SHAPE RECOVERY OF VOLUME DATA WITH

DEFORMABLE B-SPLINE MODELS

by

Allen Reed Sanderson

A dissertation submited to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

The University of Utah

August 1996

THE UNIVERSITY OF UTAH GRADUATE SCHOOL

SUPERVISORY COMMITTEE APPROVAL

of a dissertation submitted by

Allen Reed Sanderson

This dissertation has been read by each member of the following supervisory committee and by majority vote has been found to be satisfactory.

Chair: Elaine C. Cohen

Thomas C. Henderson

Dennis L. Parker

THE UNIVERSITY OF UTAH GRADUATE SCHOOL

FINAL READING APPROVAL

To the Graduate Council of the University of Utah:

I have read the thesis of ___Allen Reed Sanderson___ in its final form and have found that (1) its format, citations, and bibliographic style are consistent and acceptable; (2) its illustrative materials including figures, tables, and charts are in place; and (3) the final manuscript is satisfactory to the supervisory committee and is ready for submission to The Graduate School.

Date

Elaine C. Cohen
Chair: Supervisory Committee

Approved for the Major Department

Thomas C. Henderson
Chair

Approved for the Graduate Council

Ann W. Hart
Dean of The Graduate School

ABSTRACT

In many fields today such as radiology, images of interesting structures are obtained. From these images the physician or scientist attempts to make decisions using a variety of techniques. The existing techniques for representing and analyzing particular structures include volume rendering, and surface renderings from contours, free-form surfaces and geometric primitives. Several of these techniques are inadequate for accurate representations and studying changes in the structure over time. Further, some of these techniques have large data requirements that prevent interactive viewing.

It is believed that if the structures of interest can be extracted from the image background, viewed and analyzed in an interactive setting, more accurate decisions can be made. The research described in this dissertation explores a new technique for shape recovery with deformable models using B-spline surfaces. The current literature shows that there have been many successful attempts to create deformable models but always at a loss in shape information and/or continuity. To overcome these limitations, we show that template models each having a unique topology can be developed and the template models can be joined without a loss in smoothness at their boundaries. Further, new techniques have been developed to extract and relate image data to these deformable models.

To

Bob Gregory

my earliest mentor, who introduced me to the world of computers

and in memory of my friend,

Lori Hug.

Though we were both voted “most likely to succeed” by our high school classmates,

success should not be measured by what you have done, but by who you are.

TABLE OF CONTENTS

LIST OF TABLES

Table

ACKNOWLEDGEMENTS

I would like acknowledge my committee, Drs. Elaine Cohen, Tom Henderson, and Dennis Parker, for their support and patience during my tenure as a gradual student. In addition, I would like to thank Dr. Steve Swanson for many hours of insiteful discussion on the finite element method. I would also like to acknowledge my mother, Georgiana Sanderson and especially my father, Reed Sanderson, who have given me an incrediable amount of guidience and wisdom while allowing me to freely travel down a path of my own taking.

CHAPTER 1

INTRODUCTION

With the development of Magnetic Resonance (MR) and Computed Tomography (CT) imaging techniques physicians now have the ability to view areas of the human anatomy previously not possible without invasive procedures. These techniques have led to the ability to create three-dimentional (3D) data sets from a series of orthogonal two-dimentional (2D) images and true 3D images.[1] Although these techniques have allowed for imaging of the anatomy, the ability to view and analyze these image(s) has not been very robust. This analysis includes the ability to obtain quantitative measurements such as the size and volume measurements of an organ to more complicated analysis such as modeling the blood flow through a diseased artery. The main focus of this research is to develop a tool for accurately and precisely recovering the shape of objects in an image. The principal application of this research is the human cerebral arterial system. With this tool in place, it will be possible for a person to explore the data further, performing the necessary analysis rather than simply viewing the images.

1. A true 3D image is a single image that has been formed using 3D point sampling or 3D reconstruction techniques. tThis is in contrast to a 3D data set which can be created from a series of 2D orthogonal slices th have been formed using 2D reconstruction techniques.

1.1 History

Traditionally, 3D data sets have been viewed as series of 2D slices which forces the viewer to mentally reconstruct the 3D object. During the past decade two alternative viewing techniques for 3D data sets have been reported in the literature, surface rendering [1, 50, 57, 99] and the more recent volume rendering [31, 51]. These techniques have been instrumental in increasing the vierer's ability to view the data but do not always provide the ability to further analyze the data. Volume rendering, although important for viewing, is the least able to provide information for data analysis. This is because traditional volume rendered images treat the 3D data set as a whole and use an additive projection of the 3D data onto a 2D viewing plane. The additive projection information can not be used for secondary analysis such as determining the volume of an object. Similarly, surface rendering which requires the recovery of the boundary of an object can suffer from the same problem depending on the representation used.

The type of representation used to describe the data greatly influences the ability to visualize and analyze the data. Much of the research into this area has been done by the vision community to solve the surface recovery or surface reconstruction problem. While Bolle and Vemuri [11] provide an excellent summary of many of the current techniques found in the literature, several deserve particular attention for our purposes and are discussed in Chapter 2. Traditionally, most surface/shape recovery techniques have relied upon edge following to create a contour or fitting procedures to match a surface to the data. Recently several researchers have taken a new approach to recovering surfaces from 2D and 3D data sets [22, 62, 74, 75, 87-93]. Their techniques employ deformable surfaces to approximate the data. The deformable surface representation evolves over time, at first crudely approximating an object, and then deforming to better represent an object's surface. The deformation process is controlled through a minimization process using the strength and location of the detected boundary. Several authors have simulated physical properties, such as elasticity and plasticity in their surfaces in combination with the laws of continuum mechanics as their minimization function [90-92]. Deformable surfaces with these properties also allow the surfaces to become active and respond to external forces in much the same way a real object would. This can be of great benefit if secondary operations are to be done on the surface such as recovery of nonrigid motion and structure [75, 88]. Deformable surfaces may have either open or closed forms and have been based upon finite elements [22, 87, 90-92], superquadrics with finite elements [74, 75, 87], parametric representations [93], and geometric primitives [62].

The current work using deformable surfaces typically requires the user to actively select the type of surface topology needed for a particular application. Further, the user may be required to seed the surface within the image for the shape recovery process to work correctly. In many cases, the topologies available do not represent the topology of the object. To overcome this problem, multiple surfaces have been used. Yet when combined, they still do not represent the actual topology. However, we have observed that during most shape recovery operations the general shape of the object is already known by the user and the shape recovery provides the local detail of the object. As such, it is reasonable to not only use a series of topologically different template surfaces but also have the shape recovery process select the proper topology for the surfaces reducing the amount of user interaction.

1.2 Research Conducted

The research described in this dissertation explores several new techniques for shape recovery from images using physically based deformable models consisting of multiple surfaces represented with B-splines. Underlying the B-splines is a finite element representation which allows for the deformation process. Further, a collection of topologically different templates was developed that can be joined together to form a more complex smooth model. Using topologically different templates has, however, precluded doing any type of simple object recognition other than that based on the template topology since each template may be deformed into an infinite number of shapes. The process described above can be broken down into three parts:

1. The template selection process which includes:
   Creating a set of templates using deformable surfaces that represent the basic shapes of objects we wish to recover.
   Determine which template(s) should be used in the deformation process.
2. The boundary selection process which includes:
   Application of a boundary operator.
   Thresholding to remove the weaker boundary locations.
3. The shape recovery process which combines the template(s) and the boundary data to recover the object's shape.

A flow chart of these steps are shown in Figure 1.1. Each of these parts raises questions that will be addressed as part of this dissertation. For instance, if two templates are to be used, can their surfaces be joined so smoothness is maintained along the common boundary during the deformation process? This is an important characteristic which has been not possible with the previously developed deformable surfaces. The main focus of this research has been on the creation of a set of deformable template models and the development of a technique to select the correct template for the deformation process. In ad-

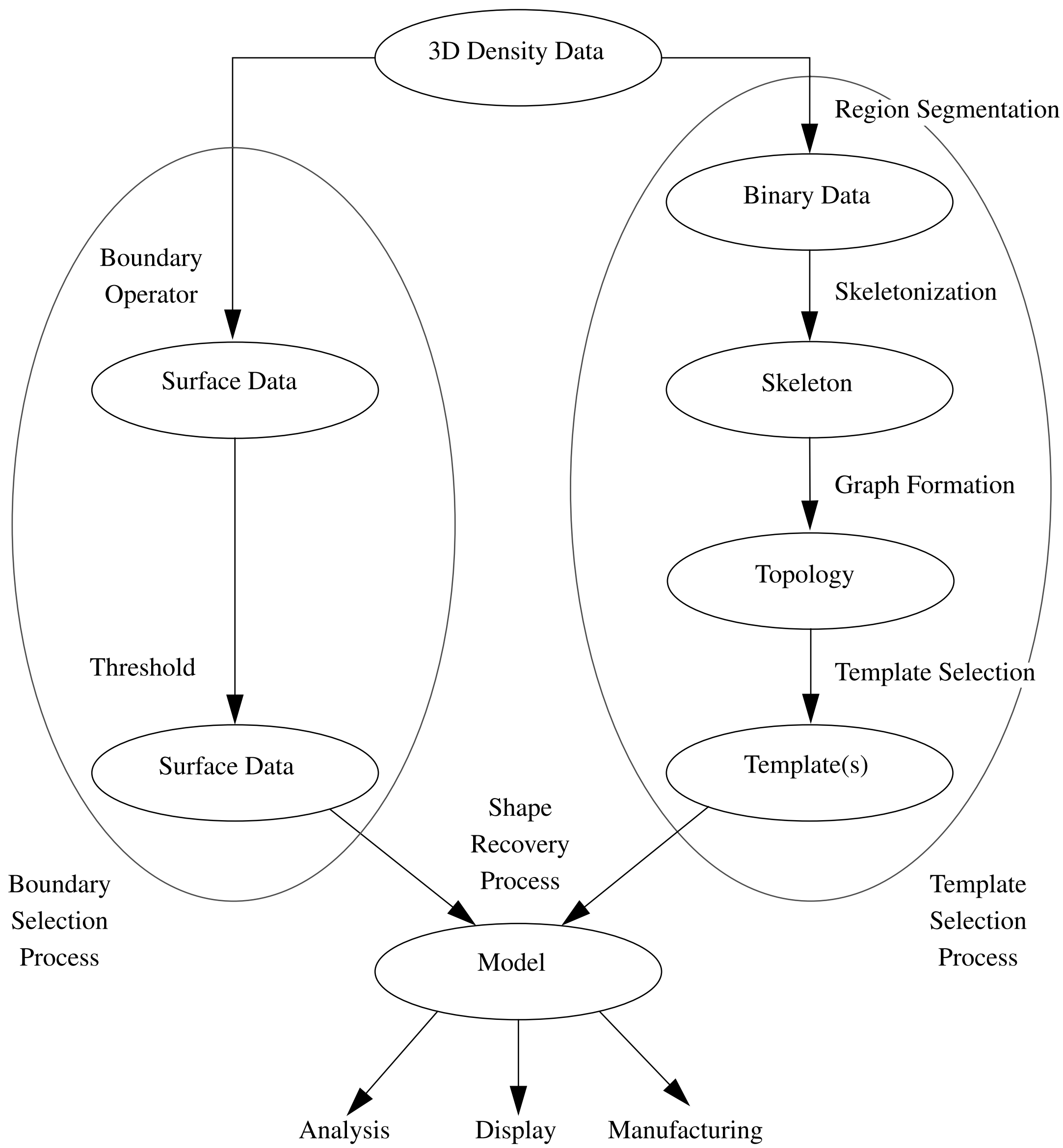


Figure 1.1. Flow diagram of the shape recovery process.

dition we have focused on the improvement of several deformation techniques and the numerical methods needed to solve them.

CHAPTER 2

BACKGROUND

Previously during the shape recovery process a bottom up approach has been taken when trying to recover the shape of an unknown object. For instance, with a 2D image, one may apply any number of gradient operators to first find the edges. To form a contiguous boundary (i.e., "stitch" the strongest edges together), a technique such as graph searching or dynamic programing has been used [32]. This technique works well when complete edge information is available but when faced with missing or noisy information it can fail. This failure is due to the technique being unable to form a contiguous boundary when information is missing or in the case of noisy information, the boundary may wander aimlessly from noise spike to noise spike.

When working with 3D data and surface fitting, similar problems may be encountered. In [11], an excellent review of the current methods is given, however, some of the more notable problems are worthy of discussion here. For instance, if the object contains concavities the initial surface placement for the fitting may be such that the concavity could be misinterpreted. Thus, a very good initial estimate to the final surface must be used. In addition, it is not always clear how the data should be mapped onto the surface when it is very irregular. Even when a mapping is obtainable, determining a minimization function can be difficult. The most prevalent error function is a sum squared differences which is minimized using an iterative method.

The above problems demonstrate that the current surface fitting techniques are not always able to give satisfactory results. The addition of global constraints, such as curvature or symmetry, help reduce the failures but at an added complexity. Instead, a dynamic process using deformable surfaces has been chosen as the technique for doing the shape recovery. This dynamic process is based on a well understood physical process which allows for the inclusion of global constraints. Further, the numerical techniques used to interactively solve for the deformations are also well understood. Deformable models, such as the ones proposed, are not without problems either, but hold more promise than other techniques proposed because of their reliance on the well developed area of finite element analysis. The remainder of this chapter is devoted to discussing the current deformable models and their advantages and disadvantages.

2.1 Deformable Models

Surfaces that are modified over time using physically based properties are called deformable surfaces and are unique in that they are able to respond dynamically to forces acting on them in a manner similar to that of a real object. A deformable surface typically begins as a simple shape such as a flat sheet, cylinder or sphere. It is then deformed according to modified rules of Newtonian or Lagrangian dynamics. The deformation can be elastic and/or plastic and is based on internal and external constraints in the form of forces. In shape recovery, the forces are based upon image potentials such as the strength of an object's boundary in the image. The potentials are converted into forces based on a heuristic function, and it is these forces which cause the surface(s) to deform. Terzopoulos, et al. first brought this idea to the computer vision community in 1987 [88]. Since then, many others have contributed to this field [61, 62, 74].

The advantage of deformable surfaces over edge linking techniques for shape recovery is that neither continuous nor complete boundary information is needed to recover the object's shape. Since the surface is continuous, it can approximate the shape where boundary information is missing. Further, depending on the internal and external physical properties applied, it is possible to obtain different results. For instance, if the internal properties such as the stiffness, are relaxed then the surface can assume a more irregular shape. Another advantage of deformable surfaces is that it is possible to use a surface with an axis of symmetry to infer 3D shape from 2D views even when the object is partially hidden [88].

Deformable surfaces may be constructed from geometric primitives, have a parametric representation or be a combination of both. The geometric primitive representation is most prevalent since there is a large body of literature on how to represent and deform objects using these primitives as finite elements. Each representation has its own advantages and disadvantages as discussed below.

Creating and using deformable surfaces have been studied by Barr [3], Miller [62], Pentland [74, 75], Terzopoulos [87-92], Thingvold [93], and others as a mechanism for 3D surface recovery and animation sequences. All of the techniques currently found in the literature are based upon physical mechanical properties except for Miller's which uses only geometric properties.

Terzopoulos [87-92], Pentland [74, 75], and Cohen [22] have advocated the use of the finite element method (FEM) in their deformable surfaces. The FEM is a standard tool used by engineers to analyze the static and dynamic behavior of objects. The advantage of the FEM is that it is very easy to relate external forces to the object and to track changes in an object through nonrigid motion, such as the motion found in a beating heart [75]. While Terzopoulos and Cohen both use a standard approach with finite elements, Pentland uses modal analysis which is computationally less expensive and allows for a closed solution to be found. Both methods however, perform poorly when complex shapes are being represented. This is because many nodes are required to adequately represent the object and thus the solution becomes computationally too expensive.

Thingvold [93] introduced deformable surfaces using a B-spline representation and Newtonian dynamics. The use of B-splines overcomes the computational expense by approximating complex objects and their physical properties. If the approximation is accurate enough then this technique is an advantage, however this technique can produce poor results if the approximation is not accurate. Thingvold models both elastic and plastic deformations in his surfaces. This allows for forces to be applied to a surface, deform it, and then be removed. This is currently not possible with some of the finite element surfaces which will contain no plastic properties and subsequently return to their “natural” or “relaxed” shape after the force is released.

Miller [62] deviated from the physically based approach, creating a deformable surface based on geometric constraints using geometrically deformable models (GDM). His technique uses a collection of triangular faces which topologically approximated a sphere. Each vertex on the sphere is moved (deformed) based on the local topology. Each vertex has a cost associated with it that must be calculated at each iteration. A local minimum is found through the use of a gradient descent method. Representing simple surfaces with GDMs may require up to 7500 vertices and 200 iterations before a local minimum is found. Miller reports that the technique is sensitive to the weighting used in the minimizing function and can cause a lack of precision in the deformation process.

The main disadvantages with current deformable surface methods being used for shape recovery are their inability to represent objects with a genus greater than one (e.g., a torus), the difficulty in mapping the image potentials to the surface, and the overall computational expense. Miller's GDMs, and Terzopoulos' and Pentland's superquadrics are genus zero. Although Terzopoulos can use a single surface to create a cylinder, torus, or sphere, it can not be used without a loss of smoothness for bifurcating objects. This is because no attempt was made to create topologically different surfaces which could be joined together.

The other disadvantage to the current deformable surface techniques is that they can be computationally expensive when used to model complex shapes. Even a simple shape such as a tooth may require several thousand vertices or nodes for an accurate representation. Although Miller describes an automatic method for local subdivision that limits the number of vertices created, the finite element techniques do not currently allow for any type of subdivision. Thus, if the original grid is too coarse, high frequency surfaces characteristics could be lost, conversely if the grid is too fine, the computational expense may be prohibitive.

Another area of concern with the current deformable models is the deformation operators. Operators that rely upon radial attractive forces have been found not to perform adequately in certain circumstances such as when the object is very symmetric or when the initial guess is very poor [23, 75]. In [23], a technique is presented that "inflates" the initial deformable surface until it can be attracted by the image boundary. This technique does help solve the problem but is an added step that the user must use to get adequate results.

As described above, there are several important research questions that if answered would improve the current deformable surfaces. These improvements include:

Using an underlying representation that allows for maximum flexibility with a minimum number of parameters.

Using multiple surfaces with smooth continuous boundaries to represent topologically different objects.

Applying new deformation operators and numerical techniques to solve for the deformations.

2.2 Surface Representations

A problem that has plagued the current deformable models has been how to represent objects in a manner that can easily be utilized for both surface viewing and also in the deformation process. Ideally, one would want a representation that could allow for maximum flexibility but with a minimum number of parameters. For example, geometric primitives [62] have maximum flexibility but may require so much information as to make them impractical during the deformation process. Superquadrics [74] that use only a few parameters fail to adequately represent complex shapes and must be broken down into smaller primitives in order to be used [87].

A goal of this dissertation is to use a representation that can accurately represent complex objects, be used during the deformation process without any conversion steps and maintain smoothness with as few parameters as possible. A brief discussion of the more popular representations is presented.

2.2.1 Geometric Primitives

Geometric primitives are the earliest representations used by researchers. In their seminal papers on marching cubes and dividing cubes, Lorensen and Cline [18, 57] showed that geometric primitives (triangles) can be used to interpolate surface data and achieve excellent viewing results without sacrificing quality. Miller and others have incorporated triangular primitives into their deformable models and have obtained satisfactory results. The main advantage of geometric primitives is that the deformation process is easily configured. However, if the surface being recovered is very complex, a very large number of primitives will be required; which is a disadvantage of geometric primitives. Another disadvantage of geometric primitives is that further manipulation and analysis can be difficult. For instance, even though specialized hardware, such as the geometry engine [17], has made their display quite rapid, other operations such as volume measurements are much more difficult because each primitive is typically considered independently of its neighbors.

2.2.2 Implicit Surfaces

Another commonly used representation is implicit surfaces. Implicit surfaces are able to describe a wide variety of shapes using a single equation. Their advantage is that it is possible to achieve a large data reduction while still representing complex shapes. Implicit surfaces are currently used when the desired shape of the object is already known and researchers are also developing techniques to use implicit surfaces for surface recovery of range data. Muraki [66] has shown that it is possible to do shape recovery using Blinn's “Blobby Model” [8], however the author admits that it is currently computationally too expensive, (some of his representations took several days to complete). Although implicit surfaces show promise due to the significant reduction in data, computational expense currently precludes their common use.

2.2.3 Superquadrics

Superquadrics are a set of parametric shapes based on quadric surfaces and solids. Superquadrics can be used to represent many different closed forms such as, spheres, ellipsoids, and cylinders although more complicated forms are possible. Pentland [74, 75], Solina and Bajcsy [82], and more recently Terzopoulos [87] have used superquadrics in the shape recovery process. Superquadrics have traditionally been used as a representation to approximate image data using a variety of minimization techniques. However, the limited number of parameters used to define superquadrics allows for only a rough approximation of the data. Terzopoulos [87] uses the basic superquadric form but then tessellates the surface into bilinear quadrilateral elements which allows him to obtain a more refined local approximation to the data. This technique does allow for better approximation but at the cost of increased complexity.

2.2.4 Parametric Surfaces

Deriving representations using parametric surfaces has been an ongoing research area since the late 1950s and has been extensively studied [7, 24, 27, 77]. There are several types of parametric surfaces that can be used for representing complex shapes. For instance, Bezier surfaces have been used for almost 30 years in the automobile and aircraft industry. There have been several systems which have been designed around parametric surfaces, including Renault’s UNISURF system developed by Bezier in the 1960s. Another system, Alpha_1, under development at the University of Utah, is based on non-uniform rational B-spline (NURBs) curves and tensor-product surfaces [35].

Parametric surfaces and, in particular, NURBs surfaces, have advantages over the other previously mentioned representations for this application. For example, NURBs can represent complex surfaces with a reduced number of parameters (when dealing with NURBs surfaces the parameters are in the form of control points and possibly knot vectors). In addition, the parameters take on the rough shape of the surface and themselves form an approximation. Lastly, NURBs have the ability to have local control over the geometry. Representing complex shapes with fewer parameters allows for a significant reduction in data while the local control allows for representing fine detail in the object where needed. The reduction in the number of parameters does, however, come with the restriction that the data are usually approximated not interpolated. Data can be interpolated, but not always with a reduction in the number of parameters. If the original approximation is not satisfactory it is possible to refine the B-spline surfaces to gain additional degrees of freedom which can be used to improve the approximation. Several algorithms such as the Oslo algorithm [20] or the improved Oslo algorithm [59] can be used to refine the surface.

Thingvold designed a system which uses B-splines as the underlying representation for the deformable surfaces [93]. However, the surfaces are deformed by using the control mesh as the finite element mesh. In principle, this technique is acceptable, if the control mesh is a good approximation to the surface or if the surface does not need to be known exactly. For animation purposes this technique is acceptable. For shape recovery it may not be since accuracy is very important.

2.2.5 Summary

Although the literature shows that there are several representations that can be used for shape recovery, many representations can only represent restricted classes of objects. For instance, it is currently not possible to represent bifurcating objects using some surfaces without using multiple overlapping surfaces or sacrificing shape information. Further, if multiple surfaces are used, there may be an undesirable loss in surface continuity and smoothness across surface boundaries. As already mentioned, geometric primitives have large data requirements to fully represent the object and are not smooth across the boundaries unless a large number are used to approximate the surface. This reduces their effectiveness for interactive viewing, data reduction, or for using the model in further analysis. A brief review of the properties of the several representations is listed in Table 2.1.

2.3 Topology Identification

Another problem that plagues the current deformable surface modeling techniques is that a one-size-fits-all approach has been taken even though apriori information about the object's shape and topology may be available. For instance, if the object, such as a tooth, which has a genus of zero, is to be recovered, then a simple sphere can be used as the basic surface. Although the sphere can deform into a tooth it may take a very long time for the surface to completely assume the shape of the tooth especially in area of the roots which extend away from the main body. Further, because the roots of the tooth extend well away from the body of the tooth the surface may not be able to fully deform, thus leaving

Table 2.1. Geometric representations and associated properties based on current implementations.

Representation	Accuracy	Boundary Smoothness Possible	Data Reduction	Multiple Topologies	Ease of Usage
Geometric Primitives	High	Yes	Small	Possible	Poor
Implicit Surfaces	Moderate	Yes	High	Yes	Moderate
Superquadrics	Low	No	High	No	Excellent
Parametric Surfaces	Mod./High	Yes	High	Yes	Moderate

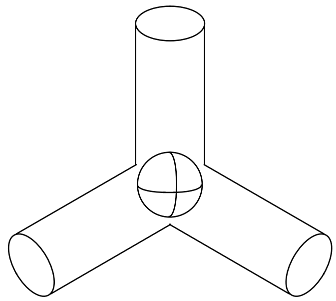


Figure 2.1. ‘Y’ Shape represented initially by a sphere.

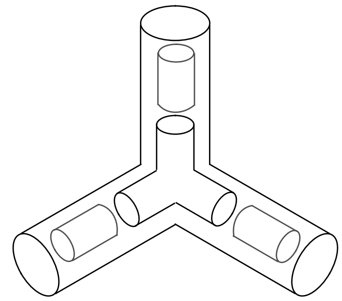


Figure 2.2. Shape represented after determining the topology.

the roots incomplete.

One goal of this dissertation is to develop a process in which the basic topology of the object could be identified automatically and used as a starting point for the deformation process. The hypothesis is that by identifying the object’s topology and a corresponding template model, a better first approximation to the object’s final shape could be obtained. Further, by identifying different topologies a more accurate geometric model can be obtained. For example, a sphere can be deformed into a ‘Y’ shape just as easily as a ‘K’ shape, (Figure 2.1). However, they are two very distinct shapes: the ‘Y’ contains a bifurcation while the ‘K’ contains a trifurcation. By simply using a sphere to recover the shape no more information about the ‘Y’ shape is gained than the ‘K’ shape. However, by first identifying the topological differences in the object, different template surfaces can be used. For instance, the ‘Y’ shape can be represented with three cylindrical templates and a single bifurcation template (Figure 2.2).

The best way to identify the topology is via a graph of the object’s skeleton. However, obtaining the skeleton requires having a binary object from which to skeletonize, thus segmentation, skeletonization, and graph formation techniques are all required in order to obtain the object's topology. This three step process is shown in Figure 2.3. A similar approach has recently been described in [42].

2.3.1 Segmentation

Segmentation techniques have been of great interest over the past three decades and are typically the first or second step in any type of image processing. These techniques can range from very simple thresholding to complicated techniques using statistical measures. Many applications use an interactively defined threshold to segment data [19]. Unfortunately, thresholding can give only crude approximations when the data are noisy and contains artifacts, which is true of medical data based on analog signals. For this research, the segmentation of anatomical structures will be used as a rough approximation of the anatomy since it will be used primarily to determine the topology.

As previously mentioned there are many ways to segment an image. Thresholding, which is a global technique, is the easiest. However, after thresholding it may be necessary

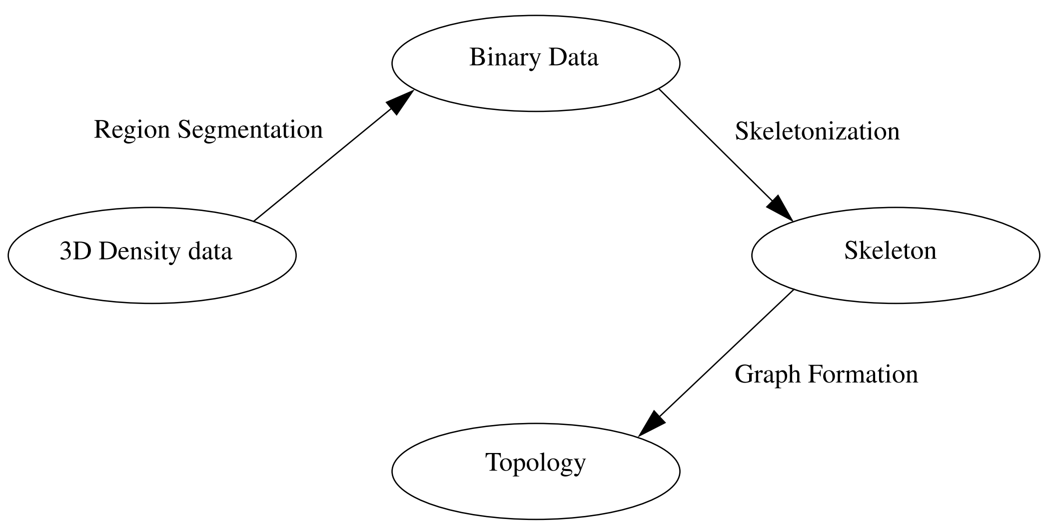


Figure 2.3. Steps used to determine the object's topology.

to label all of the connected components as being part of a particular region. Another technique is to use an edge detector and then link the edges together to form a closed region. Still further there are split and merge techniques which split large inhomogeneous regions into smaller regions while merging smaller homogeneous region into larger regions. In our particular case we are interested in finding a single well connected structure surrounded by background within the image. A natural technique to use is region growing since it can be used to segment a single region of interest and, as such, it will be the main focus of this research.

Typically region growing techniques use a local property based on regional characteristics to determined whether a voxel should be included in a region or not. On such property would be a local threshold. A local threshold would be based the gray scale characteristics for a small region surrounding the voxel under question. Using a local property is very advantageous since the region growing will be influenced only by local changes in the image. For instance, in many medical images the gray scale values can shift while still maintaining the contrast between the object and background. If this object were to be segmented with a global threshold only part of the object would be segmented whereas a region growing technique coupled with a local threshold could potentially segment all of the object. Region growing techniques using a local property assume that the histogram of the region is bimodal and that the valley between the peaks can easily be determined [2].

An interesting variation of this technique is to threshold in multidimensional space [2]. This technique uses a vector based on multispectral data, where each component in the vector is from each data source. The vector component with the most dominant peak in its histogram determines the threshold value which splits the region. This procedure is used recursively until there is no longer a dominant peak in any of the histograms. A similar approach is a classification technique based on a maximum likelihood criteria. Which uses all of the vector components together, rather than just a single dominate component. The simplest classification technique is to use a nearest-neighbor rule [2]. Classification techniques however are only as good as the features that are used in the classifier.

Another advantage of region growing is that the search space can be similar in size to the actual object space. For instance, in a 128^3 image that contains an area of interest that is 48^3 it could be computationally excessive to examine every voxel. However, if a seed location can be identified that is within the area of interest then the region growing technique might only examine 50^3 voxels rather than 128^3 voxels, a reduction of over 10^3 voxels. It has been estimated that in most applications the area of interest compromises only 10%-20% of the total volume thus region growing techniques can be advantageous.

2.3.2 Skeletonization

Skeletonization is the process of reducing an object into a “stick-figure” or skeleton. Most techniques form a skeleton by iteratively removing points (thinning) from the boundary while usually, but not always, preserving path and surface continuity between regions. Many times it is easier to identify components of an object by looking at features which represent small portions of the object, rather than by looking at the complete object. In [67], Nevatia uses the skeleton to identify limbs on biological forms (humans and animals). The skeleton provides both topological and geometrical information for the identification process. In this particular application we are interested in using the skeleton to aid in deriving only the topology although we could also use the geometric information to decide between two forms that have the same topology but are geometrically somewhat different.

The skeleton of an object can be described in terms of a medial axis transform and/or central axis. The medial axis transform (MAT) defined by Blum [10] was one of the earliest algorithms reported that produces a skeleton:

> For each point X of a set R find the a circle/sphere with center at X which is tangent to the boundary of R and does not intersect any part of it. Then the medial axis is defined as the set of all the points X having a circle/sphere which touches the boundary at least twice.

Direct implementation of this definition leads to several problems the most notable being the formation of disconnected skeletons when the original object is connected. However, it does have the advantage that the object can easily be reconstructed since a radius can be associated with each point on the axis. A less restrictive description of a skeleton would be the central axis which is concerned with maintaining only topology and connectivity. The central axis can be defined as follows:

> For each segment Y of the medial axis of a set R find those segments which are closed at each end. The central axis is defined as the set of all segments Y with both ends closed.
>
> (The central axis can also be thought of as being a local axis of inertia.)

Although the definition of a central axis may seem more restrictive since it appears that one must first obtain the medial axis then the central axis, it is not. For instance, every concavity on a boundary will produce a segment which will be part of the medial axis. These segments can easily be formed by a noisy surface and are the segments we may wish to ignore.

Depending on the criteria used many thinning algorithms produce skeletons which are central axes (they can also produce the medial axis). There are also cases were the medial and central axis are the same, such as for an ellipse. We observe that, when working with noisy data, the central axis is less sensitive to noise than the medial axis. This sensitivity is due to the definition of each axis type. The medial axis is formed based on having two tangent points on the surface whereas the central axis is only concerned with topology and connectivity. This latter property allows objects to be thinned in the presence of noise without greatly affecting the resulting axis (invariably the central axis will shift in the presence of noise).

