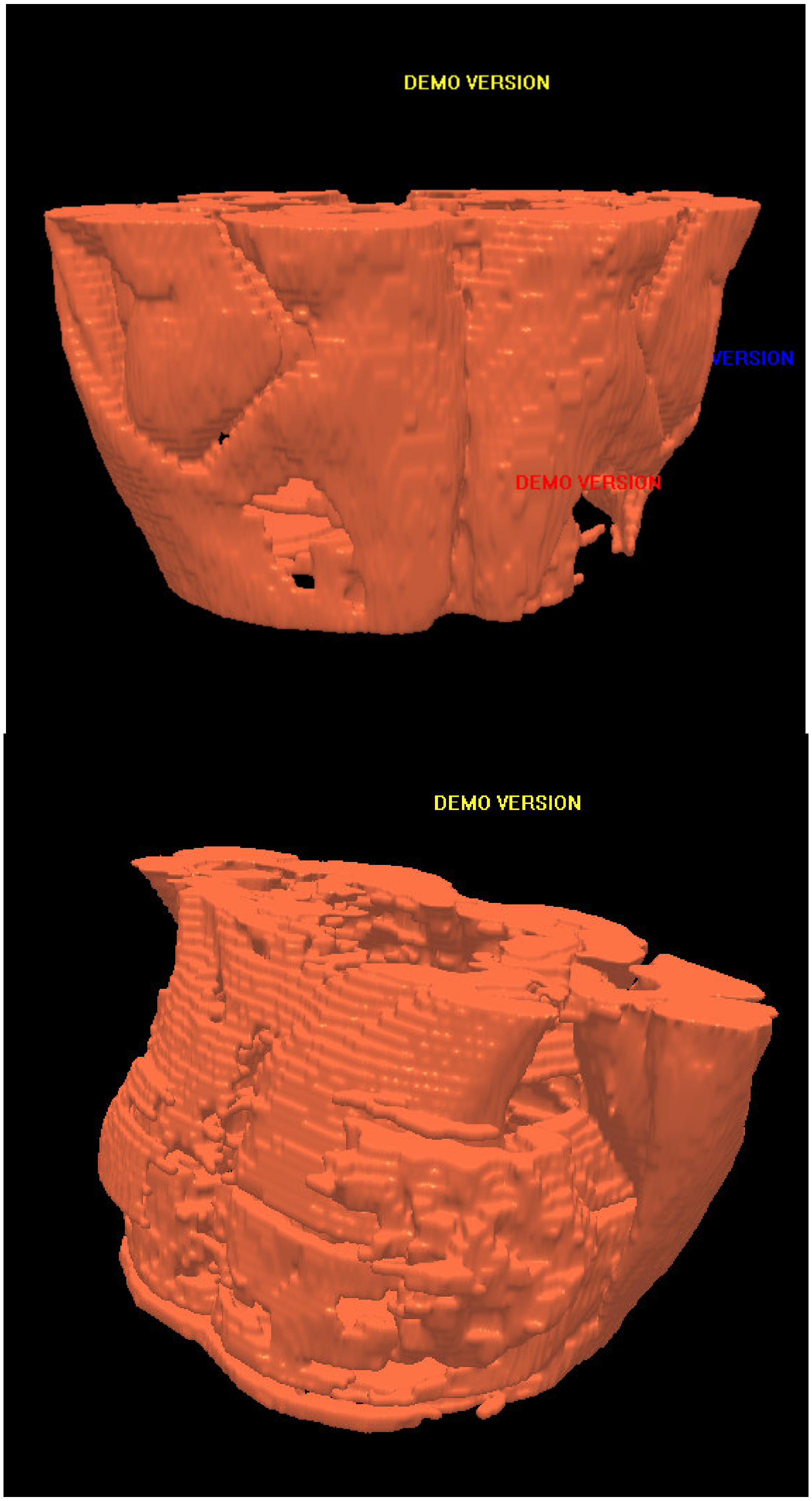


Figure-4.7: 3D surface model for muscle tissue-1

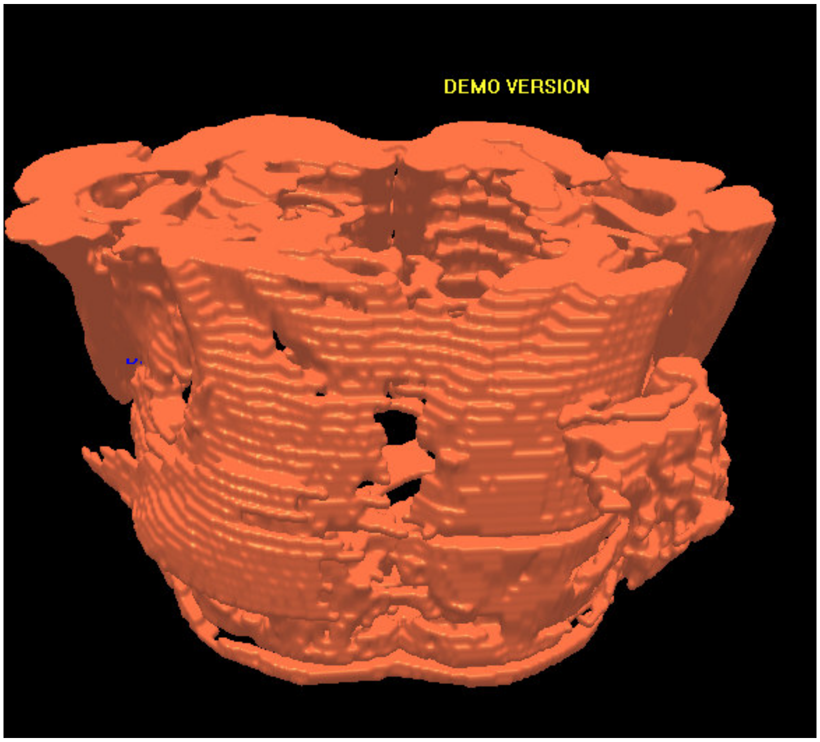


Figure-4.8: 3D surface model for muscle tissue-2

4.5. Segmentation of Heart

Since the heart as an organ contains various tissues like heart muscle and fat, it is represented with different gray levels and has a large variance, which can be observed in the original image and the base image (see Figure-1.1 and Figure-1.6 respectively). Moreover, the mean gray level of the heart shows serious differences in different slices, in contrast to the other organs and tissues. Due to the above given reasons, segmentation of the heart as an organ is the most challenging problem in our study. Therefore, obtaining a perfectly working method for segmentation of heart is a very difficult task. Below we compared the results obtained for segmentation of heart using various segmentation methods.

Because of the high variance of the heart, region based methods generally fails in segmentation of heart. Deformable methods are more useful for this task. However as in GVF Snakes, suggesting an initial contour is a hard task. Applicable methods for segmentation of the heart tissue; WTRM, active contours without edges, and GVF snakes. Comparison of these methods for segmentation of heart is given below. Numerical results for segmentation of the heart are given in Table-4.5.

- **Watershed Transformation And Region Merging**

WTRM can not perform segmentation of heart by region merging completely, instead segmentation of heart can be done manually by selecting sub-regions of heart. By observing Figure-3.12 and Table-4.5, we can state that WTRM is a very efficient tool for segmentation of the heart with the lowest NR value. The drawback of this method for segmentation of the heart is: it requires user interaction for merging regions belong to the heart, and it takes extremely long time compared to other methods.

- **Region Growing**

Because of the high variance in the gray-level value and low-contrast between its neighboring tissues (bones and muscle tissues), segmentation of the heart is not very accurate using region growing. This can be observed from Figure-2.2.7 and Table-4.5.

- **Parametric Deformable Models (GVF Snakes)**

By observing the resultant image in Figure-2.4.7 and Table-4.5, we can state that we can accurately segment the heart using GVF snakes. This is because parametric deformable models do not take the variance or the mean value of the region into account, and therefore using GVF snakes for segmentation of the heart looks appealing. However, GVF snakes are not very practical, because the geometry in torso is variable and therefore we must suggest an initial contour for every image in the volume dataset.

- **Active Contours without Edges**

Similar to WTRM, active contours without edges method can not perform segmentation of the heart automatically; instead a user interaction is required for selecting sub-regions belonging to the heart after primitive segmentation using level sets. By observing Figure-2.4.40 and Table-4.5, we can state that active contours without edges method with three level set functions is a very efficient tool for segmentation of the heart with a very low NR value. The drawback of this method for segmentation of the heart is: it requires user interaction for merging regions belonging to the heart.

We can easily state that results of WTRM and active contours without edges algorithm for segmentation of the heart are comparable. Therefore, we used both of these methods for generating a 3D surface model of the heart. In Figure-4.9 and Figure-4.10, a 3D surface model for heart is generated by using the segmented images obtained with active contours without edges method. The 3D surface models in Figure-4.11 and Figure-4.12 are obtained by using WTRM.

