THE UNIVERSITY OF HULL

ACCURATE GEOMETRY RECONSTRUCTION OF VASCULAR STRUCTURES USING IMPLICIT SPLINES

being a Thesis submitted for the Degree of Doctor of Philosophy

in the University of Hull

by

Qingqi Hong
B.Sc, M.Sc Computer Science

April 2012

To my parents

Abstract

3-D visualization of blood vessel from standard medical datasets (e.g. CT or MRI) play an important role in many clinical situations, including the diagnosis of vessel stenosis, virtual angioscopy, vascular surgery planning and computer aided vascular surgery. However, unlike other human organs, the vasculature system is a very complex network of vessel, which makes it a very challenging task to perform its 3-D visualization. Conventional techniques of medical volume data visualization are in general not well-suited for the above-mentioned tasks. This problem can be solved by reconstructing vascular geometry. Although various methods have been proposed for reconstructing vascular structures, most of these approaches are model-based, and are usually too ideal to correctly represent the actual variation presented by the cross-sections of a vascular structure. In addition, the underlying shape is usually expressed as polygonal meshes or in parametric forms, which is very inconvenient for implementing ramification of branching. As a result, the reconstructed geometries are not suitable for computer aided diagnosis and computer guided minimally invasive vascular surgery.

In this research, we develop a set of techniques associated with the geometry reconstruction of vasculatures, including segmentation, modelling, reconstruction, exploration and rendering of vascular structures. The reconstructed geometry can not only help to greatly enhance the visual quality of 3-D vascular structures, but

also provide an actual geometric representation of vasculatures, which can provide various benefits.

The key findings of this research are as follows:

1. A localized hybrid level-set method of segmentation has been developed to extract the vascular structures from 3-D medical datasets.
2. A skeleton-based implicit modelling technique has been proposed and applied to the reconstruction of vasculatures, which can achieve an accurate geometric reconstruction of the vascular structures as implicit surfaces in an analytical form.
3. An accelerating technique using modern GPU (Graphics Processing Unit) is devised and applied to rendering the implicitly represented vasculatures.
4. The implicitly modelled vasculature is investigated for the application of virtual angioscopy.

Acknowledgements

First and foremost, I do deeply appreciate my supervisor, Dr. Qingde Li. I could not have wished for a better supervisor and mentor both in academy and life. This thesis would not have been possible without his insightful guidance, constant support, and continuous encouragement.

My thanks are also due to the members of my research committee, Dr. Chandra Kambhampati and Dr. Leonardo Bottaci, for their advice on each stage of my research.

I am thankful to Dr. Jie Tian and his group in Institute of Automation, Chinese Academy of Sciences, for supporting experimental data for this research. Special thanks to Dr. Yan Zhang, Dr. Fangzhong Su, and Mr. James Ward for their fruitful discussions on this research. My thanks are also due to all the members of the SimVis lab. I am especially grateful to Mr. Mike Bielby and Mr. Adam Hird for maintaining my desktop PC.

I would like to acknowledge the University of Hull, Xiamen University, and China Scholarship Council (CSC) for giving me this opportunity and financial support during my study.

Last, but certainly not least, I am thankful beyond words to my parents and family for their endless support and encouragement.

Declaration

Parts of this thesis are published as research papers with Dr. Qingde Li and Dr. Jie Tian in the following sources:

- *Qingqi Hong, Qingde Li, and Jie Tian*. Implicit reconstruction of vasculatures using bivariate piecewise algebraic splines. *IEEE Transactions on Medical Imaging*, vol. 31, no. 3, pp. 543-553, 2012.
- *Qingqi Hong, Qingde Li, and Jie Tian*. Virtual Angioscopy based on implicit vasculatures. *Lecture Note in Computer Science (LNCS)* 6785, pp. 592-603, 2011.
- *Qingqi Hong, Qingde Li, and Jie Tian*. Local Hybrid Level-set Method for MRA Image Segmentation. *In the Proc. of the 10th IEEE International Conference Computer and Information Technology (CIT-2010)*, pp. 1397 - 1402, 2010.

Abbreviations

AABB	Axis-Aligned Bounding Box
CADS	Computer-Aided Diagnosis and Surgery
CESN	Constrained Elastic Surface Nets
CFD	Computational Fluid Dynamics
CS	Convolution Surfaces
CT	Computed Tomography
CTA	Computed Tomography Angiography
CUDA	Computed Unified Device Architecture
DICOM	Digital Imaging and Communications in Medicine
DSA	Digital Subtraction Angiography
GPU	Graphics Processing Unit
LBF	Local Binary Fitting
MC	Marching Cubes
MIP	Maximum Intensity Projection
MPUI	Multi-level Partition of Unity Implicit
MR	Magnetic Resonance
MRA	Magnetic Resonance Angiography
MRI	Magnetic Resonance Imaging
PDE	Partial Differential Equation
PET	Positron Emission Tomography
PSP	Partial Shape Preserving
SDF	Signed Distance Function
SIS	Skeleton-based Implicit Splines
SPMD	Single-Program Multiple-Data
SPECT	Single Photon Emission Computer Tomography

Table of Contents

List of Figures and Tables

Chapter 1
Introduction

1. 1 Computer-Aided Vascular Diagnosis and Surgery

Over the last two decades or so, the technique of Computer-Aided Diagnosis and Surgery (CADS) has revolutionized the way of disease diagnosis and surgery performance. It enables safer, more accurate diagnosis, and less invasive surgical intervention than conventional procedures, with the additional benefits of more precise interventions and faster patient recovery **(Peters, 2000)**. CADS is a set of computer-based techniques and technologies developed specifically for disease diagnosis, pre-surgical planning, and guiding or performing surgical interventions **(Taylor and Joskowicz, 2002)**. In a CADS system, the acquisition of the three-dimensional (3-D) images of the patient from medical imaging devices is often the preliminary stage. Then techniques of model construction and visualization are employed to generate the virtual models from scanned 3-D images. Surgeons can utilize the constructed models for the pre-operative analysis and planning, which is very necessary to simulate the outcome for certain type of surgeries **(Oentoro, 2009)**. During the intra-operative stage, the pre-operative models should be registered with the actual anatomy of the patient to generate the real-time virtual reality views of moving anatomy and surgical tools **(Taylor and Joskowicz, 2002)**.

The techniques of CADS are increasingly being used in modern clinical applications, including computer assisted neurosurgery, oral and maxillofacial surgery,

orthopaedic surgery, visceral surgery, and radiosurgery **(Wikipedia, 2011)**, and have achieved great progresses. However, due to the complexity of vasculatures, there still lacks an effective technique to specifically diagnose and perform surgery for vascular related diseases. Hence, particular attention has been given to the technique of computer-aided vascular diagnosis and surgery, which is also of great significance since vascular diseases are the leading cause of mortality in modern society. With respect to the data statistics from World Health Organization (WHO), about 17.3 million people died from various vascular diseases in 2008, and this number would rise to 23.6 million by 2030 **(WHO, 2011)**. The development of vascular-specific medical information processing system will make it possible to observe the vasculatures inside the human body. In addition, computer-aided systems are required to assist surgeons to carry out vascular surgeries, including stenting or bypass grafting for atherosclerosis, clipping or coiling for cerebral aneurysm, and stent grafting for aortic aneurysm. Computer-based information systems not only play a key role for the treatment decision making and pre-operative planning of vascular diseases, but also are an essential link between the pre-operative plan and the tools utilized by the surgeon to perform the computer guided minimally invasive vascular surgery **(Taylor and Joskowicz, 2002)**.

One of the essential and key issues involved in developing such kind of computer-aided vascular diagnosis and surgery system is the visualization and reconstruction of vasculatures, which can be a very challenging task, since the vasculature system consists of a very complex network of vessels.

1. 2 Vasculature Visualization and Reconstruction

The technique of medical visualization is to visualize the patient's anatomy inside the body without the need of dissection, which involves knowledge of medical imaging, digital image processing as well as computer graphics. Medical imaging technology enables to obtain a stack of two-dimensional cross sections from various medical scanners. Instead of having surgeon mentally correlate consecutive slices, the techniques of image processing and computer graphics make it possible to reconstruct a 3-D representation from the thin slices of 2-D cross sections **(Taylor and Joskowicz, 2002)**, which allows for viewing from various angles.

One extremely challenging task in developing medical information processing system is the 3-D visualization of vasculatures, which is of great helpful for the diagnosis and treatment of vascular diseases in a minimally invasive way **(Schumann et al., 2007)**. The visualization of vessel structures is of particular importance in many clinical applications, including the diagnosis of anomalous growths and stenosis **(Joshi et al., 2008)**, virtual angioscopy **(Bartz et al., 1999a)**, surgery planning, and computer guided vascular surgery. Depending on the purposes of application, the requirements of an appropriate visualization of vasculatures can be different from case to case. For example, in the case of diagnosing vascular diseases, it is essential to detect and depict the narrowing accurately by the local evaluation of the vessel cross section. For the purpose of intervention planning, the understanding of the topology of a vascular tree is crucial for predicting the consequences of an intervention strategy **(Schumann et al., 2007)**. During the intra-operative execution of computer guided vascular surgery, one of the essential tasks is to accurately register the pre-operative vascular tree to the patient in the operating room. In addition, the

3-D visualization of a vascular tree is crucial for virtual angioscopy, a non-invasive medical diagnosis procedure for exploring the human vascular system **(Gobbetti et al., 1998)**, which generates an interactive environment for vascular examination from and intra-vessel perspective **(Bartz et al., 1999a)**.

