Otto-von-Guericke Universität Magdeburg

Faculty of Informatics
Institute for Simulation and Graphics

Master's Thesis

Segmentation of the Sternocleidomastoid Muscle in CT Datasets of the Neck using Stable 3D Mass-Spring Models

Tahir Majeed

Otto-von-Guericke Universität Magdeburg
Faculty of Informatics (FIN)
Institute for Simulation and Graphics (ISG)

Title	**Segmentation of the Sternocleidomastoid Muscle in CT Datasets of the Neck using Stable 3D Mass-Spring Models**
Working Date	17.06.2008 — 15.12.2008
Written By	Tahir Majeed
Degree	Master in Computational Visualistic
Supervisor	**Dipl.-Ing. Jana Dornheim** **Prof. Dr.-Ing.habil. Bernhard Preim**
First Verifier	**Prof. Dr.-Ing.habil. Bernhard Preim**
Second Verifier	**Prof. Dr.-Ing.habil. Klaus D. Tönnies**
Address	Otto-von-Guericke Universität Magdeburg Universitätsplatz 2 D-39106 Magdeburg

In the name of Allah, the Most Gracious, the Most Merciful.

To my beloved parents

Abstract

The purpose of this thesis is to help in the preoperative planning of the neck dissection by an interactive visualization of the patient specific anatomy and pathological situation of the sternocleidomastoid muscle. This requires segmentation of the sternocleidomastoid muscle. The high density of soft tissues around the muscle, large variations in the shape and the size of the muscle and the inhomogeneous nature of the muscle interior makes the segmentation a challenging task.

Stable Mass Spring Model will be used for the segmentation of the muscle from CT datasets. Three different SMSM prototypes exists as blue prints and they are created on the fly, after obtaining cues about the muscle shape with respect to individual datasets. The cues about muscle shape are obtained from the strokes entered by the user. The prior shape and the size knowledge of the muscle is used to bridge gaps in the contour or bridge the muscle contour destroyed by the tumor infiltration.

The developed prototype was tested on 14 CT datasets from 14 individual patients. The segmentation is generally good with the muscle being segmented in all datasets, however, there were some problems in a couple of datasets where we experienced that in some slices the SMSM went over the weak edges. The average error between the gold standard and the segmented muscle is 0.888 mm and the average similarity score between them is 0.75. We have tried to justify that the results are much better than what the numbers show.

Acknowledgements

I would like to express my gratitude to all those who helped me in completing my thesis.

First and foremost, I am deeply grateful to my supervisor Dipl. -Ing. Jana Dornheim for her patience and guidance throughout my thesis. Her incessant support had been a great help. She always believes that things will work and they finally do at the end. She always has something to share whenever I have a query. Her advice and help went a long way in the completion of this dissertation. Then I would like to express gratitude to Dipl. -Ing Lars Dornheim who helped me a lot during the initial stages of my thesis despite being very busy himself. His guidance allowed me to get a grasp of the MSML library and get going. The insight that he provided into the Stable Mass Spring Model was invaluable.

I would like to recognize the support of Prof. Dr. -Ing. habil. Bernhard Preim and Prof. Dr. -Ing. habil. Klaus D. Tönnies. My discussions with Prof. Tönnies showed me new ways and amicably resolved my ambiguities. I would also like to thank Prof. Dr. rer. nat. habil. Stefan Schirra for his guidence.

I am very grateful to my colleagues Peter Hahn and Dirk J. Lehmann for their help in my work. I had some very fruitful discussions with them that broadened my vision. They also provided me a lot of technical information that equipped me to deal with the challenges that lie ahead.

I would also appreciate the support and help of my brother Zaigham Faraz Siddiqui and my friends Sarmad AL-Saiegh and Sohaib Anwar in tackling some problems related to Latex and 3D Studio Max. I would also like to thank my sisters Shabana Ahmed and Rizwana Naseem and father Abdul Majeed Siddiqui in proof reading my thesis. I am deeply grateful to my parents for their relentless support throughout this work and my master program all in all. Their hearty prayers and support had been a great inspiration for me in difficult times. Lastly, I would like to thank all my friends in Germany who have supported me one way or the other during the course of my thesis.

Tahir Majeed
15.12.2008

Abbreviations

AAM	Active Appearance Models
ASM	Active Shape Models
ASSM	Active Shape Structural Models
CGAL	Computational Geometry Algorithms Library
CT	Computed Tomography
FEM	Finite Element Method
FFD	Free Form Deformation
FORMS	Flexible Object Recognition and Modeling System
GUI	Graphical User Interface
HU	Hounsfield Unit
MDL	Module Definition Language
MRI	Magnetic Resonance Imaging
MSML	Magdeburg Shape Model Library
PCA	Principal Component Analysis
PDM	Point Distribution Model
PET	Positron Emission Tomography
SDK	Standard Development Kit
SMSM	Stable Mass Spring Model
VTK	The Visualisation ToolKit

Contents

Abstract ix

Acknowledgements xi

Abbreviations xiii

1 Introduction 1
- 1.1 Motivation 2
- 1.2 Goals of the Proposed Work 2
- 1.3 Thesis Organization 3

2 Medical Background 5
- 2.1 Anatomy of the Neck 5
- 2.2 Computed Tomography 6
  - 2.2.1 Artifacts in CT Dataset 8
- 2.3 Muscle Appearance in CT Datasets 11
- 2.4 Summary 15

3 State of the Art 17
- 3.1 Image Based Segmentation 17
  - 3.1.1 Threshold Based Segmentation 18
  - 3.1.2 Region Growing 18
- 3.2 Model Based Segmentation 19
  - 3.2.1 Structural Models 20
  - 3.2.2 Statistical Models 22
  - 3.2.3 Deformable Models 23
  - 3.2.4 Stable Mass Spring Model 29
- 3.3 State of the Art for Muscle Segmentation 34
- 3.4 Successful Applications of the SMSM 35
- 3.5 Summary 37

4 Concept **39**
4.1 Different Types of SMSM . . . 39
4.2 Constructing Elliptical SMSM . . . 43
4.2.1 Interpolation . . . 43
4.2.2 Slice Selection . . . 46
4.2.3 Estimating the Intensity Interval for the Elliptical Model . . . 47
4.2.4 Estimating the Ellipse . . . 48
4.2.5 Construction of the Customized 3D Elliptical SMSM . . . 49
4.3 Constructing 3D Curved Rectangular SMSM . . . 51
4.3.1 Construction of the Customized 3D Curved Rectangular SMSM 53
4.4 Constructing Elliptical Elongated SMSM . . . 54
4.4.1 Construction of the Customized 3D Elliptical Elongated SMSM 55
4.5 Preprocessing . . . 58
4.5.1 Re-Sampling Datasets . . . 58
4.5.2 Cutting Gray Values . . . 58
4.6 Ideas Taken From The Previous Work . . . 58
4.7 Summary . . . 60

5 Implementation **61**
5.1 Software . . . 61
5.2 Prototype For Segmenting The Sternocleidomastoid Muscle . . . 62
5.2.1 MeVisLab Modules . . . 62
5.2.2 MLMSML Modules . . . 63
5.3 SternocleidomastoidMuscleSegmentation Module . . . 65
5.3.1 Graphical User Interface . . . 65

6 Evaluation **69**
6.1 Test Environment . . . 69
6.2 Parameter Estimation . . . 70
6.3 Comparison Coefficients . . . 71
6.4 Qualitative Assessment . . . 73
6.4.1 Cutting Gray Values . . . 73
6.4.2 Re-Sampling . . . 76
6.4.3 Structure Of SMSM . . . 77
6.4.4 Inner and Outer Layer of Masses . . . 77
6.5 Quantitative Assessment . . . 78
6.6 Unsuccessful Segmentation . . . 81
6.6.1 Weak Edges . . . 81
6.6.2 Problems due to Artifact . . . 82
6.7 Summary . . . 83

7 Conclusion **85**
7.1 Discussion 85
7.2 Future Work 86

A Muscle Variations **89**

B 3D Segmented Datasets **91**

C 3D SMSM **93**

Bibliography **95**

List of Tables

2.1 Hounsfield values for some anatomical structures 9

6.1 Parameter values for SMSM 71
6.2 Results 79
6.3 Time required to segment muscle from datasets 80

List of Figures

2.1 Muscle attachment at the Sternum and Clavicle 6
2.2 Side view of the muscle showing attachment at the mastoid process. . 7
2.3 The muscle attachment at the Mastoid Process. 7
2.4 CT Scanner 8
2.5 CT Dataset showing one slice 8
2.6 CT Dataset Artifacts 10
2.7 Artifacts found in test CT datasets 10
2.8 Categorization of muscle into three different parts 11
2.9 Elliptical shape of the Sternocleidemastoid muscle 12
2.10 Curved rectangular shape of the Sternocleidemastoid muscle 13
2.11 Thin Elongated Shape of the Sternocleidemastoid Muscle 13
2.12 Soft Tissue Movement Between Slices 13
2.13 Variation in the width of the muscle near the mastoid part. 14

3.1 Threshold Based Segmentation 18
3.2 Region Growing Segmentation 19
3.3 Structural Models 21
3.4 Active Shape Model 23
3.5 An overview of the template matching techniques 24
3.6 Active Contour Model 26
3.7 Mass Spring Model 30
3.8 SMS Model vulnerability scenarios 32
3.9 Torque for 2D SMS Model 33
3.10 Torque for 3D SMS Model 33
3.11 Musculoskeletal segmentation process 35
3.12 Different 3D SMSM Models 36

4.1 Different parts of the muscle. 40
4.2 Shapes of the middle part of the muscle 41
4.3 Shape of the upper part of the muscle 41
4.4 Work flow diagram of the segmentation process 44

4.5 Linear Interpolation . . . 45
4.6 Schematic diagram for ray casting. . . . 48
4.7 Ray casting in a slice. . . . 49
4.8 Schematic diagram of the calculated positions of an elliptical SMSM where masses will be placed. The diagram shows the relative position of the masses with respect to each other in a slice. . . . 50
4.9 3D Elliptical SMSM . . . 52
4.10 Schematic diagram of the calculated positions of curved rectangular SMSM where masses will be placed. The diagram shows the relative position of the masses with respect to each other in a slice. . . . 53
4.11 3D Curved Rectangular SMSM . . . 54
4.12 Schematic diagram of the calculated positions of elliptical elongated SMSM where masses will be placed. The diagram shows the relative position of the masses with respect to each other in a slice. . . . 56
4.13 3D Elliptical Elongated SMSM . . . 57
4.14 SMSM distracted by local minima when the dataset has not been resampled. . . . 59
4.15 SMSM distracted by high gradient force before the cutting of intensity values of the dataset. . . . 59

5.1 MeVisLab network diagram for muscle segmentation . . . 63
5.2 MeVisLab network diagram for comparing model segmentation with manual segmentation . . . 64
5.3 GUI . . . 67

6.1 SMSM adaptation in different regions . . . 74
6.2 SMSM deforms because of strong gradient. . . . 75
6.3 SMSM deforms because of local minima and its solution. . . . 76
6.4 SMSM structure . . . 77
6.5 Effect of inner layer on SMSM adaptation. . . . 78
6.6 Failed segmentation in a dataset. The SMSM went over the weak edges that marks the true muscle boundary. . . . 81
6.7 Failed segmentation in another dataset. The SMSM went over the weak edges that marks the true muscle boundary. . . . 82
6.8 Problem due to the presence of an artifact in the slice. . . . 83

Chapter 1

Introduction

In the beginning medical imaging was solely used for visualization and inspection of the human organs. X-Rays provided the first ever medical image data about the internal organs of a human body [38]. The invention of X-Ray Computed Tomography by Godfrey Hounsfield in 1968 has led to rapid developments in the field of medical imaging technologies. These developments have enabled radiologists and physicians to extract useful life saving information by intra-operatively navigating through the human body. The radiologists and physicians can use information from medical image data for diagnosis, therapy planning and post operative monitoring. The volumetric data obtained from the medical imaging devices gives a better idea about the relative positioning of the anatomical structures which is useful for minimizing the collateral damage to other healthy organs [35]. In turn this helps the patient with less risky, reduced pain, shorter hospitalization and faster recovery process [39].

With the advances in the medical imaging technologies like Computed Tomography (CT), Magnetic Resonance (MR), Positron Emission Tomography (PET), it has become increasingly important to extract useful information about the anatomical structures from the imaged data [47]. The use of the computers has opened new spectrum to extract accurate, repeatable and quantitative data [1].

The main hurdles in obtaining accurate information from the data are due to the huge size of the datasets and the variability of the anatomic structures. The problem becomes more complex with the introduction of the imaging artifacts, spatial aliasing, noise and indistinct boundaries of the neighboring anatomical structures. The main task is then to glean accurately all the elements of one structure and excluding all the elements of the neighboring structure [35].

Low level image processing techniques operating just with the local information fail to extract the accurate boundaries due to the lack of global knowledge about the

---

[1] Duncan et al. [20] gives a detailed overview of the technological and the algorithmic developments in the area of medical image analysis during 1980s and 1990s

size and shape of the target structure. Deformable Models combine the advantages of the low level image processing algorithms together with the global information and are therefore, better suited to segmenting anatomical structures from these datasets. They use bottom-up and top-down strategies to accurately identify the target structure boundaries. They are also flexible enough to adjust to the significant variability of the anatomical structure across different patients [35].

1.1 Motivation

The motivation behind this dissertation is to support the preoperative planning of the neck dissection by a 3D visualization of the patient specific anatomy and the pathological situation of the sternocleidomastoid muscle. This requires the segmentation of the sternocleidomastoid muscle from the neck CT dataset which could be used by orthopaedists, biomechanicians and kinesiologists. The segmentation of the muscle is a challenging task because the muscle is surrounded by quite a few soft tissues of high signal intensity. The proximity of the muscle to the skin creates additional problems, not to mention the inhomogeneous internal region of the muscle and its shape and size variability. The most challenging task of all is bridging the invisible edges in region where the muscle is adjacent to another muscle and in regions where the muscle boundary has been destroyed due to tumor affection.

Traditional image based techniques fail to segment the muscle in such challenging labyrinth. The solution lies in combining the local image information such as gradient and intensity information with the shape and size information of the muscle. In this work *Stable Mass Spring Model (SMSM)* proposed by L. Dornheim [15] have been used for segmenting the sternocleidomastoid muscle. The model combines the local image information with the shape and size information of the muscle and has already been successfully used for segmenting anatomical structures from CT datasets like [19, 14, 13, 32].

1.2 Goals of the Proposed Work

The goal of this thesis is to segment the sternocleidomastoid muscle from the CT dataset using the Stable Mass Spring Model. The SMSM could either be generated automatically through MSML library or a customized SMSM could be build automatically without using the MSML library and human intervention. Cues about the shape and the orientation of the muscle are provided by the user strokes. The sub goals include evaluating whether the shape knowledge incorporated in the SMSM is able to bridge invisible and destroyed muscle boundaries, automating the SMSM and simulation parameters and evaluating the segmentation accuracy.

1.3 Thesis Organization

Thesis is organized as follows;

1. **Chapter 1** provides an **Introduction** and discusses motivation and goals behind this thesis.
2. **Chapter 2** provides a **Medical Background**. It provides anatomical details of the neck. Then it briefly discusses the basics of the CT technology and the appearance and shape of the muscle in the CT datasets.
3. **Chapter 3** reviews the **State Of The Art**. In this chapter, image based and model based techniques have been discussed and then some applications of Stable Mass Spring Model (SMSM) have been presented.
4. **Chapter 4** provides details of the technique developed to segment the sternocleidomastoid muscle.
5. **Chapter 5** provides details about the **Implementation**.
6. **Chapter 6** provides the results and **Evaluate** and analyse them.
7. **Chapter 7** provides a **Conclusion** about the developed technique.

Chapter 2

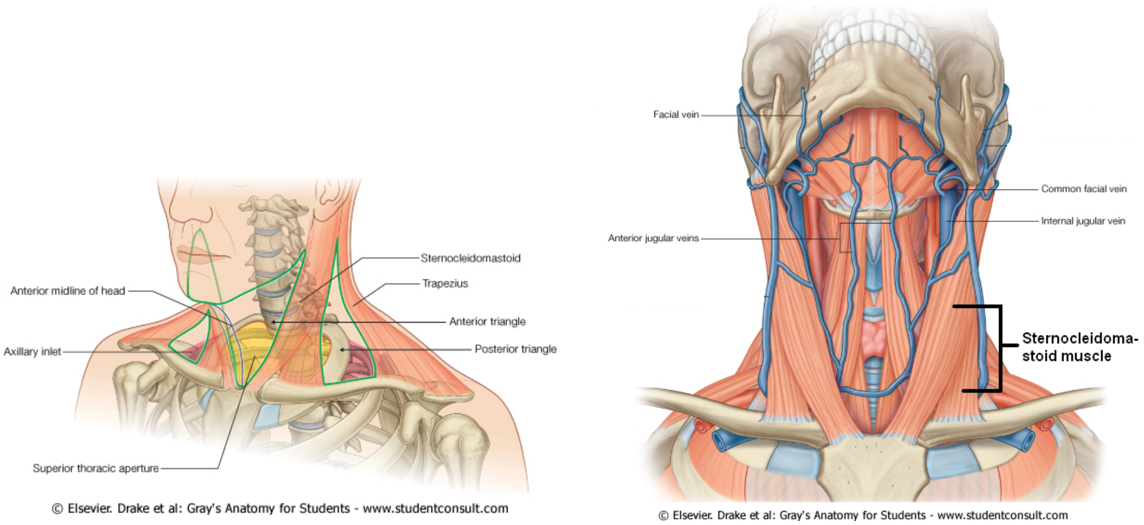
Medical Background

This chapter will present medical related basic information about the anatomical structure that needs to be segmented. Together with this information it also seems relevant to discuss the CT technology, the main source of data, from which the muscle will be segmented. Some basic information about the CT technology and CT datasets are provided. A brief description of the artifacts that crop up in the CT datasets is also given. At the end the appearance of the muscle in the CT datasets is given, with primary focus on the challenges that will arise during the muscle segmentation process.

2.1 Anatomy of the Neck

The name of the muscle *sternocleidomastoid* has been derived from its location in the human body. It starts from the manubrium of the sternum (sterno) and the clavicle (cleido), and joins with the mastoid process of the temporal bone of the skull. The sternum and the clavicle form its two heads [24]. It is a paired muscle situated on either side of the neck. Sternocleidomastoid muscle is situated near the surface in the anterior portion of the neck. The pair of muscle helps in flexing, rotating and protruding the head [24, 36].

The muscle slants across the side of the neck. At both ends it is thin and broad while in the middle, it is thick and narrow (see figure 2.1, 2.2). The sternal head forms a rounded cluster of muscle fibers which connects the muscle to bone at the front. The back part of the muscle which arises from the upper part of the anterior surface of manibrium sterni is fleshy. From this point the muscle slants across the side of the neck moving upward and backward. The clavicular head (see figure 2.1) which emerges from the superior border and anterior surface of the medial third of the clavicle is also fleshy. From this point, the muscle moves almost vertically upwards. The two points from where the muscle originates are separated from each other. As



© Elsevier. Drake et al: Gray's Anatomy for Students - www.studentconsult.com

(a) Side View

© Elsevier. Drake et al: Gray's Anatomy for Students - www.studentconsult.com

(b) Front View.

Figure 2.1: Muscle attachment at the Sternum and Clavicle. Image Source : [40]. The images have been edited.

they move upward, they move closer to each other and just below the middle of the neck they slowly blend together forming a triangle. The junction where the two parts of the muscle meet is thick and rounded. As the muscle moves further upward, a strong tendon[1] joins it with the lateral surface of the mastoid process from apex to superior border. A thin aponeurosis[2] joins the muscle with the lateral half of the superior nuchal line of the occipital bone [24]. The mastoid process and the superior nuchal line are shown in Figure 2.3.

The clavicular head of the muscle varies greatly. The breadth of the clavicular head could be as narrow as the sternal head or it could be as large as 7.5cm.

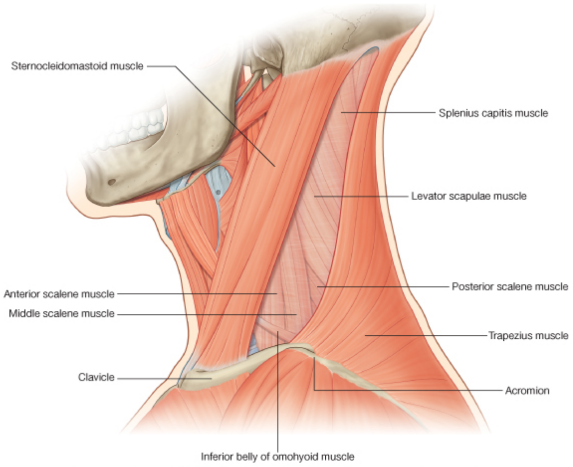
2.2 Computed Tomography

Radiological scanning devices provide data for medical imaging and visualization. These scanning devices include Computed Tomography (CT), Positron Emission Tomography (PET), 3D Ultrasound, Magnetic Resonance Imaging (MRI) and others. The purpose of a medical image data is to help radiologists and physicians in diagnosis, therapy planning, intra operative navigation and post operative monitoring.

---

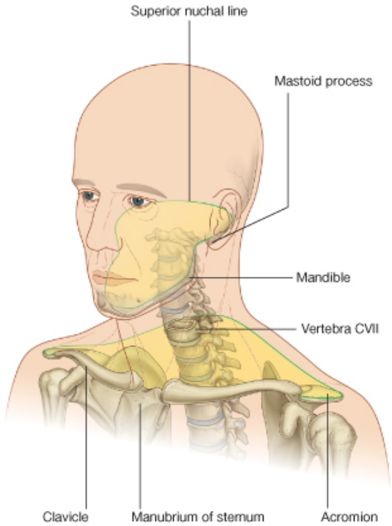
[1] A chord or band of dense, tough, inelastic, white, fibrous tissue serving to connect muscle with a bone

[2] A whitish, fibrous membrane that connects a muscle to a bone or fascia.



© Elsevier. Drake et al: Gray's Anatomy for Students - www.studentconsult.com

Figure 2.2: Side view of the muscle showing attachment at the mastoid process. Image Source : [40].



© Elsevier. Drake et al: Gray's Anatomy for Students - www.studentconsult.com

Figure 2.3: The muscle attachment at the Mastoid Process. Image Source : [40].

During the last 20 years these devices have considerably improved both in terms of the quality of data obtained and the time required to obtain the data. CT and MRI datasets are better than the conventional X-Ray imaging because of their high resolution, accurate localization of anatomical structures in depth, being more sensitive to soft tissues and having good signal to noise ratio [38].

Medical Imaging started in 1895 when William Konrad Röntgen discovered the X-Rays. For the next many decades, X-Rays were the only medium used for the diagnosis by inspecting the X-Ray images. These X-Ray images provided only a 2D view of the internal anatomical structures.

Sir Godfrey Hounsfield invented the first commercial CT scanner in 1968. Figure 2.4 shows a modern CT scanner from Toshiba [3]. The scanner for the first time provided a volumetric representation of objects [38]. A series of individual X-Ray images are combined together to form a volume dataset called CT dataset. These individual X-Ray images or slices could be inspected individually too. This corresponds to viewing a 2D X-Ray image. This slice by slice viewing enables the radiologists and physicians to select and inspect each and every voxel of the dataset. Figure 2.5 shows a slice of the CT dataset. When these slices are combined together and viewed, the dataset shows the 3D structure of the anatomical structures and their relative positioning with respect to the other structures.

The individual X-Ray image is obtained by the scanner that rotates around the object of interest. Different biological objects absorb different amount of X-Rays. A

---

[3] The image is taken from http://www.wired.com/gadgets/miscellaneous/news/2008/04/Toshiba_CTScanner

Figure 2.4: CT Scanner

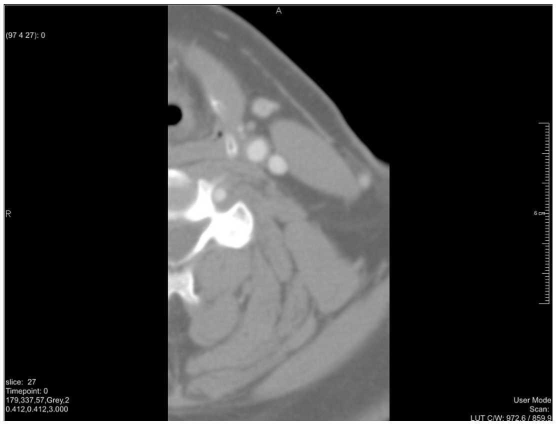


Figure 2.5: CT Dataset showing one slice

dense or thicker object absorbs more radiation as compared to lighter or thinner object as a result denser object appears brighter than a thinner object in CT dataset. A detector positioned on the opposite side from where the X-Rays are emitted, measures the amount of X-Rays received. The amount of X-Ray received is then transformed into intensity value which can then be viewed on a screen.

The intensity value represents the densities of the different anatomical structures. The unit for the intensity value is Hounsfield Unit (HU). It is normalized (see Eq. 2.1) and scaled to cover a range represented by a 12 bit range [38]. The intensity value of water is 0 while that of air is -1000.

$$HU = \left(\frac{\mu - \mu * H_2O}{\mu * H_2O} \right) * 1000 \tag{2.1}$$

Table 2.1 taken from [38] shows the Hounsfield values of different anatomical structures. The table shows that some structures have similar or overlapping values.

CT scans are much better than the conventional X-Ray as they are more sensitive to soft tissues. X-Rays are unable to distinguish between soft tissues like liver and pancreas; while CT scans can distinguish between them. CT scans also do not suffer from the drawback of X-Rays, where an X-Ray after being absorbed in many different soft tissues forms an image [38].

2.2.1 Artifacts in CT Dataset

Although CT datasets are better than conventional X-Ray imaging, they could still experience artifacts such as the following (see [4]).

1. **Aliasing Artifacts or Streak** These artifacts produce dark lines which are

`Tissue Type`	`Hounsfield Value Interval`
Air	-1000
Lung Tissue	-900 ... -170
Fat Tissue	-220 ... -30
Water H_2O	0
Pancreas	10 ... 40
Heart	20 ... 50
Liver	20 ... 60
Kidney	30 ... 50
Bones	45 ... 3000

Table 2.1: Hounsfield values for some anatomical structures

oriented outward from sharp corners. They arise due to motion during scanning (see Figure 2.7b). If the scanned object is metallic, then these artifacts appear because of under sampling (see Figure 2.7a). They might also appear because of insufficient X-Ray tube current which results in insufficient X-Ray penetration [4].

2. **Partial Volume Effect** Partial volume effect causes blurring around sharp edges. When high density material (e.g. bone) and low density material (e.g. cartilage) are adjacent to each other, the scanner is not able to differentiate between them. During the processing stage an average of the two values is taken which results in information loss. This effect can be reduced by increasing the resolution of the data (see Figure 2.6a) [4].

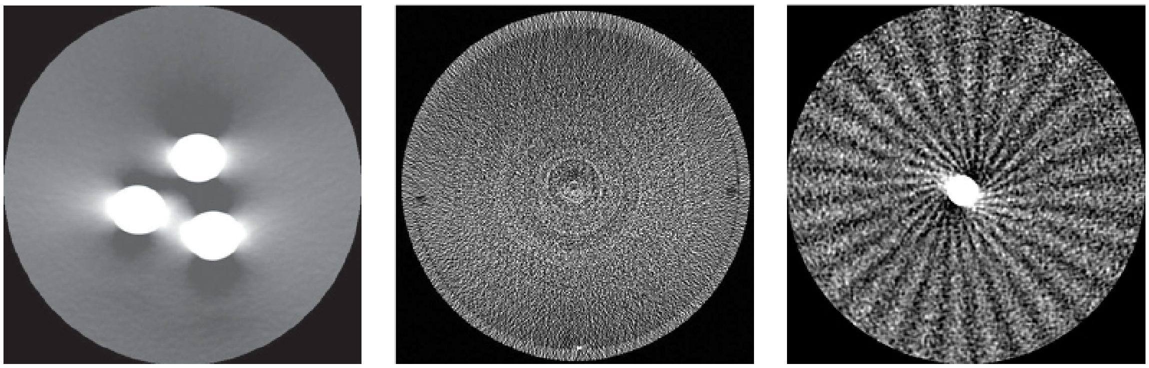
3. **Ring Artifacts** This artifact causes rings to appear within an image. This artifact appears because of a mechanical fault in the detector (see Figure 2.6b) [4].

4. **Noise Artifact** Low signal to noise ratio causes grains to appear on the image. Noise could either be caused by thin slices or by X-Rays not having enough energy to penetrate anatomical structures [4].

5. **Windmill Artifact** Intersection between the detector and reconstruction plane causes streaking to appear on the image. This effect can be reduced by either decreasing the pitch or using filters (see Figure 2.6c) [4].

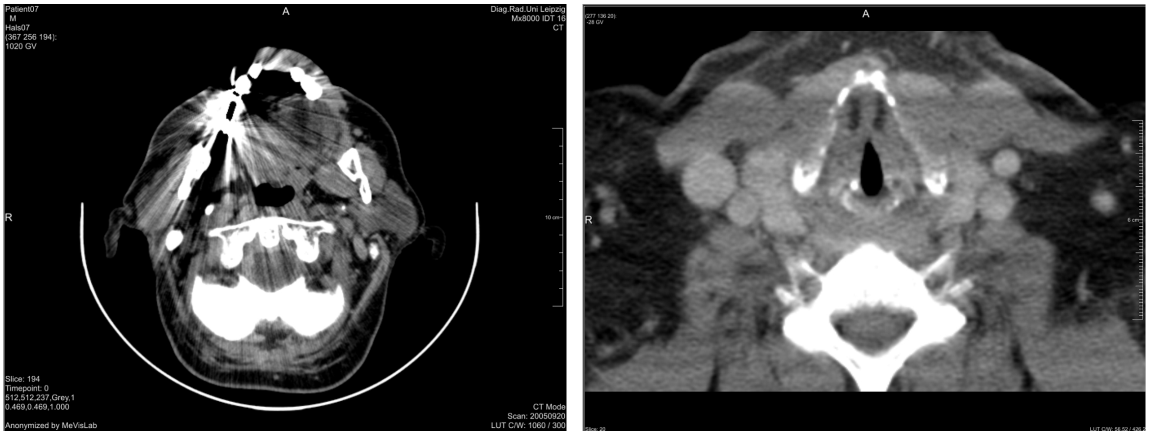
6. **Beam Hardening Artifact** When center of the object is denser than its edges, X-Rays at and around the center are blocked while those around the edges pass through resulting in a *Cupped Appearance*. This effect can be corrected by filtering [4].