Table 4.5: Numerical Results for segmentation of the heart

Method	Negative Rate Metric (NR)	Elapsed Time (sec)
Region Growing	0.22029	28.4
Watershed Transformation and Region Merging	0.0029	>8128
Active Contours without Edges with three level set functions	0.00892	101.2
Parametric Deformable Models(GVF Snakes)	0.09819	85.6

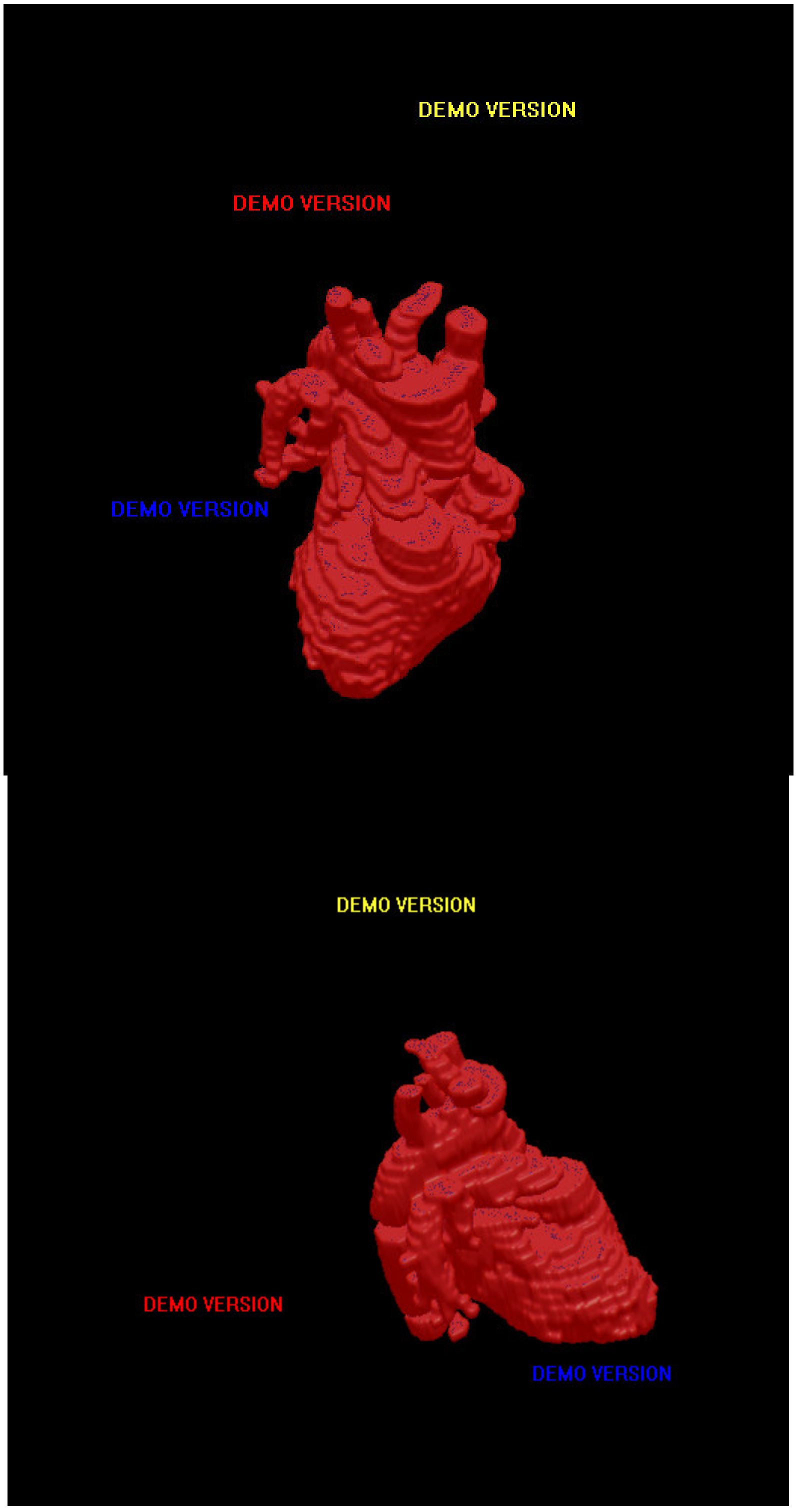


Figure-4.9: 3D surface model for the heart-1 (generated from segmented images obtained by active contours without edges method)

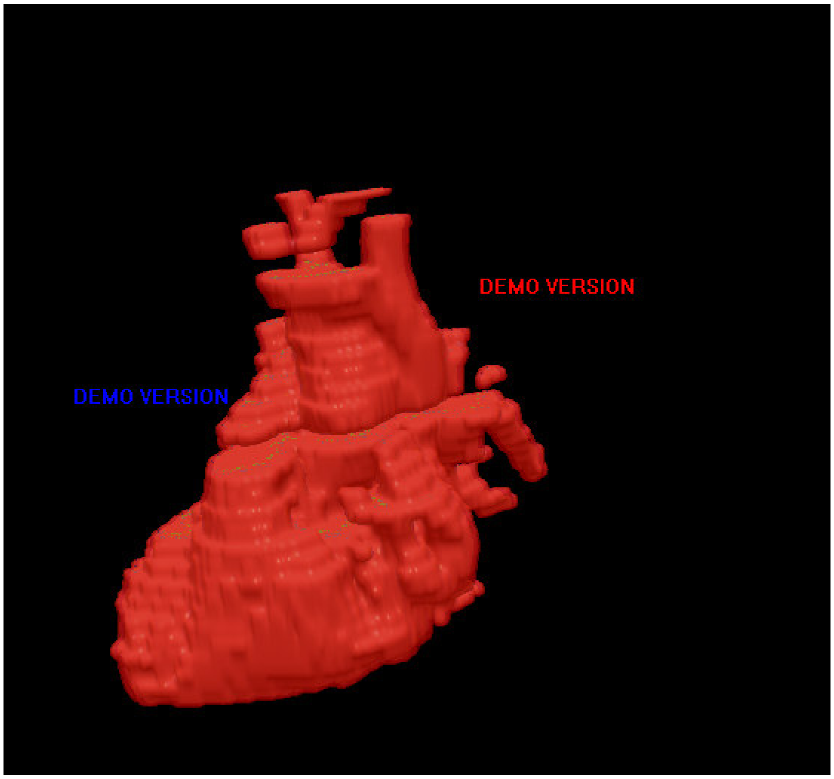


Figure-4.10: 3D surface model for the heart-2 (generated from segmented images obtained by active contours without edges method)