Unlike other organs of the human body, the vasculature system is a very complex network of vessels, which makes the 3-D visualization of vasculatures very challenging. In addition, due to image noise and limited resolution of Computed Tomography (CT) or Magnetic Resonance (MR) scanners, conventional techniques of medical volume data visualization, such as isosurface rendering, direct volume rendering or Maximum Intensity Projection (MIP), are not well-suited for the above-mentioned tasks **(Oeltze and Preim, 2005)**. Isosurface rendering can easily produce heavy artifacts and discontinuities due to inhomogeneity of contrast agent distribution. Image inhomogeneity may easily result in disconnected appearance of vessel structure in the periphery. In addition, other organs with the similar intensity values could be also included in the visualization of vasculatures **(Hahn et al., 2001)**. For MIPs, depth relations cannot be reflected correctly, and small vessels tend to disappear completely. The direct application of volume rendering techniques usually leads to severe aliasing artifacts when the vascular structures are rather small, as the resolution of the original data is relatively low for the direct volume rendering of vasculatures **(Pommert et al., 1992)**. All these problems arisen by conventional techniques can be solved by reconstructing the vascular geometry.

Reconstructing the actual vascular structures from the volumetric image datasets based on vessel geometric properties, such as vessel centreline, diameter, and curvature, has commonly been considered as an effective way for providing high-

quality visualization **(Gerig et al., 1993)**. The reconstructed geometry can help to greatly reduce the distracting aliasing effects and discontinuities due to the limited resolutions and discrete nature of radiological data, and achieve high visual quality of the vessel structures **(Boskamp et al., 2005)**. Furthermore, there are also various benefits of having an actual geometric representation of vasculatures:

- For the planning of vascular surgery, the accurate detection and detailed assessment of the pathologic lesion is necessary. With reconstructed vessel geometry, it is possible to locate more accurately the lesion position, and perform some quantitative analysis, which is crucial for making treatment planning decisions.

- In developing a computer-aided vascular surgery system, one of the major issues is to register the pre-operative vessel trees with the actual vascular objects. Unfortunately, it can be an extremely challenging task to locate and position the vessel structures directly from the image generated from direct volume rendering. On the other hand, with reconstructed vessel geometry, the registration problem can be solved more easily.

- The geometry representation of the vasculature not only makes it possible to achieve high visual quality from a perspective inside the vessel structures, but also makes the implementation of an interactive virtual angioscopy a much easier task, as the issue of collision detection of the virtual camera with vascular objects can be easily solved when there is an actual geometry representation for the vasculatures.

- Reconstructed vessel geometry with high-level smoothness also plays a crucial role in Computational Fluid Dynamics (CFD) to guarantee correct simulation results and to avoid numerical instabilities **(Schumann et al., 2008)**. The CFD of the blood flow enables us to study the hemodynamic characteristics such as intra-aneurysmal flow patterns or wall shear stress, which is of great help for the treatment decision of an aneurysm **(Schumann et al., 2008).**

- An explicit geometry representation of vasculature makes it much easier to design and evaluate possible modification of patient-specific vascular structures, which is crucial for implementing "manual hemodynamic optimisation" in the surgery planning phase **(Pekkan et al., 2008)**, and for some interventional procedures, such as coiling, stenting, and bypass operations **(Preim and Oeltze, 2007)**.

1. 3 The Aim and Objectives of this Research

In general, developing accurate, robust, reliable techniques for the visualization and reconstruction of vascular structures remains a tough task. Although various methods have been developed for reconstructing vascular structures, most of these approaches are model-based; in particular, a circular cross-section is often assumed. These methods can achieve a certain level of smoothness at a relatively fast speed, but they are far from accurate, as the model used in the reconstruction process is usually too ideal to correctly represent the actual variation presented by the cross-sections of a vascular structure. As a result, the reconstructed geometries are not suitable for computer aided diagnosis and computer guided minimally invasive vascular surgery. Further research is therefore required to investigate the major

issues associated with the development of vasculature visualization techniques. The main aim of this research is to develop associated techniques to accurately reconstruct the vessel geometry for the purposes of visualization and computer-aided clinical applications. The objectives are presented as follows:

1. To develop more robust segmentation techniques to extract vessel structures from raw medical datasets.

   The first task to vasculature reconstruction is to identify the vessel region from initial datasets. This is usually achieved using the technique of segmentation, which plays an important role in medical image processing. In our research, the segmentation of vessel trees from raw medical datasets is the essential preliminary step for the geometry reconstruction and visualization of vasculatures. Although many segmentation techniques have been proposed for various image segmentation problems, vessel segmentation still remains a challenging task **(Lesage et al., 2009)**. Our first objective is to investigate current techniques of medical image segmentation and to develop advanced method for the extraction of vascular structures.

2. To develop the implicit modelling techniques to reconstruct the geometry of vascular structures.

   The segmented dataset that identifies vessel structures is usually represented as a discrete point set, which is prone to produce poor visualization result and not suitable for the tasks of vessel shape analysis. It is required to develop techniques to reconstruct the solid geometry of vascular structures. Most current reconstruction techniques represent vascular structures either as polygonal

meshes or in parametric forms, which poses the difficulty of implementing blending and ramification of branching. In the area of computer graphics and geometric modelling, parametric surfaces still remain dominant position. However, increasing attention is being given to implicit surfaces, with special reference to their natural representing for solid objects and their innate blending properties **(Bloomenthal et al., 1997)**. In consequence, our second and foremost objective is to investigate and develop implicit modelling techniques for the accurate and smooth reconstruction of vascular structures.

3. To employ the technology of modern GPUs to accelerate the rendering of implicitly represented vasculatures.

   Although implicitly represented geometric objects have several advantages over parametric shapes, they usually require a relatively high computational cost to display. Fortunately, modern GPUs are very efficient in processing graphics information as well as complex computational tasks. Hence, it would be intuitive to exploit the advantages of GPU computing to achieve high rendering performance of implicitly reconstructed vasculatures.

4. To investigate the application of implicitly reconstructed vasculatures to virtual angioscopy.

   Virtual angioscopy is a non-invasive medical diagnosis procedure for exploring the human vascular system **(Preim and Oeltze, 2007)**. It is a useful technique for educational purposes and some diagnostic tasks, as well as intervention planning and intra-operative navigation **(Preim and Oeltze, 2007)**. The implicitly represented geometry of vasculatures is particularly suitable for the collision detection of the

virtual camera with vascular objects. Our final objective is to apply the implicitly reconstructed vascular geometry to virtual angioscopy to achieve high quality of perspective views as well as freely interactive navigation.

1. 4 Thesis Overview

The thesis mainly concentrates on the accurate geometry reconstruction and visualization of vascular structures from standard 3-D medical datasets (e.g. CT or MRI) by taking advantage of modern programmable GPUs. In addition, an intuitive and useful application (i.e. virtual angioscopy) is also explored. **Figure 1.1** illustrates the systematical diagram of our research methodology. Firstly, the vessel structures are extracted from 3-D medical datasets using our developed local hybrid level-set method to overcome the segmentation difficulty due to image inhomogeneity. Then a skeleton-based implicit modelling technique is proposed to reconstruct the continuous vessel surface from the segmented discrete vascular points. As implicitly represented geometric objects require a relatively high computational cost to display, a GPU acceleration technique is developed for the rendering of the implicitly reconstructed vasculatures. Finally, the reconstructed vasculatures are investigated for the application of virtual angioscopy, a non-invasive diagnostic procedure for exploring the human vascular systems.

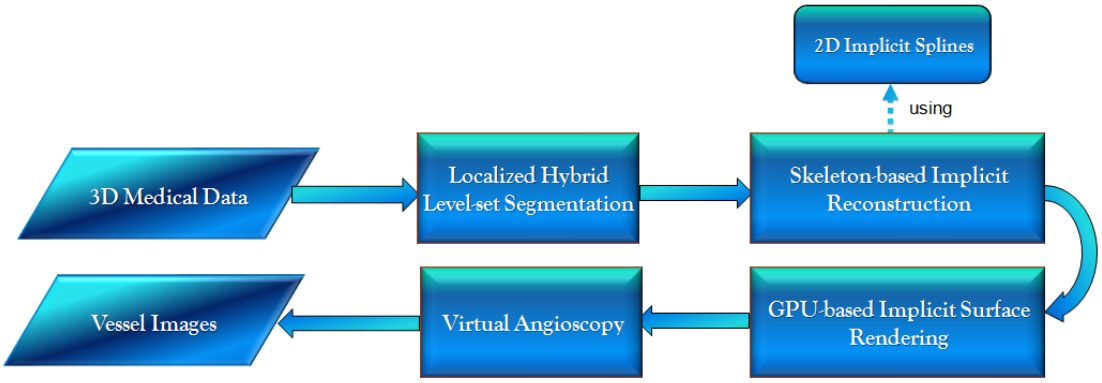


Figure 1.1: *The systematical diagram of our research methodology.*

The remainder of this thesis is structured as follows:

In **Chapter 2**, the essential background to this research is discussed, including the techniques of volume visualization, vasculature-specific visualization, reconstruction for vascular structures, and virtual angioscopy. An overview on modern GPU computing techniques is also given at the end of this chapter.

In **Chapter 3**, the techniques of medical image segmentation are investigated first. Based on the investigation, a localized hybrid level-set method for the segmentation of vasculatures is developed, which introduces a locally fitted binary energy function into the hybrid level-set framework. The introducing of local binary fitting energy into the hybrid level-set model allows one to extract local image information, which is essential for the segmentation of inhomogeneous images. Experimental results on 3-D medical images are presented to demonstrate the strengths of the proposed method.

In **Chapter 4**, a novel technique for the accurate reconstruction of the vasculature along its skeleton without model-assumptions is developed. Firstly, a method for

modelling implicit generalized cylinders is described. In this modelling method, freeform cross-sections are specified implicitly using the 2-D piecewise algebraic splines **(Li and Tian, 2009)**, and then, different cross-section profiles are weighted and summed up along the skeleton to form the smooth generalized cylinders. The smooth piecewise polynomial blending operation **(Li, 2007)** is utilized to blend together the branches of implicitly constructed generalized cylinders to construct complex anatomic tree structures. The proposed modelling method is subsequently applied to real medical datasets to demonstrate how to achieve an accurate reconstruction of a vasculature. Some implementation details are also discussed to refine our proposed methods. Finally, the reconstruction results with a comparison to other techniques, and the qualitative and quantitative validation are demonstrated at the end of this chapter.