A complete list of the artifacts and their avoidance techniques has been discussed in detail by Barrett et al. [4]. Figure 2.7 shows some of the images of the artifacts found in the CT datasets used for testing the prototype we have developed. Motion artifacts and metal implant artifacts were the only artifacts that created problems during the automatic segmentation process.



(a) Partial Volume Effect **(b)** Ring Artifact **(c)** Wind Mill Artifact

Figure 2.6: Artifacts found in CT datasets - *Image Source : Artifacts in CT: Recognition and Avoidance by Julia F. Barrett and Nicholas Keat* [4].



(a) Aliasing Artifacts Due to Metal Implants **(b)** Motion Artifact

Figure 2.7: Artifacts found in test CT datasets

2.3 Muscle Appearance in CT Datasets

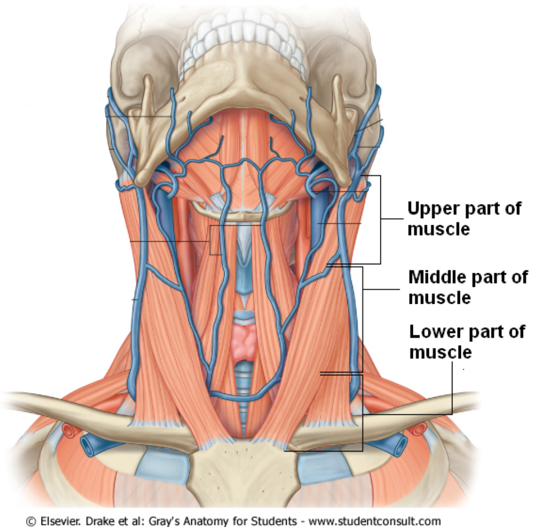

Upper part of muscle
Middle part of muscle
Lower part of muscle
© Elsevier. Drake et al: Gray's Anatomy for Students - www.studentconsult.com

Figure 2.8: Categorization of muscle into three different parts. Image Source : [40]. The image has been edited.

The shape of the sternocleidomastoid muscle can be divided into three parts; lower, middle and upper as can be seen in figure 2.8. The lower part of the muscle starts from the clavicle and the sternum. In this region the muscle is divided into two separate strands. As you move up, the two strands come closer and finally they combine together into a single unit. This is where the lower part of the muscle ends and the middle part of the muscle starts. As you move further up the middle part of the muscle starts to develop a thin, elongated structure. This is where the middle of the muscle ends and the upper part of the muscle begins. The upper part of the muscle then ends at the mastoid process.

We have not tried to segment the lower part of the muscle so we will not provide its appearance in detail. Segmenting the lower part of the muscle is not a trivial task as the muscle divides into two parts. Sometimes instead of appearing as two different parts, the two parts appear as a single structure therefore it was out of the scope of this thesis to segment the lower part.

The middle part is generally elliptical as can be seen in figure 2.9 but in some datasets its shape resembles a curved rectangle as can be seen in figure 2.10. The muscle has been marked with a green outline. All other shape variations fall within these two categories. The upper part of the muscle which is near the mastoid process is elliptical at one end and thin, elongated and rectangular at the other end as can be seen in figure 2.11. As can be seen from these figures that no one model is general

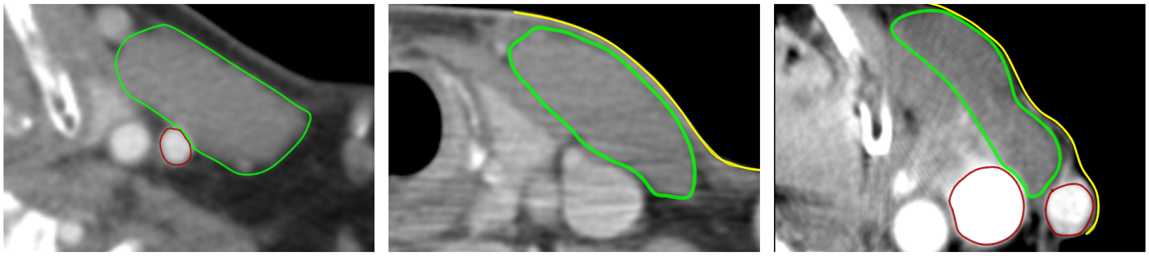

(a) Muscle and Blood Vessel **(b)** Muscle and Skin **(c)** Muscle Skin and Blood Vessel

Figure 2.9: Elliptical shape of the Sternocleidemastoid muscle

enough to encompass all the variations found in the sternocleidomastoid muscle. As will be explained in the chapter 4, three different Stable Mass Spring Models (SMSM) are constructed for the three different muscle shapes. It should be noted that the following description is with respect to the muscle which appears to the right of the user in the CT datasets. The same is true for the muscle which appears to the left of the user in the CT datasets, only that the structures that appear to the left of the right muscle, appears to the right of the left muscle.

The top left portion of the starting point of the middle part of the muscle is adjacent to another muscle as can be seen in figure 2.9a, 2.9b. Both muscles possess the same intensity interval, therefore, it is very difficult to identify the muscle boundaries even with the naked eye. The prior shape knowledge incorporated into the Stable Mass Spring Model (SMSM) will be used to bridge the non existent boundary.

In some datasets the right boundary of the muscle is very close to the skin, so close that at some places the muscle and the skin touch each other as can be seen in figure 2.9c, while in other datasets there is only a very weak edge between the muscle and the skin as can be seen in figure 2.9b. The boundary of the skin generates a very high gradient strength which contributes in making the model unstable. The muscle boundary in figures 2.9b, 2.9c is shown in green while the skin has been outlined in yellow.

Blood vessels are situated at the left boundary of the muscle in the middle (see figure 2.10). Blood vessels have higher intensity values than the muscle and they also generate high gradient strength which also contributes in making the model unstable. The muscle being surrounded by the blood vessels on the left and skin on the right ensures that SMSM can not be placed outside the muscle boundary. Therefore, it is imperative that the SMSM must always be placed inside the muscle during initialization.

If the right muscle boundary is far away from the skin then a soft tissue is present between the skin and the muscle boundary as can be seen in figure 2.12. The muscle has been marked with a green border while the soft tissue has been marked with

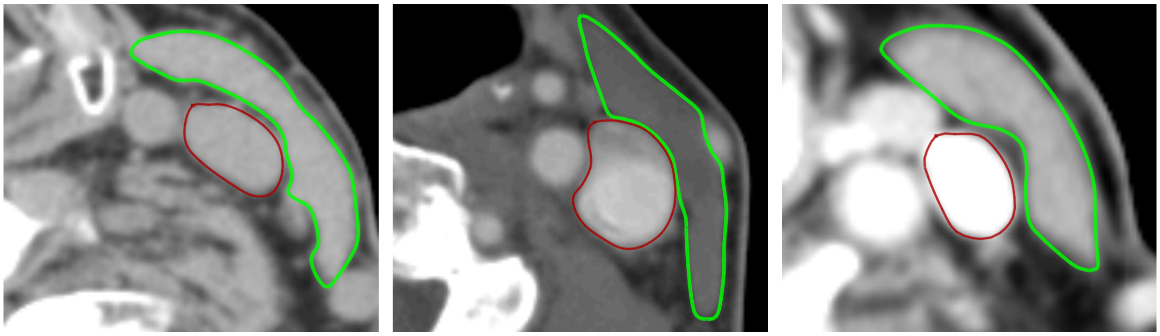


Figure 2.10: Curved rectangular shape of the Sternocleidemastoid muscle

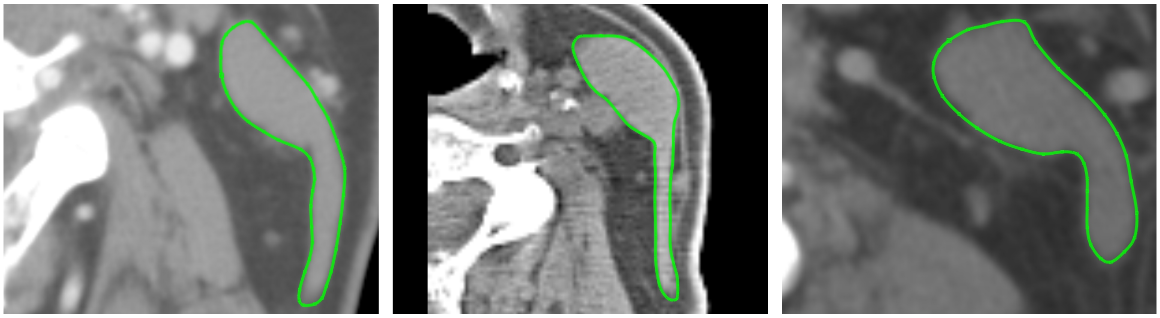
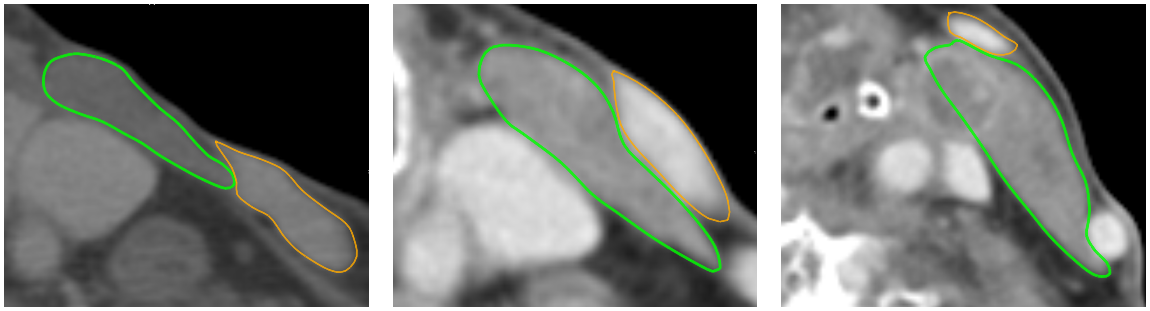


Figure 2.11: Thin Elongated Shape of the Sternocleidemastoid Muscle



(a) Soft tissue situated near the lower right boundary of muscle

(b) Soft tissue situated near the middle of muscle

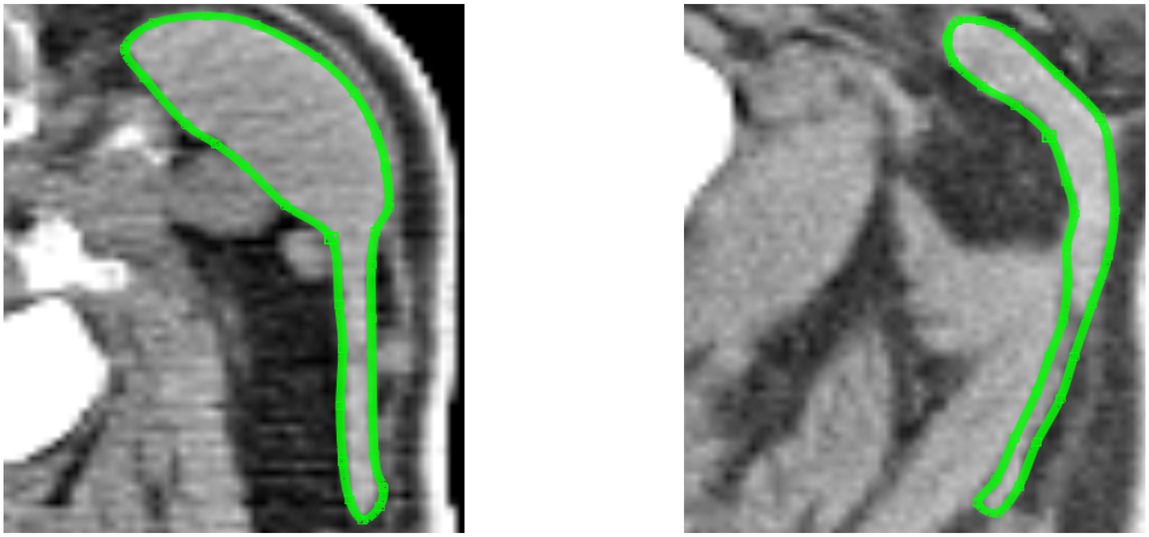
(c) Soft tissue situated near the upper right boundary of muscle

Figure 2.12: Soft Tissue Movement Between Slices

an orange border. Moving from the sternum and the clavicle towards the mastoid process the soft tissue appears at the lower right boundary of the muscle (see figure 2.12a). As you move further towards the mastoid process, the soft tissue moves upward reaching the middle of the muscle (see figure 2.12b). It moves still further upwards terminating at the upper right boundary of the muscle (see figure 2.12c). This movement of the soft tissue can create problem for the 3D SMSM. This soft tissue too possesses higher intensity value than the muscle.

In some datasets the muscle is tumor affected as can be seen in figure 2.12c. The tumor normally affects the upper left portion of the muscle and is not limited to the muscle boundaries but spreads out into the neighboring structures, destroying the distinct muscle boundary if any. So the segmentation task is also to bridge these indistinct boundaries. As shall be explained in chapter 6 that these indistinct boundaries have been bridged using shape knowledge that is incorporated into the 3D Elliptical SMSM.

As said earlier that the part of the muscle near the mastoid process is elliptical at one end and thin, elongated and rectangular at the other end. The elliptical part of the muscle is wider near the middle of the neck as shown in figure 2.13a, than it is near the mastoid process as can be seen in figure 2.13b.



(a) Wider part of the muscle **(b)** Thinner part of the muscle

Figure 2.13: Variation in the width of the muscle near the mastoid process.

2.4 Summary

This chapter is intended to provided background information related to the anatomical structure Sternocleidomastoid muscle which needs to be segmented from the CT dataset. At first the anatomical information about the muscle shape, its originating and terminating locations and its size is provided so that the reader gets familiar with the specific muscle that will be discussed in the rest of the thesis. It was also important that the reader gets some basic information about the CT technology which is the prime source of data from which the muscle will be segmented. Some artifacts that could arise in the CT datasets are presented. These artifacts create problems for the segmentation process and degrades the segmentation result. At the end the appearance of the muscle in the CT datasets is given. The primary focus is on presenting the difficulties that will be encountered during the segmentation process and how these difficulties helped shaped the technique developed.

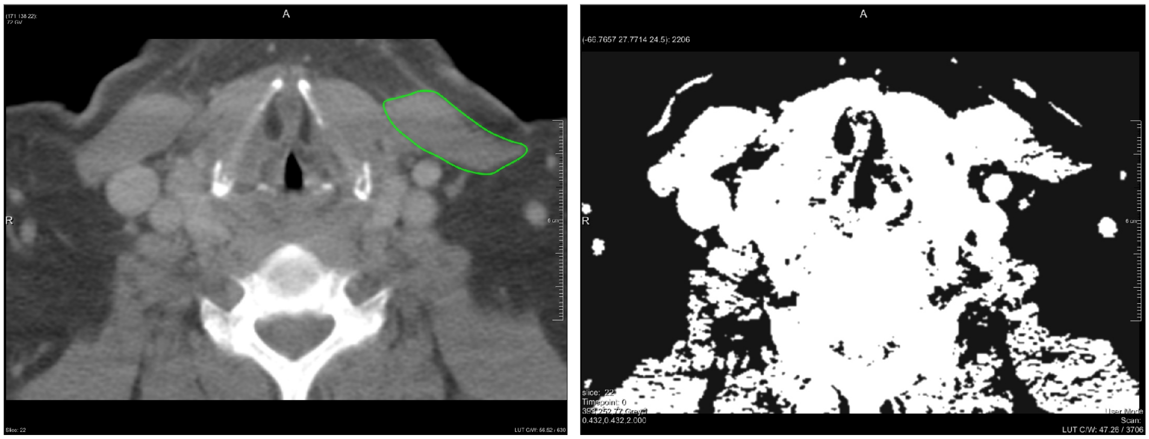
Chapter 3

State of the Art

In this chapter an overview of the state of the art is put forward. A comparison of two types of segmentation techniques namely image based segmentation techniques (see section 3.1) and model based segmentation techniques (see section 3.2) have been presented. Image based segmentation techniques are those that use only local image information and do not have any knowledge about the shape and size of the target structure [35]. On the other hand, model based segmentation techniques are those that combine the shape and the size information of the target structure with the local image information for segmenting the target structure [35]. The Stable Mass Spring Model (SMSM) which belongs to the class of model based techniques is described in section 3.2.4. The SMSM will be used for segmenting the sternocleidomastoid muscle. Section 3.4 displays some areas where SMSM has been successfully used in segmenting anatomical structures from medical imaging datasets.

3.1 Image Based Segmentation

Traditional image based segmentation techniques depend only upon the image data and lack shape, size and location information that makes them vulnerable when applied to medical image dataset [35]. They use image features like intensity value, edge and neighborhood information for segmenting objects. These techniques are computationally more efficient than the model based techniques. Their drawback is that they require substantial human guidance to accurately segment anatomical structures from CT dataset. As medical image datasets suffer from aliasing artifacts (see section 2.2.1), they can misguide the image based segmentation techniques. Aliasing artifacts also pose problems for model based segmentation techniques but they are better equipped to deal with them as compared to image based segmentation techniques. Image based segmentation techniques like threshold based segmentation and region growing will be briefly discussed in the following paragraphs as they will



(a) Target object outlined in green **(b)** Result of applying Threshold

Figure 3.1: Threshold Based Segmentation

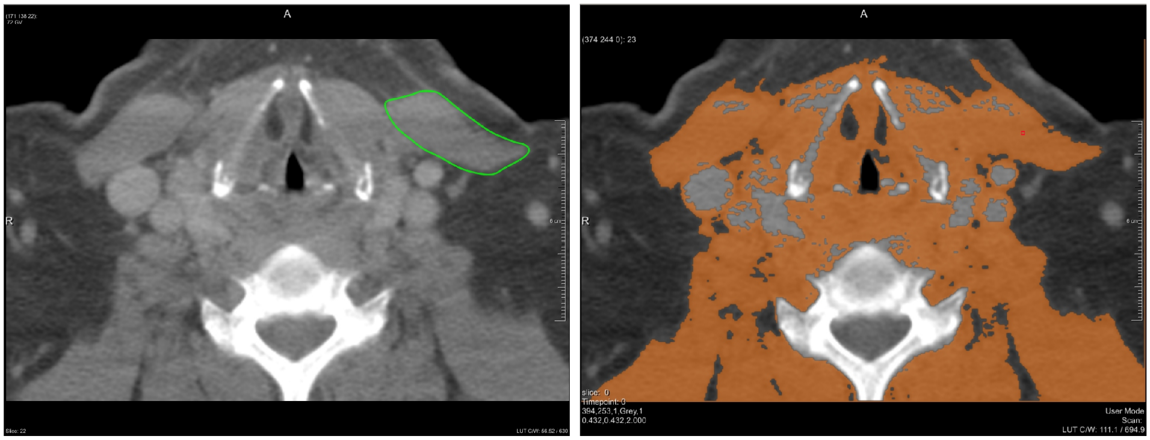
show that only local image information is not enough for segmenting the sternocleidomastoid muscle. It will also highlight why shape and size information of the muscle is so important.

3.1.1 Threshold Based Segmentation

Threshold based segmentation is the easiest of image based segmentation techniques. It is computationally efficient but not accurate especially when dealing with CT datasets. A lower and an upper boundary relating to the intensity interval of the target structure is defined. All voxels having intensity value between the interval are marked as belonging to the target object [12],[38]. Since there is no shape, size and location information, any other object possessing the same interval is also classified as belonging to the object as can be seen in figure 3.1. Figure 3.1a shows the target structure outlined in green. As can be seen from figure 3.1b, the target structure has not been accurately segmented. All the white regions in figure 3.1b have been classified as belonging to the object.

3.1.2 Region Growing

Region growing is similar to the threshold based segmentation technique. In addition to the intensity interval, starting seed point is also defined. The neighborhood of the starting seed point is considered. If the neighborhood point lies inside the interval, its neighbors are considered and so on and so forth [12, 38]. Region growing has the problem of leaking. With the medical image data, the anatomical structures are



(a) Target object outlined in green **(b)** Result of applying Region Growing

Figure 3.2: Region Growing Segmentation

adjacent to each other having similar intensity interval. It is more often than not that the segmentation leaks into neighboring structures as can be seen in figure 3.2 in orange color. Figure 3.2a shows the target structure outlined in green. As can be seen form figure 3.2b that the segmentation has leaked into neighboring structures.

3.2 Model Based Segmentation

The main disadvantage of the image based segmentation techniques was that they did not have any shape and size information of the target structure and work only with local information. This drawback renders them useless for segmenting soft tissues like the sternocleidomastoid muscle or structures whose intensity interval overlaps with other anatomical structures and are adjacent to those structures from medical images. Model based segmentation techniques incorporating shape and the size information of the target object with the local information are much more competitive in such environments. Model based segmentation techniques can be divided into the following classes as classified by Al-Zubi [1].

1. **Structural Models** They are formed by combining different components of the shape which is then used for matching with a database.

2. **Statistical Models** They are formed by obtaining statistical variation of desired shape from training data and then the allowed variation is used to match a target object.

3. **Dynamic Deformable Models** They are governed by dynamic physical laws [5]. These laws are used to deform the model to fit it to the data.

3.2.1 Structural Models

The structural models are formed by joining together different components of the shape, for example, a dog could be defined as having seven different components namely head, central body, tail and four legs. These components could further be subdivided into more simpler units. The structural models are better equipped to represent objects that exhibits large variations in their shapes than the statistical models as shown in figure 3.3. The dog can assume different postures while running walking or even standing. The construction rules define how different components are combined together.

Siddiqui et al. [45] introduced shock grammar. They have defined a robust algorithm for computing shocks using subpixel local detectors and classifiers and global interaction between shocks through shock grammar. The shocks were originally presented by P. D. Lax [31]. The First to the fourth order shock define protrusion, neck, bend and seed respectively. All the shocks together define the skeleton of the shape. A directed acyclic graph called *Shock Graph*[1] is constructed by combining adjacent shocks of same order into a single node. A grammar describes the set of all possible valid combination of shocks. The grammar and topology together also helps in shape recognition.

Sebastian et al. [43] presents a new recognition technique using shock graphs. They compute the similarity between two shapes by calculating the cost of least action path that deforms one shape to another. First they partition the shape space into equivalence classes of shapes and deformations thus effectively reducing its dimensionality. Then using graph edit distance algorithm, an optimal deformation path between two shapes is searched in polynomial time.

Lei He et al. [27] used graphs for object recognition and recovery. A model of an object is constructed using the object's skeleton and contour. The skeleton and contour are themselves obtained from the segmented object. Structural and statistical methods are combined to perform skeleton matching. The use of structural and statistical methods in sequential manner reduces the search space. The skeleton provides information about the length, width and location of object's parts. After a skeleton match has been found, landmark segments on the contour are determined. If the landmark segments on the segmented contour differ from those of the model a fine tuning of these segments is done by maximizing a posteriori probability, given the contour segments model and image features.

---

[1] A Shock Graph basically segments a shape into its constituent parts and defines relationship between those parts.

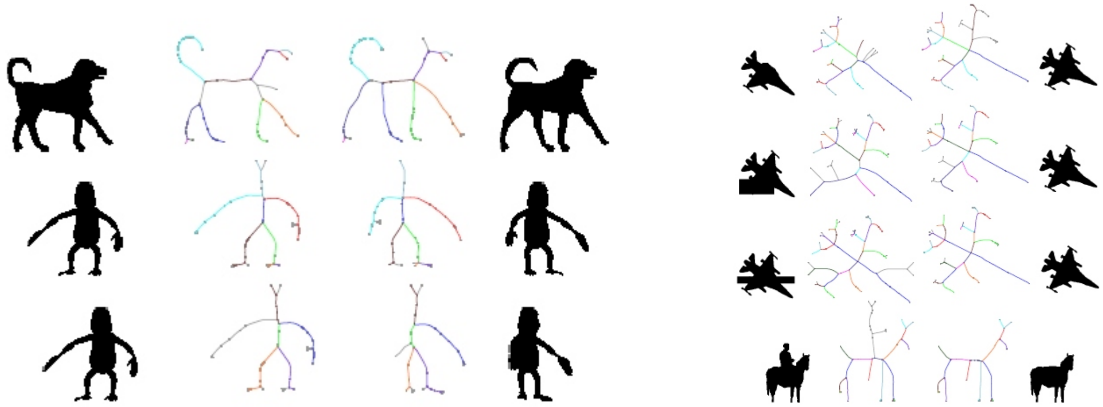


Figure 3.3: Structural Models. Image source is [43]

Flexible Object Recognition and Modelling System (FORMS) has been presented by Zhu and Yuille [51], to illustrate objects or to identify them from their contours. The shape of an object is constructed from two primitive shapes, worm and circles. They describe two different approaches PCA and Finite Element Methods (FEM) to deform the worms and circles to get a closer match to object contour. The worms and circles depicting local contour structure are combined together to form a complete object using grammar rules. While on the other hand, if the task is to identify object, the object's information in terms of worms and circles and how they are linked together is extracted from the image. This information is then compared with the information stored in the databases for object identification. The ability of FORMS to illustrate objects is limited due to the use of only two primitives, worms and circles.

Sebastian and kimia [42] has given a comparison of curve matching and skeleton matching in object recognition. They note that skeleton matching is computationally more expensive as compared to curve matching but it is also more accurate than the curve one, given that the object contains different parts that can assume different arrangements. If the object does not contain parts or parts that can assume different arrangements then the curve matching is more suitable because of its lower computational time and having almost equal accuracy to skeleton matching.

Prior to the work of Stephan Al-Zubi and Klaus Tönnies [2], the structural models had no prior knowledge about the target object's shape. *The Active Shape Structural Models (ASSM)* defined by Al-Zubi et al. describes the shape both at the structural and statistical level. These models combine the strong points of both the structural and statistical models by empowering statistical models to address structural variability and structural models to possess a priori knowledge about object's shape. Shapes similar in structure are used for statistical measurements. The similarity of the structure helps in reducing the statistical variation of different components of the shape. The statistical variations are also related to the structure of the shape with specific variations occurring in specific structural context.

3.2.2 Statistical Models

Before statistical models , deformable models were allowed to deform to encompass the different variations found in natural objects but there was no limit on how much should the model deform and in what ways should it deform , opening the door for the models to deform in ways that are not relevant for the class of objects they represent. To overcome this inadequacy of the deformable models, Cootes et al. [8] presented their *Active Shape Models (ASM).*

The main idea behind the technique is not just to capture the shape of the target object but also the variations that are found in the class of the target object as shown in figure 3.4. A sample set of the target object possessing different variations found in the class is used for training. The shape of the object's contour is defined through a set of points that are defined on each training sample at landmark positions. These points are then processed to remove any affine variations(translation, rotation, scaling), so that the equivalent points on different samples could be compared. These points define the *Point Distribution Model (PDM).* Mean shape is calculated from the shape vectors of the training samples. The contour points of the training samples define the shape vectors.

PCA is applied on the combined set of shape vectors to capture the variations in the training samples. Most of the variations found in the training samples can be captured by considering only the first few largest eigenvalues and corresponding eigenvectors. Any shape in the training set can be obtained by adding a linear combination of the first few selected eigenvectors to the mean shape.

To segment the object, a deformable model created by a rough estimate of the target object's shape is placed inside the image. A refinement in the model occurs through an iterative process. The initial rough estimate need not be very close to the target object [8]. At each iteration point's new location under the global constraints is calculated with respect to the image data. The global constraints ensure that the model deformation is in accordance with the training set. The iterative process terminates once the deformation falls under a certain threshold.

Cootes et al. in [7] suggests multi resolution searching process to improve the efficiency and robustness of the model. The initial searching is carried out at coarser level as it is fast and less prone to get stuck in local minima. Clues from the coarser level are then used in finer resolutions.

Active Appearance Models (AAM) presented by Cootes et al. [10, 7] combine the shape and the intensity information of the target object and then try to match the model to image data. Just like the ASM [8] described above, AAM also learn the

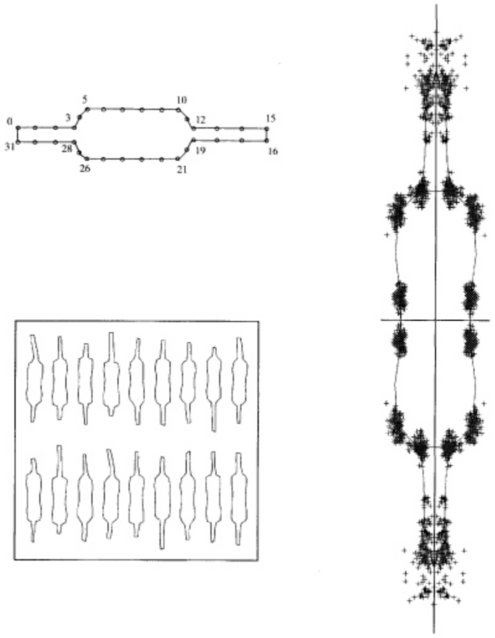


Figure 3.4: Active Shape Model. Image source is [8]

.

shape and intensity variations from a training set. A shape model containing shape variations is constructed in a similar way as in ASM. The intensity information inside the contour defined by the user is considered. This intensity information is normalized and PCA is applied to extract the important modes of variations. A further PCA is applied on the combined vector of shape and intensity information, to remove any correlation between the shape and intensity variations. This results in linear models for shape and intensity which can then be applied to shape fitting. A comparison of ASM and AAM models has been given in [9].