In this research we are interested in topology as a source for directing the type of template(s) used in the shape recovery process. As such, we have directed our research efforts towards thinning algorithms which produce a central axis. For instance, [54], [95] and [44], describe 3D thinning algorithms based on local connectivity and topology. All of these algorithms iteratively use local operators to decide whether a point can be removed from the boundary. Although connectivity and topology are used in the decision process, they, alone, do not guarantee that a skeleton will be produced. In fact, if used alone, objects of genus zero would be reduced to a single point. Thus, each algorithm uses an end condition such as not removing points with a single neighbor to control the thinning. As previously mentioned, it is possible to get a medial or central axis with many thinning algorithms. For instance in [44] and [95], a central axis was obtained using the criteria that any voxel with only one neighbor (8 or 26 connected) cannot be thinned. More complicated criteria will lead to a medial axis.

For our purposes the following criteria must be met for obtaining a central axis:

1. Preserve surface and object topology.
2. Maximal thinning (single voxel thick centerline).
3. Close approximation to the true centerline of the object.

There are many techniques available that will accurately and precisely produce some but not all of these results. In addition, one must be acutely aware of the criteria being used to keep or remove a voxel or unacceptable results will be obtained.

2.3.3 Graph Search

Analogous to the way that the skeletonization drove the need for segmentation, graph formation drives the need for the skeletonization. Although the results of the skeletonization are not graphs they are very close to being graphs. All that remains is to identify the end and branch points which become nodes in the graph. By forming a graph the topology of the object is known which allows the selection of template(s) to be used during the shape recovery process.

The simplest method of converting the skeleton into a graph is to track the skeleton voxel by voxel [72]. During the tracking, each voxel is identified as being either an end, normal, or junction point. An end point is identified by having only one connected neighbor whereas a normal point has two neighbors. All points with more than two neighbors are labeled as junction points. This technique relies on having a one voxel thick skeleton and is easily used on 2D or 3D data. The tracking is implicitly defined as a depth-first-search (or level-order-search) and as such cycles will be found during the tracking.

Other methods that scan a 2D image line by line and track the perimeter of the skeleton have been reported in [84]. The advantage of these methods is that they do not need to have a one pixel wide skeleton to perform correctly. However, these methods can only be used on 2D images and have not been shown to be extensible to 3D images. In addition, line by line scanning is not very efficient when used on images containing sparse data.

One of the current drawbacks of tracking on a voxel by voxel basis is that a single voxel thick skeleton is required. Previously we have noted this criterion as part of our thinning requirements. Although this criterion can be met, the labeling technique can fail at junction points where it is possible to have a maximally thinned junction with voxels having more than two neighbors (Figure 2.4).

2.3.4 Summary

A three stage process has been outlined for the identification of the topology of a given object. This process relies on segmentation, thinning, and graph labeling. The literature contains many techniques that can be used for this particular application. Region growing techniques are most applicable for the type of segmentation needed. Algorithms proposed by Tsao and Fu and Lobgert et al., once improved, will serve as the basis for our thinning. The final step, graph labeling, has also had some research, but currently does not meet our criteria.

2.4 Deformation Operators

A surface can deform in response to a variety of forces. For instance, a surface could deform when the user introduces an external force on it, or as described below, from images. Since the focus of this research is on surfaces that use properties of physical sys-

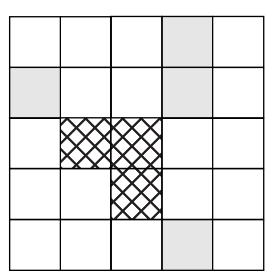


Figure 2.4. A maximally thinned junction composed of voxels each having three neighbors.

tems, the deformation operators should mimic some physical properties. One such deformation operator is a spring force, $\boldsymbol{F} = k\boldsymbol{x}$ where $\boldsymbol{F}$ is the force, k is the spring constant and $\boldsymbol{x}$ is the differential length vector from the spring's resting position. The main difficulty in using this type of operator is deciding how and where to attach the spring to the surface.

The parameters for different current deformation operators can be divided into two categories, those specified by the user and those from external data. Those forces specified by external data may originate from many sources, such as range data, monocular images, surface data, and image data. In this particular case, the ability to create forces from high contrast 2D and 3D medical images was of interest.

2.4.1 Current User Specified Deformations

Several types of operators might be appropriate to deform a surface. The spring deformation operator described above is one such operator that can be controlled by the user. For instance, the spring would "attach" between the surface and an anchor point [93]. The user could specify the location of both the anchor and attachment points on the surface. By adjusting the spring constant, k, the surface could be deformed until the desired shape is achieved. The impact of deformation on the surface from this simple system is easy to understand. The drawback is that each spring is considered independent and there may not be any interdependence between attachment points so that the desired results may be difficult to achieve.

The simplest user deformation would be to push or pull the surface at a particular location (conversely a point on the surface could be fixed at a particular location while the rest of the surface is deformed). Although there are certainly many other types of operators that could be developed for deformable surfaces we have chosen not to rely upon any user intervention in the deformation process relying instead only on image based deformations.

2.4.2 Current Image Based Deformations

This research focuses on shape recovery using high contrast image data. Image based deformations have the same effect as user based deformations but with more irregular effects on the shape. Typically the process of converting image data into potentials (forces) is accomplished using a heuristic function [62, 88]. The heuristic function used is usually a gradient or boundary detection operator that gives the probability of a point lying on the boundary of a surface. It is this probability which is converted into a potential. There are several ways in which image potentials can effect a surface. Image potentials can act as an attractive or repulsive force on the surface in the same manner that a magnet would. Conversely the image potentials can act as an obstacle and impede the surface from deforming any further.

If image potentials are used as an attractive or repulsive forces there must be a way to relate the image data to the surface. Witkin [98] introduced different ways to attach two objects together using spring forces. The attachments could be in the form of surface-to-surface attachments, surface-to-fixed point attachments, floating attachments, and others. Point attachments have the disadvantage of being only able to directly influence a single point on the surface. Although the single point on the surface will indirectly influence many other points which will in turn effect other points, it can be a slow and cumbersome way to deform a surface.

In his 2D symmetry-seeking surfaces, Terzopoulos [88] projects the 3D surface into the 2D image plane and uses the image data that is near the boundaries. For 3D image data, the surface does not need to be projected since both the surface and image data are 3D [61]. The image data then act as a radial force field to deform the surface (the forces are attracted to the nodes that define the surface). This type of image potential is referred to as having a one-to-many relationship. That is, each image potential affects many locations on the surface. This method is able to achieve good results when the object being recovered lacks symmetry, is not initially placed in the center of the image data, and is close to the final position. This technique can be considered to be the current best results for comparison.

Pentland [75] follows a similar technique but rather than create a radial force field he projects the 3D image data onto the surface using an ellipsoidal coordinate system defined by the image data's axes of inertia. The image data are then "attached" to the surface using a virtual spring model. Since image data rarely correspond directly to a node point the data are distributed to those nodes that surround the data. Further, more than one image data point can be "attached" to the surface thus a one-to-many relationship is used. Pentland reports that he is able to achieve an accurate and stable solution when the axes of inertia of the image potentials and the surface are within 15 degrees of each other. Obviously, when the initial guess is greatly different from the final solution, this is a drawback. The other drawback is that for complicated objects this calculation must be done at each iteration which can be computationally expensive. Further, as part this research, it was found that using a similar approach for distributing the forces to the corresponding nodes could cause the surface to deform past the true boundary.

Miller [62] takes an opposite approach using the image data that is intersected by a surface normal located at the vertices on the initial surface. However, this can cause problems if data is missing or ignored so that an intersection is impossible (Miller is able to avoid this problem by using a closed boundary). Further, the normal and intersection must be recalculated for every node at each iteration. Another drawback is that not all of the image data is utilized during each iteration since a one-to-one correspondence is used.

2.4.3 Summary

Several techniques have been described for deforming surfaces. All of these techniques follow physical processes such as springs or gravitational systems with one exception, Miller's, which is not tied to any physical phenomena. However, each has specific limitations which if not properly addressed will give undesirable results causing additional work for the user. Some of the limitations have been addressed but at the expense of requiring two deformation operators [23].

CHAPTER 3

SURFACE REPRESENTATION

The choice of the underlying representation was an important decision for many reasons. For instance, it is necessary to have one representation that can be used for various topologies, have a high degree of flexibility, and can be used in the deformation process. The decision was made to use only tensor product NURBs surfaces as the underlying representation since many of their shape retaining approximation properties and computational algorithms are supportive of the types of techniques we needed to create. Further, many of their properties are well understood and have been implemented for a variety of applications. In what follows is a discussion of the properties of B-splines that were exploited for this research.

3.1 B-spline Surfaces and Finite Element Meshes

B-spline surfaces are usually 3D surfaces[1] defined by three items, two *knot vectors* and their associated *orders* define the basis function, and a *matrix of coefficients,* commonly referred to as the control mesh. The control meshes will be manipulated during the deformation process. As explained in Chapter 5, the finite element based deformation

---

1. It is possible to have a 2D B-spline surface which would be considered a flat sheet. One such application of a this type of surface would be in an image warping system where continuity can be maintained while the surface is being warped.

process assumes the nodes in the finite element mesh are on the surface. However, the B-spline control mesh elements may not be on the surface, even though they approximate the shape and behavior of the surface. However, Cohen [21] and Lyche [58] show that it is possible to refine the control mesh so that it can be used as an approximation to the B-Spline surface. As such, the mesh can be changed into a finite element mesh with very little effort as described below. It should be noted that direct manipulation of the surface is certainly possible [38], but becomes impractical when thousand of locations may need to be moved.

To overcome the problem of the control mesh not lying on the surface poses, we have developed a hierarchical process where each B-spline surface is refined so its corresponding control mesh approximates the surface to within a given tolerance. Refinement of a B-spline surface is accomplished by increasing the number of knots in the knot vector and replacing the control mesh with a new mesh containing an additional row or column for each knot inserted [4, 36]. Although the new knot vectors and control mesh are different from the originals, the surface they define is the same as the original surface. Normally when refining a B-spline surface, one is concerned with where the refinement will take place since the user might want to increase the degrees of freedom in one area but not another. For this particular research, however, the amount of refinement was also of concern since the goal was to refine the surface until it could be approximated by the control mesh. Both where and the level of refinement are controlled by the definition of the definition of the B-Spline surface (i.e., the knot vectors and control mesh).

Once the B-spline surface was approximated by the control mesh, the control mesh was converted directly into a finite element (FE), mesh. This approximation did not come without a penalty. The goal of approximating complex objects with a minimal number of parameters was lost! To manage the number parameters (i.e., control points) a tolerance between the approximated surface (i.e., control mesh) and the B-spline surface was set.

The second step in the hierarchical process was done after the FE mesh had been deformed. Although the FE mesh was deformed we assumed that it was still an accurate approximation to the B-spline surface. This assumption holds if there is not a large amount of deformation which is true in our particular case. The deformed FE mesh was converted back into a control mesh and then had a data reduction operator applied to it [60]. In a similar manner to the refinement, the level of reduction was dictated by controlling the tolerance between deformed FE mesh (i.e., control mesh) and the reduced B-spline surface.

Both the refinement and reduction were previously implemented as functions as part of the Alpha_1 modeling systems and were used as needed. This hierarchical process gave the necessary freedom to fully recover the object's shape while allowing the user to dictate how many parameters would be needed to obtain satisfactory results.

3.2 Smoothness Between Surfaces

A major effort in this research was to develop a smooth deformable B-spline model composed of multiple individual surfaces. Further, we required the model to maintain smoothness across the boundaries between surfaces as part of the deformation process, (the surfaces could have been part of a single model or from different models). To create and represent such models, a mechanism that would geometrically "stitch" the surfaces together had to be created. This required that the common boundary between the two surfaces be identical, however this does not imply that the common boundary had to have the same curve representation. It does require that the representation with the fewest degrees of freedom dictated how both surfaces would be manipulated.

To obtain the necessary smoothness between the surfaces, a nonlinear constrained optimization system was developed. This technique, described below, used the minimum higher order continuity, known as tangent plane continuity, to obtain the required smoothness. By using the minumum higher order continuity and not higher, there are more degrees of freedom to work with during the joining and deformation processes. Although high order continuity was desirable, the low order continuity had sufficient smoothness to meet our requirements.

3.2.1 Tangent Plane Continuity

For two surfaces that share a common parametric boundary curve, $H(t)$ and have $C^{(0)}$ continuity, it is possible to modify the surfaces to achieve higher order continuity or smoothness mathematically. For instance, if $C^{(1)}$ continuity is needed, then the cross boundary tangents, $F'(t)$ and $G'(t)$, for each surface along the boundary curve must have the same direction and magnitude, that is $F'(t) = G'(t)$ (Figure 3.1). Higher orders of continuity are also possible. Unfortunately, enforcing such higher orders of continuity can be difficult and overly restrictive. The least restrictive higher order continuity above $C^{(0)}$ is tangent plane, or $G^{(1)}$, continuity. Tangent plane continuity occurs when the cross bound-

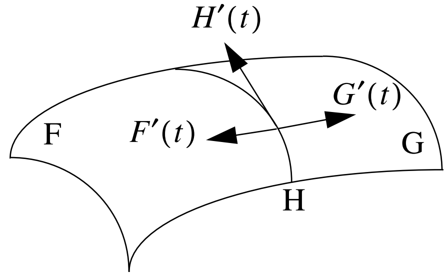


Figure 3.1. Two continuous surfaces, F and, G with their respective cross boundary tangents, $F'(t)$ and $G'(t)$. Also shown is the boundary tangent, $H'(t)$.

ary tangents, and , along the boundary share the same plane, that is, they may, but need not have the same direction and magnitude as long as they and the boundary tangent, $H'(t)$, are in the same plane

Determining tangent plane continuity between two surfaces is done by examining cosine of the angle formed by $F'(t)$ and the normal of the tangent plane containing $G'(t)$ and $H'(t)$. The cosine can be formed as the triple product of the boundary tangent, $H'(t)$ and cross boundary tangents, $F'(t)$ and $G'(t)$. If the triple product formed is identically zero for every value along the boundary then the surfaces are tangent plane continuous. That is, define $\Phi(t)$ as

$$\Phi(t) \equiv F'(t) \cdot (G'(t) \times H'(t)) . \tag{3.1}$$

Then the surfaces, F and G, meet with $G^{(1)}$ continuity along H if and only if $\Phi(t) \equiv 0$. That is,

$$\Phi(t) = \left[F'(t)\ G'(t)\ H'(t)\right] \equiv 0 . \tag{3.2}$$

We refer to $\Phi(t)$ as the triple curve since it is a triple scalar product of the curves defining the tangent planes.

In [28], DeRose described the necessary and sufficient conditions for maintaining tangent plane continuity across the boundary of two Bezier surfaces (Figure 3.2). Assuming each of the tangent curves, $F'(t)$, $G'(t)$, and $H'(t)$ can be written as Bezier curves, $\varphi(t)= \sum \varphi_j B_j^{n_\varphi}(t)$, and $\varphi' = F'$, G', or H', DeRose's condition uses a symbolic computation of the Bezier representation of $\Phi(t)$. That is,

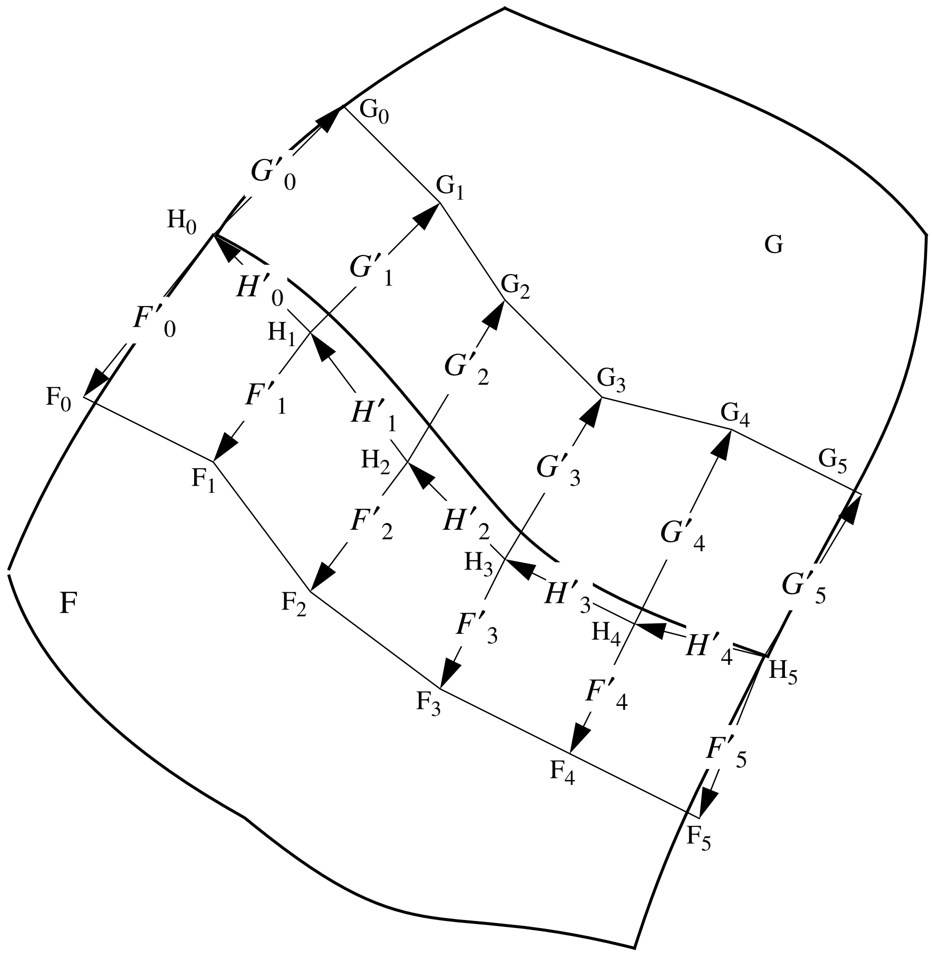


Figure 3.2. Difference vectors for two bi-quintic Bezier surfaces to be joined with tangent plane continuity.

$$\Phi(t) = \sum_{m=0}^{D} \left\{ \sum_{j+k+l=m} \frac{\binom{n_{F'}}{j}\binom{n_{G'}}{k}\binom{n_{H'}}{l}}{\binom{D}{m}} \left[\mathrm{F}_j{}' \ \mathrm{G}_k{}' \ \mathrm{H}_l{}' \right] \right\} B_m^D(t) \,, \tag{3.3}$$

where $\{\mathrm{F_j}'\}$, $\{\mathrm{G_k}'\}$, and $\{\mathrm{H_l}'\}$ are the coefficients of $F'(t)$, $G'(t)$, and $H'(t)$, respectively. The value in the parenthesis, written shorthand as Q_m, is the m-th coefficient composed of the summation of the triple products of the tangent curve coefficients, and $\{B_m^D\}_{m=0}^{D}$ are the associated Bernstein blending functions for the curve.

Since the set $\{B_m^D\}_{m=0}^{D}$ forms a basis for polynomials of degree D, DeRose observed that $\Phi(t) \equiv 0$ if and only if each coefficient of B_m^D, Q_m, $m = 0, \ldots, D$ is identically zero. That is,

$$Q_m = \sum_{j+k+l=m} \left[F_j^* \ G_k^* \ H_l^*\right] = 0, \ m = 0, \ldots, D, \tag{3.4}$$

where $F_j^* = \binom{n_{F'}}{j} F'_j$, $G_k^* = \binom{n_{G'}}{k} G'_k$, and $H_l^* = \binom{n_{H'}}{l} H'_l$. Checking to see if each coefficient in equation (3.4) is zero is very easy since they are in terms of just the elements of the control meshes.

The beauty of the coefficients in equation (3.4) is that for Bezier surfaces it is not necessary to fully evaluate the tangents. In fact, all that is needed, is to evaluate the difference between the control points along the boundary and insure that the summation of the determinants is zero. However, we are using B-spline surfaces, thus, we must reformulate the condition, $\Phi(t)$, using a B-spline basis instead to

$$\Phi(t) = \sum Q_i B_{i, \tau, \upsilon}(t) \equiv 0 \ . \tag{3.5}$$

In equation (3.5), the coefficients, Q_i, are the analytical form of the summation of the triple products of the coefficients from the three tangent curves and $\{B_{i, \tau, \upsilon}(t)\}$ are the appropriate B-spline basis sequence with knot vector, τ, and order, υ, determined from F', G', and H'. As in the Bezier case, the surfaces meet with tangent plane continuity if and only if these coefficients are all zero. Also, similarly to the Bezier case, the tangent curves can be written in the form of B-spline curves. In [65], Morken describes a technique for the multiplication of B-spline curves to obtain the analytical formation of Q_i . However, this a very expensive operation, O(n^4). As an alternative, it is possible to have $\Phi(t)$ uniquely interpolate a zero spline, which will also force the coefficients, Q_i to be zero thus insuring tangent plane continuity. This alternative is less complex and is easier to implement, which is especially important when maintaining smoothness across multiple boundaries.

All that remains is to determine the formulation of $\Phi(t)$ to use and where to evaluate. For this, we return to the original formulation of the tangent plane condition given in equation (3.2) and apply it to the two B-spline surfaces. This equation allows complete evaluation of the three tangents at different locations along the common boundary. However, in order for a B-spline curve to uniquely interpolate the zero spline it must met the Schoenberg-Whitney condition, [26]. This condition states that for a B-spline curve to uniquely interpolate a series of data points then $\Phi(t)$ must be evaluated at values spaced "synchronously" along the knots. Thus, there is a very important relationship between the knot vector and the interpolation locations. One such set of "synchronous" locations are the node values of $\Phi(t)$, which are easily calculated given the knot vector [26].

Although we are going to evaluate each of the three tangents at the nodes, they are not the nodes of the two surfaces but the node values of the triple curve, $\Phi(t)$. Assuming that the tangents have been derived from surfaces in form of B-spline curves we can obtain the node values for $\Phi(t)$ once the order and knot vector for $\Phi(t)$ has been determined. When multiplying B-spline curves together as would be done if we were to analytically form $\Phi(t)$, there are several characteristics of B-splines dictate the properties of the resulting curve [65]. First, the multiplication of B-spline curves together results in another

B-spline curve on the order of one more than the sum of the degree of each curve, equation (3.6).

$$\upsilon = n_{new} = \sum_i (n_i - 1) + 1\,, \tag{3.6}$$

where $(n_i - 1)$ is the degree of each curve being multiplied and n_{new} is the order of the new curve. Second, the continuity of the resulting curve can not be any greater than that of any of the original curves. Third, to guarantee the continuity, the new domain must have the same parametric range as the original domains and each knot value in the original knot vector must be included, perhaps multiple times, in the new knot vector. It is necessary to have the same parametric range as the original domain since the triple curve is never directly formed but is a triple product of the evaluation of the original boundary tangent curves.

Determining the order of the new curve is straightforward, since it is dictated by the characteristics above. However, we require the original surfaces to have a minimum of $C^{(2)}$ continuity which forces the cross boundary tangent curves, F' and G', to also have $C^{(2)}$ continuity while the boundary tangent curve, H', will have $C^{(1)}$ continuity. This insures that the triple curve, $\Phi(t)$, will have a minimum of $C^{(1)}$ continuity. The knot vector is formed using the original knot vector along the boundary, H, while maintaining open end conditions. To form the knot vector, while maintaining the correct continuity, we derived equation (3.7) to determine the multiplicity of each knot value in the triple curve.

$$m_{triple} = n_{triple} - (n_H - m_H)\,, \tag{3.7}$$

where n_{triple} is the order of the triple curve from equation (3.6), and n_H is the order of the boundary curve H which we assume to be the B-spline curve with the lowest continuity as defined by the knot value multiplicity, m_H. Without a loss in generality, assume the surfaces have same order, that is $n = n_{F'} = n_{G'} = n_{H'} + 1$ and the same knot vectors along the boundary with knot value multiplicity, m. This reduces equation (3.6) and equation (3.7) to:

$$n_{triple} = 3(n-1) \tag{3.8}$$

and

$$m_{triple} = 2n - 2 + m, \tag{3.9}$$

respectively.

For example, assume the mutual boundaries of two bi-quintic, $n = 6$, surfaces are being smoothed, where both have knot vectors containing one internal knot value of multiplicity, $m = 3$ ($C^{(2)}$ continuity),

$$<0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.5, 0.5, 0.5, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0 >. \tag{3.10}$$

Using equation (3.8), the triple curve has order, $n_{triple} = 15$, while using equation (3.9) the single internal knot value has multiplicity, $m_{triple} = 13$. This results in a triple curve having $C^{(1)}$ continuity whose knot vector is,

$$\begin{gathered}<0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,\\ 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5,\\ 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0>.\end{gathered} \tag{3.11}$$

Having computed the knot vector for the triple curve, $\Phi(t)$, and its subsequent nodes allows us to evaluate the triple curve at these nodes values. However, as previously mentioned, forming an analytical version of the triple curve was not practical, so we evaluate the original tangent and cross tangent curves at the node locations of the triple curve and take the triple product of these values according to equation (3.2). It is the values of the triple curve that must be zero at each of its node locations to form a unique interpolant and must be zero to insure tangent plane continuity.

At this point we diverge slightly and look at the effect of internal knots on the overall freedom of the system. Two surfaces of order n with no internal knots that share a common boundary have 9n degrees-of-freedom (n control points along the boundary[2] and n control points on each side of the boundary each having three degrees-of-freedom). This results in 3(n-1) nodes in the triple curve which must be evaluated to guarantee continuity. Thus, there are approximately three times more degrees-of-freedom than equations. Using equation (3.8) and equation (3.9), when an internal knot value has multiplicity of one and is added to the original boundary there will be 2n - 1 internal knots added to the knot vector of the triple curve. This translates into an additional 2n - 1 node locations for each internal knot. However, each internal knot adds only nine degrees-of-freedom. Thus, when the knot value multiplicity of the triple curve is greater than the degrees-of-freedom added, freedom is lost. As such, it is beneficial not to add internal knots when the order is greater than five. However, if internal knots are needed it is best to add a single knot with multiplicity, m, rather than m knots with multiplicity one, since it is necessary to replicate each internal knot value from the original knot vector in the knot vector of the triple curve.

2. In some cases the boundary was held constant which although reducing the amount of freedom allowed the optimization process to work much faster and still give acceptable results.

3.2.2 Constrained Optimization

Having developed the necessary and sufficient conditions for tangent plane continuity a technique for enforcing these conditions is needed. The tangent plane continuity conditions are really a set of constraints that must be met. If we couple these constraints with an objective function we wish to minimize, we have the classic constrained optimization problem [5, 14]. For our, problem we have chosen to use Lagrange multipliers coupled with the Quasi-Newton Method developed by Davidon, Fetcher, and Powell [30]. There is a considerable amount of literature on this subject, as such, we give no details on the implementation used, instead the reader is referred to the literature [5, 14].

Before setting up the optimization, other constraints had to be placed on the system. For instance, for our application we will have the initial surface description is in the form of the template models (which may or may not be deformed) as described in Section 3.3. As such, we need to insure the tangent plane continuity without changing the overall shape of the model. This leads to another set of constraints, «change as little as possible but meet the tangent plane continuity». These constraints are easily setup as a series of distance measurements between the original location of the control points and their final location. Thus, the penalty for not meeting the constraint is increased as the distance the points are moved increases.

The last step is to determine an objective function which can be minimized. The typical function when working with surfaces is to minimize the surface curvature or the change in change in curvature [64]. However, because we wish to change the model as little as possible (especially after the deformation process) we have elected not to use any objective function since we found have that it is possible for the minimization of objective function to overshadow the constraint satisfaction when using Lagrange multipliers. It should also be noted that the calculation of curvature is extremely time expensive. In preliminary tests, we found that a single iteration could be reduced approximately 2 orders of magnitudes when the change in curvature calculation was removed from the optimization. This also led to our decision not to use an explicit objective function.

3.3 Template Models

As previously discussed, one of the drawbacks of the current deformable models is the inability to represent objects with different topologies. For instance, the vascular system has multiple branches which cannot be modeled with only cylinders without a loss in smoothness. Furthermore, a branching network that is interconnected, such as the vascular system cannot be modeled using a closed form such as a sphere. With the ability to use B-spline surfaces, and finite elements, and to smoothly join multiple surfaces together it is possible to overcome this drawback. All that remains is to choose a series of template models that can be used for the actual shape recovery.

Two distinct topologies can be identified in our application to the vascular system, the vessel bifurcation (junction points) and the vessel body (points between junction points and end points). To represent the two topologies, two template models were developed. These templates not only represent the topology but also approximate the desired final shape. It is reasonable to have the templates approximate the final shape since in many shape recovery operations the general shape of the object is already known to some degree. For instance, if the object being recovered is the outline of the kidney, the model can easily be spherical.

The vessel bifurcation and body topologies can be formed using a variety of representations. However, some of these representation can lead to unsatisfactory results such as an uncontrollable loss in continuity or are too cumbersome to work with in a deformation process. For instance, The simplest technique for modeling a bifurcation is to join two generalized cylinders together (Figure 3.3). However, this leads to several problems, a cusp is formed between the two cylinders and the there is a section defined by both cylinders. In addition, it is not possible to topologically differentiate between a bifurcation and the vessel body.

3.3.1 Representation of Cylinders and Bifurcations

Another aspect of this research was to develop a parametric representation for various anatomical structures. Of particular interest was representing the vascular system to aid in the development and analysis of vascular images and vascular anomalies. There were two aspects to this research: the first is a representation of the vessel body and the second is the representation of the vessel bifurcation. A parametric representation of the vessel body is trivial since generalized cylinders are an acceptable solution. Two major issues have dictated the bifurcation template modeling: the surface smoothness and shape fidelity. Surface smoothness is very important since many surfaces must be joined together.

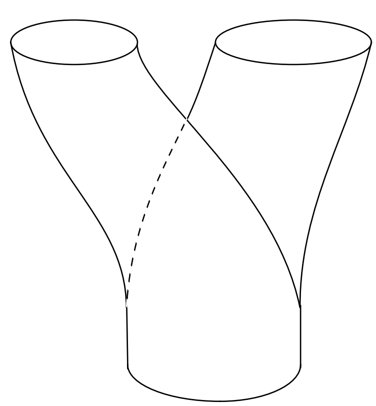


Figure 3.3. Creating a bifurcation with two cylinders.

However, shape fidelity is also very important. Shape fidelity is a very subjective term but can be simply stated as a measure of how well behaved the surface is. This measure is typically reflected in the surface curvature or the change in surface curvature, [64]. For instance, two surfaces may join with the appropriate smoothness but in order to do this the surfaces may contain many undulations. These undulations may not be desirable if the object being modeled should be relatively flat.

3.3.2 Partitioning Issues

Partitioning a cylinder is simple and has been successfully done as a single surface in the form a generalized cylinder. Partitioning a bifurcation (a.k.a. branching surface) has also been done successfully using four-, five- and six-sided surfaces known as s-patches which are generalizations of biquadratic and bicubic surfaces [56], but not with four sided NURBs surfaces alone. Further, standard four-sided NURBs surfaces have been used widely by many researchers and their properties are well known [77] whereas other variations including the s-patches are not a well understood nor easy to manipulate. There are many ways to partition a bifurcation using four-sided NURBs surfaces which we may wish to consider, three very different to partition a bifurcation are shown in Figure 3.4. We

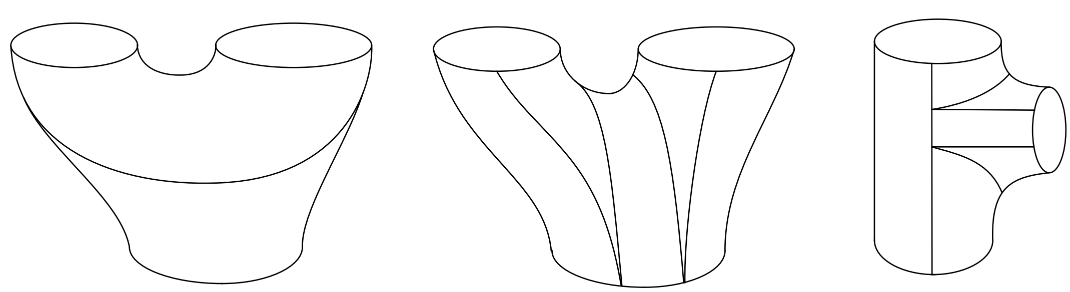


Figure 3.4. Alternate partitioning of a bifurcation.

explored these and other ways to partition a bifurcation into a series of four sided NURBs surfaces. The final partition configuration is shown in Figure 3.5. It was found that this tessellation used the fewest number of surfaces had the ability to adapt to a variety of bifurcation types as demonstrated in the results section. In addition, this partition formed a natural set of isolines (Figure 3.6)

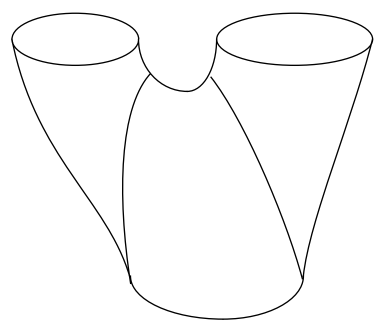


Figure 3.5. Partitioning of the bifurcation using three surfaces.