In **Chapter 5**, the technology of modern GPUs is investigated to accelerate the rendering of implicitly represented vasculatures. We firstly, introduce the major approaches for rendering implicit surfaces, as well as the GPU programming techniques. Then a GPU accelerating technique is devised and applied to achieve high performance in rendering our implicitly reconstructed vasculatures. At the end of this chapter, we also conducted the relevant performance analysis.

In **Chapter 6**, a virtual angioscopy technique based on vasculature geometry reconstructed using our skeleton-based implicit modelling technique is developed. The major issues associated with virtual endoscopy are firstly investigated. Then a virtual angioscopy system based on our implicitly reconstructed vascular geometry is implemented, which can achieve much higher quality perspective views inside the

vessel as well as freely interactive navigation. Some experiments have been carried out to demonstrate the strengths of our technique.

Finally, **Chapter 7** summarizes the contributions of this research and presents some suggestions for future work.

Chapter 2
Research Background

This chapter discusses the essential background to this research. Firstly, the general techniques of volumetric data visualization are introduced. Following the investigation of specific techniques for visualizing vasculatures, we focus on the reconstruction techniques of vascular structures. This is because, with reconstructed vascular geometry, it is quite easy to produce smooth visualization of vessel surfaces. In addition, finding the accurate geometric representation of vascular structures is crucial in developing computer aided vascular surgery systems. Then the technique of virtual angioscopy is investigated, as it is an important and intuitive application based on the reconstructed vasculatures. Finally, we present an overview on modern GPU computing techniques, which can be exploited for achieving higher computing performance of medical image processing.

2.1 Techniques of Volume Visualization

Volume data visualization plays an essential role in modern scientific research and applications **(Drebin et al., 1988)**. Today, our vision and understanding to the internal structure of an opaque object, whether it be organic or inorganic, has been greatly enhanced by means of advanced scanners and sensors, which usually produce information on the internal structure of the object as a three-dimensional array of

numbers. Developing techniques to visualize such kind of datasets is therefore the first priority for a wide range of applications.

One obvious need arising in medicine is to visualize volumetric data obtained from various medical scanners, including Computed Tomography (CT), Magnetic Resonance Imaging (MRI), Emission-Computed Tomography [i.e., SPECT (Single Photon Emission Computer Tomography), and PET (Positron Emission Tomography)], and Ultrasound **(Bartrolí 2001)**. Particularly, a vascular-specific technique called Angiography has been invented to produce scanned information for the lumen of blood filled structures, such as arteries, veins, the heart chambers, and so on **(Straka, 2006)**. By intravenously inserting contrast medium into vasculatures, high-spatial resolution CT acquisition can give rise to volumetric data called CT Angiography (CTA). Similarly, Magnetic Resonance Angiography (MRA) is a variant of MRI with special settings of parameters to obtain information on blood in vessels **(Straka, 2006)**.

Generally, the volumetric data obtained from scanners is usually stored as a stack of 2-D slices (see **Figure 2.1**). Unfortunately, by means of viewing individual slices, it is quite difficult to obtain 3-D interior structure of a volume. So far, a variety of techniques of volume visualization has been developed to extract meaningful information from raw volumetric data by taking advantage of interactive graphics and imaging, concerning volume data representation, manipulation, and rendering **(Kaufman, 1991)**. Generally, these techniques can be classified into two main categories: volume rendering and surface rendering.

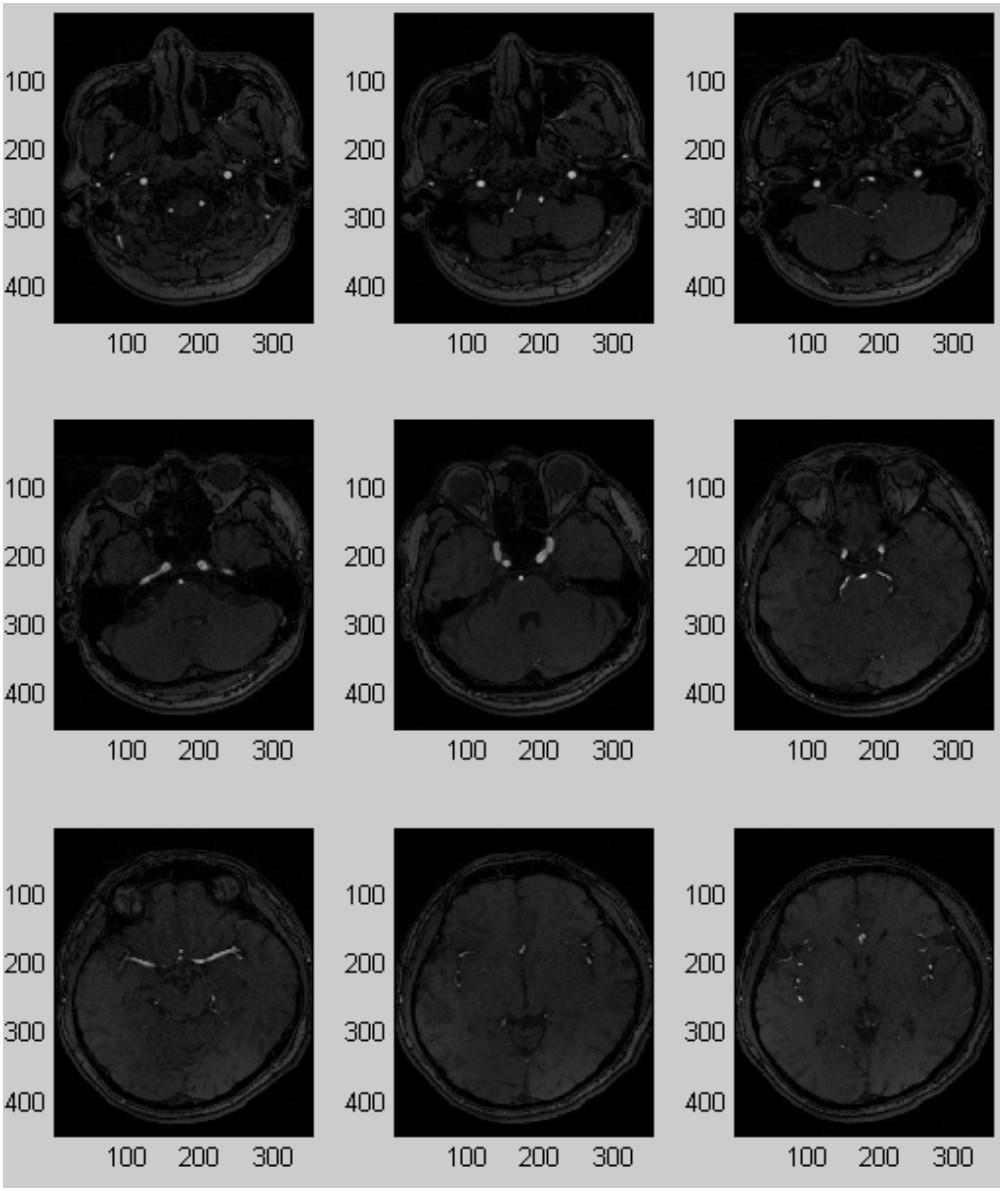


Figure 2.1 : *Some 2-D slices of the Magnetic Resonance Angiography (MRA) of brain.*

2.1.1 Volume Rendering Techniques

Volume rendering (sometimes called direct volume rendering) is a technique for directly displaying 3-D sampled datasets in the form of 2-D projection, without the intermediate geometric primitive representations **(Drebin et al., 1988)**. To render a 2-D projection of a 3-D dataset, a camera is firstly defined in space relative to the volume. Then, an RGBA (for red, green, blue, alpha) transfer function is used to

define the RGBA value for every possible voxel value. Then those converted RGBA values are composited and projected on a corresponding pixel of the frame buffer for displaying. The effectiveness and the efficiency of visualizing a volume dataset largely depend on the used rendering techniques **(Drebin et al., 1988)**.

Some popular volume rendering techniques include ray casting **(Tuy and Tuy, 1984)**, splatting **(Westover, 1990)**, texture mapping **(Cabral et al., 1994)**, and shear warp **(Lacroute and Levoy, 1994)**,

Ray casting

As the computation emanates from the output image, volume ray casting is classified as an image-space-based volume rendering technique **(Kaufman, 1997)**. In this technique, a ray from the centre of the preset camera (i.e. the eye point) is fired for each pixel on the image plane, floating in between the camera and the volume to be rendered. Then the ray is sampled at regular or adaptive intervals throughout the volume. The intensity values at each sample point are interpolated, and a transfer function is then invoked to find the colours and opacities for the corresponding intensity values. All of the colour and opacity values along the same ray are then composited in either back-to-front or front-to-back order to yield the final colour value of the corresponding image pixel to be displayed on the screen. The process is repeated for every pixel to form the completed image **(Pawasauskas, 1997)**.

Splatting

The splatting method is regarded as an object-space-based technique, as the computation derives from the input volume **(Kaufman, 1997)**. With this technique, the pixel colour is obtained by projecting each voxel onto the screen instead of casting rays from the eye. Generally, a front-to-back object-order traversal of the volumetric voxels is performed in the technique of splatting **(Elvins, 1992)**. In other words, the closest voxel to the image plane is splatted first, and all the voxels in the same slice are splatted before the next slice is started. Then the value at each voxel is transferred to the user-specified colour and opacity according to the transfer functions. The next step is to project the splatted voxels into image space and find their contributions to the image buffer. Each voxel's contribution is determined by its corresponding reconstruction kernel, which leaves a 2-D footprint on the screen. Then all the weighted footprints are accumulated into the final image.