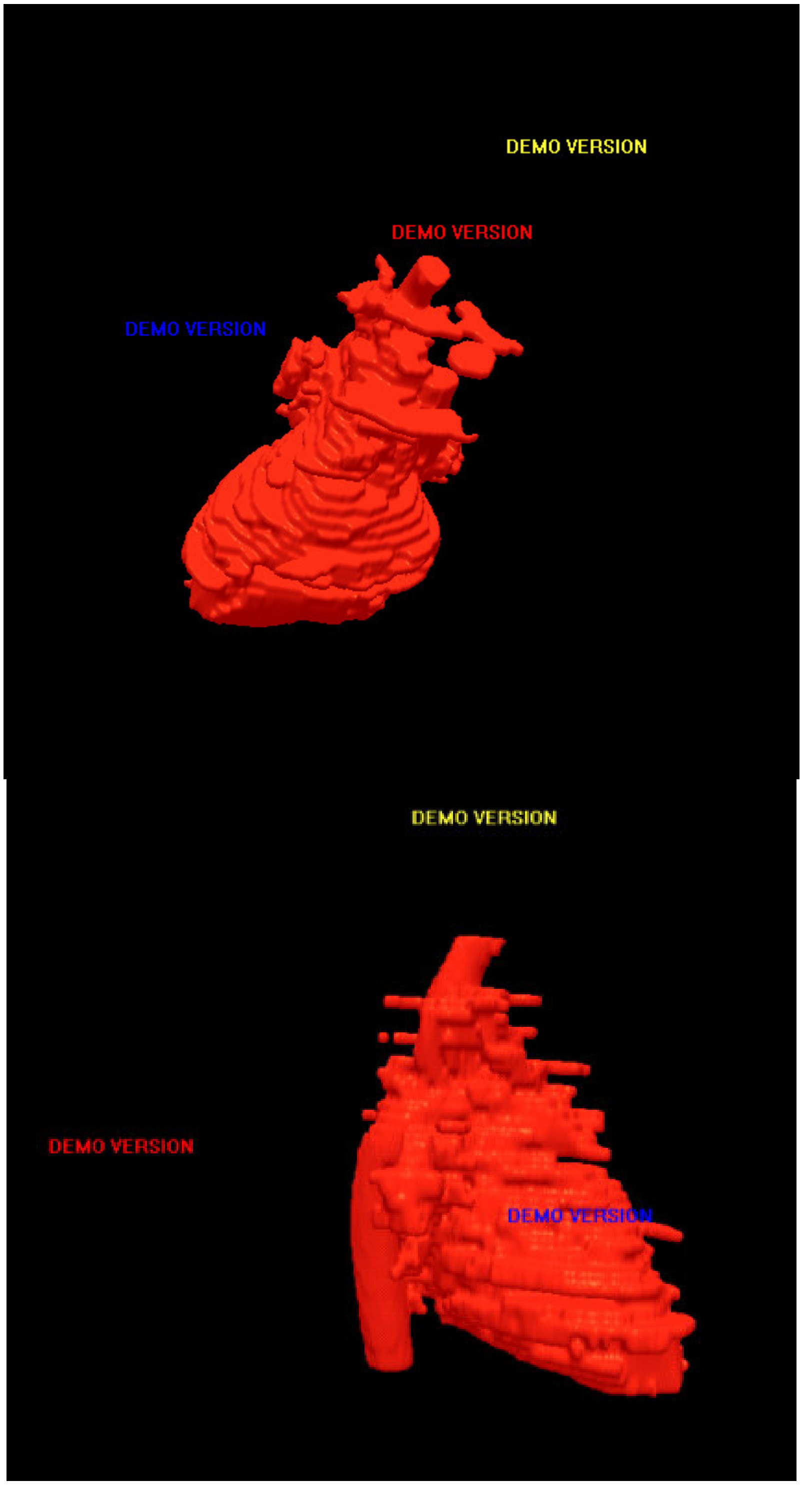


Figure-4.11: 3D surface model for the heart-3 (generated from segmented images obtained by watershed transformation and region merging method)

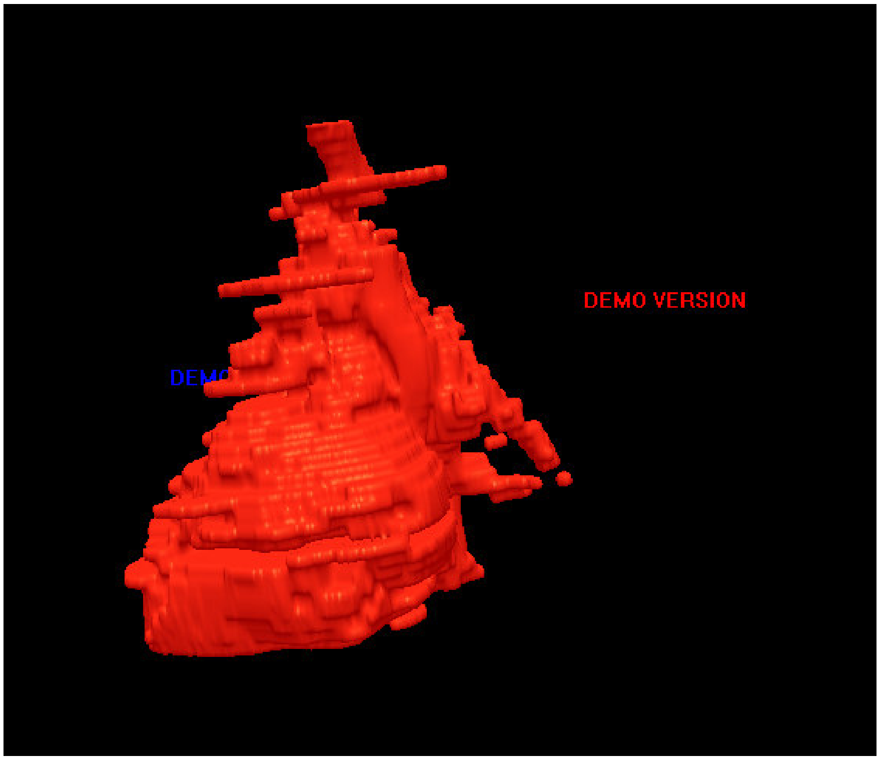


Figure-4.12: 3D surface model for the heart-4 (generated from segmented images obtained by watershed transformation and region merging method)

CHAPTER 5

DISCUSSION AND CONCLUSIONS

In this study, we focused on segmentation of main tissues and organs in the human torso using fundamental medical image segmentation algorithms. These algorithms are applied to CT images and their performances are evaluated for each tissue in the torso.

Segmentation results given in Chapter-2 and Chapter-3 shows that none of the segmentation methods can give a complete solution for segmentation of all the tissues in torso. Instead, each method has some advantages as well as some drawbacks, and produces accurate results for segmentation of a specific tissue or tissues.

Multi-level thresholding provides accurate results for segmentation of the bones with three threshold values, however using more than three threshold values requires an extremely long time, and it does not take into account the spatial characteristics of the image. Therefore, it is very sensitive to noise and produces scattered and disconnected regions. On the other hand, it produces accurate results for segmentation of bones.

Parametric deformable models, like GVF snakes are very efficient segmentation methods and widely used in segmentation applications. However, for segmentation of most of the tissues in the torso, this method is not very practical. For example, parametric snakes can not split or merge during deformation, and therefore segmentation of bones is not practical with this method. Moreover, parametric deformable methods are not very efficient in obtaining boundary concavities. Therefore, using parametric deformable models for segmentation of muscle and fat tissues is not appropriate. On the other hand, since parametric deformable models deform under the effect of internal and external forces, and do not depend on the variance or mean value, these methods can be used for segmentation of the heart. However, an initial contour and some parameters must be suggested by the user for each image.

We implemented active contours without edges method, which is classified under geometric deformable models. We can state that because active contours without edges method with one level set function divides the image into only two disjoint regions, it is not possible to use it for segmentation of fat, muscle, heart, bones or lungs separately. Therefore, we extended this method by using more than one level set function to partition the image into more than two disjoint regions. By using three level set functions, we performed the complete division of the image into eight disjoint regions, and obtained accurate results for segmentation of the heart, lungs, fat and muscle tissues. For segmentation of the bones, obtained results were acceptable. We can easily state that active contours without edges method with three level set functions provides complete solution for segmentation of all of the tissues except bones.

In our study, watershed transformation and region growing algorithms are implemented as region based methods. Implemented region-growing method is based on seeded region growing, and integrated with edge detection. By integrating region growing and edge detection, flooding of growing process through salient edges is prevented, and satisfactory results within relatively acceptable computation times are obtained for every tissue except heart. Region growing method produced best results for segmentation of bones, lungs and muscle tissue, and comparable results for other tissues in the sense of NR values. The drawback of this method is that it requires user defined seed regions and is very sensitive to noise.

Watershed transformation is a very useful tool because it produces a complete division of the image in separated regions. However, these primitive regions must be classified further to obtain a complete segmentation of each tissue. Therefore, we applied watershed transformation and region merging algorithm to merge similar regions using a region dissimilarity function. However, the region merging process takes too much time and sometimes fails to merge regions belonging to the same tissue. Therefore, we used K-means clustering to obtain a pre-classification before region merging process, in order to decrease the computation time. Nevertheless, this segmentation method requires too much computation time. On the other hand, by using this method we can automatically obtain satisfactory results for segmentation of all tissues except bones.

There are still unresolved problems and room for improvements in this study. In future work following studies can be performed for further improvements:

- Different approaches may be applied to overcome the problem of obtaining a complete solution for segmentation of the heart. A possible solution can be obtained by using a hybrid method which integrates parametric deformable models, region growing and intensity based segmentation methods. In this method, we can use the advantage of one method to compensate other's disadvantage. For example, by using seeded region growing, an initial segmentation can be obtained. Moreover, at the borders of the resultant region, a GVF snake can be initialized. In this way we can get rid of the manual initialization of the snake, and the dependency to image statistics (variance, mean value) by using GVF snakes.

- Clustering based methods can be implemented, and resultant unconnected regions can be connected by integrating these methods with some other methods to obtain useful hybrid methods.

- Deformable models and region growing algorithm can be implemented directly in 3D [63, 59, 64,65].

- Active contours without edges method can be implemented by using more than three level set functions, where we can divide the image into four or more disjoint regions [66].

- In watershed transformation, oversegmentation can be decreased by using markers for initially starting the flooding process [67].

- Because heart is a dynamic organ that deforms during pumping blood to the body, segmentation of this organ can be managed more efficiently using 4D CT images, which are obtained in one cardiac cycle and has time information in addition to all three spatial dimensions.