3.2.3 Deformable Models

The name *Deformable Model* has been taken from elasticity theory. In elasticity theory force under certain constraints is applied on elastic bodies which produce deformation in them [35]. Deformable models are a combination of geometry, physics and approximation theory. The target objects shape is defined using geometry. All kinds of shapes could be defined using triangulation or splines. Physical laws are used to define how the model's shape changes in spatial and temporal dimensions. Since the geometrically defined shape is allowed to change under physical laws, it helps in encompassing shape and size variations that are normally found in natural objects and anatomical structures. The shape degeneration is also avoided by the constraints imposed by the physical laws such as torsion force in the case of SMSM [15] and inflation force in the case of snakes [6], which ensures that the model parts does not evolve independently. The fit of the model to the data is defined by approximation theory.

Deformable models are also called *Dynamic Deformable Models* as they are governed by real world dynamic behaviour. These models were introduced by Terzopoulos and Fleischer [49] to image processing in 1988. The mass spring model used in this research work fall in the category of dynamic deformable models.

Generally there are two types of forces, internal and external that govern the deformation of dynamic deformable models in spatial and temporal dimensions. If there are no external forces the body is in a state of equilibrium or rest position. The internal forces keep the body stable. However, when external forces are applied the body deforms. The more the body deforms, the more the internal forces increase. These internal forces act as restoring force, trying to restore the body back to its stable state. The difference between the internal and the external forces keeps the body in motion. As the body deforms more and more, the internal forces increase until they are equal to the external forces. At this point the deformation stops as the internal and the external forces are in a state of equilibrium with respect to each other. The internal forces could be the smoothness, symmetry or the elastic forces, while the external forces could be defined in terms of image data, like the edge strength or intensity levels.

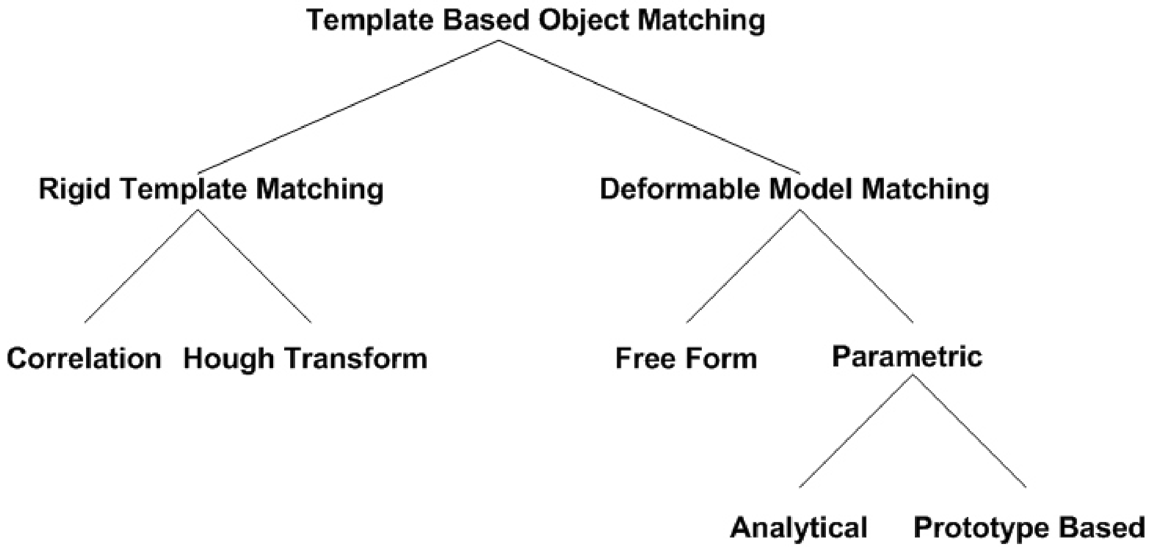


Figure 3.5: An overview of the template matching techniques

Kass et al. in 1988 [30] presented the first deformable model called *The Active Contours* or *Snakes*. Figure 3.5 gives a classification of object matching techniques. The classification has been done by A. K. Jain et al. [28]. Object matching techniques are divided in to two types, *Rigid Template Matching* and *Deformable Model Matching*. Only *Deformable Model Matching* techniques will be covered in this thesis. Deformable model matching techniques can be sub-divided into *Free Form* and *Parametric* based techniques. *Parametric* techniques can themselves be sub-divided into *Analytical* and *Prototype Based* techniques. All these *Deformable Model Matching* techniques are discussed in the following paragraphs.

Free Form Deformable Models

They are also called *Active Contour Models*. Any arbitrary shape can be represented by these free form models. They only have to satisfy the continuity and smoothness constraints. These models were first introduced by Kass et al. [30].

A snake is defined as a spline function (see figure 3.6). This function represents the energy of the contour. The minimum of this function in the image space represents the shape of the object. The energy function Eq. 3.1 given by Kass et al. [30] is :

$$E^*_{snake} = \int_0^1 \left(E_{int}(v(s)) + E_{image}(v(s)) + E_{con}(v(s)) \right) ds \tag{3.1}$$

where,

E_{int} = Internal Energy.

E_{image} = Image Energy or Force.

E_{con} = External Constraint Forces defined by user. (optional).

There are three types of forces that act on the snake as given by Eq. 3.1.

1. **Internal Forces** These forces represent the stretchiness and the smoothness of the contour.

2. **Image Forces** These forces are exerted on the snake by the image data. They could either be exerted from the bright or the dark region in the image or from the gradient information contained in the image.

3. **External Constraints** These forces are generated from the user defined points. They are mainly used to guide the snake towards the correct object boundaries. The points defined by the user could either generate attractive force that forces the snake to move towards them as shown in the figure 3.6 by springs or they could generate repulsive forces that force the snake to move away from them

as shown in the figure 3.6 by a volcano. These points also help in pushing the snake out of the local minima.

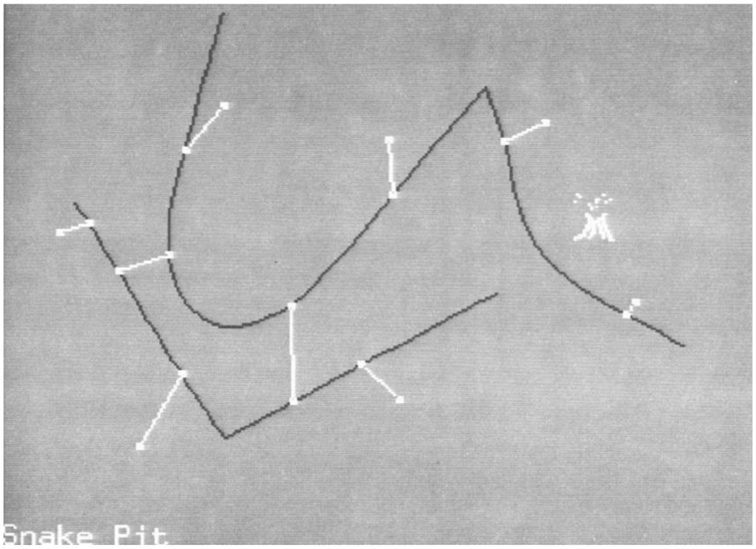


Figure 3.6: Active Contour Model. Image source is [30]

.

The snake when initialized in the image tries to minimize the energy between the different forces. The external image forces deform the snake. As a result of this deformation the internal forces increase, which try to restore the original shape. The deformation stops when an equilibrium is reached between the different types of forces. The snakes are sensitive to initialization point. A randomly initialized snake will fail to converge to the true boundaries of the target object, a common drawback of all deformable models. Also as the snakes are *myopic*, depending only on the local image information, they are vulnerable to local minima in the image. Many different variations of the snake have been proposed to avoid these problems which are briefly discussed below.

Cohen [6] presented the idea of the inflated balloons to solve some of the problems of snakes. He proposed that the snake be allowed to expand like a balloon, in which air is being pumped. So if the snake is initialized inside the boundaries of the target object, it does not matter if the target object boundaries are closed. The additional force called *inflation force*, expands the snake like a balloon. If the snake encounters a local minima or weak edge, then the the inflation force together with the internal forces pushes the snake out of these weak edges. On the other hand, if the snake encounters a true edge, the edge force (external force) is stronger than the combined inflation and internal forces consequently the curve stops growing. Cohen [6] also proposed the reconstruction of the 3D object by applying the 2D snake model on a series of successive slices and then merging together the extracted object contours from each slice.

Neuenschwander et al. [37] proposed that the user should specify only two extreme points of the target object's contour instead of complete polygonal approximation. The snake's energy is minimized the same way as described earlier. Here the only difference is that the target object's edge information travels from the end points of the snake towards its center instead of applying the complete edge information on the whole snake in one go. This results in a better convergence towards the target contour and also helps in reducing the chances of getting stuck in undesirable local minima.

Fua and Berchbhler [21] proposed to apply hard constraints in the form of attractor and tangent constraints on the snake. Their results show a better convergence towards the target contour, without increasing the computational cost. The attractor points attracts the snake towards them or make them pass through them just like in [30], while the tangent constraint forces the snake to be tangent between a certain line segment on the contour of the object.

Ronfard [41] suggested a new snake optimizing technique instead of the traditional cost function optimization. He defines a region based approach which relies on the statistical distribution of the object and background regions. A region which if identified as belonging to the object, pushes the snake out using centrifugal force. On the other hand, a snake is pulled inside through centripetal force if a region is identified as belonging to the background.

Parametric Deformable Models

Parametric deformable models are used in situations when some information about the shape of the target object is available and this information can be represented by few parameters [28]. The shape information is used to define a global shape model which helps in bridging gaps in the boundary. The parameterization of the model gives a concise description of the target shape.

The parametric deformable models can be subdivided into two types depending on how the shape information is represented. If the shape information is represented by a parametric formula then such models are called as *Analytical Deformable Templates* and if the shape information is represented by a prototype shape then they are called *Prototype Based Deformable Templates*. Figure 3.5 shows the classification of the *Parametric Deformable Models*. Both these parametric deformable models are discussed in the following sections.

In these parametric models, the template deforms dynamically by changing the parameters when it interacts with the image forces. The template deforms as long as there is an imbalance between the internal and external forces. The internal forces are those that help define the template while the image forces are external forces.

Analytical Deformable Templates

Shapes are defined through mathematical functions. Shape variations are obtained by changing the parameters of the function within certain bounds. The drawback with analytical deformable templates is that the shape should be defined through a mathematical function. It puts a lot of constraints on the number of shapes that could be defined through analytical deformable templates, as observed by A.K. Jain et al. [28]. He also observes that they are not insensitive to random initialization. It is also important for these models that a good estimate of the size and the orientation of the target structure is available beforehand as they are used in the mathematical description.

A.L. Yuille et al. [50] extracted facial features e.g eye using parameterized templates. A template of feature of interest(e.g. eye, mouth etc.) is created, incorporating the priori knowledge about the shape and size through adjustable parameters. The template of the eye contains a circle with radius r to represent the iris and the central region of the eye and two parabolas representing the upper and lower boundaries of the eye. In all there are 11 parameters attached to the circle and the parabolas. A rough estimate of the initial parameters of the template is obtained by preprocessing. The template is then initialized inside the image. Different kinds of forces representing different details and structures of the feature of interest, act on the template. These forces could be generated from the salient image features e.g. peaks, valleys in the image intensity, edges and intensity of the image. Under the influence of these forces the template deforms and tries to fit to the data. A solution is found, when the minimum of the energy function defined in terms of the forces is reached. Steepest descent in the parameter space is used to update the parameters which is computationally fast and robust. The final value of the parameters give the shape and size of the feature of interest. The method is robust with respect to scale, tilt, rotation of head and lighting conditions.

One example of analytical deformable templates is of super quadric presented by Bardinet et al. [3]. Here the 3D surface is defined through a parametric model. Points are uniformly distributed over the surface of super quadric (super ellipsoid, super hyperboloid etc.) using the implicit equation. An optimal fit of the points to data is achieved through least square minimization. Since the shapes that could be described by a parametric model is limited, they are unable to represent complex anatomical structures, therefore, to define the anatomical shape intricacies a global volumetric deformation called *Free Form Deformation (FFD)* is used. FFD models deform solid objects in a free form manner using small number of parameters. Even after the deformation, the target object can still be represented by the parametric model. The super quadric models has the advantage that they require only a small number of parameters to describe a 3D surface and are robust with respect to sparse data. On the other hand, the parametric nature of the model limits them to surfaces that could be roughly estimated through mathematical definitions.

Prototype Based Deformable Templates

As the name implies, the model is defined through a standard or a prototype shape which provides the most likely or average shape of the class of the target structure. These prototype shapes then deform to describe a specific shape. The amount of deformation is controlled through a set of parameters. In contrast to analytical deformable template, prototype based models are not constrained by the number of shapes they can represent.

Staib et al. [46] defines parametric models of the boundary of the object using the boundary elliptic fourier descriptors. Fourier coefficients define the parameters of the model. The model is biased towards a particular shape by defining a probability distribution on the fourier coefficients. The deformation in the parametric model is achieved by changing the fourier coefficients which obey the probability distribution. To control the allowed shape variations, a priori knowledge about shape variability is obtained through the training set. Using this priori knowledge, the spread of the distribution is controlled thus limiting the shape variations to only valid shapes. A correlation between the model and boundary information is used to create a likelihood function. A bayesian decision based on the likelihood function is used to get an estimate of the boundary.

3.2.4 Stable Mass Spring Model

This section is mainly a summary of the paper written by L. Dornheim [15] where he presents his mass spring model. Here a brief overview of the SMSM and its dynamics are presented. The Stable Mass Spring Model can be classified as belonging to *Prototype Based Deformable Templates* in the above classification of the *Deformable Model Matching.*

Stable dynamic 3D shape models defined in [15] are based on physical mass spring models. They are dynamic models in which masses move under the influence of external and internal forces. Masses are interconnected by elastic springs as shown in figure 3.7. Spring and torque forces represent the internal forces. External forces are exerted through sensors attached to the masses. Depending on the sensor type, the sensor could either be sensitive to edge information or intensity information of the image data. The simple mass spring model obeys hook's Law given by Eq. 3.2.

$$F_s = -k * x \tag{3.2}$$

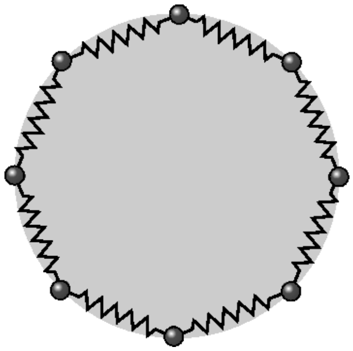


Figure 3.7: Mass Spring Model. Image source is [13]

.

where,
F_s = Force exerted by the spring.
k = Spring Constant(force exerted per length difference)
(Large k for stiff springs, small k for soft springs).
x = Displacement from the equilibrium position.

The stable dynamic 3D shape models are theoretical models that are defined by mass points having masses m_i, rest lengths $l_{0i,j}$ and spring constants $k_{i,j}$. The topology, shape and size of the models play an important role in the model simulation and adaptation to the data. Figure 3.12 shows three different 3D SMSMs having different topology and shape, which is important for segmenting different structures. The velocity of a mass point m_i depends on the internal and external forces acting on it. If only the spring forces $F_{i,j}$ are present the velocity of the mass point m_i is given by the motion equation in Eq. 3.3.

$$v_f = v_i + a * t \tag{3.3}$$

Since

$$a = F/m \tag{3.4}$$

where,

$$F_{i,j} = k_{i,j} * (||\overrightarrow{s_j} - \overrightarrow{s_i}|| - l_{0i,j}) * \left(\frac{\overrightarrow{s_j} - \overrightarrow{s_i}}{||\overrightarrow{s_j} - \overrightarrow{s_i}||} \right) \tag{3.5}$$

where,
$\overrightarrow{s_i}$ and $\overrightarrow{s_j}$ represents the position vectors that are spatially associated with the mass points i and j.

$\overrightarrow{F_{i,j}}$ in Eq. 3.5 represents the spring forces of the elastically deformed springs acting on the mass points m_i. The magnitude of the force depends on the position of incident sensors i and j. The velocity of the mass point m_i at time $t + \Delta t$ starting at time t is given by Eq. 3.6.

$$\overrightarrow{v}_{i_{t+\Delta t}} = \overrightarrow{v}_{i_t} + \left(\frac{\sum_j \overrightarrow{F}_{i,j}}{m_i} \right) * \Delta t \tag{3.6}$$

The term $\sum_j \overrightarrow{F}_{i,j}$ gives the sum of all spring forces j acting on mass point m_i. To reach equilibrium, a damping factor will be added to Eq. 3.6. Damping of the speed only occurs when the masses are in motion. The spring forces represent the size of the model. As mentioned earlier external forces $\overrightarrow{F}_i$ represent the edge and the intensity information contained in the image data. These forces act on the mass point m_i through sensors attached to them. External force component $\overrightarrow{F}_i$ will also be incorporated in Eq. 3.6.

The type of the sensor (intensity or gradient) attached to the mass point is important for a successful model adaptation. To create a balance or give importance to spring or sensor forces respective weight coefficients will be added to Eq. 3.6. The resulting equation is shown by Eq. 3.7.

$$\overrightarrow{v}_{i_{t+\Delta t}} = \left(\overrightarrow{v}_{i_t} + \frac{w_f \sum_j \overrightarrow{F}_{i,j} + w_s(i) \overrightarrow{F}_i}{m_i} * \Delta t \right) * (1 - d) \tag{3.7}$$

where,
d = damping factor.
w_f = weight of the spring force component.
$w_s(i)$ = weight of the sensor force component.

When a model simulating through Eq. 3.7 reaches equilibrium, signals the balance between the shape information incorporated into the model and data information in the image. This results in the segmentation of the data through a model based technique.

3D shape models are more susceptible to shape collapse as compared to 2D shape models. Figure 3.8 shows possible shape collapse scenarios for a 2D SMSM. The vulnerability of the 3D shape models increases even more so due to the additional degree of freedom. Some solutions have been proposed to solve the problem of shape collapse in 3D.

First one mentioned by Bergner [5], describes the creation of the mass spring model with a dense mesh of springs. The drawback of this approach is that due to the high density of springs it requires large computational time. The high computational cost renders them useless for real time applications. Second solution is mentioned by Bardinet [3] with respect to implicit models. It uses super quadric that defines

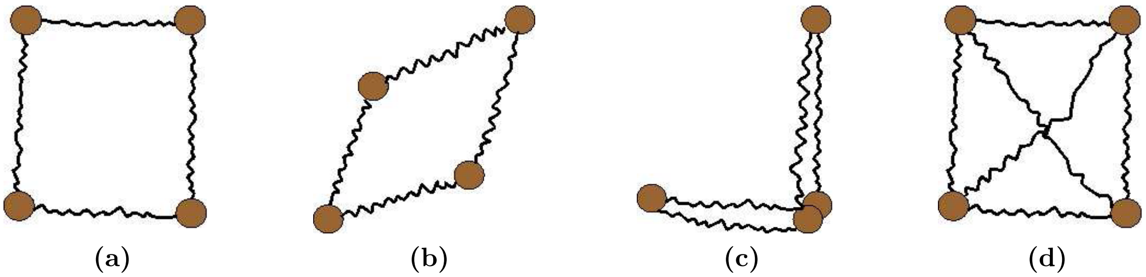


Figure 3.8: SMS Model vulnerability scenarios. (a) A simple square mass spring model. (b) SMS model is twisted without changing the spring lengths. (c) SMS model collapsed without changing spring lengths. (d)Possible solution to avoid SMS model twist or collapse. Image source is [15]

global constraints on the model. As a result they are difficult to control locally, a common drawback of implicit models(see [28]). A third solution is presented by Cohen [6], where additional force component allows the model to expand like an inflated balloon. Their drawback is that there is a trade off between the stability of the model and the loss of shape and size information of the target structure. Simplex Meshes presented by Delingette [11] is a fourth solution. Simplex Meshes are defined as a 3D mass spring surface mesh where masses are allowed to move freely toward the closest data point. They also do not contain the exact shape and size information.

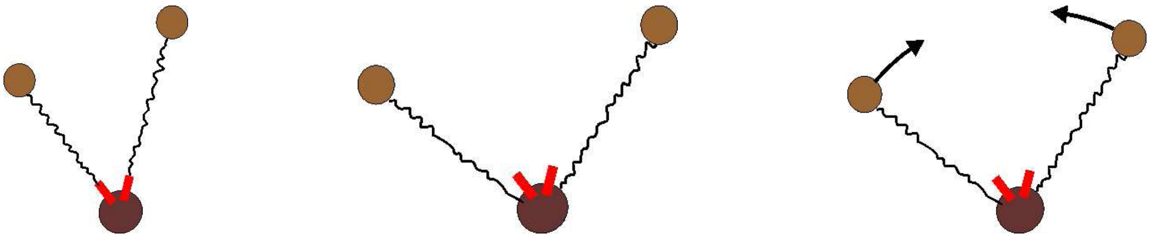
Torsion force has been added to the mass spring model to provide shape stability and to solve the problem of contortion for 3D models [15]. Such models have been successfully used in the segmentation of anatomical structures [19, 14, 13, 32, 26]. Torque also called as moment, measures the tendency of a force to rotate an object about some axis as given by Eq. 3.8.

$$T = r * F \tag{3.8}$$

where,
r = The particles position vector relative to the fulcrum.
F = The force acting on the particle.

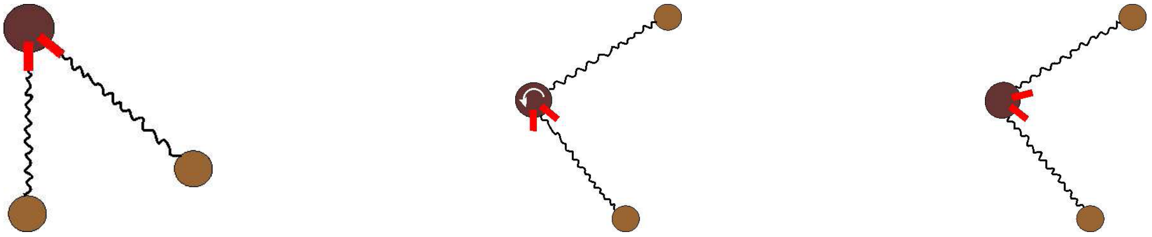
Torque was first defined by Bergner [5] as a solution for the instability of 2D explicit dynamic models. The solution provided only consider angles between two springs with a common mass point as shown in figure 3.9. The solution provided is not applicable for 3D models where the extra dimension allows the angle plane to turn. To solve the problem of the turning of the angle plane, additional angle forces are required which then lead to problems similar to dense cross linking.

L. Dornheim [15] suggests the use of normalized rest directions $r_{i,j}$ of the springs starting from mass point m_i to all its adjacent masses m_j along the lines of the spring



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

Figure 3.9: Torque for 2D SMS Model. (a) Shows springs in their rest direction. (b) Springs are twisted due to external force. (c) Torque acts in the direction of the rest direction to counter the external force. Image source is [15]



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

Figure 3.10: Torque for 3D SMS Model. (a) Shows springs in their rest direction. (b) Springs are twisted due to external force. (c) Mass rotates to counter the external force. Image source is [15]

rest lengths . The twisting of the springs from their rest directions produces torque which is then balanced by twisting the mass which results in model rotation as shown in figure 3.10. The torsion force produce is given by Eq. 3.9.

$$\overrightarrow{F}_{i,j} = t_i * \left(\frac{<(\overrightarrow{r}_{i,j}, \overrightarrow{r}_{0_{i,j}})}{||\overrightarrow{r}_{i,j}||}\right) * \left(\frac{\overrightarrow{n}_{i,j}}{||\overrightarrow{n}_{i,j}||}\right) \tag{3.9}$$

where,
$\overrightarrow{r}_{i,j} = \overrightarrow{s}_j - \overrightarrow{s}_i$
t_i = Torsion constant (Torsion Moment per torsion angle).

The direction of torsion forces $\overrightarrow{n}_{i,j}$ is given by Eq. 3.10. Torsion forces act as internal forces alongside the spring forces. Where spring forces define the size of the model, torsion forces (direction of springs) define the shape of the modelled object. Motion equation defined in Eq. 3.7 has been extended by torsion forces $\overrightarrow{F}_{j,i}$ with a corresponding weighting factor w_t is given in Eq. 3.11. Because of the rest directions of all springs, the problem of mutually shifting angle planes does not arise.

$$\overrightarrow{n}_{i,j} = r_{0_{i,j}} - \left(\frac{< \overrightarrow{r}_{i,j}, \overrightarrow{r}_{0_{i,j}} >}{||\overrightarrow{r}_{i,j}||^2} \right) * \overrightarrow{r}_{i,j} \tag{3.10}$$

$$\overrightarrow{v}_{i_{t+\Delta t}} = \left(\overrightarrow{v}_{i_t} + \frac{w_f \sum_j \overrightarrow{F}_{i,j} + w_t \sum_j \overrightarrow{F}_{j,i} + w_s(i)\overrightarrow{F}_i}{m_i} * \Delta t \right) * (1 - d) \tag{3.11}$$

where,
w_t = weight of the torsion force component.

3.3 State of the Art for Muscle Segmentation

During the survey of the state of the art techniques, we did not find any literature that tried to segment the sternocleidomastoid muscle. There was some literature available that tried to segment the muscle fibers but it was not related to our work.

However, we found one paper where Gilles et al. tried to model and segment the musculoskeletal system from MRI datasets [22]. They have used the scalable discrete deformable surfaces, the simplex meshes on the hip joints and thigh MRI dataset. They define a simplex mesh model. A vertex position in the mesh is uniquely defined by three geometric parameters which are; two barycentric coordinates and curvature. These three parameters are also used for smoothing and calculating shape constraint forces to regularize the segmentation. The 1D registration of the intensity profile with a standard profile obtained from a reference segmentation provides the external force. They apply affine transformation for local smoothing and global regularization which in turn is used to regularize the external force field.

Figure 3.11 shows the segmentation process. Multi-resolution scheme was used to improve the computational speed, robustness and to avoid getting trapped in local solutions. Since muscles and bones are segmented together, they apply topological constraints to model the different parts. Radial constraints were applied to model muscle contractions. To initialize the model, they manually define ten landmark positions. Their model also allows interaction during simulation to guide the simulation toward towards the correct goal. Their results showed that their average distance from the gold standard was 1.25mm for bones and 1.7mm for muscles. The total segmentation process took around 15 min. on a standard modern PC. It can be observed from figure 3.11 that the experimental segmentation environment is simple. There is no other data to distract the model.

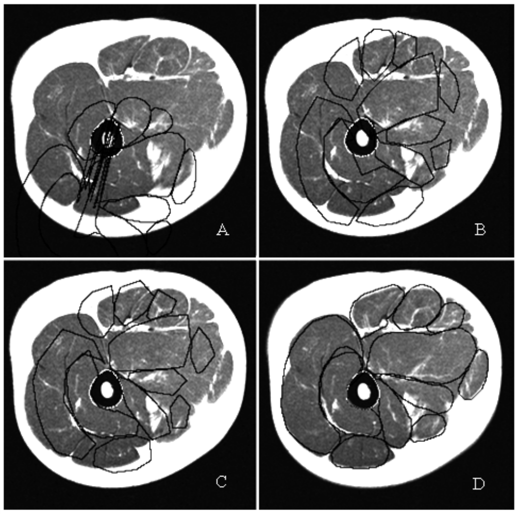


Figure 3.11: Musculoskeletal segmentation process. Image source is [22]

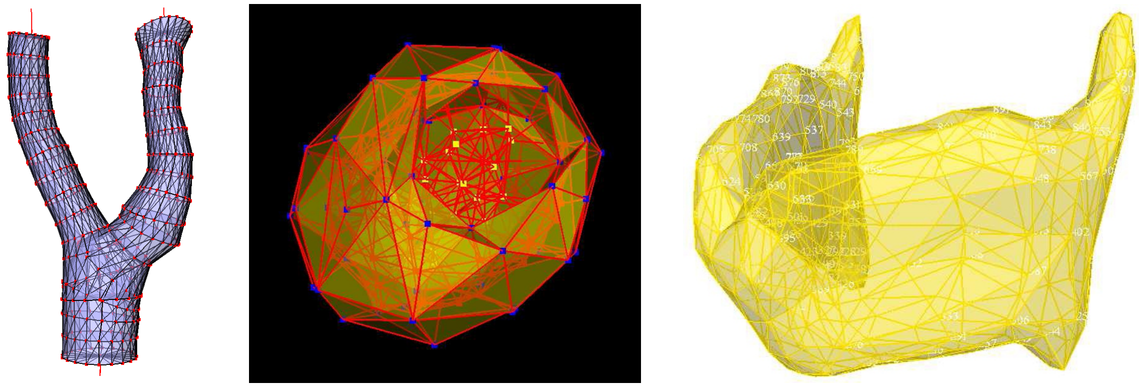
.

3.4 Successful Applications of the SMSM

Mass Spring Models have been used successfully during the last few years, in segmenting anatomical structures from a human body especially from the neck. L. Dornheim et al. [19] has used it successfully in the segmentation of left ventricle in the 3D SPECT data. The shape knowledge is intrinsically defined through a mesh of masses and springs. The shape is defined through internal forces exerted by springs. The masses move under the influence of external forces generated by the image data. Model deformation continues until an equilibrium is reached between internal and external forces. Figure 3.12b shows a 3D SMSM of left ventricle.