Texture mapping

The use of 3-D texture mapping for volume rendering was introduced by Cabral et al. **(Cabral et al., 1994)**. Firstly, the volumetric data is interpreted as a 3-D texture and loaded into the 3-D texture memory of the graphics hardware. Then, polygonal slices parallel to the viewing plane are re-sampled using hardware-implemented trilinear interpolation. The reconstructed scalar values of the slices can be converted to RGB colours via a look-up table. Finally, the slices are then correctly blended using back-to-front or front-to-back compositing. By means of texture mapping, 3-D volume slices can be rendered efficiently with reasonable quality. However, the main drawback of the method is that the 3-D texture memory usually has a limited size, which may require the data set to be divided into bricks **(Bartrol ĺ 2001)**.

Shear warp

The shear warp algorithm is regarded as a hybrid technique between image and object-space algorithms, which is based on the shear-warp factorization of the viewing matrix **(Yagel and Kaufman, 1992)**. Firstly, a base plane is chosen corresponding to the face of the volume data that is most parallel to the viewing plane. Then the volume is divided into slices parallel to the base plane, and transformed to sheared object-space for a parallel projection by translating each slice. Sheared object-space is an intermediate system where all the viewing rays are parallel to the third coordinate axis **(Lacroute and Levoy, 1994)**. Afterwards, the slices are projected onto the base plane using a back-to-front or front-to-back compositing to generate a 2-D intermediate image in sheared object space, which is finally transformed to image space by warping it according to warping matrix.

2.1.2 Surface Rendering Techniques

Surface-based rendering typically extracts surfaces by fitting geometric primitives (i.e. polygons or patches) to constant-value contour surfaces in volumetric datasets **(Elvins, 1992)**. Broadly speaking, surfaces can be represented discretely as polygonal meshes, or continually as mathematical functions either parametrically or implicitly. A polygon mesh is a collection of vertices, edges and faces that defines the shape of a polyhedral object. Generally, the faces consist of triangles, quadrilaterals or other simple convex polygons. In order to render a polygonal mesh, the information of vertex list, normal list and face list is essential. For a parametric surface, each point on the surface can be expressed explicitly over a two dimensional parametric space. In general, a parametric surface can be represented as

$x = f_x(u,v)$, $y = f_y(u,v)$, $z = f_z(u,v)$, where u,v are two parameters indicating the geometric dimensions of the surface. This kind of surface provides a convenient mapping from an object to its embedded 3-D space **(Bloomenthal et al., 1997)**. For a implicit surface, on the other hand, the coordinates of the surface points are treated as functional arguments rather than functional values **(Bloomenthal et al., 1997)**. This kind of surface is represented by implicit functions defined in form $f(x,y,z) = c$, where c is a scalar constant (usually 0). In the field of computer graphics and geometric modelling, parametric surfaces still remain dominant position. However, increasing attention is being given to implicit surfaces, with special reference to their natural representing for solid objects and their innate blending properties **(Bloomenthal et al., 1997)**.

Surface rendering techniques usually begin with the segmentation of anatomical structures on the slices from the acquired medical datasets. Segmentation is the process of isolating individual parts from initial datasets. There exist various segmentation algorithms, ranging from manual delineation to semi-automatic **(Whitaker et al., 2001; Zhukov et al., 2003)** to fully automatic **(Singh et al., 1998)**, which remains to be an open problem in image segmentation. Once the contour of a structure is extracted, a 3-D surface model can be created. Polygonal mesh (i.e., triangle or square) based techniques are the most popular approaches used in practice to approximate the surface. These methods are usually called tiling algorithms **(Bartrolí 2001)**. Various algorithms and criteria have been proposed to obtain the optimal surface between the contours **(Keppel, 1975; Fuchs et al., 1977)**. Several surface rendering techniques for volume visualization have been developed,

including contour connecting **(Keppel, 1975)**, opaque cube **(Herman and Liu, 1979)**, marching cubes **(Lorensen and Cline, 1987)**.

Contour connecting

Firstly, contours are extracted for each data slice by the process of segmentation. Once the slice contours are found, the problem becomes to find an optimal tessellation (usually triangles) to connect the curves in each two adjacent slices **(Elvins, 1992)**. In **(Keppel, 1975)**, this problem has been transferred to find a path in a directed graph using heuristics. This problem were further specified in **(Fuchs et al., 1977)** as that of finding a minimum cost path in a directed toroidal graph. Finally, the strips of triangles are passed to a surface renderer, of which viewing, lighting, and rendering parameters have been properly set.

Opaque cube

The algorithm of Opaque cube (Cuberille) is a first widely used method for visualizing volume data, which is firstly proposed in **(Herman and Liu, 1979)**. The first step of this algorithm is to ensure that the volume is made up of cubic voxels. In other words, if the distance between the slices is larger than the size of the pixels in the slice, which is generally the common case, then the data must be resampled so that the gap between the slices is the same size as the pixels. Then an iso-value is chosen, and a traversal of the volume is processed to search for cells that are intersected by the isosurface **(Elvins, 1992)**. In the second step, each such cell generates six polygons (one for each face of the cell), of which geometric descriptions are passed to a surface renderer for image generation. In the case of multiple chosen thresholds, each corresponding set of cell polygons is rendered in a

different colour (as well as with a different opacity) to distinguish the generated multiple isosurface **(Elvins, 1992).**

Marching cubes

Among the techniques of surface rendering, Marching Cubes (MC) is comparatively easy to implement and is the most commonly used method. It firstly locates the surface corresponding to a user-specified value. Once triangles are created, the normals at each triangle vertex on the surface are also calculated to ensure a quality image of the surface **(Lorensen and Cline, 1987)**. Generally, the steps of the MC algorithm can be described as follows **(Lorensen and Cline, 1987)**:

1. Scan two slices and create cube.
2. Process binary vertex assignment: set cube vertex to value of 1 if the vertex value exceeds the preset iso-value; otherwise, set cube vertex to 0.
3. Calculate index for cube based on vertices.
4. Use index to lookup list of edges intersected.
5. Use densities to interpolate edge intersections and generate the triangle vertices.
6. Calculate normal to each triangle vertex by interpolating the unit normals at each edge vertex.
7. Export the triangle vertices and vertex normals.
8. March to next position and repeat.

One main problem with the MC method is that it sometimes produces ambiguous results with different topologies **(Chernyaev, 1995)**. The reason is that the MC method

only uses a variant of the isosurface topology for each configuration in the edge lookup table, while the trilinear function permits several different variants. Chernyaev addresses the this problem and increases the set of topologically different patterns from 15 to 33 **(Chernyaev, 1995).**

Another variant of marching cubes is marching tetrahedra **(Shirley and Tuckman, 1991; Bourke, 1997)**. In this improved algorithm, each cell is split into 5 or 6 tetrahedra, and facet vertices are determined by linearly interpolating where the isosurface cuts the edges of the tetrahedron. Two triangles are sufficient to display the isosurface inside the tetrahedral cell, as a tetrahedron has only six edges. This technique can avoid the problem of ambiguities in the traditional marching cubes algorithms. However, more processing and memory are required to generate more triangles **(Elvins, 1992)**.

2.1.3 Summary of Volume Visualization Techniques

Generally, the two categories of volume visualization techniques have their own advantages and disadvantages. Direct volume rendering is a technique for displaying 3-D volumetric datasets in the form of 2-D projection, without using geometric primitives as an intermediate representation, which is especially appropriate for visualizing volume dataset containing amorphous features, such as clouds, fog, and fire **(Elvins, 1992)**. However, one of the main disadvantages of volume rendering techniques is the expensive computational time, since it is necessary to traverse the entire dataset each time when an image is rendered. Moreover, because the 3-D scene is represented in discrete form, a low resolution volume data may yield high aliasing artifacts when the discrete data is sampled during rendering **(Kaufman, 1997)**. The surface-based rendering technique, on the other hand, utilizes geometric

primitives to explicitly represent the isosurface contained in the volumetric datasets. Compared to direct volume rendering, it has the advantages of simple implementation and low complexity. However, there are also several main drawbacks to this approach. It could suffer from the problems of occasionally producing ambiguous results with different topologies, incorrect handling with branching, and so on **(Elvins, 1992)**. In addition, amorphous phenomena cannot be adequately represented using explicit surfaces.

2.2 Vasculature-specific Visualization

The general techniques of volume visualization are applicable to the visualization of vessel structures from raw MRI or CT datasets. However, due to the complexity of vessel structures, dedicated techniques are required to clearly recognize the complex topology of vascular tree as well as the spatial relations with its surrounding structures **(Preim and Oeltze 2007)**. The suitability of visualization techniques also depends on specific clinical purposes. For therapy planning tasks, it is vital to understand the branching patterns of the vascular structures and the spatial relations with other relevant structures **(Boskamp et al., 2005)**. For diagnosis purposes, the accuracy of visualization is more important.

2.2.1 Volume Rendering of Vasculatures

Direct volume rendering, such as ray casting and splatting, is able to present realistic depth-cues by considering the spatial depth information of objects hidden within the volume datasets **(Schumann et al., 2007)**. However, when the vascular structures are rather small, the direct application of volume rendering techniques usually leads to

severe aliasing artifacts **(Pommert et al., 1992)**. The main reason is that the resolution of the original data is relatively low for the direct volume rendering of vasculatures, since they are relatively small objects inside the medical datasets. Pommert et al. suggest oversampling four times in all three dimensions to achieve more natural-looking vessels **(Pommert et al., 1992)**.