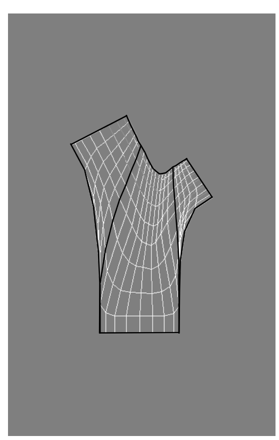


Figure 3.6. The isolines (white lines) and surface boundaries (black lines) of the bifurcation.

3.4 Summary

There are several surface representation that can be used to represent anatomical shapes. However some suffer from a lack of flexibility and control. Tensor product B-spline surfaces were chosen because of their ability to achieve a large data reduction while representing a wide variety of shapes. It has been shown that two distinct topologies can be represented using B-spline surfaces. The templates can be joined smoothly together to form complex shapes that until recently could only be crudely approximated. Further it has been demonstrated that $G^{(1)}$ continuity can be maintained between the surfaces while maintaining the overall shape fidelity.

CHAPTER 4

TOPOLOGY IDENTIFICATION

To select different template models for the deformation process, it must be possible to identify the topology of the object we wish to recover since each template model is associated with a specific topology. As previously described, there are two main topologies found in the vascular system: bifurcations and bodies. By developing a technique that identifies the topology we are also able to meet our goal of obtaining a first approximation to the object's shape. This first approximation is important since it greatly reduces the amount of deformation required to obtain the object's actual shape.

A three-step technique has been developed that allows the topology to be identified. These steps include segmentation, thinning, and graph labeling. Each step had several requirements placed on it as outlined below.

The segmentation step requires that only a crude estimate be obtained. Expensive techniques, although more accurate and precise, will not yield any added information that can be used in the topology identification process. For the segmentation, we developed a seeded region growing technique coupled with a local thresholding.

The thinning process has specific requirements that must be met if accurate results are to be obtained. These requirements are: maintaining topology, maximal thinning, and accurate centerline location. Several techniques have been reported in the literature which partially meet these requirements. For our method, we build upon those techniques that meet some of the necessary criteria.

Finally, a graph labeling technique must be used to identify the topology. A simple technique such as voxel by voxel tracking has been previously shown to produce accurate results everywhere except at junction points. We extend this technique to correctly identify junction points.

4.1 Seeded Region Growing with Adaptive Thresholding

To perform the segmentation for a single 3D MR image, we developed a seeded region growing technique using an adaptive threshold. The threshold is based on local gray scale statistics from voxels previously labeled as vessel. The comparison value incorporated not only the voxel being inspected but also several of its local neighbors. A similar but more restrictive technique is described in [6] for single 2D images. The complete implementation is described below using the assumption that the vessels of interest are brighter than the surrounding background, (the complete implementation was such that the segmentation could be performed on either bright or dark voxels).

The segmentation is done by dividing the N x N x N neighborhood surrounding an unlabeled seed voxel into two sets: S_1 contains those voxels already labeled as vessel; S_2 contains the N/2 brightest voxels which are connected to the seed voxel but not labeled as vessel (Figure 4.1). The neighborhood size, N is defined by the user and remains constant during the segmentation. A threshold value of the mean minus M standard deviations is calculated from the gray scale voxel values in S_1, where M is a user defined number of standard deviations typically ranging from 1.0 to 4.0. Only the mean gray scale value is calculated for S_2. A voxel is labeled as vessel if the mean value, $\bar{S}_2$ is greater than the

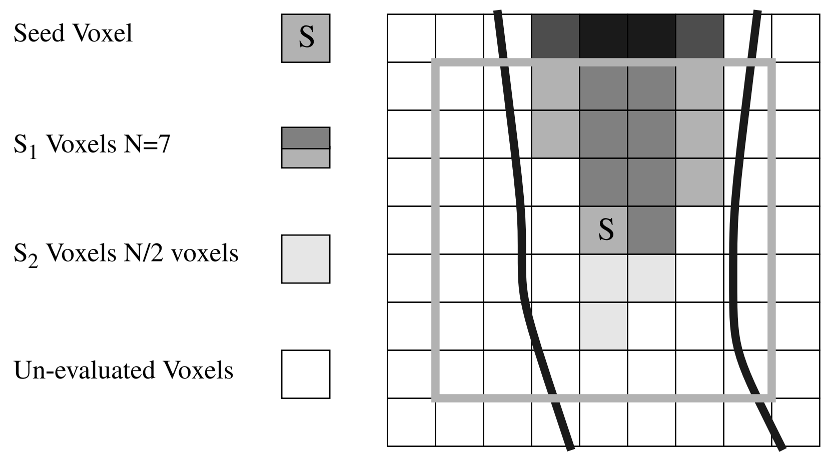


Figure 4.1. Search neighbor (N=7) for one voxel.

threshold value, $\bar{S}_1 - M \times \sigma_1$ otherwise it is labeled as background, equation (4.1).

$$Label = \begin{array}{l} Vessel \rightarrow \bar{S}_2 \geq \bar{S}_1 - M \times \sigma_1 \\ Background \rightarrow Otherwise \end{array} \tag{4.1}$$

Each voxel is subjected to the thresholding beginning with a seed voxel selected by the user. The region growing is done by placing the seed voxel's 26 nearest neighbors into a priority queue. The voxel with the highest gray scale value is retrieved from the queue and acts as the seed voxel for the next iteration. Similarly, if that voxel is labeled as vessel its 26 nearest neighbors not already in the queue or labeled as vessel are added to the queue. The region growing is repeated until the queue is empty.

A similar region growing technique is implemented for segmenting a 2D X-ray image. The only difference is that a 2D region consisting of the eight nearest neighbors is used.

4.1.1 Multimodality Image Segmentation

In many applications image information from multiple sources is available. For instance, in this research, angiographic data are available not only from 3D MR images but also from 2D X-ray images. To combine the two modalities all of the images had to be registered and placed into a world coordinate system. Further, the world coordinate system is required to have a resolution greater than all of the images. This requirement ensured that no MR voxels or X-ray pixels were skipped during the region growing process due to having different scaling factors within each image. The registration and establishment of the world coordinate system are based on the worked by Vandermeulen, et al. [97]. The complete details of this implementation and the difficulties of it are outside the scope of this dissertation and are partly discussed in [96] and [97]. However, it is sufficient to say that registration is very difficult and the results are typically not satisfactory.

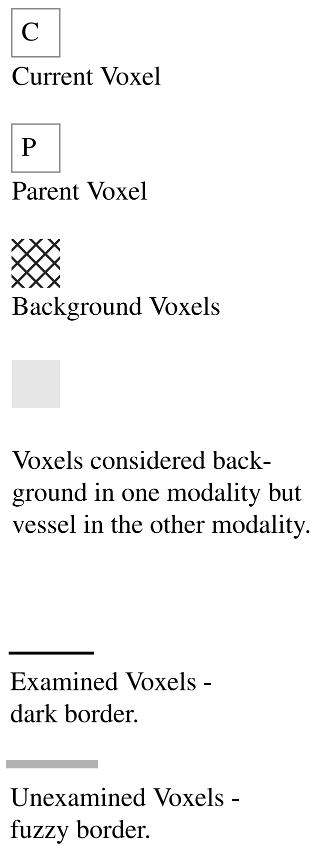
By establishing a world coordinate system which related each modality, simultaneous segmentation is possible. The segmentation is accomplished by conducting the region growing in the world coordinate system and projecting only the seed voxel into each of the 2D or 3D images. This allowed the region growing technique to capture the local gray scale characteristics in all of the images using the local image resolution. The same thresholding criterion is used in the simultaneous segmentation as with an individual image, with the exception that either the MR or X-ray image could be used to determine whether a voxel would be labeled as vessel or background.

A restriction had to be placed on the X-ray images to prevent the segmentation from growing into ambiguous vessel segments that could arise from the backprojection of multiple X-ray images. This restriction required each of the pixels in X-ray images had be labeled as vessel before the corresponding backprojected voxel in world coordinates could be labeled as vessel. For instance, with two X-ray images it is possible to choose a point that when projected into one image is vessel but when projected into the other image is background. This case would be considered to be a failure. However, failing that criterion, it is still possible to have the voxel labeled as vessel depending on the outcome of the MR thresholding.

To ensure that the region growing process would continue independent of which modality is used to label the voxel as vessel it is necessary to modify the gray scale values of those voxels or pixels labeled as background in one modality but labeled as vessel in the other. For instance, if a voxel in the MRA image is labeled as background by its thresholding criteria but labeled as vessel by the X-ray images then the MRA voxel would assume the same gray scale value as its parent voxel. Since the parent voxel had to have been previously labeled as vessel it is presumed that if the current voxel had the same value, that it too, would have been labeled as vessel. This modification caused vessels to grow into areas that if segmented using a single modality would have been labeled as background and the region growing process would have stopped prematurely (Figure 4.2 and Figure 4.3). This growing process insures that when unexamined voxels are evaluated they have a neighborhood containing at least one voxel labeled as vessel that can be used in the thresholding process.

4.2 The Central Axis

In [95], Tsao and Fu present an algorithm for 3D thinning using local transforms to first find the medial surface, which is then thinned into a skeleton. In [44], Hafford and Preston note that Tsao and Fu's algorithm can be simplified by using eight subfields rather the six primary directions (north, south, east, west, and up, down) as subfields along with



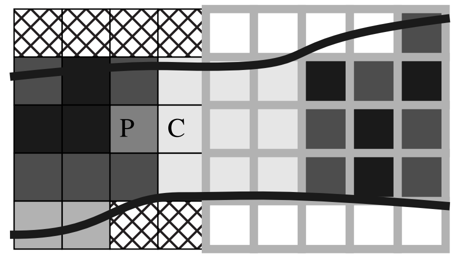


Figure 4.2. Results of single modality vessel segmentation. Note large portion of vessel is not examined because of lack of connecting voxels.

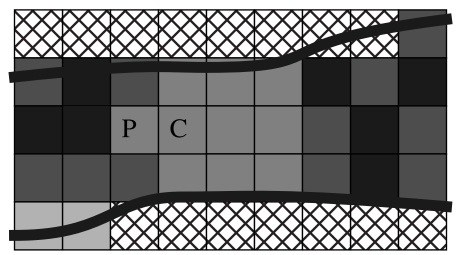


Figure 4.3. Results of multiple modality vessel segmentation. Note: the center portion of the vessel has been filled in using the gray scale value of the parent voxel and all voxels have been examined.

three checking planes (north-south, east-west, and up-down) (Figure 4.4). By changing the number of subfields from six to eight the thinning can be based on 26 neighbor or six neighbor connectivity. The subfields are labeled in such a manner that no two neighboring subfields have the same label. This change allows for each subfield to be thinned independently of it neighbors and in theory, a parallel implementation.

Only those voxels on the boundary are candidates for thinning. The boundary is defined as those voxels which are six-connected (for the 3D case, or four-connected for the 2D case) to both the foreground and the background. All other voxels are considered to be interior voxels and are not considered for thinning. This is not to say that interior foreground voxels will never be thinned. After each of the eight subfields have been examined

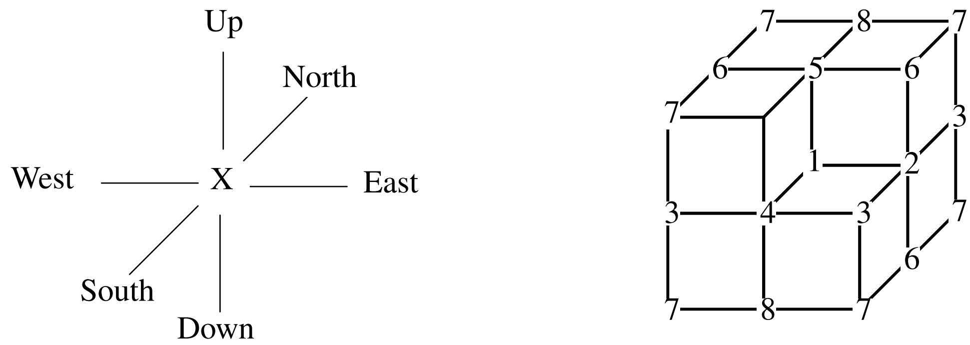


Figure 4.4. Original subfields surrounding a voxel, X, (left). The three checking planes would be in the east-west, north-south, and up-down directions. A simplification using eight subfields, (right).

and the possible boundary voxels thinned a new boundary is determined which will contain some of the voxels previously labeled as interior voxels.

The local transform used by Tsao and Fu to determine whether a voxel may be thinned is described in [54]. This transform checks the surface connectivity before and after the possible removal of a voxel. If the connectivity remains the same then the voxel is removed. However, this is a necessary but not sufficient condition and was first discussed in [94]. To determine which points may be thinned two rules must be followed:

The surface connectivity of the object and the background must be maintained.

The number of connected components must be maintained (object connectivity).

Thus, we have found it necessary to also implement a check of the object connectivity before removing a voxel, since it is possible to preserve the surface connectivity while dividing the object into two parts (Figure 4.5).

Figure 4.5. Incorrectly thinning of the central voxel from a 26 connected solid object using only the surface connectivity (object has been split into two parts).

4.3 Graph Labeling

As previously discussed, three different labels are used to identify points in the skeleton: end, normal, junction. At a bifurcation, it is possible to have up to three points each having three neighbors to be labeled as a junction (Figure 4.6). However, since it is necessary to have a one-to-one correspondence, a decision must be made as to which point is the “true” junction. This can be solved using one of two methods. First, it is possible to move the junction points so that after being relabeled only one point will have three neighbors and be labeled as a junction (Figure 4.7 and Figure 4.8). However, this technique requires having a look up table for 2^{16} combinations for the 4x4 2D case and 2^{64} combinations for 4x4x4 3D case! Although there are many symmetric cases which greatly reduce these numbers it is still impractical to implement.

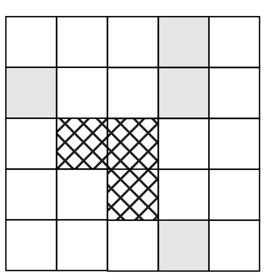


Figure 4.6. Junction with each point having three neighbors.

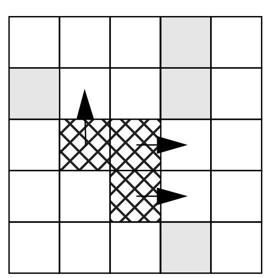


Figure 4.7. One of three points can be moved without breaking continuity.

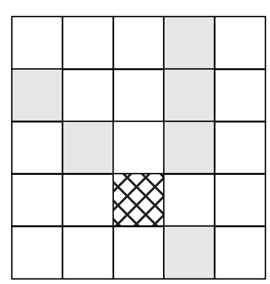
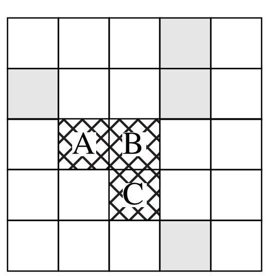


Figure 4.8. After moving the central point, only one point is labeled as a junction.

We devised a much simpler method which is based on weighting the faces, edges, and vertices of each junction voxel. For example, if a voxel shares a common face with another voxel then a weighting factor of five is assigned. Similarly, voxels with a common edge are given a weight of three, while those sharing a common vertex are given a weight of one. The weightings are set so that at least two or three edges or vertex voxels would be needed to outweigh a face or edge voxel, respectfully. (Other combination are possible but at least three or more voxels are needed). By summing the weights from each shared face, edge, and vertex together a total weight is obtained. The greater the weight, the more strongly connected the voxel is to the other voxels. The voxel with the greatest weight is considered to be the "true" junction whereas the others are defined as mutual points (Figure 4.9). This calculation is done using a lookup table with 26 different entries (six faces, twelve edges, and eight vertexes).

4.3.1 Graph Search

Once the voxels are labeled, the next step is to form branches from them which become the basis of a graph. A branch is composed of either two end voxels, or two junction voxels, or a combination of both. These voxels are defined as the nodes in the graph. Be-



Weightings:	Voxel A:	Voxel B:	Voxel C:
Face - 5	Edge - 3	Edge - 3	Edge - 3
Edge - 3	Vertex - 1	Edge - 3	Vertex - 1
Vertex - 1	Vertex - 1	Vertex - 1	Vertex - 1
	Total - 5	Total - 7	Total - 5

Figure 4.9. Weighting the junction voxels based on connectivity results in voxel B being the "true" junction with voxels A and C being mutual junctions.

tween these two voxels is a set of normal and/or mutual voxels, which are defined to be the arcs in the graph. If a branch is terminated by a junction voxel then it is defined as a parent branch with two children. The branches are constructed as part of a depth-first search (or level-order-search) of the skeleton as defined below:

Step 1: Select a root voxel and make it the first node in a queue.

Step 2: Get the next node in the queue make it the current node and voxel.

Step 3: Get the next voxel connected to the current voxel and make it the current voxel.

Step 4: Determine if the current voxel is part of the arc, a new node, or completes a cycle.

MIDDLE VOXEL or MUTUAL VOXEL (arc):

1. Add the voxel to the current arc.
2. Return to Step 3.

UNEXAMINED JUNCTION VOXEL or END VOXEL (new node/end of arc):

1. Create a new child node from the current voxel.
2. Add the new child to the node queue.
3. Create an arc between the parent and child node.
4. Return to Step 2.

EXAMINED JUNCTION VOXEL (cycle/end of arc):

1. Find the child node that leads into the junction node.
2. Update the arc between the parent and children nodes.
3. Return to Step 2.

The search begins when the user selects an end voxel as the root node. In many application there may be one or two voxels which can be considered to be the root node. In the particular case of a branching structure such as the vascular system there is typically only one source vessel which in turn supplies other vessels.[1]

---

1. This is not completely true for the cerebral vascular system. This system is a balanced system with four supplying vessels and several communicating vessels. Communicating vessels interconnect the four supplying vessels in such a manner that the circulation can be maintain in the absence of one or more of the supplying vessels. The most prominent vascular feature displaying the communicating vessels is the Circle of Willis. Topologically speaking, the Circle of Willis is a cycle.

4.4 Summary

A three stage technique has been described that allows for the determination of the topology of a given object. In the first stage, segmentation, a new technique using a seeded region growing with an adaptive thresholding is developed. The user is expected to have some knowledge for the quality (signal to noise ratio and contrast to noise ratio) of the data in order to select the appropriate region growing size and the threshold variance.

During the second stage, skeletonization, several new improvements to previously developed thinning algorithms are implemented. The improvements reduce the complexity of the thinning process into an iterative eight-stage process. There is no need for user interaction once the process begins. The user may have to slightly modify the data to remove holes or other minor artifacts to obtain accurate results.

Finally, in the third stage, graph formation, a technique to label the voxels according to the number of neighbors coupled with a tree traverse in the form of a level-order-search has been implemented. The results of this process are a graph and the identification to the object's topology. It is this topology that is used to select the template models in the surface recovery process.

CHAPTER 5

DEFORMABLE SURFACES

At this point in the shape recovery process the topology and the gross shape of the complete object can be identified and described. All that remains is to modify the surfaces so that they accurately represent the objects. Previously, in Section 3.1 we described a hierarchical technique to convert B-spline surfaces into finite element meshes. These meshes are then deformed using finite element techniques.

In this chapter the basic description of finite elements is given along with the several numerical techniques that are used to solve for the deformations. This is not the first time finite elements have been used to deform a surface. However, this is first time the notion of using finite elements coupled with B-splines has been presented. In the course of this research we made several modification to the finite element method (FEM). These modifications, as described below, were made to reduce the complexity of solving for the displacements.

5.1 Model Elements

A finite element mesh is typically composed of elements each having mass, damping, and stiffness. Although during the shape recovery process, the masses, damping coefficients, and stiffness constants used do not have any physical correlation, the system was developed so that it would resemble a “true” FEM mesh as close as possible. A basic review of the components of the FEM is given.

For each element in the mesh, the equation of motion can be written as:

$$\boldsymbol{f}_{ext} = m\ddot{\boldsymbol{q}} + c\dot{\boldsymbol{q}} + k\boldsymbol{q}, \tag{5.1}$$

where $\boldsymbol{f}$ is the external force vector (i.e., the image potentials or other forces), m, the element mass matrix, c the element damping matrix, k the element stiffness matrix, and $\boldsymbol{q}$, the element displacement vector with its respective first and second derivatives (velocity and acceleration vectors). This equation represents a second order differential equation of motion for the element. It is in matrix form because there is an equation of motion for each degree-of-freedom in the element. For this application, each element was limited to three degrees-of-freedom. The equation of motion describes the relationship between one element and each of its neighbors. Each element has three, four, or six neighbors depending on whether it is an external boundary, inside, or internal boundary element, respectively.

5.1.1 The Stiffness Element

In many FEM applications the stiffness of each member is modeled as a spring or bar element. Initially a combination of both spring and bar elements were tried. On the long axis between two nodes the member behaves like a spring having minimal stiffness, whereas in the other two axial directions the member has a much greater stiffness. This element can be thought of as a telescoping pole that can easily lengthened or shortened, but is very difficult to bend. This element was initially chosen because it allows for the transfer of moments (coupled forces) from one member to another. However, it was found that the particular deformation forces we initially chose were difficult to use. In the end, a traditional spring element having no stiffness in the cross axis was chosen. Using this element greatly simplified calculating each of the spring force components when the member was arbitrarily oriented since it was necessary to calculate only one axis.

5.1.2 The Damping Element

For most applications objects do not move as if they are in a vacuum. There is almost always a force that resists their movement. When an object is moving slowly, this resistive force is linearly proportional to the velocity of the object in the form of damping.[1] Without damping, an object, once moving, would never come to rest unless an "equalizing" force is applied. Since it is not possible to include such forces, we must rely upon damping to stop the object.

A spring-dash pot system is a simple 1D system with damping. Normally, the node movement is only along the principal axis (Figure 5.1). As such, the damping is also along the principal axis. However, during our deformation process, the nodes are not restricted in their movement, each node is free to move in any direction. This unrestricted movement requires each node to be uniformly damped in all directions. Having uniform damping has

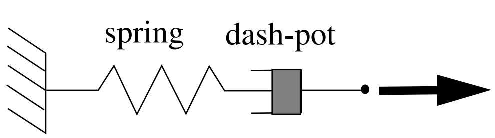


Figure 5.1. Simple 1D spring-dash pot system.

1. Technically, the resistive force has two terms, the first term is proportional to the velocity, while the second term is proportional to the velocity squared. The first term is related to the viscosity of the fluid in which the object is moving. The second is related to the turbulence generated by the object. At low speeds the viscous term dominates the turbulent and as such it is ignored. For simplicity, we assume low speeds.

two advantages over a system that is not uniformly damped. First, the damping becomes time-invariant, which is very important when solving for the surface displacements since it does need to be recalculated at each iteration. Second, the damping matrix is very close to being a circulant matrix and has some unique properties which we can take advantage of, and is discussed in Section 5.5.

5.1.3 The Mass Element

Every object is assumed to have mass, that when accelerated, creates an inertial force. This inertial force, similarly to the damping forces, is not always necessary for the calculation of displacements. At first this seems to be contradictory, how can something exist without any mass? Something that is neutrally buoyant is one such example. It has no apparent mass, it can have a force applied to it, and the movement may be damped. Thus, it is possible to create a system with or without a mass.

A massless system may take longer to move towards its final destination but it is less apt to overshoot it than a system containing mass. The primary system implemented in this dissertation was massless but the system was designed so that masses could be used for comparative studies.

5.2 Assembly of the Global Matrices

For each element in the mesh, equation (5.1) describes the motion of one element as it relates to another element. To solve for the displacement of each element in the mesh, the equations from each element must be combined into a single global equation. This process is known as superposition, equation (5.2). For example, each element stiffness matrix would be expanded to the total order of the total stiffness matrix (N = nodal elements * degrees-of-freedom) and superimposed into one matrix.

$$\boldsymbol{G} = \sum_{e=1}^{N} \boldsymbol{g}^{(e)}\,, \tag{5.2}$$

where $\boldsymbol{g}^{(e)}$ represents a local matrix such as the stiffness matrix between two elements. Note: The summation sign does not mean simple summation but that the local matrices are combined using the principle of superpositioning.

Superposition results in the following equation of motion:

$$\boldsymbol{F}_{ext} = \boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{K}\boldsymbol{q}\,. \tag{5.3}$$

Although equation (5.1) and equation (5.3) look the same, the underlying matrices are different in size (2 * degrees-of-freedom vs. nodal elements * degrees-of-freedom) and equation (5.1) represents the equation of motion for one element while equation (5.3) represents the complete system.

5.3 Solving for the Displacements

In traditional finite element analysis, every node in the mesh contains a mass. This mass creates an inertial force that typically causes the nodes (displacements) to oscillate. How this oscillation takes place is a function of the spring constants and the damping as well as the mass. For instance, if the node is not damped, it will oscillate forever. If the node is underdamped it will oscillate many times before coming to rest. Likewise, over damping the node will never allow it to oscillate but also may take a long time for it to come to rest. Finally, a critically damped system will not oscillate and will come to rest in a reasonable time. It would seem that the natural course of action would be to find the critical damping for the system. This is rather easy for a single-degree-of-freedom system but not so for a multiple-degree-of-freedom system such as a surface containing many nodes. Since each surface being deformed is different, the mass, stiffness and damping coefficients and deformation potentials are normalized to be approximately one. By normalizing it is easier to understand the effects of the coefficients and potentials.

Since perfect damping of a mass system can be hard to achieve, an alternative to a massless system was explored. A massless system does not contain inertial forces and, as such, when the force is removed the node stops moving. The obvious advantage of this system besides removing the need for finding damping coefficients, is that the solution is far easier to determine.

Since we are simulating a dynamic process, the system is time-varying, equation (5.3) represents a set of N coupled second order differential equations, numerical integration techniques are needed to solve for the displacements. Although $\boldsymbol{M}$ and $\boldsymbol{C}$ are time-invariant, $\boldsymbol{K}$ and $\boldsymbol{F}$ are time-varying and must be updated at each time step. Many numerical integration techniques can be used to solve equation (5.3). For the purposes of this research, three distinct methods have been chosen for comparison, Euler, Runge-Kutta, and Central Differences. Three methods, a first-order and a fourth-order Runge-Kutta and Central Differences are used with mass systems, while a simple first order Euler Method is used for a massless system. Details of each system are briefly described below, for complete details see [25].

5.3.1 Massless System via Euler Method

The simplest solution for the displacements is to assume a massless system and solve using a first-order Euler method. By assuming a massless system, equation (5.3) can be rewritten as

$$\boldsymbol{F}_{ext} = \boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{K}\boldsymbol{q}\,. \tag{5.4}$$

The velocity is estimated in terms of the difference in the displacements over one time step:

$$\dot{q}_t \approx \frac{q_{t+\Delta t} - q_t}{\Delta t} \tag{5.5}$$

or in matrix notation

$$\dot{\boldsymbol{q}}_t \approx \frac{1}{\Delta t}(\boldsymbol{q}_{t+\Delta t} - \boldsymbol{q}_t)\,. \tag{5.6}$$

Rewriting equation (5.4) using equation (5.6) the following is obtained

$$\boldsymbol{q}_{t+\Delta t} \approx \boldsymbol{q}_t + \Delta t \cdot \boldsymbol{C}^{-1}(\boldsymbol{F}_{ext} - \boldsymbol{K}\boldsymbol{q})\,. \tag{5.7}$$

The displacements $\boldsymbol{q}_{t+\Delta t}$ are then solved for using equation (5.7) with the results of each iteration being used as the basis for the next iteration.

The implementation is straightforward, but is very unstable due the velocity estimation. There are also other concerns. For instance, inverting the damping matrix $\boldsymbol{C}$ can be computationally very expensive since inversion is $O(n^3)$. Fortunately this matrix is time-invariant and needs to be inverted only once during the deformation process.

5.3.2 Mass System via First-Order Runge-Kutta

The Runge-Kutta method relies upon using state variables for the system of equations. For instance, both the node displacement vector, $\boldsymbol{q}$, and node velocity vector, $\dot{\boldsymbol{q}}$, are considered to be unknowns and can be written as a single state variable $\boldsymbol{x}$:

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{q} \\ \dot{\boldsymbol{q}} \end{bmatrix}. \tag{5.8}$$

Rewriting equation (5.3) in terms of the acceleration:

$$\ddot{\boldsymbol{q}} = -\boldsymbol{M}^{-1}\boldsymbol{K}\boldsymbol{q} - \boldsymbol{M}^{-1}\boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{M}^{-1}\boldsymbol{F}_{ext}. \tag{5.9}$$

Using the identity for $\dot{\boldsymbol{q}}$, equation (5.8) can be rewritten as:

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{\boldsymbol{q}} \\ \ddot{\boldsymbol{q}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{O} & \boldsymbol{I} \\ [-\boldsymbol{M}^{-1}\boldsymbol{K}] & [-\boldsymbol{M}^{-1}C] \end{bmatrix} \begin{bmatrix} \boldsymbol{q} \\ \dot{\boldsymbol{q}} \end{bmatrix} + \begin{bmatrix} \boldsymbol{O} \\ \boldsymbol{M}^{-1}\boldsymbol{F}_{ext} \end{bmatrix}, \tag{5.10}$$

where $\boldsymbol{O}$ is a null matrix, $\boldsymbol{I}$ is the identity matrix.

The Runge-Kutta method assumes that the approximate solution to $\boldsymbol{x}_{t+\Delta t}$ can be found from $\boldsymbol{x}_t$ using a Taylor series expansion. All that needs to be done is to decide on how many terms in the expansion are needed. By using a first order solution (i.e., the first two terms of the Taylor series are kept) the following is arrived at

$$\boldsymbol{x}_{t+\Delta t} = \boldsymbol{x}_t + \Delta t \dot{\boldsymbol{x}}. \tag{5.11}$$

In a similar fashion to equation (5.7), equation (5.11) can be solved to obtain not only the displacements but also the velocities at $t + \Delta t$ with the previous solution being used the basis for the next iteration.

Using a Runge-Kutta method has several advantages. The method is self-starting; does not need any additional values other than the initial values given, which is not the case for Central Differences described in Section 5.3.4. Further, although a matrix inversion is necessary, it is trivial since the mass matrix only has values along the main diagonal. It main disadvantage is its stability.

5.3.3 Mass System via Fourth-Order Runge-Kutta

The first-order solutions are just that, first-order, and their accuracy leaves a lot to be desired in many circumstances. As such, we also developed a fourth-order Runge-Kutta solution. However, we do not have easy access to the high-order derivatives needed so we use the following the approximation from [25], so that the solution can be written in terms of first derivatives:

$$\boldsymbol{k}_1 = \boldsymbol{x}_t,$$

$$\boldsymbol{k}_2 = \boldsymbol{x}_t + \frac{\Delta t}{2}\dot{\boldsymbol{k}}_1,$$

$$\boldsymbol{k}_3 = \boldsymbol{x}_t + \frac{\Delta t}{2}\dot{\boldsymbol{k}}_2,$$

$$\boldsymbol{k}_4 = \boldsymbol{x}_t + \dot{\boldsymbol{k}}_3,$$

$$\boldsymbol{x}_{t+\Delta t} = \boldsymbol{x}_t + \frac{\Delta t}{6}(\dot{\boldsymbol{k}}_1 + 2\dot{\boldsymbol{k}}_2 + 2\dot{\boldsymbol{k}}_3 + \dot{\boldsymbol{k}}_4) \quad , \tag{5.12}$$

where k_n represents the *n-th* derivative and can be thought of as taking four intermediate steps in order to get the final step. The unfortunate part of using a fourth-order solution in terms of the first derivative is that we must evaluate the equation of motion four times to obtain the solution at each iteration. This is computationally very expensive.

5.3.4 Mass System via Central Differences

Another approach is to use a central difference predictor. Similarly to the case for the massless system, the velocity and acceleration can written in terms of the displacements:

$$\dot{q}_{i+1/2} = \frac{q_{i+1} - q_i}{\Delta t}, \tag{5.13}$$

$$\ddot{q}_i = \frac{\dot{q}_{i+1/2} - \dot{q}_{i-1/2}}{\Delta t} \tag{5.14}$$

or in matrix terms:

$$\dot{\boldsymbol{q}}_t = \frac{1}{2\Delta t}(\boldsymbol{q}_{t+\Delta t} - \boldsymbol{q}_{t-\Delta t}), \tag{5.15}$$

$$\ddot{\boldsymbol{q}}_t = \frac{1}{\Delta t^2}(\boldsymbol{q}_{t+\Delta t} - 2\boldsymbol{q}_t + \boldsymbol{q}_{t-\Delta t}). \tag{5.16}$$

Substituting equation (5.15) and equation (5.16) into equation (5.3) the following is obtained:

$$\left(\frac{1}{\Delta t^2}M + \frac{1}{2\Delta t}\boldsymbol{C}\right)q_{t+\Delta t} = \boldsymbol{F}_{ext} - \left(\boldsymbol{K} - \frac{2}{\Delta t^2}\boldsymbol{M}\right)\boldsymbol{q}_t - \left(\frac{1}{\Delta t^2}\boldsymbol{M} - \frac{1}{2\Delta t}\boldsymbol{C}\right)\boldsymbol{q}_{t-\Delta t}. \tag{5.17}$$

In a similar fashion to equation (5.7), equation (5.17) can be solved to obtain the displacements at $t + \Delta t$ with the previous two solutions being used the basis for the next iteration.