The mass spring model depicting the left ventricle shape is constructed automatically from a sample segmentation. L. Dornheim et al. [19] presents two ways to correctly place and orient the model automatically inside the dataset without any human intervention. The shape of the model is matched to find a similar pattern in the dataset using correlation or pattern vector angle. After initializing the model, shape adaptation is carried out by only using external forces from intensity sensors. The estimated shape of the model tries to realign itself using clues provided by the data. After the shape adaptation is completed the final step is to locate the boundaries of left ventricle by activating the edge sensors. The edge sensors should be close enough to the edges to locate them.

J. Dornheim et al. [14] presents another application of stable 3D mass spring model where it is used in the segmentation of lymph nodes in the neck. Here for the first time intensity, gradient and shape information were combined into a single



(a) 3D Blood Vessel Model **(b)** 3D Left Ventricle Model **(c)** 3D Thyroid Cartilage Model

Figure 3.12: Different 3D SMSM Models. Image source : figure 3.12a is taken form [32], figure 3.12b is taken form [17], figure 3.12c is taken form [13]

.

model. The problems faced during the segmentation of lymph nodes were due to the infiltration of lymph nodes into neighboring soft tissues, high contrast structures in neighborhood, internal inhomogeneity due to necrosis and the variable size of the lymph nodes. Considering that the shape of lymph nodes is spherical, a spherical mass spring model is defined. Gradient sensors are attached to the outer layer of the masses. An inner layer of masses parallel to the outer layer at some distance d is defined and intensity sensors are attached to the masses. The springs connecting masses of the two layers are assigned high spring constants to make them stiff. The model was initialized inside the lymph node and then it grows out towards the true contour of the lymph node. The gray value range was determined from the two seed points defined by the user. These seed points are also used to provide an estimate of the lymph node's position and size. After this model adaptation is carried out.

J. Dornheim et al. [13] presents another application of SMSM in the segmentation of thyroid cartilage. Just like in previous work [14], the thyroid cartilage model is semi-automatically constructed from a given manual sample segmentation. An outer surface of the model is defined, attaching directed gradient sensors to the masses. Some mass points are fixed by the user at key landmark position and the model is adapted to the individual patient's anatomy using spring and torsion forces only. After the shape adaptation is completed, gradient sensors are activated and precise model adaptation according to the data is carried out. Figure 3.12c shows a 3D SMSM model of thyroid cartilage.

L. Dornheim et al. [18] used SMSM model for the verification of the presence and correctness of the mounted parts in industrial optical quality control system.

The state of a part is compared to a reference state image and then used to locate the part in the image. The SMSM created are specifically tailored to represent the part's shape characteristics. More masses are placed in areas where curvature changes rapidly as compared to where it does not change rapidly. To every mass point two sensors are attached that look for contour normal direction and curvature information instead of intensity and gradient information. The total number of masses placed on the contour of the object was 40 that was obtained empirically.

3.5 Summary

This chapter gives a survey of the state of the art. The chapter opens with a brief description of the image based segmentation techniques and the reasons for their not being suitable for segmenting anatomical structures from CT dataset. Then broad overview of the model based segmentation techniques have been given. Three different types of model based segmentation techniques; structural , statistical and deformable models have been discussed. A short survey of different techniques falling under each category has been given. Stable Mass Spring Models, a type of deformable model used for segmenting the sternocleidomastoid muscle have been described in-depth and important features of the model essential for segmenting 3D structures have been identified. Then some applications where SMSM have been successfully employed for segmenting 3D structures from medical imaging datasets are presented.

Chapter 4

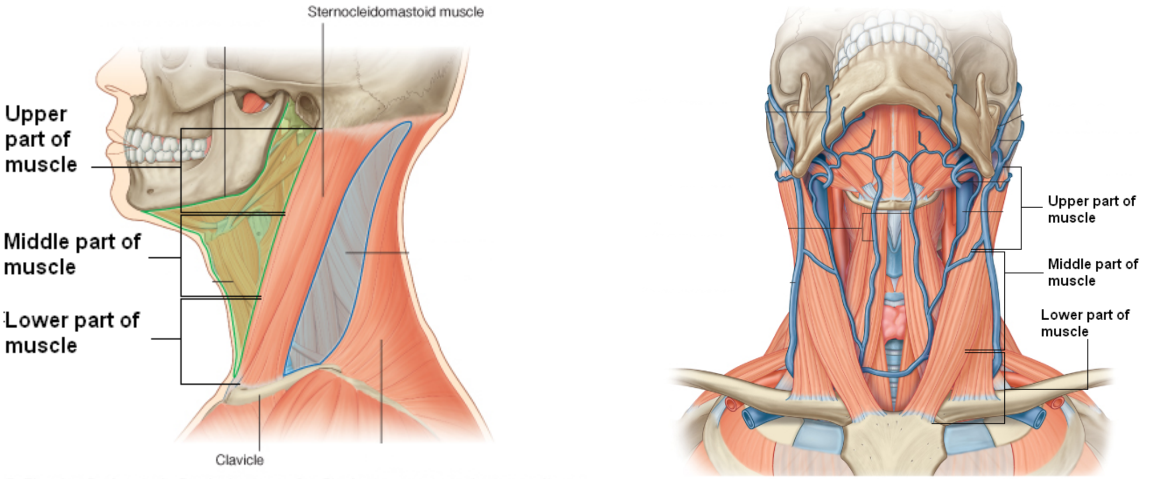
Concept

In this chapter the main technique will be presented. The section 4.1 will provide details about three different types of SMSMs which were constructed and and the reasons behind their creation. Details of the interpolation process and the procedure of estimation of the muscle intensity interval are given in the section 4.2. The details about the interpolation and the intensity estimation has been provided in detail only for the *Elliptical SMSM*. The same procedure holds true for the other two types of SMSMs, therefore explanation about their interpolation and intensity estimation have been eschewed. The focus is only on presenting the minor differences that exist. Then specifications concerning the three SMSMs have been provided in the sections 4.2, 4.3 and 4.4. Section 4.5 discusses the preprocessing details that are applied on the CT datasets. In the end, section 4.6 gives details about the ideas taken from the other applications of the SMSM and how they help shape the technique for segmenting the sternocleidomastoid muscle.

4.1 Different Types of SMSM

In this thesis we have divided the complete muscle in three different parts. These parts are shown in figure 4.1 which are:

1. **Lower Part** : This part starts from the sternum and the clavicle and ends where the two strands of the muscle combine together.
2. **Middle Part** : The middle part of the muscle starts from where the two strands of the muscle combine together and ends at where one end of the elliptical shape starts to change into a thin, elongated and rectangular structure.
3. **Upper Part** : The upper part of the muscle starts from where the middle part ends and terminates when the muscle terminates at the mastoid process.



© Elsevier. Drake et al: Gray's Anatomy for Students - www.studentconsult.com

© Elsevier. Drake et al: Gray's Anatomy for Students - www.studentconsult.com

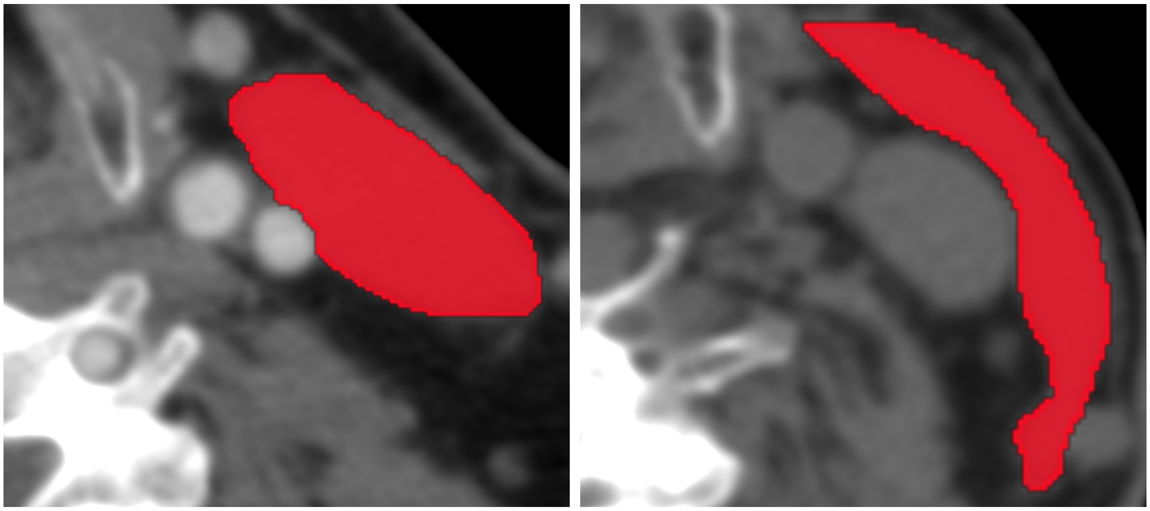
(a) Side view **(b)** Frontal view

Figure 4.1: Different parts of the muscle. Image Source : [40]. The images have been edited.

We have only tried to segment the middle and the upper parts of the muscle. The lower part of the muscle has been not been segmented as a completely different strategy is required to segment the muscle. Sometimes the two muscle strands originating form the sternum and the clavicle are disjoint and at other times they are combined.

The shape of the sternocleidomastoid muscle can be categorized with three different structures. The middle part of the muscle is elliptical as shown in figure 4.2a or curved and rectangular as shown in figure 4.2b. The middle part of the muscle then can be segmented using either the *Elliptical SMSM* if the shape of the muscle is elliptical or *Curved Rectangular SMSM* could be used for segmenting this part of the muscle if the muscle shape is curved and rectangular. Figure 2.9 shows some of the possible elliptical shapes of the muscle which could be segmented using the *Elliptical SMSM*. Figure 2.10 shows some of the possible curved rectangular shapes of the muscle which could be segmented using the *Curved Rectangular SMSM*. The upper part of the muscle which is near the mastoid process and the temporal bone is also elliptical at one end and thin, elongated and rectangular at the other as shown in figure 4.3. If the shape of the muscle is elliptical at one end and thin, elongated and rectangular at the other then the muscle is segmented using the *Elliptical Elongated SMSM*. Figure 2.11 shows some of the possible elliptical, thin, elongated and rectangular shapes of the muscle which could be segmented using the *Elliptical Elongated SMSM*.

The user is required to enter strokes after every 8 or 9 slices. These strokes should try to approximate the medial axis of the shape of the muscle in a particular slice.



(a) Elliptical shape of the muscle **(b)** Curved rectangular shape of the muscle

Figure 4.2: Shapes of the middle part of the muscle

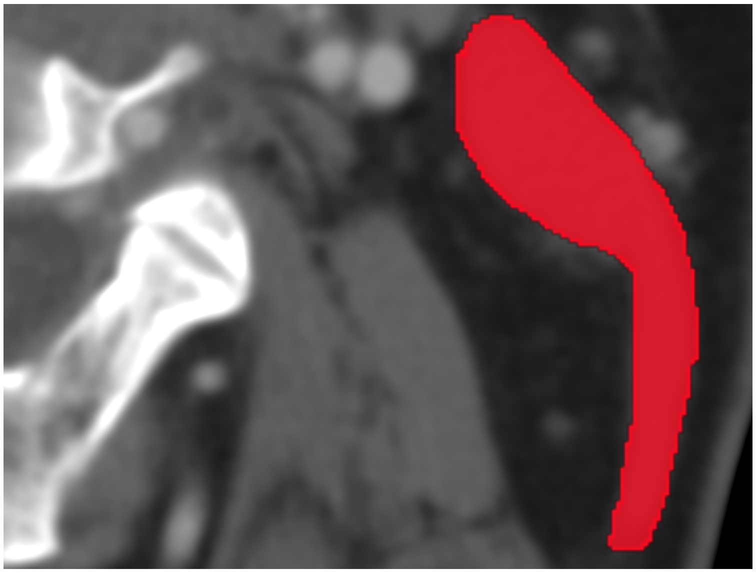


Figure 4.3: Shape of the upper part of the muscle

The position and the cues about the shape of the muscle are then obtained from these user strokes. As explained in section 2.1 that the muscle slants across the side of the neck, therefore, these strokes help in obtaining the position, length and the orientation of the muscle in the intervening slices. The change in position, length and orientation of the muscle could easily be estimated through linear interpolation (see figure 4.5).

While entering the strokes the user specifically defines the starting slice number of the upper part of the muscle (elliptical elongated part) by checking a check box. If the middle part of the muscle is not elliptical but curved and rectangular then the user checks a check box indicating that the middle part of the muscle should be estimated through a curved rectangular model.

Figure 4.4 gives the work flow diagram of the segmentation process. The user enters the strokes from which a skeleton of the muscle is created using interpolation. After the skeleton has been created, it is used for estimation the intensity interval of the muscle. Ray casting is done to get an estimate of the length of the minor axis of the elliptical part of the muscle. Then the ellipse is generated. Points on the ellipse are calculated which are used to generate the SMSM. For rectangular shape of the muscle ray casting is not required to generate the SMSM. After generating the elliptical or the curved rectangular SMSM, simulation is carried out and then the simulated model is used to segment the muscle.

After segmenting the middle part of the muscle either through elliptical or curved rectangular SMSM, the SMSM is destroyed and the process is repeated for the upper part of the muscle. The SMSM for the upper part of the muscle is then used to segment the remaining part of the muscle. The two segmentations are then combined together to form the complete segmented muscle. These steps are discussed in detail in the following sections.

It is not necessary to always use two SMSM models. During the segmentation of the muscle in 14 test datasets some datasets were only segmented using the curved rectangular model. The prototype allows using different combinations like;

1. Using *Elliptical SMSM* to segment middle part of the muscle and then using *Elliptical Elongated SMSM* to segment upper part of the muscle. This combination has been used most often.

2. Using *Curved Rectangular SMSM* to segment middle part of the muscle and then using *Elliptical Elongated SMSM* to segment upper part of the muscle. This combination has not been used on any of the 14 test datasets.

3. Using only the *Elliptical SMSM* to segment the complete muscle. This also has not been used on any of the 14 test datasets.

4. Using only the *Curved Rectangular SMSM* to segment the complete muscle. This has been used on few of the 14 test datasets.

5. Using only the *Elliptical Elongated SMSM* to segment the complete muscle. This has not been used on any of the 14 test datasets.

Any one of the combination could be selected just by selecting two check boxes. It is left up to the user's discretion to select anyone of the combinations provided as the developed prototype can not itself identify which model should be used for muscle segmentation. These five combinations are the natural outcome of the three different SMSMs that were constructed for segmenting different parts of the muscle. Combination 1 and combination 4 are the only two combinations that were used on the 14 test datasets, therefore, experience says that it is most likely that these are the only two combinations that will be used for segmenting the muscle from unseen datasets but then again situations might arise where other combinations could be used.

4.2 Constructing Elliptical SMSM

This section discusses the construction of the *Elliptical SMSM*. *Interpolation*, *Slice Selection* and *Intensity Estimation* are elaborated in depth with respect to the *Elliptical SMSM*. The *Interpolation*, *Slice Selection* and *Intensity Estimation* concept also holds true for the *Curved Rectangular SMSM* and *Elliptical Elongated SMSM*.

4.2.1 Interpolation

The strokes are used to create the skeleton of the elliptical part of the muscle. Let N be the total number of strokes entered by the user and let M be the total number of slices between the strokes. The starting point P_{start_i} and ending point P_{end_i} of the first stroke is obtained. Similarly, the starting point $P_{start_{i+1}}$ and ending point $P_{end_{i+1}}$ of the second stroke is also obtained. Using these four points the starting and the ending points of the intervening slices between the strokes are obtained through linear interpolation using equation 4.1.

Figure 4.5 shows the linear interpolation. The top most and the bottom most slices are the slices in whihc user has entered the strokes. The two slices between the top most and the bottom most slices are the interpolated slices. Linear interpolation is done between starting points of the two strokes and between the ending points of the two strokes. The intersection of this linear interpolation with the two slices in between gives the interpolated points in those slices.

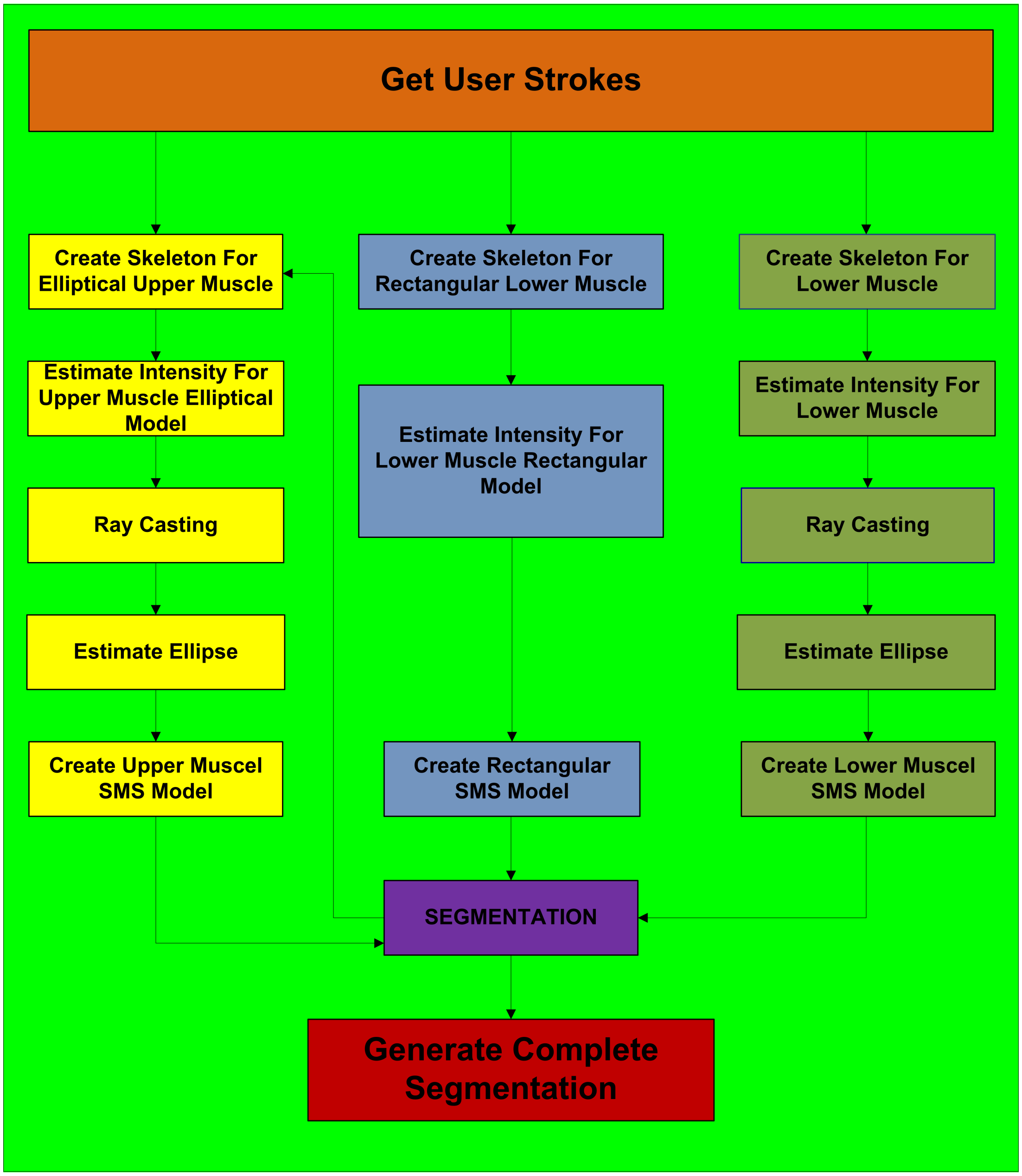

Get User Strokes
Create Skeleton For Elliptical Upper Muscle
Create Skeleton For Rectangular Lower Muscle
Create Skeleton For Lower Muscle
Estimate Intensity For Upper Muscle Elliptical Model
Estimate Intensity For Lower Muscle Rectangular Model
Estimate Intensity For Lower Muscle
Ray Casting
Ray Casting
Estimate Ellipse
Estimate Ellipse
Create Upper Muscel SMS Model
Create Rectangular SMS Model
Create Lower Muscel SMS Model
SEGMENTATION
Generate Complete Segmentation

Figure 4.4: Work flow diagram of the segmentation process.

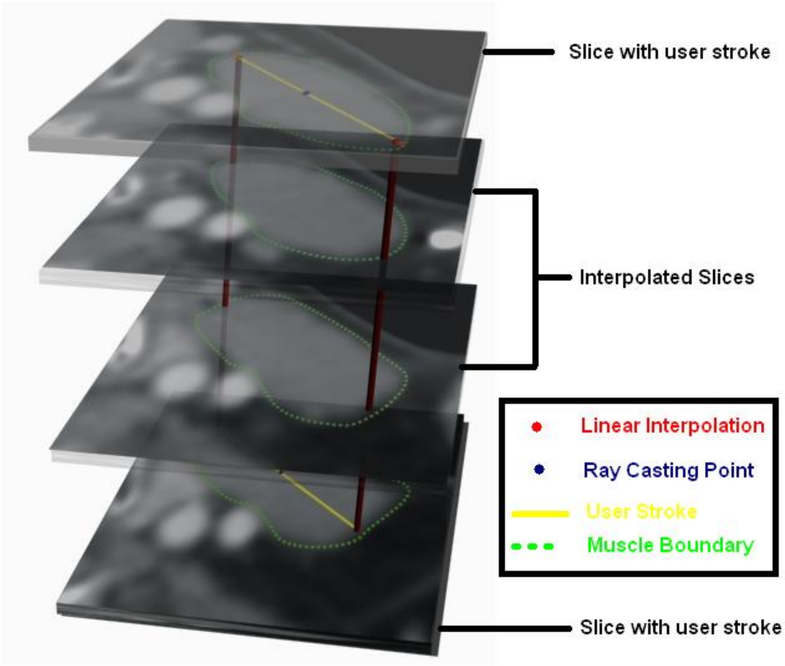


Figure 4.5: Linear Interpolation

$$\begin{aligned} x_j &= (1 - t_j) * x_i + (t_j) * x_{i+1} \\ y_j &= (1 - t_j) * y_i + (t_j) * y_{i+1} \\ z_j &= (1 - t_j) * z_i + (t_j) * z_{i+1} \end{aligned} \tag{4.1}$$

where,
$i = 1 \cdots N - 1$
$j = 1 \cdots M - 1$

The z_j coordinate value of the intervening slice is known and using $z_i = P_{start_i} \rightarrow z$ and $z_{i+1} = P_{start_{i+1}} \rightarrow z$; the value of t_j is calculated using equation 4.2.

$$t_j = \frac{z_j - z_i}{z_{i+1} - z_i} \tag{4.2}$$

where,
$i = 1 \cdots N - 1$
$j = 1 \cdots M - 1$

Using the value of t_j the x and y coordinates of the start and the end points of all the intervening slices are calculated.

4.2.2 Slice Selection

Start point P_{start_i} and end point P_{end_i} of only selected slices between strokes are calculated. An upper limit is defined on the total number of masses $Mass_{total}$ in one layer. There are two layers of masses namely inner layer and outer layer, so the total number of masses for a complete model is 800. The figure 800 has been chosen after experimentation. Having masses more than 800 slows down the simulation process. Another variable $Mass_{perslice}$ defines how many masses will be placed per slice. Number of slices $Slice_{total}$ that encompasses the elliptical shape of the muscle are calculated by subtracting the starting stroke from the ending stroke. The number of slices required $Slice_{required}$ to construct the whole model is calculated using equation 4.3.

$$Slice_{required} = \frac{Mass_{total}}{Mass_{perslice}} \tag{4.3}$$

If the $Slice_{total} \leq Slice_{required}$ then all the slices are used in creating the elliptical model, otherwise only selected slices are used. First of all, all the slices which were used for the strokes $Slice_{totalstrokes}$ are selected. The remaining slices $Slice_{remaining}$ is calculated from equation 4.4.

$$Slice_{remaining} = Slice_{required} - Slice_{totalstrokes} \tag{4.4}$$

These remaining slices $Slice_{remaining}$ are then proportionally distributed between the strokes. To calculate the proportion for the $Slice_{remaining}$ between strokes, the percentage of the slices between strokes $Percentage_{betweenstroke}$ is calculated using equation 4.5.

$$Percentage_{betweenstroke} = \frac{Stroke_{i+1} \rightarrow z - Stroke_i \rightarrow z}{Stroke_n - Stroke_1} \tag{4.5}$$

where,
$i = 1 \cdots N - 1$

$Slices_{betweenstrokes}$ is calculated using equation 4.6

$$Slices_{betweenstrokes_i} = Percentage_{betweenstroke_i} * Slice_{remaining} \tag{4.6}$$

where,
$i = 1 \cdots N - 1$

Since a real number $Slices_{betweenstrokes_i}$ could not be used, therefore, if the decimal part of the $Slices_{betweenstrokes_i}$ is less than 0.5 then the $Slices_{betweenstrokes_i}$ is rounded off towards the lower side and the decimal part is added to the next $Slices_{betweenstrokes_{i+1}}$.

If the decimal part of the $Slices_{betweenstrokes_i}$ is greater than 0.5 then the $Slices_{betweenstrokes_i}$ is rounded off towards the upper side and $(1 - decimalpart)$ is subtracted from the next $Slices_{betweenstrokes_{i+1}}$. This way the $Slice_{remaining}$ gets proportionally distributed between the strokes.

Once the slice count between strokes $Slices_{betweenstrokes}$ have been calculated, these slices are equally distributed between the strokes. It is for these slices only that the x and y coordinate of start P_{start_j} and end P_{end_j} points are calculated.

4.2.3 Estimating the Intensity Interval for the Elliptical Model

The start and end points of the stroke for a particular slice have already been calculated. Using these points a rectangular region of width $Width_{intensityestimation} = 4$ is defined. The length of the rectangle corresponds to the length of the start and end points. All points lying inside the rectangular region are used for the intensity estimation. The procedure is repeated for all the selected slices.

Through observation and experimentation it has been noticed that the muscle intensity interval width is around 120 units for datasets where the muscle intensity is greater than 500. For datasets where the muscle intensity is less than 500 the intensity interval width is around 70. Using this, the upper and the lower limit around $MeanIntensity$ are defined. For datasets where the muscle $MeanIntensity$ value is greater than 500, the upper and lower limits around mean is 60 and for datasets where the muscle $MeanIntensity$ is less than 500, the upper and lower limit is 35. This is used to filter out the pixels extracted from rectangular regions, with intensity outside the range.

During the first iteration all pixel values are used and a $MeanIntensity$ is calculated using equation 4.7. Any intensity value lying outside the interval ($MeanIntensity-60$, $MeanIntensity+60$) are filtered out and a new $MeanIntensity$ is calculated. The process is repeated until the $MeanIntensity$ value does not change anymore. Once the $MeanIntensity$ stabilizes, $StandardDeviation$ of the intensity values is calculated using equation 4.8 and the final intensity interval is defined $(\overline{x} - 3 * \sigma, \overline{x} + 3 * \sigma)$. A standard deviation of 3 has been used as it encompasses 99.6% of the values.

$$\overline{x} = \frac{x_1 + x_2 + \cdots + x_n}{n} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{4.7}$$

where,
n = total number of pixels extracted from the rectangular region lying within the intensity interval

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^2} \tag{4.8}$$

where,
n = total number of pixels extracted from the rectangular region lying within the intensity interval

4.2.4 Estimating the Ellipse

To create an elliptical 3D model, the length of major and minor axis and the orientation of the ellipse needs to be estimated. To estimate these parameters ray casting is used. For the elliptical model rays are casted from the midpoint of the start P_{start} and end P_{end} points of the stroke and for the intervening slices between strokes from the mid-point of the interpolated start and end points as shown in figure 4.6. 360 rays are casted out from the mid point corresponding to each degree between 0 - 359. A ray terminates when it hits a pixel whose intensity lies outside the estimated intensity interval. Figure 4.7 shows the result of ray casting in one of the slices. The process is repeated for all the selected slices.

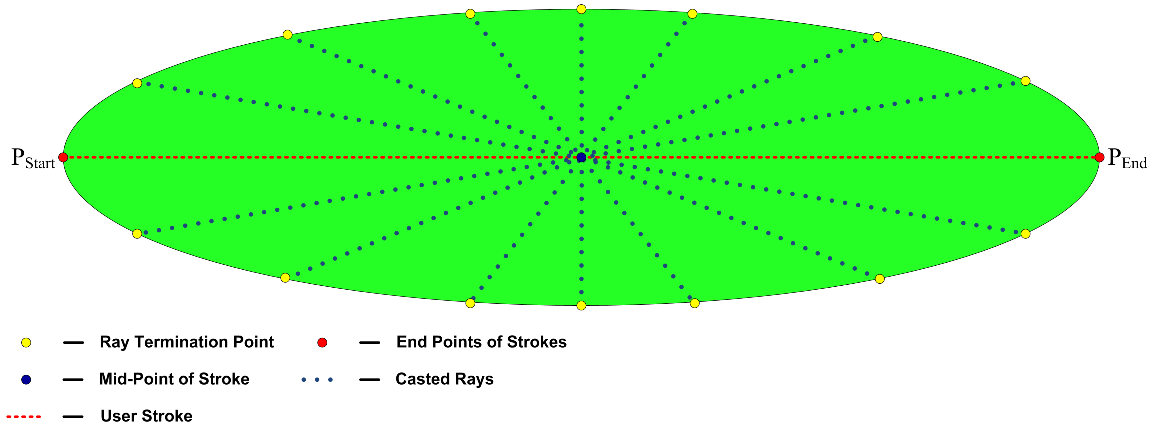


Figure 4.6: Schematic diagram for ray casting.