The design of transfer function, which controls what parts of the data are visible, is especially important for the volume rendering of vasculatures, since there is only a small fraction (1 or 2%) of vessel structures within the overall dataset **(Vega-Higuera et al., 2005)**. Unfortunately, the ability of the traditional one-dimensional transfer function is limited to classifying vascular structures from their surroundings **(Preim and Oeltze 2007)**. Vega et al. **(Vega-Higuera et al., 2003)** have explored 2-D transfer function based on measured values and gradient magnitudes extracted from the CT-Angiography (CTA) data in the clinical environment (see **Figure 2.2**). However, for different acquisition devices, the gradient values of the generated volumetric data are also quite different. Hence, it is difficult to adjust the 2-D transfer function **(Schumann et al., 2007)**. Vega et al. **(Vega-Higuera et al., 2005)** have also implemented an adequate 2-D transfer function editor to obtain high quality volume representation using a reduced number of voxels.

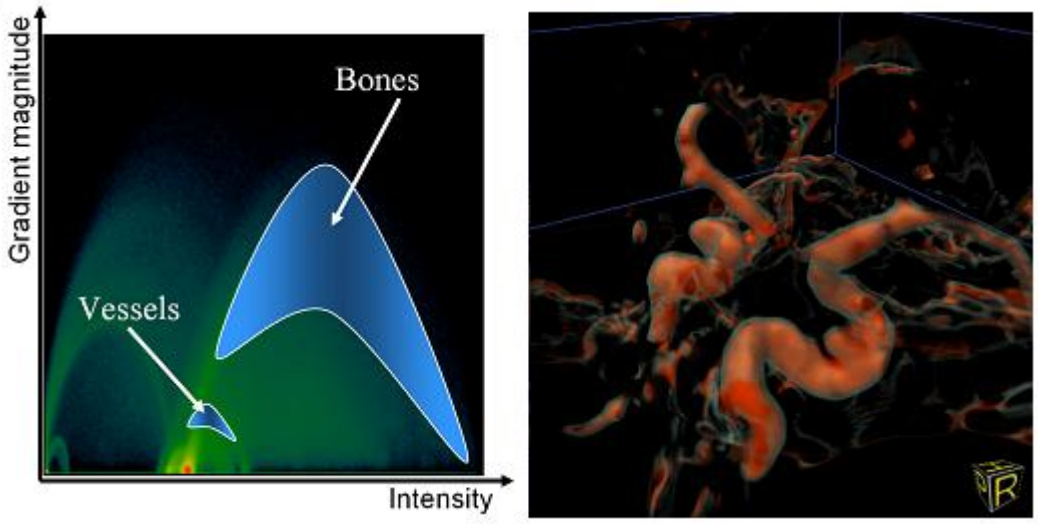


Figure 2.2: *Volume rendering of vasculatures with transfer function. 2-D transfer function based on measured values and gradient magnitudes (left); 3-D rendering result using 2-D appropriate transfer function (right).* **(Preim and Oeltze, 2007)**

Besides designing appropriate transfer functions, the enhancement of vascular structures is also crucial for the volume rendering of vasculatures. The vessel enhancement procedure as a pre-processing step of volume rendering is of great help to improve small delineation and reduce organ over projection **(Frangi et al., 1998)**. Frangi et al. have proposed a multiscale vessel enhancement filter, which utilizes the eigenvalue analysis of the Hessian matrix for the presence of a cylindrical structure **(Frangi et al., 1998)**. One of the limitations of this kind of methods is that the strong single cylinder assumption could fail at vessel branch points **(Joshi et al., 2008)**. Instead of using the single cylinder assumption, a non-parametric formulation is utilized in **(Qian et al., 2008)** to enhance regions in both cases of single vessels and branch points.

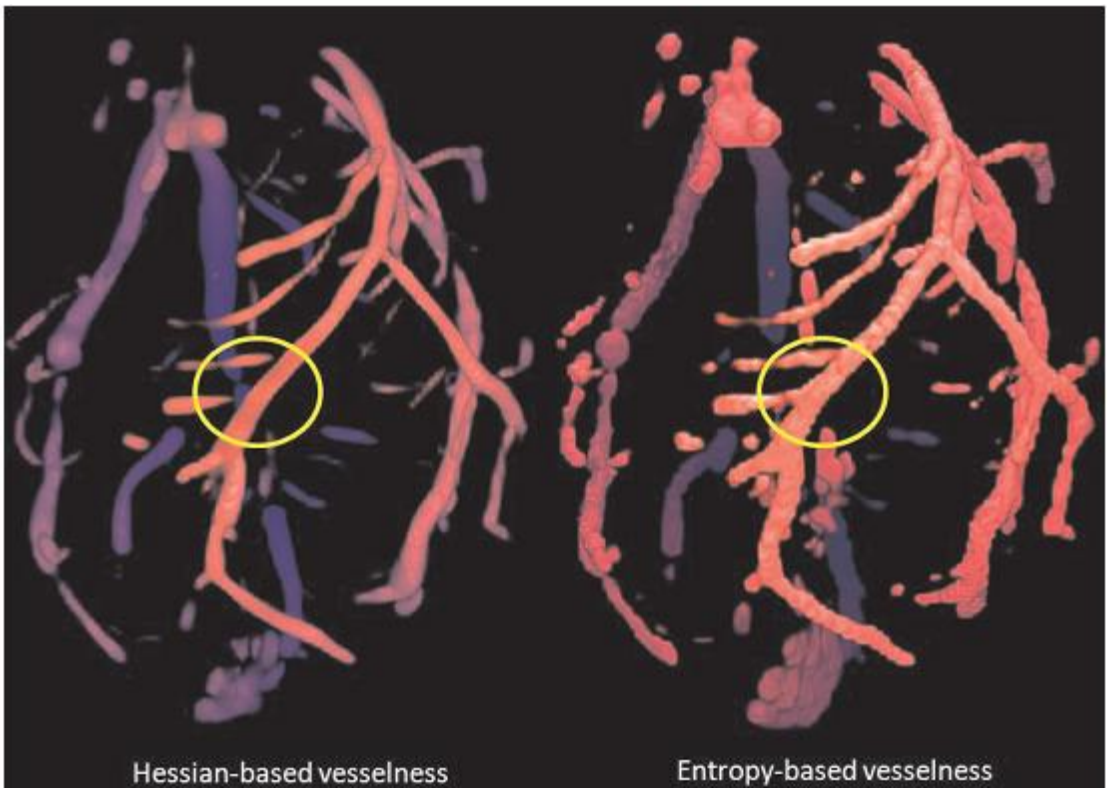


Figure 2.3: *The comparison of visualization result using polar profile (left) and Hessian techniques (right). As can be seen from the regions marked with a yellow, the polar profile based technique performs better at branch points.* **(Joshi et al., 2008)**

Based on the Non-Parametric Vessel Detection Method, Alark Joshi et al. have developed a method for efficient visualization of complex vascular structures **(Joshi et al., 2008)**. In this method, the Polar Profile-Based Vesselness Coefficient is firstly calculated for vessel detection. Then, this computed vesselness is incorporated into the volume rendering pipeline to visualize the raw data. The results show that this algorithm works particularly well for visualizing branch points in vessels. This technique better identifies branches and connections with other vessels (see **Figure 2.3**), when compared to standard Hessian based techniques, which are "fine-tuned to identify long cylindrical structures" **(Frangi et al., 1998)** .

2.2.2 Surface Rendering of Vasculatures

Direct application of isosurface rendering to vascular volume datasets is easily to produce heavy artifacts and discontinuities due to image noise and inhomogeneity of contrast agent distribution. Image inhomogeneity may easily result in disconnected appearance of vessel structure in the periphery. In addition, other organs with the similar intensity values could be also included in the visualization of vasculatures. Therefore, in many cases, vascular structures need to be explicitly segmented **(Schumann et al., 2007)**, though vascular segmentation still remains a challenging task **(Lesage et al., 2009)**. So far, many segmentation techniques have been proposed for various image segmentation problems, ranging from threshold based methods, to pattern recognition based methods, and to deformable model based methods **(Ma et al., 2009)**. Two excellent surveys on vessel extraction techniques can be found in **(Kirbas and Quek, 2004; Lesage et al., 2009)**.

Marching Cubes (MC) **(Lorensen and Cline, 1987)** is the most commonly used surface rendering technique for the segmented vascular volume data. However, due to the use of linear interpolation for filling the gap between neighbouring voxels, the quality of visualization result based on MC is relatively low **(Schumann et al., 2007)**. Although simple smoothing procedures can be employed to reduce the aliasing artifacts of surface visualization, current smoothing methods for visualizing vascular structures still do not lead to the desired results in general **(Bade et al., 2006)**. Laplacian smoothing procedure, which simply replaces the location of a vertex with the average of the locations of vertices in the neighbourhood on the mesh, is not appropriate at all since small branches may easily collapse **(Schumann et al., 2007)**. Some improved approaches, such as LowPass-filtering, which extends the Fourier

analysis to surfaces of arbitrary topology **(Taubin, 1995)**, and the improved Laplacian smoothing **(Vollmer et al., 1999)**, can achieve better results because they try to prevent shrinking. For the visualization of binary volume data, a few smoothing techniques have teen also developed to avoid typical terracing artifacts **(Schumann et al., 2007)**. Constrained Elastic Surface Nets (CESN) **(Sarah and Frisken, 1998)** is the relatively better method to achieve smooth visualization of binary segmentation results. In this method, surface nets are created by connecting the notes on the surface of the binary segmented volume. Then the node positions are adjusted to reduce energy in the surface nets while following the constraint that each note should remain within its original surface cube. Although this method can achieve a "good trade-off" between smoothness and accuracy, it still produces unsatisfactory results for small vascular structures **(Preim and Oeltze, 2007)**.