Unlike the previous two numerical approaches, equation (5.17) uses the solution at t as well as the solution at $t - \Delta t$. This requires that a starting procedure be used since the initial values are only at $t = 0$. Since the first order Runge-Kutta is self-starting it is used to start this procedure.

One complication with equation (5.17), is that a matrix composed of not only damping terms but also the mass terms must be inverted in order to find a solution. Fortunately though, this matrix is time-invariant and must be inverted only once during the deformation process.

5.3.5 Summary

The implementation of each of these numerical integration techniques is straightforward. However, with each method, a large matrix must inverted in order to solve for the displacements. The inverse of this matrix is typically $O(n^3)$. (In the case of Runge-Kutta the inversion of mass matrix is trivial since it contains terms only along the main diagonal). Although it is possible to use special cases for inverting the matrices and reduce it to under $O(n^3)$, this inversion can become the limiting factor when hundreds of nodes are present.

5.4 Element by Element Solution

Careful examination of the mass, damping, and stiffness matrices shows that each of these matrices are very sparse (typically they are heavily diagonalized). Because these matrices are very sparse many of the multiplications and additions associated with the matrix multiplication are unnecessary since their result is zero. Rather than assembling each of the matrices it is possible to obtain the displacement solutions on an element by element (EBE) basis, [48]. This greatly reduces the computational requirements while also reducing the amount of memory required to store the assembled matrices without any loss in accuracy since the results are exactly the same as if the matrices were assembled. In [61], they briefly discuss using this technique; however no details are given as to how to perform the inversion of damping matrix on an element-by-element method.

5.4.1 Massless System via Euler Method with EBE

In equation (5.7), the stiffness matrix $\boldsymbol{K}$ is assembled and a matrix multiplication with the current displacements $\boldsymbol{q}$ is performed. This is a O(n^2) operation and is very wasteful operation considering the stiffness matrix is sparse. Instead, the individual element stiffness matrices can be multiplied by the displacements that are associated with that element and then assembled into a vector. This can essentially be thought of as calculating an internal force vector, with each component in the vector being from a different spring connection equation (5.18) and equation (5.19).

$$\boldsymbol{F}_{int} = -\sum_{e=1}^{N} \boldsymbol{kq}^{(e)} \tag{5.18}$$

$$\boldsymbol{q}_{t+\Delta t} = \boldsymbol{q}_t + \Delta t \cdot \boldsymbol{C}^{-1}(\boldsymbol{F}_{ext} + \boldsymbol{F}_{int}) \tag{5.19}$$

5.4.2 Mass System via Runge-Kutta with EBE

A similar approach was taken for applying the EBE method to equation (5.10) which is used for both the first fourth order solutions except that not only was the stiffness matrix done on an EBE basis but also the multiplication of the damping matrix $\boldsymbol{C}$ and the current velocity vector $\dot{\boldsymbol{q}}$. Thus the internal force is composed of not only the spring force but is also damped equation (5.20) and equation (5.21).

$$\boldsymbol{F}_{int} = \sum_{e=1}^{N} (-\boldsymbol{k}\boldsymbol{q}_t^{(e)} - \boldsymbol{c}\dot{\boldsymbol{q}}_t^{(e)}) \tag{5.20}$$

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{\boldsymbol{q}} \\ \ddot{\boldsymbol{q}} \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{q}} \\ \boldsymbol{M}^{-1}\boldsymbol{F}_{int} \end{bmatrix} + \begin{bmatrix} \boldsymbol{O} \\ \boldsymbol{M}^{-1}\boldsymbol{F}_{ext} \end{bmatrix} \tag{5.21}$$

5.4.3 Mass System via Central Differences with EBE

When using central differences the “internal” force becomes more complicated equation (5.22) and equation (5.23) and can not be explained in simple physical term.

$$F_{int} = \sum_{e=1}^{N} \left(-\left(\boldsymbol{k} - \frac{2}{\Delta t}\boldsymbol{m} \right) \boldsymbol{q}_t^{(e)} - \left(\frac{\boldsymbol{m}}{\Delta t^2} - \frac{\boldsymbol{c}}{2\Delta t} \right) \boldsymbol{q}^{(e)}{}_{t-\Delta t} \right) \tag{5.22}$$

$$\left(\frac{\boldsymbol{M}}{\Delta t^2} + \frac{\boldsymbol{C}}{2\Delta t} \right) \boldsymbol{q}_{t+\Delta t} = \boldsymbol{F}_{ext} + \boldsymbol{F}_{int} \tag{5.23}$$

5.4.4 Summary

In performing the EBE method it was obvious that some savings could be gained since a large matrix multiplication could be replaced by many small matrix multiplications. But what is the savings and is it always practical? Examining the savings from just the stiffness matrix multiplication for the massless system equation (5.18), each element stiffness matrix is 6x6 (2 elements, 3 degrees-of-freedom) and each element has 4 neighbors the multiplication becomes $O(n * 4/2 * 6^2)$ (since each neighbor was counted twice the result must be divided by 2). Since the multiplication of the assembled matrices was an $O(n^2)$ operation when there are more than 24 elements each having 3 degrees-of-freedom, it was computationally less expensive to use the EBE method.

5.5 Approximating the Damping Matrix

Using the EBE method, a large savings can be gained during each iteration. However, this savings is greatly overshadowed by the $O(n^3)$ matrix inversion operation. Thus, in order to save a significant amount of computational expense, the complexity of the matrix inversion must reduced. In this section, we explore the possibility of approximating the matrix so that less complex techniques may be used.

The damping matrix used in equation (5.7) for an n x n cylindrical surface containing nodes each having 1 degree-of-freedom would be:

$$\begin{array}{c}
\begin{array}{ccccccccccccccccccccc}
N_1 & N_2 & N_3 & & N_{n-2} & N_{n-1} & N_n & N_{n+1} & N_{n+2} & N_{n+3} & & N_{n*(n-1)-2} & N_{n*(n-1)-1} & N_{n*(n-1)} & N_{n*(n-1)+1} & N_{n*(n-1)+2} & N_{n*(n-1)+3} & & N_{n*n-2} & N_{n*n-1} & N_{n*n}
\end{array} \\
\begin{bmatrix}
\mathbf{3} & -1 & & . & & & \mathbf{-1} & -1 & & & . & & & & \mathbf{0} & 0 & 0 & . & & & \mathbf{0} \\
-1 & \mathbf{3} & -1 & . & & & & & -1 & & . & & & & & \mathbf{0} & & . & & & \\
& -1 & \mathbf{3} & . & & & & & & -1 & . & & & & & & \mathbf{0} & . & & & \\
. & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . \\
& & & . & \mathbf{3} & -1 & & & & & . & -1 & & & & & & . & \mathbf{0} & & \\
& & & . & -1 & \mathbf{3} & -1 & & & & . & & -1 & & & & & . & & \mathbf{0} & \\
\mathbf{-1} & & & . & & -1 & \mathbf{3} & \mathbf{0} & & & . & & & -1 & & & & . & & & \mathbf{0} \\
-1 & & & . & & & \mathbf{0} & 4 & -1 & & . & & & \mathbf{-1} & -1 & & & . & & & \\
& -1 & & . & & & & -1 & 4 & -1 & . & & & & & -1 & & . & & & \\
& & -1 & . & & & & & -1 & 4 & . & & & & & & -1 & . & & & \\
. & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . \\
& & & . & -1 & & & & & & . & 4 & -1 & & & & & . & -1 & & \\
& & & . & & -1 & & & & & . & -1 & 4 & -1 & & & & . & & -1 & \\
& & & . & & & -1 & \mathbf{-1} & & & . & & -1 & 4 & \mathbf{0} & & & . & & & -1 \\
\mathbf{0} & & & . & & & & -1 & & & . & & & \mathbf{0} & \mathbf{3} & -1 & & . & & & \mathbf{-1} \\
& \mathbf{0} & & . & & & & & -1 & & . & & & & -1 & \mathbf{3} & -1 & . & & & \\
& & \mathbf{0} & . & & & & & & -1 & . & & & & & -1 & \mathbf{3} & . & & & \\
. & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . \\
& & & . & \mathbf{0} & & & & & & . & -1 & & & & & -1 & . & \mathbf{3} & -1 & \\
& & & . & & \mathbf{0} & & & & & . & & -1 & & & & & . & -1 & \mathbf{3} & -1 \\
\mathbf{0} & & & . & & & \mathbf{0} & & & & . & & & -1 & \mathbf{-1} & & & . & & -1 & \mathbf{3}
\end{bmatrix}
\end{array} . \tag{5.24}$$

A similar matrix is formed when the nodes have 2 or 3 degrees-of-freedom. Close examination of this matrix shows that it has a band-diagonal structure and is close to being a circulant matrix. If the terms in bold, **0**, **-1**, and **3** instead were -1, 0, and 4, respectively, then the damping matrix would be a circulant matrix. (Interestingly enough, for a 2D periodic curve the damping matrix is circulant.)

5.5.1 Circulant Matrices

Circulant matrices have some unique properties that can be taken advantage of when performing matrix operations, which we discuss below. More rigorous and concise details may be found in [83].

A circulant matrix is defined as a matrix with constant main, upper and lower diagonal terms. In addition, the off diagonal terms are periodic, that is, the upper diagonal terms wrap around to become terms in the lower diagonal as shown in equation (5.25).

$$\boldsymbol{C} = \begin{bmatrix} c_0 & c_1 & \dots & \dots & c_{n-1} \\ c_{n-1} & c_0 & c_1 & \dots & \dots \\ \dots & c_{n-1} & c_0 & c_1 & \dots \\ \dots & \dots & \dots & \dots & \dots \\ c_1 & \dots & \dots & c_{n-1} & c_0 \end{bmatrix} \tag{5.25}$$

A circulant matrix can be represented as a vector in the following form:

$$\boldsymbol{c} = \{c_0, c_1, c_2, \dots, c_{n-1}\} , \tag{5.26}$$

where each component in the vector represents one of the diagonal terms in the circulant matrix. It has been shown in [83], that multiplying a circulant matrix $\boldsymbol{C}$ and a vector $\boldsymbol{q}$ is the same as convolving the vector $\boldsymbol{c}$ and $\boldsymbol{q}$, equation (5.27).

$$\boldsymbol{Cq} = \boldsymbol{c} * \boldsymbol{q} . \tag{5.27}$$

Since convolution can also be done using Fourier transforms, we can multiply the Fourier transform of $\boldsymbol{c}$ by the Fourier transform of $\boldsymbol{q}$, and take the inverse Fourier transform to obtain the same result. Thus, the matrix multiplication can be reduced to a series of Fourier transforms, equation (5.28)

$$\boldsymbol{Cq} = \boldsymbol{c} * \boldsymbol{q} = F^{-1}[F(\boldsymbol{c})F(\boldsymbol{q})] . \tag{5.28}$$

However, in equation (5.7) it is not the damping matrix that is being multiplied but its inverse. Fortunately, since the inverse of a circulant matrix is also circulant and we can write the following:

$$\boldsymbol{C}^{-1}\boldsymbol{q} = \boldsymbol{c}^{-1} * \boldsymbol{q}\,, \tag{5.29}$$

where $\boldsymbol{c}^{-1}$ is a vector composed of the diagonal terms from the matrix $\boldsymbol{C}^{-1}$. However, using Fourier transforms it is possible to divide on an element-by-element basis the Fourier transform of the vector $\boldsymbol{q}$, by the Fourier transform of the vector $\boldsymbol{c}$, equation (5.30) (Note: this division does not take place if $F(\boldsymbol{c})_i$ is singular, instead the result is set to zero)

$$\boldsymbol{C}^{-1}\boldsymbol{q} = \boldsymbol{c}^{-1} * \boldsymbol{q} = F^{-1}\left(\frac{F(\boldsymbol{q})_0}{F(\boldsymbol{c})_0}, \frac{F(\boldsymbol{q})_1}{F(\boldsymbol{c})_1}, \frac{F(\boldsymbol{q})_2}{F(\boldsymbol{c})_2}, \ldots \frac{F(\boldsymbol{q})_n}{F(\boldsymbol{c})_n}\right)\,, \tag{5.30}$$

where $F(\boldsymbol{x})_i$ is the *i-th* component of the forward Fourier transform of the vector $\boldsymbol{x}$. Thus, the inverse of the damping matrix $\boldsymbol{C}$, is not needed. Instead, a series of Fourier transforms can be used.

It should be noted that since the matrix is also symmetric, the discrete Fourier transform of the vector $\boldsymbol{c}$, results in only real components, i.e., the imaginary component is always zero. This makes the division much simpler. Although typically computational more efficient, the fast Fourier transform, (FFT), can not be used because zero padding can destroy the matrix symmetry, which would result in a nonzero imaginary component. Thus, we are restricted to using a Discrete Fourier Transform.

Using Fourier transforms makes the matrix multiplication easier but at the added expense of needing two Fourier transforms at each iteration, the matrix $\boldsymbol{C}$ is time-invariant so the Fourier transform of the vector $\boldsymbol{c}$ needs to be calculated only once. The discrete Fourier transform (DFT) is $O(N^2)$, so for N degrees-of-freedom, $2\ N^2$ operations are needed. But, the matrix multiplication is also $O(N^2)$, so not much is gained. However, the computational savings lies, not at each iteration, but in not having to take the initial inverse, which was O(N^3). Instead, it has been replaced by a Fourier transform of O(N^2).

Unfortunately, using Fourier transforms is not always the fastest method if more than N iterations are needed for N degrees-of-freedom. This can be shown by comparing the number of operations required for performing the matrix inversion and the matrix multiplication for M iterations, $N^3 + MN^2$, with the number of operations to perform an initial Fourier transform and two additional transforms for M iterations, $N^2 + M2N^2$. When M = N, it takes N^2 more operations to use the Fourier transforms.

5.5.2 Approximation of the Damping Matrix with a Circulant Matrix

As previously shown in equation (5.24), the damping matrix for a cylindrical surface is almost, but not quite, a circulant matrix. Upon first examination, it would seem reasonable to just approximate the damping matrix with a circulant matrix. However, this results in unacceptable behavior at the nodes where corresponding approximations have been made on the boundary of the surface. All of the interior nodes behave correctly since no approximations are made to their behavior. This difference can be directly related to the fact that the interior nodes have four neighbors while the border nodes have three neighbors.

In experiments we performed, it was found that if only the interior nodes are used then the above approximation is indeed a reasonable approximation. However, since the boundary nodes are needed in the shape recovery process, it is not feasible to ignore them. Instead, nodes are added to the boundary that are not needed in the shape recovery process, but are used in the matrix operations. These nodes are referred to as phantom nodes and are symbolically shown in Figure 5.2. Two types of phantom nodes are used. The first

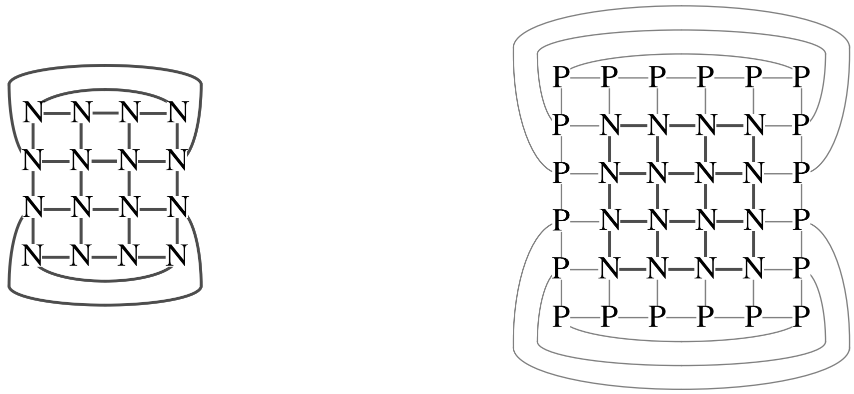


Figure 5.2. Original mesh with normal nodes, N (left) and mesh with phantom nodes, P (right).

type of node relies upon the periodic nature of the mesh as would be found with a cylindrical mesh and duplicate other nodes. The second type of node, are on the boundary of the mesh and are used for padding.

The use of phantoms nodes is well established in other disciplines [26]. The addition of phantom nodes can be thought of as the unwrapping of the cylinder in much the same way periodic B-splines can be changed into floating B-splines by adding knots and duplicating points in the control mesh. The extra knots and control points can effectively thought of as being phantom nodes. Another example, is in the field of data fitting, where one might fit a cubic curve to four points. Very often only the middle section interpolated by the curve is kept since the outer sections tend to be unstable. In this case, we can think of the outer points as being phantom nodes.

The use of phantom nodes results in the damping matrix being changed from:

$$
\begin{array}{c}
\begin{array}{ccccccccccccccccccccc}
N_1 & N_2 & N_3 & & N_{n-2} & N_{n-1} & N_n & N_{n+1} & N_{n+2} & N_{n+3} & & N_{n*(n-1)-2} & N_{n*(n-1)-1} & N_{n*(n-1)} & N_{n*(n-1)+1} & N_{n*(n-1)+2} & N_{n*(n-1)+3} & & N_{n*n-2} & N_{n*n-1} & N_{n*n}
\end{array} \\
\begin{bmatrix}
\mathbf{3} & -1 & & \cdot & & & \mathbf{-1} & -1 & & & \cdot & & & & \mathbf{0} & 0 & 0 & \cdot & & & \mathbf{0} \\
-1 & \mathbf{3} & -1 & \cdot & & & & & -1 & & \cdot & & & & & \mathbf{0} & & \cdot & & & \\
& -1 & \mathbf{3} & \cdot & & & & & & -1 & \cdot & & & & & & \mathbf{0} & \cdot & & & \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
& & & \cdot & \mathbf{3} & -1 & & & & & \cdot & -1 & & & & & & \cdot & \mathbf{0} & & \\
& & & \cdot & -1 & \mathbf{3} & -1 & & & & \cdot & & -1 & & & & & \cdot & & \mathbf{0} & \\
\mathbf{-1} & & & \cdot & & -1 & \mathbf{3} & \mathbf{0} & & & \cdot & & & -1 & & & & \cdot & & & \mathbf{0} \\
-1 & & & \cdot & & & \mathbf{0} & 4 & -1 & & \cdot & & & \mathbf{-1} & -1 & & & \cdot & & & \\
& -1 & & \cdot & & & & -1 & 4 & -1 & \cdot & & & & & -1 & & \cdot & & & \\
& & -1 & \cdot & & & & & -1 & 4 & \cdot & & & & & & -1 & \cdot & & & \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
& & & \cdot & -1 & & & & & & \cdot & 4 & -1 & & & & & \cdot & -1 & & \\
& & & \cdot & & -1 & & & & & \cdot & -1 & 4 & -1 & & & & \cdot & & -1 & \\
& & & \cdot & & & -1 & \mathbf{-1} & & & \cdot & & -1 & 4 & \mathbf{0} & & & \cdot & & & -1 \\
\mathbf{0} & & & \cdot & & & & -1 & & & \cdot & & & \mathbf{0} & \mathbf{3} & -1 & & \cdot & & & \mathbf{-1} \\
& \mathbf{0} & & \cdot & & & & & -1 & & \cdot & & & & -1 & \mathbf{3} & -1 & \cdot & & & \\
& & \mathbf{0} & \cdot & & & & & & -1 & \cdot & & & & & -1 & \mathbf{3} & \cdot & & & \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
& & & \cdot & \mathbf{0} & & & & & & \cdot & -1 & & & & & -1 & \cdot & \mathbf{3} & -1 & \\
& & & \cdot & & \mathbf{0} & & & & & \cdot & & -1 & & & & & \cdot & -1 & \mathbf{3} & -1 \\
\mathbf{0} & & & \cdot & & & \mathbf{0} & & & & \cdot & & & -1 & \mathbf{-1} & & & \cdot & & -1 & \mathbf{3}
\end{bmatrix}
\end{array}
\tag{5.31}
$$

to:

$$
\begin{array}{c}
\begin{array}{ccccccccccccccccccccc}
P_1 & P_2 & P_3 & & P_{p-2} & P_{p-1} & P_p & P_{p+1} & N_1 & N_2 & & N_{n*n-1} & N_{n*n} & P_{p*(p-1)} & P_{p*(p-1)+1} & P_{p*(p-1)+2} & P_{p*(p-1)+3} & & P_{p*p-2} & P_{p*p-1} & P_{p*p}
\end{array} \\
\begin{bmatrix}
\mathbf{3} & -1 & & \cdot & & & \mathbf{-1} & -1 & & & \cdot & & & & \mathbf{0} & 0 & 0 & \cdot & & & \mathbf{0} \\
-1 & \mathbf{3} & -1 & \cdot & & & & & -1 & & \cdot & & & & & \mathbf{0} & & \cdot & & & \\
& -1 & \mathbf{3} & \cdot & & & & & & -1 & \cdot & & & & & & \mathbf{0} & \cdot & & & \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
& & & \cdot & \mathbf{3} & -1 & & & & & \cdot & -1 & & & & & & \cdot & \mathbf{0} & & \\
& & & \cdot & -1 & \mathbf{3} & -1 & & & & \cdot & & -1 & & & & & \cdot & & \mathbf{0} & \\
\mathbf{-1} & & & \cdot & & -1 & \mathbf{3} & \mathbf{0} & & & \cdot & & & -1 & & & & \cdot & & & \mathbf{0} \\
-1 & & & \cdot & & & \mathbf{0} & 4 & -1 & & \cdot & & & \mathbf{-1} & -1 & & & \cdot & & & \\
& -1 & & \cdot & & & & -1 & 4 & -1 & \cdot & & & & & -1 & & \cdot & & & \\
& & -1 & \cdot & & & & & -1 & 4 & \cdot & & & & & & -1 & \cdot & & & \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
& & & \cdot & -1 & & & & & & \cdot & 4 & -1 & & & & & \cdot & -1 & & \\
& & & \cdot & & -1 & & & & & \cdot & -1 & 4 & -1 & & & & \cdot & & -1 & \\
& & & \cdot & & & -1 & \mathbf{-1} & & & \cdot & & -1 & 4 & \mathbf{0} & & & \cdot & & & -1 \\
\mathbf{0} & & & \cdot & & & & -1 & & & \cdot & & & \mathbf{0} & \mathbf{3} & -1 & & \cdot & & & \mathbf{-1} \\
& \mathbf{0} & & \cdot & & & & & -1 & & \cdot & & & & -1 & \mathbf{3} & -1 & \cdot & & & \\
& & \mathbf{0} & \cdot & & & & & & -1 & \cdot & & & & & -1 & \mathbf{3} & \cdot & & & \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
& & & \cdot & \mathbf{0} & & & & & & \cdot & -1 & & & & & -1 & \cdot & \mathbf{3} & -1 & \\
& & & \cdot & & \mathbf{0} & & & & & \cdot & & -1 & & & & & \cdot & -1 & \mathbf{3} & -1 \\
\mathbf{0} & & & \cdot & & & \mathbf{0} & & & & \cdot & & & -1 & \mathbf{-1} & & & \cdot & & -1 & \mathbf{3}
\end{bmatrix}
\end{array} .
\tag{5.32}
$$

Although difficult to see, the matrices are different. The latter matrix has $(N+2)^2$ elements while the former had N^2. Also, two types of nodes are now present, the original N^2 nodes plus an additional 4(N+1) phantom nodes. With the addition of phantom nodes all of the original nodes are now interior nodes and have the exact same circulant form. This leaves only the damping of the phantom nodes to be approximated (those values shown in bold are the values that will be changed).

In addition, the net force vector $\{F_{net}\} = \{F_{ext}\} + \{F_{int}\}$ must be modified to reflect the additional phantom nodes in the damping matrix. This modified vector contains $(N+2)^2$ values each representing a net force. If the mesh is column periodic, the phantom node forces on the left hand side of the mesh assume the same value as the nodes on the right hand side of the original mesh. The opposite is done for the phantom nodes on the right hand side of the mesh and similarly for the nodes on the top and bottom if the mesh is row periodic (Figure 5.3). This is similar to the duplication of control points when changing a period B-spline into a floating B-spline. If the mesh is not row or column periodic, the phantom node forces that remain along the boundary are assigned a value of zero. This is similar to zero-padding a common technique used when working with Fourier

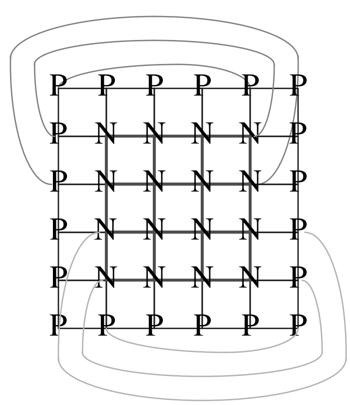


Figure 5.3. Period relationship of the phantom nodes to the original nodes.

transforms. The corners of the phantom mesh are always assigned a value of zero. The completed vector is given in equation (5.33).

$$
\begin{array}{ccccccccccccccccccccc}
P_1 & P_2 & P_3 & & P_{p-2} & P_{p-1} & P_p & P_{p+1} & N_1 & N_2 & & N_{n*n-1} & N_{n*n} & P_{p*(p-1)} & P_{p*(p-1)+1} & P_{p*(p-1)+2} & P_{p*(p-1)+3} & & P_{p*p-2} & P_{p*p-1} & P_{p*(p-1)} \\
\end{array}
$$
$$
\begin{bmatrix} 0 & 0 & 0 & \ldots & 0 & 0 & 0 & F_{n*n} & F_1 & F_2 & \ldots & F_{n*n-1} & F_{n*n} & F_1 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 \end{bmatrix} \tag{5.33}
$$

The displacements can be found using the circulant part of equation (5.32) and equation (5.33) in combination with a series of Fourier transforms.

5.5.3 Summary

The use of a circulant matrix reduces the computational operations by a factor of N. However, the use of a circulant matrix does require the approximation of the damping mesh. A direct substitution leads to unacceptable errors. However, with the addition of phantom nodes along the border of the damping matrix and the resulting force vector a substitution is possible since all of the interior nodes remain stable even though the phantom nodes are not. This stable technique is further demonstrated in Chapter 7.

5.6 Summary

The approximation of the pseudo-physical deformation process of a surface is a critical step. This research uses a standard finite element based system. It does, however, use several approximations to the reduce the computational expense. The approximation comes in with the introduction of using a circulant matrix which is presented for the first time with the finite element method. Through the use of the EBE method the computational expense is reduced to O(N) while the circulant matrix inversion is reduced to O(N^2). The combination of these two factors makes the deformation process computationally realistic for many applications.

CHAPTER 6

DEFORMATION OPERATORS

In Chapter 5 all of the forces that act upon the deformable surfaces were discussed in detail except for the external forces. A surface can deform in response to a variety of external forces. For instance, a surface could deform when the user introduces an external force, or from a force based on the image potentials. Since the focus of this research is on surfaces that are based on physical properties, the deformation operators should also be based on physical properties. One such deformation operator is a spring force, $\mathbf{F}=k\mathbf{x}$, where $\mathbf{F}$ is the force, k is the spring constant and $\mathbf{x}$ is the differential length vector from the spring's resting position. The main difficulty in using this type of operator is deciding how and where to attach the spring to the surface.

The parameters for different deformation operators can be divided into two categories, those specified by the user and those from external data. Those forces specified by external data may originate from many sources, such as range data, monocular images, surface data, and image data. In this particular case, the ability to create forces from high contrast 2D and 3D medical images was of interest.

6.1 Boundary Detection

The first step in determining the image potentials is to determine the strength of the surface's boundary. The surface boundary is defined as those points whose gradient is maximal at the surface normal. Several approaches can be used for determining the surface boundary. Most use gradient information and are in the form of edge operators [2]. An edge operator will not determine the boundary exactly but will give statistical measures indicating the probability of a particular point being on the boundary. After thresholding to remove the weaker indicators, a boundary will remain that is possibly several voxels thick. This technique, however, does not guarantee that a continuous boundary will be found. Previous researchers have used many different types of edge operators, each of which has advantages and disadvantages. In general, these operators can be placed into one of three general classes, mathematical gradient operators, such as the Sobel or Laplacian operators, template matching operators as the Kirsh or Nevatia-Babu templates, and parametric edge models were one attempts to fit a function to the data. The merits of each of these operators will not be discussed although detail descriptions of each can be found in [2] and [79].

Experiments on vascular image data using both gradient and template operators showed that an ideal step edge operator such as the Nevatia-Babu [68] worked very well on 2D vascular data (Figure 6.1). The edges are well defined when compared to the background with very little response in other areas of the image. Although the optimum response of this operator occurs with linear features it also works remarkably well with curved features. This is because when curved features are examined on a pixel level, they are really composed of a series of linear features.

The 2D Nevatia-Babu operator is a set of twelve 5 x 5 masks corresponding to an

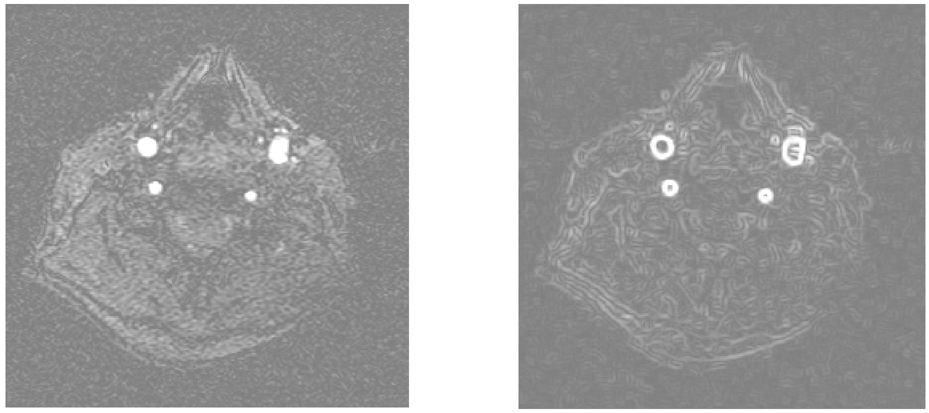


Figure 6.1. A “typical” image slice (left) and response to a 2D Nevatia-Babu Filter (right)

ideal edge and can be implemented to match edges at almost any angle (Figure 6.2).[1] Each of 12 masks are convolved with the image slice, with the mask giving the largest response at each pixel being recorded in the edge data. However, it is preferable to use a 3D filter rather than a 2D filter since the image data are 3D. We extend the 2D Nevatia-Babu filter into a 5 x 5 x 5 3D filter by duplicating 2D mask in the Z=-2, -1, +1, +2 planes (Figure 6.2). This mask was then rotated about the Z and X axis to form a total of 64 masks, each representing the ideal step surface. Each mask has a maximal response to an ideal step surface (i.e., the response is maximal for flat surfaces). The response from the 3D filter was greater than the 2D filter and produced fewer edges than the 2D filter (Figure 6.1).

6.2 Image Potentials

Once the strength of the boundaries have been determined there are two issues that must be investigated. One is converting the boundary strength into a force. The other is de-

1. The number of angles the filter is able to distinguish is limited due to round off error, in this particular case twelve masks representing lines at 0^o, 30^o, 60^o, 90^o,..., and 330^o were used.

-100	-100	0	100	100
-100	-100	0	100	100
-100	-100	0	100	100
-100	-100	0	100	100
-100	-100	0	100	100

(a) 0^o

-100	32	100	100	100
-100	-78	92	100	100
-100	-100	0	100	100
-100	-100	-92	78	100
-100	-100	-100	-32	100

(b) 30^o

.100	100	100	100	100
-100	-78	92	100	100
-100	-100	0	100	100
-100	-100	-92	78	100
-100	100	-100	-32	100

(c) 60^o

100	100	100	100	100
100	100	100	100	100
0	0	0	0	0
-100	-100	-100	-100	-100
-100	-100	-100	-100	-100

(d) 90^o

100	100	100	100	100
100	100	100	78	-32
100	92	0	-92	-100
32	-78	-100	-100	-100
-100	-100	-100	-100	-100

(e) 120^o

100	100	100	32	-100
100	100	92	-78	-100
100	100	100	-100	-100
100	78	-92	-100	-100
100	32	-100	-100	-100

(f) 150^o

Figure 6.2. 2D Nevatia-Babu edge masks in six directions.