The orientation of the ellipse is obtained from the orientation of the line joining the start P_{start} and end P_{end} points . The length is also estimated from the start and the end points. The length of the minor axis is mainly estimated from the data obtained through ray casting. Principal Component Analysis[1] (PCA) [29] is applied on the

---

[1] A good tutorial on PCA is available at

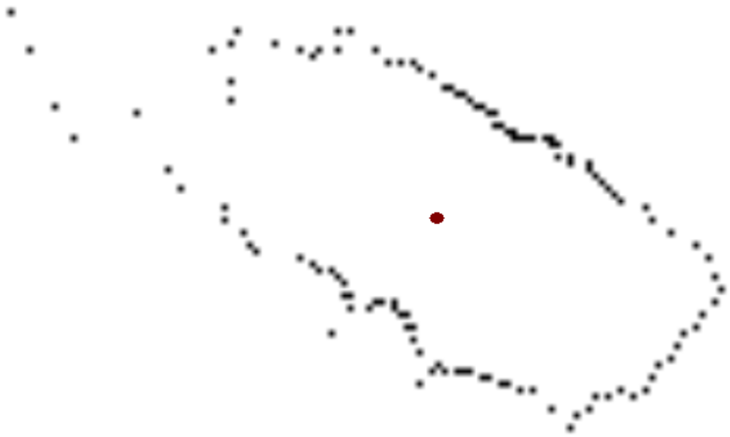


Figure 4.7: Ray casting in a slice.

ray casted data and its eigen vectors are calculated. The eigen vector corresponding to lower eigen value corresponds to the length of the minor axis of the ellipse. The length of the minor axis $Length_{minoraxis}$ is given by equation 4.9. The σ is calculated during PCA. The value 1.4 has been chosen through experimentation on different dataset. It tends to give the best estimate of the size of the muscle's width.

$$Length_{minoraxis} = 1.4 * \sigma \tag{4.9}$$

The length of the minor axis for all the selected slices is averaged and this average length is used as the length of the minor axis for all ellipses that are fitted to individual slices. The length of major axis on the other hand is not averaged because it is best calculated form the start P_{start} and end P_{end} points of the strokes and interpolation. Once the ellipse estimation is complete, points on the ellipse are calculated using the standard equation 4.10 of ellipse. The standard equation of the ellipse represents an ellipse whose major axis is along the x-axis and centered at origin. The points calculated are then used to create the customized 3D Elliptical SMSM.

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \tag{4.10}$$

4.2.5 Construction of the Customized 3D Elliptical SMSM

After obtaining the ellipse equation, points on the ellipse are calculated (see figure 4.8). These points are used for placing the masses. As curvature changes rapidly

---

http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf

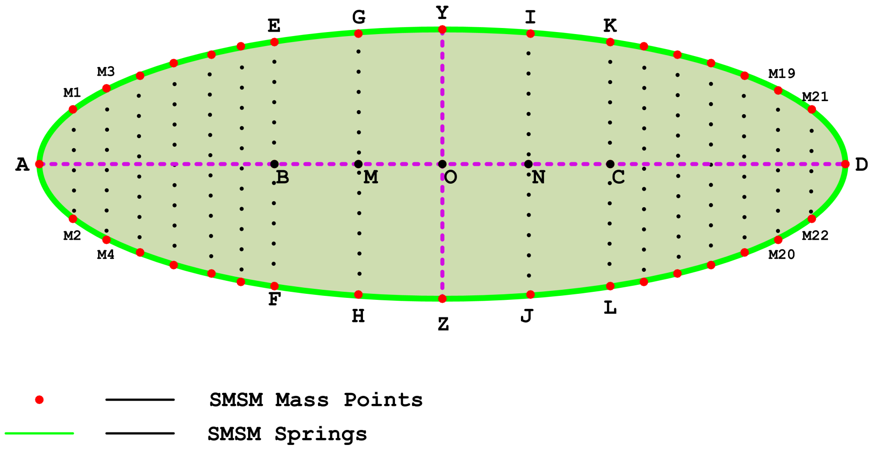


Figure 4.8: Schematic diagram of the calculated positions of an elliptical SMSM where masses will be placed. The diagram shows the relative position of the masses with respect to each other in a slice.

between points A and B and points C and D, therefore, more masses are placed between A and B and C and D and less number of masses between points O and B and O and C. A mass is placed at positions A and D which are the end points of the major axis and at positions Y and Z which are the end points of the minor axis. Distance $\overline{AB}$ and $\overline{CD}$ are calculated using equations 4.11 and 4.12.

$$\overline{AB} = 0.48 * \overline{OA} \tag{4.11}$$

$$\overline{CD} = 0.48 * \overline{OD} \tag{4.12}$$

After calculating points B and C, points E, F, K, L are calculated and a mass is placed at each of these positions. Points M and N are calculated using equation 4.13 and 4.14.

$$\overline{OM} = 0.5 * \overline{OB} \tag{4.13}$$

$$\overline{ON} = 0.5 * \overline{OC} \tag{4.14}$$

After calculating points M and N points G, H, I, J are calculated and a mass is placed at each of these positions. The rest of the masses are distributed equidistantly between A and B and C and D as shown in figure 4.8 by points M1, M2 ... M21, M22.

The number of masses on the ellipse is 32. After experimentation with different number of masses it was found that 32 masses are optimal. With less number of masses, the muscle structure could not be accurately estimated while on the other hand increasing the number of masses gives too few slices to estimate the middle part of the muscle.

After calculating all the mass points on the ellipse, the points are rotated around the origin by an angle given by the orientation of the line joining the start P_{start} and end P_{end} points. The points are then translated by the vector(meanX,meanY) which was calculated when PCA was applied. The same process is repeated for all the selected slices to which an ellipse has been fit.

Figure 4.9 shows the 3D Elliptical SMSM. After the masses has been placed the masses are connected to each other through springs. These masses are called *Outer Layer Masses*. The springs connecting the masses of the same slice are assigned low spring constant so that they can expand. These springs are shown in dark blue in the figure 4.9. The masses are also connected through springs to the masses in the adjacent slices through 1 to 1 correspondence. These springs are assigned high spring constant so that if some masses in one slice get stuck in local minima, the masses in the adjacent slices can pull them out of the local minima. These springs are shown in yellow in the figure 4.9. Masses are also diagonally connected to the masses in the adjacent layers through springs. These springs are assigned very low spring constant so that they do not have any affect on model simulation. They are present only because they are important for SMSM construction. These springs are shown in light blue in figure 4.9. Directed edge sensors are attached to the masses which are sensitive to the edge information contained in the image.

A parallel layer of masses called *Inner Layer Masses* to *Outer Layer Masses* at a distance d is automatically constructed using MSML library. This layer of masses is shown in green in figure 4.9. Intensity sensors are attached to the masses of the parallel layer. These sensors are configured to work within the intensity interval that was estimated earlier(see section 4.2.3). The two layer of masses are connected to each other through springs in 1 to 1 correspondence. These springs are assigned very high spring constant so that the intensity sensors could stop the directed edge sensors from getting distracted by undesirable edges. These springs are shown in red in figure 4.9. All masses are programmed to moved only along x and y axis.

4.3 Constructing 3D Curved Rectangular SMSM

The curved rectangular model is created if the user checks a check box to indicate that such a model should be created. The skeleton of the shape is created through interpolation. Interpolation is carried out the same way as explained in the section 4.2.1 but instead of having just two points P_{start} and P_{end}, interpolation here is done for 15 points, as the shape is too complex to be estimated through just two points. 15 equidistant points each on the two adjacent strokes are obtained and interpolation

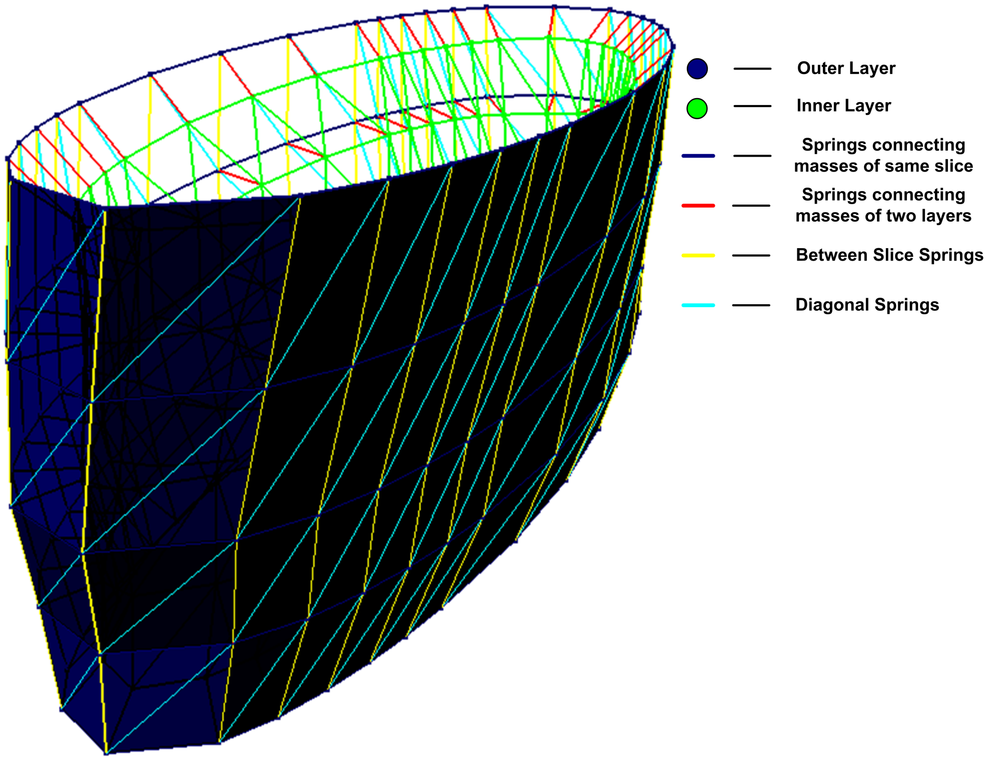

Outer Layer
Inner Layer
Springs connecting masses of same slice
Springs connecting masses of two layers
Between Slice Springs
Diagonal Springs

Figure 4.9: 3D Elliptical SMSM

is done between the these points. The result of the interpolation is 15 new points on the slices between the strokes. These 15 points try to capture the shape intricacies of the muscle as shown in figure 4.10 by points marked P1, P2 ... P14, P15.

The rest of the procedure for slice selection and intensity estimation is same as explained in section 4.2. The width of the curved rectangular model $Width_{CurvedRectangular}$ is predefined which is 6 voxels. No ray casting is required to construct this model, since there is no ellipse to be estimated. Figure 4.4 shows the work flow diagram for creating a curved rectangular SMSM.

4.3.1 Construction of the Customized 3D Curved Rectangular SMSM

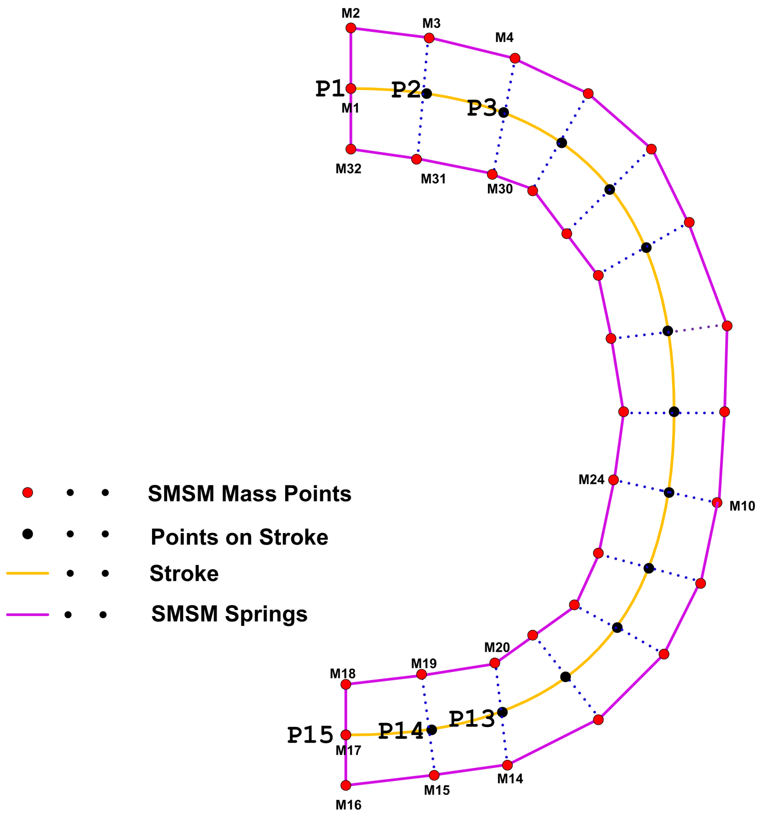


Figure 4.10: Schematic diagram of the calculated positions of curved rectangular SMSM where masses will be placed. The diagram shows the relative position of the masses with respect to each other in a slice.

Figure 4.10 shows the schematic diagram of the curved rectangular model. The total number of masses per slice $Mass_{perslice}$ is 32. Two masses are placed at positions

P1 and P15 which are the starting and the ending points of the stroke. Two masses are placed at positions M2 and M32 with respect to the slope of the line joining P1 and P2. The line joining masses M2 and M32 is perpendicular to the line joining P1 and P2. The length of the line joining M2 and M32 is 6 voxels. 6 voxels have been selected because empirical testing showed that 6 voxels gave optimal estimation of the width of the muscle in the re-sampled datasets. Similarly the same process is repeated for the rest of the points P2, P3 .... P15 on the stroke.

Figure 4.11 shows the 3D Curved Rectangular SMSM. Directed edge sensors are attached to the masses of the *Outer Layer*. The Outer Layer and its masses are shown in dark blue in the figure 4.11. The masses in the same slice are connected to each other through springs. These springs are assigned low spring constants for reasons explained earlier in section 4.2.5. The masses are also connected to the masses in the adjacent slices through springs. These springs are shown in yellow in figure 4.11. A parallel layer of masses is created at a distance *d* voxel. The parallel layer, its masses and springs are shown in green in figure 4.11. Intensity sensors are attached to the masses of the parallel layer. These sensors are configured to work within the intensity interval calculated during intensity estimation process. All masses are programmed to moved only along x and y axis.

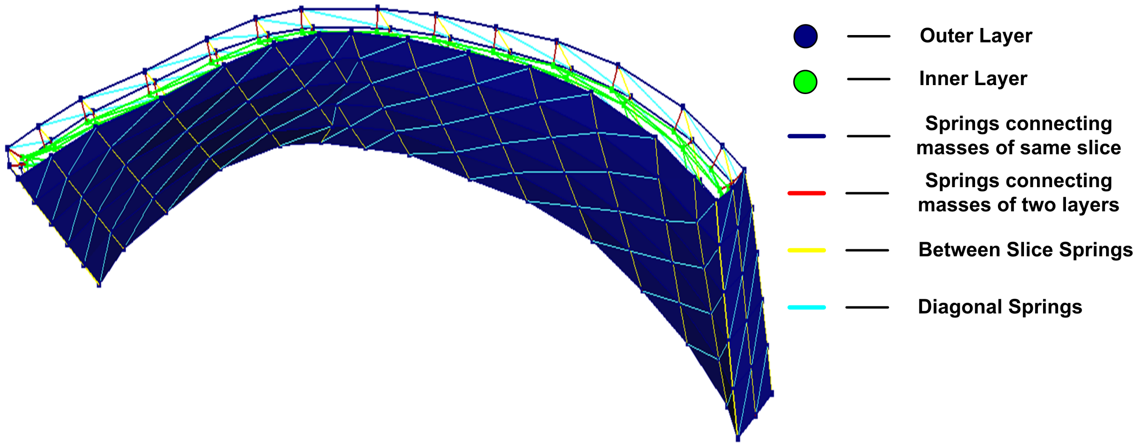

Outer Layer
Inner Layer
Springs connecting masses of same slice
Springs connecting masses of two layers
Between Slice Springs
Diagonal Springs

Figure 4.11: 3D Curved Rectangular SMSM

4.4 Constructing Elliptical Elongated SMSM

As said at the beginning of the chapter that user specifically marks the slice number from where the upper part of the muscle starts, by checking a check box. The user then goes on to enter the strokes that will be used for interpolation. The interpolation is done in the same way as explained in section 4.2.1. Here five points are used for interpolation between strokes, as the shape is again too complex to be estimated through just two points. As can be seen form figure 4.12 that one end of the model is elliptical while the other end consists of rectangular regions. These points are shown in the figure 4.12 as A, B, C, D, E.

The procedure for selecting the number of slices $Slice_{required}$ also depends on the number of masses per slice $Mass_{perslice}$ and calculated in the same way as explained earlier in section 4.2.2. The intensity estimation procedure is also the same as explained in section 4.2.3. The only difference is that here smaller rectangular regions are defined using points A-B, B-C, C-D and D-E. The elliptical part of the model is estimated the same way as explained in section 4.2.4. Here the rays are casted from point B. The average length of the minor axis $Avg-Len_{minor_j}$ is estimated the same way as explained the section 4.2.4. As explained in section 2.3 that the point from where the upper part of the muscle starts, its elliptical part is broader than the elliptical at the points where the upper part of the muscle terminates at the mastoid process. Figure 2.13 shows the difference in the width of the elliptical part of the upper part of the muscle. To compensate the difference in the width of the elliptical parts, the width in the starting slice of the upper part of the muscle is calculated using the equation 4.15. The value of the factor starts from 0.3 when $j = 1$ and it decreases linearly to 0 when $j = (M-1)/2$. It linearly decreases further to -0.3 when $j = M$. This results in broader elliptical part of the SMSM at the starting point of the upper part of the muscle and thinner elliptical part of the SMSM at the ending point of the upper part of the muscle. The length of major axis and the orientation of the ellipse is estimated using points A and C.

$$Avg-Len_{minor_j} = Avg-Len_{minor_j} + Avg-Len_{minor_j} * factor \tag{4.15}$$

where,
factor = 0.3 .... 0 .... -0.3
j = 1 ... M

4.4.1 Construction of the Customized 3D Elliptical Elongated SMSM

There are two parts of the model, the elliptical part and the thin elongated rectangular part. 6 masses are placed at locations G, H, I, J, K and L which are calculated using points C, D and E. A mid point X between C and D is obtained. Depending upon the slope of the line CD, points G and H are calculated. If the slope of the line CD is greater than 1, then $X \rightarrow y$ is decremented and incremented by $Width_{Rectangular}$

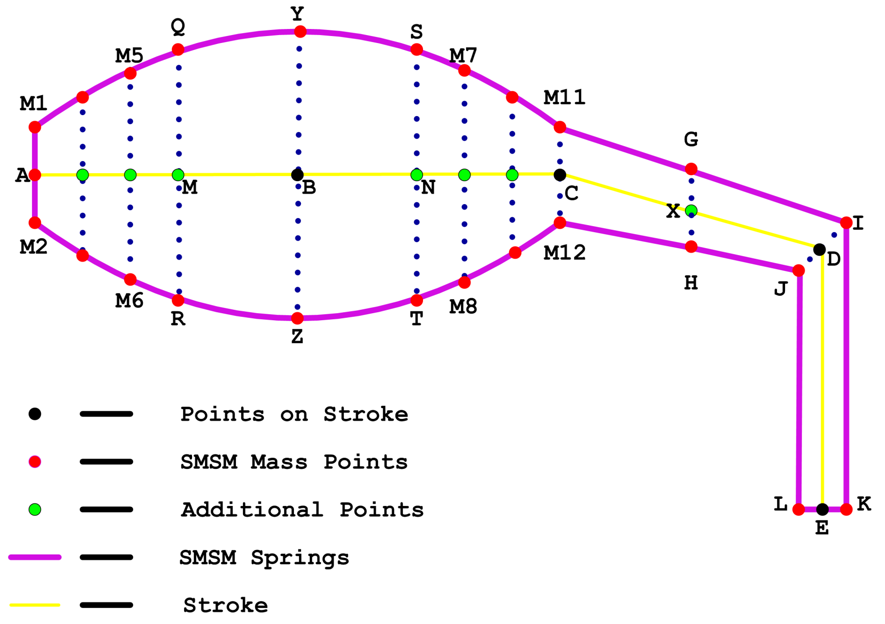


Figure 4.12: Schematic diagram of the calculated positions of elliptical elongated SMSM where masses will be placed. The diagram shows the relative position of the masses with respect to each other in a slice.

to calculate $G \rightarrow y$ and $H \rightarrow y$ respectively, as given by equation 4.16. The x-coordinate of G and H is equal to the x-coordinate of X. If the slope of the line CD is less than 1 then $X \rightarrow x$ is decremented and incremented by $Width_{Rectangular}$ to calculate points $G \rightarrow x$ and $F \rightarrow x$ respectively, as given by equation 4.17. The y-coordinate of G and H is equal to the y-coordinate of A. The same process is used for calculating points I and J using point D and K and L using point E. The rest of the masses are placed on the elliptical part.

$$\begin{aligned} G \rightarrow y &= X \rightarrow y - Width_{Rectangular} \\ G \rightarrow x &= X \rightarrow x \\ H \rightarrow y &= X \rightarrow y + Width_{Rectangular} \\ H \rightarrow x &= X \rightarrow x \end{aligned} \tag{4.16}$$

$$\begin{aligned} G \rightarrow y &= X \rightarrow y \\ G \rightarrow x &= X \rightarrow x - Width_{Rectangular} \\ H \rightarrow y &= X \rightarrow y \\ H \rightarrow x &= X \rightarrow x + Width_{Rectangular} \end{aligned} \tag{4.17}$$

The procedure for placing the masses on the ellipse is almost the same as described earlier in section 4.2.5. There are some differences regarding the position of the masses. A mass each is placed at Y and Z which are the end points of the minor axis. Distances $\overline{AM}$ and $\overline{CN}$ are calculated using equations 4.18, 4.19.

$$\overline{AM} = 0.4 * \overline{AB} \tag{4.18}$$

$$\overline{CN} = 0.4 * \overline{BC} \tag{4.19}$$

where
AB = BC = 1/2 * Length of Major Axis

After calculating points M and N, points Q, R, S, T are calculated and a mass is placed at each of these positions. The rest of the masses are placed equidistantly between A and M and C and N. No masses are placed between M and B and B and N. The total number of masses in one slice is 32. The number of masses placed on the ellipse is 26 while 6 masses are placed on the rectangular part. All masses are programmed to moved only along x and y axis. After calculating all the mass points on the ellipse, the points are rotated and translated just the way as explained in section 4.2.5. The steps for the SMSM model creation are also the same as explained earlier in section 4.2.5. Figure 4.13 shows the 3D Elliptical Elongated SMSM.

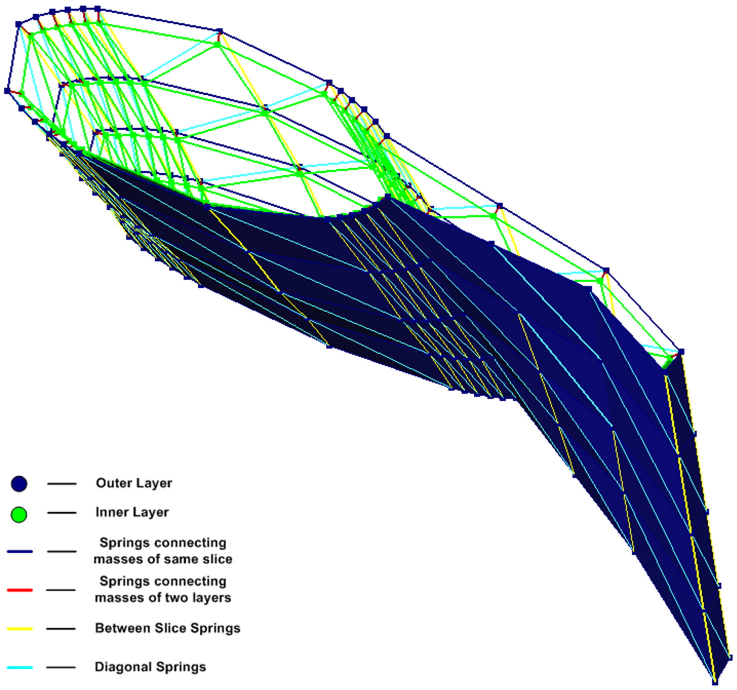

Outer Layer
Inner Layer
Springs connecting masses of same slice
Springs connecting masses of two layers
Between Slice Springs
Diagonal Springs

Figure 4.13: 3D Elliptical Elongated SMSM

4.5 Preprocessing

This section will present details about the two preprocessing steps done on the datasets. This includes re-sampling of the datasets and the cutting of the gray values. These steps helped make the results better.

4.5.1 Re-Sampling Datasets

The re-sampling of the datasets was necessary for two important reasons. First it allows to minimize the effect of the local minima. Second it is desirable to have isotropic voxels from SMSM simulation perspective.

It was noted during the initial testing that local minima were a major problem. The SMSM when placed inside the muscle was distracted by these local minima and failed to converge towards the true muscle boundaries as can be seen in figure 4.14. The remedy was to down sample the data. Down sampling was mainly done on the x and y axis. x and y axis spacing in original test datasets varied from 0.254 mm to 0.805 mm. It was down sampled to 1 mm. On the other hand z axis spacing varied from 1 mm to 5 mm. Here the datasets were up sampled to 1 mm. This provided isotropic voxels of size 1 mm, which was also desirable from the SMSM simulation perspective. Isotropic voxel size also made it easy to automate the parameters of simulation and the SMSM apart from decreasing the dataset size.

4.5.2 Cutting Gray Values

As mentioned in section 2.3 that if the muscle is close to the skin, the gradient force generated in this region is very high. This high gradient force is instrumental in making the SMSM unstable as shown in figure 4.15.

The remedy suggested by P. Hahn [26] is cutting the gray value. As will be explained in the following sections that the muscle intensity range is estimated by $(MeanIntensity - 3 * \sigma, MeanIntensity + 3 * \sigma)$. Anything outside the $(MeanIntensity$ - 6 * σ , $MeanIntensity + 6$ * $\sigma)$ is cut off by a threshold value of $(MeanIntensity - 6 * \sigma)$. This way the generated gradient force is not high and does not make the SMSM unstable. The value of $6*\sigma$ has been chosen to ensure that it is well outside the intensity interval of the muscle and yet not very far away from the muscle intensity interval.

4.6 Ideas Taken From The Previous Work

The model for segmenting sternocleidomastoid muscle has many similarities to the lymph node model presented by J. Dornheim et al. [14]. The sternocleidomastoid model also combines intensity, gradient and shape information. Gradient sensors are

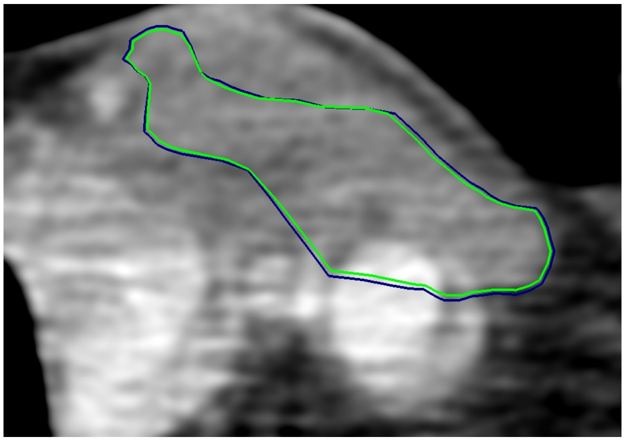


Figure 4.14: SMSM distracted by local minima when the dataset has not been resampled.

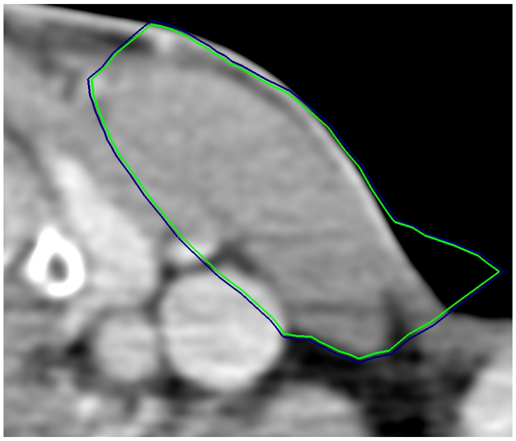


Figure 4.15: SMSM distracted by high gradient force before the cutting of intensity values of the dataset.

attached to the outer layer of masses to look for muscle boundaries, while intensity sensors were attached to inner layer of masses to keep the model within the intensity interval specific to the muscle. The springs connecting the outer and the inner layer masses were assigned high spring constant to make them stiff. Many of the problems faced during the lymph node segmentation were also faced during the muscle segmentation. The muscle is adjacent to another muscle. Both muscles have a similar intensity interval, due to which there are no distinct edges to separate the two muscles. The target muscle is also surrounded by high density soft tissues like blood vessels and the muscle interior is inhomogeneous. Not only the muscle size varies but also its shape to further complicate the segmentation task. Due to the shape and the size variations in the muscle, strokes are used instead of just seed points.

For sternocleidomastoid muscle segmentation, the model can not be created from sample segmentation as was done by J. Dornheim et al. [13] for segmenting the thyroid cartilage because of the large variations found in the muscle shape. Figures 2.9, 2.10, 2.11, 2.12 and appendix A shows many of the variations found in the muscle shape. It was also observed that it was imperative to have a regular structure which could not be created using the automatic mass placement method provided by the MSML and MLMSML libraries. The structure and the steps required in creating specific SMSM models have been described in detail earlier in this chapter.

The way masses are placed on the sternocleidomastoid model is inspired from the work of L. Dornheim et al. [18]. Masses on the sternocleidomastoid model are placed according to the curvature but it differs in the curvature estimation method. For elliptical SMSMs, more masses have been placed near the end points of the major axis of the ellipse as curvature changes rapidly in that region as compared to the change in curvature near the end of the minor axis. For the curved rectangular SMSMs, equidistant points corresponding to the number of mases to be placed, on the stroke are obtained and masses are placed according to the slope of the stroke in that region.