REFERENCES

1. Rolf Adams and Leanne Bischop, “Seeded Region Growing,” IEEE *Transactions on Pattern Analysis and Machine Intelligence,* Vol. 16. No.6, June 1994.
2. G.B. Coleman and H.C. Andrews. Image segmentation by clustering. *Proc. IEEE*, 5:773–785, 1979.
3. Andrew Menhert and Paul Jackway, “An Improved Sedded Region Growing Algortihm,” *Pattern Recognition Letters,* Vol: 18, (1997),1065-107
4. G.B. Coleman and H.C. Andrews. Image segmentation by clustering. *Proc. IEEE*, 5:773–785, 1979.
5. J.C. Dunn. “A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters,” *Journal of Cybernetics*, 3:32–57, 1973.
6. Tianhu Lei and Wilfred Sewchand. “Statistical approach to X-Ray CT imaging and its applications in image analysis – part II: A new stochastic model-based image segmentation technique for X-Ray CT image,” *IEEE T. Med. Imag.*, 11(1):62–69, 1992.
7. R.C. Gonzalez and R.E. Woods. *Digital Image Processing*. Addison-Wesley, 1992.
8. Z. Akalın Acar, “Electro-Magnetic Source Imaging Using Realistic Head Models,” *PhD Thesis, Department of Electrical and Electronics Engineering*, METU, June 2005
9. Matthieu Ferrant, Olivier Cuisenaire1 and Benoit Macq,“Multi-object segmentation of brain structures in 3D MRI using a computerized atlas,” *Medical Imaging 1999: Image Processing* Proc. SPIE Vol. 3661, p. 986-995

10. Y. Zhang, M. Brady, S., "Segmentation of Brain MR Images Through a Hidden Markov Random Field Model and the Expectation-Maximization Algorithm," Smith, *IEEE Transactions on Medical Imaging*, vol. 20, no. 1, pp.45, January 2001.

11. C. Baillard and C. Barillot,IRISA, "Robust 3D segmentation of anatomical structures with level sets," *Lecture Notes In Computer Science; Proceedings of the Third International Conference on Medical Image Computing and Computer-Assisted Intervention,* Vol. 1935

12. C. Baillard , P. Hellier , C. Barillot, "Segmentation of brain 3D MR images using level sets and dense registration," *Medical Image Analysis 5* (2001)

13. 185–194 D Ilea, O Ghita, K Robinson, R Sadleir, M Lynch, D Brennan, PF Whelan.," Identification of Body Fat Tissues in MRI Data," *Optimization of Electrical and Electronic Equipment,* Pages: 227-232, Brasov, Romania, 2004

14. Xiangrong Zhou, Takeshi Hara, Hiroshi Fujita, Ryujiro Yokoyama, Takuji Kiryu, and Hiroaki Hoshi, "Automated segmentations of skin, soft-tissue, and skeleton from torso CT images," *Medical Imaging 2004: Image Processing. Proceedings of the SPIE,* Volume 5370, pp. 1634-1639 (2004).

15. S. Barros, N. Sa Couto, F. B. Sachse, C. D. Werner, and G. Seemann. "Segmentation and tissue-classification of the visible female dataset - thoracic muscles, bones and blood vessels," *Biomedizinische Technik*, volume 46-1, pages 510–511, Sep. 2001.

16. Martin, N. Bordes, T. Hugh, B. Pailthorpe "Semi-Automatic Feature Delineation In Medical Images," *Australasian Symposium on Information Visualisation, Christchurch, New Zealand, 2004. Conferences in Research and Practice in Information Technology*, Vol. 35, pp 127, N. Churcher and C. Churcher Eds (2004).

17. Michal Holtzman-Gazit, Ron Kimmel, Nathan Peled, Dorith Goldsher, “Segmentation of Thin Structures in Volumetric Medical Images,” *IEEE Transactions on Image Processing,* Vol:15, Issue:2 354- 363, 2006

18. Ray T Burton, Kristin K Isaacs, John S Fleming, Ted B Martonen, “ Computer Reconstruction of a Human Lung Boundary Model From Magnetic Resonance Images,” *Respiratory Care*, February 2004 VOL 49 NO 2

19. Joris Heuberger, Antoine Geissbuhler, Henning Müller, “Lung CT Segmentation for Image Retrieval,” *Medical Imaging and Telemedicine (MIT 2005)*, WuYi Mountain, China, August 2005.

20. Abena B. Redwood, Jon J.Camp, Richard Robb, “Semiautomatic segmentation of the heart from CT images based on intensity and morphological features,” *Medical Imaging 2005: Image Processing*, p.1713-1719, Apr 2005

21. Dimitris Metaxas, Ting Chen, Xiaolei Huang, and Leon Axel, “Cardiac Segmentation from MRI-Tagged and CT Images,” In *Proc. of the 8th WSEAS International Conf. on Computers, special session on Imaging and Image Processing of Dynamic Processes in biology and medicine,* also appearing in WSEAS Transactions, July, 2004

22. Tim McInerney and Demetri Terzopoulos, “A Dynamic Finite Element Surface Model for Segmentation and Tracking in Multidimensional Medical Images with Application to Cardiac 4D Image Analysis,” Journal *of Computerized Medical Imaging and Graphics*, 1994 Jan-Feb;19(1):69-83.

23. André Neubauer and Rainer Wegenkittl, “Analysis of Four-Dimensional Cardiac Data Sets Using Skeleton-Based Segmentation,” *Journal of WSCG,* Vol.11, No.1.2003

24. M. Lorenzo-Valdes , G. I. Sanchez-Ortiz , R. Mohiaddin, and D. Rueckert, “Atlas-based segmentation and tracking of 3D cardiac MR images using non-rigid registration,” *Proceedings of the 2004 IEEE*

International Symposium on Biomedical Imaging, 15-18 April 2004. IEEE 2004

25. *"Dicom Works,"* http://www.dicomworks.com, Last accessed date February 2006

26. J. B. Zimmerman, S. B. Cousins, K. M. Hartzell, M. E. Frisse, and M. G. Kahn, "A Psychophysical Comparison of Two Methods for Adaptive Histogram Equalization." *Journal of Digital Imaging*, vol. 2, pp. 82-91, 1989.

27. Dzung L Pham, Chenyany Xu, Jerry L. Prince. "A Survey o Current Methods in Medical Image Segmentation," *Annual Review of Biomedical Engineering,* January 19,1998.

28. Nikhil R. PAL nad Sankar N. PAL, *"*A review on image segmentation techiniques*," Pattern recognition,* Vol. 26, No:6, pp-1277-1284, 1993

29. Mehment SEZGİN, Bülent SANKUR, *"*Survey over image thresholding techniques and quantative performance evaluation*" ,journal of Electronic Imaging 12(1),* 146-165 (January 2004)

30. N. Otsu, "A threshold selection method from gray-level histogram," *IEEE Transactions on System Man Cybernetics*, Vol. SMC-9, No. 1, 1979, pp. 62-66.

31. Z. Akalın-Acar, N. G. Gençer, "Electro-Magnetic Source Imaging with Realistic Head Models Using the Genetic Algorithms," *Proc. of BIOMAG* 2004, Boston, USA.

32. I. P. Pappas, Puja Malik and Martin Styner, "A new method to assess the registration accuracy of CT-MR images of the head" , *Injury*, 2004 June;35 Suppl 1:S-A105-12. (2004)

33. Pıng-Sung Lıao, Tse-Sheng Chen* And Pau-Choo Chung, "A Fast Algorithm for Multilevel Thresholding" , *Journal of Informatıon Scıence And Engıneerıng 17, 713-727 (2001)*

34. J. H. Wang and Y. W. Yu, "Image segmentation based on region growing and edge detection," *Proc. IEEE International Conf. on Systems, Man, and Cyber*., pp. 798-803, Tokyo, Japan, 1999.

35. Xiaohan, Y., Yla-Jaaski, J., Huttunen, "Image segmentation combining region growing and edge detection," *International Conference on Pattern Recognition,* August-1992 , Vol. C. The Hague, Netherlands, pp. 481-484.

36. V. Grau, A. U. J. Mewes, M. Alcañiz, R. Kikinis, and S. K. Warfield, "Improved Watershed Transform for Medical Image Segmentation Using Prior Information," *IEEE Transactions on Medical Imaging*, VOL. 23, NO. 4, April 2004

37. Meyer F. And Beucher S., "Morphological segmentation," *Journal of Visual Communication and Image Representation,* Vol 1, No.1, pp. 21-46 Sept 1990, Academic Press.

38. J. L. Vincent, "Morphological grayscale reconstruction in image analysis: Applications and efficient algorithms," *IEEE Trans. Image Processing*, vol. 2, pp. 176–201, 1993.

39. J. Weickert, "Fast segmentation methods based on partial differential equations and the watershed transform," in *Proc. DAGM Symp.*, 1998, pp. 93–100.

40. J. Weickert, "Fast segmentation methods based on partial differential equations and the watershed transform," in *Proc. DAGM Symp.*, 1998, pp. 93–100.

