SEMI-AUTOMATIC/USER-GUIDED SEGMENTATION OF MITOCHONDRIA ON TRANSMISSION ELECTRON MICROSCOPY IMAGES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

MUSTAFA ÇÖÇELLİ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
THE DEPARTMENT OF HEALTH INFORMATICS

JULY 2015

SEMI-AUTOMATIC/USER-GUIDED SEGMENTATION OF MITOCHONDRIA ON TRANSMISSION ELECTRON MICROSCOPY IMAGES

Submitted by Mustafa ÇÖÇELLİ in partial fulfillment of the requirements for the degree of Master of Science in Health Informatics, Middle East Technical University by.

Prof. Dr. Nazife BAYKAL ________________________
Director, Informatics Institute

Assoc. Prof. Dr. Yeşim Aydın SON ________________________
Head of Department, Health Informatics

Prof. Dr. Erkan MUMCUOĞLU ________________________
Supervisor, Health Informatics, METU

Assist. Prof. Dr. Reza HASANPOUR ________________________
Co-Supervisor, Computer Engineering, Çankaya University

Examining Committee Members

Assoc. Prof. Dr. Rengül ATALAY ________________________
Health Informatics, METU

Prof. Dr. Erkan MUMCUOĞLU ________________________
Health Informatics, METU

Assist. Prof. Dr. Reza HASANPOUR ________________________
Computer Engineering, Çankaya University

Assoc. Prof. Dr. Alptekin TEMİZEL ________________________
Modeling and Simulation, METU

Dr. Nurcan TUNÇBAĞ ________________________
Health Informatics, METU

02/07/2015

ABSTRACT

SEMİ-AUTOMATIC/USER-GUIDED
SEGMENTATION OF MITOCHONDRIA ON TRANSMISSION
ELECTRON MICROSCOPY IMAGES

Çöçelli, Mustafa
Department of Health Informatics
Supervisor : Prof. Dr. Ünal Erkan Mumcuoğlu
Co-Supervisor : Assist. Prof. Reza Hassanpour

July 2015, 109 pages

The purpose of the study is to develop and implement some user-guided semi-automatic methods to segment mitochondria on transmission electron microscopy images. Earlier automatic active contour based detection and segmentation algorithm applied on slices of mitochondria dataset are subject to some errors. Therefore, semi-automatic methods are necessary to adjust structure of wrongly detected and segmented mitochondria. There are six semi-automatic methods introduced in this thesis, namely, splitting, merging, deletion, initiation of automatic segmentation algorithm at intended position, selection of low scored mitochondria and manual drawing. In summary, firstly, splitting is separation of wrongly detected mitochondrion into two pieces with the help of generic surface defined by the user. Merging is the combination of two mitochondria which results in formation of one valid mitochondrion. Deletion allows removing false detections in non-mitochondrial region. Initiation of automatic algorithm at user-specified position is done by softening validation phase parameters of automatic segmentation algorithm. Lastly, manual drawing is to form 3D shape of a mitochondrion from user-defined points. Prior to the implementation of these semi-automatic methods, mitochondrial images and the output of the automatic algorithm were visualized to ensure user evaluation on mitochondrial images.

Keywords: Mitochondria, Transmission Electron Microscopy, Tomography, Volume Rendering, VTK, Semi-Automatic Segmentation, 3D

ÖZ

TRANSMİSYON ELEKTRON MİKROSKOBU GÖRÜNTÜLERİNDE
MITOKONDRİLERİN YARI
OTOMATIK/KULLANICI GÜDÜMLÜ BÖLÜTLENMESİ

Çöçelli, Mustafa
Sağlık Bilişimi Bölümü
Tez Yöneticisi : Prof. Ünal Erkan Mumcuoğlu
Ortak Tez Yöneticisi : Yrd. Doç. Dr. Reza Hassanpour

Temmuz 2015, 109 pages

Çalışmanın amacı, transmisyon elektron mikroskobu görüntülerinde mitokondrileri bölütlemek için kullanıcı güdümlü yarı otomatik yöntemler geliştirmek ve uygulamaktır. Daha önce yapılan, algılanmaya ve bölütlenmeye dayalı otomatik aktif kontur algoritması mitokondri veri kümeleri üzerinde hatalı çalışabiliyor. Bundan dolayı, yarı otomatik yöntemler yanlış algılanmış ve bölütlenmiş mitokondrileri düzeltmekte kullanılacaktır. Bölme, birleştirme, silme, otomatik bölütleme algoritmasının kullanıcının istediği pozisyonda başlatılması, düşük puan almış mitokondrilerin seçimi ve elle çizim teknikleri, kullanıcıya sunulan altı tane yarı otomatik tekniği oluşturur. Özetlemek gerekirse, .bölme işlemi, yanlış bölütlenmiş bir mitokondrinin, kullanıcı tarafından tanımlanan genel bir yüzey yardımı ile ikiye bölünmesidir. Birleştirme işlemi, iki mitokondri sonucunun birleştirilerek, bir tane geçerli mitokondri elde edilmesidir. Silme tekniği, mitokondri olmayan alanlardaki yanlış tespitlerin silinmesini sağlar. Bir diğer teknik, otomatik bölütleme algoritmasının, kullanıcının istediği bir pozisyonda, geçerlilik kriterlerinin yumuşatılatak başlatılmasıdır. Son olarak, elle çizim tekniği, kullanıcı tarafından tanımlanan noktalar ile bir mitokondrinin üç boyutlu görüntüsünün oluşturulmasıdır. Bu yarı otomatik teknikler gerçekleştirlmeden önce, kullanıcının değerlendirme yapabilmesi amacı ile, mitokondri resimleri ve otomatik algoritmanın çıktıları görselleştirilmiştir.

Anahtar Kelimeler: Mitokondri, Transmisyon Elektron Mikroskobu, Tomografi, Hacim Görüntüleme, Yarı-Otomatik Bölütleme.

To My Family

ACKNOWLEDGMENTS

I wish to thank to my advisor, Prof. Dr. Ünal Erkan MUMCUOĞLU and my co-advisor, Assist. Prof. Dr. Reza HASANPOUR for their endless support and encouragement.

I wish to thank especially to, my dear friend, Özcan Aslan for allocating his precious time to complete the editing of my thesis and the proofreading of my thesis. I would like to thank him for his effort to correct the wording of the thesis, also. His detailed work is adorable considering tight schedule.

I wish to thank especially to Serdar Taşel for sharing his work bountifully and for his endless support.

I also wish to thank Özgü Şenol, Ufuk Poyraz and Ethem Arkın for sharing their experiences while preparing thesis.

Finally, I would like to express my endless thanks to my family for their endless support. I wish to thank to my mom, Fatma Çöçelli, for her patience.

TABLE OF CONTENS

LIST OF TABLES

LIST OF FIGURES

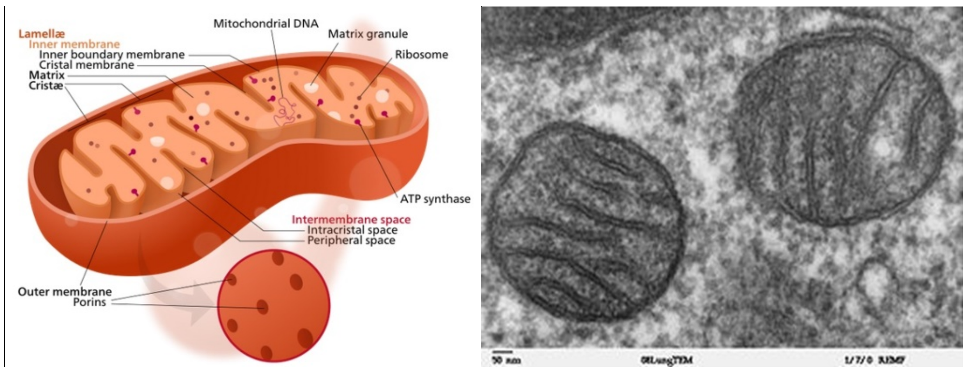
LIST OF ABBREVATIONS

2D	Two Dimensional
3D	Three Dimensional
CCDB	Cell Centered Database
CT	Computed Tomography
FN	False Negatives
FP	False Positives
MinIP	Minimum Intensity Projection
MIP	Maximum Intensity Projection
MRI	Magnetic Resonance Imaging
MPU	Multi-level Partition of Unity
NCMIR	National Center for Microscopy and Image Research
ROI	Region of Interest
SSD	Shaded Surface Display
TEM	Transmission Electron Microscope
TN	True Negatives
TP	True Positives
VTK	Visualization Toolkit

CHAPTER 1

INTRODUCTION

Mitochondrion is one of the important organelles inside a cell, which is responsible for the generation of metabolic energy necessary for the cell's survival. Shape of the mitochondrion is evolved to increase the area of cristae structures to maximize productivity. Mitochondria are bounded by a double-membrane system, consisting of inner and outer membrane[1]. The representative structure of mitochondria and a mitochondria example from electron microscopy image are shown in Figure 1.1 below.



(a) (b)

Figure 1.1 Mitochondrion Structure[2]

(a) Representation of Mitochondrion Structures. (b)Mitochondria Structures from Electron Microscopy Image.

The purpose of the study is to develop and implement some user-guided semi-automatic methods to segment mitochondria on transmission electron microscopy images because of their critical roles in different fields like cell death, aging and energy metabolism of the cell [3]. Moreover, position and orientation of mitochondria in cell attracts the attention of researchers doing various studies [4-6]. Therefore, extraction of mitochondria's location and shape from medical images in large scale rapidly gains more importance. Segmentation of mitochondria and visualization of this information allows researchers to interpret the information through their own perspective practically and help them draw conclusions more rapidly. There are various studies conducted to segment mitochondria automatically from electron microscopy images. However, we are focusing only on the studies that will be mentioned in the motivation section of this chapter.

Earlier automatic active contour based detection and segmentation algorithm applied on slices of mitochondria dataset are subject to some errors. While it succeeds to detect mitochondria with a remarkable performance, there are some wrong outputs of the algorithm due to the nature and quality of the images. It needs human perception and guidance to detect the rest of the mitochondria correctly. Therefore, semi-automatic methods will be applied to adjust structure of wrongly detected and segmented mitochondria. Some of these methods involve splitting and merging techniques. Firstly, these techniques will be applied on 2D slices of mitochondria. The expertise and know-how cultivated from this phase will be used to correct wrongly detected mitochondria in 3D. In other words, these techniques will be expanded to 3D model. While expanding to 3D model, new tools like deletion, validation and manual initiation of a snake will also assist the mentioned techniques.

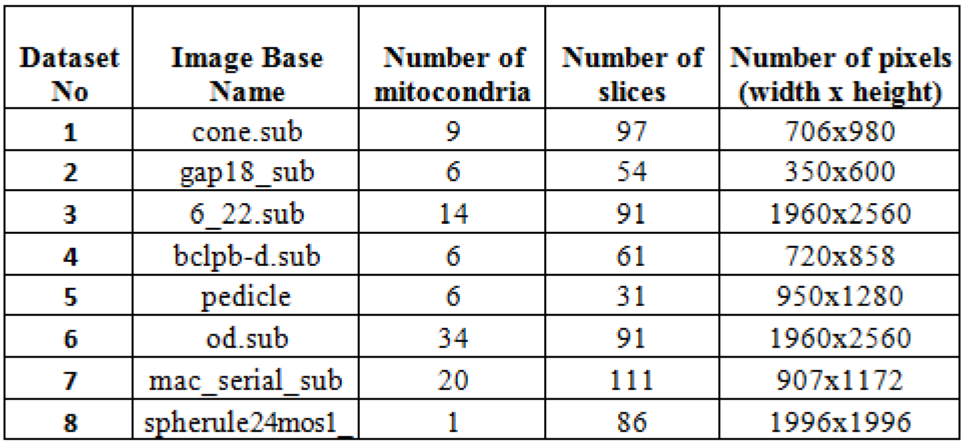
There is a necessity of interaction and visualization of outputs while doing semi-automatic segmentation. Therefore, visualization of mitochondria in 3D will be needed in an effective manner. This part is also the noteworthy part of this thesis since it serves all visualization and the interaction of the information required for the analysis and evaluation.

1.1. Motivation

The main motivation for this study is to make a contribution to semi-automatic section and visualization section of the project that is Mitochondria detection and segmentation on Electron Microscopy Project directed by *Prof. Dr. Erkan Mumcuoğlu* in the *Health Informatics department of Middle East Technical University.* This project is carried out in collaboration with the National Center for Microscopy and Image Research (NCMIR) at the University College of California San Diego.

There are mitochondrial images that are provided by NCMIR from their Cell-Centered Database (CCDB; http://ccdb.ucsd.edu) Project which is sustained by Martone et al.[7-9] waiting for segmentation of mitochondria study. CCDB is web accessible database including high resolution 2D, 3D and 4D data from light and electron microscopy. CCDB is a project that allow users to explore and re-use imaging data for different types of researches [10]. Datasets of mitochondrial images are supplied from CCDB for this study. Basically, each data set has hundreds of volume's slices and each of slices consists of approximately ten mitochondria in them. Image base name in CCDB, the number of mitochondria and the number slices of the provided dataset are given in Table 1.1 below.

Table 1.1 Mitochondria Datasets from CCDB



Dataset No	Image Base Name	Number of mitocondria	Number of slices	Number of pixels (width x height)
1	cone.sub	9	97	706x980
2	gap18_sub	6	54	350x600
3	6_22.sub	14	91	1960x2560
4	bclpb-d.sub	6	61	720x858
5	pedicle	6	31	950x1280
6	od.sub	34	91	1960x2560
7	mac_serial_sub	20	111	907x1172
8	spherule24mos1_	1	86	1996x1996

Datasets given in Table 1.1 includes various types of mitochondrial images having different shape and characteristics as indicated in Figure 1.2.

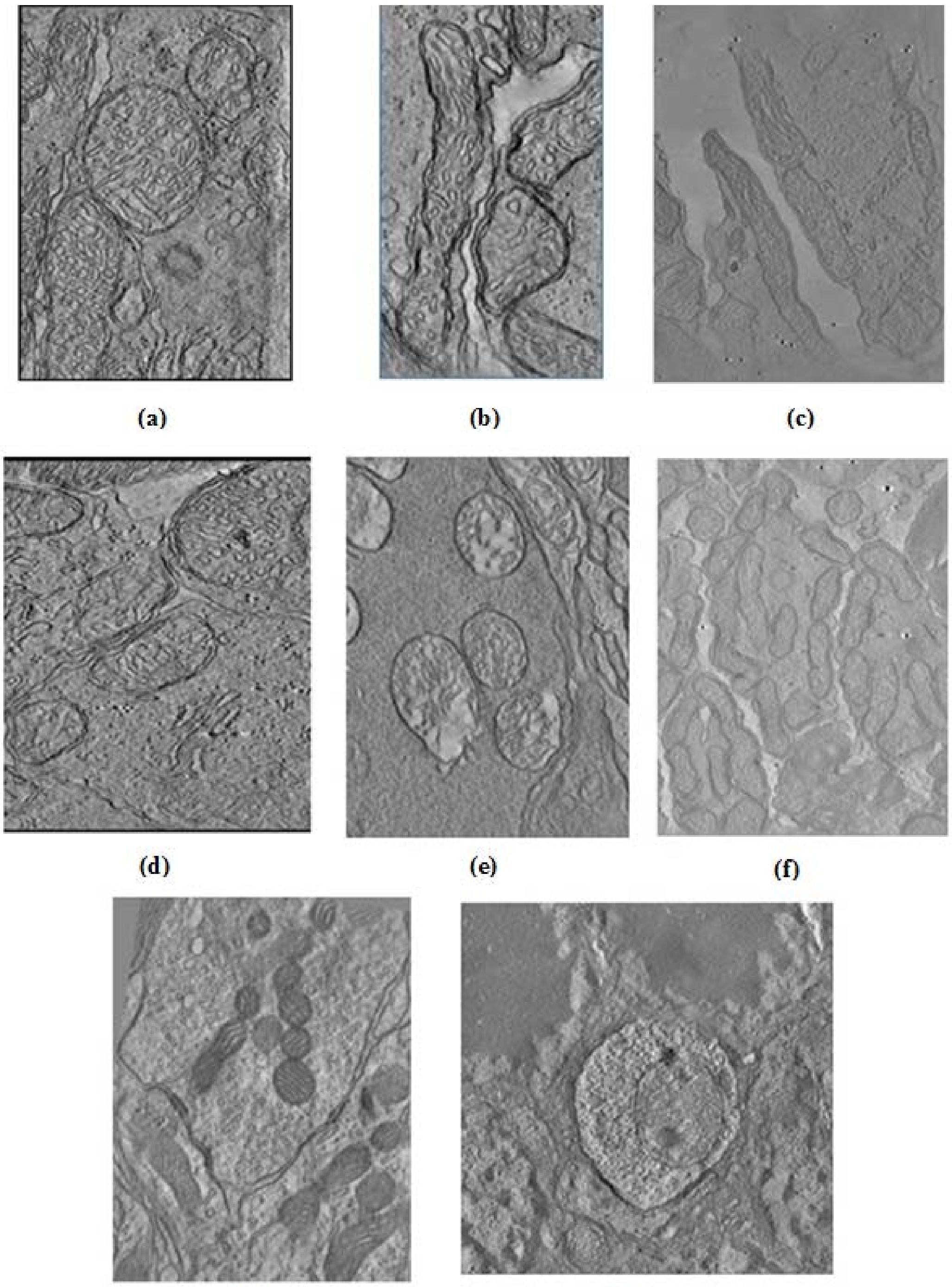


Figure 1.2 Mitochondrial Image Samples from CCDB
Image base name in CCDB : (a) cone.sub (b) gap18_sub (c) 6_22.sub (d) bclpb-d.sub (e) pedicle (f) od.sub (g) mac_serial_sub (h) spherule24mos1_

1.2. Organization of the Thesis

This thesis has 8 Chapters as following:

Chapter 1: Gives the introduction and motivation for the thesis.

Chapter 2: Gives previous studies specifically done on mitochondrial images or applicable to those images in the field of automatic segmentation, semi-automatic segmentation

Chapter 3: Gives the problem statement of the thesis that reveals the necessity of this study.

Chapter 4:.Gives the prior work that is the origin of this thesis.

Chapter 5: Presents detailed information about implementation methods developed for visualization phase of this thesis.

Chapter 6: Presents detailed information about semi-automatic detection and segmentation methods developed for this thesis.

Chapter 7: Gives the results of the methods and implementation.

Chapter 8: Concludes the thesis by discussing the results and proposing future works that should be done.

CHAPTER 2

BACKGROUND

Positions of cellular organelles and morphology information of them are necessary for cancer simulation studies. In order to obtain such information, the segmentation of the organelles from electron microscopy images is crucial. Mitochondrion is one of the most crucial organelles to observe cancer formation thanks to its characteristic shape convenient to analyze for researchers[11]. Moreover, analysis of mitochondrion morphology is important for the studies done in areas of neurodegenerative diseases including ALS, Parkinson's, Alzheimer's and diabetes which are common diseases seen all over the world[12]. Other than these, some studies search potential relation of mitochondrial morphology with various aspects of cardiovascular biology. All of these studies show that it is important to scan and view mitochondria structure with the use of today's technology. Segmentation of this organelle to analyze its morphology emerges to meet scientific data need of such research[13].

There are various methods like magnetic resonance imaging (MRI), computed tomography (CT), digital mammography to investigate the anatomy and physiology of a subject in details. Those types of technologies provide researchers many images with valuable information to evaluate in different areas of medical imagining. In addition to these techniques, transmission electron microscopy (TEM) is the one responsible for defining our sight of organelle structure, since it serves the high resolution cellular medical images for researchers[14].

Mitochondria were one of the first intracellular organelles that has been the subject of widespread research conducted by electron microscopy while the advances of methods of sample preparation of cells and tissues for TEM raised awareness of how mitochondria were important in energy metabolism [14]. Since there is a lot of data coming from studies done by TEM, valuable information needs to be extracted from those high resolution cellular images. Realization of this task is very time-consuming procedure when operators overhaul datasets manually. At this point, image segmentation methods take an important part in many medical imaging applications by automating the operation of segmentation, accelerating and making the delineation of anatomical structures easier to handle the increasing amount of medical images on a large scale [15]. Techniques for segmentation on medical images are varying by type, quality and purpose of the input datasets. There are various studies done automatically and semi-automatically to extract the valuable information from mitochondrial medical images and other types of images whose segmentation methods could be applied to mitochondrial images as well.

2.1. Automatic Segmentation

There are many studies to segment mitochondria and its substructures from different types of imagining sources automatically due to the importance of this organelle in different research areas like its role in cell death, aging and energy metabolism of cell. Sampe et al [16] used the method based on support vector machine to segment mitochondria correctly even mitochondria with different intensities and lost in noisy environment. The study of Lucchi et al is based on supervoxel-based segmentation using graph cuts to identify the location and morphology of mitochondria in 3D space[17]. Seyedhosseini [18] introduced a fully automated technique that makes use of both shape data and statistics information in regions to extract shape of chaotically formed mitochondria from electron microscopy (EM) images. Algebraic curves are utilized structures on images in order to extract shape and texture features from these images. Extracted features are used to predict the locations of mitochondria. Then, researchers of this study benefited from the algebraic curves and regional information for the segmentation of mitochondria located at those places. Jorstad and Fua [19] introduced an active surface-based method for detecting the boundary surfaces of mitochondria by benefiting from features of thick dark membranes and focusing on shape of the inner membrane. Giuly et al [20] presented a method, which includes of three major steps that are patch classification ,contour-pair classification and seed a level set operation. Mumcuoğlu et al [21] presented a method to segment fully or partially observed mitochondria by benefiting from elliptical shape and double membrane boundary of mitochondria structure. After active contours filters the roughly detected shapes, reliable seed points along the contour are determined automatically and incorporation between a live-wire graph search algorithm and these seed points succeeds to reach final results. Tasel et al [22] introduced the active contour detection algorithm which mainly takes the output of curve fitting process of the algorithm as input. This thesis is based on the continuation of Tasel’s work. It aims to complete the segmentation work with a collaboration of previous works.

2.2. Semi-Automatic Segmentation

Automatic segmentation method is not successful for every case in medical images due to various reasons. The reason of this phenomenon is variations of size and shapes of the desired structures or poor quality of the scanned images. This situation causes necessity of user-guided segmentation methods to handle unsuccessful cases of automatic algorithms. Olabarriaga and

Smeulders [23] defines the flow of interactive segmentation as indicated in Figure 2.1.

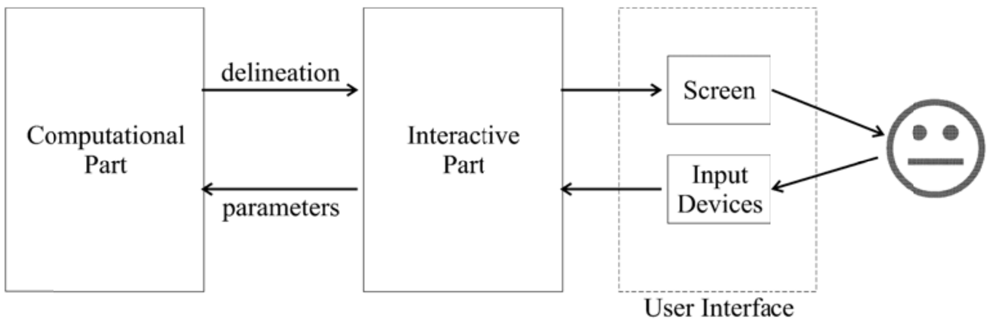


Figure 2.1 Main Components of Interactive Segmentation Methods [23]

The interactive part of Figure 2.1 consists of the visualization phase of this thesis. The user can follow and analyze from a rendered picture on the screen and interact with input devices with constructed instances. Interactive part of the component is connected with the computational part that consists of variation of automatic segmentation method implemented previously. The main philosophy of the interactive segmentation method is to help users with effective methods and reach results prompt as possible.

There have been various studies about semi-automatic segmentation methods in the field of segmentation to reduce user interaction time and increase efficiency. Fundamental techniques of image processing like image enhancement, region growing, amplitude segmentation are main tools in all operations of segmentation procedures according to Heinonen et al [24]. The idea is benefiting from convenient combination of these techniques to make medical image segmentation in a more practical way. Kang et al [25] created a kind of 3D editing tool, which could be utilized for correction or improvement of outputs of automatic segmentation methods. Mainly, their editing tools are composed of tools of hole-filling, point-bridging, and surface-dragging. They made demonstration of how their 3D approaches were successful compared to processing of 3D datasets slice-by-slice that is commonly used in medical image processing world. Yushkevich et al [26] created an open source application named ITK-SNAP, which is focused on active contour segmentation and level-set segmentation in its implementation. The application is benefit from seed points taken from user input to initialize segmentation algorithm. The user can reinitiate the segmentation with more accurate parameters by utilizing previous results and analysis. Segmentation results seem to be encouraging compared to manual segmentation. Barrett and Mortensen [27] introduced a tool of live-wire segmentation for effective, correct, and consistent boundary extraction that needs a considerable level of user interaction. Armstrong et al [28] presented Live

Surface that gives users the opportunity of segmentation and rendering of complex surfaces obtained by 3D image volumes. By using Live Surface approach, 3D image volumes are segmented with analogous usage of the Live Wire algorithm applied on 2D images. Jurrus et al [29] feed autonomous methods like machine learning and image processing techniques with user inputs to segment neuron membranes. If the ideas of the mentioned studies are synthesized, autonomous and manual techniques are interacted with each other and work together under the name of semi-automatic methods. One of them uses the output of the other as input at the first step, while it is the other way around next step. Therefore, it is vital to visualize the results to make human interpretation effective, and absolutely necessary to provide user friendly interfaces to ease the interaction of the user with the results.

2.3. Visualization

Visualization of the results is a crucial step for the analysis of the outputs of image processing techniques. While 2D visualization still has significant contributions to researches, the crucial point is the visual conversion of the outputs of slices of related 2D images into 3D environment. Analysis of 2D images slice by slice requires patience for researchers and it is difficult to evaluate all information obtained from every slice together. There have been many studies to resolve above these issues to ease analysis of researchers on all dataset ever since the birth of the volume rendering technique as a result of studies belonging to the Mayo Clinic in the 1970s [30]. Bruckner [31] studied on Maximum Intensity Projection (MIP) that is a common volume rendering technique for describing vascular structures visually. Bruckner used capabilities of The Visualization Toolkit (VTK) that is an open source and free software project for visualization and 3D computer graphics to implement MIP and its various extensions. Calhoun et al [30] studied the details of the implementation of MIP and Shaded Surface Display (SSD) that are widely used for volume rendering. Similarly, Pavone [32] presented steps of commonly used algorithms that are MIP and SSD to construct 3D volume from 2D slices of magnetic resonance images. Kremer et al [33] introduced a software system which is called IMOD, as a package for analysis and visualization of 3D biological images. IMOD has capabilities to visualize image data by various methods, which allow users to obtain valuable information from the dataset. Tomography, serial section and optical section reconstructions are available for researchers to study and model 3D biological images. Sandberg et al [34] presented a Fast Fourier Summation algorithm to reconstruct tomographic 3D biological images of transmission electron microscopy. 3D volume is obtained via slices of 2D images like other studies have done. Marker et al [35] introduced a volumetric method to reconstruction of a well-shaped surface from a few parallel binary contours. They focused on monotonicity-constraining cubic splines for forming clean-cut, continuous shapes between binary contours. They presented how their study was effective by running the algorithm on a various different contour datasets. Similarly, Braude et al [36] introduced a method to construct a continuous, closed surface from 2D contours obtained from scanning in 3D. The method evaluated points belonging to contours as they are located in 3D environment and applied Multi-level Partition of Unity (MPU) implicit models to construct a volume that nearly fits to those points located in 3D. They claimed that MPU implicit models brought an excellent viewpoint to the problem of contour-based surface reconstruction for noisy data sets through the implementation of adaptive implicit functions which close the points locally within a manageable error bound. In the study of Yan et al [37], gastrointestinal tract was reconstructed and visualized digitally in 3D with the usage of VTK. Reconstructed model had capabilities of zooming, panning, and rotation, which provided examination on interior fields of the stomach, the small intestine, and the large intestine.

CHAPTER 3

PROBLEM STATEMENT

Manual analysis of medical images is not efficient while dealing with large datasets and time is limited to extract needed features and information from those images. Moreover, it is prone to errors and the results are not standard since there is human factor during the analysis and segmentation of the images. Therefore, there is obvious need to benefit from automatic segmentation methods in order to be able to analyze medical images while obtaining the required information in reduced time, increased productivity and with standardized information. Automatic segmentation methods are vital for the analysis of medical images. However, they are also subject to some errors due to the variable quality of the medical images and the difficulty in determining standard parameters that are inputs for mostly automatic segmentation algorithm for every sample needed to be detected. In this case, human aid is needed to guide the artificial algorithm to detect and segment related information correctly. Human aided segmentation methods are named as semi-automatic segmentation methods since they are the combination of automatic approaches and manual guidance.

Physical effort, time and financial means are valuable resources that need to be spent carefully while segmenting and detecting mitochondria from transmission electron images. Therefore, manual operation on its own is not a choice to do this task. Inevitably, automatic methods have been implemented for the segmentation and detection of mitochondria from NCMIR database whose details are given in Table 1.1. The algorithm and implementation of Tasel et al [22] serves the automatic part of the exact solution of mitochondria segmentation from TEM images. This method segment majority of mitochondria from those images, but there are still some undetected or wrongly detected mitochondria that are waiting for a solution. Tasel et al [22] argued that automatic solution is suffering from inability of hundred percent coverage for detecting all mitochondria in the image database. It needs a continuation of its work that brings semi-automatic method to cover the rest of unsuccessful trails for the extraction of required mitochondrial structures as much as possible. At this point, semi-automatic and user guided methods which are the main topic of this thesis gain importance to handle rest of the part that needs to be solved. Realization of this process is tightly bound to visualization of sources and extracted information of auto segmentation part that forms the other critical part of this thesis.

This thesis is planned as the complementary work of the automatic phase. The aim of the study is to succeed improvement on results of the automatic segmentation algorithm after executing semi-automatic operations on these results with visualization support. Main flow of the entire process for detecting and segmentation mitochondria from TEM images is indicated in Figure 3.1.

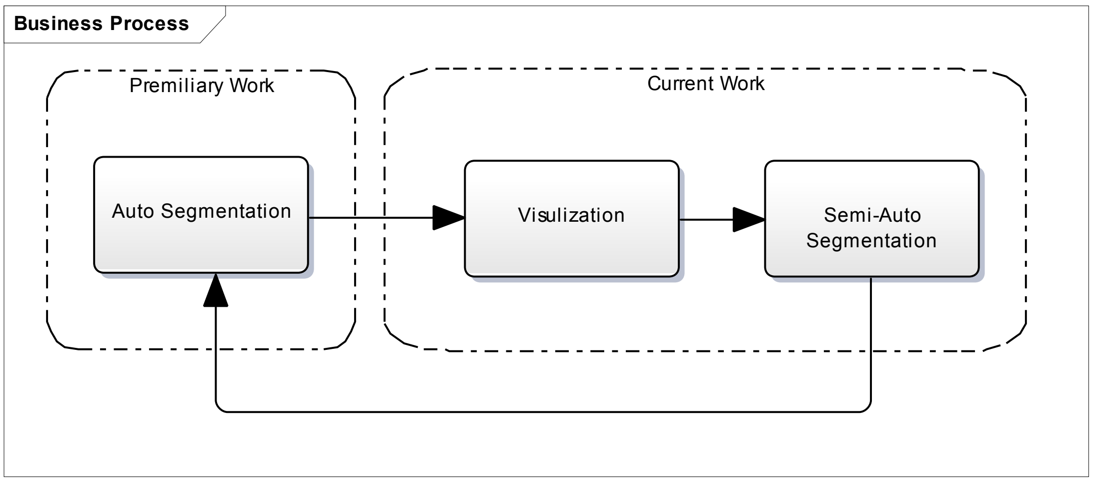


Figure 3.1 Business Process Model for the Semi-Automatic Approach

As shown in Figure 3.1, automatic segmentation algorithm serves the necessary information for the visualization part. Finally, the visualization phase provides all the requirements for the semi-automatic work.