2.3 Reconstruction of Vascular structures

Due to the limited resolutions and discrete nature of radiological data, conventional techniques of volume visualization are prone to producing annoying visual artifacts. To tackle this problem, reconstructing the actual vascular structures from the volumetric image datasets based on vessel geometric properties, such as vessel centreline, diameter, and curvature, has been considered for improving the quality of visualization **(Gerig et al., 1993)**. With reconstructed vascular structures, it can be very easy to produce smoothly rendered blood vessel surfaces **(Boskamp et al., 2005)**. In addition, finding the accurate geometric representation of vascular structures is also an essential task in developing computer aided vascular surgery systems, as surgeons need not only to visualize clearly the vascular structures, but also know their spatial geometric shapes. Furthermore, the reconstruction of vascular geometry

is also a essential step in performing quantitative analysis, the interactive modification of the vessel geometry, virtual angioscopy, and so on **(Preim and Oeltze, 2007)**.

As the vasculature system is a very complicated network of vessel, the reconstruction of vascular structures is a very challenging task. One of the solutions is to use skeleton curves to represent the topology of vasculatures, since the skeleton of an object has the ability to naturally capture important shape characteristics in three-dimensional contexts **(Faugeras, 1993)**. Actually, the skeleton-based reconstruction has been regarded as the most natural option to efficiently construct complete vascular structures **(Wu et al., 2010)**.

2.3.1 Explicit Reconstruction of Vascular Structures

Various skeleton-based methods have been proposed for the explicit reconstruction of vascular structures from segmented dataset. When performing skeleton-based vascular surface reconstruction, the first thing that needs to be done is to determine the vessel centreline and the local vessel cross-sections **(Preim and Oeltze, 2007)**. The generation of the skeleton is usually achieved by the technique of morphological thinning, or a step by step approach, which moves a small sub-volume, such as parallelepiped or sphere, to recursively slide along the vessel tree **(Kirbas and Quek, 2004)**. Once the skeleton has been extracted from the input segmented data, the explicit geometry models of vasculatures can be constructed by using the geometric information provided for the vessel cross-sections. When only diameter is available for the vessel cross-section, certain model-based approaches can be used to construct the vascular surfaces by using certain types of geometric primitives, such

as cylinders **(Masutani et al., 1996)**, truncated cones **(Hahn et al., 2001)**. However, the quality of the vascular surfaces generated using this method depends on the number of vertices used to approximate the cross-section profile and the number of geometric primitives associated with the skeleton **(Preim and Oeltze, 2007)** (see **Figure 2.4**). Although these methods can achieve a certain level of smoothness at a relatively fast speed, when the model is incorrect, which is often the case, they may lead to geometric surfaces that are quite unreal. In addition, this type of methods is prone to producing discontinuities at branching when geometric primitives are fitted together.

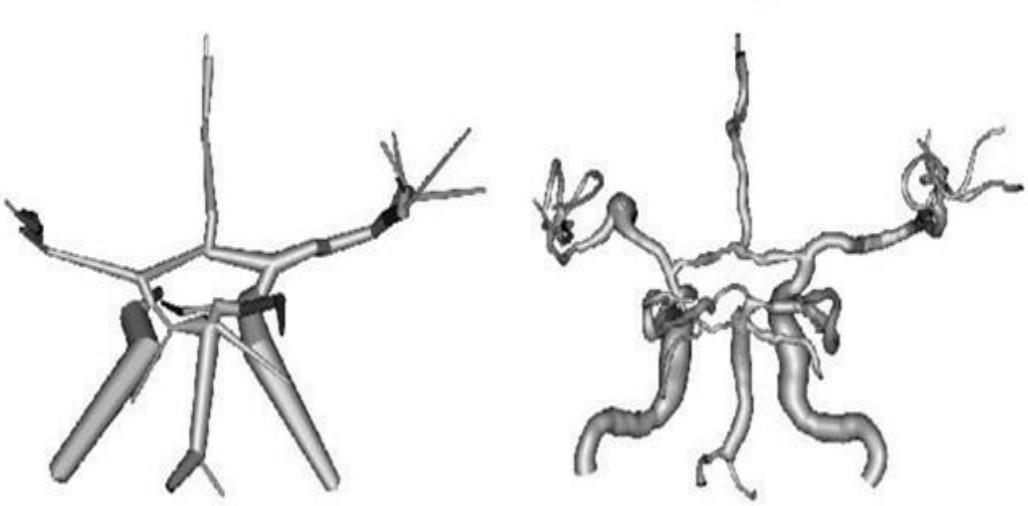


Figure 2.4: *The reconstructed results of MRI cerebral vessels with different resolutions: moderate sampling rate (left), and maximum sampling rate (right).* **(Hahn et al., 2001)**

Various shape modelling techniques have been suggested to construct geometric shapes along skeletons such that the constructed shapes can achieve smooth transition at the points of branching. Hohne et al. have proposed an approach to approximate small vascular structures and nerves using B-spline surfaces **(Hohne et al., 2000)**. Felkel et al. **(Felkel et al., 2002; Felkel et al., 2004)** have proposed a fast and

simple method (i.e. subdivision method) for generating surfaces of branching tubular structures with given centrelines and radii (see **Figure 2.5**). This approach serves surface meshes as the data representation of a vessel tree, which is suitable for real-time Virtual Reality operation planning and operation support within a medical application. Based on connection of polygonal cross-sections along the medial axis and subsequent refinement, Bornik et al. have presented a method (Simplex Meshes) for the reconstruction and representation of vascular structures (see **Figure 2.6**) **(Bornik et al., 2005)**. However, all the methods mentioned above are based on explicit models, which express the underlying geometric shape either as a polygonal mesh or as a parametric surface. As has been pointed out, they are prone to generating visual artifacts and are difficult to perform shape blending operations in general **(Bloomenthal, 1995)**. Recently, Wu, X., et al. **(Wu et al., 2010)** have improved Felkel et al.'s meshing method to achieve relatively accurate and smooth vascular structures. However, this approach is based on the fitting of explicit circles or ellipses to the cross-sections of vessel, which in general cannot provide an accurate approximation of the actual vascular objects. In addition, the underlying shape is expressed as polygonal meshes, so it is difficult to implement the ramification of branching **(Bloomenthal, 1995)**.

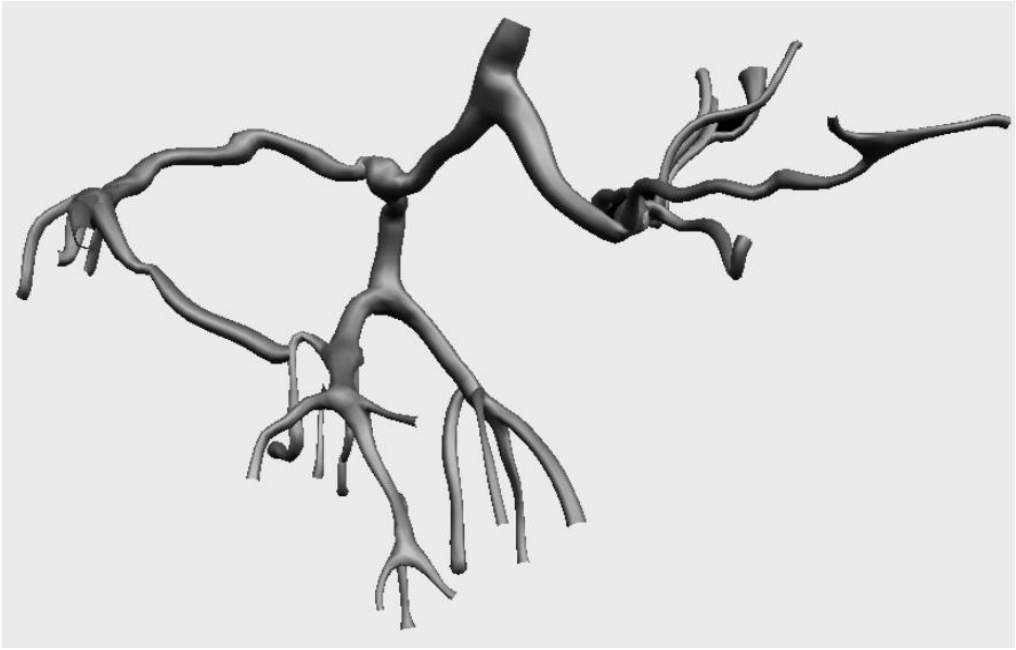


Figure 2.5: *Subdivision surfaces applied to the reconstruction of a liver vessel tree*. **(Felkel et al. 2004)**

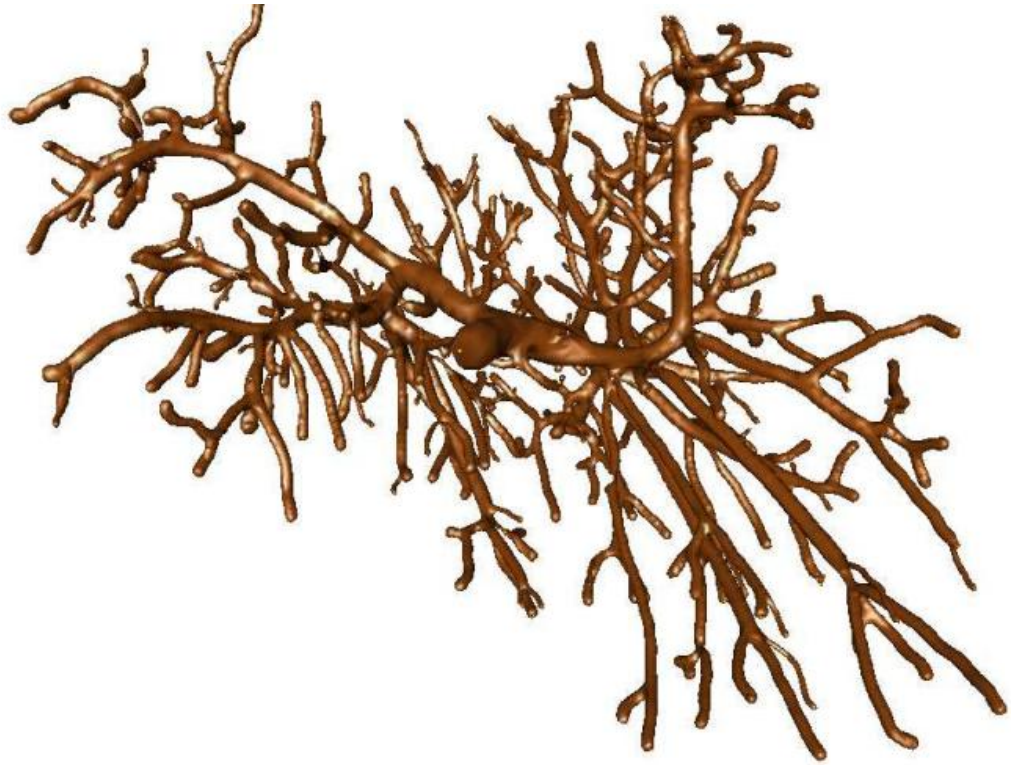


Figure 2.6: *Simplex meshes applied to the reconstruction of a portal vein tree.* **(Bornik et al., 2005)**