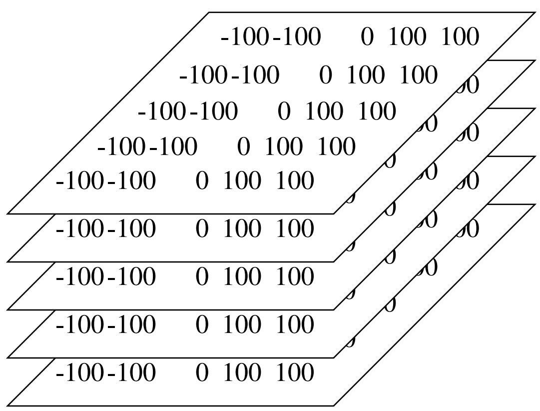


Figure 6.3. 3D Nevatia-Babu edge mask for a 0^o-0^o surface.

termining how the force will effect the surface. For instance in the first case, the force can be proportional to strength of the boundary. As such, all that remains is to find the correct proportionality coefficient. This technique has been widely used in several other surface recovery techniques using deformable surfaces [23, 75, 87]. Determining how the image

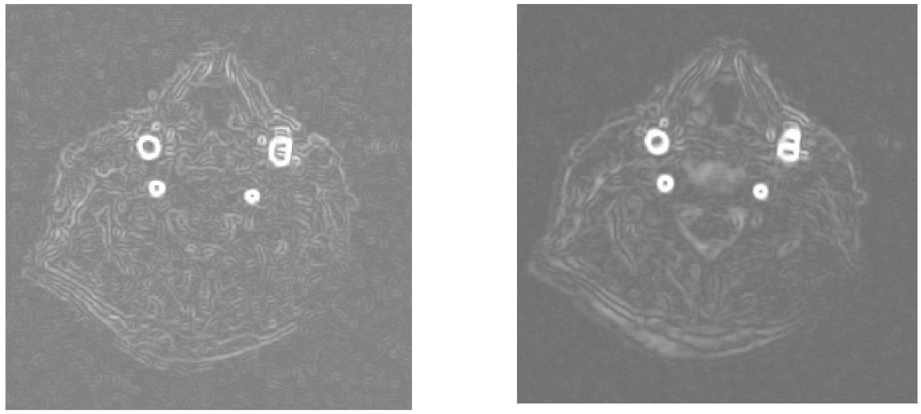


Figure 6.4. The response from a 2D Nevatia-Babu Filter (left) and a 3D Nevatia-Babu Filter (right) for the same slice.

potentials affect the surface is a much more difficult problem to solve. These issues are independent.

As previously discussed, image potentials are discrete measurements of image data which are used to deform the surfaces. The discrete measurements are the result of using boundary detection operators on the image which returned the likelihood that a particular point lies on a boundary. The stronger the likelihood, the stronger the measurement and the greater chance for the surface to deform towards or around that particular point.

One of the first types of potential explored was one that could impede the motion of a deforming surface. The surface was deformed similarly to a balloon in a jug being expanded in the presence of an internal pressure. As the balloon expands and comes into contact with the walls its motion is stopped. In the case of a surface and image potentials, the surface would expand until it came into contact with an image potential that could stop it. Since a single potential would rarely stop the deformation, multiple image potentials were combined until their sum formed a large enough potential to stop the motion. This is similar to the idea of shoveling snow along a sidewalk, at first it is easy to move the snow but with time more snow accumulates and it becomes harder to move the snow.

Although conceptually easy to understand, implementation became impractical. For instance, each potential had to be treated as an individual object so that a surface/point intersection could be detected. Further, at each intersection, a resultant force had to be calculated and in the case of multiple intersections, multiple reactant forces. This became an exercise in calculating multibody collisions and their reactions. Although certainly reasonable, a simple image contains thousands of image potentials, each of which would have been an independent body. Because of the large number of image potentials, this idea was soon abandoned.

As previously discussed, the current deformable surfaces use the image potentials as an attractive force to deform the surfaces. We explored similar techniques for our image potentials. However, a different method for "attaching" the potentials to the surface was developed as well as different force relationships since previously reported "attachments" had problems with objects that were symmetric.

The first question that must be asked is what the correspondence between the surface and the image data should be. The correspondence can be one-to-many, i.e., each image potential would act as a force field in a way similar to that of a gravitational field. The force would be proportional to the boundary likelihood and inversely proportional to the distance between the surface and the point. The beauty of this approach is that it allows the attractive forces to act in all directions. Further, the influence of each image point is not limited to only one location on the surface, but would influence a more global area. Although accurate, this would require relating each image potential to many locations on the surface, a process which is computationally expensive. This approach is described in [87]. The disadvantage of such a system is that if the image potentials are equally spaced around the surface (i.e., symmetric), the surface may not deform completely and typically requires user intervention [23].

As previously discussed in [75], the force is projected on to the surface and then distributed among the closest nodes since the projection of the force rarely lies on a node. However, this method suffers from being restricted to simple objects (no concavities) and the original surface must be oriented to within 15 degrees of its final position [75]. In our evaluations of this technique we found that in certain circumstances distributing the forces can cause the surface to deform past the boundaries. This unexpected result was due to the imbalance among the equivalent forces at the nodes (Figure 6.5). In this example, the center node gets an unneeded vertical component which will move it beyond the boundary during the deformation process.

To increase the computational efficiency and to address the above problems with the image potentials we require each potential to affect only one point on the surface, but allow one point on the surface to be affected by multiple image potentials. The location on the surface which is affected by the image potential was determined to be the surface node located closest to the image potential. This simple technique has no physical analogue (al-

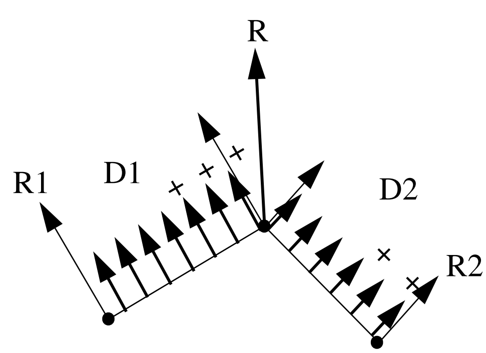


Figure 6.5. Calculation of the resultant force R for the center node using a distributed force technique. Where R = R1 + R2 and R1 = 3.5 * D1 and R2 = 3 * D2 and D1 > D2. The center assumes a force that moves it beyond the boundary, marked with +'s.

though one could think of each potential as being an creature with one arm that reaches out and grabs the node closest to it). As will be demonstrated in the results, this technique solves both the symmetry problem, and the distributed force problem.

The last step is determining how to calculate the image potential force. We developed two methods. The force is proportional to the boundary strength but inversely proportional to the distance squared r^2, i.e., gravity, equation (6.1). The second method makes the force proportional to the boundary strength and the distance r, i.e., a spring, equation (6.2).

$$F = \|O_{NB}(I)\| e^{-\left(\frac{r^2}{const}\right)} \tag{6.1}$$

$$F = \|O_{NB}(I)\| \frac{r}{const} \tag{6.2}$$

where $O_{NB}(I)$ is the Nevatia-Babu operator applied to the image, *I*. Similar forces have been described before for spring type potential in equation (6.2) but not for the gravity potential where an r^{-2} term is usually used. We chose to use a function such as e^{-r^2} because it would not magnify the potential strength when $r < 1.0$. In addition, the function falls off very rapidly as the distance, r, increases. For simplicity, these potentials are call directional potentials (as opposed to radial potentials). Thus direct-gravity-potentials and direct-spring-potentials are being introduced and explored.

6.3 Summary

Various techniques currently exist for establishing a relationship between the image potentials and the surface. Many of these techniques fail when the object is symmetric or has a concave shape. However, by deviating from a system where each image potential affects every node on the surface, and instead allowing each potential to affect a single node, these problems have been overcome while decreasing the computational expense and increasing the accuracy.

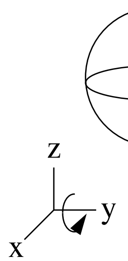
An interesting issue not explored is the use of image potentials derived from multiple data sources. For instance, in collecting image data from multiple views, a point could be seen in multiple views thus allowing for multiple samples for that point. Rather than excluding the duplicate data, it could be used to enhance the deformation process using a weighted average of each data point. This process has been successfully used in many image reconstruction techniques where the projection has been multiply defined [49].

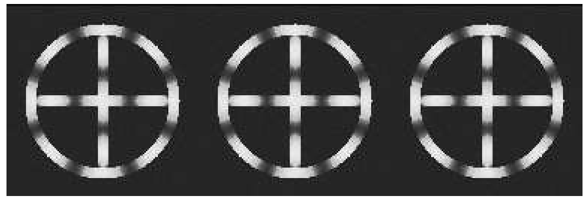
CHAPTER 7

RESULTS

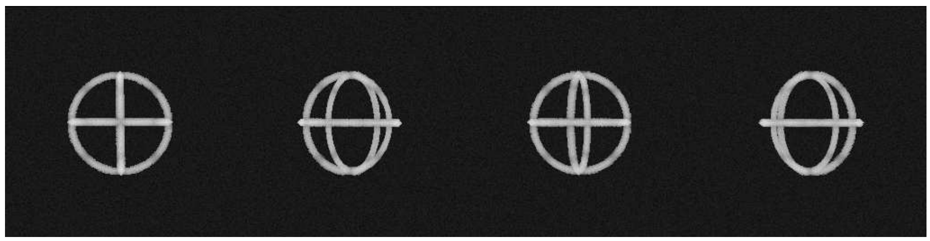
In this chapter the results of the shape recovery process are presented. The process consists of three main areas, the topology identification, modeling seeding, and the deformation process.

All of the techniques were applied to both real and synthetic data. The synthetic data provided a test bed for evaluating each technique. The synthetic data consisted of a 3D image (128^3) of three orthogonal interlocking rings as would be generated by magnetic resonance and four 2D images (256^2) which would be created by digital x-ray techniques (Figure 7.1). The four digital x-ray views were generated at 0, 30, 80, and 145 degrees about the y axis using a summed divergent projection. ($\Theta = 0$ is normal to the XY plane). For both images, a small amount of Gaussian noise was added to each image (signal-to-noise ratio of 20:1). To simulate image degradation, signal losses were introduced every 45 degrees along each ring. These losses were offset by 22.5 degrees for each modality so that there would be over lap in signal in each image (i.e., for MR the signal losses were at 0, 45, 90, etc., whereas for the X-ray the signal losses were at 22.5, 67.5, 112.5, etc.) These images are similar in projection angles and in signal losses to real MR and X-ray angiograms but were not meant to be a true representation of actual images since we only wanted them for test theoretical aspects of the topology identification.





128 by 128 orthogonal Maximum Intensity Projections (MIP) MR images in the XY, XZ, and YZ planes respectively.



256 by 256 summed divergent X-ray images at 0, 30, 80, 145 degree respectively.

Figure 7.1. Wire-frame of the three interlocking rings and simulated MR and X-ray images.

Other synthetic images were created for testing the deformation process. These images were typically 64^3 and consisted of various shapes such as a potato shape with various signal-to-noise levels.

The patient data sets presented are from 3D Magnetic Resonance Angiograms (MRA) and from 2D Digitally Subtracted Angiograms (DSA). They were obtained from a General Electric 1.5T MRI scanner and a General Electric Advantex Bi-Plane Digital X-ray System, respectively [41]. Different vascular imaging protocols were used to obtain as wide of variety of data as possible. These protocols included the MOTSA [71] and 3D Time of Flight [29] for the 3D MR angiograms whereas the X-ray angiograms were obtained using a standard angiographic bi-plane technique including the injection of a angiographic contrast medium [70]. The data were obtained from healthy volunteers as well as from patient volunteers with suspected vascular abnormalities.

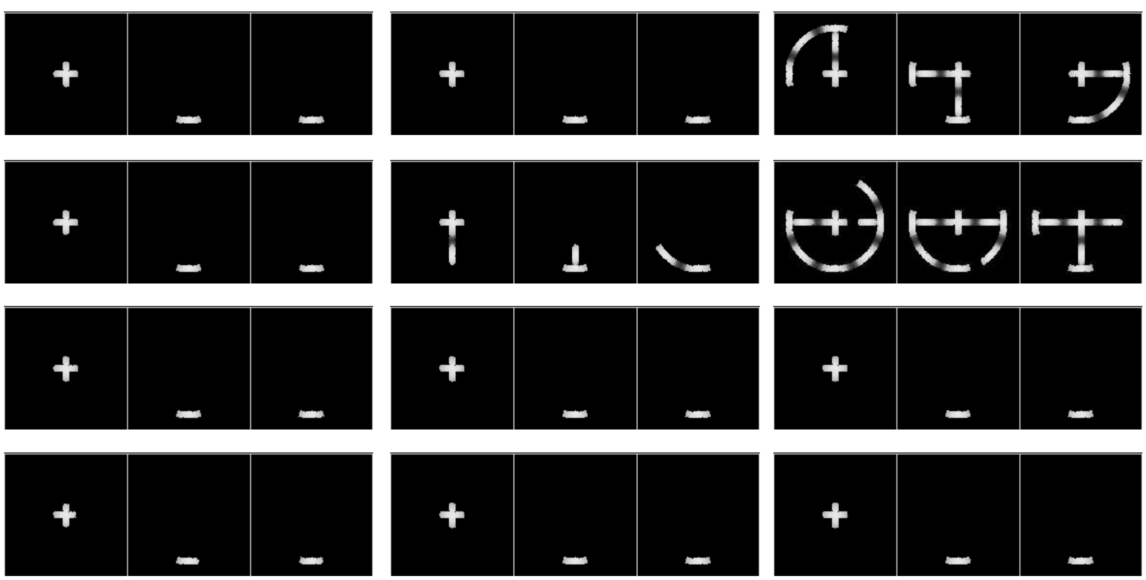
7.1 Topology Identification

As previously described, the topology identification consisted of a three step process, segmentation, thinning, and graph labeling. Each of these processes relied upon the results of the previous process. As such, it is critical that the result of the previous process be acceptable before continuing with the next. For instance, after the segmentation step it is possible to have holes in an otherwise solid binary object. If these holes are not eliminated, then the thinning process would be confused into thinning what should have been a solid object into a series of hollow objects. This would lead the graph labeling process to misidentify the topology. Thus, the user has to insure the results of each process before continuing to the next. This was done by initiating the next process and reviewing the results. If the results were unacceptable then the previous step would be reviewed for corrections. This loop process was iterated upon until the results were acceptable. Usually only a few iterations were needed.

7.1.1 Segmentation

The seeded region growing technique was first applied to synthetic data before being applied to several 3D MRA data sets. The synthetic data provided a test bed for evaluating the region growing technique and the segmention of multimodality images. The only parameters used by the region growing technique to determine vessel labels were the number of standard deviations, M, below the local gray scale mean value, $\bar{S}_1$, equation (4.1) and the size of the search neighborhood, N (Figure 4.1). By adjusting these values it is possible to fine tune the segmenting process. Typically two or three standard deviations and a neighborhood size of seven or nine were needed to obtain satisfactory results. The results of using various neighborhood sizes and different standard deviations on the synthetic 3D MRA data are shown in Figure 7.2. In almost all of the cases the region growing was not able to span any of the gaps between the rings.

Although in these examples, the region growing failed to fully segment the rings the following observations were made for the first 1000 iterations. Increasing the number of standard deviations, M, for a given neighborhood size decreased local mean gray scale value, $\bar{S}_1$, and increased the standard deviation, σ_1. Whereas the current point's mean gray scale value, $\bar{S}_2$, decreased. In most cases, none of the these values changed dramatically. Overall increasing the number of standard deviations decreased the threshold value thus increasing the number of points labeled as vessel (Table 7.1). These results were expected, except that the effect was most dramatic for the smaller neighborhood sizes.



Neighborhood size of five, seven, nine, and 11 with two standard deviations.

Neighborhood size of five, seven, nine, and 11 with three standard deviations.

Neighborhood size of five, seven, nine, and 11 with four standard deviations.

Figure 7.2. Orthogonal MIP images using the region growing on the 128^3 MR image as a function of neighborhood size and standard deviations.

Table 7.1. Number of segmented point versus threshold standard deviation and neighborhood size.

neighborhood size	standard deviation	segmented points	standard deviation	neighborhood size	segmented points
5	2	1101	2	5	1101
5	3	1120	2	7	1081
5	4	5916	2	9	1084
			2	11	965
7	2	1081			
7	3	2050	3	5	1120
7	4	10172	3	7	2050
			3	9	1115
9	2	1084	3	11	1114
9	3	1115			
9	4	1129	4	5	5916
			4	7	10172
11	2	965	4	9	1129
11	3	1114	4	11	1131
11	4	1131			

(a) (b)

Varying the neighborhood size, N, had several interesting effects. Increasing the neighborhood size caused both the local mean gray scale value, $\bar{S}_1$, and its standard deviation, σ_1, to increase. The increase in standard deviation was directly related to having a greater number of samples to chose from, which were typically less uniform in their gray scale values. At the same time, increasing the neighborhood size decreased the mean gray scale value of the current voxel. This was due to having the larger neighborhood incorporating more dark points. Overall increasing the neighborhood size resulted in the region growing process being more conservative, thus reducing the number of points labeled as vessel (Table 7.1). This was not strictly true for every case as a large number of points were labeled as vessel for a neighborhood size of seven with the threshold being set at four standard deviations below the local mean gray scale value. In addition, decreasing the neighborhood size increased the number of gaps between vessel segments. This was due to having a combination of a high local mean gray scale value, $\bar{S}_1$, and a low current mean gray scale value, $\bar{S}_2$.

In summary, a more conservative threshold meant that more seed points were needed to obtain a complete segmentation. While a liberal threshold ran the risk of the region growing leaking out into the background or as described below, the jumping of the region growing between two objects.

The initial segmentation of four registered simulated digital x-ray images using a neighborhood of seven with two standard deviations is shown in Figure 7.3. The region growing was able to segment approximately one-half of the image using a single seed point. However, not only have the rings been segmented but a large plane in the interior has been segmented as well. Also several small spurious rings have been formed. These artifacts demonstrate the classic problem of backprojecting with a limited number of views [49]. The large plane was due to the ring in the XZ plane being horizontal in every view, thus it was not possible to differentiate a solid plate from a ring structure. Similarly for the spurious rings, there are several places where the rings from the XY and YZ planes cannot be differentiated from each other in the original images, thus the artifact.

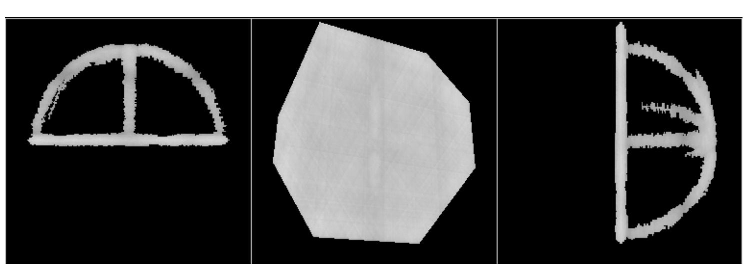


Figure 7.3. Orthogonal MIP s from the 256^3 reconstruction x-ray image.

By combining the MR and X-ray images during the segmentation process we are able to overcome the problem of segmenting only part of the ring structure since each image is able to contribute to the labeling decision (i.e., if the MR failed then the X-ray was used). Further, the MR image was used to guide the region growing and allowed it to grow only into valid regions in the MR image. This was done by placing a global background threshold on the image. This threshold excluded any points with low values from being considered in the region growing.

The results of the region growing using a single seed and examining approximately 200,000 points yielded a complete segmentation of all three rings with no artifacts (Figure 7.4). The 3D image generated was 256^3 in size, twice the original resolution of the MR image. Thus, by combining the two data sets we have been able to take advantage of the best information in each data set, the MR providing low resolution 3D information, and the X-ray images providing high resolution 2D information. This has allowed the generation of a high resolution 3D image which was not possible using either modality alone. Also during this process, the original images were modified, the gaps that occurred due to signal loss have been filled in using the region growing information from the other modality. This has allowed the region growing to continue without having to reseed different regions.

This technique can be fully appreciated when segmenting real images. For instance, the MRA image obtained on one particular patient contained several regions of signal loss (Figure 7.5). Segmenting the MR image alone yielded very little structure (Figure 7.6). However, similar X-ray images were complete but were not segmented because of the many ambiguities that would arise (Figure 7.7). By combining the angiographic images from both X-ray and MR modalities we are able to segment the major portions of the

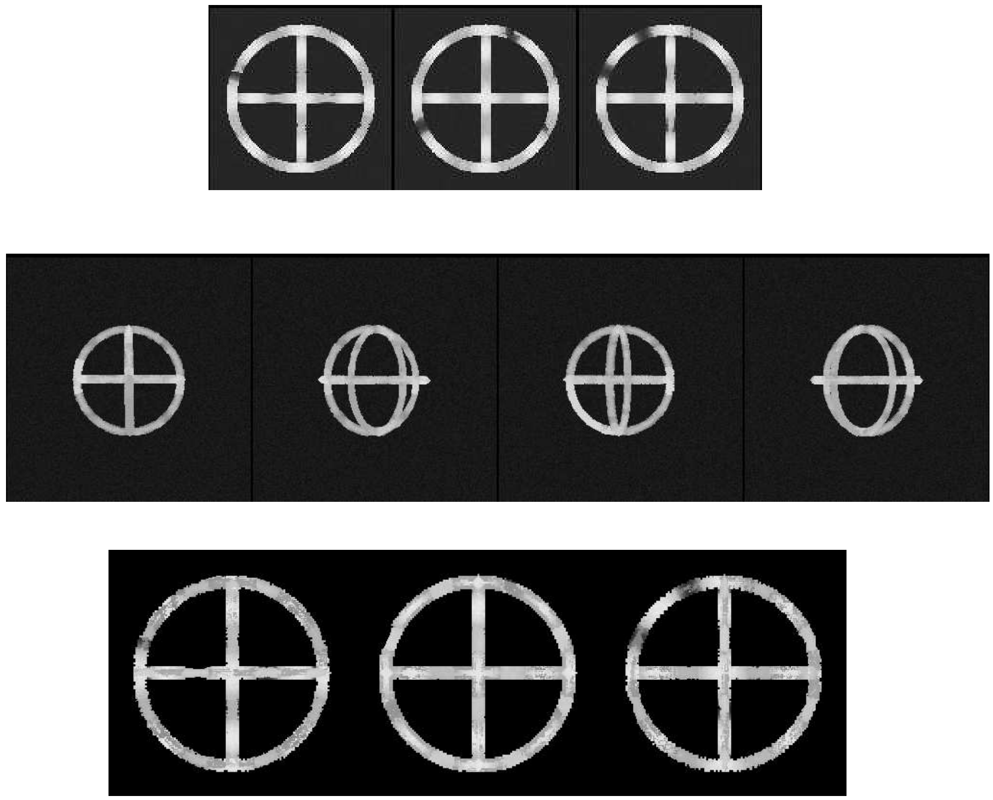
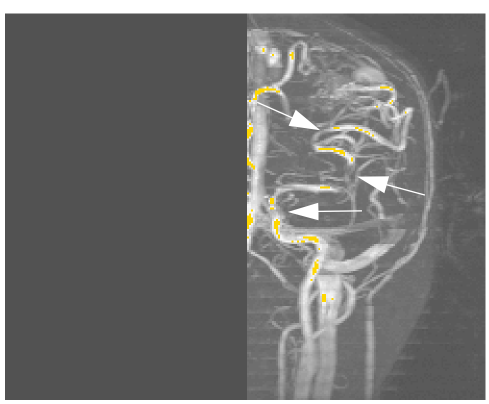


Figure 7.4. Results of the region growing on the MR data (top), X-ray data (middle), and the combined high resolution image (bottom).



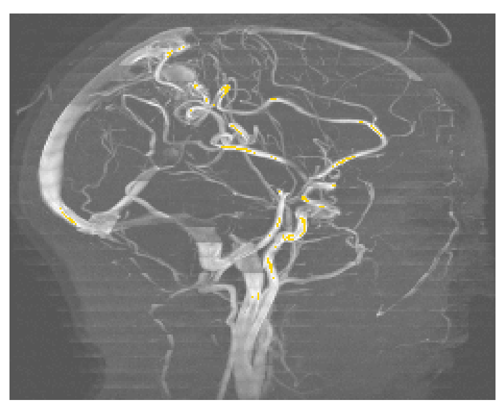


Figure 7.5. Orthogonal MIP views of a patient MRA data set. The arrows points the areas of signal loss.

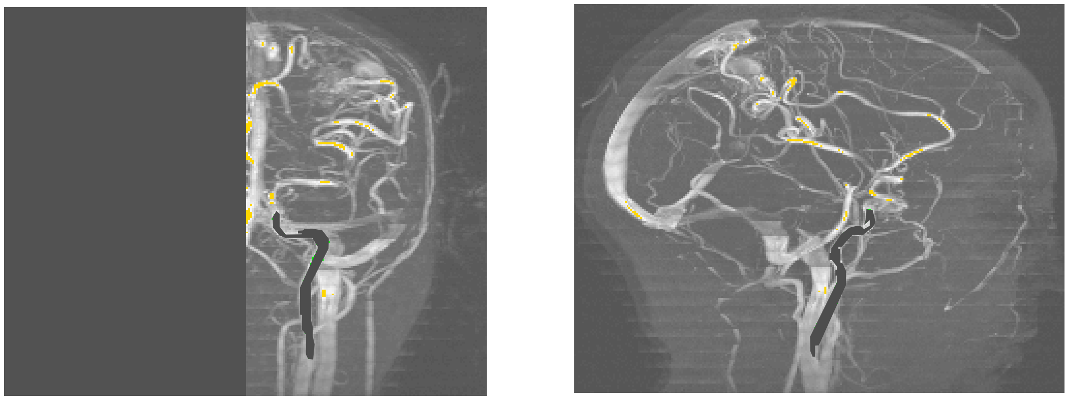


Figure 7.6. Orthogonal views of the MR image after using a single seed point for segmentation.

Figure 7.7. Bi-plane X-ray angiograms of the patient in Figure 7.5.

cerebral artery (Figure 7.8)

In another example that used a single seed point, part of the vertebral/basilar/posterior arterial system located in the central portion of a 3D MRA image was segmented (Figure 7.9). Much of the vasculature was not segmented due to having considerable signal loss in the data. This signal loss was a known characteristic of the imaging technique and does not reflect poor segmentation ability.

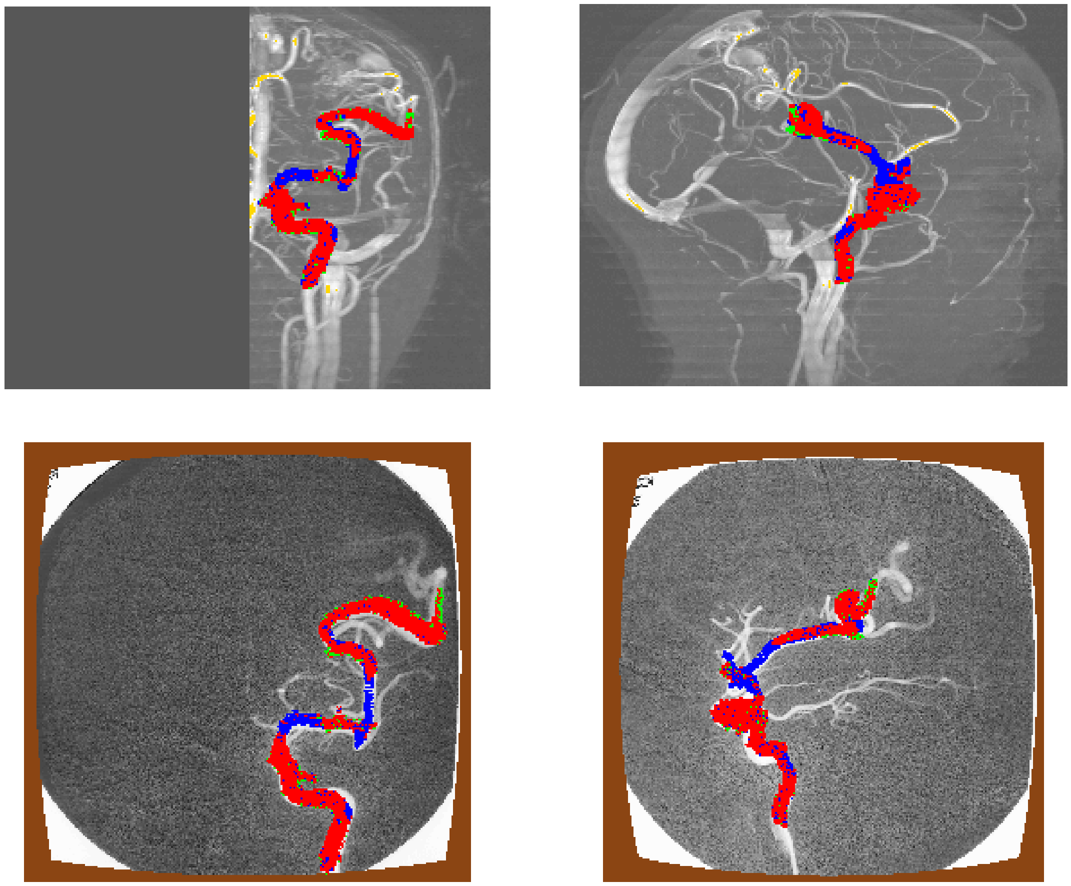


Figure 7.8. Orthogonal views of the MR image after segmentation with the both bi-plane X-ray angiograms.

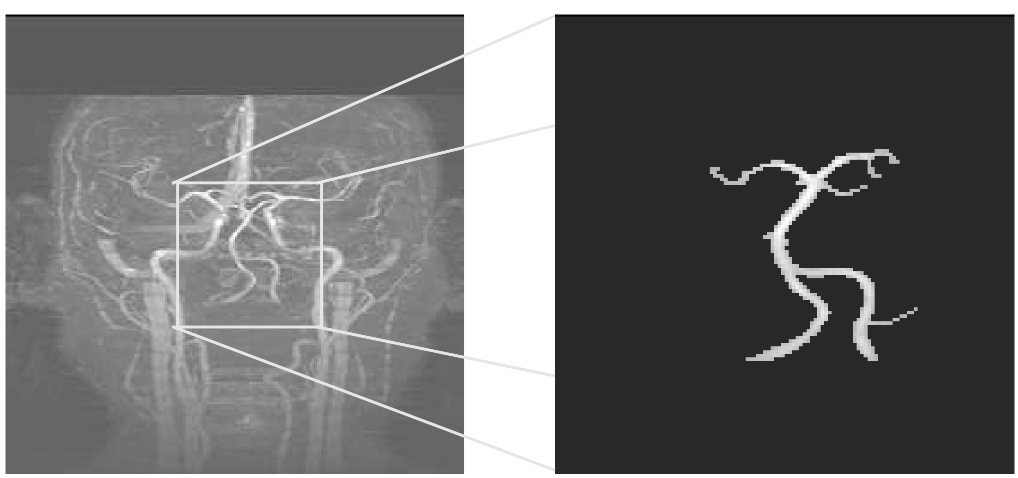


Figure 7.9. Results of vessel segmentation for the central portion of the image using a single seed point.

One of the most difficult problems encountered with the segmentation of real data was the balancing of the region growing to segment noisy regions and the jumping of the region growing between two unrelated high intensity objects. For instance, the segmentation of the cerebral arterial system using a single seed point was stopped after examining 20,000 voxels (Figure 7.10). The segmentation includes not only the arterial system but also part of the venus return. Although there are cases when shunts (connections between two vessels) occur between the two systems this is not the case. In this example, the two seemingly unrelated vessels are in close enough proximity to each other so that one vessel influences the segmentation of the other (Figure 7.11). Upon inspection, it was found that the distance between the two vessels was one to two voxels thick. This caused the segmentation to rely upon the bright voxels in the vein as well as the background voxels near the artery to determine the current mean gray scale value, $\bar{S}_1$ (Figure 7.12). As a result, brighter current mean gray values for $\bar{S}_2$, were obtained for background voxels. This caused the dim background voxels to be treated as though they were small local regions of signal loss surrounded by high intensity signal, and as such, they were falsely labeled as

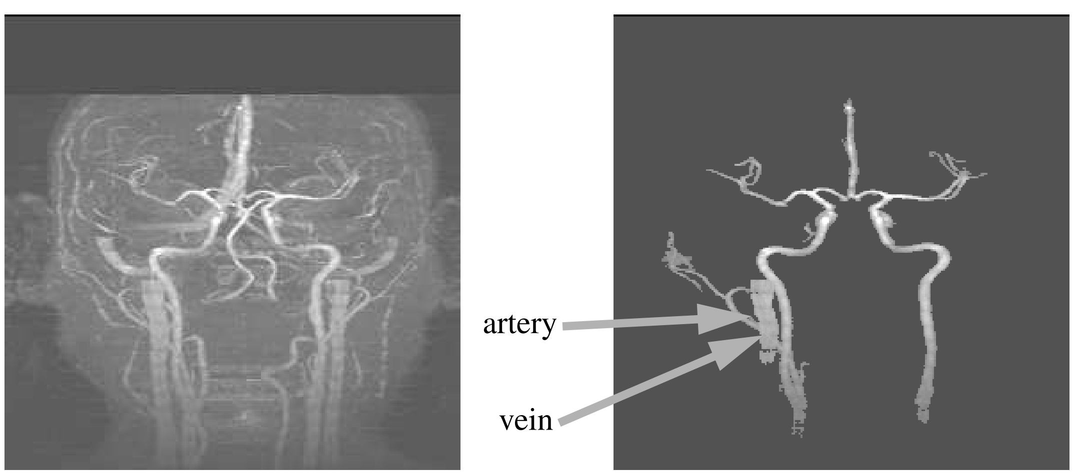


Figure 7.10. Segmentation of the artery showing the jump into the venous return. Note: most of the artery is hidden behind the vein due the maximum intensity projection.