4.7 Summary

This chapter starts by giving details of the preprocessing stage. Down sampling and cutting of gray values were the two steps carried out in this stage. Then the main technique is presented. Three different SMSM could be created for segmenting the complete muscle. Details about the *Interpolation*, *Slice Selection*, *Intensity Estimation* and optionally *Ellipse Estimation* are given. These steps are necessary to create any of the three different SMSM. Then details about constructing *Elliptical*, *Curved Rectangular* and *Elliptical Elongated* SMSM are presented.

Chapter 5

Implementation

5.1 Software

Microsoft Visual C++ 2003 .Net Standard Development Kit (SDK) was used for programming. Following is a list of the other libraries and softwares used during the development.

1. **MeVisLab 1.6** MeVisLab was used for visualization and building of the graphical user interface (GUI) of the application through Module Definition Language (MDL). It is a graphical, module based programming environment which allows for quick integration and testing of the new algorithms or application prototypes. MeVisLab provides, image processing, visualization and interaction modules that are combined together in a network to solve clinical problems related to image processing and visualization. MeVisLab uses QT application framework, the OpenInventor 3D visualization toolkit and OpenGL. A detailed information about MeVisLab can be found at [23], [25].

2. **OpenGL** OpenGL[1] is a graphics library used for creating 2D and 3D computer graphics. MeVisLab internally uses OpenGL for visualization.

3. **Open Inventor** Open Inventor[2] is an object oriented 3D toolkit used for creating interactive graphical applications. It is built on top of OpenGL. It is also integrated in MeVisLab.

4. **VTK** The Visualisation ToolKit[3] is a software system for 3D computer graphics, image processing and visualization. It is also integrated in MeVisLab. MSML library makes use of the VTK integrated in the MeVisLab.

---

[1] Available at : http://www.opengl.org/

[2] Available at : http://oss.sgi.com/projects/inventor/

[3] Available at : http://www.vtk.org/

5. **CGAL** The Computational Geometry Algorithms Library[4] provides efficient and reliable geometric algorithms. CGAL provides algorithms in the areas of computer graphics, scientific visualization, computer aided design and modeling, geographic information systems, molecular biology, medical imaging, robotics and motion planning, mesh generation, numerical methods etc. The MSML library makes use of the CGAL.

6. **MSML** Magdeburg Shape Model Library is developed by L. Dornheim. The library provides functions for the creation, simulation and evaluation of the Stable Mass Spring Model.

7. **MLMSML** MLMSML provides a bridge between the MSML and the MeVisLab. It provides inventor modules for the MeVisLab through which MSML library can be used.

5.2 Prototype For Segmenting The Sternocleidomastoid Muscle

An inventor module **SternocleidomastoidMuscleSegmentation** was developed which encompasses the muscle segmentation process. **SternocleidomastoidMuscleSegmentation** module together with ML and inventor modules from MeVisLab and MLMSML modules developed by L. Dornheim creates the complete prototype. Figure 5.1 shows the MeVisLab network diagram where the interconnection between the different types of modules can be seen. A short description about the functionality of **SternocleidomastoidMuscleSegmentation** module, ML and inventor modules from MeVisLab and MLMSLM modules is given below. Figure 5.2 shows few additional modules used for the comparison of the segmentation obtained from SMSM with the gold standard.

5.2.1 MeVisLab Modules

Following are the MeVisLab modules used to create the MeVisLab network[5].

1. **SoExaminerViewer** SoExaminerViewer renders the 3D scene and provides event handling functionality. It is used to view the SMSM.

2. **View3D** View3D module provides Volume Rendering for the 3D dataset.

---

[4] Available at : http://www.cgal.org/

[5] Complete list of the MeVisLab modules can be found at http://www.mevislab.de/fileadmin/docs/MLabModules/std/help/html/private/index.html

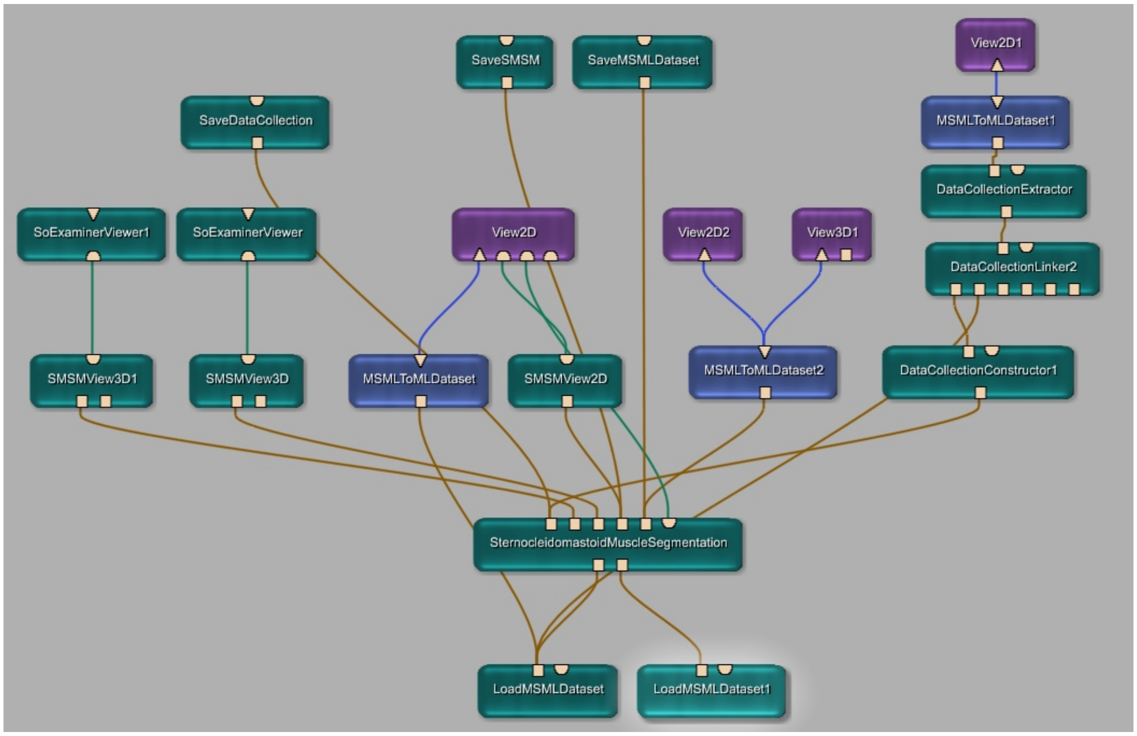


Figure 5.1: MeVisLab network diagram for muscle segmentation

3. **View2D** View2D is a macro module and is used to view 2D objects. It is used to view the CT dataset as it provides the functionality to scroll through the slices of the CT datasets, showing a single slice in 2D at a time. The View2D is also used to obtain the user strokes.

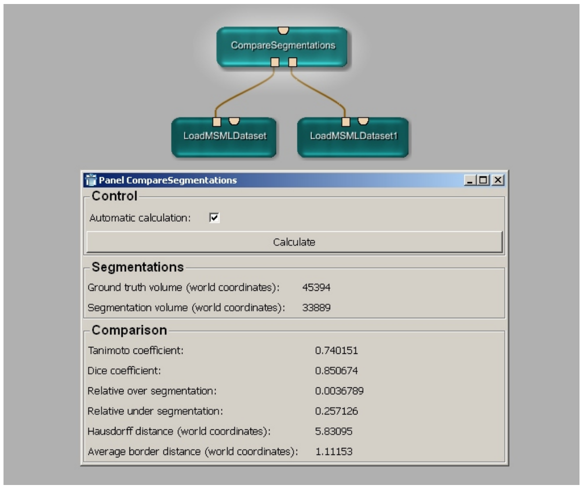
5.2.2 MLMSML Modules

Following are some of the MLMSML modules used to create the MeVisLab network.

1. **LoadMSMLDataset** LoadMSMLDataset module loads the MSML dataset from a file. The MSML library defines its own format for the images and CT datasets.

2. **SaveMSMLDataset** SaveMSMLDataset module is used to save the MSML dataset to a file.

3. **MSMLToMLDataset** MSMLToMLDataset is used to convert the MSML dataset to ML dataset. ML dataset is the format used by the MeVisLab modules.



Automatic calculation: ☑

Ground truth volume (world coordinates): 45394
Segmentation volume (world coordinates): 33889

Tanimoto coefficient: 0.740151
Dice coefficient: 0.850674
Relative over segmentation: 0.0036789
Relative under segmentation: 0.257126
Hausdorff distance (world coordinates): 5.83095
Average border distance (world coordinates): 1.11153

Figure 5.2: MeVisLab network diagram for comparing model segmentation with manual segmentation

4. **DataCollectionConstructor** DataCollectionConstructor is used to create the processed data. There are many options in which data can be processed, the important ones regarding the muscle segmentation are; threshold data and interval plateau data. DataCollectionConstructor module has been incorporated into the **SternocleidomastoidMuscleSegmentation** module.

5. **SaveDataCollection** SaveDataCollection module is used to save the DataCollection to a file.

6. **SMSMSimulator** SMSMSimulator module is used to simulate the SMSM model. It provides many features like setting the SMSM parameter values, repositioning, scaling, rotating the SMSM etc in addition to starting and stopping the SMSM simulation. The functionality of this module has also been incorporated into the **SternocleidomastoidMuscleSegmentation** module.

7. **SaveSMSM** SaveSMSM module is used to save the SMSM to a file.

8. **SMSMView2D** SMSMView2D helps in viewing the SMSM in View2D module. It also provides different features like changing the colour of springs and masses or enabling different parts of the SMSM.

9. **SMSMView3D** SMSMView2D helps in viewing the SMSM in SoExaminerViewer module. It also provides almost all features that the SMSMView2D provides.

10. **SMSMVoxelizer** SMSMVoxelizer module converts the volume data encompassed by the SMSM in to a segmented dataset. This functionality of this module has been incorporated into the **SternocleidomastoidMuscleSegmentation** module.

11. **CompareSegmentation** CompareSegmentation module is used to compare the segmentation result obtained from the SMSM and the one obtained manually. This module provides comparisons like hausdorff distance, average boder distance, tanimoto coefficient, dice coefficient etc. Figure 5.2 shows the network digram for comparing the result obtained from the SMSM model segmentation with that obtained from manual segmentation. A detailed description about the comparison criteria is given in chapter 6.

5.3 SternocleidomastoidMuscleSegmentation Module

An inventor module **SternocleidomastoidMuscleSegmentation** was created that provides all the functionality for creating SMSM specific to the muscle. **SternocleidomastoidMuscleSegmentation** module can be seen in figure 5.1.

The inputs to the module are two MSML datasets. One dataset is the resampled dataset. This dataset is used to adapt the SMSM according to the data. Other dataset is the original dataset. This dataset is used for segmenting the muscle from it using the adapted SMSM.

The outputs from this inventor module are MSML segmented dataset, SMSMs for the upper and the lower half of the muscle, SMSM created by combining together upper and lower half SMSMs and a MSML data collection.

The module encompasses all the functionality discussed in chapter 4 from accepting strokes from the user, performing interpolation, estimating intensity, and constructing SMSM. The module also incorporates the functionality of MLMSML module **SMSMSimulator** therefore, it is possible to simulate the model from the module. The **SternocleidomastoidMuscleSegmentation** module also incorporates the functionality of the MLMSML module **SMSMVoxelizer**. The next section gives a detailed description of the purpose and the functionality of the different sections of the GUI.

5.3.1 Graphical User Interface

GUI was created using the MeVisLab scripting language Module Definition Languag (MDL). The GUI of the application is shown in figures 5.3. The different panels of the GUI are described below.

The **Data** panel gives detail about the estimated minimum and the maximum intensity value of the muscle, the distance between the inner and the outer layers of the SMSM in voxels, the number of masses in one layer in a single slice and the total muscle segmentation time starting from the point of the user strokes.

The **Simulation Process Information** panel gives detail about the SMSM. The important one is the *Remaining Simulation Steps* count. The maximum simulation step count is 500 after which the simulation stops. The simulation could also stop if the model reaches equilibrium shown by the *Balanced Status.*

The **Select / Create Model** panel allows to select from different types of SMSMs by checking the check boxes. The buttons *Create Upper SMSM* and *Create Lower SMSM* creates the corresponding SMSM and places it within the dataset.

If the user needs to create the *Elliptical SMSM* then he should just start entering the strokes. If he reaches a point from where he wants to estimate the SMSM by Elliptical Elongated SMSM then he should check the check box *Check when entering Upper Strokes* and then go on entering the strokes. After entering the stokes and pressing the button *Create Lower SMSM* will first create *Elliptical SMSM.* The simulation could be started by pressing the *Initialize Simulation* and *Start Simulation* buttons. At the end of the simulation the *Elliptical Elongated SMSM* will be created and placed inside the dataset. The simulation for this model could then be started by again pressing the *Initialize Simulation* and *Start Simulation* buttons. At the end of the simulation the complete muscle is segmented from the dataset using the two SMSMs and outputted.

If instead of creating an *Elliptical SMSM* the user wants to create a *Curved Rectangular SMSM* then he should check the check box *Check to create Rectangular SMSM* before entering the strokes. After entering the stokes and pressing the button *Create Lower SMSM* will create *Curved Rectangular SMSM.*

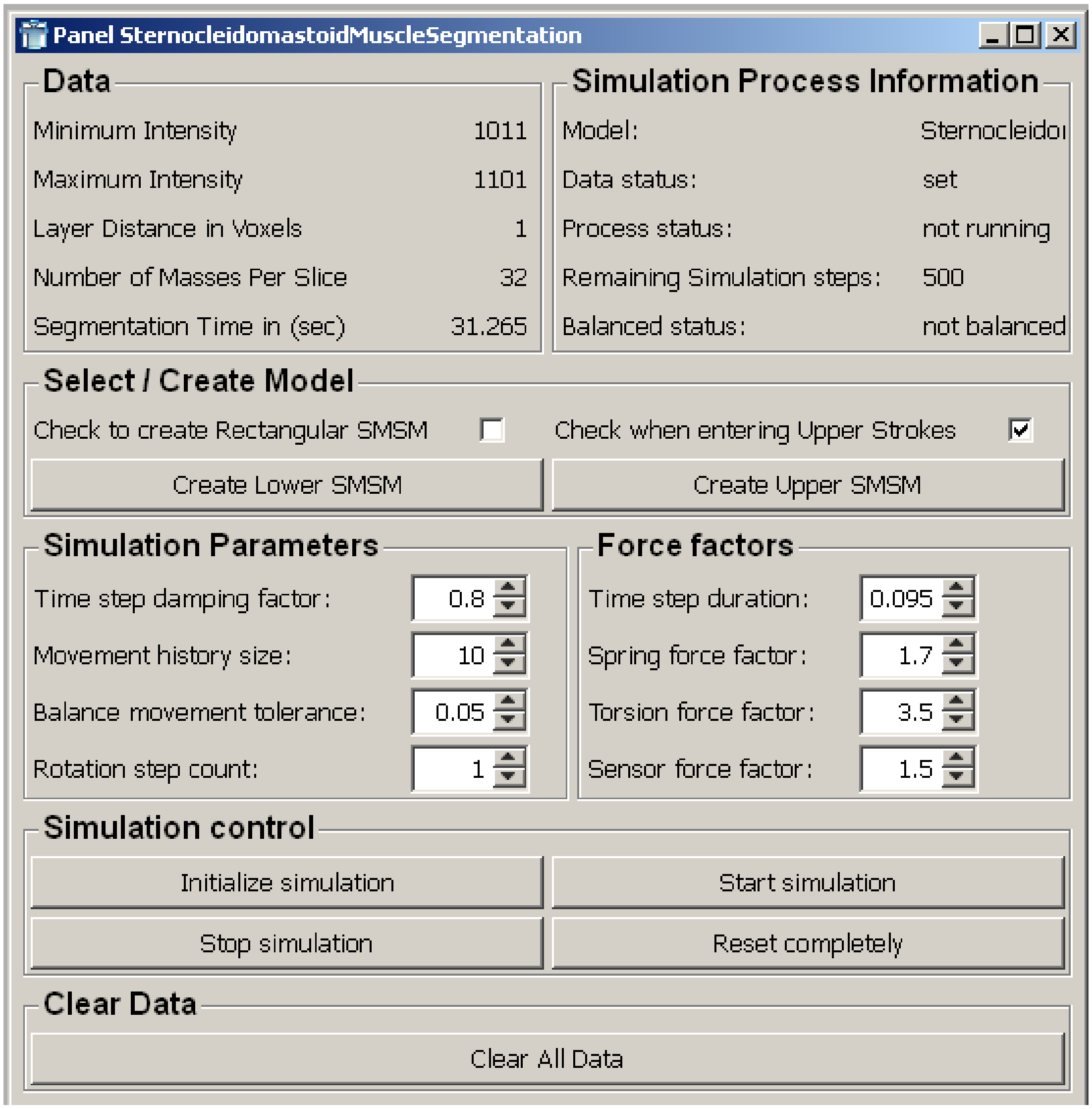
Selecting the check box *Check when entering Upper Strokes* and then entering strokes and selecting nothing else will create only the *Elliptical Elongated SMSM* when the button *Create Upper SMSM* is pressed. These models could also be simulated by pressing the *Initialize Simulation* and *Start Simulation* buttons.

Similarly if only the check box *Check to create Rectangular SMSM* is selected will create only the *Curved Rectangular SMSM* when the button *Check Lower SMSM* is pressed. If non of the check boxes are selected and the strokes are entered, pressing the *Create Lower SMSM* will create only the *Elliptical SMSM.*

The **Simulation Parameters** and the **Force Factors** panel allows to changes the SMSM simulation parameters. The SMSM simulation parameters have already been adjusted but they can be varied within a certain range for smooth simulation. The adjusted parameter values and the variation interval is given in table 6.1.

The **Simulation Control** panel provides buttons for **Initializing Starting** and **Stopping** the SMSM simulation. The **Reset Completely** button resets the SMSM to its learned state. After the SMSM has been created and placed inside the dataset, it has to be initialized before starting the simulation.

The **Clear Data** button deletes all the data and allows to create a new SMSM from the beginning. The strokes entered by the user are also deleted, therefore, the user will have to insert new strokes.



Panel SternocleidomastoidMuscleSegmentation

Data

Minimum Intensity	1011
Maximum Intensity	1101
Layer Distance in Voxels	1
Number of Masses Per Slice	32
Segmentation Time in (sec)	31.265

Simulation Process Information

Model:	Sternocleido
Data status:	set
Process status:	not running
Remaining Simulation steps:	500
Balanced status:	not balanced

Select / Create Model

Check to create Rectangular SMSM ☐

Check when entering Upper Strokes ☑

Simulation Parameters

Time step damping factor:	0.8
Movement history size:	10
Balance movement tolerance:	0.05
Rotation step count:	1

Force factors

Time step duration:	0.095
Spring force factor:	1.7
Torsion force factor:	3.5
Sensor force factor:	1.5

Simulation control

Clear Data

Figure 5.3: GUI

Chapter 6

Evaluation

6.1 Test Environment

The developed prototype was used on a standard PC (Intel Pentium 4 - 3.2 GHz Processor , 3.2 GB RAM) with a modern 3D Graphics Card (ATI RADEON 9600 Series AGP, 256 RAM).

The prototype was tested on 14 3D-CT datasets of human neck from 14 different patients. There is a pair of muscle in every dataset. The prototype developed was tested only on muscles of one side, therefore, there were in all 14 different muscles. As described earlier all datasets were down sampled to isotropic voxels of size 1 mm. The original dataset spacing varies from 0.254 mm to 0.805 mm in x and y direction, 1 mm to 5 mm in z direction.

Some problems arose due to different artifacts present in the datasets; 5 datasets were affected by tumor, 8 datasets were affected by motion artifacts, 1 dataset was affected by beam hardening artifact, 3 datasets were affected by noise and 2 datasets suffered from low contrast. Through the use of down sampled datasets the effect of motion and beam hardening artifacts was reduced. The SMSM was able to bridge the absent boundary in tumor affected regions mainly through the shape knowledge encoded in the SMSM.

10 manual segmentations done by other students were used as gold standard. Though the segmentation was time consuming but it was not difficult for the other students to easily demarcate the muscle boundaries. Only in places where the muscle is adjacent to another muscle the muscle boundary is not visible but it could be easily identified by traversing through the slices before and after the slice of interest. The same holds true for the part of the muscle boundary that has been destroyed by tumor infiltration. The gold standard segmentations were used for a quantitative assessment whose results are presented below. 6 different similarity measures between the gold standard and the segmentation obtained from the SMSM were calculated.

These different similarity measure are briefly described below, after which the results are presented.

The SMSM model was automatically generated for every dataset. The parameters for SMSM were adjusted using 3 datasets and then the adjusted parameters were used on all datasets for segmentation. The complete SMSM model consisted of 768 masses, 2618 springs and 1468 contour faces. The average spring degree on a mass was 3.41. The SMSM model is a little complex but possesses a simple and a vivid structure as explained in chapter 4.

6.2 Parameter Estimation

Down sampling made it easier to automate the parameters as the spacing in all dimensions was equal. The parameters were adjusted using 3 datasets and then the same parameters were used on all datasets. The following table 6.1 shows the parameters for the SMSM and simulation. There are three fields in the table, *Parameter* giving the name of the parameter, *Value* giving the parameter value and *Tolerance* giving the tolerance for the parameter. The tolerance has not been used for the test dataset. The tolerance value has been given for those who wants to modify the parameters on their own.

The table shows that the individual mass weights w_o, w_i, torsion weights t_o, t_i, spring weights sp_o, sp_i and sensor se_o, se_i are all kept 1. The inner and the outer mass layers ld have a distance of 1.0 voxel. The springs connecting the inner and the outer mass layers bl_{exp} have been assigned high spring constant 75 so that the inner masses could stop the outer masses from tracking false edges. Similarly the springs connecting the outer masses in one slice with the masses in the adjacent slice bs_{exp} have been assigned high spring constant 75 so that if masses in one slice get stuck in local minima the masses in the adjacent slices can pull them out of the local minima because of stiff springs. The global sensor $w_s(i$, spring w_f and torsion w_t force constant have been set to 1.2, 2.0 and 4.0 respectively.

The simulation parameters time step ts have been adjusted to 0.095 and the damping factor tsd to 0.78. The two simulation stopping criteria use were balance movement tolerance bm and simulation step count sc. Through experimentation bm have been found to be 0.05 and sc to be 500. The simulation stops if either the model reaches the balanced state or the simulation step count reaches 500. For a detailed review of all the parameter see [16].

Parameter	Symbol	Value	Tolerance
outerMassWeight	w_o	1.0	± 0.0
innerMassWeight	w_i	1.0	± 0.0
outerTorsion	t_o	1.0	± 0.0
innerTorsion	t_i	1.0	± 0.0
outerSensor	se_o	1.0	± 0.0
innerSensor	se_i	1.0	± 0.0
outerExpansion	sp_o	1.0	± 0.0
innerExpansion	sp_i	1.0	± 0.0
modelTorsionForceFactor	w_t	4.0	± 0.5
modelSpringForceFactor	w_f	2.0	± 0.2
modelSensorForceFactor	$w_s(i)$	1.2	± 0.3
betweenLayerExpansion	bl_{exp}	75	± 5.00
betweenSliceExpansion	bs_{exp}	75	± 5.00
layerDistance (Voxel)	ld	1.0	± 0.00
modelTimeStepDuration	ts	0.095	± 0.005
modelTimeStepDampingFactor	tsd	0.780	± 0.030
modelBalanceMovementTolerance	bm	0.050	± 0.000
SimulationStepCount	sc	500.0	± 0.000
ModelEstimationWidth	m_w	2.0	+ 1.0
IntensityEstimationWidth	i_w	2.0	+ 1.0

Table 6.1: Parameter values for SMSM

6.3 Comparison Coefficients

1. **Tanimoto Coefficient** Tanimoto Coefficient measures the similarity between two sets. It is calculated by dividing the intersection of two set with the union of the two sets as given by equation 6.1). It is also known as Jaccard coefficient [44].

$$TanimotoCoefficient = \left(\frac{|A \bigcap B|}{|A \bigcup B|}\right) \tag{6.1}$$

2. **Dice Coefficient** Dice Coefficient is measured as twice the intersection of two sets A and B divided by the sum of the two sets as given by equation 6.2) [34].

$$DiceCoefficient = \left(\frac{2 * |A \bigcap B|}{|A| + |B|}\right) \tag{6.2}$$

3. **Relative Over Segmentation** Relative Over Segmentation is defined as the

pixels not part of the manual segmentation but are segmented by the SMSM model. It can be calculated using equation 6.3 [33][1]

$$ROS = \left(\frac{O_p}{U_p + D_p} \right) \tag{6.3}$$

where
O_p = the number of pixels that should not be included in the segmentation result.
U_p = the number of pixels that should be included in the segmentation result.
D_p = the number of pixels that are included in manual segmentation as well as the SMSM segmentation.

4. **Relative Under Segmentation** Relative Under Segmentation is defined as the pixels part of the manual segmentation but are not segmented by the SMSM model. It can be calculated using equation 6.4 [33][2]

$$RUS = \left(\frac{U_p}{U_p + D_p} \right) \tag{6.4}$$

where
O_p = the number of pixels that should not be included in the segmentation result.
U_p = the number of pixels that should be included in the segmentation result.
D_p = the number of pixels that are included in manual segmentation as well as the SMSM segmentation.

5. **Hausdorff Distance** Hausdorff Distance is a technique to compare two different sets. It is generally used in pattern recognition to find how closely two patterns resembles. For every point in set A a minimum distance to the points in set B is calculated and vice versa. The maximum of the two minimum distances is the Hausdorff distance as give by equation 6.5. Hausdorff distance gives a good measure of the bilateral closeness of the two sets [48].

---

[1] The notion used in the paper is opposite to what has been used here. In the paper [33] they define over-segmentation as being pixels that were not segmented by the model but are part of the manual segmentation and under segmentation being the pixels that were segmented by the model but are not part of the manual segmentation.

[2] The notion used in the paper is opposite to what has been used here. In the paper [33] they define over-segmentation as being pixels that were not segmented by the model but are part of the manual segmentation and under segmentation being the pixels that were segmented by the model but are not part of the manual segmentation.

$$H(A,B) = max\left(h\left(A,B\right), h\left(B,A\right)\right) \tag{6.5}$$

where,

$$h\left(A,B\right) = max_{a\in A}min_{b\in B}\left\|a-b\right\| \tag{6.6}$$

$$h\left(B,A\right) = max_{b\in B}min_{a\in A}\left\|b-a\right\| \tag{6.7}$$

$h\left(A,B\right)$ is called the *directed Hausdorff Distance* from set A to B and $h\left(B,A\right)$ is called the *reverse Hausdorff Distance* from set A to B.

6. **Average Border Distance** Average border distance is the same as the hausdorff distance except that it calculates the average distance between points A and B instead of taking the maximum distance.

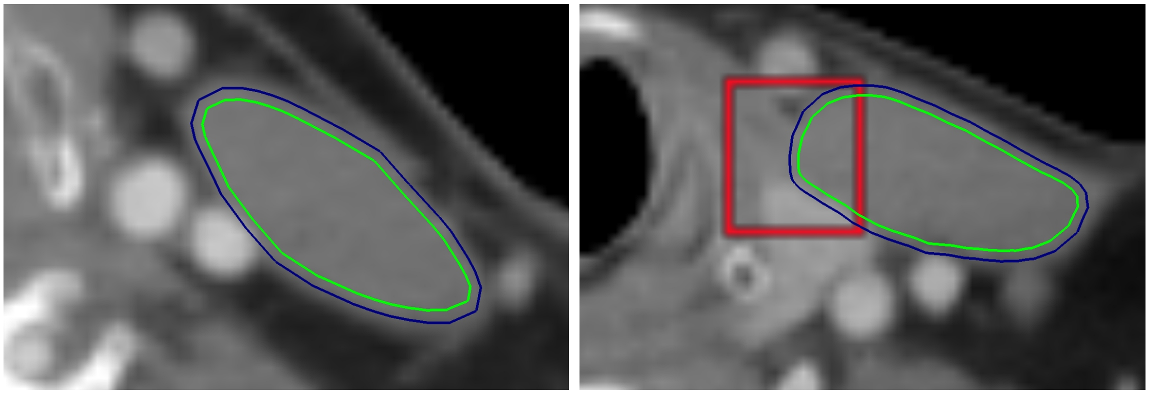
6.4 Qualitative Assessment

The quality of the segmentation is generally good as shown in figure 6.1. The SMSM was able to find the true muscle boundaries most of the time as can be seen in figure 6.1a. It was also able to bridge the indistinct and invisible boundaries in regions where muscle is adjacent to another muscle as shown in figure 6.1b. In some regions the infiltration of tumor had destroyed the muscle boundaries. This situation occured in 5 out of 14 test datasets. Using the shape and size knowledge of the muscle, the SMSM was able to bridge the destroyed boundaries in all 5 cases as shown in figure 6.1c.

There were four aspects of the developed prototype that were important in obtaining good segmentation. They are discussed in the following sections.