CHAPTER 4

PREVIOUS WORK

Automated detection and segmentation of mitochondria is an indispensable requirement to ease and accelerate research done for diseases originated from mitochondria and aging process. Automation of this process is a demanding task because of the complex nature of mitochondria structures. To meet demands arising from this task, Taşel et al proposed a solution in 2D space [22] and preparing solution for 3D space [38], which is the current work in progress. The brief description of those algorithms can be described as follows.

4.1. 2D Automatic Segmentation Algorithm

Mumcuoğlu et al [21] proposed a solution to the addressed problem, which is based on approximate elliptical shape mitochondria and boundary with double membrane. This study is followed by the next one, which is the main motivation of this thesis. Taşel et al [22] introduced a study that was based on curve fitting approach around the boundary of the mitochondria. Steps of the algorithm can be summarized as preprocessing, curve fitting, active contour and shape validation as shown in Figure 4.1.

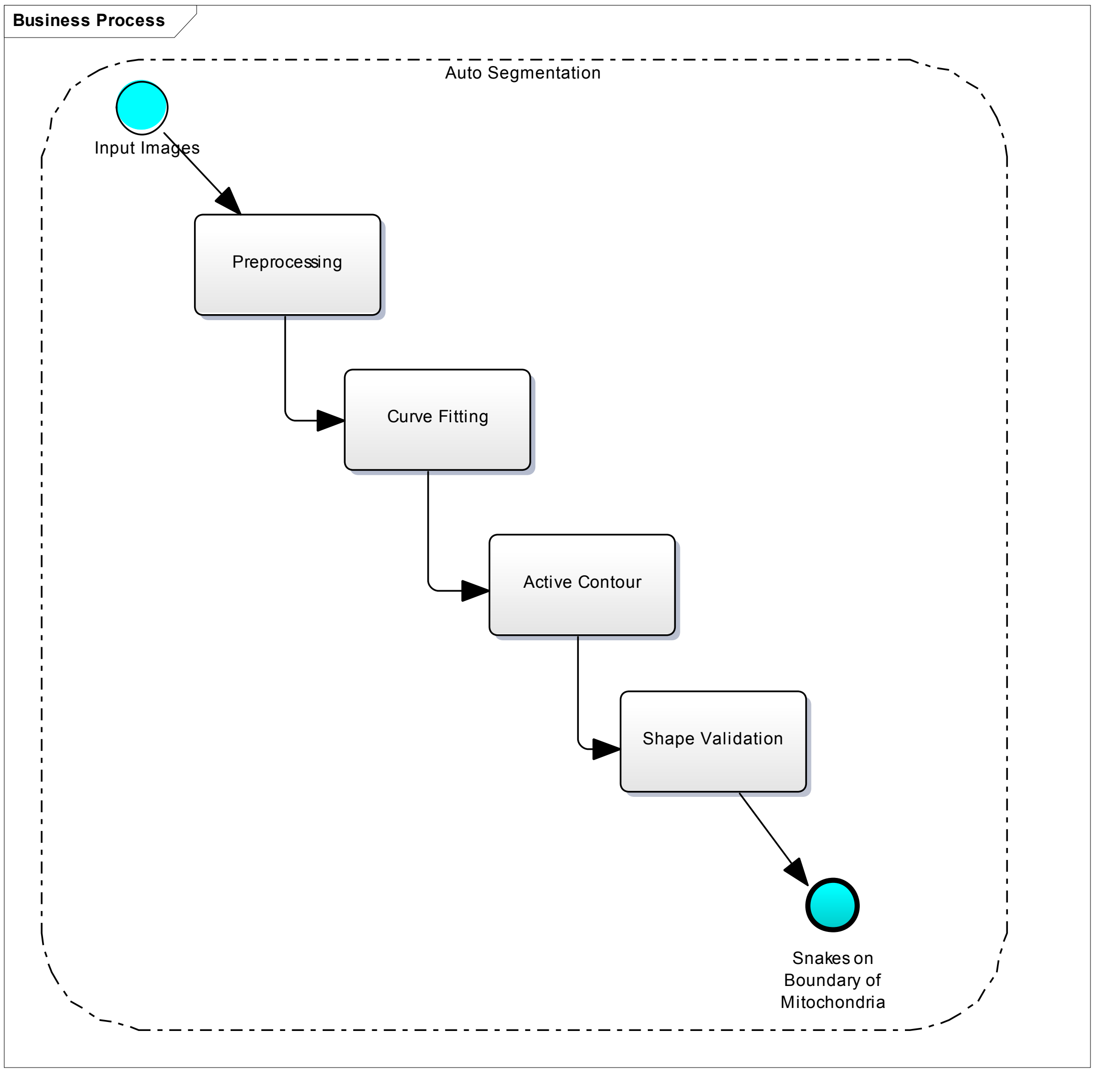


Figure 4.1 Steps of Automatic Segmentation Method in 2D

4.1.1. Preprocessing

Preprocessing step begins with the application of an auto-contrast adjustment algorithm on input images to decrease the contrast difference among different datasets and variable strength of mitochondrial membrane. This step ends with the application of a bilateral filtering[39] that is needed to decrease noise over areas that have no mitochondria.

4.1.2. Curve Fitting

The mitochondria boundary has the characteristic of large parabolic structures and low degree of curvature while a crista has smaller parabolic structures and higher degree of curvature. In this step, the algorithm benefits from the mentioned shape of the membrane. Firstly, ridges on mitochondrial membranes are detected to initiate the curve fitting process. The resolution is sacrificed by sub-sampling image of ridges to save computation time. Then, a ridge energy map is created by computing the total energy for each ridge direction inside the specified window on each sample point. The curve is started as a single point, and this curve is finalized when the algorithm finds maximized energy on the ridge energy map.

4.1.3. Active Contour

Mitochondria cannot be formed by combining curves detected in the previous step due to discontinuities on the membrane structures of the electron microscope images. Therefore, active model contour model is applied for the detection of each mitochondrion by utilizing curves detected thanks to the success of this model on broken boundaries. The minimization of internal and external energy terms defines snake structure [40]. External energy terms enforce the snake for converging on the edge of strong contours while the internal energy terms prevent deformity of the snake. Moreover, the inflation weight parameter has control on the inflation behavior of the balloon snake. Equation 1 indicates relation of these energies in the form of mathematical formulation.

$$E_{snake}\big(v(t)\big) = E_{int}\big(v(t)\big) + E_{ext}\big(v(t)\big) + E_{inf}\big(v(t)\big) \qquad (1)$$

The internal energy term is calculated by the sum of the first and the second derivative of *v(t)* which represents the vertices of the snakes with respect to *t* as indicated in Equation 2. [22].

$$E_{\mathrm{int}}\big(v(t)\big) = \frac{1}{2}\left(w_{\mathrm{a}}\left\|\frac{\mathrm{d}v}{\mathrm{d}t}\right\|^2 + w_{\mathrm{b}}\left\|\frac{d^2v}{\mathrm{d}t^2}\right\|^2\right) \qquad (2)$$

The weight parameter w_a determines the contribution of tension on the boundary of snake while the weight parameter w_b defines the contribution of curvature existence on the snake boundary to internal energy[22].

External energy is the driving force of snake boundary expansion. This energy reaches its minimum value when it faces curved structures stopping growth. Therefore, external energy is directly affected by the detected curve structures. Let us say that Ω_j means series of points of jth curve and $E(\Omega_j)$ is the total energy of the jth curve. The energy image of this curve is described as follows in Equation 3. [22]:

$$E_{\mathrm{curve}}(x,j) = \begin{cases} \dfrac{E(\Omega_j)}{|\Omega_j|} & \text{if } x \in \Omega_j \\ 0 & \text{otherwise} \end{cases} \qquad (3)$$

The total energy image is obtained by taking all curves into account as shown in Equation 4. [22].

$$E_{\text{total}}(x) = \sum_j E_{\text{curve}}(x,j) \tag{4}$$

Equation 5 indicates how external energy is calculated as the weighted version of this energy image[22].

$$E_{\text{ext}}(x) = -w_c E_{\text{total}}(x) \tag{5}$$

The weight parameter w_c arranges the contribution of external energy to total energy. Boundary of snake is initialized as unit circle and each iteration updates the boundary of snake according to Equation 6 [22].

$$v_{i+1} = v_i - \gamma \nabla E_{\text{snake}}(v_i) \tag{6}$$

The symbol γ denotes the step size of snake growth at each iteration. It is chosen to be enough not to miss the boundary of mitochondria. This update (Equation 7) continues until snake boundary is converged to convenient area[22].

$$\nabla E_{\text{snake}}(v) = \nabla E_{\text{int}}(v) + \nabla E_{\text{ext}}(v) + \nabla E_{\text{inf}}(v) \tag{7}$$

Update of inflation energy term is defined as follows in Equation 8:

$$\nabla E_{\text{inf}}(v) = -w_d \vec{n}_v \tag{8}$$

While normal vector of snake's vertex *v* determines the direction of inflation force over that point, weight parameter w_d determines the contribution of inflation force to the total energy. This weight parameter is not a fixed value. It is changing in a certain range to detect varying size of mitochondria[22].

4.1.4. Shape Validation

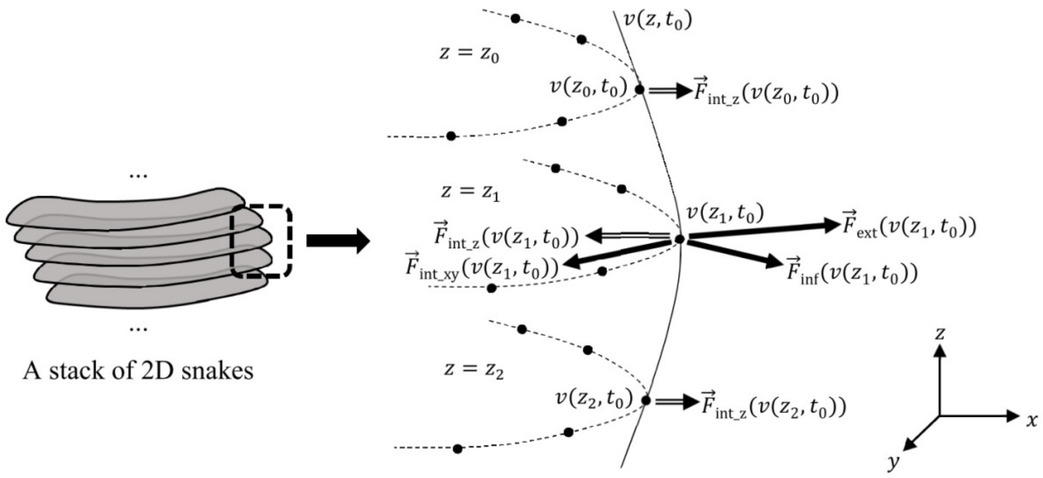
Shape energy, area, boundary continuity, curvature and signature of snakes are used as parameters for separating snakes on boundary of mitochondria from snakes that are in the non-mitochondrial regions.

4.2. 3D Automatic Segmentation Algorithm

The classical 2D snake model has a strategy of the progress of a 2D snake boundary formed by vertices whose coordinates denoted by (*x, y*) are under the control of internal and external forces A further inflation force effect the vertices in the evaluation process of snake [22]. The 3D extension of the standard snake model locates each vertex in 3D space and defines the location of the vertex by an additional z-axis. All forces that play an important role over the growth of classical snake model are defined again by the means of 3D space. This intention expands the capability of forces through the agency of change on z position of the vertices [38].

In the classical snake method, the external forces are formed by the power of the curves mostly detected on membranes. Since those curves are lying on the xy-plane, they could not be obtained as a result of processed records obtained by tomography. Therefore, there is no external force created for z-axis. Hence, the previously mentioned procedure is not applicable to 2D classical system [38].

Instead, Taşel suggests a new approach that is called as 2.5D snake model that is the process of 2D snakes in 3D coordinated space. It means that vertices can change their (x,y) coordinates with the effect of living forces, but they cannot change their initial z-position. It does not mean that 3D space does not have any effect in the growth of snake model. While external forces cannot play a role in z-axis, internal forces effect the growth of snake model crucially by using neighboring vertices in z-axis as shown in Figure 4.2.



A stack of 2D snakes
$v(z, t_0)$
$z = z_0$
$v(z_0, t_0)$
$\vec{F}_{\text{int\_z}}(v(z_0, t_0))$
$z = z_1$
$v(z_1, t_0)$
$\vec{F}_{\text{int\_z}}(v(z_1, t_0))$
$\vec{F}_{\text{int\_xy}}(v(z_1, t_0))$
$\vec{F}_{\text{ext}}(v(z_1, t_0))$
$\vec{F}_{\text{inf}}(v(z_1, t_0))$
$z = z_2$
$v(z_2, t_0)$
$\vec{F}_{\text{int\_z}}(v(z_2, t_0))$
z
x
y

Figure 4.2 A stack of snakes and acting forces on a vertex of the snake[38]

Internal force consists of the effect of the internal force vector lying on xy-plane and the effect of the internal force vector on z-axis as Equation 9.

$$\nabla E_{\text{int}}(v) = \nabla E_{\text{int\_xy}}(v) + \nabla E_{\text{int\_z}}(v) \qquad (9)$$

where;

$$\nabla E_{\mathrm{int\_xy}}(v) = w_{\mathrm{at}} \frac{\partial^2 v}{\partial t^2} + w_{\mathrm{bt}} \frac{\partial^4 v}{\partial t^4} \qquad \textbf{(10)}$$

$$\nabla E_{\mathrm{int\_z}}(v) = w_{\mathrm{az}} \frac{\partial^2 v}{\partial z^2} + w_{\mathrm{bz}} \frac{\partial^4 v}{\partial z^4} \qquad \textbf{(11)}$$

Equation 10 and 11 reveals the details of the internal force described in Equation 9.

CHAPTER 5

VISUALIZATION METHODS

While the automatic method for segmentation and detection of mitochondria takes a big load off the user, it still leaves some problematic cases which need to be resolved by the user himself. In some cases, two mitochondria that are very close to each other can be detected as one mitochondrion, or one mitochondrion could be detected as two mitochondria due to discontinuities on boundary of membranes. These cases are challenging for the automatic algorithm due to insufficient information and low image qualities. Therefore, the need of user perception arises to guide the algorithm for better segmentation and detection. Firstly, the user needs to analyze the problematic case and give necessary input to this or another algorithm in order to fix the problematic case. Precisely at this point, there is a need for the visualization of wrongly detected mitochondria for the user to decide and analyze the situation in a much better way. Therefore, an important part of this thesis covers the visualization of mitochondria to give opportunity to researchers for evaluation of datasets and detected mitochondria. Visualization part of this study mainly consists of two main phases that are visualization from transmission electron microscope images and visualization from snakes detected by the automatic segmentation algorithm. The details of these phases are elaborated in the following sections. Before doing that, it is better to explain the principle of VTK used to achieve visualization and 3D reconstruction.

5.1. Visualization Tool Kit (VTK)

The Visualization Toolkit (VTK) [41] is a free and open-source software package for 3D computer graphics, image processing and visualization. VTK is based on the C++ library and several interoperations of this library in different programming languages such as Tcl/Tk, Java, and Python. Kitware which is the creator of the toolkit continues to improve the system for further capabilities and extensions. VTK includes many visualization algorithms containing: scalar, vector, tensor, texture, and volumetric methods; and advanced modeling techniques. VTK has a set of 3D interaction widgets, implements parallel processing, and work with many GUI toolkits like Qt and Tk. VTK runs on multiple operating systems like Linux, Windows, Mac and Unix.

5.1.1. Architecture

VTK is formed by two main parts that are C++ core implementation and an interpretation of core in different wrapper libraries [42]. The interpreted layer consists of Tcl/Tk, Java, and Python currently. C++ core implements basis infrastructure that consist of data structures, algorithms and time-critical functions. While speed and efficiency are featured properties in the heart of core implementation, interpreted layer provides flexibility and extensibility. Layers of architecture for a VTK's application are shown in Figure 5.1 below.

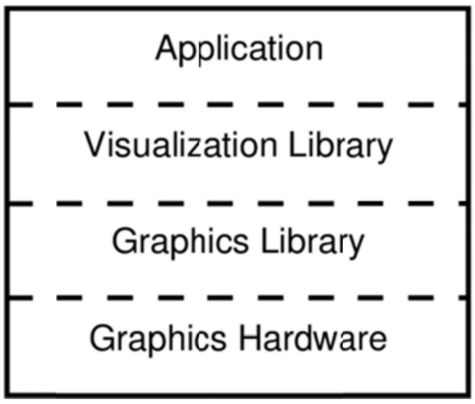


Figure 5.1 VTK Architecture

VTK is the part of visualization library as indicated in Figure 5.1. It is constructed on graphics library which is responsible for the interaction with the graphics hardware. A developed application is isolated from existence of graphic library and graphics hardware by using VTK. This abstraction philosophy helps the application developer deal with lower layers.

5.1.2. The Graphics Model

The graphics model forms an abstract layer above the graphics library to guarantee cross-platform portability [42]. Since there is no guarantee for any graphic library such as OpenGL to become standardized in the industry, VTK has an implementation of abstraction for that layer. It has an extensible architecture for new coming graphic libraries.

The names of the classes in the graphics model are drew inspiration from the film industry [42]. Lights, cameras, actors, and props are implemented classes that are instantiations used to generate a scene. The graphic model driven for 3D polygonal rendering consisting lights, cameras, actors is similar to the other model implemented for volume and other types of rendering including lights, cameras, and volumes. Illustration of VTK's Graphics Model is shown in Figure 5.2.

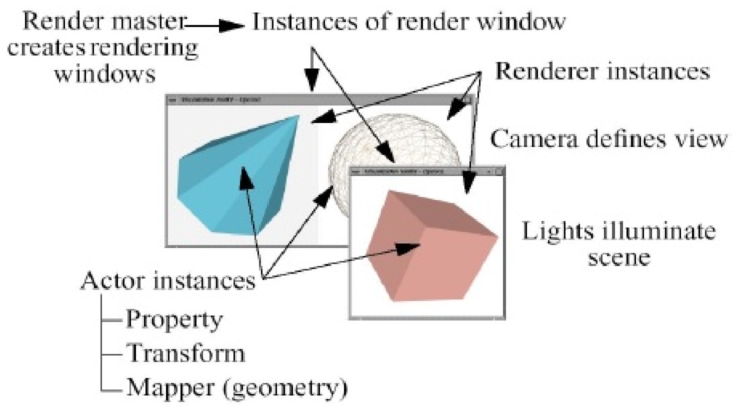


Figure 5.2 VTK Graphics Model [43]

There is an order of implementation for these classes while application is developed. This order is called the visualization pipeline explained in the following section.

5.1.3. The Visualization Pipeline

VTK's visualization pipeline turns source data into structures that can be used by the graphics subsystem for display purposes or turns them into other forms that the pipeline can work on [42]. Many operations in VTK are succeed by the use of a pipeline architecture where multiple elements are connected together to complete a display process [44]. The details of the elements are showed and explained in Figure 5.3 [44].

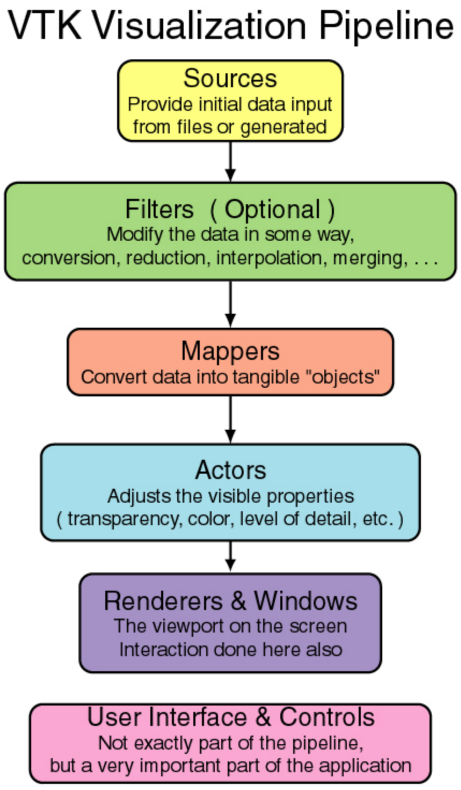

VTK Visualization Pipeline
Sources
Provide initial data input from files or generated
Filters (Optional)
Modify the data in some way, conversion, reduction, interpolation, merging, . . .
Mappers
Convert data into tangible "objects"
Actors
Adjusts the visible properties (transparency, color, level of detail, etc.)
Renderers & Windows
The viewport on the screen
Interaction done here also
User Interface & Controls
Not exactly part of the pipeline, but a very important part of the application

Figure 5.3 VTK Visualization Pipeline [45]

Sources: These classes prepare input data for the rest of the pipeline. For instance, there are classes such as vtkJPEGReader reading image from a specified file and producing a convenient data for the next step. Moreover, other classes such as vtkConeSource generates virtual objects with specified parameters of developer.

Filters: These filters run on input sources to apply desired modification on these sources. For example, vtkImageGaussianSmooth function on a source image data to apply Gaussian smoothing algorithm and to generate a new smoothened version of source image with the specified parameters.

Mappers: These describe the connection between source data and graphics primitives or rendering techniques. More than one mapper could use the same source as input, but could process it with different methods.

Props/Actors: Mapper provides the necessary input to actors for generation of the visible representation of input data. Property of actors such as color contains the information of how actors are displayed. Every actor should have their own mapper as provider of source input.

Renderer: Renderer is an entity that manages the rendering process for actors. Rendering is the operation of transforming actors, lights, and cameras in the scene to an image. Coordinate transformation among worlds coordinates, view coordinates which is coordinate system of the graphics rendering, and display coordinates which is the actual screen coordinates on monitor is managed at this step.

Render Window: The Render Window is the interface through which the user can see the results of the rendering process. This window provides the user interaction with application with the help of input devices such as mouse and keyboard.

5.2. Visualization from TEM Images

In the previous section, we examined steps of VTK's visualization pipeline that end with resulting rendered picture. We will examine the work done for the visualization in the thesis with the same manner. Firstly, it is better to introduce steps of evaluation components from mitochondrial images. As indicated in Figure 5.4, volume rendering process gets the images and constructs a volume from them. From this volume, anatomical planes and a region of interest (ROI) is extracted.

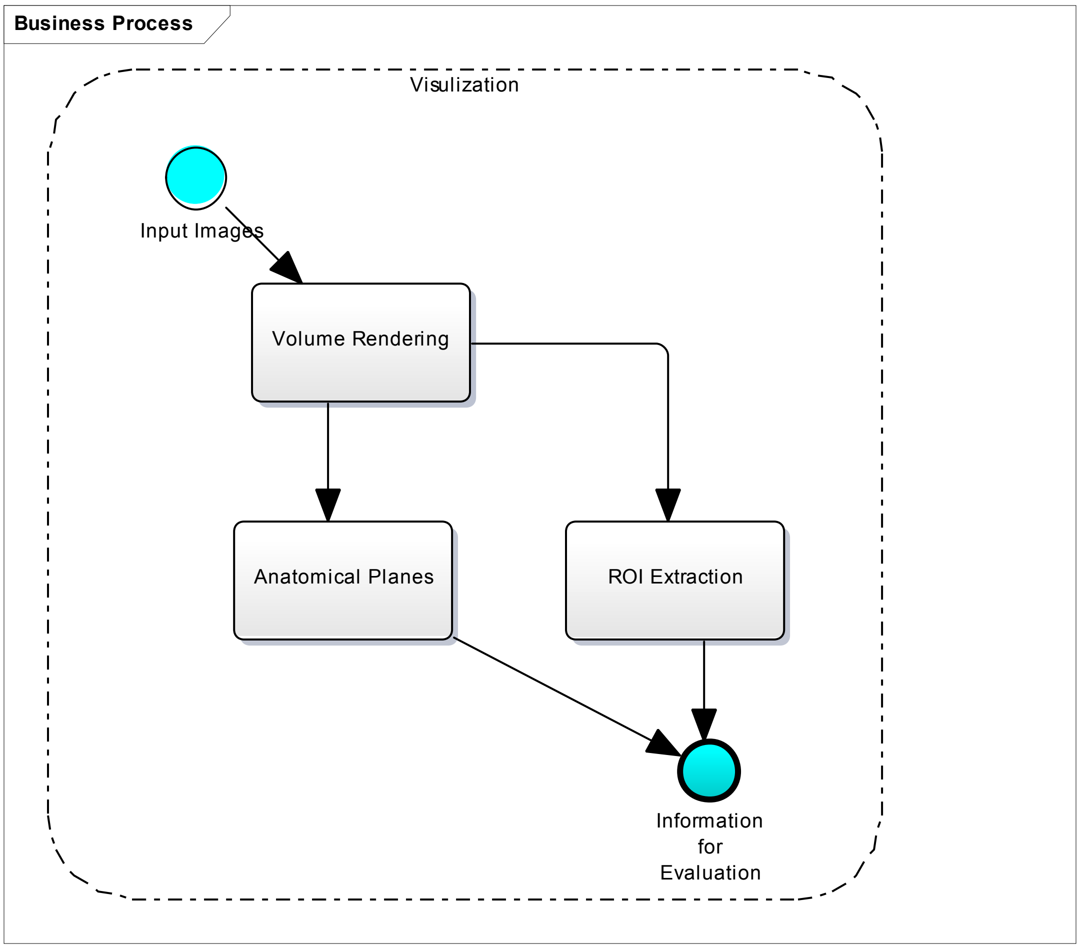


Figure 5.4 Steps of Visualization Phase in 3D

Sources of the implementation are directly coming from original mitochondrial TEM images that are used for segmentation in the automatic algorithm. To provide harmony between snakes' data that are output of automatic segmentation algorithm, some steps of preprocessing phase of that algorithm are also applied here. These steps are resampling of the images and extracting region of interest from all images with the specified value. These preparation steps are operated on OpenCV images in Tasel's automatic algorithm. To provide the reuse same sources with the automatic algorithm, the implementation in this work also begins with the same preprocessing steps. In the end, they are resampled and region of interest extracted slices of OpenCV images are obtained since visualization phase of the thesis is basically using VTK as a volume rendering tool, there is a necessity of converting these OpenCV images to VTK images. Original images have some useless regions near the edges of the images. Moreover, resolution of original image should be rearranged to provide consistency with the automatic segmentation algorithm. Preprocessing phase gets rid of these unnecessary near edges and rearranges resolution of the images according to parameters used in automatic segmentation algorithm using OpenCV image operations. After this process, this image is converted to VTK Image format for further work to construct visualization components. The summary of this preparation is given in Figure 5.5.

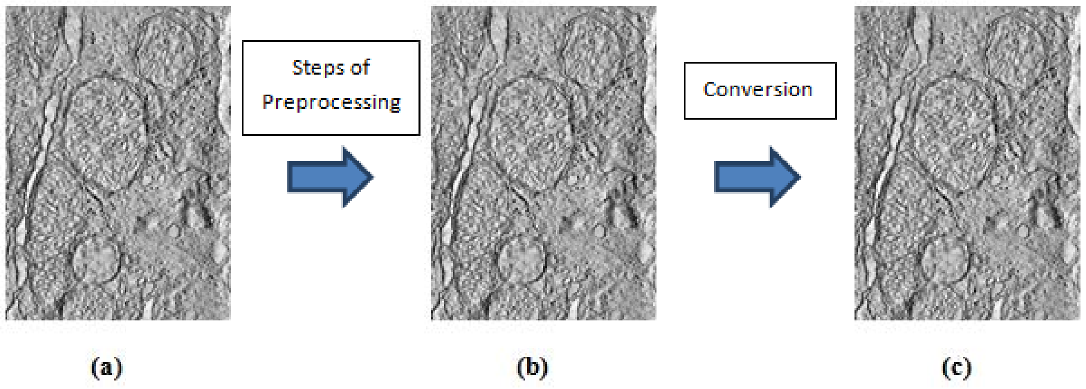


Figure 5.5 Conversion From OpenCV Image to VTK Image
(a) OpenCV Image (b) Processed OpenCV Image (c) VTK Image

After this preparation, 2D slices of mitochondrial images are ready to be constructed as 3D volume. Pavone et al [32] show the process as in the figure below, which summarizes knowledge behind the construction of volume from these 2D slices. Slices of images come together and gaps between slices are filled by interpolation techniques which are based on pixels between two neighborhood images. As a result of this construction, 3D volume is obtained as visualized in Figure 5.6.

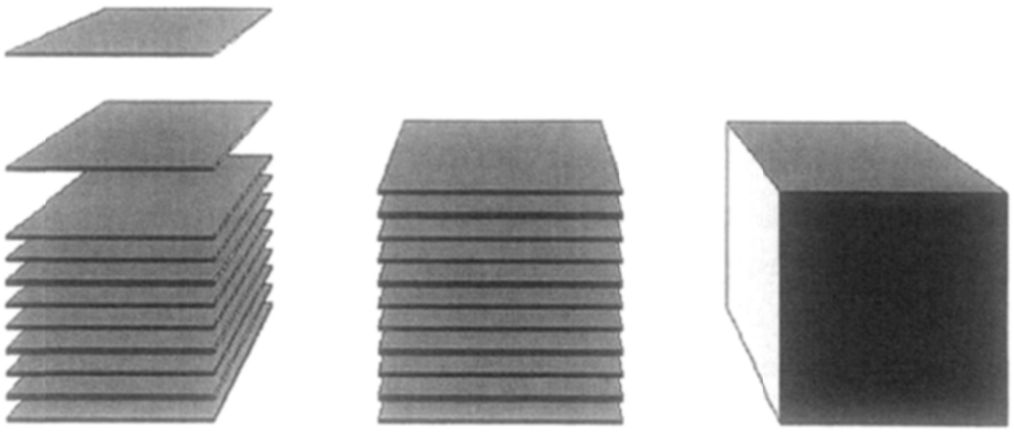


Figure 5.6 Construction of Volume From Slices by Interpolation[32]

In above figure, the second shape, which is the collection of slices, are obtained by vtkImageAppend class that receives the components from multiple sources and merges them into one resulting output. VTK brings these images together with the help of appending classes of its own implementation. After the combination of the data of slices, VTK's filter is activated to make changes on this data. As can be seen from Figure 5.4, Two phases that are the creation of anatomical planes and extraction of volume of interest are using two different filters and renderers to obtain the desired result.

5.2.1. Anatomical Planes

In this part, anatomical planes from the source data are tried to be obtained. Anatomical planes divide the body of an object into two parts, which can be seen in Figure 5.7.