2.3.2 Implicit Representation for Vascular Structures

Another way to represent the underlying geometric shape is to use an implicit function, such as the methods proposed in Bloomenthal's work **(Bloomenthal et al., 1997)**. A classical example of an implicit function is the description of a sphere. Given a sphere centred at $\mathbf{c}(x_0, y_0, z_0)$, with radius r, the corresponding implicit representation of the sphere can be given by

$$(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 - r^2 = 0 \tag{2.1}$$

which represents all space points $P(x, y, z)$ on the surface of the sphere centred at $\mathbf{c}$ with radius r. In general, let $F(x, y, z)$ be a mapping from R^3 to R, and let α be a scalar, then $F(x, y, z) = \alpha$ uniquely defines a 3-D geometric surfaces. The geometric shape defined using $F(x, y, z)$ can also be interpreted as a solid object shown below

$$F_\alpha = \{(x, y, z) \mid F(x, y, z) \le \alpha\} \tag{2.2}$$

Generally, implicitly represented geometric objects have several advantages over parametric shapes. First of all, when objects are modelled as implicit function, we can directly tell whether a point lies inside or outside the shape, and the problem of boundary detection can be easily solved **(Lin and Gottschalk, 1998)**. Secondly, different implicit shapes are inherently easy to combine in various ways **(Barthe et al., 2003; Pasko et al., 2005; Li, 2007)**. In other words, it is easy to implement ramification of

branching, which is a tough problem when the underlying geometric shape is expressed as a polygonal mesh or as a parametric surface. In addition, the implicit function value at a point can be used to measure the distance from the point to the surface **(Li et al., 2006)**.

One popular way of constructing an implicit surface along a given skeleton is to use the Convolution Surfaces (CS) **(Bloomenthal and Shoemake 1991)**. With convolution surfaces, the scalar value at $P(x,y,z)$ is calculated according to the following equation

$$F_S(P) = \int_a^b e^{\left(\frac{-|S(t)-P|^2}{2}\right)} dt \qquad (2.3)$$

where S denotes a skeleton curve defined parametrically over $[a,b]$.

The technique of CS has been employed in **(Oeltze and Preim, 2005)** to reconstruct vascular structures. The vessel skeleton and local diameter information serve as the input. The filter design has been fine-tuned to avoid irritating bulges and to represent the course of the vessel diameter faithfully. The CS based technique can produce smooth transitions at branchings and closed, rounded ends (see **Figure 2.7**), which usually shows much better visual quality, when compared with the truncated cone method **(Hahn et al., 2001)** (see **Figure 2.8**). However, the reconstructed vascular geometry is only a morphological approximation, which is far from accurate and does not meet the basic requirements of computer aided vascular diagnosis and

computer guided vascular surgery, due to its model-simplifying assumption of circular cross-sections.

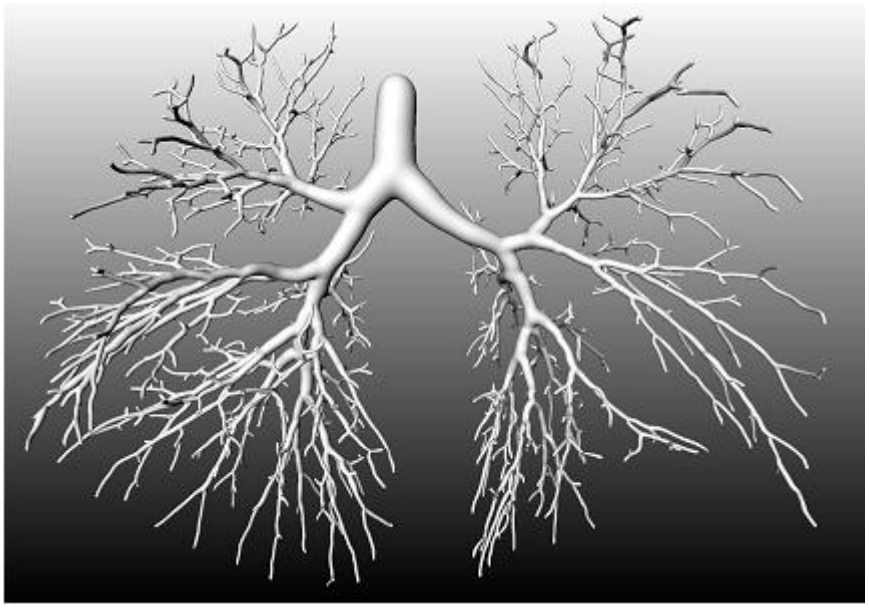


Figure 2.7: *Reconstruction of a bronchial tree derived from a clinical CT dataset.* **(Oeltze and Preim, 2005)**

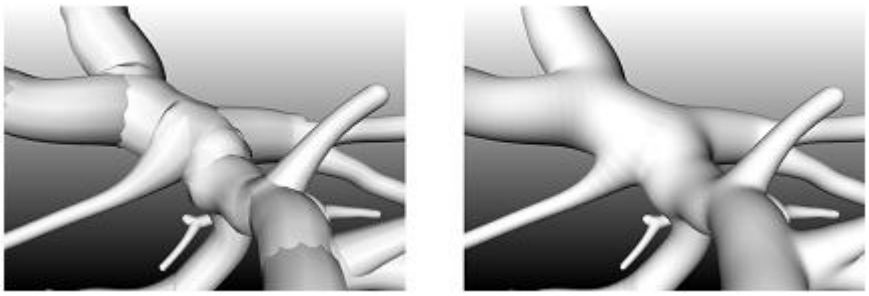


Figure 2.8: *Close-up of a reconstruction with truncated cones with artifacts along the seams (left). Smooth reconstruction with implicit functions of the same dataset (right).* **(Oeltze and Preim, 2005)**

Schumann et al. **(Schumann et al., 2007; Schumann et al., 2008)** have proposed a method for the implicit vessel surface reconstruction based on Multi-level Partition of Unity

(MPU) Implicits **(Ohtake et al., 2003)**. The basic principle of this method is to convert the segmentation result into a point cloud that is transformed into a surface representation by means of MPU Implicits. In MPUI technique, piecewise quadratic functions are fitted to capture the local shape of the surface; weighting functions are utilized to blend together these local shape functions. In other words, the global implicit function representing the whole shape is computed as the summation of all weighted local approximations. Generally, the implicit surfaces generated by MPUI are able to represent a given geometry smoothly without explicitly reconstructing the geometry **(Preim and Oeltze, 2007)** (see **Figure 2.9** and **Figure 2.10**). However, due to the complex and fine nature of most vascular tree, the quality of the resulting surfaces is usually poor [see **Figure 2.10**: **(c)**] and has to be improved using an additional remeshing step, either based on parameterization, or fitting of subdivision surfaces **(Boubekeur et al., 2006; Schumann et al., 2008)**, which subsequently increases the effort required from the reconstruction process and inevitably introduces further errors. In addition, the techniques based on MPU Implicits cannot guarantee to define an implicit volume **(Schmidt and Wyvill, 2005)**. This is because that the blending of local approximations with finite support may generate additional internal isosurfaces **(Reuter, 2003)**.

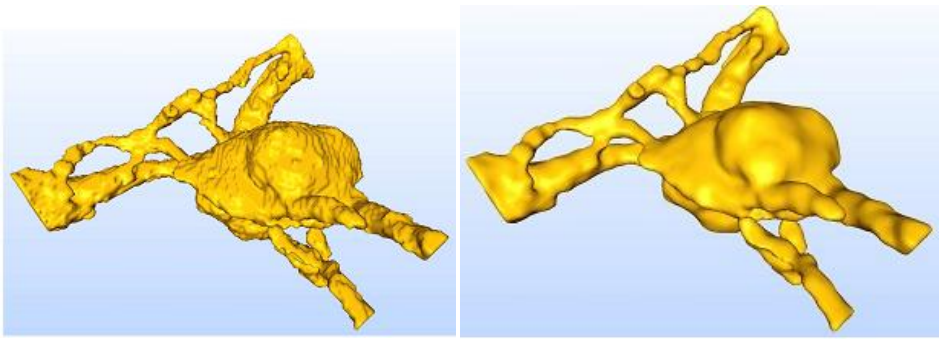


Figure 2.9: *Reconstruction of an aneurysm with MC (left) and MPUIs (right).* **(Schumann et al., 2007)**