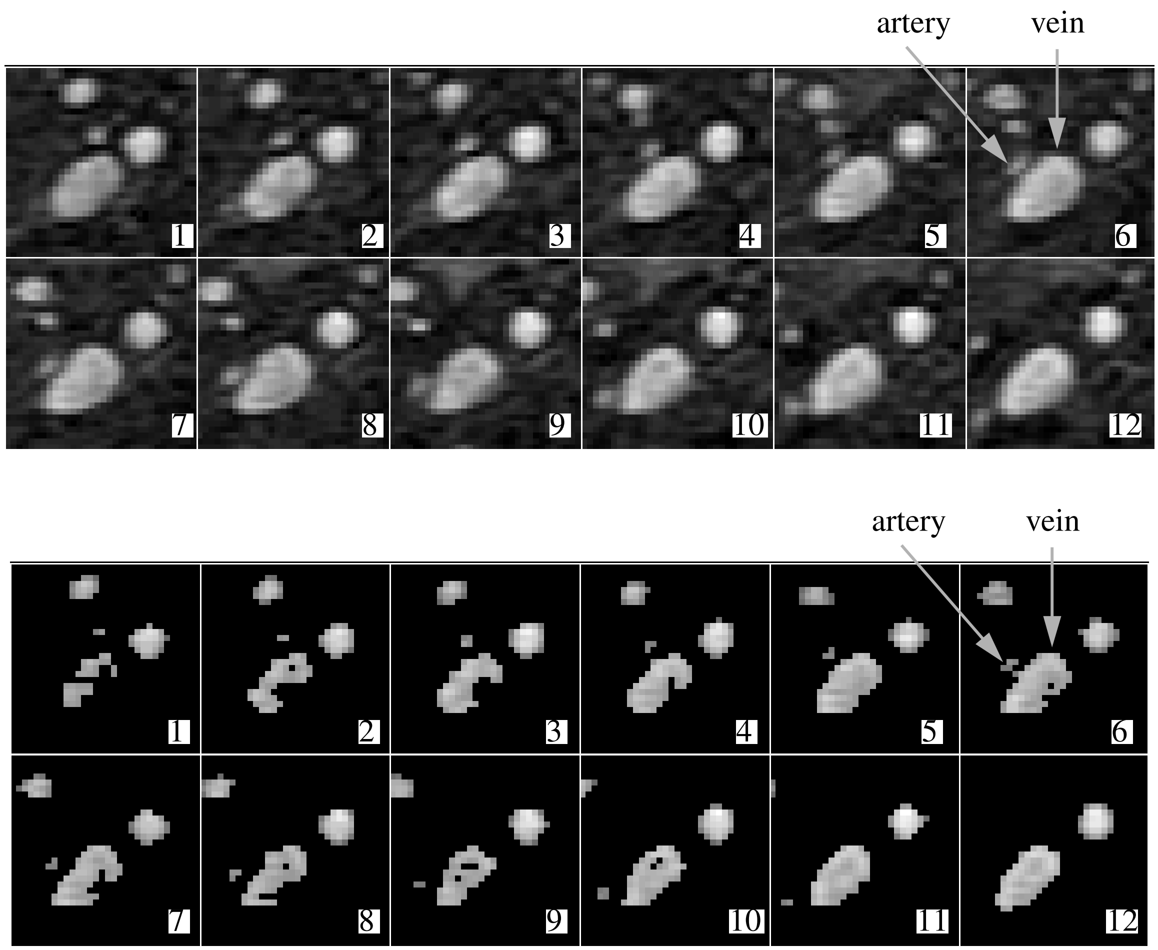


Figure 7.11. Sequence of 12 axial images pre- and post segmentation showing an artery passing close enough to the vein so that the segmentation jumps to the vein. Note the closeness of the artery to the vein in image 6 where the segmentation jumps.

vessel. Had the other vessel not been in close proximity, then only dim background voxels would have been in $\bar{S}_2$ and the region growing would not have grown into the background and into another vessel.

The final segmentation demonstrates the region growing on a MRA carotid data set (Figure 7.13). In this case, only the right common carotid artery and it subsequent vessels have been segmented. Using additional seeds it would be possible to extract the remaining portion of the vasculature.

Although not part of this dissertation, one of the side affects of segmenting the data

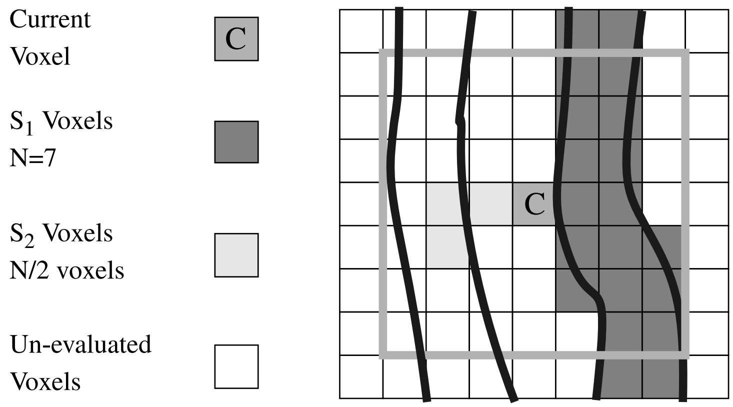


Figure 7.12. The growing of one object into an unrelated object due to local proximity.

Figure 7.13. Segmentation of the right common carotid artery using a single seed point.

was the ability to more easily view the vasculature without overlying vessels or the background present. This technique was especially useful for segmenting “black blood images” where the vessels of interest were dark, surrounded by light background tissue, which was in turn surrounded by dark background air. These images are very difficult to view using traditional volume rendering techniques such as minimum intensity projections.

7.1.2 Thinning

The improved thinning operator has been implemented and applied to both synthetic and real data. Using previously segmented synthetic data we have used the thinning operator to reduce the data into a series of central axis (Figure 7.14). This central axis represents the approximate centerlines of the original object. The results from thinning two different patient data sets are presented (Figure 7.15 and Figure 7.16). Examining the thinned data closely reveals that the algorithm properly thinned the binary data into one

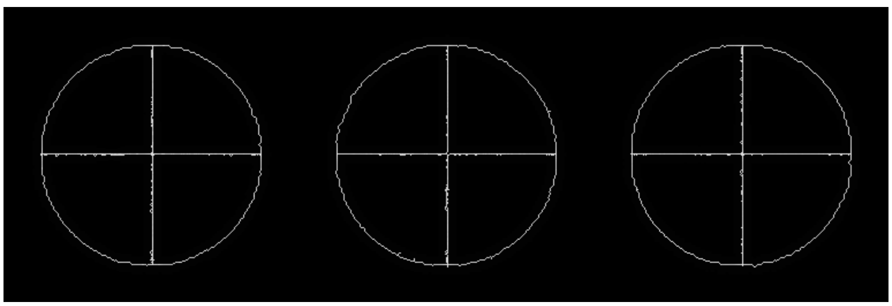


Figure 7.14. Three orthogonal views of the thinned synthetic data.

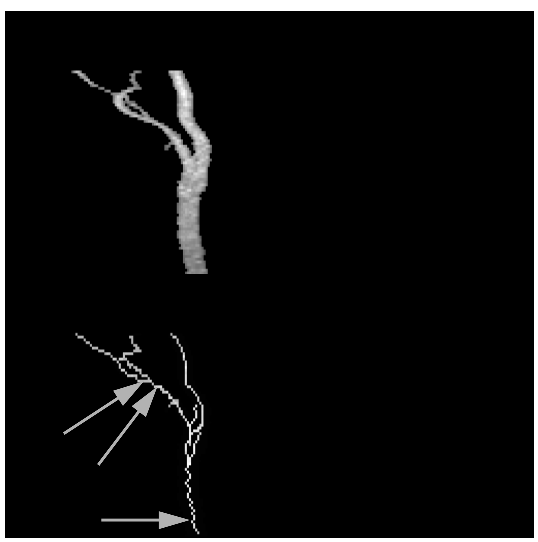


Figure 7.15. Thinned vasculature from the segmented (binary) Figure 7.13 containing three “twigs” as shown with the arrows.

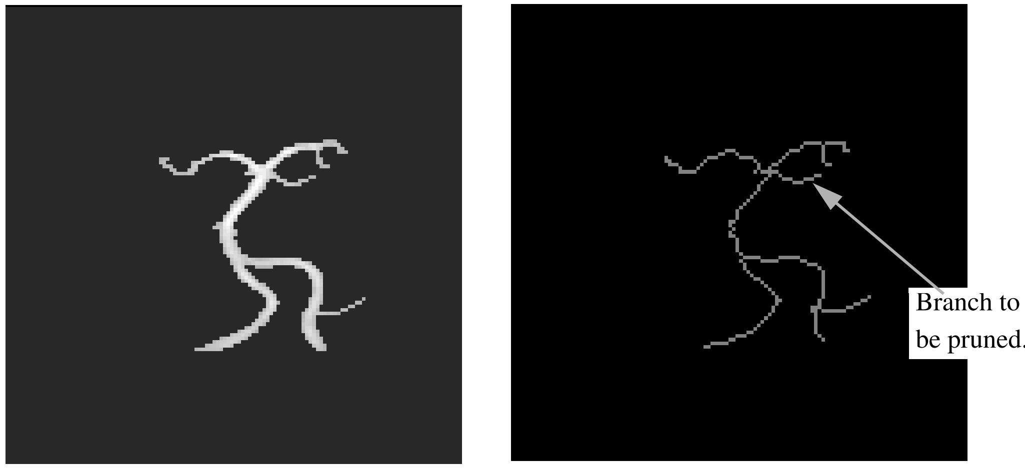


Figure 7.16. Results of thinning the previously segmented (binary) data in Figure 7.9

voxel thick centerlines. However, not all cases were as easy. Many times it was necessary to apply a morphological operator such as dilation and/or hole operator before thinning. These prethinning operators were necessary to ensure that the vessels were indeed solid objects that could be thinned into meaningful centerlines. The dilation operators had to be applied very judiciously because it was possible to grow two vessels together which could result in false cycles being identified during the graph labeling procedure. Several holes and cavities in the segmentation of the common carotid artery of Figure 7.13 were filled before thinning. However, the resulting thinned data were still not perfect as described below. Another problem that arose when thinning noisy data was the tendency to produce many spurious centerlines ("twigs") which complicated the graph formation process. These twigs contain one to five points that did not represent a true vessel centerline but were caused by the segmented data being noisy, i.e., containing irregular boundaries. The process of removing twigs and unwanted centerlines was known as "pruning." The pruning was accomplished by starting at the end point of a particular branch, deleting it and subsequent points until the parent bifurcation was found. After pruning, the remaining data was rethinned so as to ensure that the centerlines were still one voxel thick.

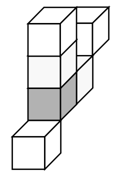
Although difficult to differentiate, the thinned vasculature in Figure 7.15 suffers from having three “twigs.” These twigs were automatically removed since all three consisted of just an end point which was not considered to be a “true” branch. Other vasculature does not demonstrate this problem (Figure 7.16). However, it suffered from a related problem of having a small branch that was not possible to process since a definition for a trifurcation had not been developed and had to be removed. This was accomplished using the same pruning technique described for the spurious twigs.

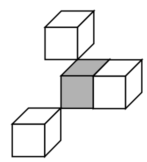
7.1.3 Graph Labeling

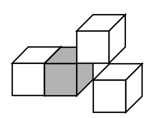
The results of the graph labeling process produced a symbolic representation of branching tree like structures. Cycles were identified as were all other parent-child relationships. It should be noted that only junctions in a 3x3x3 region were originally identified. This was because the junction labeling routine was based on the assumption that only bifurcations would be found. This was a reasonable assumption since most vascular branches are bifurcations, although as previously shown, there are instances of tri- and quad-furcations. A second, more elaborate, junction labeling routine was implemented for higher level branching since their junctions were in a region larger than 3x3x3. (One reason the branch in Figure 7.16 was pruned was because it was part of a trifurcation which the current modeling system was not able to handle.)

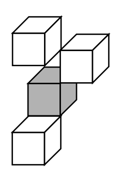
Applying the original junction labeling technique to the previously thinned data in Figure 7.16 resulted in four junction nodes (Figure 7.17). Examination of the four junctions yielded only one junction containing mutual junctions. For this case, two of the points had the same weightings, as such one of the two was chosen arbitrarily as the junc-

Type	X	Y	Z	Weight
main junction	149	103	103	
exit point	149	102	102	
exit point	149	105	104	
exit point	149	103	105	
junction point	149	103	103	13
junction point	149	104	103	11
junction point	149	103	104	13
main junction	129	123	121	
exit point	130	123	121	
exit point	128	124	122	
exit point	128	124	120	
junction point	129	123	121	7
main junction	135	131	145	
exit point	134	131	145	
exit point	136	131	146	
exit point	136	132	144	
junction point	135	131	145	9
main junction	151	132	152	
exit point	152	132	153	
exit point	151	131	151	
exit point	150	133	153	
junction point	151	132	152	7









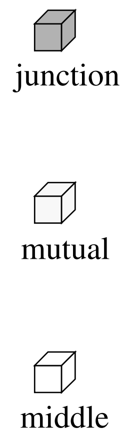


Figure 7.17. Examination of the four junctions from Figure 7.16 using the original junction labeling technique. Only one junction, (node 2) contains mutual junctions; points with more than three neighbors. (Weightings: Face-5, Edge-3, Corner-1).

tion. This information along with the other point labeling was used to create a symbolic representation that allowed one to traverse the tree starting at the root node (0) and continuing to any of the subsequent junction nodes (1, 2, 3, and 7) and six end nodes (4, 5, 6, 8, and 9) (Figure 7.18).

The second junction labeling routine implemented used a simple region growing technique to identify a junction. If a point had three or more neighbors, it was placed in a queue. Points in this queue had each of their foreground neighbors examined. If the neighbor had three or more neighbors it was also placed into the queue since it was a possible junction point. Otherwise, if the neighbor has two or less neighbors, it was considered to be a part of a branch exiting from the bifurcation and was counted as such. When all of the points in the queue were examined, the point that had the greatest neighbor weight was chosen as the main junction point (if there were multiple points with the greatest weight, the point closest to the junction centroid was chosen as the main junction point) (Figure 7.19).

Applying the new labeling operator to the data in Figure 7.16, again resulted in four junction nodes. However this time, the extra branch was not pruned, thus the third junction in Figure 7.17 now had five possible junction points and four exit points instead of one junction point and three exit points (Figure 7.20). Also, the new junction was labeled according to its neighbor weighting.

The result of the new junction labeling routine is demonstrated on the synthetic data. These data consisted of six junctions all containing a trifurcation (Figure 7.21). . Each of the junctions contained at least three or more points which could have served as the main junction because of their maximal weighting. Since multiple points existed, the junction point was selected to be the point closest to the junction centroid. Combining this in-

Type	X	Y	Z	Level	Node
Node	118	99	96	0	0
Child	129	123	121	1	1
Node	129	123	121	1	1
Parent	118	99	96	0	0
Child	149	103	103	2	2
Child	135	131	145	2	3
Node	149	103	103	2	0
Parent	129	123	121	1	1
Child	151	099	098	3	4
Child	164	101	110	3	5
Node	135	131	145	2	3
Parent	129	123	121	1	1
Child	109	115	147	3	6
Child	151	132	152	3	7
Node	151	99	98	3	4
Parent	149	103	103	2	2
Node	164	101	110	3	5
Parent	149	103	103	2	2
Node	109	115	147	3	6
Parent	135	131	145	2	3
Node	151	132	152	3	7
Parent	135	131	145	2	3
Child	158	121	151	4	8
Child	153	126	147	4	9
Node	158	121	151	4	8
Parent	151	132	152	3	7
Node	153	126	147	4	9
Parent	151	132	152	3	7

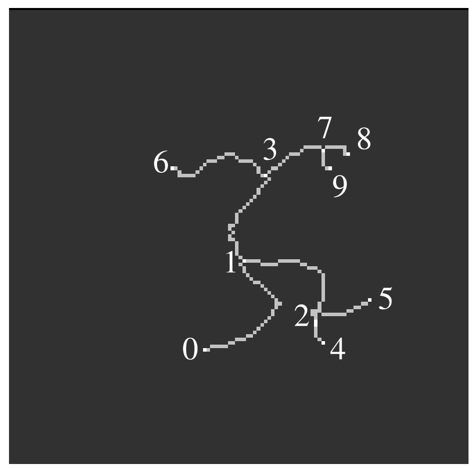


Figure 7.18. Symbolic representation of the vascular tree shown. The tree was labeled according to the output of the level-order-search. Node 0 in the image was the root node.

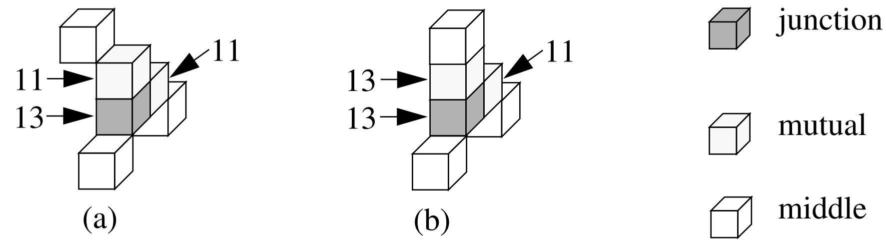


Figure 7.19. Two junctions where (a) contains a single best connected point whereas (b) contains two, thus the junction was the point closest to the centroid.

Type	X	Y	Z	Weight
main junction	136	131	145	
exit point	135	133	143	
exit point	138	133	145	
exit point	134	131	145	
exit point	137	132	147	
junction point	136	132	144	08
junction point	136	131	145	16
junction point	137	132	145	10
junction point	135	131	145	14
junction point	136	131	146	10

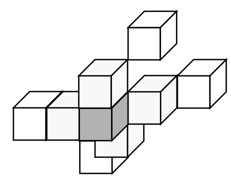


Figure 7.20. Reexamination the third junction from Figure 7.16 using the new junction labeling technique which allows for junctions larger than 3x3x3 region.

formation with the rest of the point labeling allowed for the construction of a graph of the synthetic data (Figure 7.22). In this example, it is now possible to see the effects of having cycles. Several of the nodes have multiple parents. The presence of multiple parents indicated cycles in the graph. No attempt was made to identify the exact nature of the cycles, only that at least one cycle was present per multiple parent. For Figure 7.22, there were seven multiple parents, but eight cycles were present.

Type	X	Y	Z	Weight
main junction	129	128	33	
exit point	129	126	33	
exit point	131	129	33	
exit point	126	129	33	
exit point	129	131	33	
junction point	129	127	33	13
junction point	129	128	33	13
junction point	130	128	33	11
junction point	128	129	33	13
junction point	127	129	33	13
junction point	128	130	33	11
main junction	129	33	128	
exit point	129	3	3126	
exit point	131	33	129	
exit point	126	33	129	
exit point	129	33	131	
junction point	129	33	127	13
junction point	129	33	128	13
junction point	130	33	128	11
junction point	128	33	129	13
junction point	127	33	129	13
junction point	128	33	130	11
main junction	33	129	128	
exit point	33	129	126	
exit point	33	131	129	
exit point	33	126	129	
exit point	33	129	131	
junction point	33	129	127	13
junction point	33	129	128	13
junction point	33	130	128	11
junction point	33	128	129	13
junction point	33	127	129	13
junction point	33	128	130	11

Type	X	Y	Z	Weight
main junction	129	223	128	
exit point	129	223	126	
exit point	131	223	129	
exit point	126	223	129	
exit point	129	223	131	
junction point	129	223	127	13
junction point	129	223	128	13
junction point	130	223	128	11
junction point	128	223	129	13
junction point	127	223	129	13
junction point	128	223	130	11
main junction	223	129	129	
exit point	223	127	129	
exit point	223	129	127	
exit point	223	131	129	
exit point	223	129	131	
junction point	223	128	128	09
junction point	223	129	129	13
junction point	223	130	129	13
junction point	223	129	130	13
main junction	129	129	223	
exit point	127	129	223	
exit point	129	127	223	
exit point	131	129	223	
exit point	129	131	223	
junction point	128	128	223	09
junction point	129	129	223	13
junction point	130	129	223	13
junction point	129	130	223	13

Figure 7.21. Results of the junction labelling for the synthetic data. The six main junctions each contain four exit points, and four to six points with three or more neighbors (mutual junction points).

Type	X	Y	Z	Level	Node
Node	129	128	33	0	0
Child	223	129	129	1	1
Child	129	33	128	1	2
Child	33	129	128	1	3
Child	129	223	128	1	4
Node	223	129	129	1	1
Parent	129	128	33	0	0
Child	129	33	128	1	2
Child	129	223	128	1	4
Child	129	129	223	2	5
Node	129	33	128	1	2
Parent	129	128	33	0	0
Parent	223	129	129	1	1
Child	33	129	128	1	3
Child	129	129	223	2	5
Node	33	129	128	1	3
Parent	129	128	33	0	0
Parent	129	33	128	1	2
Child	129	223	128	1	4
Child	129	129	223	2	5
Node	129	223	128	1	4
Parent	129	128	33	0	0
Parent	223	129	129	1	1
Parent	33	129	128	1	3
Child	129	129	223	2	5
Node	129	129	223	2	5
Parent	223	129	129	1	1
Parent	129	33	128	1	2
Parent	33	129	128	1	3
Parent	129	223	128	1	4

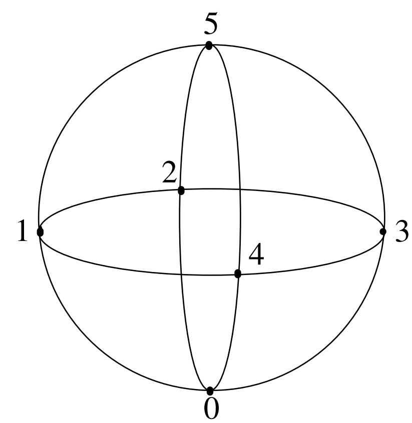


Figure 7.22. Graph labeling of the synthetic data. Cycles are present due to several nodes having multiple parents.

7.1.4 Topology Identification Discussion

The seeded region growing technique coupled with an adaptive thresholding was able to accomplish the needed goals, segment strongly connected regions while being able to adapt to a varying gray scale. The adaptive thresholding is a simple and computationally inexpensive technique. The ability to vary the neighborhood size and/or number of standard deviations from the local mean gave enough flexibility to obtain usable results under a variety of conditions. However, the conditions did vary enough that it was rarely possible to segment the complete vasculature without using multiple seeds or having to use some user interaction or morphological operators to insure that solid objects were segmented.

The main problems with the segmentation were the jumping of the region growing between two distinct objects and the leaking of the segmentation into the background. Requiring the region growing to be more conservative typically would prematurely stop the segmentation, forcing the user to enter more seeds points.

Before beginning the skeletonization process one must decide on what results are needed. If the goal is to have a representation that can be used to reconstruct the object then a medial axis is needed. Conversely, if the goal is to determine the local topology then a less restrictive central axis can be used. We have chosen the latter and have made improvements to an existing 3D thinning algorithm which provided such results. These improvements greatly reduce the complexity of the thinning process by employing eight subfields instead of the original six subfields and three “checking planes.” In addition, the algorithm is complete in that both the connectivity and number of connected components are used to determine whether a voxel can be thinned. The algorithm was able to thin the data into a one voxel thick central axis. The exception to this are the bifurcations where multiple thick regions were encountered.

The graph formation technique developed relies on the assumption that the thinned image was no more than one voxel thick. The graph search used a simple depth-first-search (or level-order-search) by iteratively tracking each voxel until a junction or end voxel was found. The junction labeling relied on a region growing technique to identify all of the points locally contained in the junction. Implicit in the labeling and graph search was the identification of the object's topology. Although not all of this information is currently being fully utilized (the ability to traverse the tree given any node), it does serve as the basis for selecting the template models during the surface recovery stage.

7.2 Model Seeding

Having identified the topology of each object we are then able to select from our "tool box" the proper template models to use. As previously described there are two principal templates devised for this application, a vessel bifurcation and a vessel body (Figure 7.23). These surfaces, as well as all other vessel body surfaces were bi-quintic B-spline surfaces with and without internal knots. The vessel body is composed of two individual surfaces, whereas the bifurcation is composed of six surfaces.

Figure 7.23. Initial (predeformation) surfaces for the vessel body (composed of two surfaces) and vessel bifurcation (composed of six surfaces).

7.2.1 Single Template

The simplest use of the templates was for a single vessel (Figure 7.24). In this case, the topology identification process identified a single vessel body. Using this information we are able to produce a B-spline surfaces for it (Figure 7.25).

7.2.2 Multiple Templates

Using the results of the topogology identification from Figure 7.18 which contained both vessel bodies and vessel bifurcations we are able to automatically generate the complete vessel structure using the two templates (Figure 7.26). For each of the bifurca-

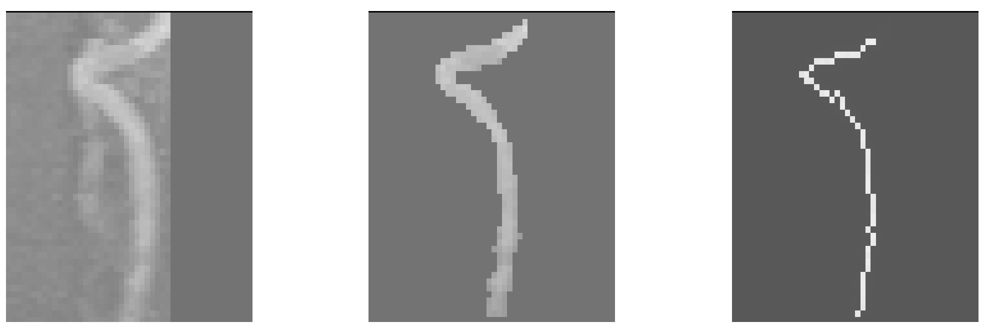


Figure 7.24. Real data (left), results of a 3D segmentation (center), and thinning/graph labeling (right) of a section of the vertebral artery.

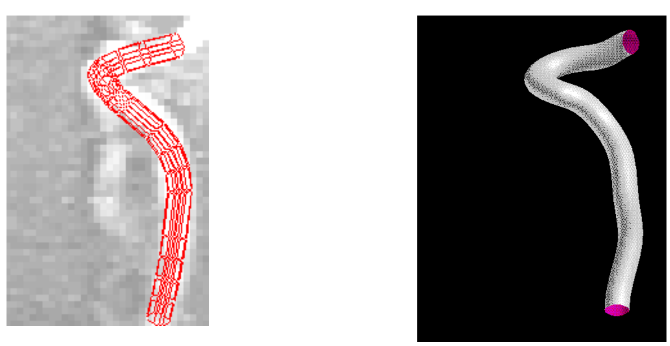


Figure 7.25. Results of the model seeding for a single template showing a control mesh (left) and a rendered image (right).

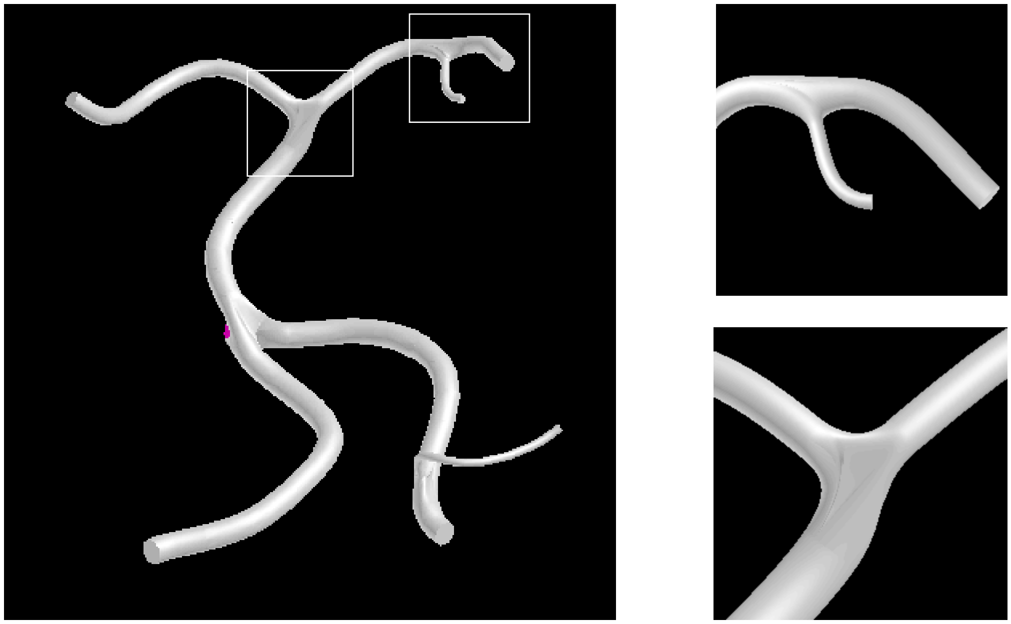


Figure 7.26. Results of the model seeding for a vessel structure containing four bifurcations and nine vessel bodies, along with two close-ups of the bifurcations.

tions, the same template model was used and was able to adapt to the geometry of the different bifurcations. The success of this template was due to the modeling code developed which, as previously discussed, replied on six parameters, the vessel diameter and direction vector for each vessel. Knowing only this information we are able to deduce an approximation that was acceptable for almost every case.

An other example of the model seeding is from Figure 7.15 where there are six bifurcations and thirteen vessel bodies (Figure 7.27). (Note: all of the vessels were given the same diameter for easier visualization). Many times it is difficult for a physician to visualize the vascular structure when viewing a single image. This example clearly shows one of the benefits of the model seeding as it is much easier to visualize the vasculature structure in the right image than the left image of Figure 7.27. Knowing the exact vascular structure is very important during medical procedures where a catheter may be introduced, wrongly

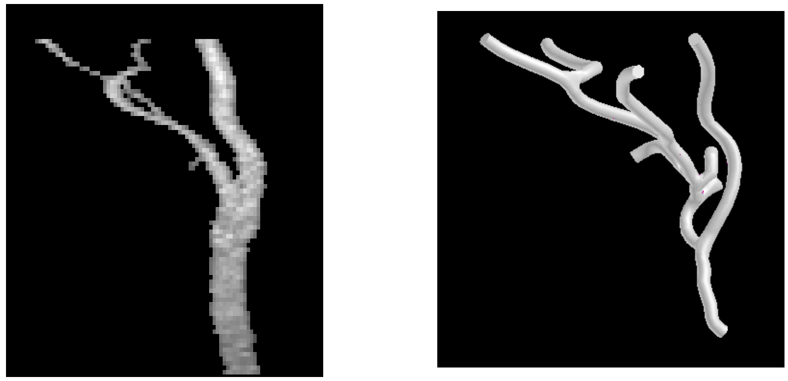


Figure 7.27. Results of the model seeding for a complicated vascular image (right), based upon the segmented image (left).

placing a catheter not only wastes time but can endanger the patient.

7.2.3 Modeling Seeding Discussion

In this section we have sucessfully demonstrated the ability of the model seeding. A total of nine very different bifurcations have been successfully modeled using a single template. The vessel bodies have been used both alone and inconjuction with the vessel bifurcations for form two complete models. These models can be used as a first approximation for visualizing the vascular structure or as will be demonstrated latter, in the deformation process.

7.2.4 Tangent Plane Continuity

Originally, we proposed to combine the deformation process and the tangent plane continuity into one process, unfortunately, both processes are computationally very expensive when working with large numbers of control points and many boundaries. Instead, we maintain the tangent plane continuity in one of two stages, after the model seeding but before the deformation process or after the deformation process.

For a single bifurcation and its associated parent/children vessels, tangent plane continuity was enforced after the model seeding because the initial description of the surfaces in bifurcation template were not tangent plane continuous (Figure 7.28). This was by far the easiest case, since enforcement only had to occur along six boundaries since the other 15 boundaries were already tangent plane continuous. Unfortunately, because all of the boundaries are dependent on each other we had to consider all of the boundaries together not just those boundaries that were not tangent plane continuous. For a single bifurcation, this resulted in 436 different tangent plane constraints, 242 distance constraints for the "move as little as possible" constraint, and 726 degrees-of-freedom. Solving this type of constraint system quickly proved to be impractical.

As a result, the first experiment was to enforce the tangent plane continuity only along those six boundaries of the 12 surfaces that needed it. This resulted in 134 different tangent plane constraints, 52 distance constraints, and 156 degrees-of-freedom. The constrained optimization was run as single process even though it could have been split into two parallel processes since none of the boundaries were independent of each other. This

Figure 7.28. Initial bifurcation before the enforcement of tangent plane continuity. The arrows indicate the boundaries that are not tangent plane continuous.

proved to provide satisfactory results as the other boundaries were only slightly effected by the optimization process (Figure 7.29). However, it took 10,000 iterations and over four days of CPU time on a Sun Ultra Sparc to achieve these results!

The second experiment was to make the problem “parallel” by tessellating the surfaces into smaller patches which were $C^{(0)}$ continuous. These patches contained the same tangent plane discontinuities as before but their effects were much more local. In addition, the boundaries were held constant, which although reducing number of degrees-of-freedom did not impact the ability to force tangent plane continuity since the initial solution was very close to the final solution; however this was not always the case as was seen in enforcing the constraints postdeformation process discussed below. This resulted in two constraint processes, each having 45 tangent plane constraints, 22 distance constraints, and 66 degrees-of-freedom. The constrained optimization was then run as two parallel process for 10,000 iterations. Because we localized the tangent plane discontinuities the optimization took approximately, 45 minutes to complete, and was able to give better results than single process (Figure 7.30 and Figure 7.31).