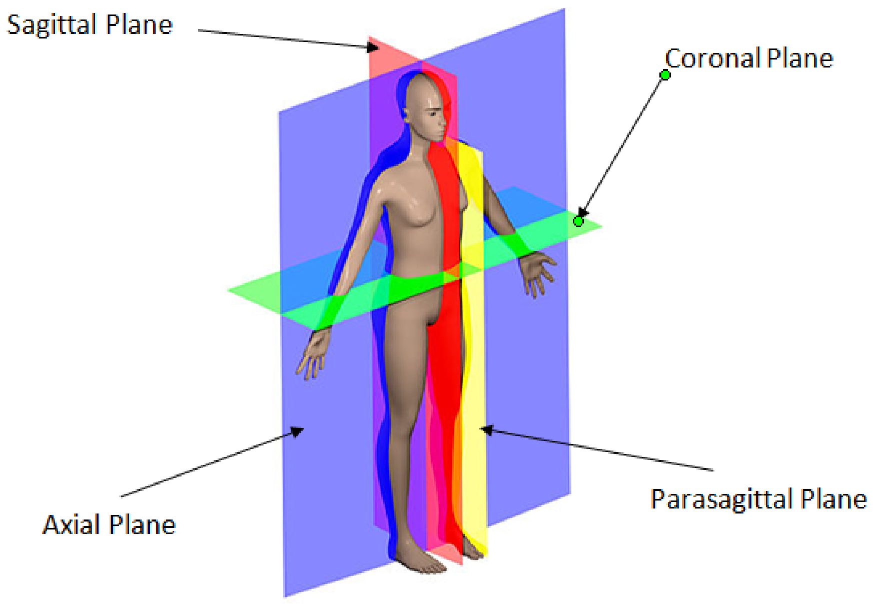


Figure 5.7 Anatomical Planes Transecting Human Body[46]

Axial planes divides the body of the object into two parts as up and down, sagittal plane does it as left and right, coronal plane does it as front and rear. Parasagittal plane is not used in our application. Merged data coming from slices of images can be considered as body of the object. Application of this thesis is getting the selection point from the user to define the intersection point of all planes, which is called the center point. With this information and source data, one of VTK's filter is applied to cut a slice from 3D source object. This filter is class vtkImageReslice. It takes the source data and orientation matrix for the desired plane and gives the resulting 2D image for further use of pipeline. In Table 5.1, the orientation matrices for anatomical planes can be analyzed separately.

Table 5.1 Orientation Matrices for Anatomical Planes

Anatomical Plane	Orientation Matrix
Axial Plane	$\begin{bmatrix} 1 & 0 & 0 & c_0 \\ 0 & 1 & 0 & c_1 \\ 0 & 0 & 1 & c_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
Coronal Plane	$\begin{bmatrix} 1 & 0 & 0 & c_0 \\ 0 & 0 & 1 & c_1 \\ 0 & 1 & 0 & c_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
Sagittal Plane	$\begin{bmatrix} 0 & 0 & 1 & c_0 \\ 1 & 0 & 0 & c_1 \\ 0 & 1 & 0 & c_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
Oblique Plane	$\begin{bmatrix} x_0 & x_1 & x_2 & c_0 \\ y_0 & y_1 & y_2 & c_1 \\ z_0 & z_1 & z_2 & c_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Each column in the orientation matrices indicates the unit vector of each axis in world coordinate system. The first column indicates x-axis, the second one does y-axis and the third one does z-axis. The last column is the center point, which is the intersection point of all planes. Moreover, there is a possibility of cutting a slice with an oblique plane with the specified unit vectors and parameters.

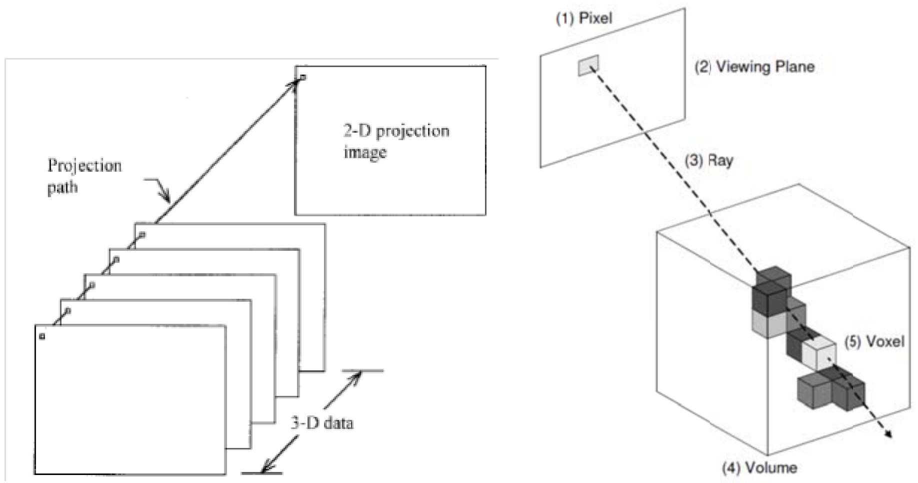
VTK's filter takes the source data and the orientation matrix to obtain the resulting image on the desired plane as mentioned before. After this operation, the modified data is directed to mapper class for the generation of the object to be rendered and displayed. vtkImageActor class is used to draw image in a rendered 3D scene as an actor in the pipeline. After renderer adds the image actor to the scene, the user can see and interact with the extracted image inside the render window.

5.2.2. Region of Interest (ROI) Extraction:

In this part, rendered volume is tried to be obtained from the source image volume data that is used to obtain anatomical planes In the visualization pipeline, filters are the next step after sources as mentioned before. They are capable of modifying the source data. A region of interest inside this image volume data needs to be examined. VTK have a filter called vtkExtractVOI class selecting a portion of an input structured points dataset. A region of interest is extracted from an image volume according to user selection on anatomical planes. The user can see a cubical part of the mitochondria around the user selected point, which is basically the intersection of anatomical planes. After the modification of the filter, the source is directed to mapper class generating an object for rendering and display.

The device display has 2D interface with the user. Therefore, 3D image volume should map to 2D display. Rotation of the object gives the 3D perception to the user at the end. Mapping from 3D data points to 2D display is done by various algorithms like volume ray casting, texture-based volume rendering, maximum intensity projection and minimum intensity projection. Nature of the mitochondrial image volume allows better performance for perception of user if minimum intensity projection algorithm is used to render the volume.

Minimum intensity projection (MinIP) is the variation of maximum intensity projection (MIP) which is invented by Wallis et al [47] to be used in nuclear medicine. In MIP method [31], there are viewing rays between the object to be displayed and the estimated operator position in front of the screen. Only relatively maximum intensity value along each ray that can be called projection path is put on 2D display device. As shown in Figure 5.8, a trace ray is passing through the object and maximum intensity value of the object along ray is displayed on the viewing plane.



(a) **(b)**

Figure 5.8 Display of 3D Object on 2D Display with MIP Method [31]
(a) 2D Projection of an Object on Screen (b) MIP Method

The MinIP algorithm is very similar to the MIP algorithm. However, in the case of MinIP, low-intensity value in an input volume along trace ray is displayed on the screen [48]. Since mitochondria have much lower intensity values on mitochondrial images, MinIP is used as the volume rendering technique in the application of this thesis. Implementation of MınIP is made by adding new MinIP class to the VTK library that is the variation of VTK's MIP class that already existed. This class is vtkVolumeRayCastMIPFunction that is a volume ray cast function of mapper class that computes the minimum value encountered along the ray. Mapper class is using this algorithm while directing the object to rendering process. vtkVolume is another type of actor that takes the mapper class with its guidance of rendering process of object and represents a volumetric entity in a rendering scene. Finally, renderer adds this volume to render window and the user can examine the volume through interaction of mouse/keyboard.

5.3. Visualization from Snake Information

Input of this phase is directly coming from automatic segmentation algorithm that detects mitochondria in every slice done by Tasel [22]. The resulting data is hundred points for each mitochondrionon each slice. These points are located on the boundary of the mitochondria and they give the shape information of mitochondria apparently. What is named for this data for one slice is snake, which is harmonious with the name of the algorithm that used to detect them. While collecting the data coming from snakes for each mitochondrion on every slice, what is obtained is the point cloud in 3D environment. It is some valuable information for the user, who is examining these mitochondria boundary points, but reconstructed surface on these points is giving the user a better sense to analyze related organelle due to its simplicity with only one constructed structure compared to hundreds of points. The main flow of this phase is shown in Figure 5.9.

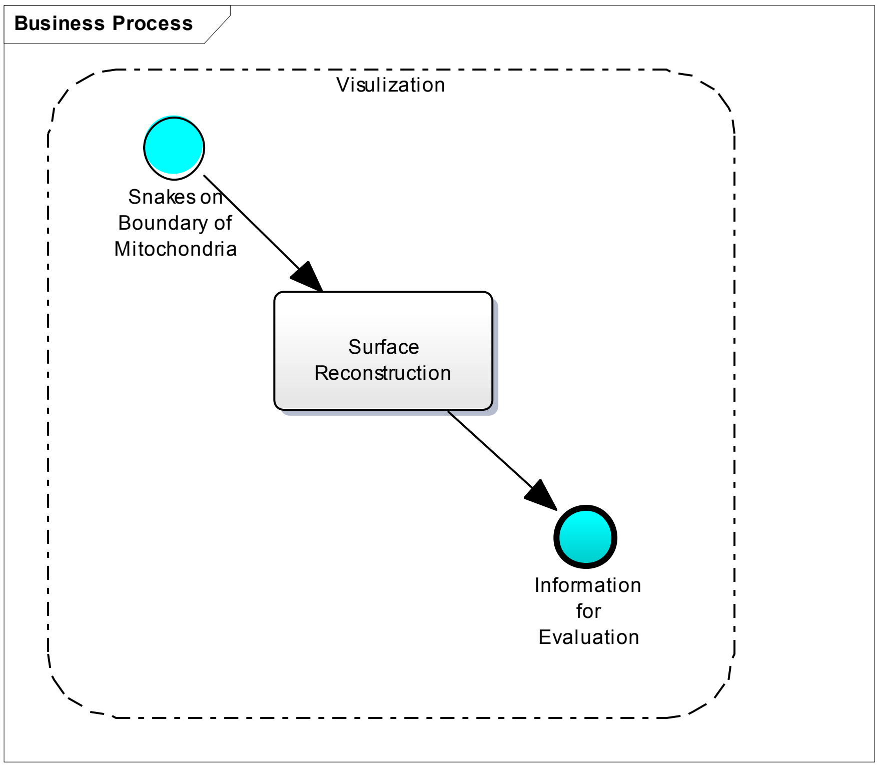

Business Process
Visulization
Snakes on Boundary of Mitochondria
Surface Reconstruction
Information for Evaluation

Figure 5.9 Visualization from Snake Information

5.3.1. Surface Reconstruction

There are various modeling algorithms such as scalar, vector, tensor for the reconstruction and visualization of virtual data [49]. In this thesis, we are focusing on scalar methods that are implemented in VTK library. Precisely, main focuses are two methods that are surface reconstruction from unorganized points and marching cubes algorithm are implemented for the purpose of surface reconstruction from boundary points on mitochondria

5.3.1.1. Surface Reconstruction from Unorganized Points

Hoppe et al [50] introduces a general method to reconstruct correct, terse, piecewise smooth surfaces from random 3D points automatically. The method described in the thesis of Hoppe needs only the 3D coordinates of the points as input for the algorithm. The method is implemented by three main phases that are initial surface estimation, mesh optimization and piecewise smooth surface optimization, details of which are given in the doctoral thesis of Hoppe.

In the phase of initial surface estimation, initial dense mesh is constructed by using 3D unorganized points that are the input of the algorithm. After that, optimized mesh is obtained by an application of an algorithm to decrease the number of surfaces and develop fitting on points. Lastly, surface of the object is converted from piecewise linear one into piecewise smooth one.

VTK library implements Hoppe's method in a filter class that is called vtkSurfaceReconstructionFilter. In our case, this filter, responsible for modifying source data, takes the points of mitochondria boundaries for every slice organizing a piecewise smooth surface. After that, mapper class takes the role to map polygonal data to graphics primitives. Actor class takes the output of the mapper class and becomes the object of the rendering scene after properties of entity area are applied. Renderer takes care of the rest of the operation to display mitochondrial smooth surface in render window.

5.3.1.2. Marching Cubes Algorithm

Lorensen and Cline introduced the algorithm as a result of their research for General Electric [51]. Marching cubes algorithm followed the divide-conquer principle while constructing the surface from images or points. In the implementation of the algorithm, there is an imaginary cube between the slices of volume and this cube determines the surface pieces of all destinations while passing through between slices.

The surface pieces of each region are calculated according to the value gained on vertices of arbitrary cube for each location. Weights of the vertices are determined by each application itself. As a result of this weight distribution, the surface is drawn according to the special cases of marching cubes algorithm shown in the figure below. In the end, there are many surface pieces all over the object and these pieces come together to form a visual image of the object.

VTK library implements the marching cubes algorithm in a filter class named as vtkMarchingContourFilter. This filter takes data points as input for generating output isosurfaces. Mapper class follows VTK pipeline by taking the description of this surface construction and transferring necessary information to actor class that is used to put on screen by renderer.

CHAPTER 6

SEMI-AUTOMATIC SEGMENTATION METHODS

Automatic segmentation techniques take over workload of researchers to a significant level, which attracts academic attention for further research. Even though these techniques have a significant contribution to their work, there are still some errors in their decisions regarding segmentation and detection due to image qualities or missing information on dataset. Sometimes, these wrongly detected segmentation and detection behavior requires user guidance and human perception to be fixed. Where automatic segmentation is not enough, human aid takes place to reach more accurate result. Therefore, semi-automatic ways for detection and segmentation in medical image processing play a significant role.

In the case of automatic segmentation and detection of mitochondria from TEM images, there is some wrong decision about detected mitochondria as could be in every other segmentation examples due to lack of valuable information. System performance of automatic segmentation by Tasel [22] reaches a prediction ratio of 75 percent. Since automatic segmentation technique cannot cover some mitochondria on dataset, we propose semi-automatic techniques to reach correct detection for the undetected mitochondria as much as possible.

Effective visualization and interaction with the outputs of the automatic segmentation method has a vital importance for user evaluation on resulting mitochondria. Anatomical planes and oblique plane are constructed, region of interest in mitochondrial images volume is visualized, surface is reconstructed from mitochondria boundaries in the visualization process of thesis as mentioned before. This phase is aimed to help the user to make necessary evaluation on the resulting mitochondria and allow him/her to correct wrong outputs.

After obtaining the results from the automatic segmentation algorithm, the user evaluates the data and the visualized object; interact with data by using semi-automatic operations such as splitting, merging and deletion to fix the problematic case. These are major approaches for user guidance and semi-automatic ways for correcting wrongly detected mitochondria by the automatic segmentation algorithm. Business process of the semi-automatic segmentation can be seen in

Figure 6.1 that begins with the user evaluation phase and followed by the operations that will be explained in the coming sections of this thesis.

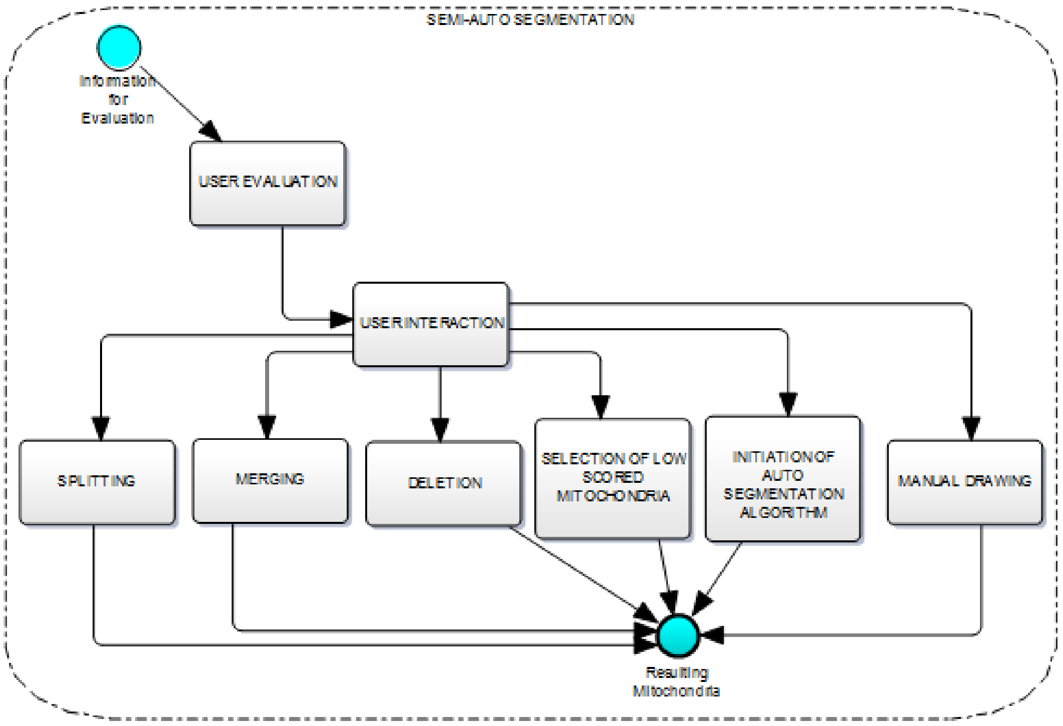


Figure 6.1 Semi-Automatic Segmentation Business Process

There are some studies on only 2D images to recognize the nature of mitochondrial input images better and define the problem in a more accurate way. The experience and information will be used to extend the solution for 3D case of the problem.

While the results of the 2,5D automatic detection algorithm are satisfactory for extracting the necessary information from electron microscopy images, they are still not covering all the existing mitochondria on images. The results give valuable information about mitochondria located in images, but segmentation and detection task is not completed with only these conclusions. While some of the results need corrections to reach real shape as much as possible, some mitochondria could still not be detected despite these facts, the automatic algorithm leaves valuable outputs giving chance of improvement by working on them. Splitting, merging, rejection, manual segmentation, evaluating rejected ones again and activation of automatic algorithm with soften parameters are approaches that will be investigated in these sections. These semi-automatic methods contribute the segmentation of mitochondria not by entering details in shape but mostly, concentrating on the correction of the existing mitochondria in that specific image. There is a method for surface dragging, which focuses on adjustments of boundaries[25]. This method will not be investigated in this thesis, but there will be a couple of ideas about it in the future works section.

Before beginning the description of semi-automatic methods, it is better to introduce common approaches that help the construction of 3D surface in various parts of these methods. The main motivation is coming from the idea of knitting spider web in space. Spider web is formed by gathering yarns, and yarn corresponds a parabola in our world, while web corresponds a 3D surface. We investigate the formation of a parabola and distribution of random points on a snake. References to these methods will be given throughout the rest of this thesis.

Finding Point on Parabola by Initial Points

Definition of a parabola can be made using at least three or more different methods. A second order parabola is formulized in Equation 12, which we will deal with:

$$y = a + bx + cx^2 \qquad \textbf{(12)}$$

To find the equation of a parabola, we need to detect coefficients shown in the above formula. To minimize the user interaction with the application, we limit the points with an amount of four that are lying on xy-plane. It is critical to find the right equation from these points since there are more than one parabola passing through these points. Input sequence of these points determines the start and end points of the parabola, which keeps apart the demanded parabola from others. To adapt the required parabola the equation works in the following manner:

1. Draw a line between the start and the end point.
2. Rotate the points till the drawn line fits a parallel position to x-axis.
3. The equation and points on the parabola will be found according to this orientation. After this process, the initial points will be rotated back to their original positions with all the detected points.

Figure 6.2 shows the details of the algorithm visually.

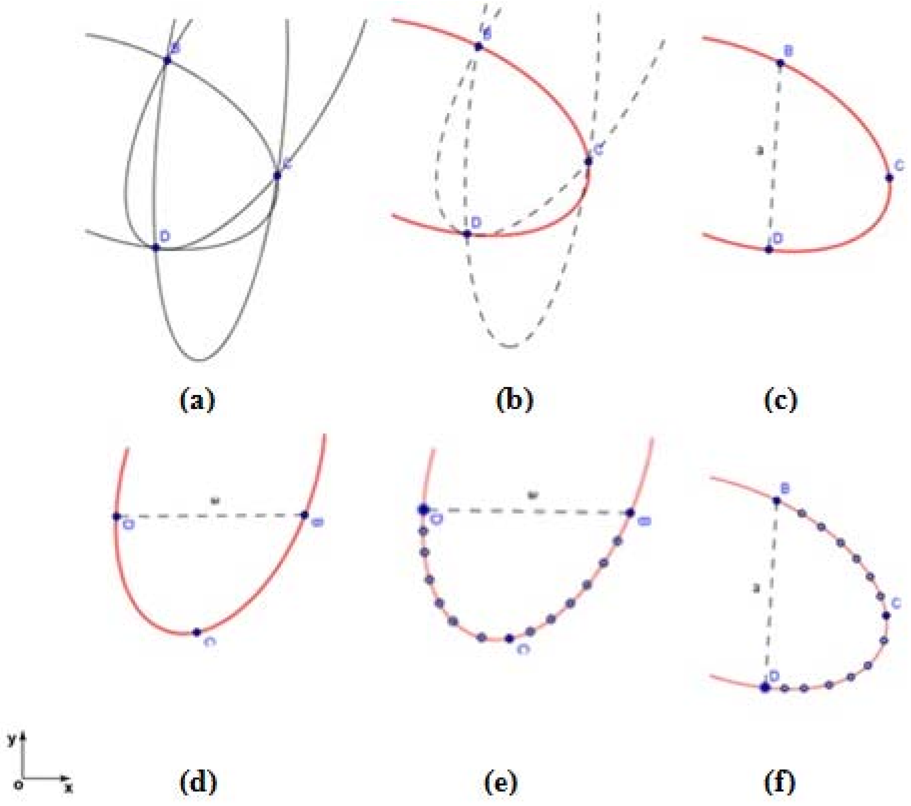


Figure 6.2 Finding Points On Parabola by Initial Points
(a) Parabolas passing through randomly located points. (b) The order of point's definition determines desired parabola. (c) Definition of the line from start point to end point. (d) Points are rotated till line becomes parallel to x axis to get necessary equation convenient to Equation 12. Equation of parabola passing through these points is found. (e) Points on parabola. (f) Rotate back the points to older position.

It is time to find the equation of the parabola using this orientation. We investigate the found equation that is a three or four points based parabola. Simple solution to a system of linear equations finds the equation of the parabola with three points as of (x1, y1) , (x2, y2) and (x3, y3).

$$A\,\underline{x} = \underline{b}$$

where;

$$\begin{aligned} a + bx_1 + cx_1^2 &= y_1 \\ a + bx_2 + cx_2^2 &= y_2 \\ a + bx_3 + cx_3^2 &= y_3 \end{aligned} \qquad (13)$$

The above linear system given in Equation 13 can be expressed in the following matrices described in Equation 14;

$$\underbrace{\begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} a \\ b \\ c \end{bmatrix}}_{\underline{x}} = \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}}_{\underline{b}} \qquad (14)$$

By solving the linear system above, the coefficients of the parabolic function are determined. There could be cases where a linear system is indeterminate if the selected points are inconsistent with each other. In such cases the matrix of the system becomes singular. This condition is checked by controlling whether determinant of this square matrix is equal to zero. In order to eliminate such cases, a change is applied on the positions of the points that does not change the characteristic of the parabola significantly. By using this parabolic function, points between the start and the end points are filled with a certain interval between each point.

There is no guarantee that a second order parabola fits on random four points. For the best fit of the given parabolic function above, least square regression approach is used. In this method, the best approximation is described as the minimization of the sum of squared differences between the data values [52]. For the user given data set, (x_1, y_1) , (x_2, y_2), (x_3, y_3) and (x_4, y_4)., best fitting parabola is found by the following way shown in Equation 15 and 16;

$$\mu = \sum_{i=1}^{4} [y_i - f(x_i)]^2 \qquad (15)$$

$$\mu = \sum_{i=1}^{4} \left[y_i - (a + bx_i + cx_i^2)\right]^2 \qquad (16)$$

We need to determine unknown coefficients with the given four points. To calculate the least square error, the first derivate of the function with respect to the coefficients *a*, *b* and *c* have to be equal to zero respectively, which is described in Equation 17, 18 and 19.

$$\frac{\partial \mu}{\partial a} = 2\sum_{i=1}^{4} \left[y_i - (a + bx_i + cx_i^2)\right] = 0 \qquad (17)$$

$$\frac{\partial \mu}{\partial b} = 2\sum_{i=1}^{4} x_i\left[y_i - (a + bx_i + cx_i^2)\right] = 0 \qquad (18)$$

$$\frac{\partial \mu}{\partial c} = 2\sum_{i=1}^{4} x_i^2\left[y_i - (a + bx_i + cx_i^2)\right] = 0 \qquad (19)$$

By expanding the above equations, we have the following calculations shown in Equation 20, 21 and 22 .

$$\sum_{i=1}^{4} y_i = a\sum_{i=1}^{4} 1 + b\sum_{i=1}^{4} x_i + c\sum_{i=1}^{4} x_i^2 \quad (20)$$

$$\sum_{i=1}^{4} x_i y_i = a\sum_{i=1}^{4} x_i + b\sum_{i=1}^{4} x_i^2 + c\sum_{i=1}^{4} x_i^3 \quad (21)$$

$$\sum_{i=1}^{4} x_i^2 y_i = a\sum_{i=1}^{4} x_i^2 + b\sum_{i=1}^{4} x_i^3 + c\sum_{i=1}^{4} x_i^4 \quad (22)$$

By solving the above linear equations, unknown coefficients, *a*, *b* and *c* can be found. Points between start and end points are filled with a certain interval between each point using this formula,

6.1. Splitting

As a result of 2,5D automatic detection algorithm, one mitochondrion can be detected instead of two real ones due to lack of curve structures between boundaries of neighbor mitochondria or excess inflation force in related case. With the help of user observation and guidance to the algorithm, our aim is to differentiate these two separate mitochondria from each other. There are two implemented methods to achieve this goal. While one of them is splitting mitochondria from each other without the help of the 2,5 D automatic algorithm, the other one makes a contribution to the model and guide the standard automatic detection. Before further explaining these methods, we need to describe how they are separated from each other by user help. In other words, it is necessary to clarify how user interacts with mitochondria to specify the points of separation.

6.1.1. Creating Splitting Surface

The user needs to describe some shapes between two mitochondria wrongly detected as one. The easiest method is to put a barrier between two mitochondria that is constructing a plane between them. By arbitrarily defining three points, a plane could be easily formed to separate mitochondria from each other vertically as shown in Figure 6.3.

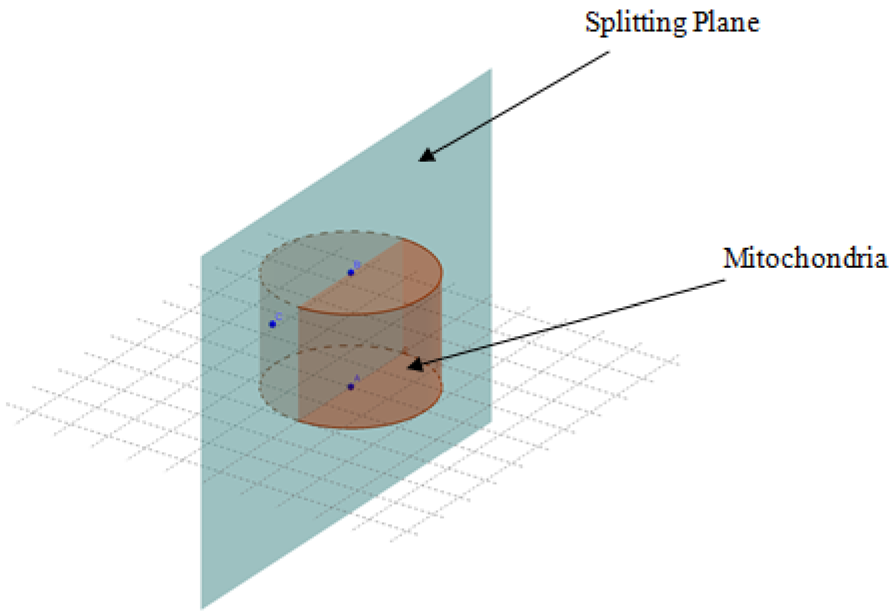


Figure 6.3 Illustration of Mitochondria with Splitting Plane

This plane corresponds a line for each slice of mitochondria. We can calculate specific force by the use of this line in each slice as shown in the figure below. Force vectors that are perpendicular to line can keep away the snake boundaries that are approaching to them during the run of automatic segmentation algorithm as indicated in Figure 6.4.

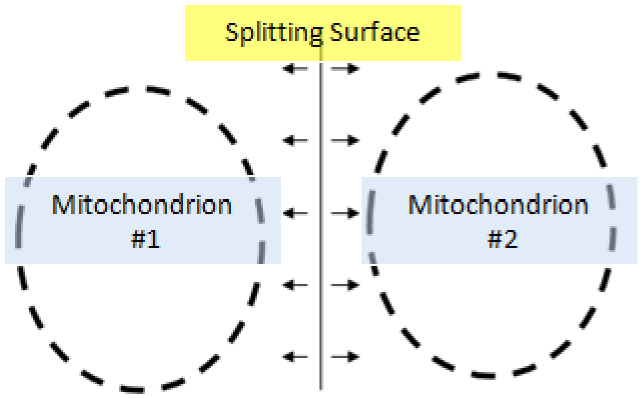
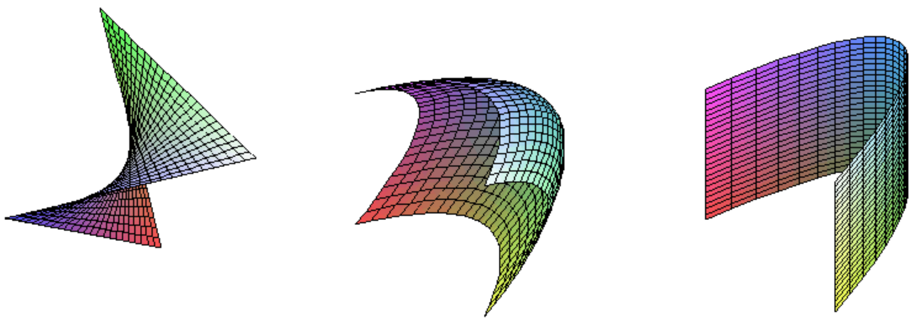


Figure 6.4 Illustration of Growth of Snakes Separated by Line on a Slice

Although it is very simple to implement a plane, this shape is a very restrictive and flat structure that is not convenient for every case. Moreover, there could be other options like defining paraboloid or hyperboloid, but they still keep the application within specific molds and probably cannot handle sudden changes of shapes of boundaries by means of concavity and convexity. Therefore, necessity of a more general form of a splitting surface arises to handle mitochondria that are neighbors of each other with various shapes of border in 3D manner. Quadratic surface which has been already covering different types of shapes like plane, paraboloid, hyperboloid is the name of the general form of this 3D shape. To define a second order quadratic surface, Equation 23 shown below should be brought out with all variables [53].

$$ax^2 + by^2 + cz^2 + 2fyz + 2gzx + 2hxy + 2px + 2qy + 2rz + d = 0 \quad (23)$$

One strategy could be to resolve Equation 23 through complicated mathematical calculations with sufficient data. However, our main goal is not to obtain the formula of quadratic surface from points, but approach an acceptable quadratic surface with the information of coordinates. Our strategy will be reducing the complexity of mathematical equations with minimum set of required data that critically affect the user effort while defining a surface. Some examples of the quadratic surfaces are given in Figure 6.5.