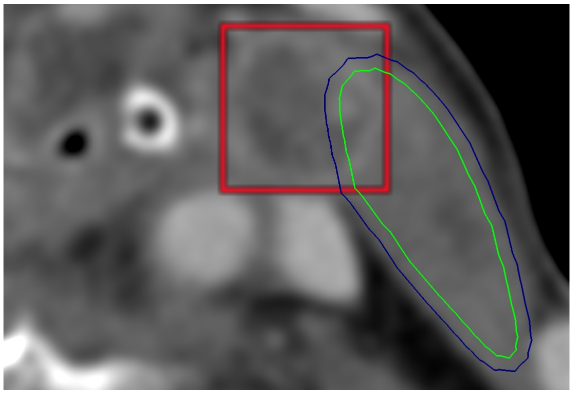
6.4.1 Cutting Gray Values

Cutting of gray values helped in automating the SMSM and simulation parameters. If the muscle boundary is adjacent to the skin, the high gradient in this region was instrumental in making the SMSM unstable as can be seen in figure 6.2b. It became difficult to get a set of parameters that worked on all datasets. The idea of cutting of gray values is inspired from the work of Peter Hahn [26] helped solved this problem. The cutting of gray values outside a certain range (see section 4.5.2 for details) helped to get a strong gradient but not strong enough to make the SMSM unstable as can be seen in figure 6.2d. Figure 6.2d shows the same slice as in figure 6.2b



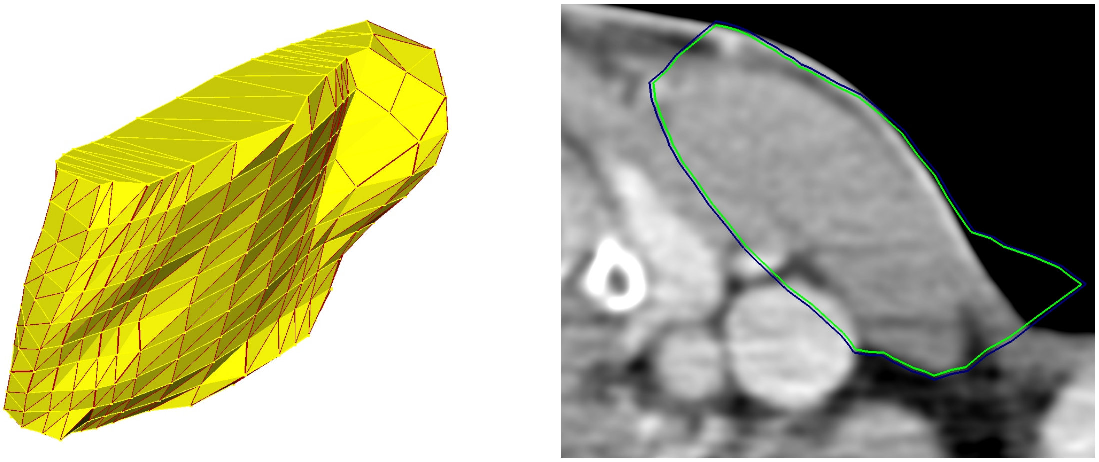
(a) Adapted SMSM

(b) SMSM adaptation in invisible boundary region. The region has been marked with a red rectangle.



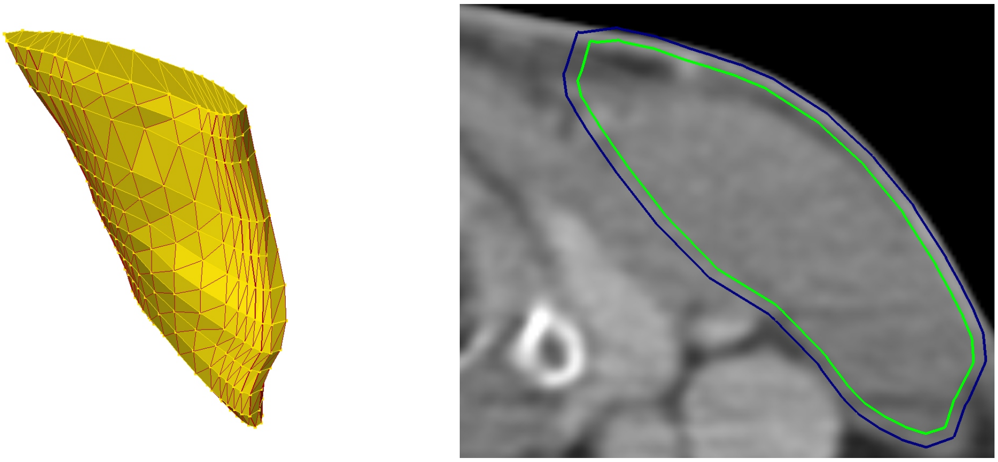
(c) SMSM adaptation in tumor affected region. The region has been marked with a red rectangle.

Figure 6.1: SMSM adaptation in different regions



(a) Deformed 3D SMSM of the lower half of the muscle

(b) Deformed SMSM in a slice



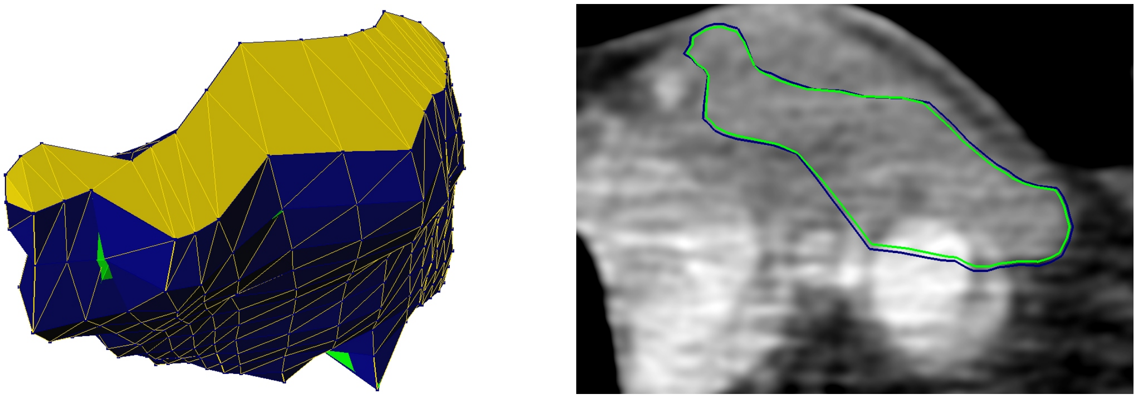
(c) Undeformed 3D SMSM of the lower half of the muscle.

(d) SMSM does not deform after the gray values have been cut.

Figure 6.2: SMSM deforms because of strong gradient.

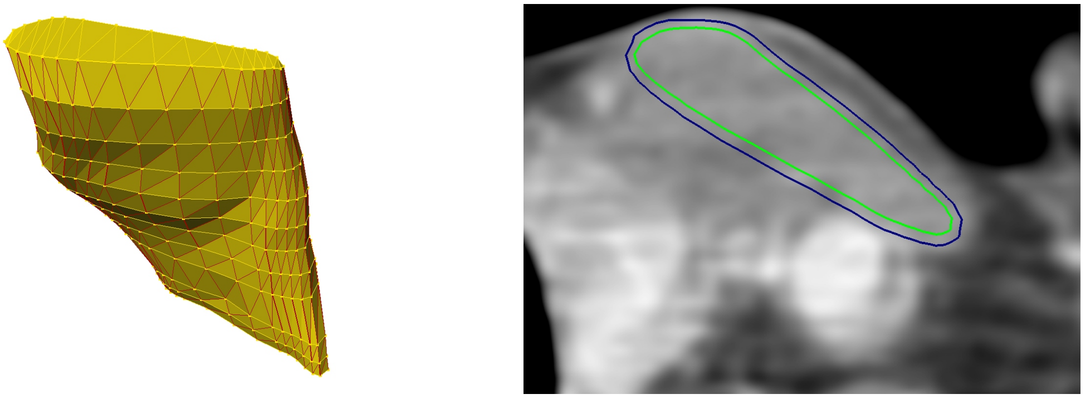
6.4.2 Re-Sampling

Down sampling was very critical in obtaining good segmentation. If the datasets were not down sampled in x and y dimensions, the local minima distracted the SMSM and it was not able to find the true muscle boundaries as shown in figure 6.3. Figure 6.3a shows the deformed 3D SMSM, while figure 6.3b shows the deformed SMSM in a single slice. Down sampling helped completely eliminate the effect of local minima and the SMSM was able to converge towards the true muscle boundary as shown in figure 6.3d. Down sampling also helped in parameter automation by allowing to have a standard spacing in x and y dimensions. It then became easier to adjust the SMSM and simulation parameters.



(a) Deformed 3D SMSM of the lower half of the muscle.

(b) Deformed SMSM in a slice



(c) Undeformed 3D SMSM of the lower half of the muscle.

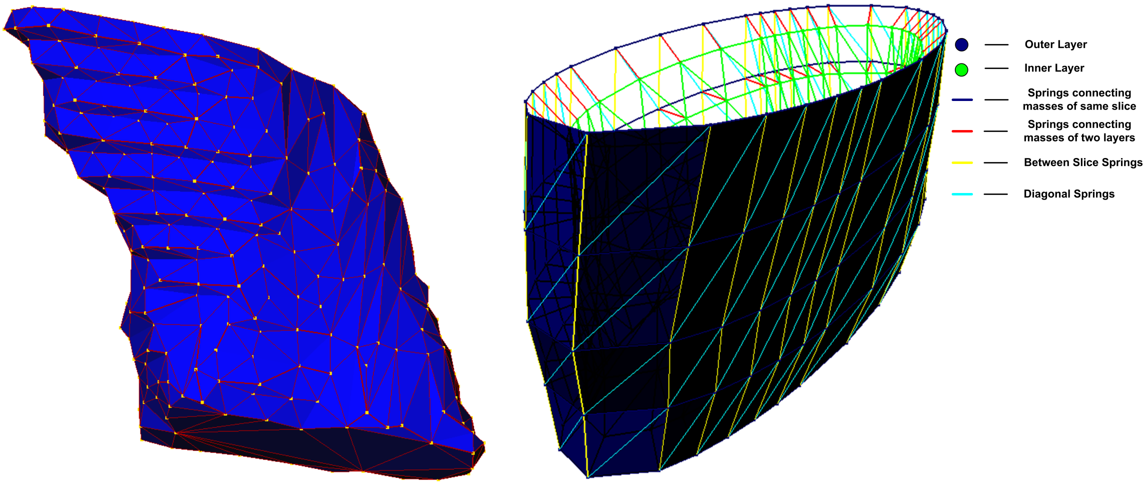
(d) Local minima problem solved by using down sampled dataset.

Figure 6.3: SMSM deforms because of local minima and its solution. Figure (a) and figure (b) shows the same slice.

6.4.3 Structure Of SMSM

The automatic mass placement facility provided by the msml library did not work here because it placed many more masses than required. The mass decimation facility provided by the msml library decimates masses according to its own algorithm as a result of which the desired structure could not be obtained as shown in figure 6.4a. The stair-case artifact visible in figure 6.4a was also undesirable.

Having a structured SMSM was important. The muscle's shape, size and position changes slowly between slices, therefore, it is easier to interpolate between the slices. The masses were placed in some of the slices and the muscle's shape, size and position between the selected slices was interpolated. If the number of masses increase 800 then the model simulation slows down, therefore, it was imperative that not all slices be used for placing the masses. This constraint also forced to select the slices carefully and to have a structured SMSM carefully constructed so as to minimize the constraint's effects as shown in figure 6.4b(see sections 4.2.4, 4.3.1, 4.4.1 for details on SMSM structure).

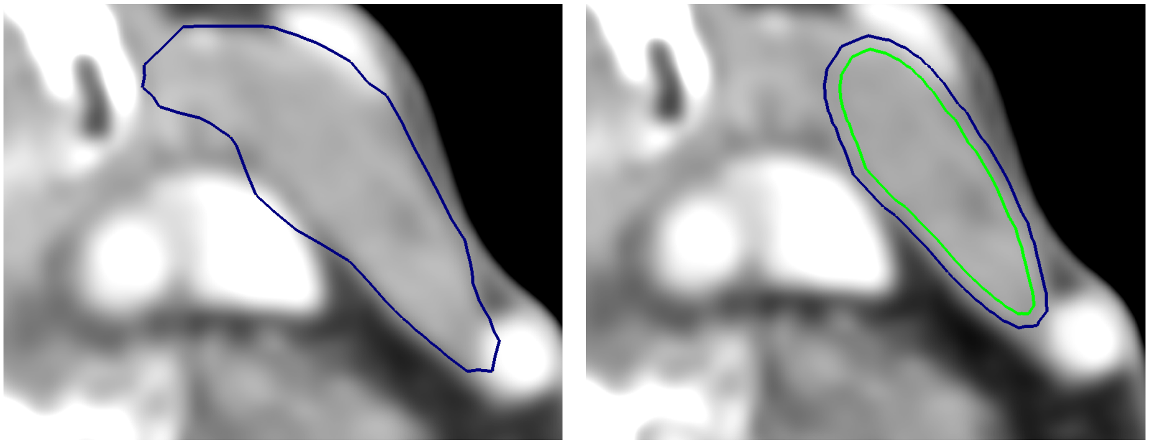

Outer Layer
Inner Layer
Springs connecting masses of same slice
Springs connecting masses of two layers
Between Slice Springs
Diagonal Springs

(a) 3D SMSM created using MSML library **(b)** 3D SMSM specifically created for the muscle

Figure 6.4: SMSM structure

6.4.4 Inner and Outer Layer of Masses

Directed edge sensors are attached to the masses of the outer layer while intensity sensors are attached to the masses of the inner layer. The inner and the outer layers are connected to each other through stiff (high spring constant) springs BL_{exp}. This combination is important even after cutting the gray values. The intensity interval which is used to cut the gray values has been deliberately kept loose so as not to cut the parts of the muscle. On the other hand the intensity sensor are configured to work within a tighter intensity interval. Therefore, the main purpose of the intensity sensors then is to stop the directed edge sensors from following wrong edges. Figure 6.5a shows SMSM fails to adapt to the true boundaries of the muscle in the absence of the inner layer. Figure 6.5b shows SMSM with both inner and outer layers adapts to the true boundaries of the muscle.



(a) SMSM with only outer layer

(b) SMSM with inner and outer layers

Figure 6.5: Effect of inner layer on SMSM adaptation.

6.5 Quantitative Assessment

Table 6.2 give the quantitative evaluation of the results. Results of 10 datasets have been presented here. The comparisons are with respect to the gold standard. There are some interesting trends that can be observed.

The values in the Tanimoto coefficients represents the similarity score between the gold standard and the SMSM segmentation. A value of 1 says that the two segmentations are exactly similar while a value of 0 says that they are completely dissimilar. The average value for the similarity between the gold standard segmentation and the SMSM segmentation is 0.750. On the face of it, an average similarity score of 0.750 looks not very good but the qualitative results are much better than what the score portrays.

The actual results are much better because the humans tend to over segment the target object while the model is more biased towards under segmentation. It is because we have used the resampled the datasets which blurs the boundaries of the

	Tanimoto Coeff.	Dice Coef.	Over Seg.	Under Seg.	Hausdorff Dist.(mm)	Average Dist.(mm)
Dataset 1	0.876	0.934	0.042	0.087	4.123	0.489
Dataset 2	0.715	0.833	0.009	0.279	5.000	0.759
Dataset 3	0.817	0.899	0.017	0.169	6.107	0.811
Dataset 4	0.731	0.845	0.011	0.260	4.690	0.931
Dataset 5	0.621	0.766	0.004	0.377	6.403	0.947
Dataset 6	0.767	0.868	0.087	0.166	6.083	1.025
Dataset 7	0.691	0.817	0.021	0.294	6.403	0.996
Dataset 8	0.758	0.863	0.010	0.234	5.099	0.914
Dataset 9	0.752	0.858	0.028	0.227	7.483	1.099
Dataset 10	0.773	0.872	0.017	0.213	7.550	0.904
Average	**0.750**	**0.856**	**0.025**	**0.231**	-	**0.888**

Table 6.2: Results

muscle. Though this blurring is good from the SMSM simulation point of view but it has its disadvantages too. The blurring destroys the sharp edges. Since the model is placed inside the muscle and then it grows out towards the muscle boundaries due to the blurred edges the model stops just short of the real muscle boundary. This makes the model biased towards under segmentation. For human the observable distance between the actual muscle boundary and the boundary found by he model is ignorable. The model's tendency to under segment and the humans tendency to over segment the target object adds up to numbers that does not look very good. Therefore an average similarity score of 0.75 is much better than what it actually portrays. The SMSM is always placed inside the muscle for reasons explained in section 2.3.

The Dice coefficient is similar to the Tanimoto coefficient. As said earlier that the model is biased toward under segmentation, this fact could be ascertained by looking at the columns *Over Segmentation* and*Under Segmentation*. A value of 0 says that there is no over segmentation while a value of 1 says that there is 100% over segmentation. An average over segmentation value of 0.025 and the rest of the values in the *Over Segmentation* column shows that there is almost no over segmentation. On the other hand we can see a strong under segmentation trend for reasons explainded earlier.

The last two columns provides the *Hausdorff Distance* and the *Average Distance*. The Hausdorff distance is generally larger than the Average distance. Looking at the Hausdorff distance which is between 4.1mm and 7.5 mm says that there is a big difference between gold standard segmentation and the model segmentation but it is only 1 mass point which is that far away. If we look at the Average distance which is between 0.49mm and 1.1mm we can see that there is not much difference between the gold standard segmentation and model segmentation. The average error between the gold standard segmentation and the model segmentation for all datasets is 0.888 which is very much acceptable. Appendix B shows some of the segmented muscles while appendix C shows some of the 3D SMSM meshes.

Tabel 6.3 shows the time required to segment each dataset. The first column *Stroke + Simulation Time(min)* shows the combined time taken for entering strokes and simulating the SMSM model. The second column *Voxelization Time(min)* shows the time taken for segmenting the muscle from the dataset. The third column *Total Segmentation Time(min)* shows the total time taken from entering the strokes till the obtaining the segmentation. The fourth column *No. of Models* tells about the number of models used for segmenting the muscle. If the muscle is segmented using two different types of SMSMs then the total time for segmenting such muscles is higher than the muscle which is segmented using only one type of SMSM model. The two different SMSMs are simulated one after the other which is the main factor in increasing the total segmentation time. There is another factor *the simulation stopping criteria* which affects the total segmentation time. If the model reaches the balanced state before the simulation step count reaches 500 then the model stops simulating thus decreasing the total segmentation. On the other hand, if the model simulates for 500 iterations then it results in increasing the total segmentation time. These are the main reasons for the variations in the total segmentation time.

`Datasets`	`Stroke + Simulation Time(min)`	`Voxelization Time(min)`	`Total Segmentation Time(min)`	`No. of Models`
Dataset 1	5.4	2.0	7.4	2
Dataset 2	2.5	2.5	5.0	1
Dataset 3	4.5	1.6	6.1	2
Dataset 4	3.5	0.5	4.0	2
Dataset 5	3.9	1.1	5.0	1
Dataset 6	6.0	2.9	8.9	2
Dataset 7	4.2	2.7	6.9	2
Dataset 8	3.6	2.0	5.6	1
Dataset 9	5.6	1.5	7.1	2
Dataset 10	6.5	1.4	7.9	2

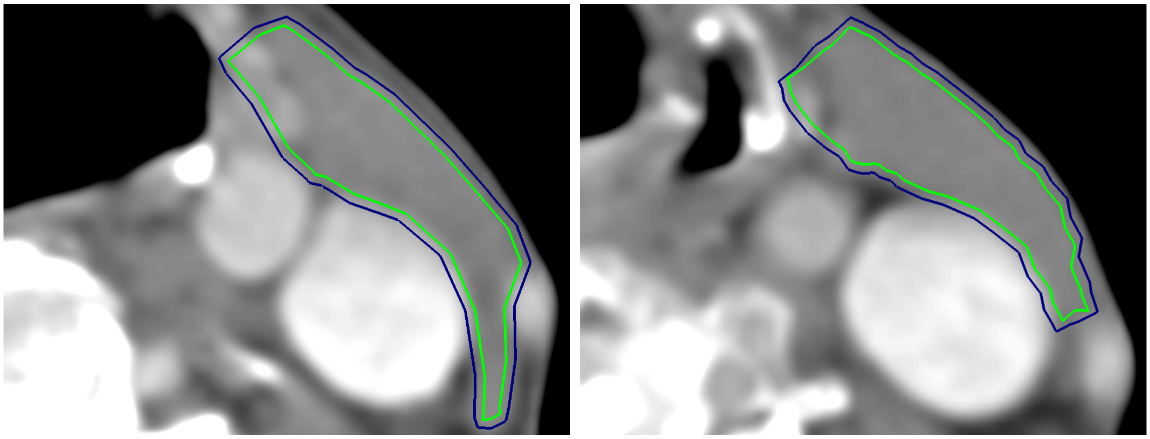
Table 6.3: Time required to segment muscle from datasets

6.6 Unsuccessful Segmentation

This section sheds some light on the problems faced during the segmentation of the muscle. The two problems arose because of weak edges and artifact presence.

6.6.1 Weak Edges

SMSM was able to segment the muscle in all of the datasets. As mentioned earlier that the segmentation was generally good but in few datasets, however, there were few slices in which the SMSM was not able to find the true muscle contour. In these cases there was a very weak edge between the muscle boundary and the neighboring structure. The segmentation overall in these datasets was satisfactory but its only in these particular regions that the segmentation was not good. Figure 6.6 shows two slices of one dataset where the SMSM went over the true muscle boundaries. Possible reasons could be the low contrast of the dataset and high sensor force and low spring and torsion forces. Figure 6.7 shows another dataset where the SMSM went over the weak edges. Contrast in this case is better but it is strong sensor force that pushed the SMSM over the weak edge.



(a) **(b)**

Figure 6.6: Failed segmentation in a dataset. The SMSM went over the weak edges that marks the true muscle boundary.

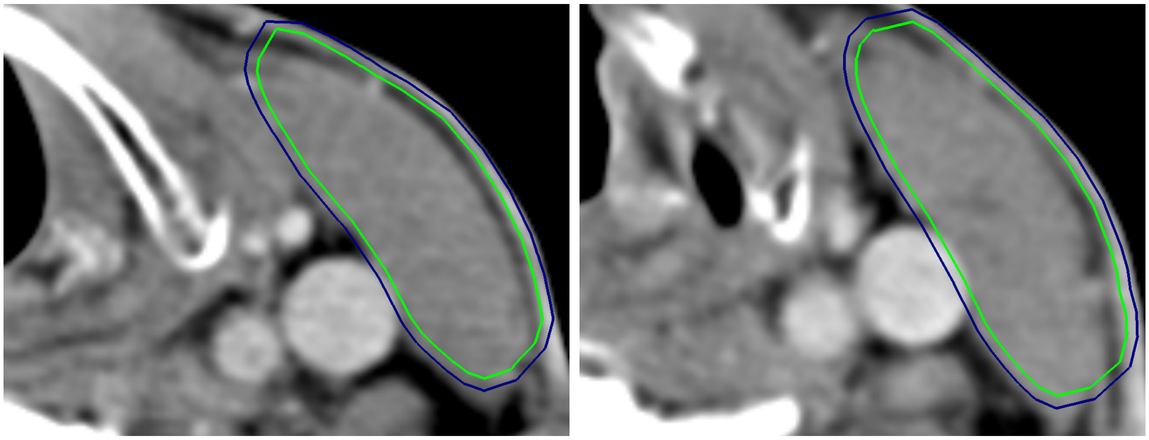
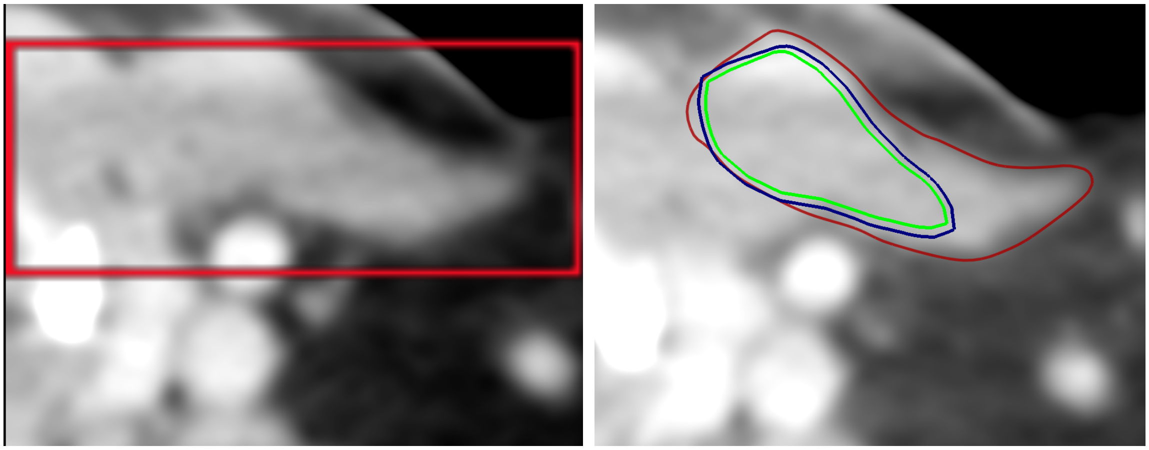

(a) (b)

Figure 6.7: Failed segmentation in another dataset. The SMSM went over the weak edges that marks the true muscle boundary.

6.6.2 Problems due to Artifact

A very small portion of the starting point of the middle part of the muscle in a couple of datasets is affected by motion artifact. The effect of the artifact is that it darkens the muscle interior in comparison to muscle appearance in other slices. This darkened muscle interior has been marked by a red rectangle in figure 6.8a. Since only a small portion of the dataset suffers this artifact, therefore, the estimated muscle intensity interval fails to capture this variation. Since the intensity sensors are configured with respect to the estimated intensity interval therefore they fail to grow out towards the true muscle boundaries. This effect can be seen in figure 6.8b where SMSM aligns itself with respect to false edges. The red contour in figure 6.8b demarcates the true muscle contour while the dark blue and green contour represents the outer and the inner layers of the SMSM. Even the re-sampling of the dataset and the push from the masses in the adjacent slices fails to minimize the effect of this artifact.

One solution to avoid such problem is to enlarge the SMSM from its estimated size, learn the shape and the size of the enlarged SMSM and then shrink the enlarged SMSM to its original size. Decrease the weight of the sensor forces in comparison to spring and torsion forces and simulate. The effect of the decreased sensor forces will be that they will fail to align with false edges and the strong spring and torsion forces will push the SMSM out of this local solution as they try to attain their learned shape and size.



(a) Dataset affected by motion artifact **(b)** Problem due to motion artifact

Figure 6.8: Problem due to the presence of an artifact in the slice.

6.7 Summary

In this chapter, the technique developed to segment the sternocleidomastoid muscle has been evaluated. The chapter starts by providing details about the test environment. Then a brief discussion about the parameters have been given and their importance and purpose have been highlighted. A brief overview of the comparison coefficients that are used for comparing our results with the gold standard have been given. A qualitative assessment of the our results tries to identify the important features of technique like cutting of gray values, re-sampling the dataset, the structure of the SMSM and the two layers of the model. Quantitative assessment follows where the results of the actual comparison with the gold standard have been provided. It has been justified why an average similarity score of 0.75 is much better than it actually looks. In the end a discussion about the problems faced during the segmentation is presented and the reasons creating the problem have been identified.

Chapter 7

Conclusion

The last chapter presented and discussed the results. This chapter tries to conclude and discuss the overall work. Some guidelines for extending this work has been presented in the section 7.2.

7.1 Discussion

The purpose of this thesis was to segment the sternocleidomastoid muscle from the CT dataset using the Stable Mass Spring Model (SMSM). This segmentation was to help orthopaedists, biomechanicians and kinesiologists in visualizing and navigating the muscle. Image based segmentation techniques were found to be inadequate for segmenting these soft tissues especially the sternocleidomastoid muscle. Segmenting this particular muscle was a challenging task and due to some inherent deficiencies of the image based techniques, they can not live up to the expectations. In regions where the muscle is adjacent to another muscle there is no visible edge or in tumor affected regions the true muscle boundary is destroyed thus forcing all edge based techniques into oblivion. The intensity interval of anatomical structures overlap rendering all threshold based techniques useless.

The dexterity of the SMSM to combine shape and size information together with the local image information was thought to be ideal for segmenting soft tissues from the CT datasets. The SMSM uses edge and intensity information to locate the visible edges and using the shape and size information, it tries to bridge the invisible or destroyed edges. The SMSM deforms to encompass the muscle shape but the deformation is kept in check by the encoded shape and size information. This constrains the SMSM from aberrant deformations.

Three different SMSMs were used to encode the specific shape and size information since there is no one model general enough to encompass all shape variations. The user decides for himself which type of SMSM is best suited to segment the muscle.

The shape cues are provided by user's strokes. The stroke corresponds to the medial axis of the muscle shape. The strokes are also used for estimating the intensity interval of the muscle. The user does not have to adjust the SMSM and simulation parameters as they were adjusted during the SMSM creation. However, the user has the freedom of varying the parameters within a certain interval if he wishes to speed up the simulation. The interval values have also been provided.

It has been identified that there are four aspects of the developed technique that are crucial for the successful segmentation of the muscle. A very high gradient outside the muscle boundary was responsible in destabilizing the SMSM. It was solved by cutting all the gray values outside the muscle intensity interval. Resampling of the dataset was important, primarily to minimize the effects of local minima and secondarily for the better simulation of the SMSM. Since there was an upper limit constraint specifically self applied on the total number of masses for segmenting the muscle, therefore, it was critical to have a well defined structure making use of the masses intelligently. Lastly, the two layers namely the inner and the outer layers were important were important. These layers complement each other and no single layer is resourceful enough to segment the muscle by itself correctly.

However there were some problems too. In a couple of datasets the SMSM was not able to find the true muscle boundary and went over the weak edges. It is either because of the low contrast or because of high sensor force and low spring forces. An artifact found in a small region of the muscle between the lower and the middle part created problems in obtaining a good segmentation.

7.2 Future Work

The prototype shows that the muscle can be successfully segmented from CT datasets using the Stable Mass Spring Model. However, since it was a prototype, therefore, few enhancements could still be done which are presented below.

1. **Automate SMSM Template Selection.** There a three different SMSM templates and the choice of the selecting the correct SMSM template has been left upon the user's discretion. The model selection process could be automated.

2. **Better Estimate of the Junction.** In the upper part of the muscle the elliptical part of the muscle is connected to the rectangular part of the muscle. This junction point right now is only roughly estimated. Currently the rough estimate of the junction does not create any problems and the SMSM adapts nicely to the muscle shape but it is desirable to have a better estimate of this junction. It will give a more accurate estimate of the junction position between the ellipse and the rectangular part.