The second stage where tangent plane continuity was enforced was after the defor-

Figure 7.29. The bifurcation after 10,000 iterations to enforce tangent plane continuity along 6 of 21 boundaries.

Figure 7.30. Parallel enforcement of the tangent plane continuity.

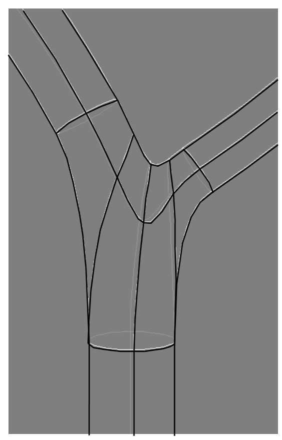


Figure 7.31. Secondary tessellation using 28 patches (the original consisted of 12 patches).

mation process since the deformation process moved points without taking into account the geometric constraints. Two different techniques were used to enforce the tangent plane continuity, first the boundaries were held constant as done above and the second, where the boundaries were allowed to float. Allowing the boundaries to float increased the number of degrees-of-freedom from 66 to 102. Not allowing the boundaries to float restricted the optimization process such that tangent plane continuity could not be met thus giving unacceptable results (Figure 7.32). Once the boundaries were allowed to float there was enough freedom to obtain tangent plane continuity (Figure 7.33).

Figure 7.32. Enforcement of the tangent plane continuity holding the boundaries constant.

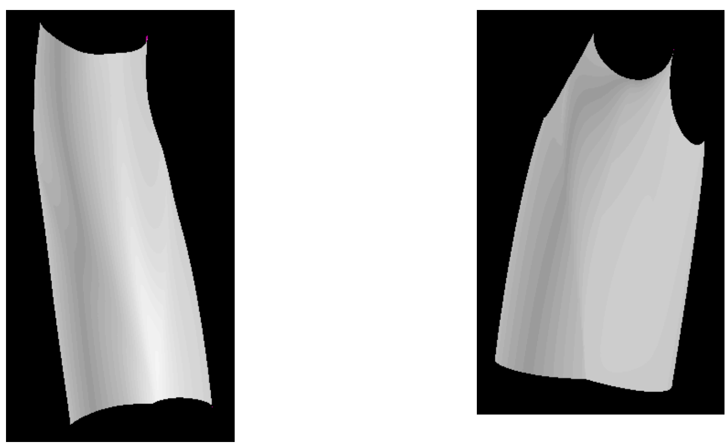


Figure 7.33. Enforcement of the tangent plane continuity after allowing the boundaries to float.

7.3 Deformation Process

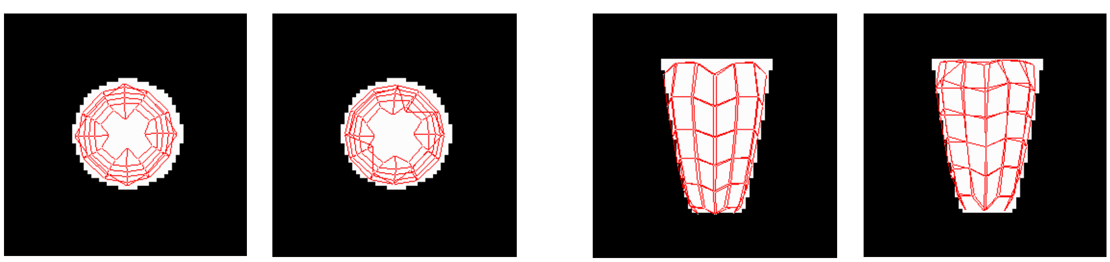
Several methods for performing the deformation process were explored in this research. Previously, three different techniques were described along with their iterative solutions for obtaining the displacements, a massless system using Euler's Method, and two mass systems using Runge-Kutta and Central Differences. Each of these systems are an approximation to the actual solution and when implemented, had their own advantages and disadvantages. For instance, Euler's Method and Central Differences both required a computationally expensive matrix inversion using a singular value decomposition (SVD). Steps were taken to minimize this computation by approximating the matrix with a circulant matrix and allowing it to be inverted using discrete Fourier transforms (DFT). This approximation, coupled with Euler's Method, as will be demonstrated, was able to give satisfactory results. Unfortunately, the use of circulant matrix with Central Differences was found to be highly unstable during all phases of this research, and was withdrawn as potential solution. However, using an SVD matrix inversion, Central Differences was found to give satisfactory results. Thus, we present results of the deformation process using five different solution techniques, Euler's Method with a SVD, Euler's Method with a DFT, Central Difference with a SVD, and First and Fourth Order Runge-Kutta.

Also presented are comparisons between the two different types of deformation potentials developed, direct-gravity and direct-springs. As previously described, the gravitational potentials were inversely proportional to the distance between surface and potential (surface boundary) whereas the springs were directly proportional. The direct (or directional) came from allowing each potential only to affect a single node on the surface.

To compare the results of the system, the three items were used as comparisons: number of iterations, the accuracy, and the precision. For the later two, comparisons were made against a human segmentation of the objects. These segmented images were considered to be "truth images." The comparisons were done by converting the resulting FEM mesh outlining the boundary of the object into a solid object. This allowed for a voxel-by-voxel comparison between the hand segmented image and the deformed model. Those voxels that were different in the two images were considered to be misses. This comparison technique was not without fault, as there were errors due to discretizing the mesh (i.e, truncation/round-off errors) and the segmentation was done only once by a single observer which was also source for errors. Thus, the missed point counts can only be construed as a "reasonable" measure of performance.

7.3.1 Circulant Matrices

The first experiments were to evaluate the usefulness of the circulant matrices, which had some interesting results. For instance, we have performed the deformation process using massless system with a circulant damping matrix in both instances on a noiseless binary image of a right cylinder (Figure 7.34). However, one matrix had phantom nodes introduced into it while the other a circulant matrix was used directly (Figure 7.34).[1] Although not fully apparent in the figure, the addition of phantom nodes had a significant impact when comparing the accuracy. Without phantoms nodes, of the 15,282 truth points, there were 3233 (20.4%) missed points whereas there were 2433 (15.4%) missed points when phantom nodes were introduced. (For comparison, an SVD solution had 2869 (18.1%) missed points.) Thus, for this example, the circulant matrix with phan-



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

Figure 7.34. Top and side views of a cylinder comparing the deformation process using a circulant matrix with (b and d) and without (a and c) phantoms nodes.

1. Phantom nodes are never needed for the 2D case since damping matrix is circulant, as such, the 2D results with the circulant matrix are discussed in the next section.

tom nodes gave better results than the SVD solution. As will be demonstrated with other examples this was the norm. The only drawback is that we must use more “nodes” which increases the computational expense.

7.3.2 2D Deformations

The simplest example of shape recovery involved the recovery of the arterial lumen in an MR angiogram from human subject (Figure 7.35).[2] An initial comparison made was against the results of the region growing segmentation with the hand segmented image (Figure 7.36). It was found that for the right carotid artery there were seven missed points out of 122 truth points (5.7%), whereas the left artery had nine missed points out of 158

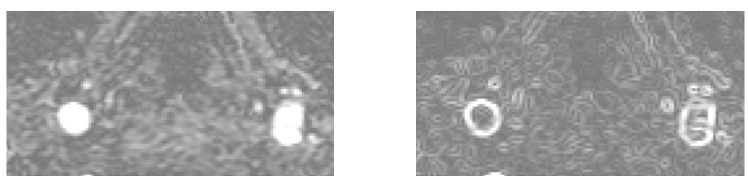


Figure 7.35. 2D slice of the carotid arteries from a MR angiogram (left) and its response to a 2D Nevatia-Babu edge filter (right).

Figure 7.36. Results of the initial region growing segmentation (left) and the hand segmentation (right).

2. The images are presented in a standard medical presentation. This presentation requires that the observer look from the patient’s feet towards the head. As such, the patient’s left is seen on the right side of the images, the converse is true of the patient’s right, it is seen on the left side of the image.

truth points (5.7%). Although the region growing was meant only for obtaining a crude segmentation these results show that in some cases very accurate results are possible

Since 2D images were used, no attempt was made to use the automatic template seeding process (although it was possible to use the results of the segmentation to obtain a center of mass for each artery and an approximate radius). Instead, circles, approximated by 12 points were used as the initial guess. The standard technique in the deformation process involved using direct gravity potentials and solving for the displacements using a massless system (Euler's Method) inconjuction with a circulant matrix approximation (DFT). The results of DFT deformation took 37 and 57 iterations for the right and left arteries, respectively. The results for the right artery are shown as a time series in Figure 7.37. Using the hand segmented data as the standard, there were 20 and 33 missed points out of possible 122 and 158 truth points, (16.4% and 20.9%) respectively. Although these results, were not as good as the segmented results, they were deemed to be acceptable.

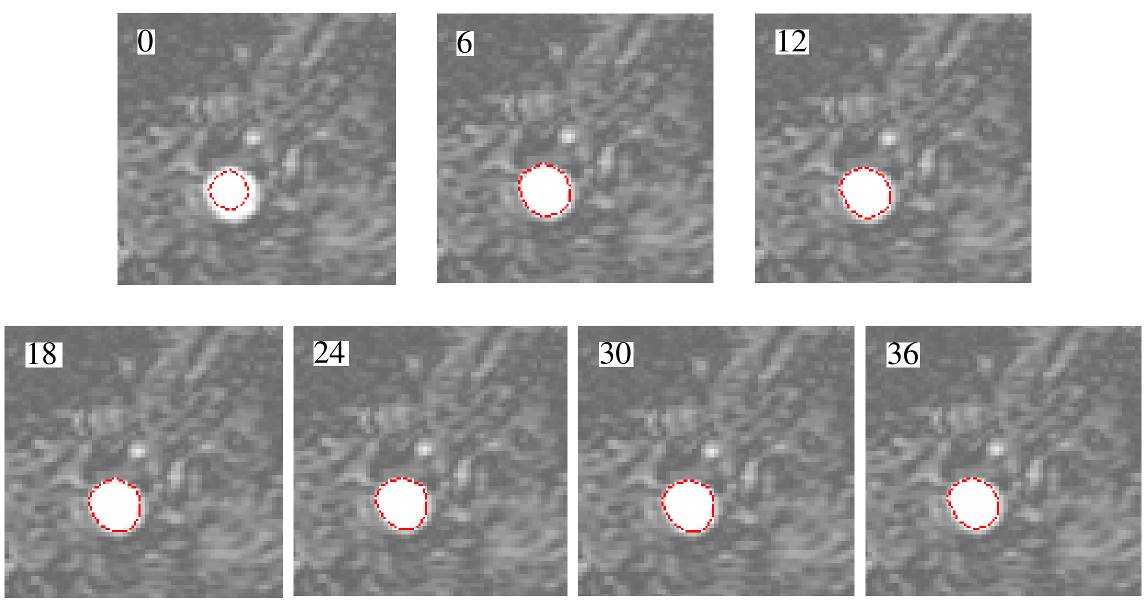


Figure 7.37. Results of the deformation process at iteration 0, 6, 12, 18, 24, 30, and 36 using a DFT solution.

For the first set of experiments, a comparison was made between the effect of the strength of the image potentials while solving for the displacements using the five different methods. The results of the comparisons with the truth image and the number of iterations are shown in Table 7.2. Several observations can be made from these results. First, a comparison between using a matrix inverse (SVD) and a circulant matrix (DFT) was made. Previously, it was noted that for the 2D case, the stiffness matrix was already circulant and that either method would give acceptable results.[3] This held true in all cases. However, the number of iterations was where the true savings was found. In all cases, the

Table 7.2. Number of iterations and missed points using the direct gravity potentials. (The image potential strength is noted in parenthesis).

	Right Artery (10X)		Left Artery (10X)	
	Iterations	Missed points	Iterations	Missed points
Euler Matrix Inversion (SVD)	49	18	58	29
Euler Circulant Matrix (DFT)	37	20	36	36
Central Differences	588	21	506	49[1]
Runge-Kutta First Order	984	19	867	36[1]
Runge-Kutta Fourth Order	573	20	527	35

	Right Artery (1X)		Left Artery (5X)	
	Iterations	Missed points	Iterations	Missed points
Euler Matrix Inversion (SVD)	92	27	84	34
Euler Circulant Matrix (DFT)	71	27	54	34
Central Differences	600	32	635	35
Runge-Kutta First Order	775	23	646	32
Runge-Kutta Fourth-Order	598	32	640	36

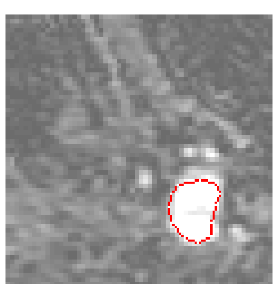
1. Unacceptable results due to the boundary being attracted to spurious potentials causing a self intersection in the boundary.

3. It should be noted that the inversion of the stiffness matrix in its original form was always ill-conditioned. As such it was necessary to use a Singular Value Deposition (SVD) and remove the singular values. For the circulant matrix case, the eigen values resulting from the DFT that were approximately zero (singular) were also removed. Thus, for both cases, the inverse was an approximation.

number of iterations for the circulant matrix was approximately the same or less, on average, 20% less.

Another observation was that, as the strength of the image potentials increased the accuracy of the shape recovery also increased. This would seem to be a natural result since the potentials are better able to lock onto the boundary. However, this was not always the case, as demonstrated with the left artery when using a mass system with potentials at 10x. In this case, the boundary was incorrectly attracted to spurious potentials due to the large potentials which caused the boundary to move from its initial position very fast. This resulted in a large inertial force which in turn caused the boundary to over shoot the true boundary, thus allowing it to be attracted to the spurious potentials. This ultimately caused a self-intersection to occur when the boundary retracted into its final position (Figure 7.38). Thus, when using a mass system strong image potentials are not always advantageous, and in fact, it is better to have neither weak nor strong potentials as the boundary should deform in a smooth manner. (Extreme values on either side of the spectrum could lead to instabilities.)

The next observation is between the first and fourth-order Runge-Kutta methods. Each method gives comparable results for each of the four different cases. Although, the



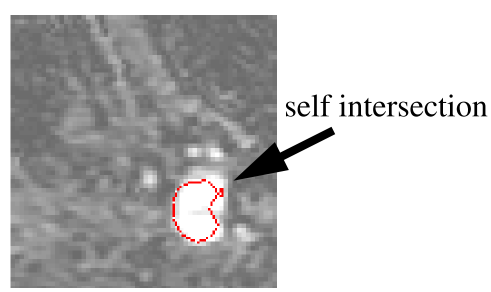


Figure 7.38. Comparison of weak (5X) and strong (10X) potentials with a mass system using Central-Differences.

forth-order solution takes fewer iterations, it takes three more evaluations than a first-order solution but the results are not consistently better. Thus, for four times the work the results are typically no better than the first-order solution. As such, a first-order solution may be a reasonable technique to use when using a mass system. This is especially true when comparing the first-order Runge-Kutta to Central-Differences which requires an expensive matrix inversion.

In the next set of experiments, a comparison between the potential types was made. Again the four different methods for solving for the displacement were used while varying the potential type from gravity to spring (Table 7.3). From these cases, one can clearly see that strong image potentials were needed for obtaining good results with the springs. Since springs are weaker with smaller displacements the closer the boundary came to the true boundary the less influence they had. Even with strong spring potentials the accuracy was still less than the gravity potentials. This was due to the inability of the springs to lock on

Table 7.3. A comparison between the number of iterations and missed points using the direct gravity and the direct spring potentials. (The image potential strength is noted in parenthesis).

	Gravity (10X)		**Spring** (20X)	
	Iterations	Missed points	Iterations	Missed points
Euler Matrix Inversion (SVD)	49	18	93	26
Euler Circulant Matrix (DFT)	37	20	95	26
Central Differences	588	21	777	19
Runge-Kutta First-Order	984	19	773	19
Runge-Kutta Fourth-Order	573	20	774	19
	Gravity (1X)		**Spring** (10X)	
	Iterations	Missed points	Iterations	Missed points
Euler Matrix Inversion (SVD)	92	27	105	42
Euler Circulant Matrix (DFT)	71	27	69	38
Central Differences	600	32	448	49
Runge-Kutta First-Order	775	23	449	50
Runge-Kutta Fourth-Order	598	32	447	49

to the boundary once it came close, instead the internal stiffness took over as the dominant force causing the boundary to move away from its true position (Figure 7.39).

From these two sets of experiments, the following conclusions are made: a mass system is able to give results with the same accuracy as the massless system but it typically took 8 to 26 times as many iterations. Thus, the mass systems were discarded from further use and analysis. In a similar manner, the massless system using a circulant matrix (DFT) typically took fewer iterations than the matrix inversion (SVD). However, as previously noted, circulant matrices were only a computational advantage when doing fewer iterations than degrees-of-freedom.[4] Unfortunately, this was not the case for the 2D images as there were 12 nodes each with two degrees-of-freedom whereas the smallest number of iteration was 37. On the average, 25% more work was done with a circulant matrix than with a matrix inversion, thus it would seem reasonable to use a matrix inverse with small meshes.

Accuracy was only one of two performance measures taken. The second, precision, was conducted using the massless system with a circulant matrix (DFT) and varying

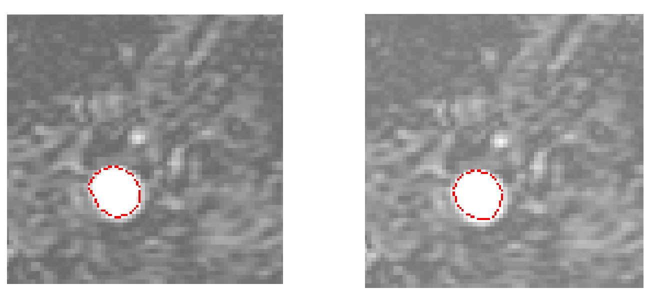


Figure 7.39. Comparison of the deformation process with a DFT solution using a gravity (left) and spring (right) type force.

4. When doing more iterations than degrees-of-freedom, the amount of extra work done is $(A+1)N^2$ where A is the number of iterations beyond the N degrees-of-freedom (i.e., previous iterations).

the starting location and by varying the radius of the initial guess. For these experiments, the number of iterations and accuracy were recorded (Table 7.4). In both test cases, the position and size standard deviation, 3.2 and 2.0, respectively, are small with respect to the average number of missed points, 20.8 and 18.6, respectively. These results demonstrate that the deformation process was fairly precise. That is, the final results are not significantly affect by the initial position nor the initial shape. However, there are always the exceptions such as when the initial position was such that spurious potentials influenced the final results.

A final observation to note was the overall accuracy of the system. Previously, it was noted that the seeded region growing segmentation technique yielded inaccuracy of

Table 7.4. A comparison of the precision versus the mesh's initial position and size using a DFT solution with direct gravity potentials.

position	Iterations	Missed points	Size	Iterations	Missed points
+0/+0	37	20	3	33	18
+1/+0	55	18	4	33	18
+0/+1	65	21	5	31	18
+1/+1	60	23	6	37	20
-1/+0	35	15	7	55	18
+0/-1	95	17	8	25	23
-1/-1	60	21	9	29	17
+0/+2	96	27	10	25	17
+2/+0	35	24	Average (std. dev.)	25.3	18.6 (2.0)
+1/+2*	120	37			
+2/+1*	161	31			
+2/+2*	78	42			
+0/-2	30	23			
-2/+0	53	20			
-1/-2	48	21			
-2/-1*	90	23			
-2/-2*	94	32			
Average (std. dev.)	55.8	20.8 (3.2)			

*Unacceptable results due to the boundary being attracted to spurious potentials.

5% to 6% whereas the deformation process was at best 15%. This was an area of great concern especially since the segmentation was the very first step in the whole process and was suppose to be only a crude approximation to the final results, yet they are better than the final result. However, one must remember that the initial segmentation is able to assume any shape whereas the current deformation is based on polygonal shapes with 12 sides. The final experiment done on the 2D data was to asses the effects of the fineness (the number of sides used to approximate a circle) on the accuracy (Table 7.5).

These results demonstrate that as the fineness increased the accuracy increased (with one exception, were the fineness equaled 33, where there was a slight loss in accuracy). These results were expected since with greater fineness comes a greater ability to assume arbitrary shapes. Unfortunately, the greater fineness comes with a cost, greater computational expense. For each additional 2D point there were two extra degrees-of-freedom. If one assumes that the deformation process is approximately an order N^3 operation, then each additional point caused an eight fold increase in the computational expense (for

Table 7.5. A comparison of the accuracy versus fineness using a DFT solution with direct gravity potentials.

Fineness	Iterations	Missed points
6	64	51
9	50	31
12	71	27
15	39	19
18	45	17
21	46	16
24	40	15
27	50	14
30	37	12
33	37	14
36	55	11

a 3D point there was an 27-fold increase).[5] Thus, although adding more points increased the accuracy, it was overshadowed by the computational expense.

7.3.3 3D Deformations

Using the results of the 2D deformations it was decided that for the 3D deformations that only the massless system would be used. As such, only experiments on the accuracy of using a singular value decomposition (SVD) matrix inverse and an approximation with a circulant matrix and discrete Fourier transforms (DFT) were done. The reason for this was twofold; the 2D results showed that there was not a significant gain in accuracy with a mass system and that a mass system required far more computational expense than a massless system.

For the 3D experiments, both synthetic and real data were used. Unlike the 2D image where it was not practical to use the template selection process, for the 3D images it was used to obtain the initial meshes.

The synthetic data were a 64x64x64 noiseless image of a right cylinder (Figure 7.40). Using a massless system with 15,828 truth points, there were 2869 (18.1%) and 2433 (15.4%) missed points for the SVD and DFT solutions, respectively. (The difference between the two methods was 436 points or a 16.5% difference.) Examining the difference image reviewed that the majority of the missed points were along the boundary and were approximately one voxel thick (Figure 7.41). This error can be contributed using a coarse mesh (12 points defining a circle) and round off error trying to discretize the mesh.

Experiments on a second synthetic image were also performed. In this case a “po-

5. This was true when number of iterations equalled the number of degrees-of-freedom. For these examples, the average number of degrees-of-freedom was 41 with 48 iterations.

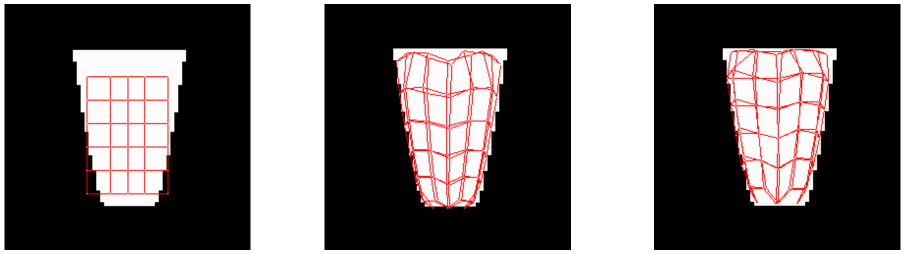


Figure 7.40. Synthetic data showing the data with the original mesh (left) and deformed mesh using a massless SVD (center) and DFT (right) solution. Note that the initial mesh is both inside and outside the cylinder.

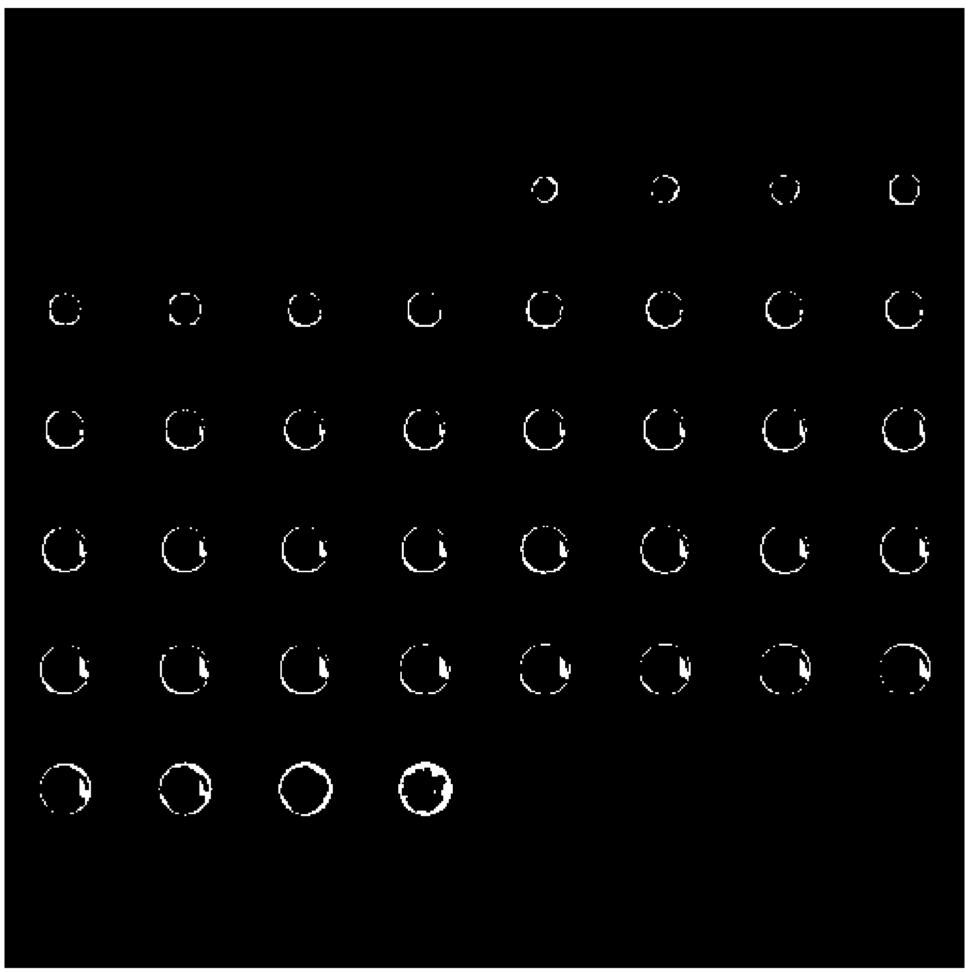


Figure 7.41. Difference image between the synthetic data and the DFT deformed model.

tato" shaped object, with several bulges in it was used (Figure 7.42). Performing just a DFT solution with 15945 truth points there were 2983 (18.7%) missed points. The majority of the error was in the top and bottom portions of the mesh, where the mesh was not able to fully deform to all of the bulges due to using a coarse mesh. After refining the mesh, visually the refined mesh appears to have recovered the shape of the potato better than the unrefined mesh (Figure 7.42). However, the accuracy did not improve; there were 3347 missed points (21.0%). After close examination of the difference images, we concluded that the technique used to convert the mesh into a solid object worked in favor of the unrefined mesh by making large approximations when filling the ends of the mesh. Had we made allowances to close the mesh this might not have been a problem.

A second set of experiments was done on the potato to assess the effects of noise on the deformation process, or more accurately, the edges which were used in the deformation process. For this experiment, four synthetic images of the potato were generated, each having a signal-to-noise ratio of 1.0, 2.0, 4.0, and 8.0. Using the initial refined mesh, the shape of the "potato" was recovered for each of the noisy images (Figure 7.43). Measurement of the error produced surprising results. Normally when noise is present, one expects an increase in the number of errors. However, in this case, the edge detector is able to produce strong edges even in the presence of noise as such the deformation process was unaffected as demonstrated by the consistent number of missed points (Table 7.6).

For both sets of synthetic data, the initial mesh was open ended. These ends were always open since other surfaces could be attached to them. Even though the ends were open, in all cases the deformation process tried to closed the ends of the mesh to form a complete solid.

Experiments on real data consisted of also using the SVD and DFT solutions on a

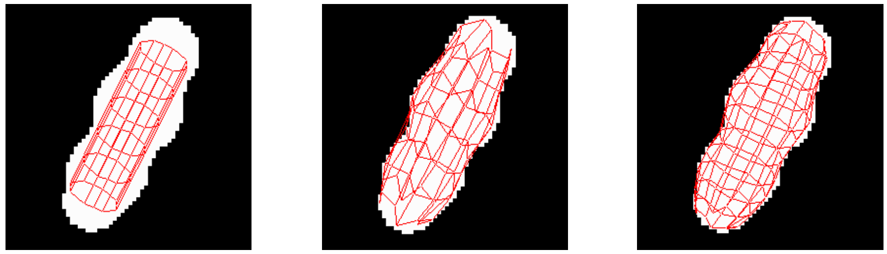
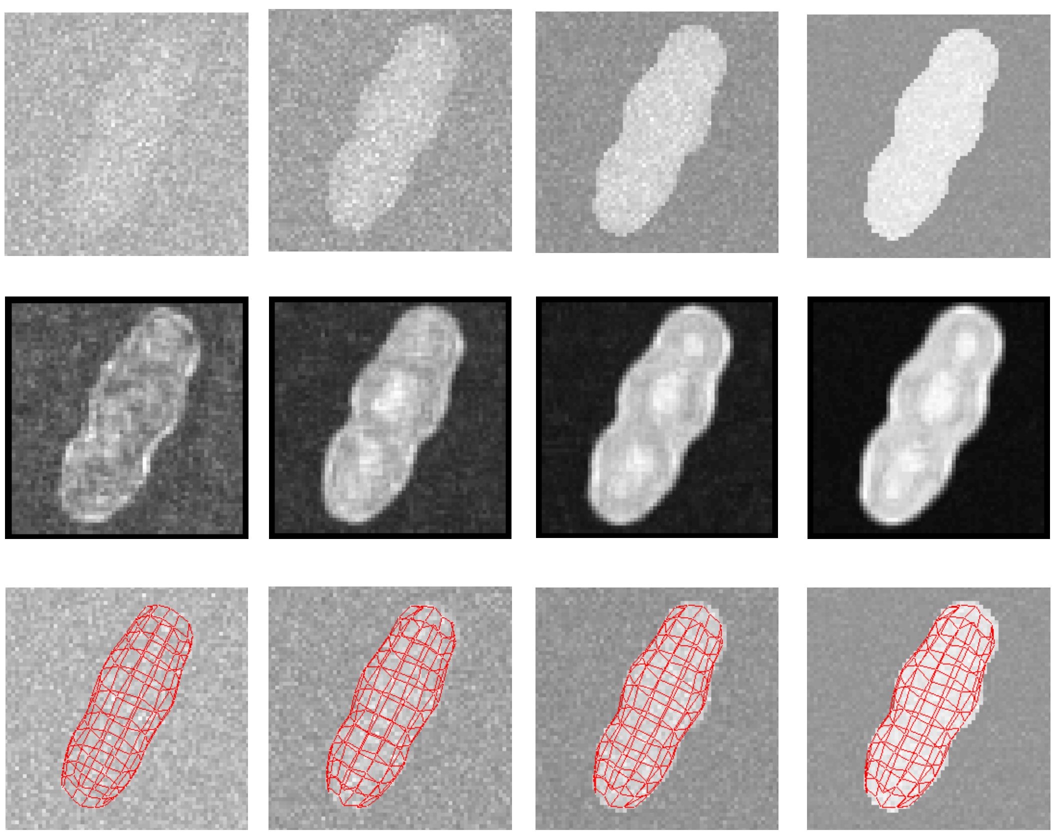


Figure 7.42. Synthetic data showing the original mesh (left), deformed mesh (center), and a deformed refined mesh (left). Both meshes were deformed using a massless DFT solution.



S/N 1 S/N 2 S/N 4 S/N 8

Figure 7.43. Deformation process in the presence of increasing signal to noise (left to right). MIP of the original images (top row), MIP of the response to the 3D edge detector (middle row), MIP of the original image with the refined mesh after deformation (bottom row).

Table 7.6. A comparison of the shape recovery using a DFT solution with direct gravity potentials as a function of noise.

S/N	Iterations	Missed points	Percent
1	344	3237	20.30
2	222	3212	20.14
4	138	3267	20.49
8	166	3266	20.48
Binary Image	113	3347	20.99

32x32x48 3D MRA of a vertebral artery. Again the template selection process was used with no user interaction except for selecting the initial seed for the region growing. The results of the deformation process are shown in Figure 7.44.

The quantitative results of the deformation process were disappointing. For instance, out of a possible 1462 “truth” points the SVD and DFT solutions had 730 (49.9%) and 731 (50.0%) missed points respectively. However, upon close examination these results are misleading when viewing the actual missed points (Figure 7.45). This image illustrates characteristics typical of many objects. First, the vessel being recovered is only a few voxels thick (diameter of approximately five voxels). Second, the majority of the missed points are one voxel thick except where the vessel is in a tight bend. The small vessel diameter was not uncommon, but trying to discretize the mesh to fit this small diameter accurately was difficult at best due to truncation error. Because of truncation error the quantitative results for this example are not a valid comparison. Had the vessel been larger as in the case of the synthetic data, the measure would have been better.