(a) **(b)** **(c)**

Figure 6.5 Examples of Quadratic Surfaces
(a) Elliptic Paraboloid (b) Hyperbolic Paraboloid (c) Parabolic Cylinder

There are other various ways of fitting a surface on points in 3D like Moving Least Squares [54] by Lancester et al and Surface Fitting Algorithms like the work done by Moonhong et al [55]. Implementations of these algorithms are not easy and costly since it is needed to calculate complicated mathematical equations. Such kinds of algorithms are very beneficial to fit a surface over point cloud which requires scanned data coming from some source. It is not a good idea of taking those kinds of data from the user. Our other goal is to define a surface with the least possible interaction of the user with model to reduce the interaction time.

While constructing the algorithm, the motivation is to use advantages of the application interface to make the operation of surface construction easier. The application has an axial, sagittal and coronal view of the constructed volume of dataset. The user can observe and roam inside the volume by interacting with those views. Root idea is to consider the axial plane as our xy-plane and coronal or sagittal as our z axis while defining our splitting surface. Basically, the user marks a couple of convenient points while roaming over axial plane, which represents xy-plane of the coordinated system, and changes z value by using coronal or sagittal planes representing z axis of the volume. The user repeats the marking of the points for other slices too. Consequently, we have three groups of points that align in the same *xy*-plane respectively. The algorithm is constructing the surface by benefiting from the position of those three groups of points. The other critical point is not to try to calculate the formula of the curves in 3D space, but follow the strategy of calculating curves in 2D with respect to *xy*-plane and *yz*-plane separately in order to find positions of points in 3D space. The main steps of the surface construction algorithm are described in detail as follows:

1. The user selects an arbitrary slice and defines a parabola for this specific slice by marking three or four points that are selected logically to separate the mitochondria from each other in axial view of the constructed volume of input dataset. The algorithm takes these points and resolves them with the approach of least squares to find the equation of the best fitting parabola on these points.
2. The user changes the slice rapidly by benefiting from sagittal and coronal views of the constructed volume. The user then repeats the action of marking points two times more in other slices.
3. In the end of the user interaction, the application has three separate parabolas lying on the *xy*-plane of different values of z-axis.
4. It is then time to change the orientation of the calculations. After finding equations of parabolas that are lying on the *xy*-plane, it is time to use those equations to fill points that lie on the parabola. Other mathematical operations are carried out through the *yz*-plane with the following strategy.
5. We increase the value of x and find every point that is on the parabola using the corresponding formula. As a result, we have three points that are lying on the *yz*-plane. This process is continued till all the points on the parabola are filled
6. At this point, there is a process of filling the points between and around these parabolas. By increasing the value of x, we have three points that lie on the *yz*-plane. It means that we find the best parabolic equation that passes through those points solving the linear system equation created by these points.

7. After finding the parabolic equation that lies on the *yz*-plane, we fill the blank slices of this sector by only using the parabolic equation. We repeat this process until the surface is filled completely.
8. In the end, there is no formula for a quadratic surface but we have a point cloud that is defining quadratic surface which is enough to contribute to the 2,5 D automatic segmentation algorithm that works by taking base slices.

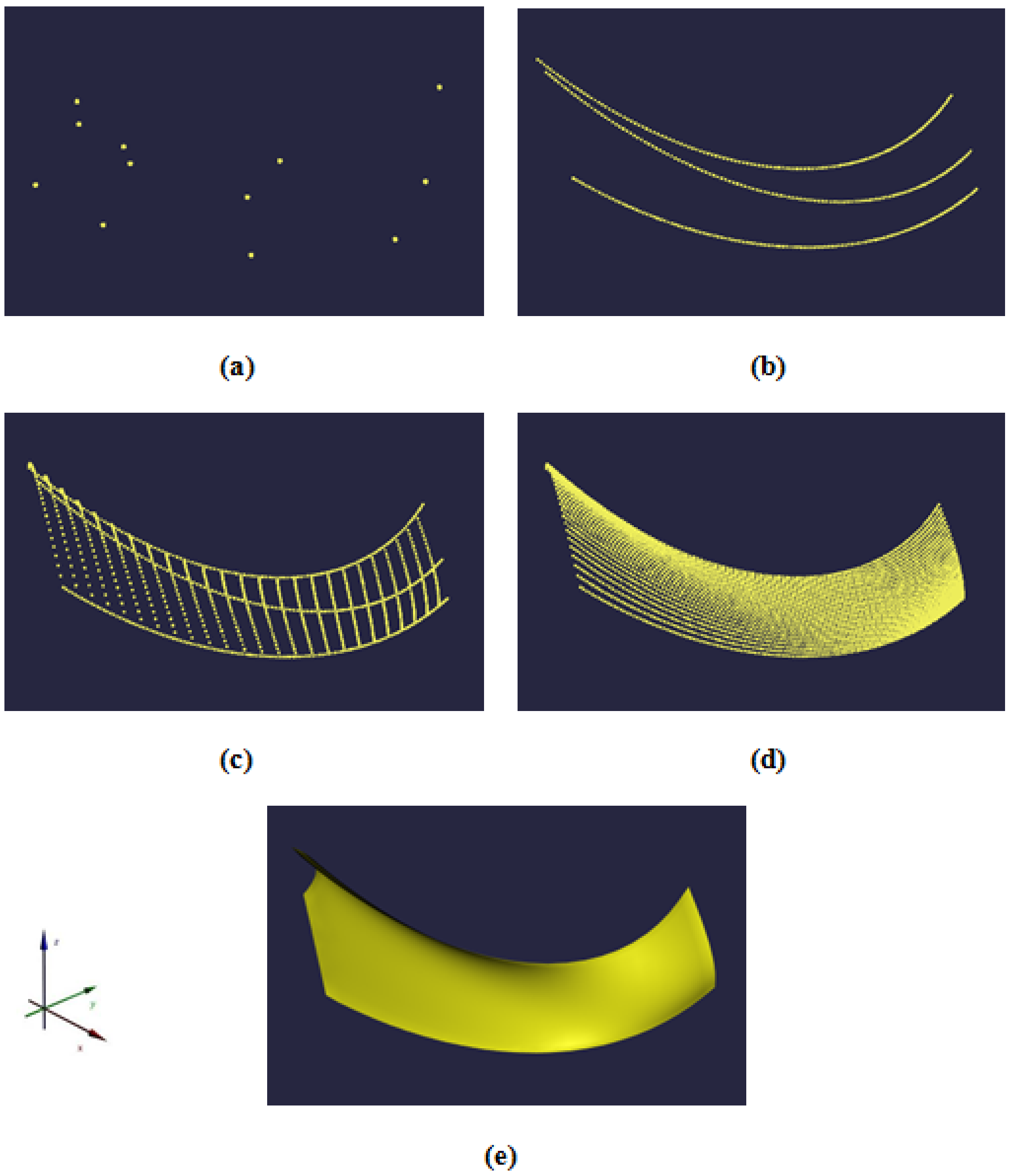


Figure 6.6 Major Steps of Surface Construction Algorithm
(a) User marked points through in 3D space. Each group of four points are lying on same xy-plane. (b) Points on parabola passing through these points. (c) Points on parabola lying on yz plane. (d) All points are lying on yz plane. Point cloud defines the splitting surface. (e) Constructed splitting surface

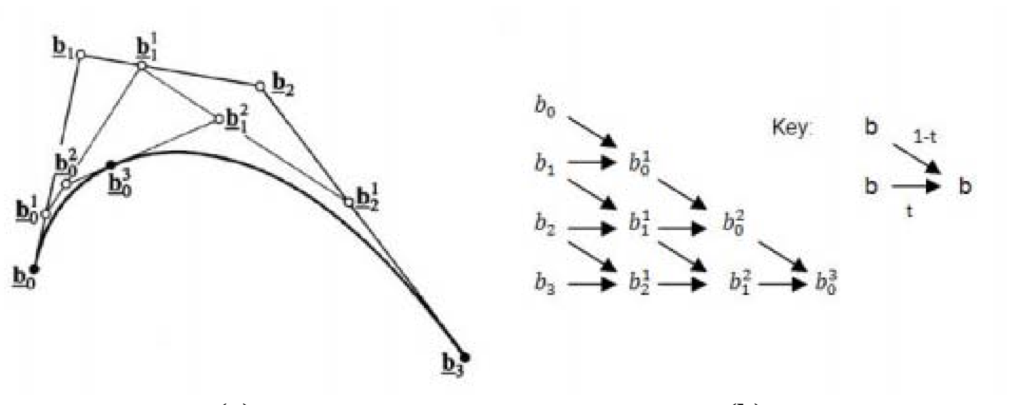
Figure 6.6 summarizes the main steps of the spitting surface construction method explained above. Our strategy is not defining boundary points in 3D space arbitrarily. The user defines points in different slices. Suggested ones are the beginning, the middle and the end of the mitochondria to reach more generic solutions. However, there is no restriction to define points in consecutive slices. The user needs to be more careful not to get illogical surfaces for distant neighbors. By restricting users with three different slices, another advantage is to finish description of a surface by reducing transitions between slices. Otherwise the user loses too much time while passing the other slices randomly. In addition to that, it gives a chance to see a much better skeleton of the splitting surface compared to randomly distributed points since it has its own order.

6.1.2. Initial Separation of Mitochondria with Constructed Splitting Surface

Before beginning the construction of splitting surface, the user should select the desired mitochondria. Harmony between splitting surface and selected mitochondria is within the responsibility of the user, which makes the algorithm user-guided. After this process, the mitochondrion is split into two by the following steps.

1. Quadratic surface corresponds a parabola in each slice and mitochondria corresponds a snake boundary in each slice of dataset. The first step of the algorithm is finding the intersection points of the parabola with the snake. Logically, the parabola should pass through the snake and there should be two intersection points. Actually, we are not using the parabolic function to find the intersection points but we are trying to find intersection points polygon, which defines the snake with the bordered lines corresponding the parabola. Therefore, it becomes the problem of finding intersection points of two groups of bordered lines. One group belongs to the snake and the other to the parabola.

2. After finding the intersection points, the snake is split into two parts with the border of the parabola passing through the snake. However, there is no such adherent border between the snakes. By using the nature of their shape of fold, we rearrange the border of these mitochondria. Between two convenient points on the snake and two intersection points, Bezier curves are constructed to give the shape of fold to the borders by using the DeCasteljau Algorithm which is a recursive procedure to reclaim parabola in the form of Bernstein ones [56]. Mechanism of the algorithm is given in Figure 6.7.



(a) **(b)**

Figure 6.7 De Casteljau Algorithm for Constructing Bezier Curves[57]
(a) Representative visualization of De Casteljau algorithm (b) Pseudocode Representation.

3. These steps are repeated for every slice of mitochondria separately. These steps construct an initial separation of the two mitochondria. It will give necessary inputs needed by the adapted automatic segmentation algorithm. Actually, the result of this step is very acceptable before passing to the automatic segmentation part. That is why we keep this mode in the application separately.

Figure 6.8 shows how well-shaped two snakes are obtained from one snake by splitting them with a user-defined surface. While it is possible to split snake by using splitting surface as border, it is not a good-looking solution for visualization. Snake on related slice is divided into two parts with respect to parabola of splitting surface in that slice. Constant portion of splitted parts closer to splitting surface is deleted and Bezier Curves are used to construct the border of those snakes instead of splitting surface itself as mentioned before.

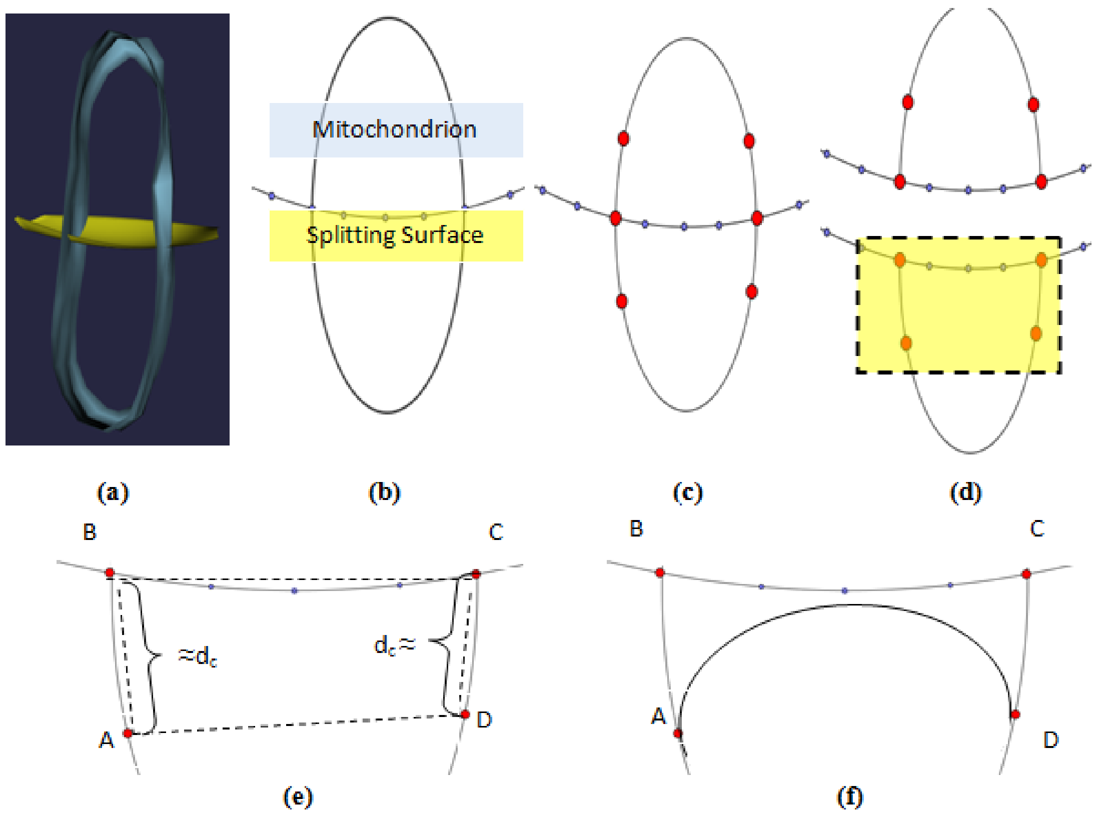


Figure 6.8 Splitting of Snake from Intersection Points
(a) 3D representation of mitochondrion with splitting surface (b) Illustration of mitochondrion and splitting surface for one slice is shown. Snake is used to define one slice of a mitochondrion. (c) Intersection points of snake with splitting surface are marked. Moreover, specific points on the snake, which are necessary for separation, are indicated (d) Focal point where the operation taken place is highlighted. (e) Details of the highlighted area is shown. *B* and *C* points are intersection points of splitting surface with snake. *A* and *D* are points that are far away from *B* and *C* with a constant distance. This constant distance (d_c) is found by specific ratio (8/10) of the remaining part of the snake after splitting surface divides the snake into two shapes as lower and upper part. (f) *A* and *D* points are closed by Bezier Curve based on a De Casteljau algorithm shown in Figure 6.7. *A,B,C* and *D* are b_0, b_1, b_2 and b_3 in Figure 6.7.Order of *A,B,C* and *D* are followed to construct Bezier Curve. Same operation is done for the upper part of snake whose details are not given in this figure. However, it is a similar operation done for lower part of snake.

Summary of the splitting operation is given Figure 6.9 from user perspective. Firstly, user selects the desired mitochondria, then defines splitting surface where splitting takes place and finishing the operation with confirmation after seeing output of algorithm.

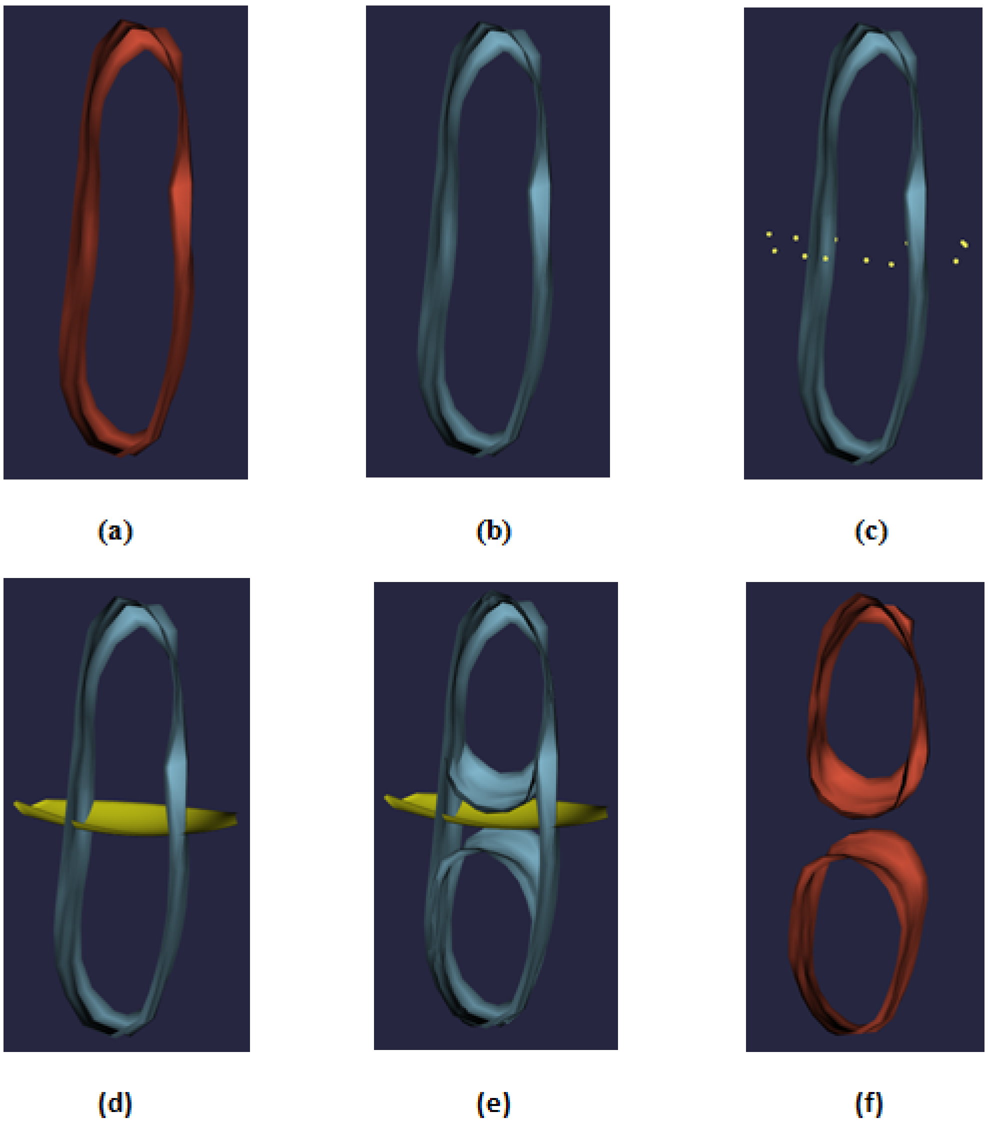


Figure 6.9 Steps of Splitting Operation
(a) Initial state of mitochondrion (b) Mitochondrion is selected by user for split purpose (c) User defines the points for splitting surface (d) Generation of splitting surface mentioned previous section in Figure 6.6 (e) Initial splitting of mitochondria with respect to defined splitting surface. Both input mitochondrion and outputs of splitting operation is shown together. (f) Confirmation of user for related splitting operation. Resulting two mitochondria are displayed.

Actually, this initial separation phase of the mitochondria is the available mode of application, since it produces acceptable split operations without the help of the automatic segmentation algorithm in prompt manner. Since there is no guarantee of automatic segmentation algorithm for finding a valid snake in every case, it is an inevitable necessity to keep that kind of mode in the application for the benefit of the user. Other than this, the initial splitting phase produces input of the automatic segmentation phase which will be explained in the next section in detail.

6.1.3. Adaptation of Automatic Segmentation Algorithm for Splitting

The automatic segmentation algorithm needs initial points to be able to start finding valid snakes in each slice. The previous phase gives the algorithm what it requires. This demand is basically two initial snakes, which are ready to grow. In addition to this, split surface is coming from the previous phase that forms a barrier between two initial snakes, which is very convenient for splitting. Inputs of the adaptation phase of the algorithm are shown in Figure 6.10.

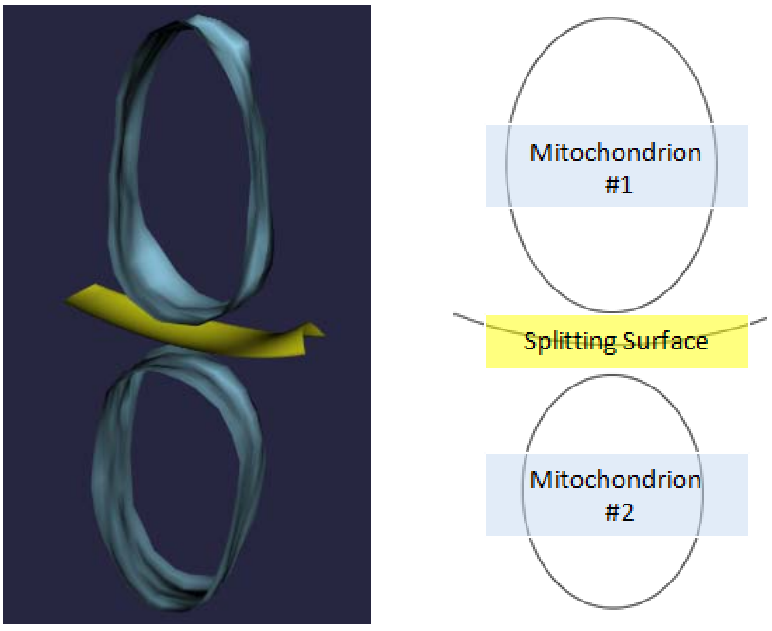


Figure 6.10 Inputs for Adaptation Phase of Splitting Method
(a) 3D Representation of Initially Splitted Mitochondria and Splitting Surface (b) Illustration of Initially Splitted Mitochondria and Splitting Surface for One Slice.

These initial snakes could be a little big for the purposes of starting the snake algorithm since it could have already exceeded the boundary of the real mitochondria slice. Therefore, the current snake is downsized with a constant value to keep its initial position inside of real borders of mitochondria. After this preparation process, it is time for the activation of the adapted version of the the automatic segmentation algorithm with specified inputs that are initial snakes and splitting surface. The energy formula of the algorithm is Equation 24 given below.

$$E_{snake}(v(t)) = E_{int}(v(t)) + E_{ext}(v(t)) + E_{inf}(v(t)) + E_{surf}(v(t)) \quad (24)$$

There are familiar forces from the classical 2,5 automatic segmentation algorithm that are external forces whose source is coming from curvature structures of mitochondria slices, internal force whose power is coming from tension and curvature structures of boundary points of snake during growth and inflation force that is responsible from exceeded weak curve structures, which is not the boundary of real mitochondria [22]. Additional term is surface energy whose resource is mainly the interaction of snake's boundary with this surface and its own existence. The aim of the additional term is to create obstacles across boundary points that are close to the splitting surface during snake growth. There are various ways for creating this force. We have implemented two alternatives and came up with a resulting strategy that should be applied in the end. Detailed explanations of these alternatives are given below.

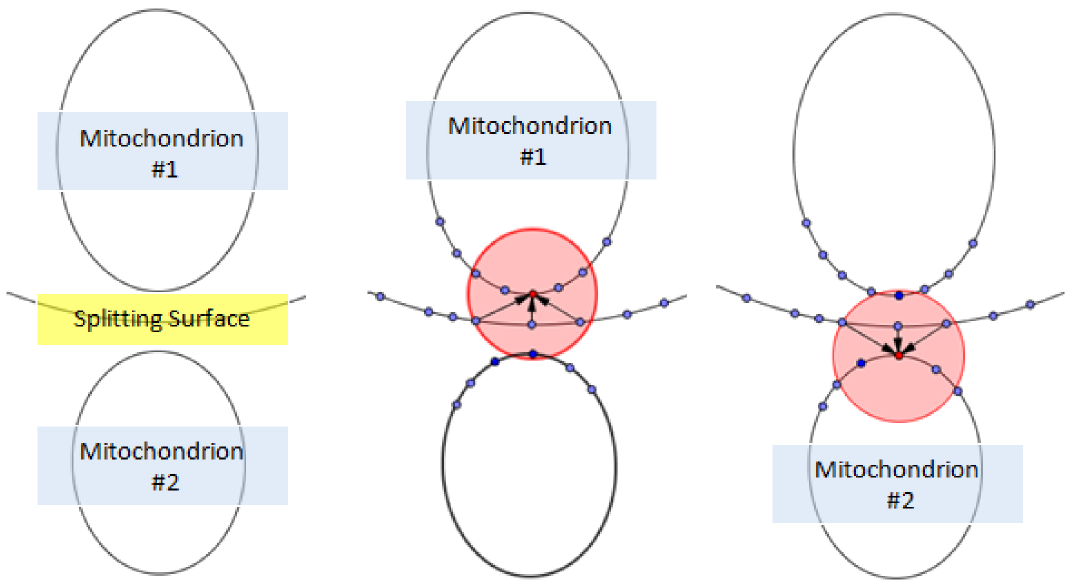
6.1.3.1. Distance Based Approach

The main idea is to calculate surface force by focusing the distance between the snake and the surface points. While snake boundaries are far away from the splitting surface, it does not have any effect on total energy and snake grows as before. When some snake points are becoming closer to the surface points, they are affected from the surface more. The distance of each vertex to points of splitting surface is calculated and this distance value becomes as an input of the surface energy as shown in Equation 25 and 26.

$$d_s = \sum_{i=0}^{n} \sqrt{|(x_i - x_s)(y_i - y_s)|} \quad (25)$$

$$E_{surf}(v(t)) = \frac{1}{1 + d_s^2} \quad (26)$$

Taking into account all the surface points is costly for the calculation and there is no point in calculating the effects of far points of surface over a snake vertex. Therefore, a fixed size area is specified around a vertex and the effect of surface points that are inside this area is measured to contribute the surface energy. If there are no surface points inside this area, there is no surface energy. Moreover, the direction of the effect is the vector between the snake vertex and the surface point for a single distance value.



(a) (b) (c)

Figure 6.11 Distance Based Approach for Surface Energy
(a) Initial State of the Execution of Automatic Segmentation Algorithm with a Splitting Surface and Two Mitochondria.(b) Sample Force Applied to a Point on Mitochondrion #1. (c) Sample Force Applied to a Point on Mitochondrion #2.

To benefit from the 3D shape of the splitting surface, neighbors of the slice also give a limited contribution to the surface energy. Since there is no possibility of changing the position of the vertex through the z axis due to the implementation of the automatic segmentation algorithm, equal number of above and below neighbors enter the algorithm to destroy the effect of them in the z-axis. Since distances between slices are constant, input of these neighbors into the algorithm resets the effect of each other. The effect of the neighboring slices is given in Figure 6.12 visually.

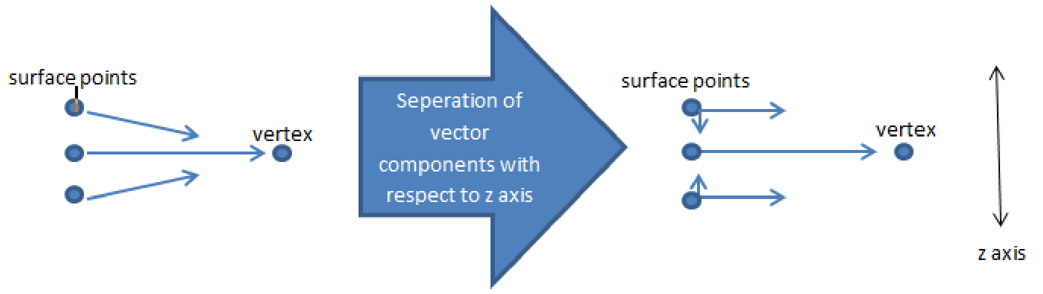
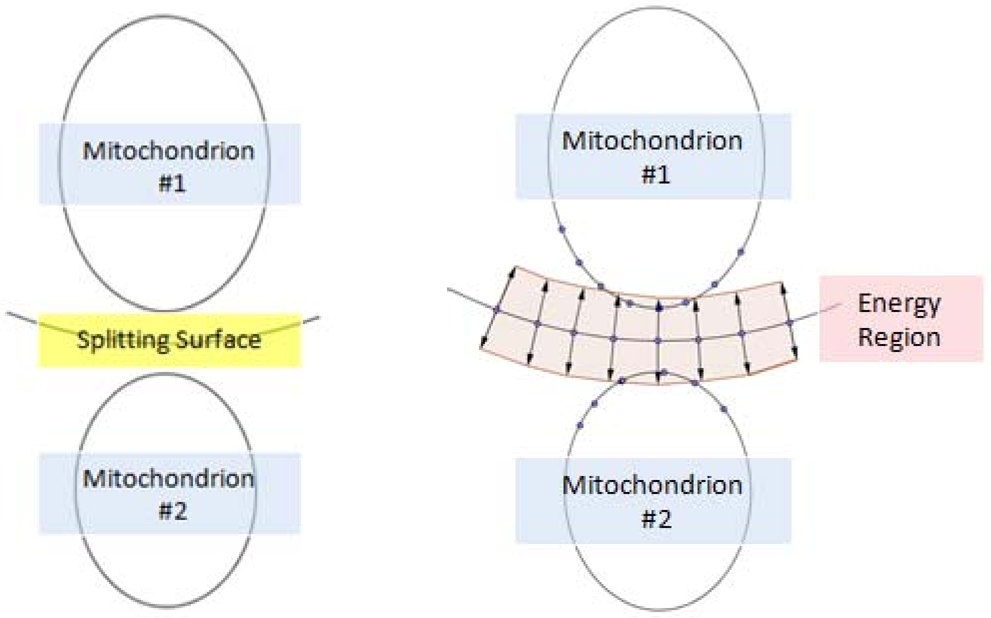


Figure 6.12 Contributions of Neighbor Slices to Surface Energy Term

While this approach benefits from the shape of the splitting surface as much as possible to get the best converged snake since the distance between the snake points and the surface points are changing in each iteration of snake growth, it takes $O(n^2)$ to calculate the energy on the surface for a snake located on each slice. When it comes to all surface points and all slices of mitochondria, $O(n^4)$is the order of calculation that leads to very long system response times which is not acceptable for the user. It takes thousands iterations for a mitochondria to converge a valid state, and the calculation of surface energy with this approach in each iteration causes the algorithm to be unable to give a finishing time. Therefore, this approach is excluded from the final application due to its time inefficiency and gives valuable feedback for the next approach.

6.1.3.2. Energy Region Based Approach

It is learnt that calculation of the surface energy at each iteration of snake growth is concluded with performance absence from the previous experience. Therefore, the next strategy should be to create energy once and contribute the snake growth with that fixed energy. Therefore, we decide to create a fixed energy field around the splitting surface area that prevents the progress of snake point position that enters the effect of the area. The surface corresponds a parabola in each slice. The main idea of the energy field is illustrated in Figure 6.13.



(a) (b)

Figure 6.13 Energy Region Based Approach for Surface Region

(a) Initial State of the Execution with a Splitting Surface and Two Mitochondria. (b) Two Mitochondria Under Force of Splitting Surface. Firstly, adapted snake algorithm is executed for Mitochondrion #1, then same algorithm is executed for Mitochondrion #2.