3. **Re-Adaptation on Higher Resolution.** It would also be nice to see how the model behaves on the original dataset after it has been initially adapted/simulated on the re-sampled dataset. After simulating on the re-sampled dataset the SMSM adapted shape needs to be marked as in a state of equilibrium (*shape learned*) and then simulate the SMSM on the original dataset. This might increase the segmentation accuracy. On the other hand care should be taken that the during the model adaptation we should not learn a model whose shape does not comply with the general shape of the muscle. If we were to learn a shape that does not comply with the general shape of the muscle then this will create an intrinsic shape defect which will then degrade the results in contrast to improving them.

4. **Eliminating Inner Layer.** Right now the there are two layers of masses, the inner and the outer layer. It would be interesting to note how the model behaves if both the directed edge sensors and the intensity sensors are attached to the outer layer, eliminating the inner layer completely. With two layers of masses the outer layer can consists of only 400 masses. If the inner layer could be eliminated, then the extra number of masses could be used in the outer layer. This will increase the number of slices containing the masses. It is better to have more slices containing the masses.

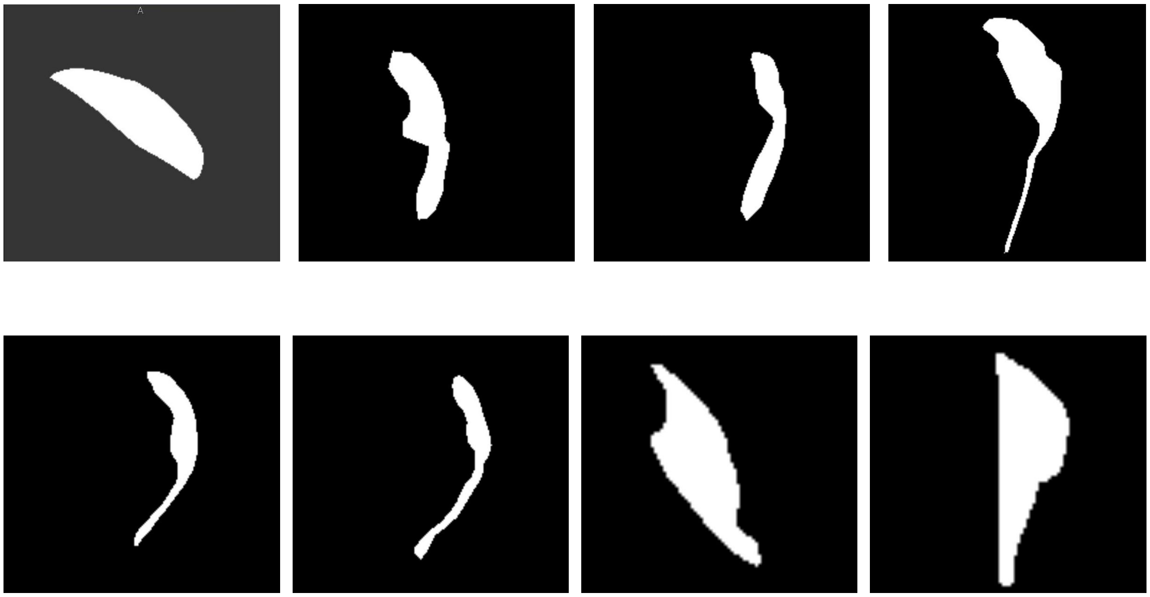
5. **Parameter Estimation.** The set of parameters works only on specific resolution of the dataset. They are not generalized and therefore are not suitable for other resolutions. Estimating parameters automatically for any resolution is a big task. There are many factors to be considered like the resolution of the dataset, the intensity profile of the image and the gradient map of the image to name few. Mathematical relations need to be found which will harmonize different aspects of the SMSM like the time step, the sensor, spring and torsion forces and balance tolerance movement. Any one factor, out of proportion will make the SMSM unstable.

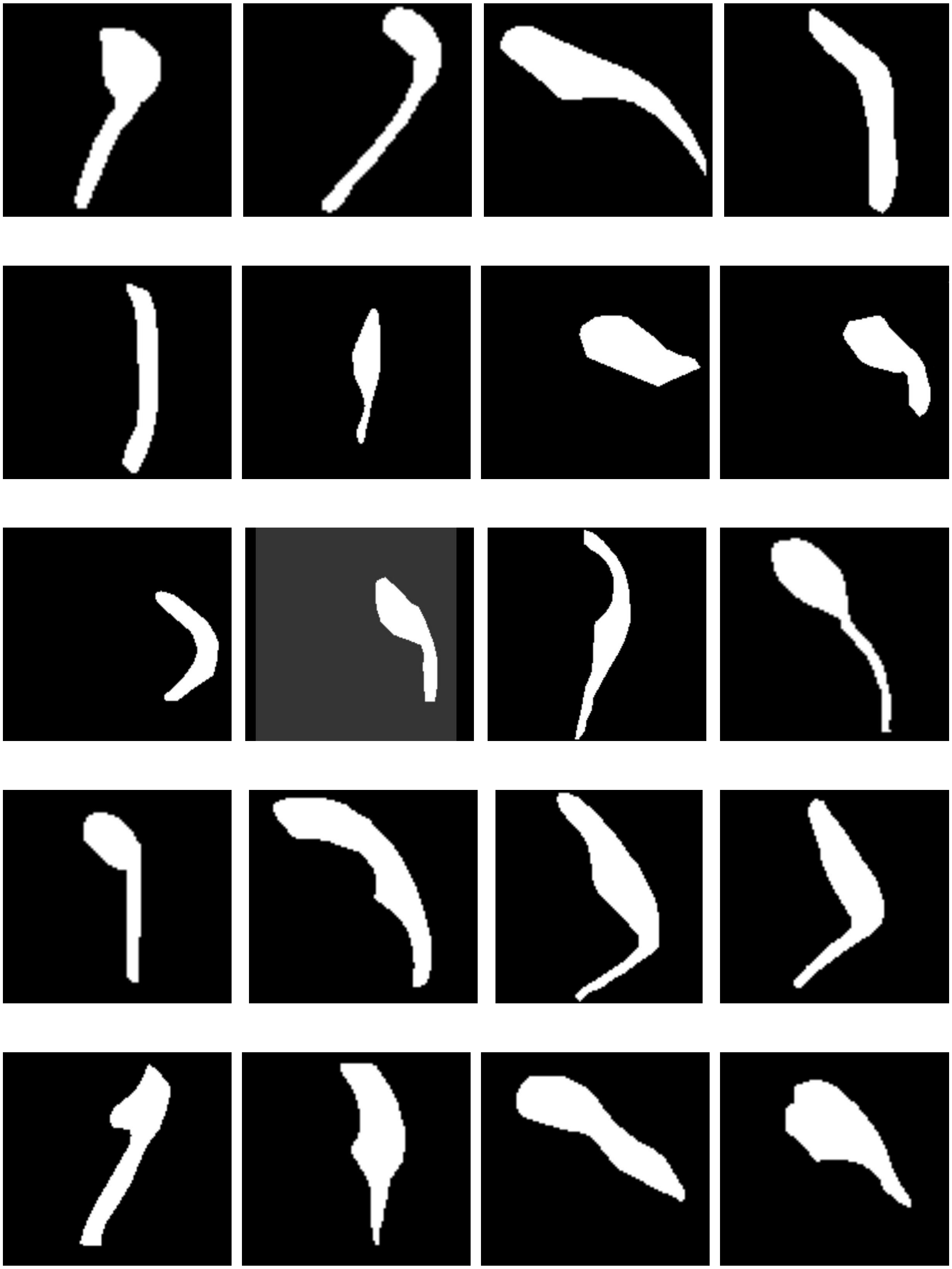
6. **Segmenting Lower Part of Muscle.** The lower part of the muscle has not been segmented in this work. The segmentation of the lower part of the muscle still needs to be done.

Appendix A

Muscle Variations

Following figures shows many of the shape variations of the seternocleidomastoid muscle found in the CT datasets. These images are manually segmented from the CT Datasets.

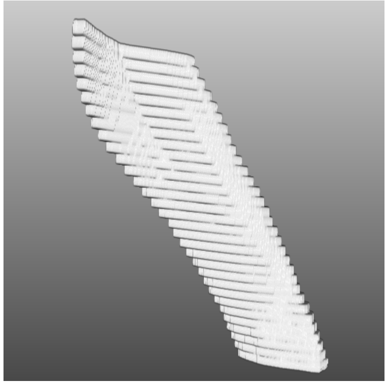
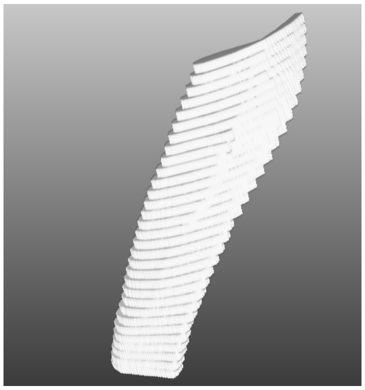
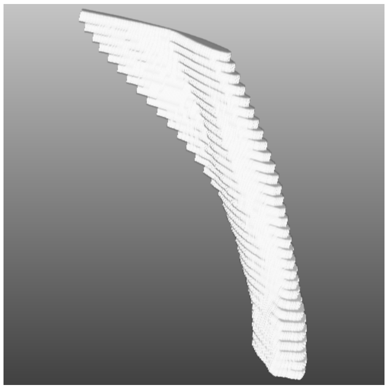
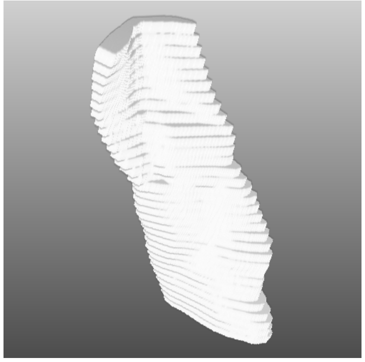
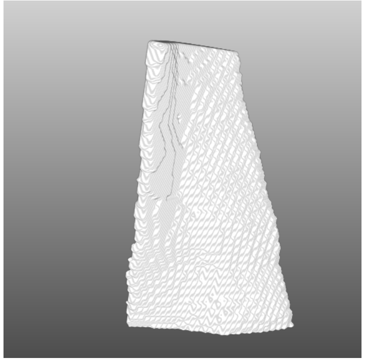
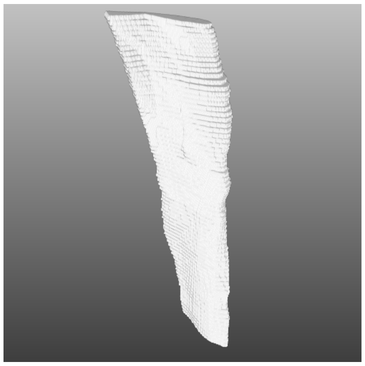


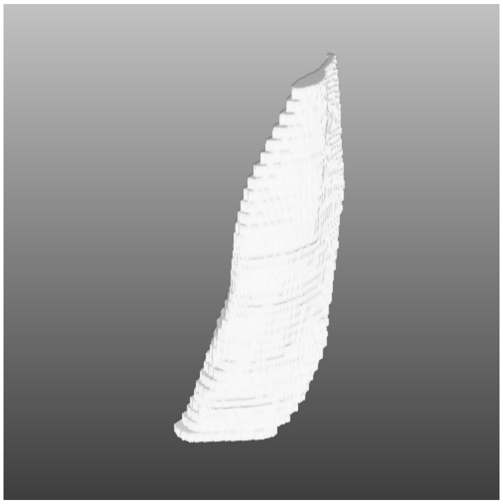


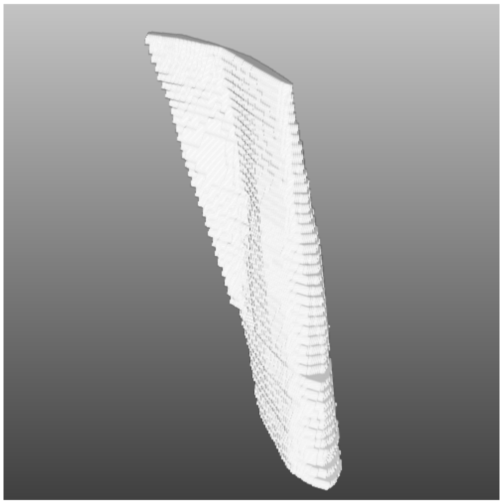
Appendix B

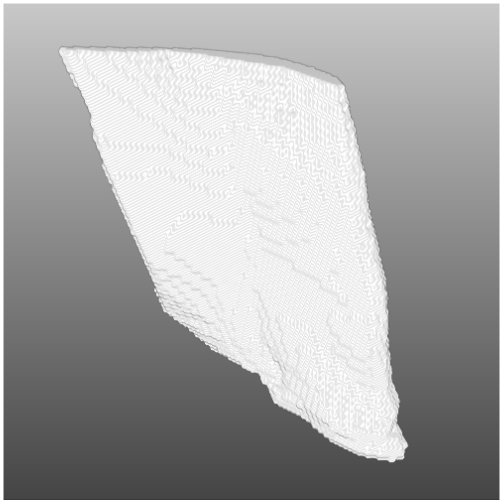
3D Segmented Datasets

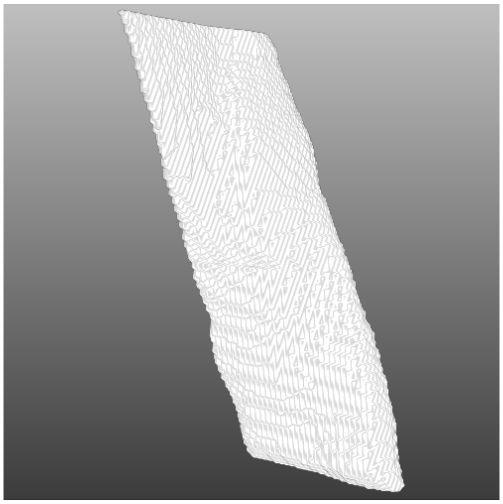
Following are the 3D Segmented Datasets.

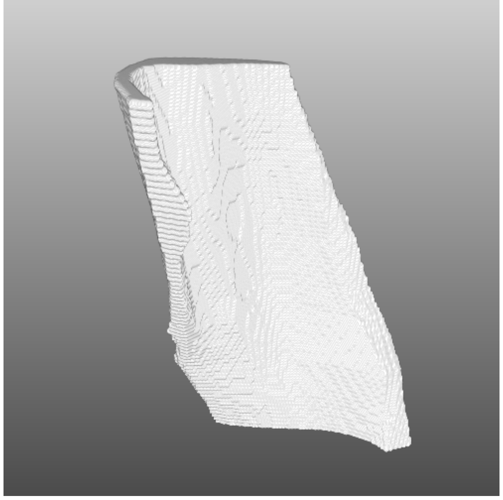







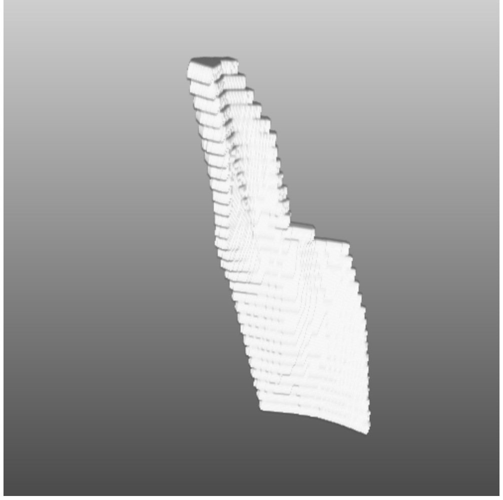








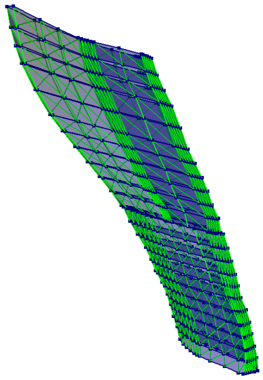


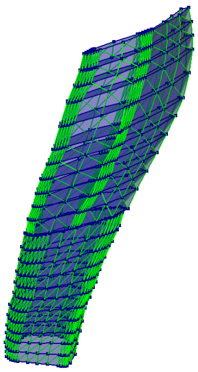


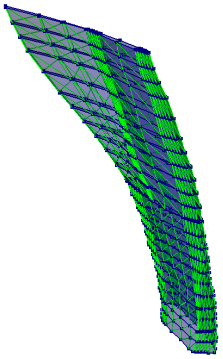
Appendix C

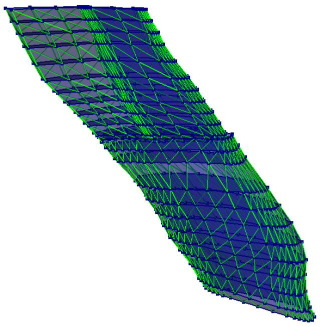
3D SMSM

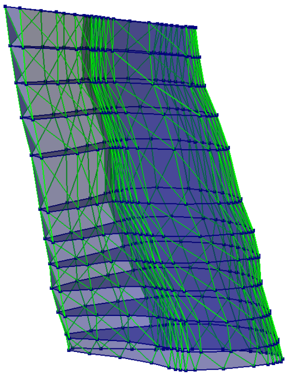
Following are the 3D SMSM.

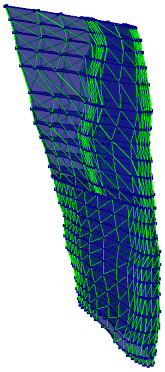


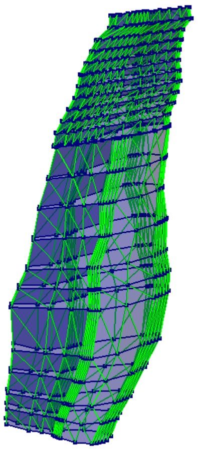


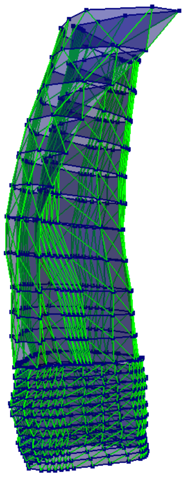


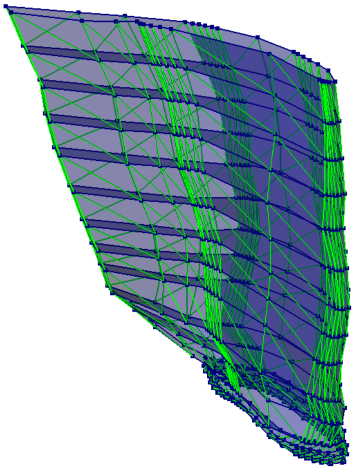


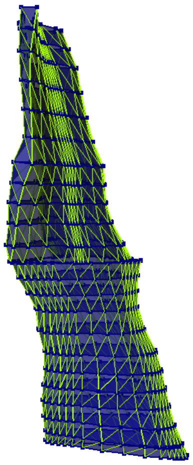


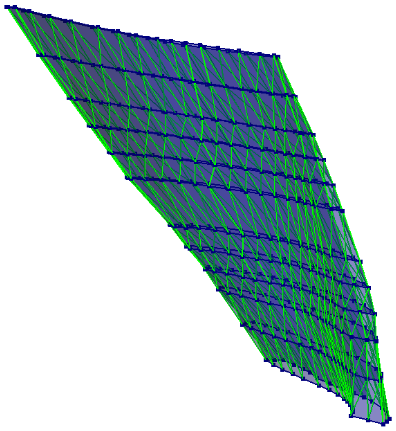


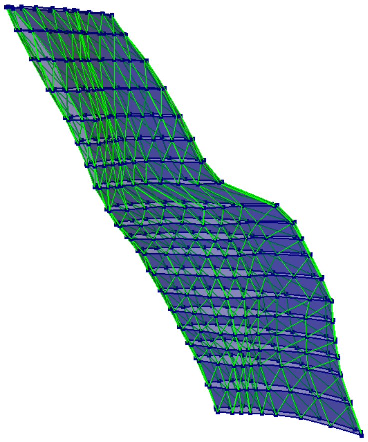












Bibliography

[1] Stephan Al-Zubi. Survey of deformable models. Technical report, Dept. of Simulation and Graphics, Otto-von-Guericke University, 2002.

[2] Stephan Al-Zubi and Klaus D. Tönnies. Extending active shape models to incorporate a-priori knowledge about structural variability. In Luc J. Van Gool, editor, *DAGM-Symposium*, volume 2449 of *Lecture Notes in Computer Science*, pages 338–344. Springer, 2002.

[3] Eric Bardinet, Laurent D. Cohen, and Nicholas Ayache. A parametric deformable model to fit unstructured 3d data. *Computer Vision and Image Understanding*, 71(1):39–54, 1998.

[4] Julia F. Barrett and Nicholas Keat. Artifacts in CT: Recognition and Avoidance. *Radiographics*, 24(6):1679–1691, 2004.

[5] S. Bergner. Structural deformable models for robust object recognition. Master's thesis, Dept. of Simulation and Graphics, Otto-von-Guericke University, 2003. http://wwwisg.cs.uni-magdeburg.de/bv/theses/thesis_bergner.pdf.

[6] Laurent D. Cohen. On active contour models and balloons. *Computer Vision Graphics Image Processing : Image Understanding*, 53(2):211–218, 1991.

[7] T. F. Cootes and C. J. Taylor. Statistical models of appearance for medical image analysis and computer vision. In *Proceeding SPIE Medical Imaging*, pages 236–248, 2001.

[8] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models—their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, 1995.

[9] Timothy F. Cootes, Gareth J. Edwards, and Christopher J. Taylor. Comparing active shape models with active appearance models. In *Proceeding British Machine Vision Conference*, pages 173–182. BMVA Press, 1999.

[10] Timothy F. Cootes, Gareth J. Edwards, and Christopher J. Taylor. Active appearance models. *IEEE Transaction Pattern Analysis Machine Intelligence*, 23(6):681–685, 2001.

[11] H. Delingette. Simplex meshes: a general representation for 3d shape reconstruction. In *Proc. of Int. Conf. on Computer Vision and Pattern Recognition (CVPR'94)*, pages 856–859, June 1994.

[12] Atam Dhawan. *Medical Image Analysis.* Wiley-IEEE Press, 2003.

[13] Jana Dornheim, Lars Dornheim, Bernhard Preim, Ilka Hertel, and Gero Strauß. Generation and initialization of stable 3d mass-spring models for the segmentation of the thyroid cartilage. In *DAGM-Symposium*, pages 162–171, 2006.

[14] Jana Dornheim, Heiko Seim, Bernhard Preim, Ilka Hertel, and Gero Strauß. Segmentation of neck lymph nodes in ct datasets with stable 3d mass-spring models. In *MICCAI (2)*, pages 904–911, 2006.

[15] L. Dornheim, K.D. Tönnies, and J. Dornheim. Stable dynamic 3d shape models. In *ICIP05*, pages III: 1276–1279, 2005.

[16] Lars Dornheim. Generierung und dynamik physikalisch basierter 3d-modelle zur segmentierung des linken ventrikels in spect-daten. Master's thesis, Fakultt fr Informatik, Otto-von-Guericke-Universitt Magdeburg, 2005.

[17] Lars Dornheim, Jana Dornheim, and Klaus D. Tönnies. Automatic generation of dynamic 3d models for medical segmentation tasks. In *SPIE: Medical Imaging.* SPIE Press, 2006.

[18] Lars Dornheim, Steffen Sauer, Erik Trostmann, Klaus D. Tönnies, and Dirk Berndt. Online generation of dynamic shape models for industrial optical quality control. Technical report, Fakultt fr Informatik, Otto-von-Guericke-Universitt Magdeburg, 2006.

[19] Lars Dornheim, Klaus D. Tönnies, and Kat Dixon. Automatic segmentation of the left ventricle in 3d spect data by registration with a dynamic anatomic model. In *MICCAI*, pages 335–342, 2005.

[20] James S. Duncan and Nicholas Ayache. Medical image analysis: Progress over two decades and the challenges ahead. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):85–106, 2000.

[21] Pascal Fua and Christian Brechbühler. Imposing hard constraints on soft snakes. In *ECCV '96: Proceedings of the 4th European Conference on Computer Vision-Volume II*, pages 495–506, London, UK, 1996. Springer-Verlag.

[22] Benjamin Gilles, Laurent Moccozet, and Nadia Magnenat-Thalmann. Anatomical modelling of the musculoskeletal system from mri. In Rasmus Larsen, Mads Nielsen, and Jon Sporring, editors, *MICCAI (1)*, volume 4190 of *Lecture Notes in Computer Science*, pages 289–296. Springer, 2006.

[23] MeVis Research GmbH. Bremen. *Germany*, http://www.mevislab.de, Accessed on 18.10.2008.

[24] Henry Gray. *Anatomy of the Human Body.* 20 edition, 1918. Retrieved from Wikimedia Commons.

[25] Horst K. Hahn, Florian Link, and Heinz-Otto Peitgen. Concepts for rapid application prototyping in medical image analysis and visualization. In Thomas Schulze, Stefan Schlechtweg, and Volkmar Hinz, editors, *SimVis*, pages 283–298. SCS Publishing House e.V., 2003.

[26] Peter Hahn. Modellbasierte segmentierung des linken ventrikels in 4d-mrt-daten. Master's thesis, Dept. of Simulation and Graphics, Otto-von-Guericke University, 2008.

[27] Lei He, Chia Y. Han, Bryan Everding, and William G. Wee. Graph matching for object recognition and recovery. *Pattern Recognition*, 37(7):1557–1560, 2004.

[28] Anil K. Jain, Yu Zhong, and Marie-Pierre Dubuisson-Jolly. Deformable template models: a review. *Signal Process.*, 71(2):109–129, 1998.

[29] I.T. Jolliffe. *Principal Component Analysis.* Springer Series in Statistics. Springer, New York, 2nd edition, 2002.

[30] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, V1(4):321–331, January 1988.

[31] P. D. Lax. *Shock Waves and Entropy.* Acadamic Press, New York, USA, 1971.

[32] Dirk J. Lehmann. Segmentierung von blutgefaeßen des halses in ct-daten mittels stabiler feder-masse-modelle. Master's thesis, Dept. of Simulation and Graphics, Otto-von-Guericke University, 2008.

[33] Jiang Liu, Tze-Yun Leong, Kin Ban Chee, Boon Pin Tan, Borys Shuter, and Shih-Chang Wang. Set-based cascading approaches for magnetic resonance (mr) image segmentation (scamis). In *American Medical Informatics Association Annual Symposium Proceedings*, pages 504–508, 2006.

[34] C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, 1999.

[35] T. McInerney and D. Terzopoulos. Deformable models in medical image analysis: A survey. *Medical Image Analysis*, 1(2):91–108, 1996.

[36] Keith L. Moore and Arthur F. Dalley. *Clinically Oriented Anatomy*. Lippincott Williams and Wilkins, 5th edition, 2005. custom.

[37] W. Neuenschwander, P. Fua, G. Szekely, and O. Kuebler. Making Snakes Converge from Minimal Initialization. In *International Conference on Pattern Recognition, Jerusalem, Israel*, pages 613–615, 1994.

[38] Bernhard Preim and Dirk Bartz. *Visualization in Medicine: Theory, Algorithms, and Applications (The Morgan Kaufmann Series in Computer Graphics)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2007.

[39] Thierry Pun, G. Gerig, and O. S. Ratib. Image analysis and computer vision in medicine. *Computerized Medical Imaging and Graphics*, 18(2):85–96, 1994. (Special Issue: Multimedia Techniques in the Medical Environment).

[40] Drake Richard, Wayne Vogl, and Adam Mitchell. *Gray's Anatomy for Students*. Elsevier/Churchill Livingstone, Philadelphia, 1st edition, 2005.

[41] Rémi Ronfard. Region-based strategies for active contour models. *International Journal of Computer Vision*, 13(2):229–251, 1994.

[42] Thomas B. Sebastian and Benjamin B. Kimia. Curves vs. skeletons in object recognition. *Signal Process.*, 85(2):247–263, 2005.

[43] Thomas B. Sebastian, Philip N. Klein, and Benjamin B. Kimia. Recognition of shapes by editing shock graphs. In *In IEEE International Conference on Computer Vision*, pages 755–762, 2001.

[44] Toby Segaran. *Programming Collective Intelligence*. O'Reilly, 2007.

[45] K. Siddiqi and B. B. Kimia. A shock grammar for recognition. In *CVPR '96: Proceedings of the 1996 Conference on Computer Vision and Pattern Recognition (CVPR '96)*, page 507, Washington, DC, USA, 1996. IEEE Computer Society.

[46] L.H. Staib and J.S. Duncan. Boundary finding with parametrically deformable models. *Pattern Analysis Machine Intelligence*, 14(11):1061–1075, November 1992.

[47] M. R. Stytz, G. Frieder, and O. Frieder. Three-dimensional medical imaging: algorithms and computer systems. *ACM Comput. Surv.*, 23(4):421–499, 1991.

[48] Pablo Suau. Adapting hausdorff metrics to face detection systems: A scale-normalized hausdorff distance approach. In *EPIA*, pages 76–86, 2005.

[49] Demetri Terzopoulos and Kurt W. Fleischer. Deformable models. *The Visual Computer*, 4(6):306–331, 1988.

[50] Alan L. Yuille, Peter W. Hallinan, and David S. Cohen. Feature extraction from faces using deformable templates. *Int. J. Comput. Vision*, 8(2):99–111, 1992.

[51] Song Chun Zhu and A. L. Yuille. Forms: A flexible object recognition and modeling system. *International Journal of Computer Vision*, 20(3):187–212, 1996.

Declaration

I hereby declare that this master thesis has been written only by the undersigned and without any assistance from third parties.

Furthermore, I confirm that no sources have been used in the preparation of this thesis other than those indicated in the thesis itself.

Magdeburg, 15.12.2008

Tahir Majeed