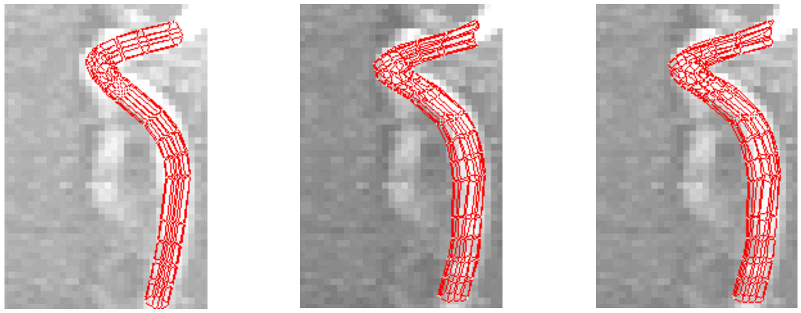


Figure 7.44. The initial mesh (left) and the results of a massless SVD (center) and DFT (right) deformation process after 183 and 161 iterations, respectively.

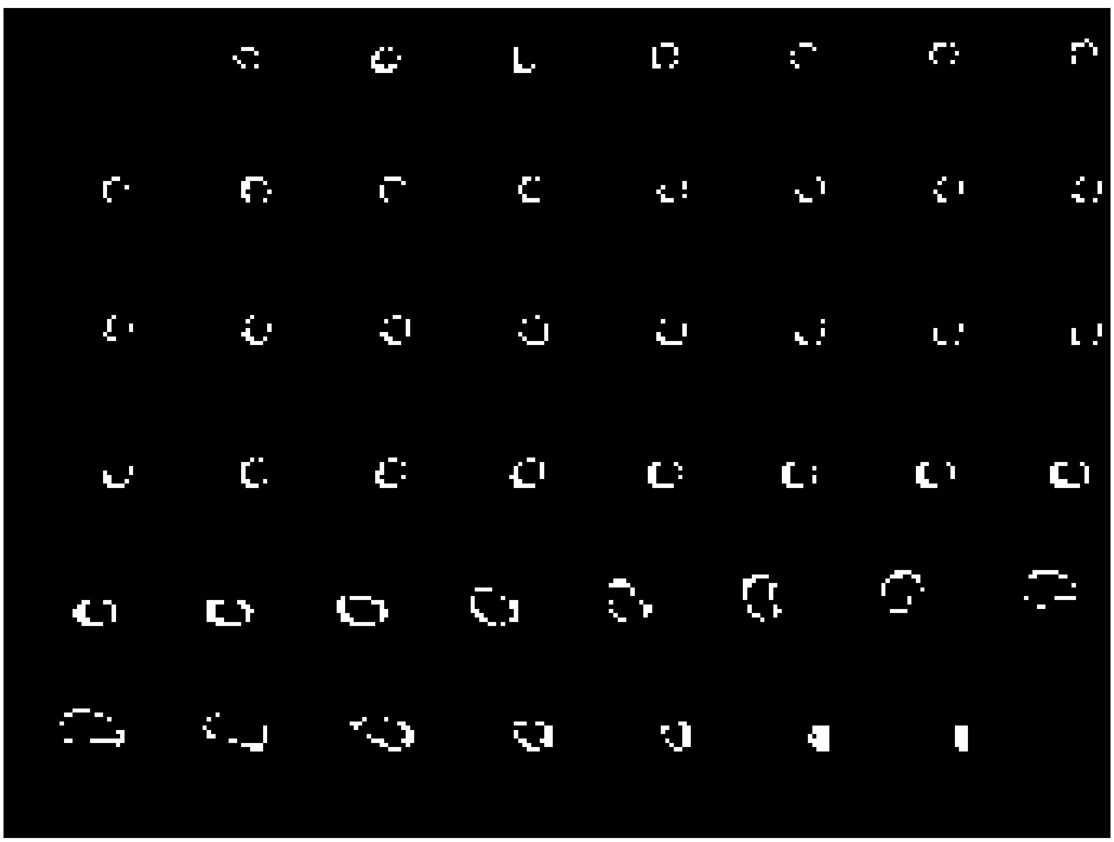


Figure 7.45. Difference image between the truth and deformed model on a slice by slice basis for the image and model shown in Figure 7.44

7.3.4 Multiple 3D Deformations

In the last phase, we used the template selection process to select the surfaces needed and then allowed those surfaces to deform. We applied this to 128^3 volume CT image of a solid model of a bifurcating artery (Figure 7.47). For this data set, a single vessel bifurcation and three vessel bodies were used. Combined, this resulted in a total of 12 different surfaces, six being used for the bifurcation and two each for the three bodies. These 12 surfaces were combined into a single finite element mesh containing 388 nodes. To deform the mesh we chose to use the SVD solution. If one recalls, when less iterations than degrees-of-freedom are done using the DFT solution is faster, which is the case here, since there are almost 1200 degrees-of-freedom. However, approximating the damping matrix with a circulant matrix was not practical when using multiple surfaces since some points had anywhere from three to six neighbors. Thus, phantom nodes would need to be added to some nodes while reduced on others. Also, we were not able to find a numbering scheme that would initially produce a matrix that was almost circulant. Our final result is the complete deformation and recovery of the bifurcation (Figure 7.46).

7.3.5 Residual Forces

The analysis previously presented shows that “in general” the deformation is accurate and precise. However, when recovering the shape of objects, it often useful to the provide the user the some measure of the performance of the system. Since the previous analysis relies on having truth data, it is not a practical analysis technique. Instead, we have developed a technique that uses the resulting potential force at each node. If the boundary has been found accurately then all of the potential forces should be balanced with the exception of a small amount of force used to overcome the internal forces due to

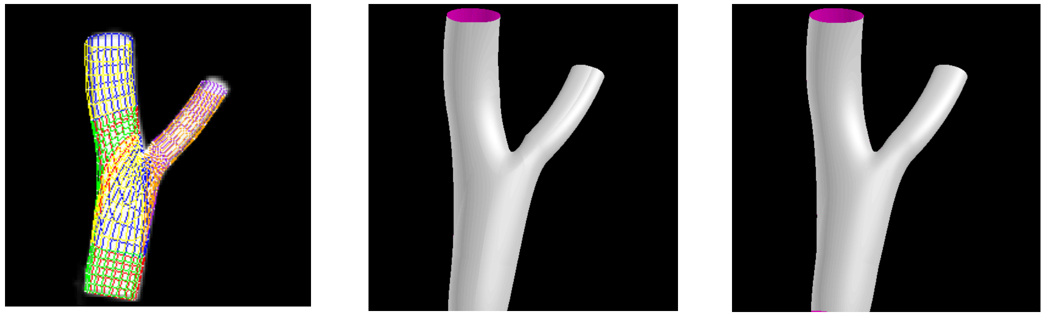


Figure 7.46. Results of the deformation process on the mesh (left) and a rendered image of the mesh as B-spline surfaces before (center) and after (right) enforcement of the tangent plant continuity.

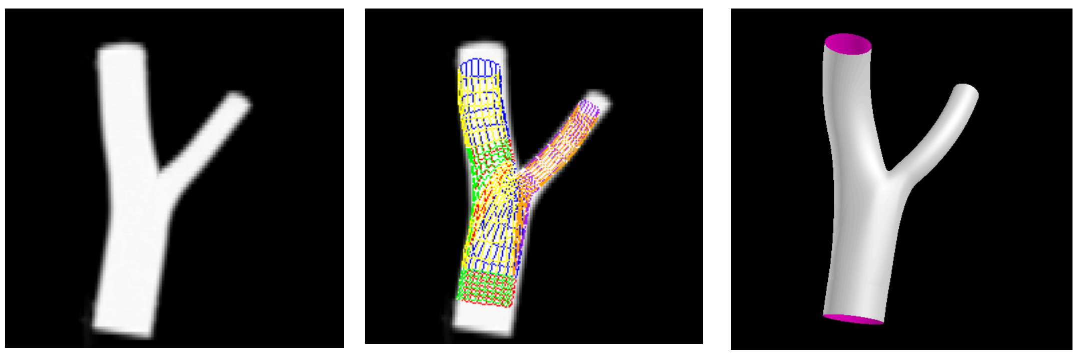


Figure 7.47. CT image of a bifurcation model (left), the initial template seeding (center), and a rendered version (right).

the spring attachments between the nodes. When these external forces are not zero then the surface has a potential to deform and is an area of questionable accuracy. This information is most easily shown in the form of a texture mapping the on the surface (Figure 7.48). In this example, there is one area that has a large amount of residual force on it. This instability is also observable in the comparison between the "truth" image and the deformed model from Figure 7.45. In this figure, slices 29 through 35 contain a region of "missed points" that are two voxels wide. Another example of the residual forces is shown in Figure 7.49. In this example, the deformation process was stopped before completion to

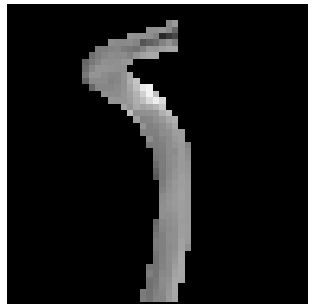


Figure 7.48. A texture mapping of the residual forces from the deformed mesh in Figure 7.44. The bright areas show large residual forces.

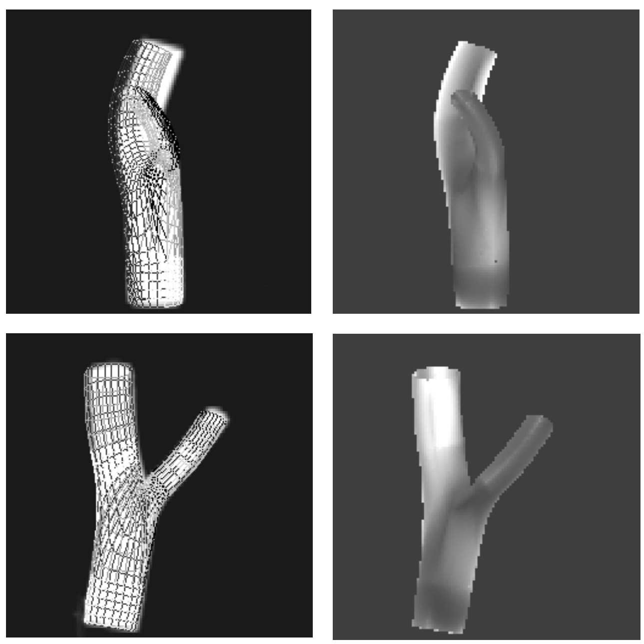


Figure 7.49. A side (top row) and front view (bottom row) of a deformed mesh (left column) and a texture mapping of the residual forces (right column). Note the large residual forces due the mesh misalignment for the large vessel above the bifurcation.

observe the residual forces, which are clearly seen in the texture mapping images due to the mesh miss alignment.

7.4 Summary

In this chapter we have demonstrated the complete shape recovery process, topology identification, model seeding and the deformation process. Each of these processes can be an end into themselves, that is, each process can supply information that can be used, not only for the next process, but for other purposes. For instance, the model seeding can be used without the topology identification to generate a vascular network of any topology if supplied with a series of centerlines and radii. Similarly, one may wish to visualize the original data without the background, thus only the segmentation needs to be performed. However, the greatest benefit is to use all of the processes together.

We have also included a preliminary analysis of our system using a known standard. Included in this analysis is not only analysis of the accuracy and precision of the system but of also an analysis of the relative speeds using different approximation schemes such as the use of a circulant matrix approximation and various numerical integration schemes. More importantly, we have provided a mechanism in which it is possible for the user to view residual forces during the deformation process on all types of shape recovery. This is information can be vital in determining whether the results are stable enough to be used.

CHAPTER 8

CONCLUSION

Shape recovery is an important part of computer vision and computer graphics. Recovering individual voxels from image data is important, but provides limited utility except for viewing purposes. Being able to recover the shape and represent it in a parametric form provides greater utility. A parametric representation can approximate the shape and still have the flexibility for other processing. Other researchers have shown it is possible to recover shape using nonparametric representations, but these representations fall short because of their complexity, data storage and/or postprocessing requirements.

The research conducted developed application specific deformable models using a B-spline representation, more specifically, NURBS, for shape recovery in 2D and 3D images. The research explored use of topologically different template models and the ability to combine different template models to recover the shape without a loss in smoothness. For this, we developed two distinct topologies, a generalized cylinder and a bifurcation. Further, we have introduced a technique to maintain the tangent plane continuity using only tensor-product NURBS with a constrained optimization process.

The key part of using topologically different template models was the ability to automatically identify the object's topology and then select the proper template. For this, we have developed a three-stage process, segmentation, thinning, and graph labeling. This process identifies the object's topology while supplying basic geometric information such as the radii and axis length for cylindrical objects which was our primary interest. In addition, we have developed new set of deformation operators to deform the surfaces using image data. Finally, we have introduced the notion of using phantom nodes during the deformation process. These phantom nodes have allowed us to use Fourier transforms in solving for the displacements in several methods, thus gaining a substantial computational saving in many cases.

Our major application of this research has been with medical images, more specifically to 2D and 3D angiograms of the human cerebral vascular system. Our research has shown that we are able to recover the shape of a vascular system with a reasonable degree of accuracy. However, the vascular system is complex enough that in order to obtain a sufficient level of detail in the deformation process a large body of compute resources must be available. From a clinical viewpoint, physicians who are seeking to obtain information from the images have found our results to be of great interest. We have succeeded in developing a technique that not only recovers the general shape of the vascular system, i.e., typically approximated by circular cross-section, but also recovers the true shape of the vessel lumen wall using a deforming parametric representation. This may be of significant clinical importance because the disease process nearly always results in noncircular vessel cross-sections. Further, this shape information can now be used to study the hydrodynamic characteristics of the blood using the true vessel shape information which has not been possible before. Still further, another application is to use the shape information to obtain geometric characteristics such as the surface curvature which may also yield information about the disease process.

In conclusion, this research provides the theory so researchers and physicians can recover the shape of the major anatomical structures in the vascular and other organ systems. This recovery and display process can act as an aid for visualization, surgical planning, and theoretical modeling. Furthermore, this approach lays the basic foundation for other shape recovery processes using topologically different deformable B-spline surfaces.

REFERENCES

[1] E. Artzy, G. Frieder, and G.T. Herman, "The Theory, Design, Implementation, and Evaluation of a Three-Dimensional Surface Detection Algorithm," *Computer Graphics and Image Processing*, vol. 15, no. 1, pp. 1-24, January 1981.

[2] D.H. Ballard and C. M. Brown. *Computer Vision*. Englewood Cliffs, NJ: Prentice-Hall, 1982.

[3] A.H. Barr, "Global and Local Deformations of Solid Primitives," *Computer Graphics*, vol. 18, no. 3, pp. 21-29, July 1985.

[4] R.H. Bartels, J.C. Beatty, and B.A. Barsky, *An Introduction to Splines for use in Computer Graphics & Geometric Modeling*. Los Altos, CA: Morgan-Kaufmann, 1987.

[5] E.M.L. Beale, *Introduction to Optimization.* New York, NY: John Wiley and Sons, 1988.

[6] G.M. Besson, "Three-Dimensional Vascular Reconstruction Using an Image Intensifier-Based Tomography System," Ph.D. Dissertation, Dept. of Electrical Engineering, The University of Utah, Salt Lake City, Utah, 1991.

[7] P. Bezier, "Mathematical and practical possibilities of UNISURF," in *Computer Aided Geometric Design*, R. Barhill and R. Riesenfeld, editors, San Diego, CA: Academic Press, 1974.

[8] J.F. Blinn, "A Generalization of Algebraic Surface Drawing," *ACM Transactions of Graphics*, vol. 1, no. 3, pp. 235-256, July 1982.

[9] J. Bloomenthal, "Modeling the Mighty Maple*," Computer Graphics*, vol. 19, no. 3, pp. 305-311, July 1985.

[10] H. Blum., "Biological shape and visual science Part I," *Journal of Theoretical Biology*, vol. 38, no. 2, pp. 205-287, Febuary 1973.

[11] R.M. Bolle and B.C. Vemuri, "On Three-Dimensional Surface Reconstruction Methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 1, pp. 1-13, January 1991.

[12] G. Borgefors, "Distance Transformations in Arbitrary Dimensions," *Computer Vision, Graphics, and Image Processing*, vol. 27, no. 3, pp. 312-345, September 1984.

[13] C. Breau, A. Shirazi-Adl, and J. de Guise., "Reconstruction of a Human Ligamentous Lumbar Spine Using CT Images - A Three-Dimensional Finite Element Mesh Generation," *Annals of Biomedical Engineering*, vol. 19, no. 3, pp. 291-302, October 1991.

[14] B.D. Bunday, *Basic Optimisation Methods.* London: Edward Arnold Ltd. 1984.

[15] G. Celniker and D. Gossard, "Deformable Curve and Surface Finite Elements for Free-Form Shape Design," C*omputer Graphics*, vol. 25, no. 4, pp. 257-266, July 1991.

[16] H.N. Christiansen and T.W. Sederberg, "Conversion of Complex Contour Line Definitions into Polygonal Element Mosaics," *Computer Graphics*, vol. 12, no. 3, pp. 187-192, August 1978.

[17] J. Clark, "The Geometry Engine: A VLSI Geometry System for Graphics," C*omputer Graphics*, vol. 16, no. 3, pp. 127-133, August 1978.

[18] H.E. Cline, W.E. Lorensen, S. Ludke, C.R. Crawford, and D.C. Teeter, "Two algorithms for the three-dimensional reconstruction of tomograms," *Medical Physics*, vol. 15, no. 3, pp. 320-327, May 1988.

[19] H.E. Cline, W.E. Lorensen, R.J. Herfkens, G.A. Johnson, and G.H. Glover, "Vascular Morphology by Three-Dimensional Magnetic Resonance Imaging," *Magnetic Resonance Imaging*, vol. 7, no. 1, pp. 45-54, January 1989.

[20] E. Cohen, T. Lyche, and R.F. Riesenfeld, "Discrete B-splines and subdivision techniques in computer aided design and computer graphics," *Computer Graphics and Image Processing*, vol. 14, no. 2, pp. 87-111, October 1980.

[21] E. Cohen and L.L. Schumaker, "Rates of Convergence of Control Polygons," *Computer Aided Geometric Design*, vol. 2, no. 1-3, pp. 229-235, September 1985.

[22] I. Cohen, L.D. Cohen, N. Ayache, "Introducing Deformable Surfaces to Segment 3D Images and Infer Differential Structure*s,"* Rapports de Recherche No. 1403, Institut National de Recherche en Informatique et en Automatique, Sophia-Antipolis, France, Mars 1991.

[23] L.D. Cohen, "Note - On Active Contour Models and Balloons," *CVGIP: Image Understanding*, vol. 53, no. 2, pp. 211-218, March 1991.

[24] S.A. Coons, "Surfaces for Computer Aided Design," MIT Technical Report, Massachusetts Institute of Technology, Boston, MA, 1964.

[25] A.F. D'souza and V.K. Garg, *Advanced Dynamics - Modeling and Analysis*. Englewood Cliffs, NJ: Prentice-Hall, 1984.

[26] C. de Boor, *A Piratical Guide to Splines*. Berlin, Germany: Springer-Verlag, 1978.

[27] P. de Casteljau, "Outillages methodes calcul," Technical Report, A. Citroen, Paris, France, 1959.

[28] T.D. DeRose, "Necessary and sufficient conditions for tangent plane continuity of Bezier surfaces," C*omputer Aided Geometric Design,* vol. 7, no. 1-4, pp. 165-179, June 1990.

[29] C.L. Dumoulin, H.E. Cline, S.P. Souza, et al., "Three-dimensional time-of-flight magnetic resonance angiography using spin saturation," *Magnetic Resonance in Medicine,* vol. 11, no. 1, pp. 35-46, July 1989.

[30] H.A. Driskill, "COMA: Constrained Optimization in Modeling and Animation," Ph.D. Dissertation, Dept. of Computer Science, The University of Utah, Salt Lake City, Utah, 1995.

[31] R.A. Drebin, L. Carpenter, P. Hanrahan, "Volume Rendering," C*omputer Graphics*, 22, no. 4, pp. 65-74, August 1988.

[32] P.H. Eichel and E.J. Delp, "Sequential Edge Linking," in *Proc. of the 22nd Allerton Conf. on Control and Computers*, Monticello, IL, October 1984, pp. 782-791.

[33] M. Ellens, "Complex Models in a Physically-Based Environment," Dissertation Proposal, Dept. of Computer Science, The University of Utah, Salt Lake City, Utah, 1992.

[34] H. Elliot and L. Srinivasan, "An Application of Dynamic Programming to Sequential Boundary Estimation," *Computer Graphics and Image Processing*, vol. 17, no. 4, pp. 291-314, December 1981.

[35] Engineering Geometry Systems, *Alpha_1 User's Manual*, 1991.

[36] G. Farin, *Curves and Surfaces for Computer Aided Geometric Design a Practical Guide,* Second Edition. San Diego, CA: Academic Press, 1990.

[37] T.A. Foley, "Scattered data interpolation and approximation with error bounds," *Computer Aided Geometric Design*, vol. 3, no. 3, pp. 163-1777, November 1986.

[38] B.M. Fowler, "Geometric Manipulation of Tensor Product Surfaces," in *Proc. 1992 Symposium on Interactive 3D Graphics,* Cambridge, MA, March 1992, pp.101-108.

[39] B.M. Fowler and R.H. Bartels, "Constraint Based Curve Manipulation," *IEEE Computer Graphics Applications*, vol. 13, no. 5, pp. 43-49, September 1993.

[40] H. Fuchs, Z.M. Kedem, and S.P. Uselton, "Optimal Surface Reconstruction from Planar Contours," *Communications of the ACM*, vol. 20, no. 10, pp. 693-702, October 1977.

[41] General Electric Medical Systems, Milwaukee, Wisconsin.

[42] G. Gerig, et al., "Symbolic description of 3-D structures applied to cerebral vessel tree obtained from MR Angiography volume data," in *Information Processing in Medical Imaging 13th International Conference,* Flagstaff, AZ, June 1993, pp. 94-111.

[43] W. J. Gordon, "Spline-Blended Surface Interpolation through Curved Networks," *Journal of Mathematics and Mechanics*, vol. 18, no. 10, pp. 931-952, November 1969.

[44] K.J. Hafford and K. Preston Jr., "Three-Dimensional Skeletonization of Elongated Solids," *Computer Vision, Graphics, and Image Processing*, vol. 27, no. 1, pp. 79-91, January 1984.

[45] P.B. Hefferman and R.A. Robb, "A New Method for Shading Surface Display in Biological and Medical Images," *IEEE Transactions on Medical Imaging,* vol. 4, no. 1, pp. 26-38, March 1985.

[46] G.T. Herman and H.K. Lu, "Three-Dimensional Display of Human Organs from Computed Tomograms," *Computer Graphics and Image Processing*, vol. 9, no. 1, pp. 1-21, January 1979.

[47] K.H. Hohne, M. Bomans, A. Pommert, M. Riemer, C. Schiers, U. Tiede, and G. Wiebecke, "3D Visualization of tomographic volume data using the generalized voxel model," *The Visual Computer*, vol. 6, no. 1, pp. 28-36, January 1990.

[48] T.J.R. Hughes, *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Englewood Cliffs, NJ: Prentice Hall, 1987.

[49] A.C. Kak and M. Slaney. *Principles of Computerized Tomographic Imaging*. New York, NY: IEEE Press, 1988.

[50] E. Keppel, "Approximating Complex Surfaces by Triangulation of Contour Lines," *IBM Journal of Research and Development*, vol. 19, no. 1, pp. 2-11, January 1975.

[51] M. Levoy, "Display of Surfaces from Volume Data," *IEEE Computer Graphics and Applications*, vol. 8, no. 3, pp. 29-37, May 1988.

[52] W.C. Lin, C.C. Liang, and C.T. Chen, "Dynamic Elastic Interpolation for 3-D medical Image Reconstruction from Serial Cross Sections," *IEEE Transactions on Medical Imaging*, vol. 7, no. 3, pp. 225-232, Sept. 1988.

[53] W.C. Lin, S.Y. Chen, and C.T. Chen, "A New Surface Interpolation Technique for Reconstructing 3D Objects from Serial Cross-Sections," *Computer Vision, Graphics, and Image Processing*, vol. 48, no. 1, pp. 124-143, January 1989.

[54] S. Lobregt, P.W. Verbeek, and F.C.A. Groen, "Three-Dimensional Skeletonization: Principle and Algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 2, no. 1, pp. 75-77, January 1980.

[55] D.L. Logan, *A First Course in the Finite Element Method*. Second Edition, NewYork, NY: PWS-Kent Publishing Co, 1992.

[56] C. Loop and T. DeRose, "Generalized B-spline Surfaces of Arbitrary Topology," *Computer Graphics,* vol. 24, no. 4, pp. 347-356, August 1990.

[57] W.E. Lorensen and H.E. Cline, "Marching Cubes: A High Resolution 3D Surface Construction Algorithm*," Computer Graphics,* vol. 21, no. 4, pp. 163-169, July 1987.

[58] T. Lyche, E. Cohen, and K. Morken, "Knot Line Refinement Algorithms for Tensor Product B-Spline Surfaces," *Computer Aided Geometric Design*, vol. 2, no. 1-3, pp. 133-139, September 1985.

[59] T. Lyche and K. Morken, "Making the Oslo Algorithm More Efficient," *SIAM Journal on Numerical Analysis*, 23, no. 3, pp. 663-675, June 1986.

[60] T. Lyche and K. Morken, "Knot Removal for Parametric B-Splines Curves and Surfaces," *Computer Aided Geometric Design*, vol. 4, no. 3, pp. 217-230, November 1987.

[61] T. McInerney and D. Terzopoulos, "A Dynamic Finite Element Surface Model for Segmentation and Tracking in Multidimensional Medical Images with Application to Cardiac 4D Image Analysis," *Journal of Computerized Medical Imaging and Graphics*, vol. 19, no. 1, pp. 69-83, January 1994.

[62] J.V. Miller, D.E. Breen, W.E. Lorensen, R.M. O'Bara, M.J. Wonzy, "Geometrically Deformed Models: A Method for Extracting Closed Geometric Models From Volume Data," *Computer Graphics*, vol. 25, no. 4, pp. 217-226, July 1991.

[63] R.R. Mercer, G.M. McCauley, and S. Anjilvel, "Approximation of Surfaces in Quantitative 3-D Reconstructions*," IEEE Transactions on Biomedical Engineering*, vol. 37, no. 12, pp. 1136-1146, December 1990.

[64] H.P. Moreton and C.H. Sequin, "Functional Optimization for Fair Surface Design," *Computer Graphics*, vol. 26, no. 2, pp. 167-176, July 1992.

[65] K. Morken, "Products of Splines as Linear Combinations of B-Splines," in *Appoximation Theory VI,* San Diego, CA: Accidemic Press, 1989.

[66] S. Muraki, "Volumetric Shape Description of Range Data using 'Blobby Model'," *Computer Graphics*, vol. 25, no. 4, pp. 227-235, July 1991.

[67] R. Nevatia, "Structured Descriptions of Complex Curved Objects for Recognition and Visual Memory," AIM-250, Stanford AI Lab, Standford University, CA October 1974.

[68] R. Nevatia and K.R. Babu, "Linear Feature Extraction and Description," *Computer Graphics and Image Processing*, vol. 13, no. 3, pp. 257-269, March 1980.

[69] J.S. Oirolo, S.J. Bresina, M.W. Vannier, D.G. Gayou, J.L. Cox, and M.K. Pasque, "Three Dimensional Solid Modeling of the Canine Biventricular Unit," in *Proc. of the Annual Inter. Conf. of the IEEE Engineering in Medicine & Biology* Society, Philidelphia, PA, November 1990, pp389-391.

[70] A.G. Osborn, *Introduction to Cerebral Angiography.* New York, NY: Harper and Row, 1980.

[71] D.L. Parker and D.D. Blatter, "Multiple Thin Slab Magnetic Resonance Angiography," *Neuroimaging Clinics of North America*, vol. 2, no. 4, pp. 677-692, November 1992.

[72] T. Pavlidis, *Structural Pattern Recognition*. Berlin: Springer-Verlag, 1977.

[73] T. Pavlidis, "An Asynchronous Thinning Algorithm," *Computer Graphics and Image Processing,* vol. 20, no. 2, pp. 133-125, October 1982.

[74] A. Pentland, "Automatic Extraction of Deformable Part Models," *International Journal of Computer Vision*, vol. 4, no. 2, pp. 107-126, March 1990.

[75] A. Pentland and S. Sclaroff, "Closed-Form Solutions for Physically Based Shape Modeling and Recognition*," IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 13, no. 7, pp. 715-729, July 1991.

[76] A. Pentland and B. Horowitz, "Recovery of nonrigid Motion and Structure*," IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 13, no. 7, pp. 730-742, July 1991.

[77] R.F. Riesenfeld, "Application of b-spline approximation to geometric problems of computer-aided design," Ph.D. Disseration, Syracuse University, Syracuse, NY, 1973.

[78] A. Rosenfeld, "A characterization of parallel thinning algorithms," *Information Control,* vol. 29, no. 3, pp. 286-291, November 1975.

[79] A. Rosenfeld and A.C. Kak, *Digital Picture Processing*. San Diego: Academic Press, 1982.

[80] A.R. Sanderson, E. Cohen, W. Davis., and D.L. Parker. Complex Vascular Phantoms for MR and X-ray Angiography", to be submitted to *Medical Physics*.

[81] J. Sequeira and F. Pinson, "Matching Free-Form Primitives with 3D Medical Data to Represent Organs and Anatomical Structures," in *NATO ASI Series, Vol F 60, 3D Imaging in Medicine,* Eds. K.H. Hohne, et al., Belin: Springer-Verlag, 1990.

[82] F. Solina and R. Bajcsy, "Recovery of Parametric Models from Range Images: The Case for Superquadrics with Global Deformations*," IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 13, no. 7, pp. 131-147, July 1991.

[83] G. Strang, *Introduction To Applied Mathematics*. New York, NY: Wellesley-Cambridge Press, 1986.

[84] W.W. Stallings, "Recognition of Printed Chinese Characters by Automatic Pattern Analysis," *Computer Graphics and Image Processing,* vol. 1, no. 1, pp. 47-65, April 1972.

[85] A. Sunguroff and D. Greenberg, "Computer Generated Images for Medical Applications," *Computer Graphics,* vol. 12, no. 3, pp. 196-202, July 1978.

[86] K. Tanne, J. Miyasaka, Y. Yamagata, R. Sachdeva, S. Tsutsumi, M. Sakuda, "Three-Dimensional model of the Human Craniofacial Skeleton: Method and Preliminary Results using Finite Element Analysis," *Journal of Biomedical Engineering*, vol. 10, no. 3, pp. 246-252, May 1988.

[87] D. Terzopoulos and K. Fleischer, "Deformable Models," *The Visual Computer*, vol. 4, no. 6, pp. 306-331, December 1988.

[88] D. Terzopoulos, A. Witkin, and M. Kass, "Constraints on Deformable Models: Recovering 3D Shape and Nonridgid Motion*," Artificial Intelligence*, vol. 36, no. 1, pp. 91-123, January 1987.

[89] D. Terzopoulos and D. Metaxas, "Dynamic 3D Models with Local and Global Deformations: Deformable Superquadrics," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, no. 2, pp. 793-714, Febuary 1990.

[90] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer, "Elastically Deformable Models," *Computer Graphics*, vol. 21, no. 4, pp. 205-214, July 1987.

[91] D. Terzopoulos, A. Witkin, and M. Kass, "Symmetry - Seeking Models and For 3D Object Reconstruction*," International Journal Of Computer Vision*, vol. 1, no. 3, pp. 211-221, October 1987.

[92] D. Terzopoulos, A. Witkin, and M. Kass, "Constraints on Deformable Models: Recovering 3D Shape and Nonrigid Motion," *Artificial Intelligence*, 35, no. 1, pp. 91-123, August 1988.

[93] J. ThingVold, "Elastic and Plastics Surfaces for Modeling and Animation," Masters Thesis, Dept. of Computer Science, The University of Utah, Salt Lake City, Utah, 1990.

[94] J. Toriwaki, et al., "Topological Properties and Topology-Preserving Transforms of a Three-Dimensional Binary Picture," in *IEEE Proc. 6th Inter. Conference on Pattern Recognition*, Munich, Germany, 1982, p414-419.

[95] Y.F. Tsao and K.S. Fu, "A Parallel Thinning Algorithm for 3-D Pictures," *Computer Graphics and Image Processing*, vol. 17, no. 4, pp. 315-331, December 1981.

[96] P.A. van den Elsen, "Multimodality Matching of Brain Images," Ph. D. Dissertation, Utrecht University, Utrecht, The Netherlands, 1993.

[97] D. Vandermeulen, P. Suetens, J. Gybels, and A. Oosterlinck, "A new software package for the microcomputer based BRW stereotactic system: integrated stereoscopic views of CT data and angiograms," in *Proc. SPIE Vol 593, Medical Image Processing*, San Diego, CA, Febuary 1985, p103-114

[98] A. Witkin, K. Fleischer, and A. Barr, "Energy Constraints on Parametrized Models," *Computer Graphics*, vol. 21, no. 4, pp. 225-229, July 1987.

[99] S.C. Wu, J.F. Abel, and D. P. Greenberg, "An interactive computer graphics approach to surface representation," *Communications of ACM*, vol. 20, no. 10, pp. 703-712, October 1977.

[100] A.L. Yuille, D.S. Cohen and P.W. Hallinan, "Feature extraction from faces using deformable templates," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Diego, CA, June 1989, pp.104-109.