41. Lue Vincent and Paul Soille, "Watersheds in Digital Spaces: An Efficient Algorithm Based on Immersion Simulations," *IEEE Transactions on Pattern Analysis And Machine Intelligence*, VOL. 13, NO.16, June 1991

42. Joe B.T.M., Roerdink and Arnold Meijster*, "The Watershed Transform: Definitions, Algorithms And Parallelization Strategies" Fundamenta Informaticae,* 41 (2001), 187-228, IOS Press

43. Chenyang Xu, Jerry L. Prince," Snakes, Shapes, and Gradient Vector Flow*," IEEE Transactions On Image Processing*, VOL. 7, NO. 3, March 1998

44. Roerdink, J. B. T. M., and Meijster, “A. Segmentation by watersheds: definition and parallel implementation,” *Advances in Computer Vision*, F. Solina, W. G. Kropatsch, R. Klette, and R. Bajcsy, Eds. Springer, Wien, New York, 1997, pp. 21{30.

45. S. Beucher, "The Watershed Transformation Applied to Image Segmentation,” *10th Pfefferkorn Conf. on Signal and Image Processing in Microscopy and Microanalysis, Sept. 16-19 1991*

46. Micheal KAAS, Andrew WITKIN, Demetri TERZOPOULOS, “Snakes: Active Contour Models,” *International journal of Computer Vision* (1988), 321-331

47. Chenyang Xu, Jerry L. Prince,” Snakes, Shapes, and Gradient Vector Flow*,” IEEE Transactions On Image Processing*, VOL. 7, NO. 3, March 1998

48. Chenyang Xu and Jerry L. Prince, “Gradient Vector Flow: A New External Force for Snakes,” *IEEE Proc. Conf. on Comp. Vis. Patt. Recog. (CVPR'97)*

49. C. Xu, D. L. Pham, and J. L. Prince, "Medical Image Segmentation Using Deformable Models," *SPIE Handbook on Medical Imaging,* Volume III: Medical Image Analysis, edited by J.M. Fitzpatrick and M. Sonka, May 2000.

50. Michael KASS, Andrew WITKIN, and Demetri TERZOPOULOS, “Snakes: Active Cont”our Models,” *International Journal of Computer Vision , 321-331 (1998)*

51. Chenyang Xu and Jerry L. Prince, “Gradient Vector Flow: A New External Force for Snakes,” *IEEE Proc. Conf. on Comp. Vis. Patt. Recog. (CVPR'97)*

52. V. Caselles, F. Catte, T. Coll, and F. Dibos*,* “A geometric model for active contours,*” Numerische Mathematik, vol. 66, pp. 1–31, 1993.*

53. R. Malladi, J. A. Sethian, and B. C. Vemuri*, “Shape modeling with front propagation: a level set approach,” IEEE Trans. Patt. Anal. Mach. Intell.,* vol. 17, no. 2, pp. 158–175, 1995.

54. G. Sapiro and A. Tannenbaum, “Affine invariant scale-space,” *Int’l J. Comp. Vis.,* vol. 11, no. 1, pp. 25–44, 1993.

55. B. B. Kimia, A. R. Tannenbaum, and S. W. Zucker, “Shapes, shocks, and deformations I: the components of two-dimensional shape and the reaction-diffusion space,” *Int’l J. Comp. Vis.,* vol. 15, pp. 189–224, 1995.

56. S. Osher and J. A. Sethian, “Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations,” *J. Computational Physics, vol. 79,pp. 12–49, 1988.*

57. J. A. Sethian, “Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Material Science. Cambridge,” *UK: Cambridge University Press,* 2nd ed., 1999.

58. C. Xu, D. L. Pham, and J. L. Prince, "Medical Image Segmentation Using Deformable Models," *SPIE Handbook on Medical Imaging,* Volume III: Medical Image Analysis, edited by J.M. Fitzpatrick and M. Sonka, May 2000.

59. Tony F. Chan, and Luminita A. Vese, “Active Contours Without Edges” *IEEE Transactions on Image Processing,* vol. 10, no. 2, February 2001

60. Thomas Brox, Dirk Farin, Peter H.N, “Multi-Stage Region Merging for Image Segmentation*,” 22nd Symposium on Information Theory in the Benelux,* 189-196, May 2001.

61. Kostas Haris, Serafim N. Efstratiadis,” Hybrid Image Segmentation Using Watersheds and Fast Region Merging*,” IEEE Transactions on Image Processing*, VOL. 7, NO. 12, December 1998

62. Kostas Haris, S. N. Efstratiadis, and N. Maglaveras. “Watershed-based image segmentation with fast region merging”. *IEEE International Conference on Image Processing,* volume 3, pages 338-342, 1998.

63. Zhukov, Z. Bao, I. Guskov, J. Wood and D. Breen, "Dynamic Deformable Models for 3D MRI Heart Segmentation", SPIE Medical Imaging 2002 Conference Proceedings, February 2002, pp. 1398-1405.

64. T. McInerney and D. Terzopoulos, “A dynamic finite element surface model for segmentation and tracking in multidimensional medical images with application to cardiac 4d image analysis,” *Comp. Med. Imag.Graph.*, 19(1):69--83, 1995.

65. I. Cohen, L. D. Cohen, and N. Ayache, “Using deformable surfaces to segment 3-D images and infer differential structures,” *CVGIP: Imag. Under.*, vol. 56, no. 2, pp. 242–263, 1992.

66. T.F. Chan and X.-C. Tai.,”Level set and total variation regularization for elliptic inverse problems with discontinuous coeffcients,” *Journal of Computational Physics*, vol. 193, pp. 40-66, 2003

67. Soille, P. ,Morphological İmage Analysis: Priciples and Applications, 2nd ed. Springer-Verlag. NY [2003].

68. Pietro PERONA and Jitendra MALIK, “Scale-Space and Edge Detection Using Anisotropic Diffusion” , *IEEE Transactıons On Pattern Analysıs And Machıne Intellıgence*, VOL. 12. NO. 7. JULY 1990

APPENDIX A

ANISOTROPIC FILTERS

Anisotropic Filters are iterative, filters that introduced by Perona and Malik [68]. In our study these filters are used for noise removing and edge preserving smoothing purposes.

Because convolving the original image with a Gaussian kernel makes it difficult to obtain accurately the locations of the “semantically meaningful” edges at coarse scales, anisotropic filters are preferred. Anisotropic filters sharpen edges while smoothing other non-edge regions, there for while smoothing the image, edge information is preserved.

Perona and Malik formulate the anisotropic diffusion filter as a diffusion process that encourages intra-region smoothing while inhibiting inter-region smoothing. Mathematically, the process is defined as follows:

$$\frac{\partial}{\partial t} I(x,y,t) = \nabla.(c(x,y,t)\nabla I(x,y,t)) = c(x,y,t)\nabla^2 I(x,y,t) + \nabla c(x,y,t).\nabla I(x,y,t) \quad \textbf{A.1}$$

I : Original image

t : time (step)

$c(x,y,t)$: Diffusion coefficient

If we assume that *c(x,y,t)* is a constant then,

$$\frac{\partial}{\partial t} I(x,y,t) = c(x,y,t)\nabla^2 I(x,y,t)$$

In addition, assume we knew the locations of the region boundaries appropriate for that time scale t. We would want to encourage smoothing *within* a region and inhibiting smoothing *across* the boundaries. This could be achieved by setting the conduction coefficient to be 1 in the interior of each region and 0 at the boundaries. The blurring would then take place separately in each region with no interaction between regions. The region boundaries would remain sharp.

However, we do not know the boundaries of the image at time t, but we can estimate it. Let *E(x,y,t)* be an estimate of image boundaries at time t, then the conduction coefficient *c(x,y,t)* can be chosen to be a function c = g ($\|E(x,y,t)\|$). According to the previously stated function for diffusion coefficient, g ( ), has to be a nonnegative monotonically decreasing function with g (0) = 1. We can estimate the image boundaries by using the gradient of the image.

$$E(x,y,t) = \nabla I(x,y,t)$$
$$c(x,y,t) = g(|\nabla I(x,y,t)|)$$

Two different diffusion functions are proposed by Perona and Malik [68], for g (.) giving perceptually similar results. (see Figure-A.1)

$$g(f) = \exp(-(\frac{\|f\|}{K})^2) \qquad \text{or} \qquad g(f) = \frac{1}{1+(\frac{\|f\|}{K})^2} \qquad \textbf{A.2}$$

According to the equation-A.2, we can write the diffusion coefficient as;

$$c(x,y,t) = \exp\left(-\left(\frac{|\nabla I(x,y,t)|}{K}\right)^2\right) \qquad \textbf{A.3}$$

$$c(x,y,t) = \frac{1}{1+\left(\frac{|\nabla I(x,y,t)|}{K}\right)^2} \qquad \textbf{A.4}$$

the first privileges high-contrast edges over low-contrast ones, the second privileges wide regions over smaller ones, and K is the diffusion constant.