Vertex that enters the area of surface energy is affected from there with a constant value of energy with a direction depending on the current location of the vertex. In order to create the energy region, the below steps are followed.

1. Draw a line from the first point of the parabola to the last point.
2. Find the direction of the vector which is perpendicular to the line found in the previous step.
3. Copy points to fixed distance away towards the vector found in previous step.
4. All quadrupled points are adjacent and their copies form a polygon. Fixed amount of energy is put inside this polygon by finding the pixels within the polygon with the method of Scanline Fill Algorithm [58]. In this algorithm, the entire area of the polygon is scanned. Scanlines are intersected with polygon edges and pairs of intersections are filled with energy field. Figure 6.14 visualizes the principle of algorithms with a polygon and a line scanning this polygon entirely.

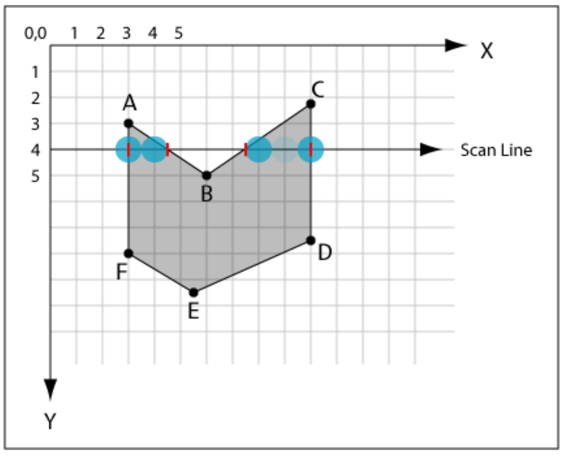


Figure 6.14 Scanline Fill Algorithm[59]

5. We use the vector which is perpendicular to the line formed between adjacent points to determine the fixed x and y components of the energy vector for each polygon area.

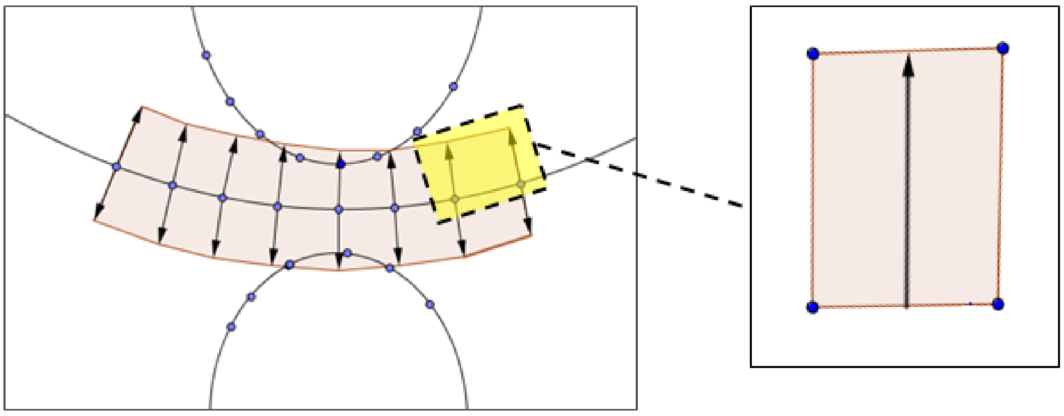

(a) **(b)**

Figure 6.15 Creation of Surface Energy Region
(a) Splitting Surface with Energy Region For One Slice. (b) Closer Look to Each Sub Region Forming Energy Region

Figure 6.15 shows how energy region is formed by its sub regions. Each sub region is a simple polygon and they are filled with Scanline Fill Algorithm as indicated in Figure 6.14 with constant amount of energy enough to stop growth of an approaching snake.

6.2. Merging

As a result of the 2,5D automatic detection algorithm, two mitochondria can be detected instead of one real mitochondrion because of the cristae structures in mitochondrion, which are producing strong curves. This leads a strong external force for the segmentation algorithm and balloon snake cannot pass these structures because of significant external force. With the help of the user observation and guidance to the algorithm, our aim is to merge these two mitochondria into one. There are two implemented methods to achieve this goal. While one of them is merging mitochondria without help of the 2,5 D automatic algorithm, the other one is making a contribution to the model and help the standard automatic detection.

6.2.1. Initial Merger of Mitochondria

The user should select the right mitochondria for the purpose of merger. There is no internal check for the selected mitochondria convenient to merge. After selecting two mitochondria, the user can activate the process of initial merger, which is a mode available in the application. Steps of the algorithm are shown for a slice of mitochondria and they are repeated for each slice separately. The steps of that algorithm are shown below:

1. Center of mass of two mitochondria is determined and they are connected with a line as shown in Figure 6.16. This line helps to find at which point these mitochondria should join the other one.
2. Combination of two mitochondria is rotated till the line connecting the mass centers becomes parallel to x-axis assuming that their coordinates lie on the *xy*-plane.
3. Peek points are found for the above and below borders of snake as follows. The points that are above the connection line have maximum distance to the connection line are the peek points of the upper border. This distance is determined as the regular distance of a point to a line. That is the length of the fictitious perpendicular line to the connecting line, which comes from the specified point. Similarly, the points that are below the connection line are the peek points of the below border.
4. While upper peek points are connected with each other, lower peek points are also connected with each other with sample points on those lines.
5. Points that are on the snake and between these peek points are deleted and the final merged shape of the mitochondria is obtained.

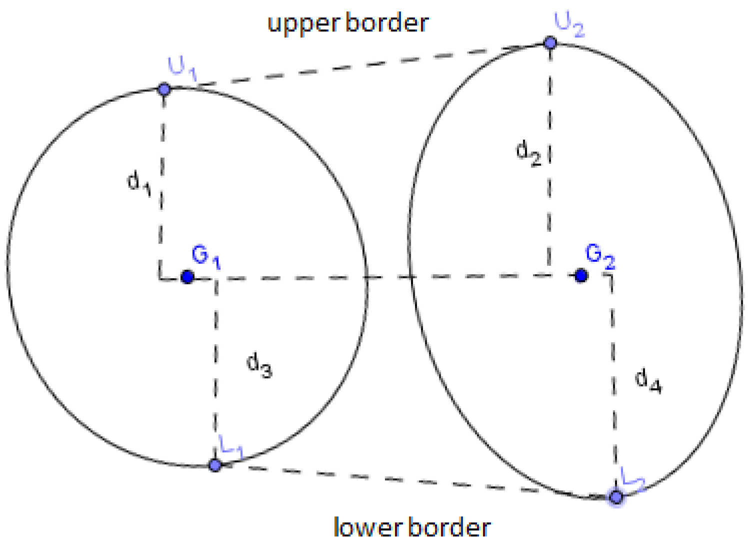
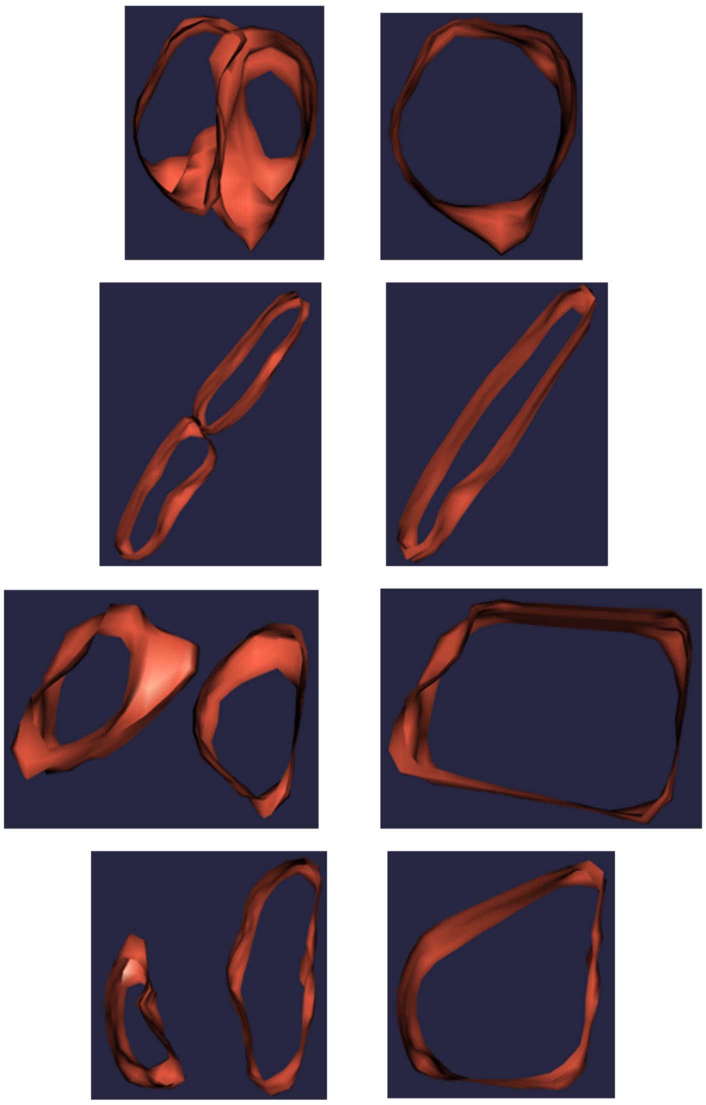


Figure 6.16 Details of Initial Merging Phase
There is a line between G_1 and G_2 which are the center of masses of these two snakes. U_1 and U_2 are the upper points that have the longest distance, d_1 and d_2, to the mentioned line on snake. L_1 and L_2 are the lower points that have the longest distance,d_3 and d_4, on mentioned snake. While U_1 and U_2 points construct the upper border, L_1 and L_2 points form the lower border to combine these two snakes.

Figure 6.16 shows the geometry that plays crucial role to define the strategy of combining every slice of two mitochondria. It seems that connecting peek points with a line form a very sharp structure on connection points. However, when all slices are connected this way and evaluated as 3D object, results are unexpectedly acceptable. Therefore, there is no further effort to obtain curved connection shape while connecting peek points. Obviously, it does not mean that there will be no progress if the algorithm follows this way.

The final shape that is handled as a result of initial merging phase can be used as the final product of merging process of mitochondria. Moreover, it is used as the input of the adapted version of the 2,5D automatic detection algorithm. Figure 6.17 shows couple of examples consist of mitochondria having various size and orientation for better understanding of merging operation. The details of how these initial merged mitochondria used will be given in the next section.



(a) (b)

Figure 6.17 Random Scenarios for Merging Operation
(a) Initial State of Randomly Chosen Mitochondria (b) State after Application of Merging Operation.

6.2.2. Adaptation of Automatic Segmentation Algorithm for Merging

Initial points for the start of automation are critical information that should be predetermined before the execution of the algorithm. The previous phase supplies an initially merged snake that shows the initial growth area to the automatic algorithm. This initial snake probably passes over the boundary of a real mitochondria slice. If the 2,5D automatic algorithm is started with this snake directly, there is the possibility of that balloon snake going away to irrelevant places that are far away from the real mitochondria. Therefore, the current snake is shrunk with a constant value to hold the location of the initial snake inside true borders of the mitochondria.

After the preparation process, adapted version of the automatic segmentation algorithm with specified input that is the initial snake is activated for the trial of finding a valid mitochondrion. Energy formula of the algorithm which is not different from the automatic version is shown in Equation 1.

All of the forces are well known from the standard 2,5 automatic segmentation algorithm, that is the external force whose responsibility is to expand the snake if certain criteria meets, internal force that is responsible for the stop growth of the balloon snake and the inflation force responsible for passing over cristae structures while the snake is growing [22]. There is no additional term like in the adapted version of the automatic algorithm, which is used for the splitting operation. Instead of this approach, we interfere the existing external force indirectly by changing the resources that are creating the related force and this process is repeated for all the slices of the mitochondria separately.

The source that is defining the external force is directly coming from the detected curve structures of the automatic detection algorithm on mitochondrial images. Ideally, those curve structures should be found mostly membranes of mitochondria's boundary that is showing characteristic of curvature shape when it is taken apart. Therefore, it is rational to use this property of membranes while detecting them. However, shapes of other internal structures inside cells are also similar to curve form. Unrelated ones are counted as weak and they are eliminated from final results, but some strong cristae structures can lead to detection of curves on those places. When such detections are too common inside mitochondria, it can divide the mitochondria into multiple parts and this can cause the detection of more than one mitochondrion during standard automatic algorithm execution. There is no indication to differentiate these cristae structures from boundary of mitochondria for the automatic algorithm. At this point, initially merged snake allows the algorithm to eliminate those unwanted curve structures. The steps of the algorithm are as follows:

1. Detected curves of the automatic detection algorithm are considered in a closed form and the center of mass of a curve is calculated approximately in the following way. Start, end and top points of the curve form a triangle and center of mass of this triangle is considered to be the mass center of the related curve. This procedure is shown in Figure 6.18 below.

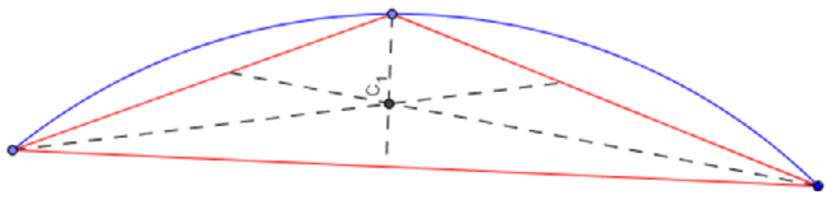


Figure 6.18 Approaching Center of Mass of Closed Curve

2. After finding the points that are the center of masses of curves, we determine whether these points are inside or outside the initially merged and shrunk snake in the corresponding slices by using the Ray Casting Algorithm [60]. A ray is spread from the center point towards around and points on snake boundaries are considered as the polygon that ray should hit. Based on the number of hits, whether the curve is inside or outside of the snake is understood.
3. Curves that are located inside the initial snake according to the above algorithm are excluded from the set of curves.
4. Calculation of the external force is done with the rest of curves for each slice.
5. With an adapted external force, execution of the automatic detection algorithm is started.

Adaptation of automatic segmentation algorithm for merging purpose is given visually in Figure 6.19.

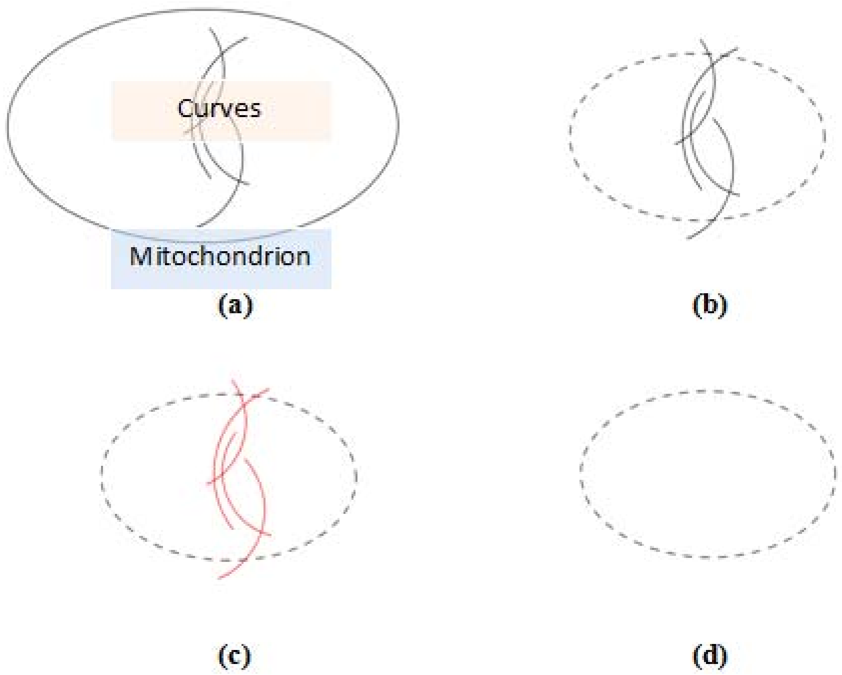


Figure 6.19 Illustration for Adaptation of 2,5D Automatic Segmentation Algorithm during Merging Operation
(a) Initial State of Snake (b) Snake is shrunk with specific ratio (c) Curves whose centers of mass (Figure 6.18) inside shrunk snake are found. (d) Curves that are inside of shrunk snake are deleted. Automatic segmentation algorithm is initiated with this snake without curves in it.

Figure 6.20 shows an example of merging process of automatic segmentation algorithm with TEM images. Cristae of related mitochondrion produce strong curves inside of this organelle. These wrongly-detected curves causes decrease on external energy of snake growth formula. These unnecessary curves are deleted with the help of initially merged snake as mentioned in steps pf merging operation.

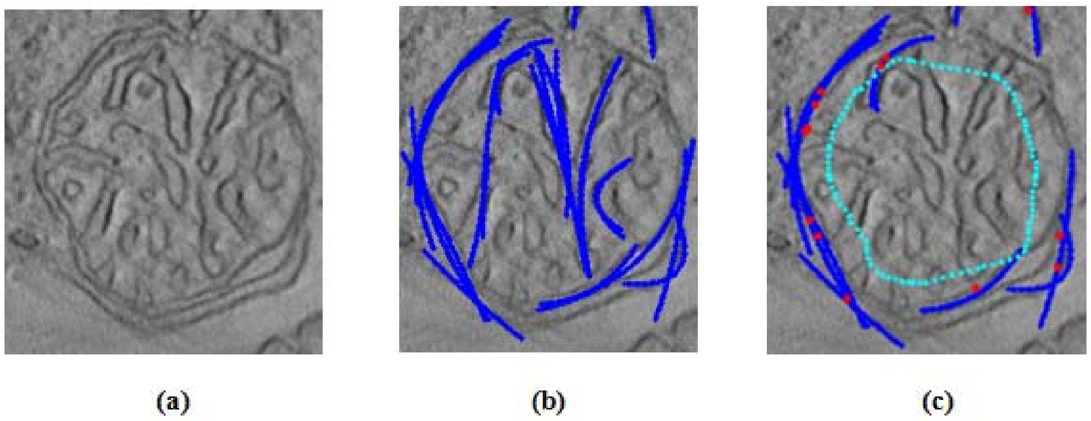


Figure 6.20 Example for Adaptation of 2,5D Automatic Segmentation Algorithm during Merging Operation
(a) Mitochondrion Image (b) Detected curves of the automatic detection algorithm are marked with blue color. (c) Shrunk snake is shown with dots. Curves inside of this snake are deleted.

Initially merged snake gives the algorithm hints that there is no border inside this area. Therefore, there is no need to create a force inside this area that will stop the growth of balloon snake. By deleting unnecessary curves inside the snake area, we eliminate the force arising from there. The snake continues to grow by getting base of other curves outside the initially merged snake.

6.3. Deletion

Despite the success of detection performance of the automatic segmentation algorithm, it can return few valid results from unrelated areas. Tasel et al uses modified version of DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm [61] to find the initialization points of snakes in each slice [38]. These initialization points could come up to non-mitochondrial region and the automatic detection algorithm can detect mitochondrion that is not intersecting with any of the existing mitochondria because of strong curve structures that are spread to this non-mitochondrial region in a particular order by chance. Other than this, position of more than one existing mitochondria can create an area tucked between their borders, and this area could be perceived as a real mitochondrion since it is surrounded by strong curves generated from membranes of these mitochondria. Besides, if initial locations of the snakes come across to this place, such kinds of variations on location and detected curve structures mitochondria group can easily affect and mislead the snake balloon algorithm in the phase of growth. Therefore, there is a need to exclude this kind of wrongly-detected mitochondria from results by the evaluation of the user. The application serves a reject tool allowing the user to remove wrongly detected mitochondria from the automatic segmentation results. In rejection mode, the user simply selects the mitochondrion that is to be deleted and takes the action of rejection. After this process, the selected mitochondrion is removed from the list of valid ones. The user gets rid of the unnecessary crowd of such structures easily. Steps of the deletion process are given in Figure 6.21 visually.

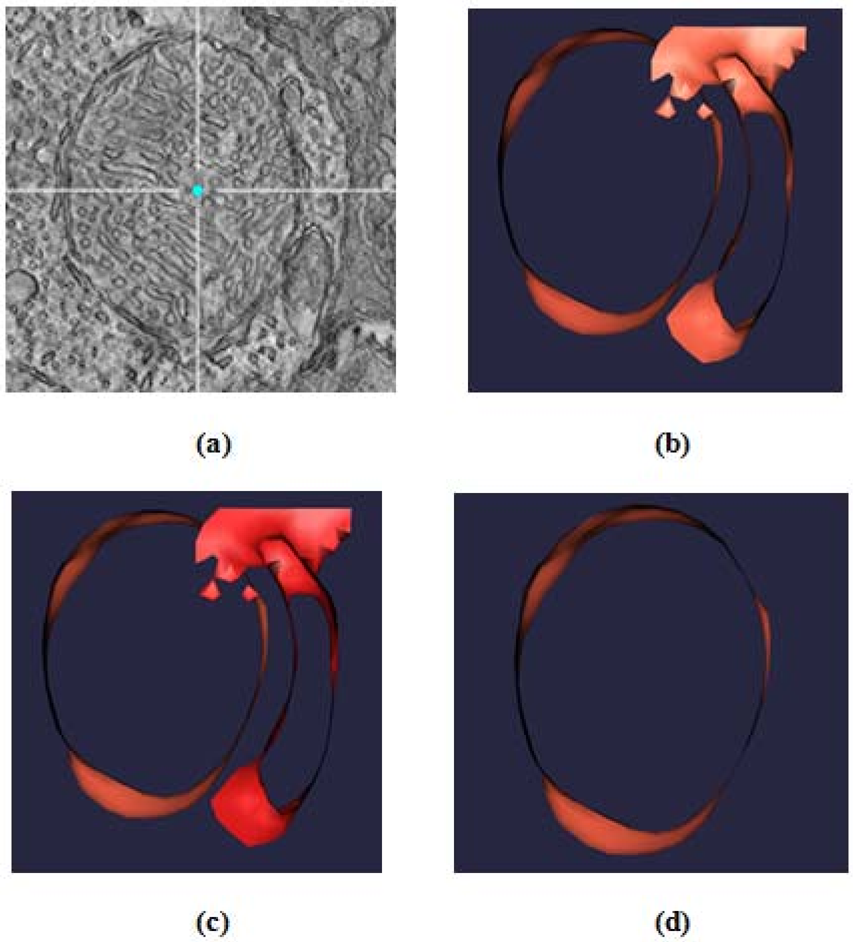


Figure 6.21 Steps of the Deletion Method
(a) There is one mitochondrion in TEM image. (b) Automatic segmentation algorithm is found one wrong mitochondrion (on the right hand side) due to the cell boundary located there. (c) User selects the wrongly detected mitochondrion from user interface. (d) Wrongly detected mitochondrion is deleted from screen. It is not on the list of valid one anymore.

6.4. Manual Drawing

There are situations that none of automatic method or the other semi-automatic methods are successful to detect these because of the absence of detected curves, due to image noise and or other structures. In such cases, the user needs to extract approximate shape of a mitochondrion with a little effort as much as possible not to leave undetected mitochondria behind until a method is proposed for such cases in future work. Steps of the manual segmentation are conducted as follows:

1. The user selects a slice by using coronal or sagittal view of the application and activates the procedure of manual segmentation. After this point, movements of the mouse are recorded as the border of the mitochondria. The user needs to start the movement, continue on the border of the mitochondria and end it in a logical place that passes through the starting point. There is no control over the manually entered points. They are interpreted as the border of the snake in the corresponding slice.
2. The user needs to repeat the above step two times for different slices. As a result, we have three borders of snake lying on the *xy*-plane that are located in separate slices.
3. To provide the consistency between slices, points on the initial snake are completed to a constant value that is equal for each three slices.
4. We want to match three points on different manually entered snakes in different slices. In order to do that, center of mass of a snake is calculated. After that, angle of the line drawn between the center of mass and the point on the snake is calculated with respect to x-axis.
5. Same mentioned angle is calculated for other points on the snake, and they are matched with the points on different slices by comparing this angle value with the value of snake points that are located in other slices. Three points whose angle values are closest to each other are matched and to form a polynomial using the least squares method. Figure 6.22 shows the idea of finding this polynomial visually.

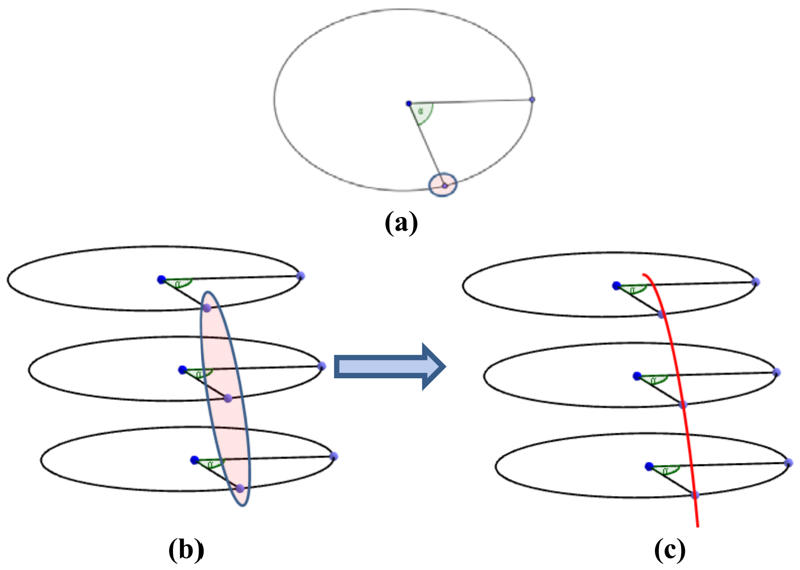


Figure 6.22 Filling Empty Slices by Manual Segmentation Method
Slice of a mitochondrion is shown as elliptical shape for the purpose of illustration. There is no such prerequisite. Angle of each point is related with the position of the point on snake with respect to center of mass of the mitochondrion. (a) Angle of each point with respect to x axis is calculated (b Three points whose angle values are closer to each other are matched (c) A parabola is formed by using these three matching points.

6. At this point, a similar approach that is applied in splitting surface construction is used to fill points around and between initially entered slices. 3D space is evaluated as x axis and *yz*-plane to find the points through z axis. Each polynomial equation is used to fill the points of all slices. At the end, we have the points for all the slices for the desired mitochondria.

All steps of manual drawing is shown in Figure 6.23. It begins with the definition of user points and end with constructed mitochondrial structure.

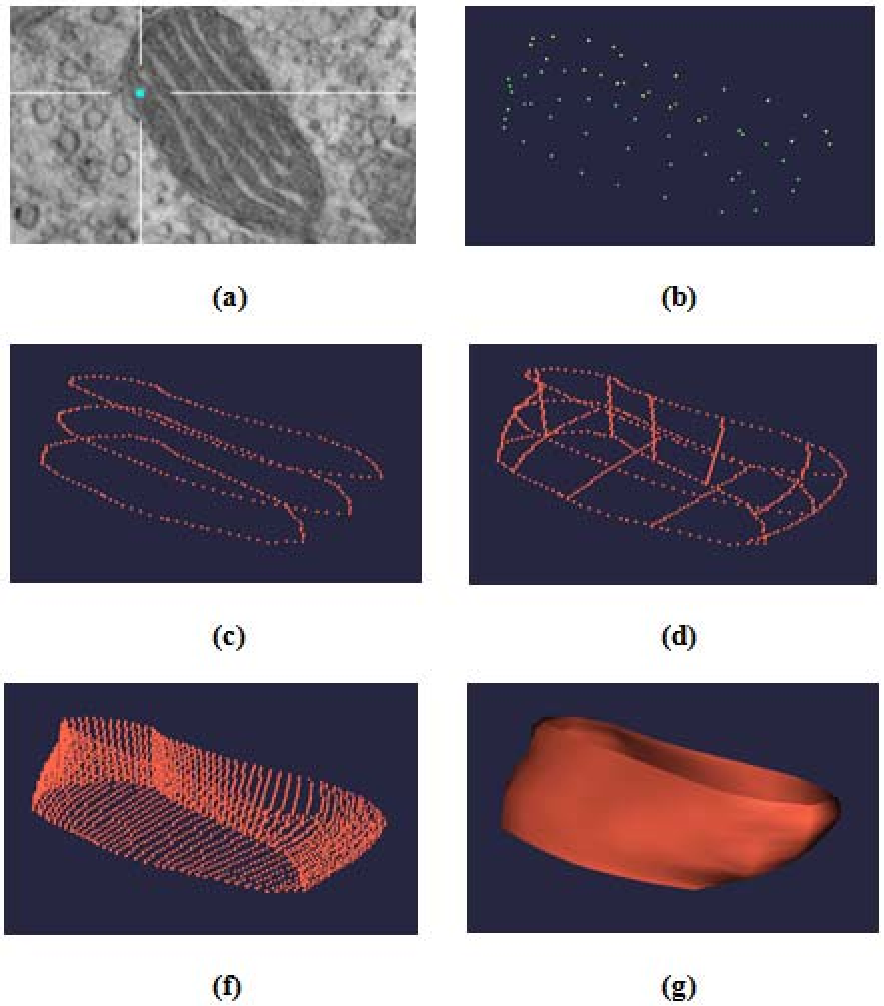


Figure 6.23 Steps of Manual Drawing
(a) User moves the mouse on a mitochondrion desired to be drawn (b) Points defined in three different slices are shown. (c) Complete the points to specific number. It provides necessary number of data for construction. (d) Couple of point on parabola found by taking matching three points on yz- plane into account. (e) All points are filled. Point cloud of mitochondria is defined. (f) Construction of surface from point cloud is finished.

6.5. Selection of Low Scored Mitochondria

The automatic detection algorithm detects almost hundreds of mitochondria for each dataset as a result of execution, but most of them fail to pass the shape validation phase that evaluates the quality of the detected structures. The shape validation utilizes energy measurements and several descriptors belonging to the membrane features in order to separate the detected shapes from the ones located in non-mitochondrial regions [22]. If the extracted shape satisfies all of the necessities, it enters the category of successful detection of mitochondrion; otherwise, it is rejected [22]. Threshold value that is optimized for all data sets causes to miss some of the real ones. Moreover, while snakes are running in every slice of the datasets, invalid growth in a couple of slices could lead to making those mitochondria invalid and rejected. Luckily, output generated from the automatic detection algorithm includes those rejected snakes. However, it is not a useful feature to show all of them to the user because of the complexity they would create in the presentation. If hundreds of rejected snakes are displayed for the service of the user, it would not be useful or understandable. Under the light of this information, we can display the minimum set of rejected mitochondria in the place that the user wants to see. This is done by the following way:

1. The user selects a slice by using coronal or sagittal view to determine the slice he/she uses to make observation.
2. The user selects a point in axial view, and the algorithm shows the rejected mitochondria if their 2D snake on that slice includes that point. This is done by using Ray Casting Algorithm [60] like in the previous sections.
3. The user can then choose a mitochondrion and make it a valid one through evaluation.

The steps of selection of low scored mitochondria are shown in Figure 6.24 as well.