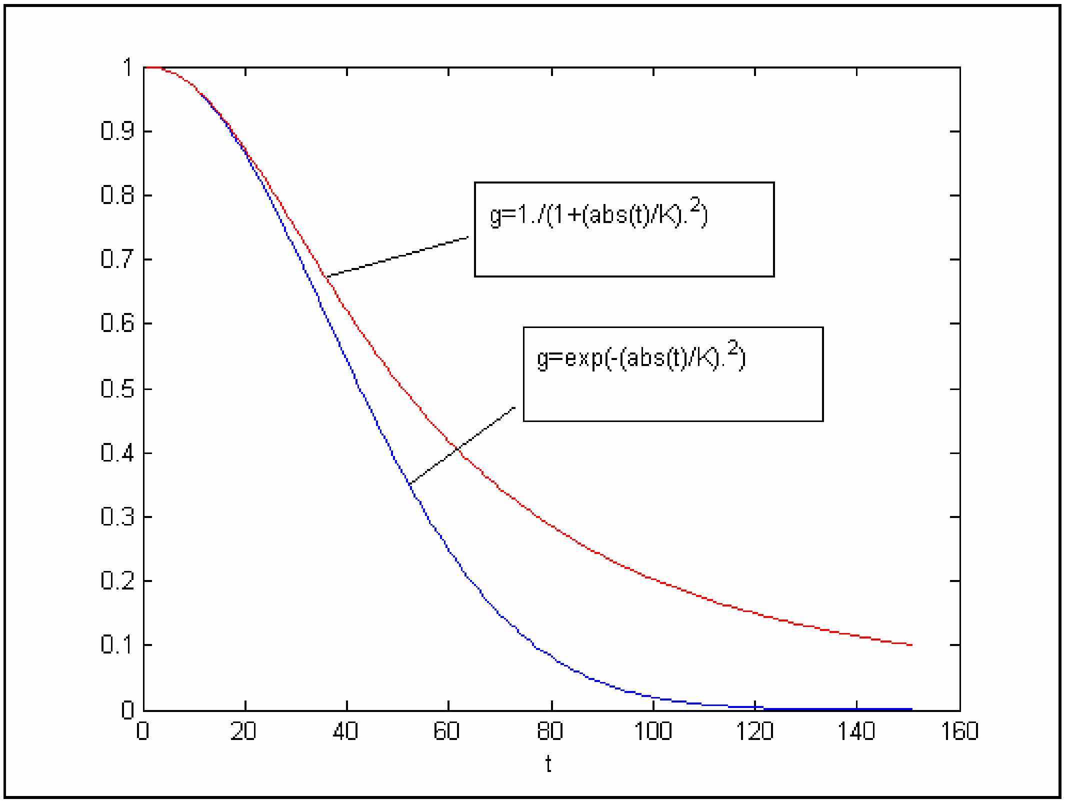


Figure-A.1 : Diffusion functions plotted as a function of a variable t

Numerical Implementation

The discrete implementation of the nonlinear anisotropic diffusion filter is straightforward. In the discrete domain, a gradient or derivative can be approximated as the difference in intensity between neighboring elements in the image, and the filter is iterative.

$$I_{i,j}^{t+1} = I_{i,j}^{t} + \lambda\left[c_N.\nabla_N I + c_S.\nabla_S I + c_E.\nabla_E I + c_W.\nabla_W I\right]_{i,j}^{t} \qquad \textbf{A.5}$$

where $0 \le \lambda \le 0.25$ for the numerical scheme to be stable, and N, S, E, W are the mnemonic subscripts for North, South, East, West, and the symbol ∇ (not to be confused with ∇, which we use for the gradient operator) indicates nearest-neighbor differences[68]:

$$\begin{aligned}
\nabla_N I_{i,j} &= I_{i-1,j} - I_{i,j}\\
\nabla_S I_{i,j} &= I_{i+1,j} - I_{i,j}\\
\nabla_E I_{i,j} &= I_{i,j+1} - I_{i,j}\\
\nabla_W I_{i,j} &= I_{i,j-1} - I_{i,j}
\end{aligned}$$

and the conduction coefficients are updated at every iteration,

$$\begin{aligned}
c_{N_{i,j}}^{t} &= g(\left|\nabla_N I_{i,j}^{t}\right|)\\
c_{W_{i,j}}^{t} &= g(\left|\nabla_W I_{i,j}^{t}\right|)\\
c_{E_{i,j}}^{t} &= g(\left|\nabla_E I_{i,j}^{t}\right|)\\
c_{S_{i,j}}^{t} &= g(\left|\nabla_S I_{i,j}^{t}\right|)
\end{aligned} \qquad \textbf{A.6}$$

The conduction coefficients are obtained from A.6, and the filtered image is obtained from A.5. Visualization of diffusion among pixels in an image *I (i,j,t)* is given in Figure-A.2.

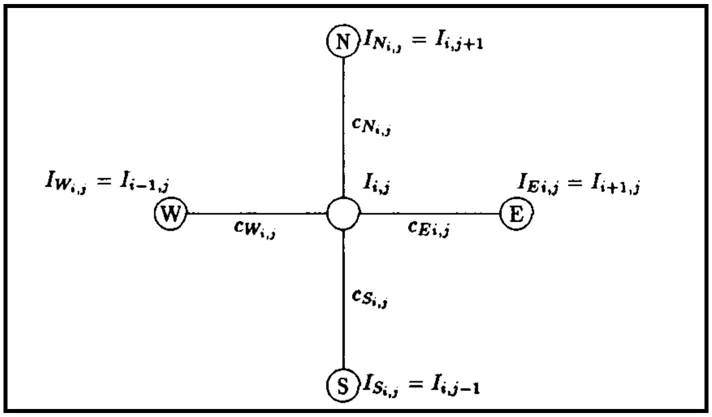

$I_{N_{i,j}} = I_{i,j+1}$
N
$c_{N_{i,j}}$
$I_{W_{i,j}} = I_{i-1,j}$
W
$c_{W_{i,j}}$
$I_{i,j}$
$c_{E_{i,j}}$
$I_{E_{i,j}} = I_{i+1,j}$
E
$c_{S_{i,j}}$
S
$I_{S_{i,j}} = I_{i,j-1}$

Figure- A.2: Visualization of diffusion among pixels in an image.

EXPERIMENTAL RESULTS

The test image is given in Figure-A.3, and the noisy image is given in Figure-A.4. Original image is exposed to Gaussian noise, and the SNR is 10 dB.

We applied anisotropic diffusion filter with the following parameters to remove the noise.

Diffusion Const (K): 30

Lambda=0.25;

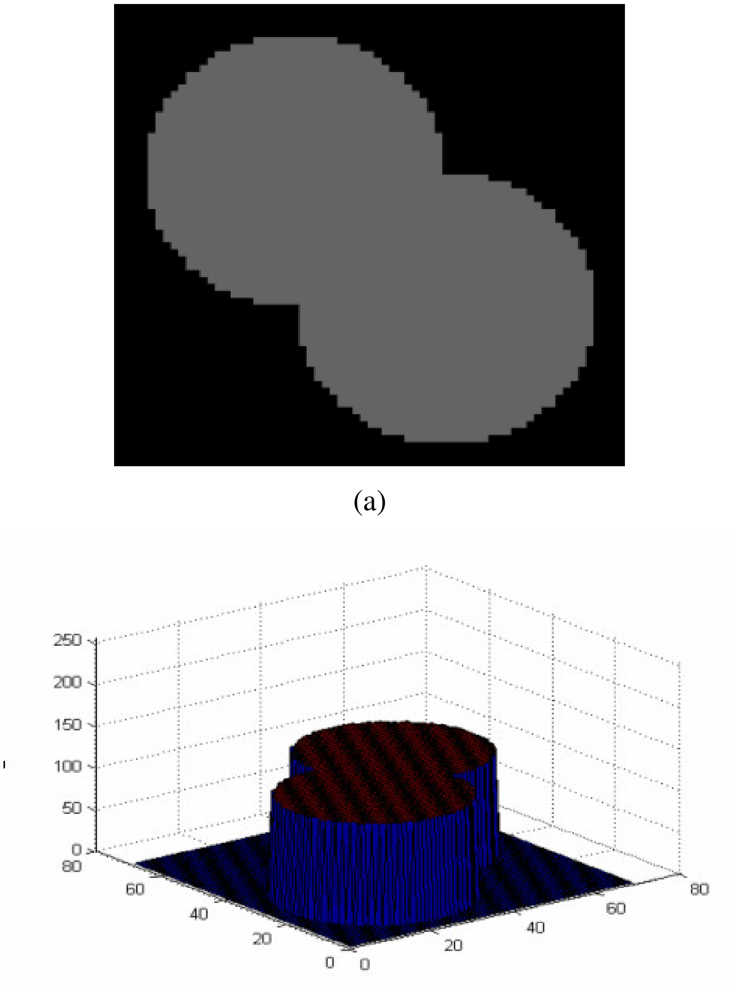
$c(x,y,t) = \exp\left(-\left(\frac{|\nabla I(x,y,t)|}{K}\right)^2\right)$, and the number of iterations is 30.

By observing Figure-A.5, result of anisotropic diffusion filtering we can state that noise is successfully removed, however we still have some peaks at the borders.

We also applied anisotropic diffusion filter with the other diffusion coefficient given in equation-A.4, with the same parameters given previously.

$$c(x, y, t) = \frac{1}{1 + \left(\frac{\left| \nabla I(x, y, t) \right|}{K} \right)^2}$$

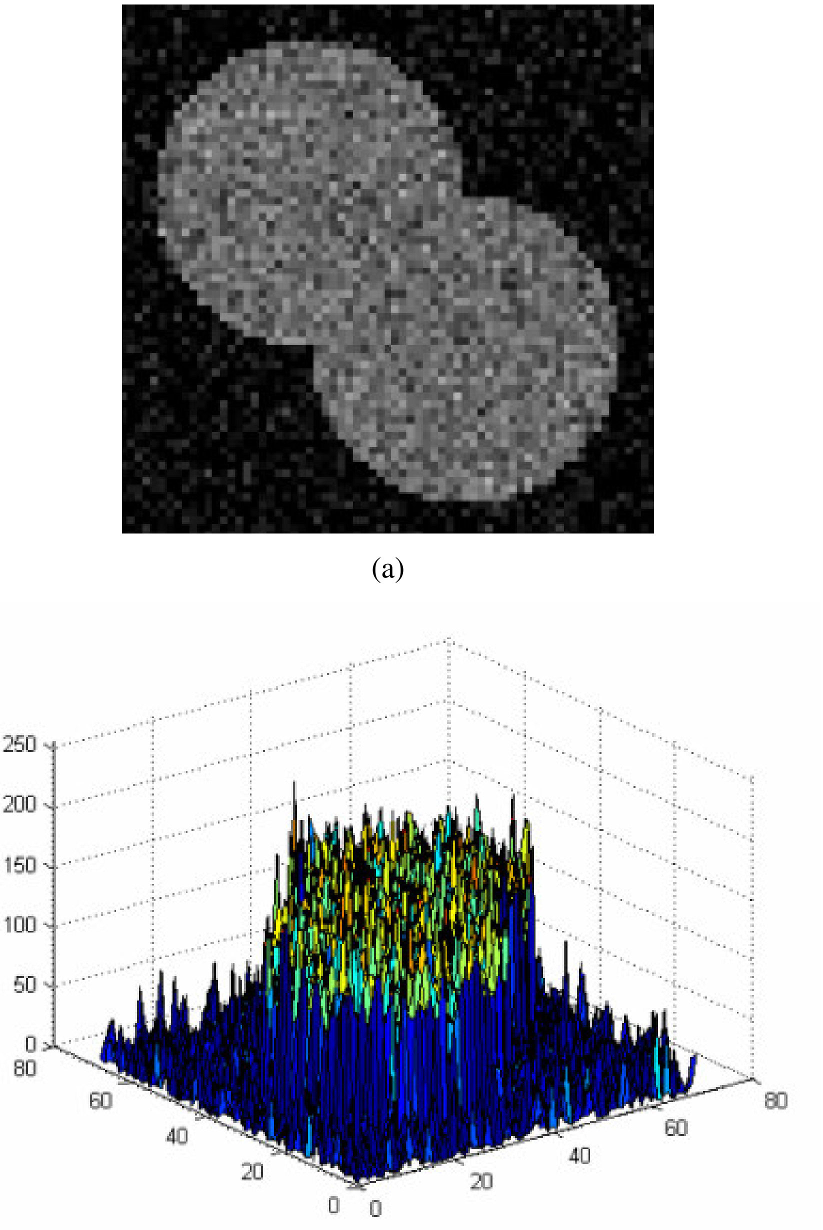
By observing the filtered image given in Figure-A.6, we can state that noise is removed successfully, without peaks at edges as in Figure-A.5, but in this case edges of the object are smoothed significantly.



(a)

(b)

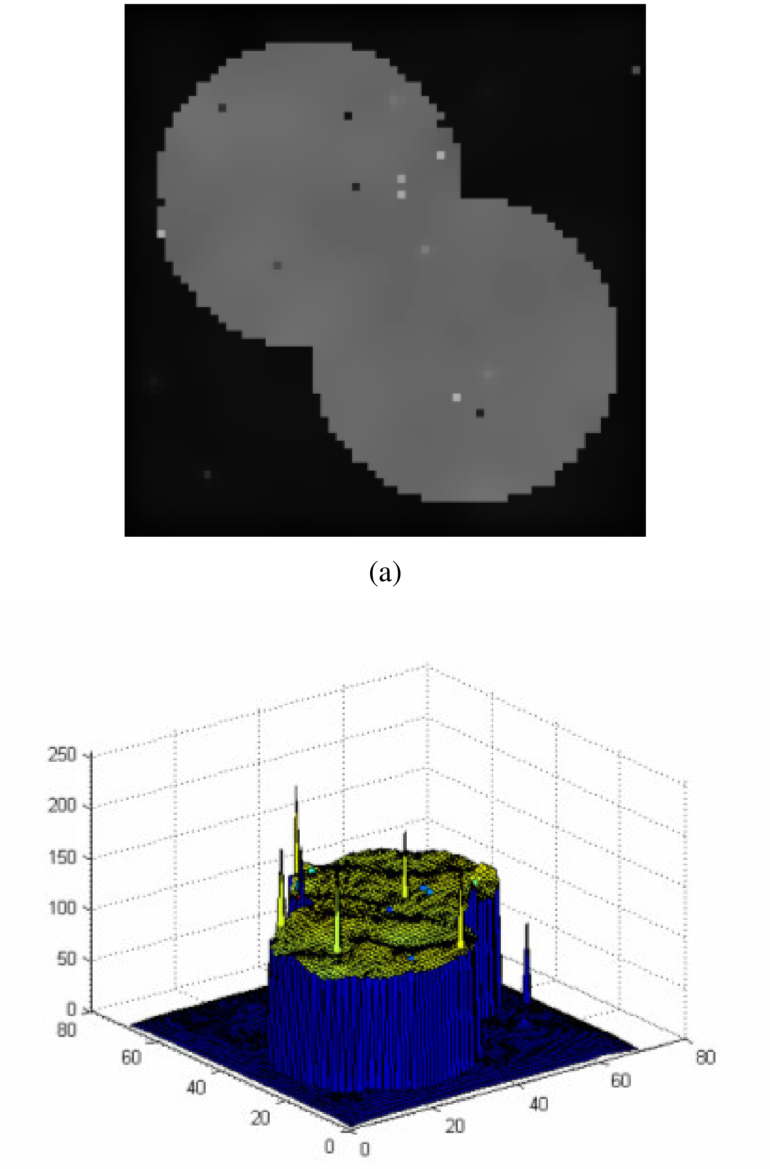
Figure-A.3: (a) Original image,(b) 3D visualization



(a)

(b)

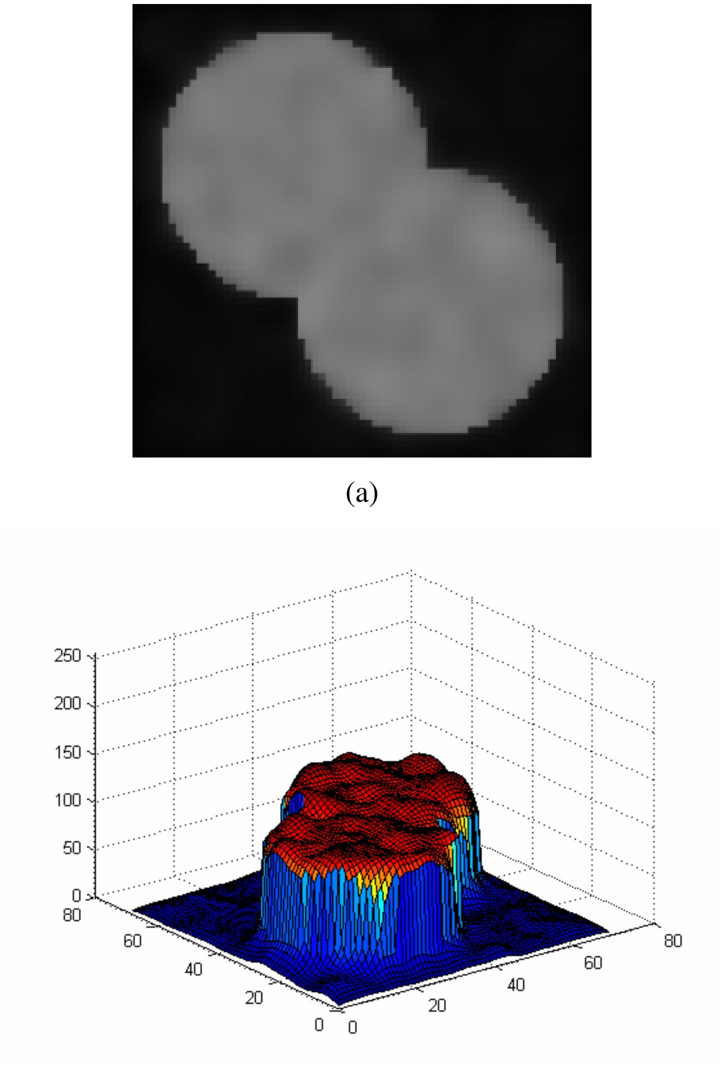
Figure-A.4: (a) Noisy image,(b) 3D visualization