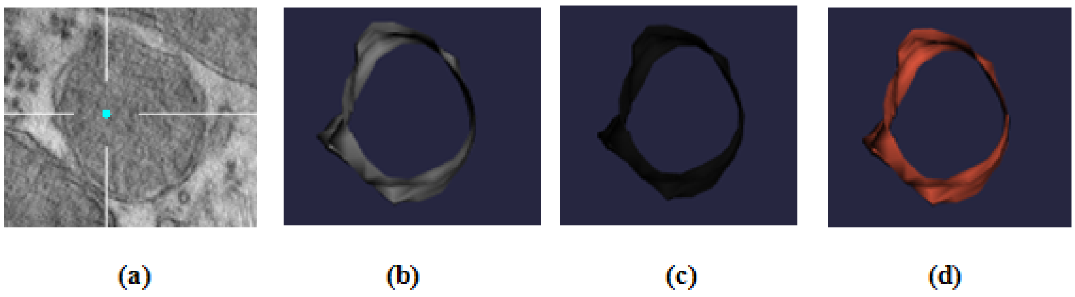


Figure 6.24 Steps of Selection of Low Scored Mitochondria
(a) Mitochondrion is selected in axial view of the application. (b) Algorithm searches invalid snakes containing selected point in related slice. Invalid snakes are shown to the user. (c) User selects the resulting mitochondrion is it is a valid snake visually. (d) Snake is confirmed and counted as a valid snake after user operation.

Selection of the invalid mitochondrion is implemented by following the rules of Ray Casting Algorithm the details of which are given in next section.

6.5.1. Adaptation of Ray Casting Algorithm for Selection

Selection of a snake is done by determining whether selection point is inside that snake or not. We can interpret snakes as polygons since they have hundreds of points with information of neighboring pixels of each one. If each of the two adjacent points are combined with a line and those lines come together, then a polygon is obtained at the end of this process. The definition of the problem then becomes whether a point is inside a polygon or not. Shimrat [60] introduced the generic algorithm to resolve this issue. Firstly, a ray spread from a point to the polygon and how many times that ray passes the polygon edges is counted. If the counted value is an odd number, then it means that point is somewhere inside the polygon. If the counted value is an even number, then it shows that point is somewhere outside the polygon. This simple and rather genius idea gives the solution to our problem definition. Figure 6.25 shows an example of the algorithm with an inner and an outer point with respect to a polygon. While the number of intersections of the ray coming from the outer point with the polygon edges is even, the number of intersections of the ray whose source is from the inner point is odd.

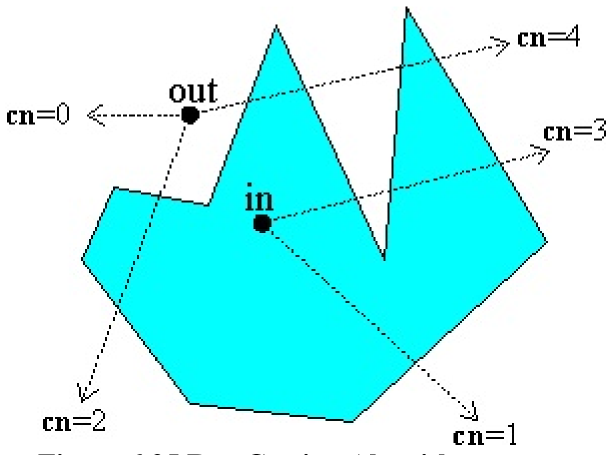


Figure 6.25 Ray Casting Algorithm

Ray spreading from point out intersects with the edges of polygon an even number of times. It shows that this point is outside of polygon. Ray spreading from point in intersects with the edges of polygon an odd number of times. It shows that this point is inside of polygon.

In the case of snakes point, it is possible to define and calculate the lines between adjacent points and apply related ray casting algorithm to reach the solution. However, calculation of equations for all lines between points is costly and unnecessary to reach the result. We adapted the ray casting algorithm in a more practical way without calculating line's equation and without dealing with all points on the snake. For the simplicity of the equation, the selection point is translated to the origin, and the points of the snakes are translated together Implementation of the algorithm searches for two neighboring points. While one should have y coordinate value greater than zero, the other one should have smaller than zero. The x axis value of the line between those points is calculated. If this value is positive, then it means that a ray spread from the origin to positive infinity on x axis hits the edge of the snake polygon. If one of the points is just on x axis, then we ignore that point and accept its adjacent point as the neighboring point of the first one.

Selection of snakes from visualized images is done by the described methods. This gives the user the perception of which snakes is active on rendering scene.

6.6. Initiation of Automatic Segmentation Algorithm by User Request

Automatic segmentation algorithm runs over the all images with a specified initial points and parameters to find all mitochondria in the electron microscopy images. There is a shape validator function that checks the validity of snake. If validity parameters are satisfied, then snake is counted as valid mitochondria, otherwise, it is rejected. Parameters of the shape validator function are shape energy, area, boundary continuity, curvature and signature. Shape energy is calculated in Equation 4. Shape is not accepted as valid one, if this energy doesn't meet necessary level of energy. Area parameter defined the area of the growing snake. If the area of the shape is too small, or too large for a cross section of a mitochondrion, then it is rejected, also. Boundary continuity defines the parameter related with gaps in the border of snake. The maximum gap length, the total gap length, the gap ratio of the shape and the gap border ration of the shape are analyzed to check the reliability of shape. The gap ratio is ratio of total gap length to the perimeter of the shape. The gap border ratio is similar condition for snake on border. If discontinuity of these parameters becomes unacceptably high, the shape is not considered as valid snake anymore. Appearances of mitochondria are defined as blob-like shape. It means that curvature on the shape of snake should be low enough. Moreover, signature parameter checks the cross-sectional shape of mitochondria having circular structure or growing in two directions. If snake does not obey this parameter, then it is counted as invalid one, also [22]. Lastly, validity parameter is threshold score of the combination of all these shape validator parameters.

The algorithm does not have any clue regarding whether there is a mitochondrion on a specific place or there is not. If there was a hint about the existence of mitochondria in a specified place, it could help soften searching criteria of the mitochondria in that place. Since our application is under user control, the user can give that hint to the algorithm just by choosing the place for the start of the algorithm. By using this information, the algorithm decreases the validity threshold value to detect the mitochondria in a specified position. Other than the validity criteria, gap parameters could also soften while mitochondria are being searched on the edges of images. This parameter is specialized to find mitochondrial structures located on the edges of the images while the automatic detection algorithm is running. Gap means the openness around the balloon snake when it is growing. It is a check for whether there is a curve structure around the snake while it is expanding. If the snake starts on the edge, it is normal not to find any curve structures on the edges of the images. In such a scenario, the automatic algorithm realizes that snake is somewhere on the edge, and apply the gap parameters specialized for the edge. By providing user guidance to the automatic detection algorithm, gap parameters could be softened to help the algorithm find possible valuable structure there. The conditions of automatic segmentation algorithm and manual initiation for validity parameters are given in Table 6.1.

Table 6.1 The Conditions of Automatic Segmentation Algorithm and Manual Initiation for Validity Parameters

Validity Parameters	**The Condition of Auto**	**The Condition of Manual Initiation**
the maximum gap length	$x \leq 100$	$x \leq 500$
the total gap length	$x \leq 125$	$x \leq 600$
the gap ratio	$x \geq 0.4$	$x \geq 0.8$
the gap border ratio	$x \geq 0.3$	$x \geq 0.6$
the validity	$x \geq 0.69$	$x \geq 0.4$

Other than that, validation of mitochondria is done by checking more than ten conditions that are related from the area of detection to gap parameters. We ignore the invalidity of certain number of conditions when the snake is in validation phase. While it increases the chance of finding real mitochondria, false detections are also becoming much more probable. There should be a balance that will not disturb the user with false detections while softening these conditions. Figure 6.26 shows the steps of the manual initiation of automatic segmentation algorithm by user.

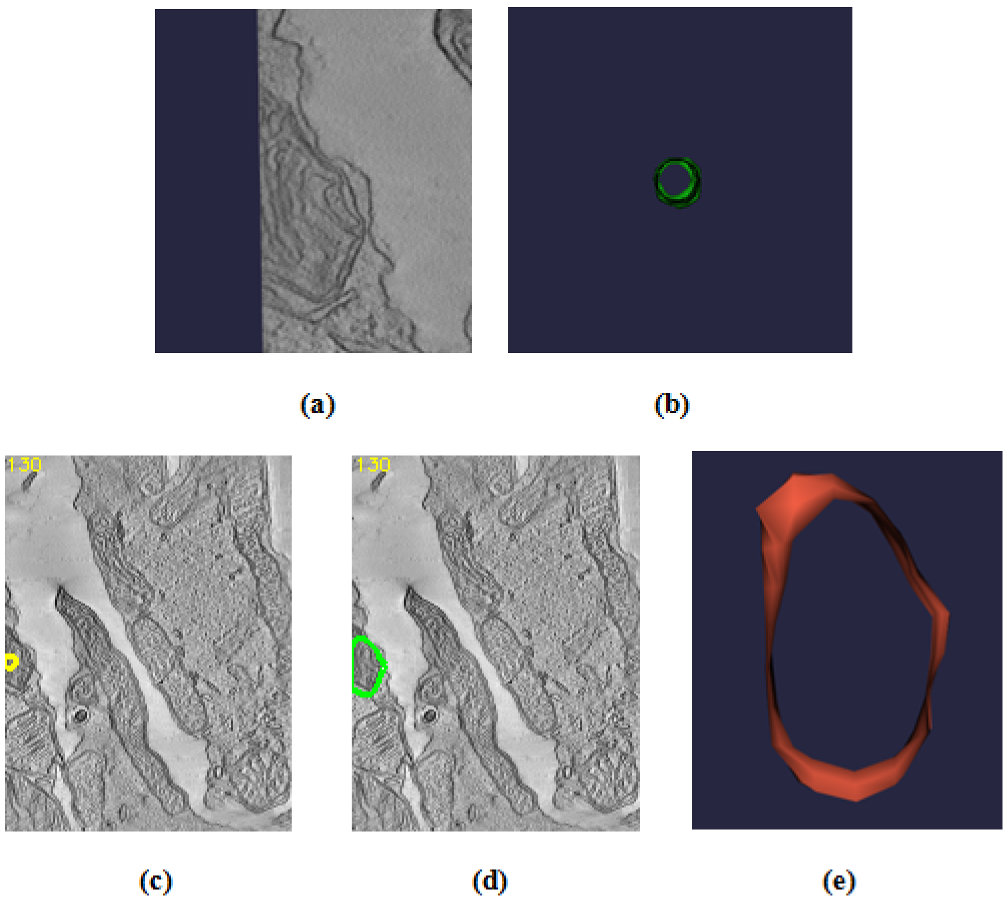


Figure 6.26 Steps of Manual Initiation of Automatic Segmentation Algorithm
(a) Mitochondrion is marked by using axial view of the application as described. (b) Initial snake at desired position is visualized (c) Automatic segmentation algorithm is initiated at intended position. (d) Automatic segmentation algorithm succeeds to converge to a valid snake. It should be noted that there is possibility of wrong detection and no detection always. (e) If segmented snake is satisfactory shape for user, it is confirmed as valid mitochondrion by user and it is added to list of valid snakes.

CHAPTER 7

RESULTS

This chapter presents the results obtained from the implementation of the applications that are developed to prove operation of theoretical work done for this thesis and referenced works. Each section reveals the results that are related with its own methods and implementation. It is possible to go through the theory behind the results from the methods and implementation chapter with similar titles.

CCDB assisted by NCMIR provides the datasets used in works that are done for this thesis [7-9].

Firstly, visualization results that are anatomical planes and volume representations will be presented. Some examples of semi-automatic segmentation results will be presented in 2D. These sections are an introduction to the features of the semi-automatic segmentation tool. More attention is paid to 3D semi-automatic segmentation result. It begins with approaches producing results. There is collaboration between automatic and semi-automatic methods while generating results. Although automatic segmentation results are outside the scope of this thesis, they are represented in the related part of implementation results to measure performance improvement after applying semi-auto method to the same dataset.

The automatic segmentation algorithm produces different results for different blocks of data like ten, twenty, thirty and full slices while working on them. Performance of the automatic segmentation algorithm can vary according to the selected blocks of input. We choose to use results of twenty and thirty slices as an input to the semi-automatic methods to find more scenarios that could be corrected.

Ease of use is a crucial point while developing software that interacts with human. Application should be user-friendly while doing its own job. There are various software evaluation methods to measure user satisfaction while using human-computer interface[62]. Surveys and questionnaire could be given to enough number of people while evaluating the software. Observation of eye movement can be recorded by the use of tracking devices in human-computer interaction laboratories. While a user realizes prearranged scenarios of software with these equipment, valuable data can be collected to expose the use of graphical interface of application. [63]. These techniques require serious financial resources and allotted time to be realized. Due to the limitation of resources and time, we choose to apply task-based evaluation method to measure how easy to use the application [62]. This method is based on definition of tasks for a software and analysis of these tasks with the use of volunteers.

Duration of run-time of algorithms is measured by choosing some sample scenarios that are convenient to apply each feature.

7.1. Visualization

Visualization is presented as axial planes, MinIP representation whose source are directly mitochondrial images and surface reconstruction whose sources are the direct output of preliminary work on the segmentation of mitochondria automatically.

7.1.1. Anatomical Planes

predicle dataset from CCDB is used to express the operation of implementation of anatomical planes. As mentioned before, these planes are extracted from a volume that is constructed from related mitochondrial images directly. There is a selection feature for the user to visualize desired part of the volume. Interaction points of planes are displayed by a point that can be seen below the resulting images, and the user can interact with planes by clicking the images to change the part displayed.

Axial plane of the mitochondrial volume is represented below. The user can see the horizontal section of the mitochondrial volume from this representation. It contains more information compared with other planes since the user can see the membranes and cristae of the selected slices clearly. By coming and going between the segments, the user can observe how mitochondria and other structures are changing in 3D.

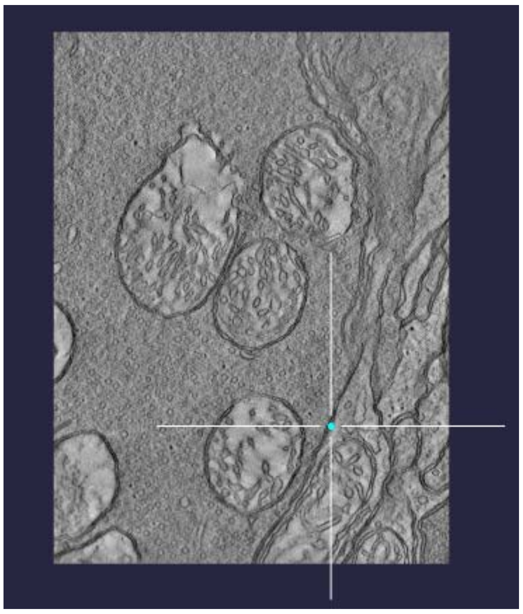


Figure 7.1 Axial Plane Representation

Coronal plane of the mitochondrial volume allows seeing vertical changes through front and rear sections on mitochondrial structures. The user can display how structures on images are changing by proceeding between these segments. This view gives an idea about the vertical size of cristae to user clearly. Moreover, it reveals how membranes are changing without coming and going between axial planes.

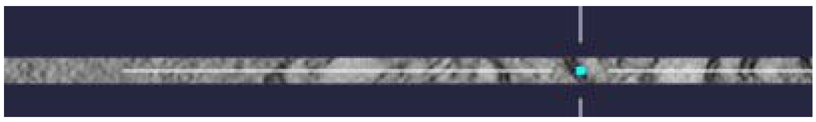


Figure 7.2 Coronal Plane Representation

Sagittal plane gives chance to examine longitudinal vertical view of the selected segment of the mitochondrial volume. This view gives an opinion about the vertical size of cristae from this perspective to the user clearly like coronal plane does. Moreover, changes on boundary of membranes of related segment are displayed for examination purposes.

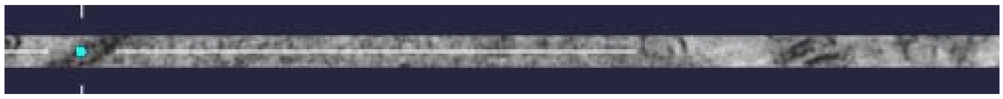


Figure 7.3 Sagittal Plane Representation

Valuable observation is done by examining images from those three planes. It gives an opinion to the user regarding how structures are changing in 3D.

The number of available electro optic images per dataset is not enough to construct enough height when lining up in a row. As shown in the representation of sagittal plane and coronal plane, these vertical views do not provide extensive opportunities as axial plane does. Because of the imbalance of dimensions, oblique view does not contribute to the observation by user.

7.1.2. 3D Volume Representation

Anatomical planes provide perception to the user on how structures are looking in 3D world, but it does not supply enough visual sense of a 3D world for the user. To compensate this half perception, MinIP and surface construction of mitochondria are implemented. This way, the user does not have to visualize structures in 3D on his own.

7.1.2.1. MinIP Visualization

cone.sub dataset from CCDB is used to display MinIP representation method. As can be seen in the image below, region of interest is extracted from the mitochondrial volume and this part is rendered by MinIP method for display purposes. In the figure below, the user selects a part that is close to the membranes and two boundary structures of the membrane are revealed clearly. Rotating the resulting volume gives much more idea to the user regarding the structure.

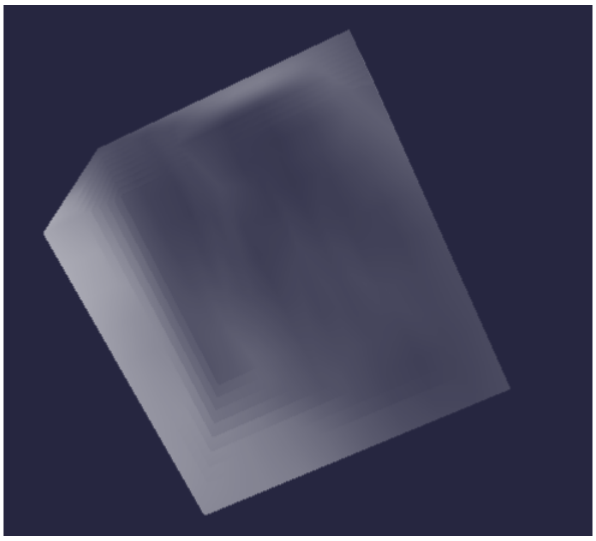


Figure 7.4 MinIP Representation

The user is free to examine any part of the volume by selecting a section. Center of the region of interest is determined by intersection point of anatomical planes that were mentioned in pervious sections.

7.1.2.2. Surface Reconstruction

Output of the automatic segmentation algorithm for *6_22.sub* dataset is used to represent the resulting representation of surface reconstruction methods. VTK's implementation of Hoppe's method [50] gives the resulting structure below using data from the automatic segmentation method. It reveals how boundary of mitochondria looks like in 3D completely. The user can examine the contour filter representation by interacting with it in 3D.

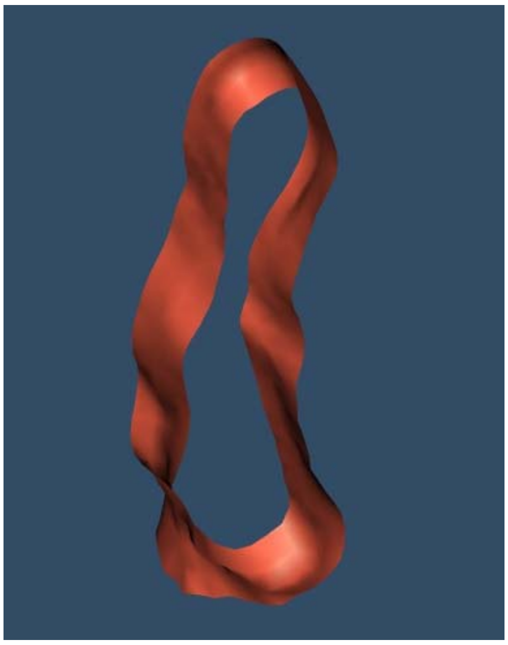


Figure 7.5 Surface Constructed by Using Contour Filter Approach

VTK's implementation of marching cubes algorithm is another option for representing this mitochondria data in 3D. Its performance and representation is very similar to Hoppe's method. Many examples should be examined to determine whether there is a significant difference between these methods.

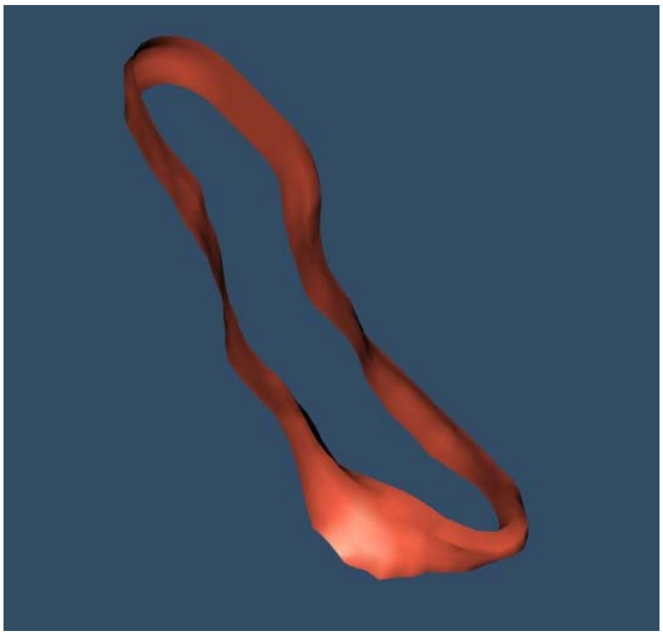


Figure 7.6 Surface Constructed by Using Marching Cubes Algorithm

These surface representation methods provide a 3D perception of the boundary of mitochondria for the researcher, which can lead various studies in this area.

7.2. 3D Semi-Automatic Segmentation

3D detection and segmentation is done by the use of methods mentioned in the method section of this thesis. These methods are splitting, merging, rejection, validation, initiation and manual drawing. We exclude the method of manual drawing while obtaining results not to hide the real effect of the main semi-automatic methods. Since it is quite possible to increase the success of 3D semi-automatic detection and segmentation by the use of manual drawing, it hides whether the performance of main methods are enough to satisfy the users. Manual drawing on the other hand, is used when there is no other choice to extract the observed mitochondria.

7.2.1. Description of Inputs for Methods

TEM images obtained from CCDB are used for visualization and observation purposes benefiting from the representation of anatomical planes of dataset by application. Other than that, output of the 3D automatic segmentation algorithm is presented to the user in the form of constructed surface. Output of the automatic algorithm is based on blocks of images that are grouping ten, twenty or thirty images. Full dataset is divided into the number of these blocks while running the automatic segmentation algorithm

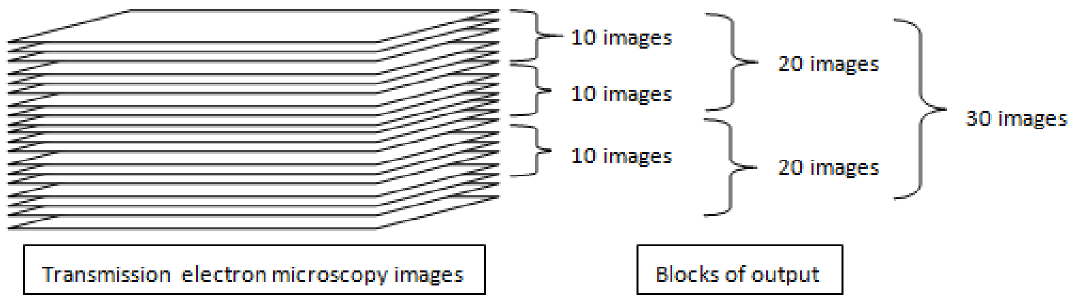


Figure 7.7 Representations of Blocks of Output Data

While evaluating the performance of the automatic segmentation algorithm, the number of detections and segmentations for each block is collected and counted as one statistic for that dataset. We use the blocks that group 20 images and blocks that group 30 images while forming the performance statistics of algorithms. Those blocks are input of the semi-automatic methods separately. The number of blocks belonging to each dataset is given below. Since the automatic segmentation algorithm produces a fixed number of blocks as output, little part of images located at the end of the dataset could not be evaluated. Combination of blocks covers most of the dataset vertically, but not all of it. For instance, *cone.sub* dataset has 97 slices. 80 (20x4) slices of *cone.sub* dataset is covered while using output of twenty slices of blocks. 90 (3x30) slices of *cone.sub* dataset is covered while using output of thirty slices of blocks. Total number of slices and number of blocks for each dataset is shown in Table 7.1.

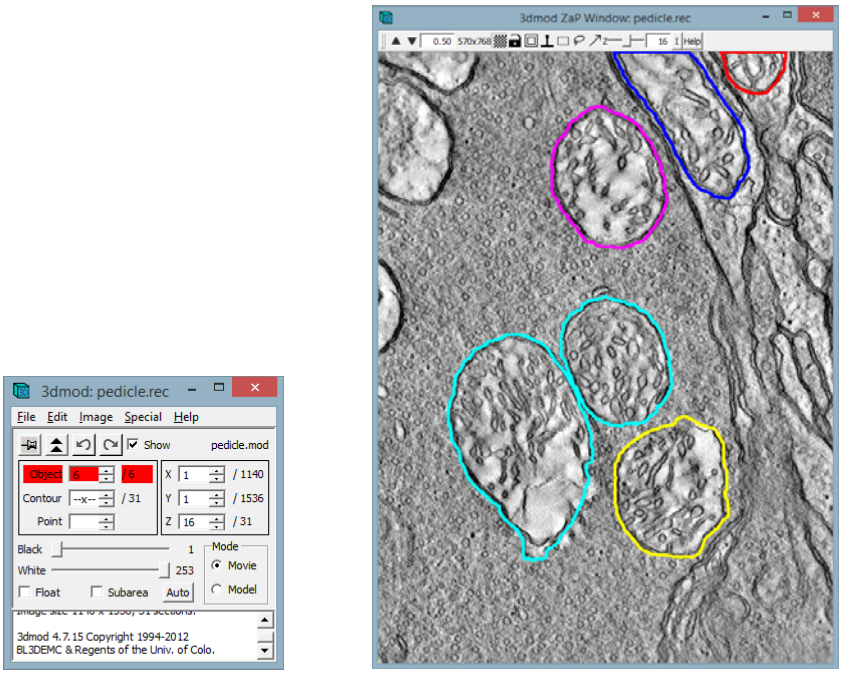
Table 7.1 Number of Blocks for Group of Images

Dataset No	Image Base Name	The number of slices	The number of blocks for 20 images	The number of blocks for 30 images
1	cone.sub	97	4	3
2	gap18_sub	54	2	1
3	6_22.sub	91	4	3
4	bclpb-d.sub	61	3	2
5	pedicle	31	1	1
6	od.sub	91	4	3
7	mac_serial_sub	111	5	3
8	spherule24mos1_	86	4	2

The automatic segmentation algorithm and semi-automatic segmentation methods are applied to each block separately for a dataset. There is no relation constructed between blocks vertically while comparing the performances of methods since output of the automatic segmentation algorithm does not give any information about this. It is possible to make a connection between blocks of dataset by looking at the intersection area of each detected mitochondria and connect the ones that are intersecting with an adequate ratio. Actually, the application is capable of merging selected mitochondria that are located in neighboring blocks. Therefore, there is no limitation to use this feature. However, combining outputs of the automatic segmentation algorithm that are products of different blocks of dataset and constructing results over this assumption is unfair for the performance of the automatic segmentation algorithm since it does not promise any relation between those blocks. There is a possibility of illusion that the automatic algorithm is not successful because of this assumption: Since we focus on the improvement of the semi-automatic segmentation algorithm over the automatic one, we evaluate outputs of each block separately.

7.2.2. Quantitative Measurement of Performance

While measuring the performance of the algorithms, firstly whether the algorithm detects and segments mitochondria is determined. This determination is made by comparing the outputs of algorithms with real drawings of mitochondria, named as ground truth, for each dataset provided by institute of NCMIR in the form of mod file that is specialized for IMOD application. IMOD is developed to make viewing of complex 3D image data of structural biology easier by serving features of view, analysis, and modeling image data in order to better comprehend the related structure[33].



3dmod 4.7.15 Copyright 1994-2012
BL3DEMC & Regents of the Univ. of Colo.

(a) **(b)**

Figure 7.8 Sample View from IMOD for Ground Truth of a Dataset (a) Control Interface of IMOD (b) Presentation Interface of IMOD

It is not a reliable method to evaluate the success of the algorithm by checking and comparing detections visually with real ones. Ground truth information of datasets gives us opportunity to determine the accuracy of each detected and segmented mitochondria that are outputs of implemented algorithms quantitatively.

Each mitochondrion is kept as a polygon with their vertices value as the output of implemented algorithms. Ground truth of datasets is read from mod files and constructed with same data structure to gather real ones and output of the algorithm under the same roof. Our rule for detection is the following method. Let us say that *S* is the polygon of real detection and *R* is the result of an algorithm. Intersection area of these polygons for every slice is calculated. If the total intersection area of real mitochondrion with the candidate one resulted from an algorithm is bigger than a certain portion of the resulting mitochondrion's total area, then it is classified as detected. Otherwise, it is counted as false detection. Total area means sum of each area located in every slice. Detection rule is summarized in Equation 27 and 28.

$$A_{INT} = A_S \cap A_R \qquad (27)$$

$$Detc(S,R) = \begin{cases} DETECTED, & \left(\sum_{k=0}^{t} A_{INT_k} \geq 0.7 * \sum_{k=0}^{t} A_{R_k}\right) \\ NOT-DETECTED, & otherwise. \end{cases} \quad (28)$$

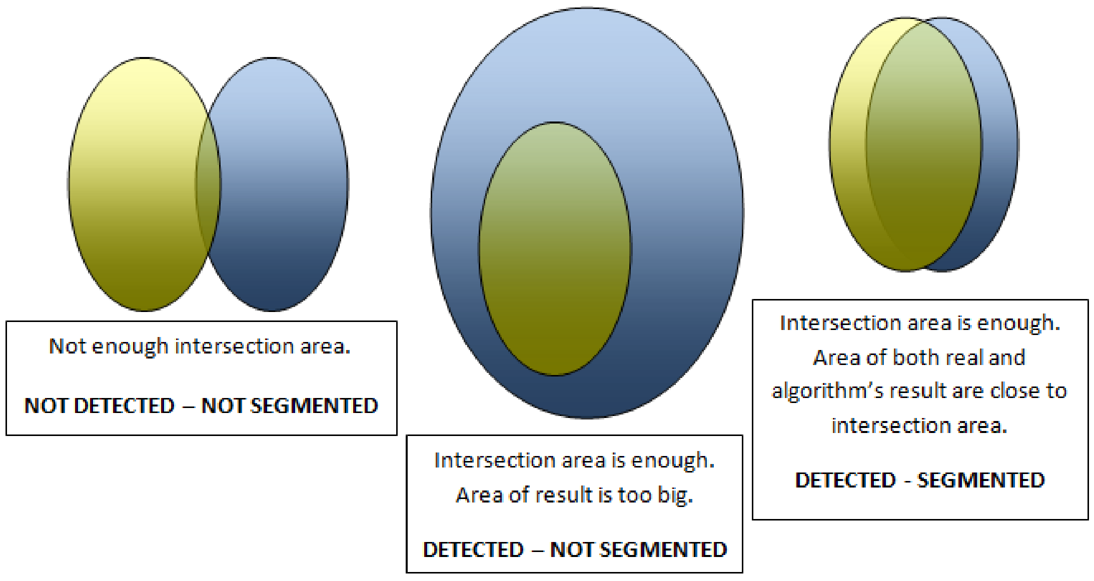
(t represents the slice of the mitochondrion)

While determining whether a mitochondrion is segmented successfully or not, our rule is evaluating that total intersection area of real mitochondrion with the result of the algorithm should be bigger than a certain portion of the resulting mitochondrion's total area and real mitochondrion's total area. In other words, resulting mitochondria should obey the rule of detection first. The other rule is to check whether intersection area is big enough compared to certain portion of the real one. A certain ratio is determined as seventy percent for both the detection and segmentation rules intuitively. Segmentation rule is summarized in Equation 29.

$$Segm(S,R) = \begin{cases} SEGMENTED, & \left(\sum_{k=0}^{t} A_{INT_k} \geq 0.7 * \sum_{k=0}^{t} A_{R_k}\right) \& \left(\sum_{k=0}^{t} A_{INT_k} \geq 0.7 * \sum_{k=0}^{t} A_{S_k}\right) \\ NOT-SEGMENTED, & otherwise. \end{cases} \quad (29)$$

(t represents the slice of the mitochondrion)

Figure 7.9 identifies the detection and segmentation rules visually.


Not enough intersection area.
NOT DETECTED – NOT SEGMENTED
Intersection area is enough.
Area of result is too big.
DETECTED – NOT SEGMENTED
Intersection area is enough.
Area of both real and algorithm's result are close to intersection area.
DETECTED - SEGMENTED

(a) (b) (c)

Figure 7.9 Quantitative Measurement of Performance

Yellow shapes represent area of the real mitochondrion. Blue shapes represent the result of the algorithm. (a) Failure Case for Both Detection and Segmentation (b) Successful Case for Detection (c) Successful Case for Both Detection and Segmentation

Figure 7.9 summarizes the rules of a mitochondrion detection and segmentation. Performances of algorithms are measured after collecting these detection and segmentation statistics for each dataset. By using this information from each dataset, precision and recall measures are used to define performances of the algorithms. Precision and recall are the measures used in the information retrieval and pattern recognition domain to measure how well a system retrieves or recognizes the relevant instances demanded by a user [64]. While precision is called as a positive predictive value, recall is known as sensitivity [65]. The terms true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN) are used for comparison of the results of the classifier under test with trusted external judgments. The terms positive and negative corresponds the classifier's prediction and the terms true and false corresponds whether that prediction is compatible with the external judgment [65]. Definitions of the measures are given as follows in Table 7.2:

Table 7.2 Precision and Recall Definitions

Measure	Formula	Explanation
Precision	TP/(TP+FP)	True positives/Total number of positives predicted
Recall	TP/(TP+FN)	True positives/Total number of actual positives

While measuring the performances of algorithms, TP represents the number of true detections or true segmentations. TP+FP represent the total number of mitochondria detected or segmented by the algorithms. Moreover, TP+FN refer to the total number of actual mitochondria. The results will be given according to these terms and measures in the next sections.

7.2.3. User Validation

The number of trials for each dataset is limited to one trial for each blocks of dataset due to the lack of resources and time. Two users join the study to execute application all over dataset once. These male users are university graduates and experienced enough with computer applications. One of them is twenty nine years old. The other one is thirty years old. Execution of the tests is done in the quiet section of the library in order not to deploy the concentration of these users. Breaks taken during test sessions by the users are not taken into account while results are collected. Users pass through all blocks of each dataset one time.

Statistics obtained from these executions could not be exact results, but it is still gives the idea about ease of use for the application. Another limitation is that trials are not done by professional biologists that can differentiate mitochondria from other cell structures easily. Since, there is no background over mitochondria structures, real results that are provided by CCDB in the form of IMOD model files are taken into account to prevent false detections and measure user satisfaction accurately. Moreover, distribution of the method's use seems to be confident since there are no multiple choices to handle scenarios. To decrease the effect of the algorithm's run-time to human-computer interaction, we chose twenty and thirty slices of results of automatic algorithm while measuring human interaction which yields shorter periods for each algorithm operations. Initiation method is the only method that calls the automatic algorithm part. Splitting and merging methods are used in manual mode of the application.

There are various methods to measure user satisfactions in different areas. We choose to apply the task-based evaluation method to evaluate user satisfaction of the application. More formal application of task-based evaluation method is implemented. GOMS (Goals, Operators, Methods, and Selection rules) is one of the favorite approaches to observe human-computer interaction [66]. This model is developed by Card et al to standardize measurement of human computer interaction on different application area[67]. A goal is something that the user tries to accomplish, operators are actions that the user made during execution, a method is series that achieves a goal[68]. If it is possible to choose different methods to achieve a goal, then these methods are separated from each other by using selection rules concepts[68].

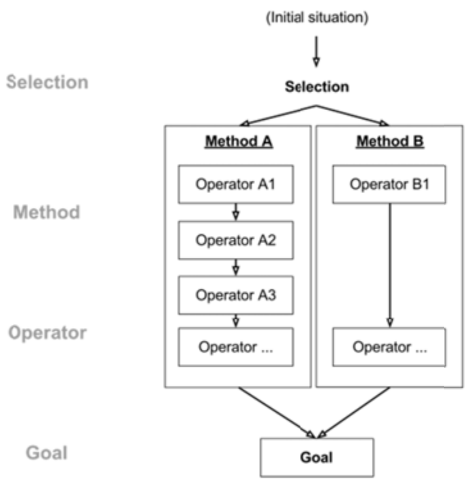


Figure 7.10 The concepts behind a GOMS model[66]

Mapping of GOMS model to semi-automatic tool can be described as follows.

- **Goal:** Improvement of the automatic detection and segmentation algorithm results obtained from dataset taken from CCDB.
- **Operators:** Splitting, merging, deletion, initiation of automatic segmentation algorithm at intended position, selection of low scored mitochondria is semi-automatic techniques that help to achieve our goal.
- **Method:** Use of semi-automatic techniques in a sequence on the output of classical automatic algorithm.
- **Selection**: There are no strict selection rules in our case. Order of usage of semi-automatic techniques can change. The user does not choose any of them from the beginning of execution. Selection rules of our case are flexible.

7.2.4. Detection Performance

We applied automatic algorithm and semi-automatic methods on twenty and thirty slices of eight different datasets from CCDB supported by NCMIR. Detection results are obtained with the approach mentioned in the previous section. These results are summarized in the tables below for blocks of twenty and thirty slices separately.

Table 7.3 Detection Performance Results for 20's Blocks

Detection - 20's blocks			Precision		Recall	
Dataset No	The number of Mitocondria	The number of blocks	Auto (%)	Semi-Auto (%)	Auto (%)	Semi-Auto (%)
1	9	4	94,74	**100,00 ± 0,00**	51,43	**65,71 ± 0,00**
2	6	2	100,00	**100,00 ± 0,00**	41,67	**66,67 ± 0,00**
3	14	4	54,55	**91,18 ± 4,15**	33,96	**58,49 ± 2,67**
4	6	3	90,00	**100,00 ± 0,00**	52,94	**82,35 ± 0,00**
5	6	1	66,67	**83,33 ± 0,00**	66,67	**83,33 ± 0,00**
6	34	4	100,00	**100,00 ± 0,00**	34,85	**53,37 ± 2,73**
7	20	5	50,00	**91,84 ± 2,89**	5,43	**48,91 ± 1,54**
8	1	4	28,57	**100,00 ± 0,00**	100,00	**100,00 ± 0,00**
Overall	96	27	73,07	**95,79 ± 0,88**	48,37	**69,85 ± 0,87**

Table 7.4 Detection Improvement Results for 30's Blocks

Detection - 30's blocks			Precision		Recall	
Dataset No	The number of Mitocondria	The number of blocks	Auto (%)	Semi-Auto (%)	Auto (%)	Semi-Auto (%)
1	9	3	100,00	**100,00 ± 0,00**	48,15	**62,96 ± 0,00**
2	6	1	100,00	**100,00 ± 0,00**	33,33	**66,67 ± 0,00**
3	14	3	59,09	**83,33 ± 7,86**	31,71	**58,54 ± 1,37**
4	6	2	100,00	**100,00 ± 0,00**	54,55	**90,91 ± 0,00**
5	6	1	80,00	**100,00 ± 0,00**	66,67	**100,00 ± 0,00**
6	34	3	100,00	**100,00 ± 0,00**	35,00	**57,00 ± 1,17**
7	20	3	50,00	**82,65 ± 4,33**	3,64	**43,64 ± 0,88**
8	1	2	25,00	**100,00 ± 0,00**	100,00	**100,00 ± 0,00**
Overall	96	18	76,76	**95,75 ± 1,52**	46,63	**72,47 ± 0,43**

It is not difficult to see how semi-automatic methods improve automatic results compared to overall detection success. For twenty slices of blocks, precision of detection is increasing from 73,07% to 95,79% while recall rises from 48,37 to 69,85. Similarly, for thirty slices of blocks, achievement on precision of detections exceeds 76,76 and becomes 95,75 while recall achievement is increasing from 46,63 to 72,47. Looking at the results from two groups of data together, it seems that precision of overall detection is approaching nearly hundred percent. However, recall finds a place around 70%. There is an undeniable success of semi-automatic methods that bring results from 45% to these values. However, there should be other improvement on automatic and semi-automatic methods to catch hundreds percent. The success of precision shows that once the methods find the candidate mitochondria; it can be though as a real one. More effort should be spent to find lost ones.

7.2.5. Segmentation Performance

Segmentation results are collected with the method mentioned in the previous section. The results of these approaches are summarized in Table 7.5 for twenty slices and in Table 7.6 for thirty slices.

Table 7.5 Segmentation Performance Results for 20's Blocks

Segmentation - 20's blocks			**Precision**		**Recall**	
Dataset No	**The number of Mitocondria**	**The number of blocks**	**Auto (%)**	**Semi-Auto (%)**	**Auto (%)**	**Semi-Auto (%)**
1	9	4	94,74	**100,00 ± 0,00**	51,43	**65,71 ± 0,00**
2	6	2	100,00	**100,00 ± 0,00**	41,67	**66,67 ± 0,00**
3	14	4	45,45	**79,41 ± 4,16**	28,30	**50,95 ± 2,67**
4	6	3	80,00	**92,86 ± 0,00**	47,06	**76,47 ± 0,00**
5	6	1	66,67	**83,33 ± 0,00**	66,67	**83,33 ± 0,00**
6	34	4	54,35	**76,71 ± 0,00**	18,94	**41,17 ± 1,77**
7	20	5	20,00	**83,67 ± 2,88**	2,17	**44,56 ± 1,54**
8	1	4	28,57	**100,00 ± 0,00**	100,00	**100,00 ± 0,00**
Overall	96	27	61,22	**89,50 ± 0,88**	44,53	**66,11 ± 0,75**

Table 7.6 Segmentation Performance Results for 30's Blocks

Segmentation - 30's blocks			**Precision**		**Recall**	
Dataset No	**The number of Mitocondria**	**The number of blocks**	**Auto (%)**	**Semi-Auto (%)**	**Auto (%)**	**Semi-Auto (%)**
1	9	3	100,00	**94,12 ± 0,00**	48,15	**59,26 ± 0,00**
2	6	1	100,00	**75,00 ± 0,00**	33,33	**50,00 ± 0,00**
3	14	3	59,09	**83,78 ± 2,00**	31,71	**56,10 ± 2,51**
4	6	2	100,00	**100,00 ± 0,00**	54,55	**90,91 ± 0,00**
5	6	1	80,00	**100,00 ± 0,00**	66,67	**100,00 ± 0,00**
6	34	3	57,14	**71,57 ± 2,99**	20,00	**42,00 ± 1,58**
7	20	3	25,00	**80,69 ± 2,05**	1,82	**41,82 ± 2,38**
8	1	2	25,00	**100,00 ± 0,00**	100,00	**100,00 ± 0,00**
Overall	96	18	68,28	**88,14 ± 0,88**	44,53	**67,51 ± 0,81**

Segmentation rule has a prerequisite that mitochondria should be detected first. Therefore, it is natural to expect lower precision and recall for segmentation compared with detection. For twenty slices of blocks, overall precision of segmentation increases from 61,22% to 89,50% while recall rises from 44,53 to 66,11. Similarly, for thirty slices of blocks, achievement on precision of segmentation changes from 68,28 to 88,14 while recall success increases from 44,53 to 67,51. While evaluating results collected from two groups of data together, it seems that precision of overall segmentation forces the band of ninety percent. This gives high reliability to segmentation methods of the application that is benefiting from automatic algorithm's outputs. However, recall is somewhere around 65%. It is a valuable effort to increase it from 44% to 65% for segmentation purposes. However, it shows that there should be still progress on automatic segmentation algorithm and semi-automatic segmentation algorithm to reach higher percentage of success without the help of manual drawing feature.

7.2.6. Detection and Segmentation Results

Previous sections give the performances for detection and segmentation separately by comparing precision and recall of the automatic and the semi-automatic methods. While gathering these results, each block of twenty and thirty slices contributes calculations by itself. Since there is no logical connection between these blocks vertically while the automatic segmentation algorithm produces results, it is necessary to evaluate each block separately and put all blocks' result together after this evaluation. It is a good strategy to follow while measuring improvement of the semi-automatic methods on performance. However, it is necessary to evaluate percentage of success of semi-automatic methods to detect and segment mitochondria completely. A relation between blocks is constructed this time by looking at the neighboring blocks. If mitochondria have a common intersected area with an enough percentage (70%) in two blocks as the lower and upper blocks, then a relation is constructed between blocks of these mitochondria and it counted as one mitochondrion having both blocks of slices. This calculation is done for every block of mitochondria to find out the complete shape of mitochondria segmented by semi-automatic methods. After completion of mitochondria, these shapes are compared with real drawings of mitochondria. Table 7.7 and Table 7.8 shows recall and precision of detection for semi-automatic methods. Values are based on the results of the user having better performance of detection and segmentation.

Table 7.7 Detection Results after Bringing 20's Blocks Together

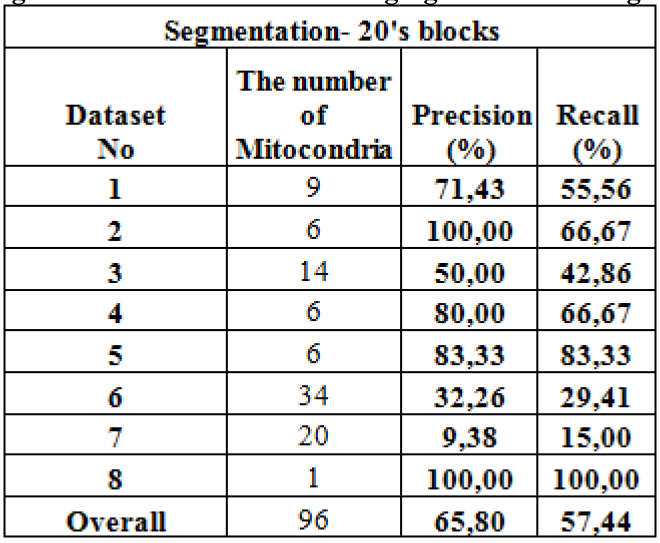
Detection - 20's blocks			
Dataset No	The number of Mitocondria	Precision (%)	Recall (%)
1	9	71,43	55,56
2	6	100,00	66,67
3	14	58,33	50,00
4	6	80,00	66,67
5	6	83,33	83,33
6	34	38,71	35,29
7	20	9,38	15,00
8	1	100,00	100,00
Overall	96	67,65	59,07

Table 7.8 Detection Results after Bringing 30's Blocks Together

Detection - 30's blocks			
Dataset No	The number of Mitocondria	Precision (%)	Recall (%)
1	9	85,71	66,67
2	6	100,00	66,67
3	14	50,00	42,86
4	6	100,00	83,33
5	6	100,00	100,00
6	34	53,85	41,18
7	20	40,00	30,00
8	1	100,00	100,00
Overall	96	78,70	66,34

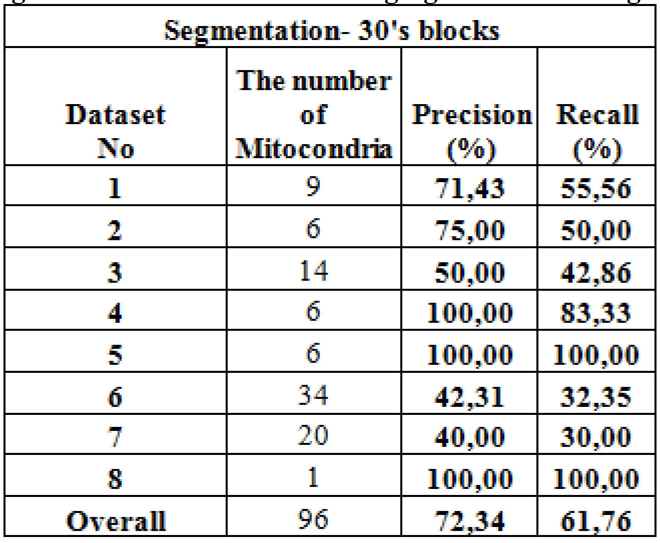
As Table 7.7 and Table 7.8 indicate, precision of detection is changing between 67% and 78% for different blocks of slices. Recall is somewhere between 59% and 66%. Moreover, Table 7.9 and Table 7.10 give precision and recall of segmentation for semi-automatic methods for each datasets. These results reveal that precision values are around 65% and 70 and recall values are about 60%.

Table 7.9 Segmentation Results after Bringing 20's Blocks Together



Segmentation- 20's blocks			
Dataset No	The number of Mitocondria	Precision (%)	Recall (%)
1	9	71,43	55,56
2	6	100,00	66,67
3	14	50,00	42,86
4	6	80,00	66,67
5	6	83,33	83,33
6	34	32,26	29,41
7	20	9,38	15,00
8	1	100,00	100,00
Overall	96	65,80	57,44

Table 7.10 Segmentation Results after Bringing 30's Blocks Together



Segmentation- 30's blocks			
Dataset No	The number of Mitocondria	Precision (%)	Recall (%)
1	9	71,43	55,56
2	6	75,00	50,00
3	14	50,00	42,86
4	6	100,00	83,33
5	6	100,00	100,00
6	34	42,31	32,35
7	20	40,00	30,00
8	1	100,00	100,00
Overall	96	72,34	61,76

.

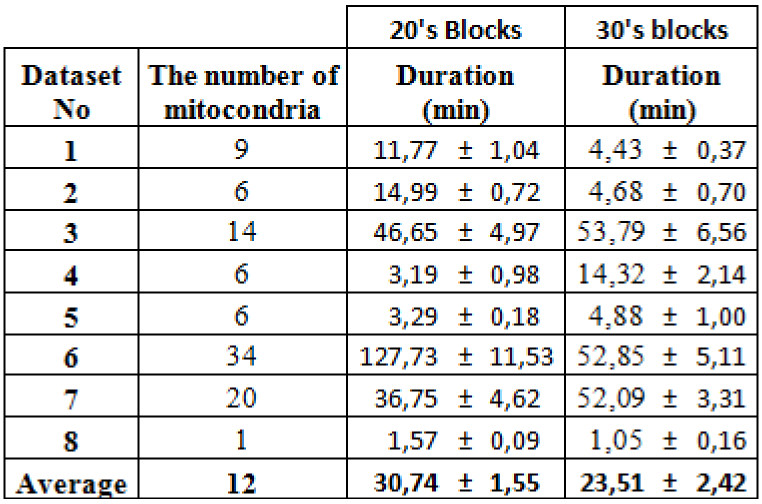
Results over all datasets are encouraging for further studies, but there is still serious work to complete detection and segmentation on these datasets fully.

7.2.7. Duration of Use of Semi-Automatic Methods

Statistics of user validation of this model is the measurement of time spent during execution of each datasets. We do not have any target values to compare the performance of our application. Still, time spent during execution is a valuable statistic to evaluate the application quality in a manner of human-computer interaction. Time elapsed while working on datasets is summarized in Table 7.11

.

Table 7.11 Semi-Automatic Method Duration (User Interaction and Automatic Run Times)



		20's Blocks	30's blocks
Dataset No	**The number of mitocondria**	**Duration (min)**	**Duration (min)**
1	9	11,77 ± 1,04	4,43 ± 0,37
2	6	14,99 ± 0,72	4,68 ± 0,70
3	14	46,65 ± 4,97	53,79 ± 6,56
4	6	3,19 ± 0,98	14,32 ± 2,14
5	6	3,29 ± 0,18	4,88 ± 1,00
6	34	127,73 ± 11,53	52,85 ± 5,11
7	20	36,75 ± 4,62	52,09 ± 3,31
8	1	1,57 ± 0,09	1,05 ± 0,16
Average	**12**	**30,74 ± 1,55**	**23,51 ± 2,42**

Time spent is changing according to complexity of the dataset and the number of mitochondria included. It is changing from 1,57 minutes to 127,73 minutes to bring the output of automatic algorithm to an acceptable level for twenty slices of blocks, while it takes 1,05 minutes to deal with the smallest dataset and 53,79 minutes to handle the largest dataset for thirty slices of blocks. This difference occurs because of the difference between the numbers of blocks for outputs. Thirty slices of blocks have fewer blocks compared to twenty slices of blocks, which leads to fewer scenarios and interaction for thirty slices of blocks. To conclude average dataset is changing from 23,51 minutes to 30,74 evaluating all results together. Actually, it is the time taken to improve all results. Since there is no available data for manual drawing of these datasets and time elapsed for automatic segmentation algorithm, it is not possible to evaluate with manual methods. However, duration for the improvement of the automatic algorithm is acceptable for an average dataset considering about the ratio of improvement mentioned in previous sections. Moreover, it forms a potential data for further research to compare their own work.

Usage distribution of the semi-automatic methods is some valuable information that shows characteristics of false detection done by automatic the algorithm. Distributions of these techniques are displayed in Table 7.12, Table 7.13 and Figure 7.11, Figure 7.12 below. These data represents the number of actions that are taken by the user for each semi-automatic method.

Table 7.12 Use of Semi-Automatic Methods for 20's Blocks

	20's Blocks - Use of Semi-Automatic Methods				
Dataset No	**Merging**	**Splitting**	**Deletion**	**Selection Of Low Scored Mitochondria**	**Manual Initiation**
1	0 ± 0	2 ± 0	8 ± 0	7 ± 0	0 ± 0
2	0 ± 0	1 ± 0	3 ± 0	3 ± 0	1 ± 0
3	6,5 ± 0,71	2 ± 0	10,5 ± 0,71	7 ± 0	4 ± 0
4	0 ± 0	0 ± 0	3 ± 0	6 ± 0	0 ± 0
5	0 ± 0	1 ± 0	2 ± 0	1 ± 0	0 ± 0
6	0 ± 0	18 ± 1,41	8 ± 0	16 ± 0	0,5 ± 0,71
7	0 ± 0	10 ± 1,41	17 ± 1,41	41 ± 1,41	1,5 ± 0,71
8	0 ± 0	0 ± 0	10 ± 0	0 ± 0	0 ± 0
Overall (%)	**3,37 ± 0,16**	**17,63 ± 0,31**	**31,89 ± 0,31**	**42,01 ± 0,31**	**3,63 ± 0,31**

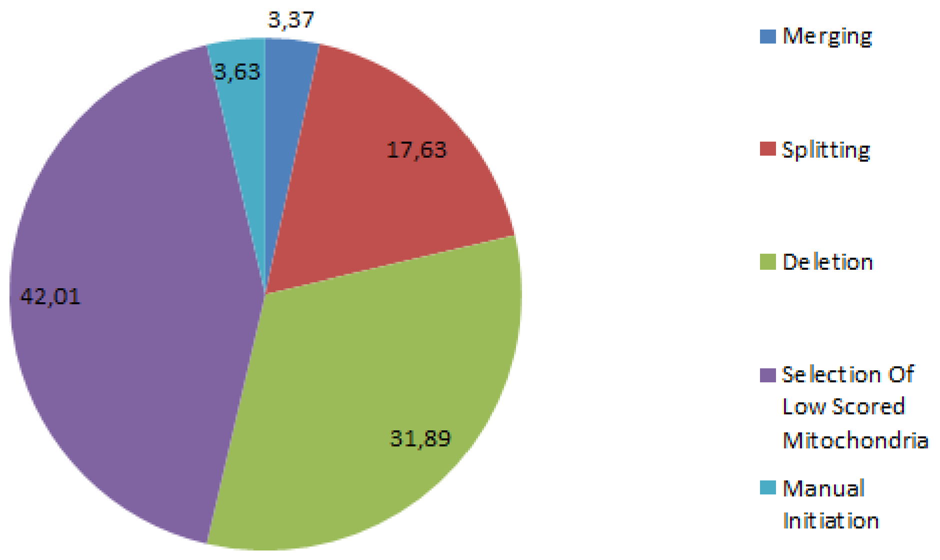


Figure 7.11 Use of Semi-Automatic Methods for 20's Blocks

Looking at the results gathered from the execution of the application for twenty slices, selection of low scored mitochondria and deletion forms the majority of usage together. After these techniques, splitting is the most preferred technique to correct false detections. Merging and initiation techniques are applied to solve special cases.

Table 7.13 Use of Semi-Automatic Methods for 30's Blocks

	30's Blocks - Use of Semi-Automatic Methods				
Dataset No	Merging	Splitting	Deletion	Selection Of Low Scored Mitochondria	Manual Initiation
1	0 ± 0	0 ± 0	0 ± 0	4 ± 0	0 ± 0
2	0 ± 0	0 ± 0	1 ± 0	3 ± 0	0 ± 0
3	4 ± 0	1 ± 0	8 ± 0	11 ± 1,41	3 ± 0
4	0 ± 0	0 ± 0	2 ± 0	4 ± 0	1 ± 0
5	0 ± 0	1 ± 0	5 ± 0	2 ± 0	0 ± 0
6	0 ± 0	9,5 ± 0,71	8,5 ± 0,71	17,5 ± 0,71	0 ± 0
7	0 ± 0	9 ± 0	25,5 ± 2,12	23 ± 4,24	0,5 ± 0,71
8	0 ± 0	0 ± 0	6 ± 0	0 ± 0	0 ± 0
Overall (%)	**2,53 ± 0,00**	**12,98 ± 0,16**	**35,45 ± 0,47**	**40,83 ± 1,41**	**2,85 ± 0,16**

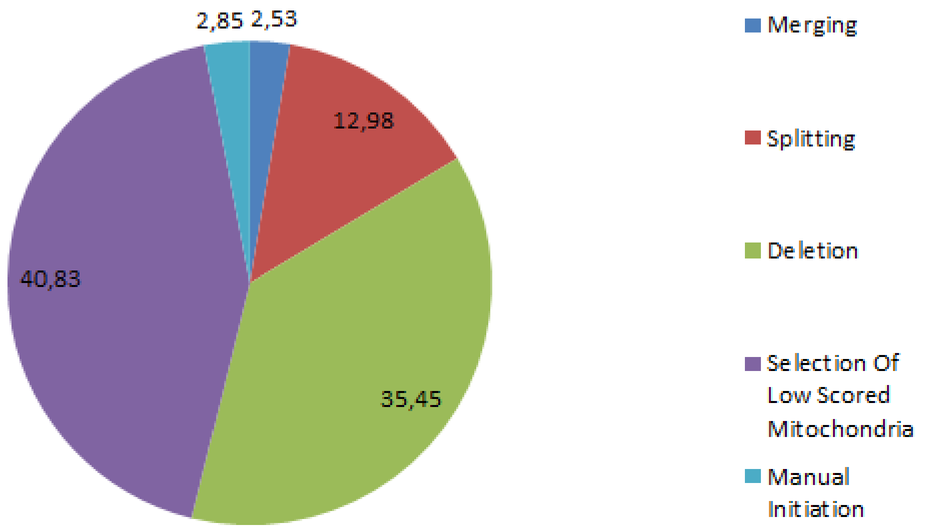


Figure 7.12 Use of Semi-Automatic Methods for 30's Blocks

The results obtained from the execution of the semi-automatic tool for thirty slices are similar to results obtained for twenty slices of blocks. Once again the selection of low scored mitochondria and deletion tools are leaders among them. The splitting tool is following these techniques. The initiation and merging are less preferred techniques to solve false detections.

Almost half of the correction is done by selection of low scored mitochondria among invalid snakes. It means that the automatic segmentation algorithm is very successful to detect mitochondrion, but validation phase of the algorithm excludes these true detections. It would not be right to soften the validation phase of the automatic algorithm since it can increase the false detections, too. Actually, a deletion method is the second most preferred method chosen. However, it would be wrong to say that the results of the automatic algorithm include too much false detection. Sometimes, flow of operator usage while correcting detections includes use of splitting and deletion together. That is to say, splitting does not have to conclude with two valid mitochondria. A false detection can result with excess border from one side. By splitting those mitochondria from here, we obtain one valid and one invalid result. Deletion tool helps us to get rid of that invalid result. This flow shows the need of a potential cutting tool features of which should be examined thoroughly before the implementation.

Even though splitting and deletion are used together, at least approximately 22% (35% - 13%) of the results were rejected because they are totally false detections. It means that at least a quarter of detections and segmentations of automatic algorithm are wrong and they need to be rejected. One potential cause of this phenomenon is wrong detected curves which is the main input of the automatic algorithm phase. They can lead wrong convergence of snake growth in unrelated regions of image.

Merging is a rare method applied by the user during execution. Actually, a few datasets need this technique, but there is no other way to correct this false detection without the help of the merging tool. Therefore, it is still a valuable feature that should be included in any case. Manual initiation of the automatic segmentation algorithm is another rare method used during trials. Curve structures are main barriers for stopping snake growth and convergence. Absence of these structures makes it impossible to detect any mitochondrion. Initialization of snake in such an area is an unsuccessful attempt and waste of time for the user. Moreover, potential convergence of a snake should have already been included in invalid ones. Selection of low scored mitochondria tool takes care of these issues. Manual initiation tool can help the user in the region that has too crowded invalid snakes, and make it impossible for the user to make a proper observation. Without any intervention to the area, the initiation tool does not have too much power because of the absence of necessary structures that form a valid snake there.

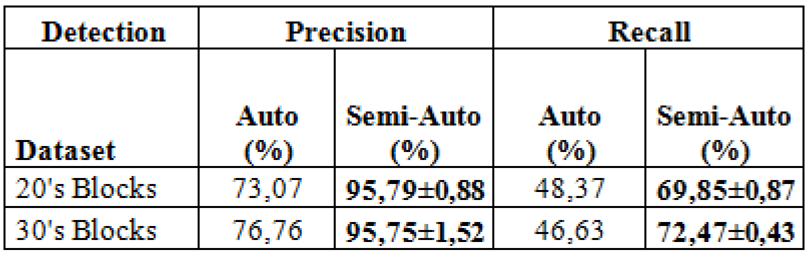
It is impossible to say that results gathered from these executions are permanent. Approaches and experience of the user could change time statistics easily. Allotted time to correct cases could lead a change on the distribution and time. However, it does not mean that there would be sharp changes on the results.

Information gathered from these executions is enough to interpret the performance of semi-automatic tools.

7.2.8. Summary of the Results

There are various results presented in previous sections. In this section, overall results of each measurement for detection and segmentation is brought together to observe the general performance of semi-automatic methods under same title. Table 7.14 summarizes the performance of detection by comparing the performance of automatic segmentation algorithm with semi-automatic segmentation methods.