(a)

(b)

Figure-A.5: Filtered image with diffusion coefficient given in equation A.3 (a) Filtered image,(b) 3D visualization



(a)

(b)

Figure-A.6: Filtered image with diffusion coefficient given in equation A.4 (a) Filtered image,(b) 3D visualization

APPENDIX B

IMPLEMENTATION of ALGORITHMS

B.1. Pseudo Code of the Region Growing Algorithm.

```
Initialization :
    • Label seed points according to their initial grouping
    • Put neighbors of seed points to $T$
    • Compute the Edge-Map
Region Growing
    while dynamic queue is not empty do
            • Get the pixel $N(x)$ from top of the dynamic queue
            • If it is not labeled with another label different from the
              region it intersects ($R_i$)
                o Apply dissimilarity Test
                o If ($N(x)$ passes the test)
                    ▪ Label $N(x)$ with the label of ($R_i$).
                    ▪ Update mean value of ($R_i$).
                    ▪ Add neighbors of $N(x)$ to the
                      dynamic queue
    end
```

B.2. Implementation of the Watershed Transformation

- First, sort all the pixels of the image in an increasing with respect to the gray level values from h_{min} to h_{max}.
- After the sorting step, flooding step of the catchment basins is started.
- Suppose that flooding step up to gray level h is done and all the catchment basins that have a *minimum* lower or equal to h is found, then we proceed to gray level $h+1$, find the pixels with gray level of $h+1$ and label them with a special value **MASK**.
  - Among those pixels that were labeled with **MASK**,
    - if their neighbors are labeled as *watershed* or labeled with another catchment basins label, they are put into a queue. The queue is in *first-in-first-out* data structure. Then utilizing the set of pixels in the queue, we extend the catchment basins by computing the geodesic influence zones.
    - The pixels which are labeled with MASK, but were not taken into the queue (because none of their neighbors are labeled as watershed or labeled with another catchment basins label) are assumed as a newly discovered catchment basin and labeled with a new label. This process goes on until $h=h_{max}$.

B.3. The Pseudo Code for Watershed Transformation and Region Merging Algorithm-1

Algorithm-1 Pseudo Code

```
    Input: watershed segmented image with M sub-regions
    Output: Segmented Image
        •  Compute the RAG
        •  $N_M^i$ : neighboring regions of $R_M^i$ // NoN: Number of Neighbors
İteration: for i=1: M // for all sub-region
            ○  while ( NoN !=j )  // check all the neighboring regions
                ▪  compute $\delta(R_M^i, N_M^{i,j})$
                    •  If( $\delta(R_M^i, N_M^{i,j})$ <T=true) then
                          Merge regions $R_M^i$ and $N_M^{i,j}$;
                          Update mean value of $R_M^i$ ;
                          Update RAG;
                          Compute new neighboring regions and NoN
                          j=1;
                    •  Else
                          j=j+1;
                    •  End If
            ○  End While;
 End For;
```

APPENDIX C

DISCRETIZATION of DEFORMABLE MODELS for NUMERICAL IMPLEMENTATION

C.1. Discretization of Snake Formulation

By approximating the derivatives given in equation 2.4.16 with finite differences, and converting to the vector notation, we obtained the following set of discrete equations:

As stated section-2.4.2, the contour is represented by a set of control points in which each control point has a position, given by (x, y) coordinates as below:

$$\overline{X}_i^n = \left(X_i^n, Y_i^n\right) = \left(X(ih, n\Delta t), Y(ih, n\Delta t)\right)$$

where *i* is the control point index of the contour, *n* is the time index, and *h* is the step size in space. Thereby, the partial derivative of X with respect to t, in equation 2.4.16 can be written as follows:

$$\gamma \frac{\partial \overline{X}}{\partial t} = \gamma \frac{\overline{X}_i^n - \overline{X}_i^{n-1}}{\Delta t}$$

Similarly the elastic energy term in equation 2.4.16 can be written as;

$$\frac{\partial}{\partial s}\left(\alpha \frac{\partial X}{\partial s}\right) = \frac{1}{h^2}\left[\alpha_{i+1}(\overline{X}_{i+1}^n - \overline{X}_i^n) - \alpha_i(\overline{X}_i^n - \overline{X}_{i-1}^n)\right]$$

and let us call the elastic energy term A_i^n.

$$A_i^n = \frac{1}{h^2}\left[\alpha_{i+1}(\overline{X}_{i+1}^n - \overline{X}_i^n) - \alpha_i(\overline{X}_i^n - \overline{X}_{i-1}^n)\right]$$

The bending energy term can be written as;

$$\frac{\partial^2}{\partial s^2}\left(\beta\frac{\partial^2 X}{\partial s^2}\right)=\frac{1}{h^4}\beta_{i-1}(\overline{X}^{n}_{i-2}-2\overline{X}^{n}_{i-1}+\overline{X}^{n}_{i})-2\beta_i(\overline{X}^{n}_{i-1}-2\overline{X}^{n}_{i}+\overline{X}^{n}_{i+1})+\beta_{i+1}(\overline{X}^{n}_{i}-2\overline{X}^{n}_{i+1}+\overline{X}^{n}_{i+2})$$

and let us call the elastic energy term B_i^n.

$$B_i^n=\frac{1}{h^4}\beta_{i-1}(\overline{X}^{n}_{i-2}-2\overline{X}^{n}_{i-1}+\overline{X}^{n}_{i})-2\beta_i(\overline{X}^{n}_{i-1}-2\overline{X}^{n}_{i}+\overline{X}^{n}_{i+1})+\beta_{i+1}(\overline{X}^{n}_{i}-2\overline{X}^{n}_{i+1}+\overline{X}^{n}_{i+2})$$

By setting the external force term as below, we obtained the discrete form of the equation-2.4.16 as in equation below .

$$\nabla P(X)=F_{ext}(\overline{X}^{n-1}_{i})$$

$$\gamma\,\frac{\overline{X}^{n}_{i}-\overline{X}^{n-1}_{i}}{\Delta t}=A_i^n-B_i^n-F_{ext}(\overline{X}^{n-1}_{i})$$

C.2. Discretization of a Level Set Function

The discretization of (2.4.36) is given as:

$$\frac{\phi_{i,j}^{n+1}-\phi_{i,j}^{n}}{\Delta t}=\delta_h(\phi_{i,j}^{n})\frac{\mu}{h^2}\Delta_-^x\left(\frac{\phi_{i,j}^{n+1}-\phi_{i-1,j}^{n+1}}{\sqrt{\left((\phi_{i+1,j}^{n}-\phi_{i,j}^{n})\Big/h^2\right)+\left((\phi_{i,j+1}^{n}-\phi_{i,j-1}^{n})^2\Big/4h^2\right)}}\right)$$

$$+\delta_h(\phi_{i,j}^{n})\frac{\mu}{h^2}\Delta_-^y\left(\frac{\phi_{i,j+1}^{n+1}-\phi_{i,j}^{n+1}}{\sqrt{\left((\phi_{i+1,j}^{n}-\phi_{i-1,j}^{n})^2\Big/4h^2\right)+\left((\phi_{i,j+1}^{n}-\phi_{i,j}^{n})\Big/h^2\right)}}\right)$$

$$-\delta_h(\phi_{i,j}^{n})v-\delta_h(\phi_{i,j}^{n})\lambda_1(I_{0,i,j}-c_1(\phi^n))^2+\delta_h(\phi_{i,j}^{n})\lambda_2(I_{0,i,j}-c_2(\phi^n))^2.$$

for more details about discretization of equation 2.4.37, the we refer the reader to [59] . The algorithm is below:

```
Initialize $\lambda_1, \lambda_2, \Delta t, h, \mu$ , $\phi^0$ by initial contour $\phi_0$, n=0,
For    total number of iterations
          Compute $c_1(\phi^n)$ and $c_2(\phi^n)$ from (2.4.35)
Compute $\phi^{n+1}$ from (2.4.36)
End
```