Table 7.14 Overall Results for Detection Performance after Use of Semi-Automatic Methods

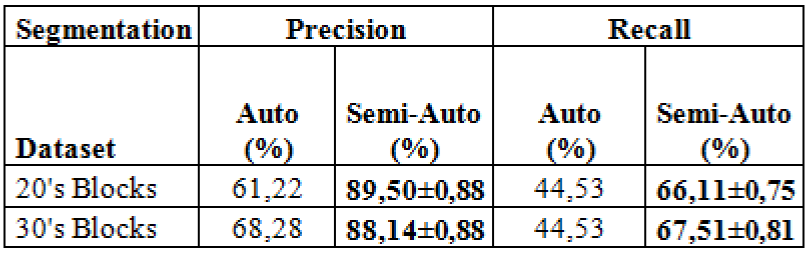


Detection	Precision		Recall	
Dataset	Auto (%)	Semi-Auto (%)	Auto (%)	Semi-Auto (%)
20's Blocks	73,07	**95,79±0,88**	48,37	**69,85±0,87**
30's Blocks	76,76	**95,75±1,52**	46,63	**72,47±0,43**

Table 7.15 summarizes the performance of segmentation by comparing the performance of automatic segmentation algorithm with semi-automatic segmentation methods.

.

Table 7.15 Overall Results for Segmentation Performance after Use of Semi-Automatic Methods



Segmentation	Precision		Recall	
Dataset	Auto (%)	Semi-Auto (%)	Auto (%)	Semi-Auto (%)
20's Blocks	61,22	**89,50±0,88**	44,53	**66,11±0,75**
30's Blocks	68,28	**88,14±0,88**	44,53	**67,51±0,81**

Table 7.16 shows the detection and segmentation performance of semi-automatic methods to detect and segment mitochondria completely. Results are based on the user having better statistics.

Table 7.16 Overall Complete Results for Detection and Segmentation after Use of Semi-Automatic Methods

	Detection		Segmentation	
Dataset	Precision (%)	Recall (%)	Precision (%)	Recall (%)
20's Blocks	67,65	59,07	65,80	57,44
30's Blocks	78,70	66,34	72,34	61,76

Table 7.17 shows the summary of the average durations of use of semi-automatic methods for twenty blocks of slices and thirty blocks of slices.

Table 7.17 Overall Duration for Use of Semi-Automatic Methods

Duration (min)	20's Blocks	30's blocks
Average	30,74 ± 1,55	23,51 ± 2,42

Table 7.18 reveals the summary of the overall distribution of semi-automatic segmentation during executions.

Table 7.18 Overall Distribution of Use of Semi-Automatic Methods

Overall (%)	Merging	Splitting	Deletion	Selection Of Low Scored Mitochondria	Manual Initiation
20's blocks	3,37±0,16	17,63±0,31	31,89±0,31	42,01±0,31	3,21±0,31
30's blocks	2,53±0,00	12,98±0,16	35,95±0,47	40,83±1,41	2,85±0,16

When analyzing the all results coming from different approach, it is inevitable to accept improvement succeed by semi-automatic methods and encouraging performance for further studies in detection and segmentation work for CCDB datasets.

7.2.9. Tool Presentation

Semi-automatic methods are implemented in a particular discipline to meet the user’s need. This effort ends up with a specific tool that is capable of applying the mentioned semi-automatic methods to specified datasets. Overview of the tool can be examined in Figure 7.13.

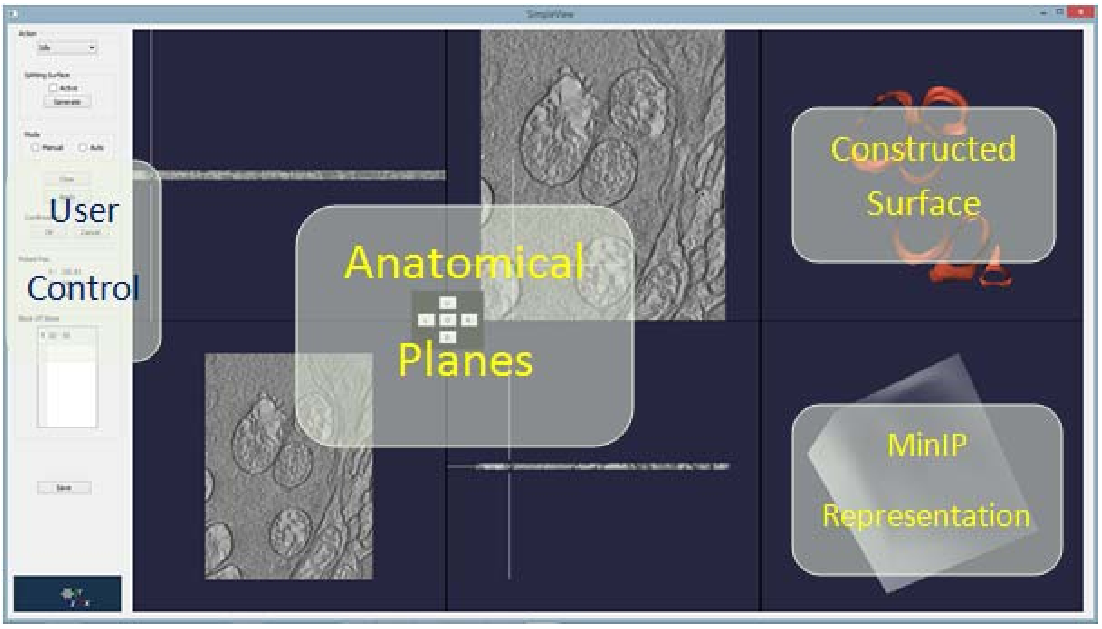


Figure 7.13 Overview of Semi-Automatic Tool

There is a region for observing anatomical planes, constructed surfaces and MinIP representation. All of these regions are interacting with user for better utilization. Left region of the tool is reserved for the user control that answers necessities to apply the semi-automatic methods.

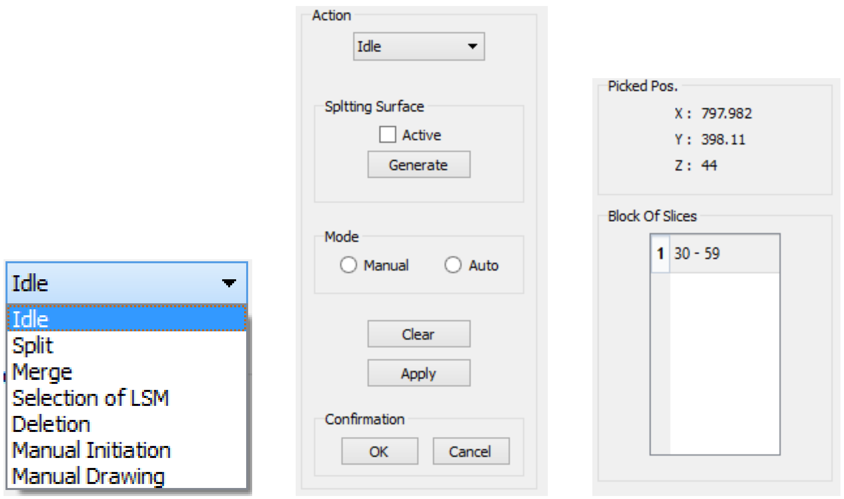


Figure 7.14 User Controls of Semi-Automatic Tool

Figure 7.14 shows the user controls located in left region of user computer interface of the application. There is a field for choice of semi-automatic methods. Other than this, there are some settings which are related to specific a method if needed. The splitting method requires defining a splitting surface. There is a control to activate definition of points belonging to that surface and another one approves the generation of this splitting surface. Mode field determines whether the user want to run the automatic segmentation algorithm in case of splitting and merging methods. If the user selects the manual mode, the application brings the result of method directly without the help of the automatic algorithm. Otherwise, the automatic algorithm works and the result consists of the output coming from this automatic process, too. If the user wants to give up and return to the initial situation while working on any methods, clear button helps the user do it. This control just clears the current situation of the application and the user can begin to work with a different method with different parameters. The apply button starts the process of methods or just realize the method. Selection of low scored mitochondria and deletion are methods that are applied immediately when this button is pushed. Splitting, merging, initiation and manual drawing are methods that are started by the use of this button. After the running process of these methods, their results need confirmation from the user. At this point, the confirmation area gives the user opportunity to approve the results of certain methods like splitting, merging, and manual initiation. Sometimes, the methods can fail in certain scenarios. In this case, user may want to change some parameters or give up the process. Confirmation phase gives the user this choice.

Unrelated controls for a certain method are disabled to prevent false user actions and protect the status of the application. User interface is not fixed for future works. There is always possibility of expand, change the view and controllers.

CHAPTER 8

CONCLUSION AND FUTURE WORKS

8.1. Conclusion

Development and implementation of some user-guided semi-automatic methods to segment mitochondria on transmission electron microscopy images were the main aims of this study since the earlier automatic active contour based detection and segmentation algorithm applied on slices of mitochondria dataset contains potential errors. Therefore, semi-automatic methods were developed and applied to correct output of wrongly detected and segmented mitochondria. The techniques developed for the use are splitting, merging, deletion, initiation of automatic segmentation algorithm at intended position, selection of low scored mitochondria and manual drawing. Splitting and merging are good techniques to arrange existing snakes. Deletion tool give the opportunity to delete false detection arising from the automatic segmentation algorithm. Selection of low scored mitochondria is useful tool to include correct detections which are counted as invalid due to the automatic segmentation tool. Manual initiation gives opportunity to use initiate a snake with softer automatic segmentation parameters at intended position. Manual drawing is a feature when the user does not have any choice left for detection of mitochondria. Moreover, visualization of mitochondria in 3D plays a critical role for the user for the interaction with semi-automatic processes.

There is a clear improvement on the performance of detection and segmentation after applying the semi-automatic methods for datasets Precision of overall detection performance is approaching nearly hundred percent. Recall value rises from 45% to 70%. The nearly hundred percent precision shows that once the methods detect the candidate mitochondrion; it can be evaluated as real mitochondrion. When it comes to the segmentation performance, precision of overall segmentation reaches the ninety percent values. The segmented mitochondria seem to be highly reliable with this value. Recall rises from 44% to 65% as a result of use of the semi-automatic methods. There is an undeniable improvement on the performance of detection and segmentation. However, there should be further studies to succeed detection and segmentation completely when looking at the recall values of the both performances.

After combining the outputs of the semi-automatic methods vertically in blocks, it is possible to evaluate the performance of these methods to detect and segment whole mitochondria with its all blocks. While precision changes between 65% and 79%, recall varies from 59% to 66%. These values are a clear evidence of precious performance, but, they are also indicators of the necessity of further studies to complete detection and segmentation entirely.

Time elapsed for detection and segmentation of the mitochondria is changing according to complexity of the dataset and the number of mitochondria. Thirty slices of blocks have fewer blocks compared to twenty slices of blocks, which results in lower interaction times. To complete semi-automatic segmentation of an average dataset, it takes approximately 23-30 minutes. This is a much preferable user interaction time when compared with the full manual segmentation (which is in the orders of many hours).

Distribution of semi-automatic methods shows that selection of low scored mitochondria and deletion are dominant methods in all. Selection of low scored mitochondria among invalid snakes forms nearly half of the most preferable methods. It means that the automatic segmentation algorithm is more successful than it seems to detect mitochondrion, but validation step of the algorithm eliminates these valuable detections. By looking at this result, it is not a good approach to soften validity parameters for automatic segmentation algorithm due the high possibility of increase for rate of false detections. Other than this, a deletion method is the second most preferred method in all. Deletion action is not the only use of this tool. Sometimes, the use of splitting and deletion together helps the user obtain one valid mitochondrion from invalid mitochondrion whose one side wrongly expanded. Merging is a method to solve special cases in datasets. There is no common usage of this tool, but, it is still necessary to have this tool to solve these special cases. Manual initiation tool is useful to initiate snakes where the automatic segmentation algorithm is not successful.

Results of the work are encouraging and carry potential capacity for further researches in different areas of mitochondria studies. There is a possibility of performance increase depending on the user evaluation and allotted time to work on the datasets. There should be more semi-automatic techniques to facilitate user activity and increase the success of detection and segmentation.

8.2. Future Works

Segmentation and detection of mitochondria based on their membrane boundary is not the only useful information that can be extracted from mitochondrial images. Cristae segmentation is another crucial component of mitochondria since it provides increase in the surface of inner membrane boundary to supply more place for activities taken in this place [69]. There are various studies for the segmentation and analysis of cristae structure of mitochondria [14, 70, 71]. These kinds of studies can be supported by the semi-automatic segmentation and visualization methods that are discussed in this thesis. There will be the need for adaptation of semi-automatic methods for the cristae structures. However, visualization infrastructure could be adapted easily for rendering both cristae structures and mitochondria boundary.

Splitting and merging are not the only effective semi-automatic methods to correct wrongly segmented mitochondria. There are some cases that could be handled much well by different approaches. Level sets is another well-known approach guided by the user control to segment cellular structures[72]. Lucchi et al stated that active contours and level sets have these two important limitations: that are initialization step of algorithms and generalization problem for variations in the target objects the success of these methods in many medical imaging cases. Therefore, these researchers prefer to use Graph-Cut algorithm to segment mitochondrial structures from electron microscopy images[17]. Moreover, Graph-Cut algorithm is a common interactive method applied for segmentation of objects semi-automatically in other studies. [73, 74]. Therefore, Graph-Cut algorithm is another alternative method whose performance should be evaluated and analyzed over mitochondrial images.

Sometimes boundaries of snakes pass through membranes or they are staying inside the mitochondria. There could be a semi-automatic segmentation tool allowing the user to drag boundaries to their real location. It is possible to use a cutting tool instead of surface dragging when detection exceeds the border of the real mitochondria. Implementation of the cutting tool can be done with the help of creation of splitting surface. The user can draw a surface in the area where he wants to get rid of, and the cutting tool can exclude the excess area from the detection.

The method of the selection of the low scored mitochondria can be expanded to other modes like intersection mode. There are hundreds of invalid detections arising from the automatic segmentation algorithm. In such cases, it is possible to observe, intersection of two invalid mitochondria could form a structure that represents a real mitochondrion. By intersecting these two invalid snakes, it is likely to obtain a real mitochondrion. It is difficult to estimate performance of this potential tool without any work, but it makes the user's job easier for such cases.

Splitting and deletion tools are used through trial executions to fulfill the task of cutting in some cases. It shows the necessity of the cutting tool that should be worked on. Splitting and deletion could not be used together for sensitive cases. Before applying the surface dragging feature, a pruning feature could help the user to get rid of excess detections.

Another option to detect mitochondria by user help is starting the process from one slice by marking boundaries manually in that slide and propagating this information to other slices as an input of active contour algorithm and The Kalman filter. This algorithm could be applied consecutively to from 3D structure of mitochondria.

Ease of use is important concept for tools requiring user interaction. Undo is applied frequently to recover from errors while interacting with an application[75]. Undo support is one of the crucial features for error handling and recovery served by a user interface management system [76]. The application developed for thesis has a feature to clean erroneous states, but, it is not an undo support model. This feature doesn’t keep the record of all user actions. It is cleaning the state and gives a chance to user begin from scratch. Therefore, there should be more serious work to support undo feature for better error recovery from user actions.

There is a need to accelerate the segmentation process by using parallel programming techniques due to long process duration of large datasets consisting of many slices. Moreover, visualization part can be supported by VTK’s parallel processing packages to accelerate the reconstruction and interaction with the material. As a result, parallel processing could be applied both to segmentation and visualization work done through these studies to improve user satisfaction.

REFERENCES

[1] Ray, S., *The Cell: A Molecular Approach.* The Yale journal of biology and medicine, 2014. **87**(4): p. 603.

[2] *Mitochondrion*. 2015; Available from: https://en.wikipedia.org/wiki/Mitochondrion.

[3] McBride, H.M., M. Neuspiel, and S. Wasiak, *Mitochondria: more than just a powerhouse.* Curr Biol, 2006. **16**(14): p. R551-60.

[4] Chada, S.R. and P.J. Hollenbeck, *Mitochondrial movement and positioning in axons: the role of growth factor signaling.* Journal of experimental biology, 2003. **206**(12): p. 1985-1992.

[5] Islam, M.S., Y. Niwa, and S. Takagi, *Light-dependent intracellular positioning of mitochondria in Arabidopsis thaliana mesophyll cells.* Plant and cell physiology, 2009. **50**(6): p. 1032-1040.

[6] Schwindling, C., A. Quintana, E. Krause, and M. Hoth, *Mitochondria positioning controls local calcium influx in T cells.* The journal of immunology, 2010. **184**(1): p. 184-190.

[7] Martone, M.E., A. Gupta, M. Wong, X. Qian, G. Sosinsky, B. Ludäscher, and M.H. Ellisman, *A cell-centered database for electron tomographic data.* Journal of structural biology, 2002. **138**(1): p. 145-155.

[8] Martone, M.E., J. Tran, W.W. Wong, J. Sargis, L. Fong, S. Larson, S.P. Lamont, A. Gupta, and M.H. Ellisman, *The cell centered database project: an update on building community resources for managing and sharing 3D imaging data.* J Struct Biol, 2008. **161**(3): p. 220-31.

[9] Martone, M.E., S. Zhang, A. Gupta, X. Qian, H. He, D.L. Price, M. Wong, S. Santini, and M.H. Ellisman, *The cell-centered database.* Neuroinformatics, 2003. **1**(4): p. 379-395.

[10] *Cell Centered Database*. 2015 [cited 2015; Available from: http://ccdb.ucsd.edu/.

[11] Nguyen-Thanh, N., T.D. Pham, and K. Ichikawa. *Segmentation of mitochondria in intracellular space*. in *Computational Intelligence in Bioinformatics and Computational Biology (CIBCB), 2013 IEEE Symposium on*. 2013. IEEE.

[12] Suesse, H., W. Ortmann, C. Janin Lautenschläger, M.K. Lautenschläger, J. Grosskreutz, and J. Denzler, *Quantitative analysis of pathological mitochondrial morphology in neuronal cells in confocal laser scanning microscopy images.* Proceedings IWBBIO, 2014.

[13] Ong, S.-B. and D.J. Hausenloy, *Mitochondrial morphology and cardiovascular disease.* Cardiovascular research, 2010. **88**(1): p. 16-29.

[14] Frey, T.G., G.A. Perkins, and M.H. Ellisman, *Electron tomography of membrane-bound cellular organelles.* Annu. Rev. Biophys. Biomol. Struct., 2006. **35**: p. 199-224.

[15] Pham, D.L., C. Xu, and J.L. Prince, *Current methods in medical image segmentation 1.* Annual review of biomedical engineering, 2000. **2**(1): p. 315-337.

[16] Sampe, I.E., H.-W. Dann, Y. Tsai, and C.-C. Lin. *Segmentation of mitochondria in fluorescence micrographs by SVM.* in *Biomedical Engineering and Informatics (BMEI), 2011 4th International Conference on*. 2011. IEEE.

[17] Lucchi, A., K. Smith, R. Achanta, G. Knott, and P. Fua, *Supervoxel-based segmentation of mitochondria in EM image stacks with learned shape features.* Medical Imaging, IEEE Transactions on, 2012. **31**(2): p. 474-486.

[18] Seyedhosseini, M., M.H. Ellisman, and T. Tasdizen. *Segmentation of mitochondria in electron microscopy images using algebraic curves*. in *Biomedical Imaging (ISBI), 2013 IEEE 10th International Symposium on*. 2013. IEEE.

[19] Jorstad, A. and P. Fua. *Refining Mitochondria Segmentation in Electron Microscopy Imagery with Active Surfaces*. in *European Conference on Computer Vision (ECCV) Workshop on Non-Rigid Shape Analysis and Deformable Image Alignment*. 2014.

[20] Giuly, R.J., M.E. Martone, and M.H. Ellisman, *Method: automatic segmentation of mitochondria utilizing patch classification, contour pair classification, and automatically seeded level sets.* BMC bioinformatics, 2012. **13**(1): p. 29.

[21] Mumcuoglu, E., R. Hassanpour, S. Tasel, G. Perkins, M. Martone, and M. Gurcan, *Computerized detection and segmentation of mitochondria on electron microscope images.* Journal of microscopy, 2012. **246**(3): p. 248-265.

[22] Tasel, S.F., R. Hassanpour, E.U. Mumcuoglu, G.C. Perkins, and M. Martone, *Automatic detection of mitochondria from electron microscope tomography images: a curve fitting approach.* SPIE, 2014. **9034**: p. 903449.

[23] Olabarriaga, S.D. and A.W. Smeulders, *Interaction in the segmentation of medical images: A survey.* Medical image analysis, 2001. **5**(2): p. 127-142.

[24] Heinone, T., P. Dastidar, P. Kauppinen, J. Malmivuo, and H. Eskola, *Semi-automatic tool for segmentation and volumetric analysis of medical images.* Medical and Biological Engineering and Computing, 1998. **36**(3): p. 291-296.

[25] Kang, Y., K. Engelke, and W.A. Kalender, *Interactive 3D editing tools for image segmentation.* Medical Image Analysis, 2004. **8**(1): p. 35-46.

[26] Yushkevich, P.A., J. Piven, H.C. Hazlett, R.G. Smith, S. Ho, J.C. Gee, and G. Gerig, *User-guided 3D active contour segmentation of anatomical structures: significantly improved efficiency and reliability.* Neuroimage, 2006. **31**(3): p. 1116-28.

[27] Barrett, W.A. and E.N. Mortensen, *Interactive live-wire boundary extraction.* Medical Image Analysis, 1997. **1**(4): p. 331-341.

[28] Armstrong, C.J., B.L. Price, and W.A. Barrett, *Interactive segmentation of image volumes with Live Surface.* Computers & Graphics, 2007. **31**(2): p. 212-229.

[29] Jurrus, E., S. Watanabe, R.J. Giuly, A.R. Paiva, M.H. Ellisman, E.M. Jorgensen, and T. Tasdizen, *Semi-automated neuron boundary detection and nonbranching process segmentation in electron microscopy images.* Neuroinformatics, 2013. **11**(1): p. 5-29.

[30] Calhoun, P.S., B.S. Kuszyk, D.G. Heath, J.C. Carley, and E.K. Fishman, *Three-dimensional Volume Rendering of Spiral CT Data: Theory and Method 1*. Radiographics, 1999. **19**(3): p. 745-764.

[31] Bruckner, S. *Performing Maximum Intensity Projection with the Visualization Toolkit*. in *Seminar Paper, Austria*. 2002.

[32] Pavone, P., G. Luccichenti, and F. Cademartiri, *From maximum intensity projection to volume rendering.* Semin Ultrasound CT MR, 2001. **22**(5): p. 413-9.

[33] Kremer, J.R., D.N. Mastronarde, and J.R. McIntosh, *Computer visualization of three-dimensional image data using IMOD.* J Struct Biol, 1996. **116**(1): p. 71-6.

[34] Sandberg, K., D.N. Mastronarde, and G. Beylkin, *A fast reconstruction algorithm for electron microscope tomography.* Journal of Structural Biology, 2003. **144**(1-2): p. 61-72.

[35] Marker, J., I. Braude, K. Museth, and D. Breen. *Contour-based surface reconstruction using implicit curve fitting, and distance field filtering and interpolation*. in *Proceedings of the International Workshop on Volume Graphics*. 2006.

[36] Braude, I., J. Marker, K. Museth, J. Nissanov, and D. Breen, *Contour-based surface reconstruction using MPU implicit models.* Graphical models, 2007. **69**(2): p. 139-157.

[37] Yan, R.-g., X.-d. Guo, and C. Xu, *Reconstruction and Visualization of Human Gastrointestinal Tract.* International journal of biomedical science: IJBS, 2012. **8**(1): p. 22.

[38] Tasel, S., *Phd. Thesis Report for Thesis Supervising Committee, Detection and Segmentation of Mitochondria from Electron Microscope Tomography Images*. 2014, INFORMATICS INSTITU: MIDDLE EAST TECHNICAL UNIVERSITY. p. 3-4.

[39] Tomasi, C. and R. Manduchi. *Bilateral filtering for gray and color images*. in *Computer Vision, 1998. Sixth International Conference on*. 1998. IEEE.

[40] Kass, M., A. Witkin, and D. Terzopoulos, *Snakes: Active contour models.* International journal of computer vision, 1988. **1**(4): p. 321-331.

[41] *VTK - The Visualization Toolkit.* 2014.

[42] Schroeder, W.J., L.S. Avila, and W. Hoffman, *Visualizing with VTK: a tutorial.* Computer Graphics and Applications, IEEE, 2000. **20**(5): p. 20-27.

[43] Schroeder, W.J., K.M. Martin, and W.E. Lorensen. *The design and implementation of an object-oriented toolkit for 3D graphics and visualization*. in *Proceedings of the 7th conference on Visualization'96*. 1996. IEEE Computer Society Press.

[44] Papademetris , X. and A. Joshi, *An Introduction to Programming for Medical Image Analysis with the Visualization Toolkit*. 2 ed. 2009. 283.

[45] Bell, J.T. *Visualization Toolkit (VTK) Tutorial*. 2004; Available from: http://www.cs.uic.edu/~jbell/CS526/Tutorial/Tutorial.html.

[46] *Anatomical plane*. 2015 [cited 2015; Available from: https://en.wikipedia.org/wiki/Anatomical_plane.

[47] Wallis, J.W., T.R. Miller, C.A. Lerner, and E.C. Kleerup, *Three-dimensional display in nuclear medicine.* Medical Imaging, IEEE Transactions on, 1989. **8**(4): p. 297-230.

[48] Cody, D.D., *AAPM/RSNA Physics Tutorial for Residents: Topics in CT: Image Processing in CT 1.* RadioGraphics, 2002. **22**(5): p. 1255-1268.

[49] Schroeder, W.J. and K.M. Martin, *Overview of visualization.* Visualization Handbook, Elsevier, 2005: p. 3-35.

[50] Hoppe, H., T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, *Surface reconstruction from unorganized points*. Vol. 26. 1992: ACM.

[51] Lorensen, W.E. and H.E. Cline. *Marching cubes: A high resolution 3D surface construction algorithm*. in *ACM Siggraph Computer Graphics*. 1987. ACM.

[52] *Linear least squares (mathematics)*. 2015; Available from: http://en.wikipedia.org/wiki/Linear_least_squares_(mathematics).

[53] Weisstein, E.W., *Quadratic Surface.* Delta, 2008. **3**(4): p. 5.

[54] Lancaster, P. and K. Salkauskas, *Surfaces generated by moving least squares methods.* Mathematics of computation, 1981. **37**(155): p. 141-158.

[55] Baeg, M., H. Hashimoto, F. Harashima, and J.B. Moore. *Pose estimation of quadratic surface using surface fitting technique*. in *Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on*. 1995. IEEE.

[56] Farin, G., *Curves and surfaces for computer-aided geometric design: a practical guide*. 2014: Elsevier.

[57] *De Casteljau's Algorithm*. 2015 [cited 2015; Available from: https://dai.fmph.uniba.sk/upload/0/07/Ex02.Info.pdf.

[58] Foley, J.D., A. Van Dam, S.K. Feiner, J.F. Hughes, and R.L. Phillips, *Introduction to computer graphics*. Vol. 55. 1994: Addison-Wesley Reading.

[59] *Filling Areas*. [cited 2015; Available from: http://www.e-cartouche.ch/content_reg/cartouche/graphics/en/html/raster_learningObject3.html.

[60] Shimrat, M., *Algorithm 112: position of point relative to polygon.* Communications of the ACM, 1962. **5**(8): p. 434.

[61] Ester, M., H.-P. Kriegel, J. Sander, and X. Xu. *A density-based algorithm for discovering clusters in large spatial databases with noise*. in *Kdd*. 1996.

[62] Helander, M.G., *Handbook of human-computer interaction*. 2014: Elsevier.

[63] Jacob, R.J., *The use of eye movements in human-computer interaction techniques: what you look at is what you get.* ACM Transactions on Information Systems (TOIS), 1991. **9**(2): p. 152-169.

[64] Ting, K.M., *Precision and recall*, in *Encyclopedia of machine learning*. 2010, Springer. p. 781-781.

[65] *Precision and recall*. 2015 [cited 2015; Available from: https://en.wikipedia.org/wiki/Precision_and_recall.

[66] *GOMS (Goals, Operators, Methods, and Selection rules)*. 2015 [cited 2015; Available from: https://en.wikipedia.org/wiki/GOMS.

[67] Card, S.K., A. Newell, and T.P. Moran, *The psychology of human-computer interaction*. 1983.

[68] Diaper, D. and N. Stanton, *The handbook of task analysis for human-computer interaction*. 2003: CRC Press.

[69] Gauthier, N. *Mitochondrial Cristae: Definition, Function & Quiz*. 2015; Available from: http://education-portal.com/academy/lesson/mitochondrial-christae-definition-function-quiz.html.

[70] Perkins, G.A., M.H. Ellisman, and D.A. Fox, *Three-dimensional analysis of mouse rod and cone mitochondrial cristae architecture: bioenergetic and functional implications.* Mol Vis, 2003. **9**: p. 60-73.

[71] Perkins, G., C. Renken, M. Martone, S. Young, M. Ellisman, and T. Frey, *Electron tomography of neuronal mitochondria: three-dimensional structure and organization of cristae and membrane contacts.* Journal of structural biology, 1997. **119**(3): p. 260-272.

[72] Padfield, D., J. Rittscher, N. Thomas, and B. Roysam, *Spatio-temporal cell cycle phase analysis using level sets and fast marching methods.* Medical image analysis, 2009. **13**(1): p. 143-155.

[73] Kwatra, V., A. Schödl, I. Essa, G. Turk, and A. Bobick. *Graphcut textures: image and video synthesis using graph cuts*. in *ACM Transactions on Graphics (ToG)*. 2003. ACM.

[74] Freedman, D. and T. Zhang. *Interactive graph cut based segmentation with shape priors*. in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. 2005. IEEE.

[75] Akers, D., M. Simpson, R. Jeffries, and T. Winograd. *Undo and erase events as indicators of usability problems*. in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2009. ACM.

[76] Yang, Y., *Undo support models.* International Journal of Man-Machine Studies, 1988. **28**(5): p. 457-481.

TEZ FOTOKOPİ İZİN FORMU

ENSTİTÜ

Fen Bilimleri Enstitüsü	☐
Sosyal Bilimler Enstitüsü	☐
Uygulamalı Matematik Enstitüsü	☐
Enformatik Enstitüsü	☐
Deniz Bilimleri Enstitüsü	☐

YAZARIN

Soyadı : ÇÖÇELLİ
Adı : MUSTAFA
Bölümü : Sağlık Bilişimi

TEZİN ADI (İngilizce) : SEMI-AUTOMATIC/USER-GUIDED SEGMENTATION OF MITOCHONDRIA ON TRANSMISSION ELECTRON MICROSCOPY IMAGES

TEZİN TÜRÜ : Yüksek Lisans ☐ Doktora ☐

1. Tezimin tamamı dünya çapında erişime açılsın ve kaynak gösterilmek şartıyla tezimin bir kısmı veya tamamının fotokopisi alınsın. ☐

2. Tezimin tamamı yalnızca Orta Doğu Teknik Üniversitesi kullancılarının erişimine açılsın. (Bu seçenekle tezinizin fotokopisi ya da elektronik kopyası Kütüphane aracılığı ile ODTÜ dışına dağıtılmayacaktır.) ☐

3. Tezim bir (1) yıl süreyle erişime kapalı olsun. (Bu seçenekle tezinizin fotokopisi ya da elektronik kopyası Kütüphane aracılığı ile ODTÜ dışına dağıtılmayacaktır.) ☐

Yazarın imzası .... Tarih ....