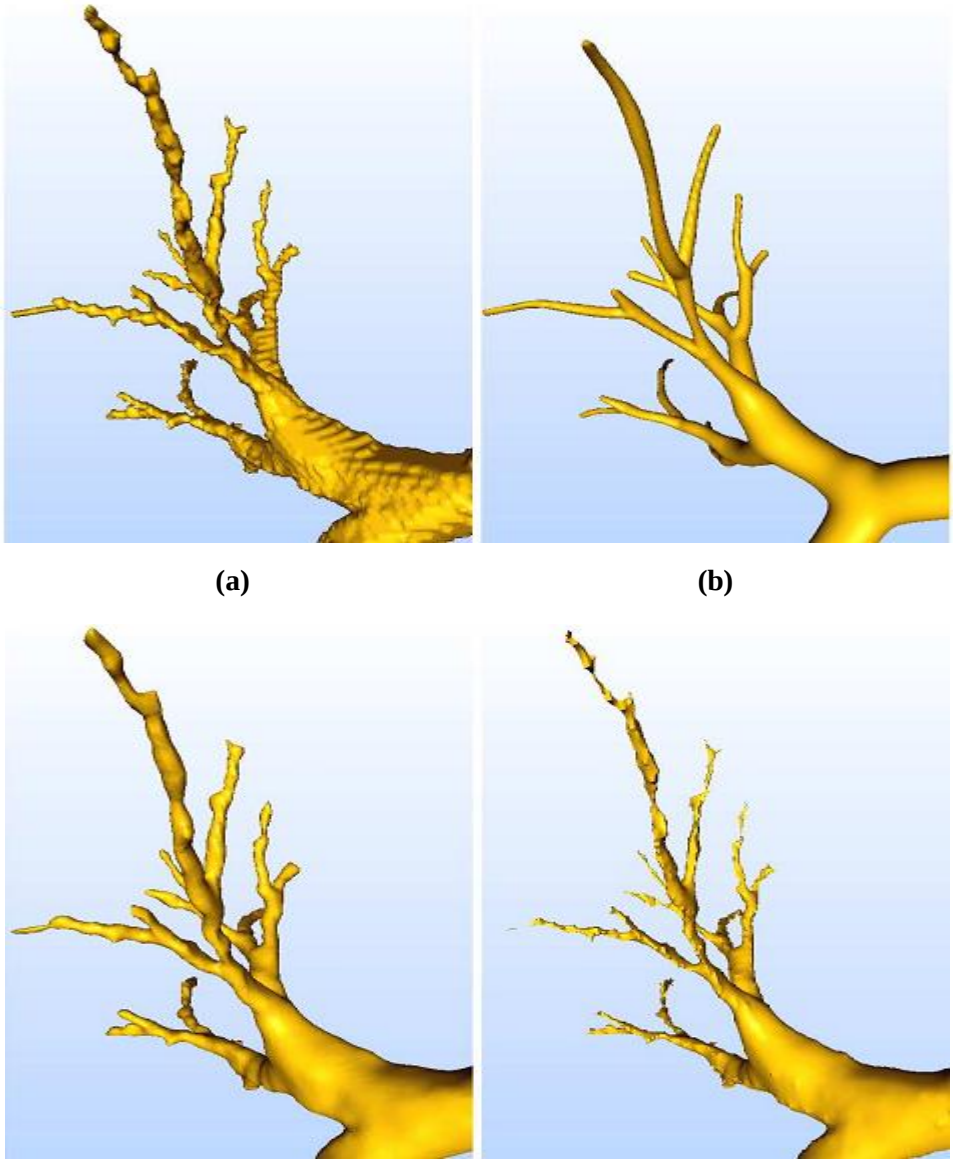


Figure 2.10: *A detailed look at the reconstruction results of brochial tree: Recontruction using MC* **(a)**, *CS* **(b)**, *MPUI* **(c)** *and CESN* **(d)**. **(Schumann et al., 2007)**

2.4 Virtual Angioscopy

Virtual endoscopy is a technique of diagnosis based on 3-D image processing to provide simulated visualization **(Geiger and Kikinis, 1994; Rubin, 1996)** of specific hollow organs, similar or equivalent to those produced by standard endoscopic procedures **(Wickham, 1994)**. Using a real endoscope is always painful or at least

uncomfortable for the patient, whereas virtual endoscopy provides us a non-invasive diagnostic technique.

The common visualization technique used for developing a virtual endoscopy is surface rendering, which produces images close to those of a real endoscopy. In some cases, external structures are of interest, then the combination with volume rendering techniques is necessary **(König and Gröller, 2001)**. By controlling a virtual camera to fly through the simulated hollow structures, virtual endoscopy can generate an interactive environment for the examination of the patient specific organ from a point of view inside the hollow organ. Additionally, the technique of virtual endoscopy facilities us to achieve several control options, such as the adjustment of viewing parameters (scale, angle, and direction), immediate translocation to new views, measurement, and lighting, which is not possible with real endoscopy **(Robb, 1996)**. Furthermore, in several clinical practices, some important human body systems that are incompatible with real invasive endoscopy, including spinal canal, inner ear, heart, large blood vessels, biliary and pancreatic ducts, are suitable anatomic structures where virtual endoscopy can be applied **(Robb, 1996)**.

Virtual fly-through of vascular structures is a useful technique for educational purposes and some diagnostic tasks, as well as planning vascular interventions. This technique is usually called virtual angioscopy, a special instance of virtual endoscopy **(Preim and Oeltze, 2007)**. In general, it is essential to combine detailed views of the inner structures with an overview of the anatomic structures (see **Figure 2.11)**.

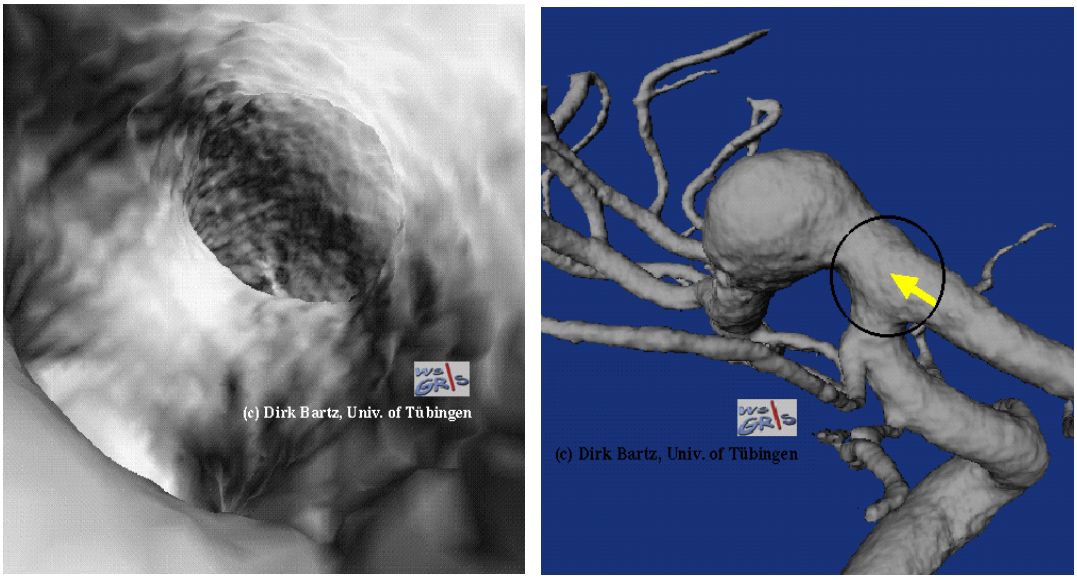


Figure 2.11: *Virtual angioscopy for the diagnosis of a cerebral aneurysm. Perspective view inside the vasculatures (left); 3-D overview of the vessel tree (yellow arrow represents the current position and direction of virtual camera) (right).* **(Bartz, 2003)**

One of the main problems associated with virtual angioscopy, as well as other areas of virtual endoscopy, e.g. virtual colonoscopy, is to find an appropriate navigation path to control the motion of the virtual camera **(Carvalho, 2003)**. The interaction of the user to control the movement of the virtual camera inside simulated hollow organs is not an easy task. It requires operations of mapping the input device movements to camera parameter modifications, which should not make the user get the feeling of "lost in space" **(Vilanova et al., 1999)**.

2.5 GPU Computing

Processing medical volume data requires considerable computations. With the development of scanner technology, current medical scanners are able to obtain the fine features of the scanned objects, which results in much larger volumetric data. Furthermore, in some clinical situations, real-time visualization of the medical data

is essential. Hence, it would be advisable to investigate the techniques of hardware acceleration for achieving higher performance of medical image processing. The modern Graphics Processing Unit (GPU) is not only a powerful graphics engine, but also being used for general-purpose computing **(Owens et al., 2007)**. The highly distributed architecture with massively parallel processors and huge memories, as well as high programmability, make modern GPUs not only very efficient in processing graphics information, but also as a sort of general purpose graphics processor for effectively processing a range of complex computational tasks. A broad range of applications and tasks have been successfully implemented on modern GPUs, including the areas of geometric computing, global illumination, signal processing, physically based simulation, databases and data mining **(Owens et al., 2007)**.

2.5.1 Architecture of Modern GPUs

Nowadays, the GPU has evolved from a fixed-function processor for processing 3-D graphics into a mature parallel programmable processor **(Owens et al., 2008)**. For the first generation of GPUs, the graphics pipeline, including vertex operations, primitive assembly, rasterization, fragment operations, and composition, is a fixed-function pipeline, which is implemented with fixed hardware units for vertex transformation and fragment texturing **(Al-Suyyagh, 2009)**. In other words, the operations available at the vertex and fragment stages are configurable but not programmable. Afterwards, in the first generation of programmable GPUs, the fixed per-vertex and per-fragment operations are replaced with user-specified programmable processor units **(Owens et al., 2008)**. The programmable processor units enable programmers to apply a set of algorithms to the vertex and fragment

operations, which is essential for expressing more complicated shading and lighting effects. However, in the early days of the programmable GPU architecture, as the instruction set of the vertex and fragment programs are quite different, the programmable processor units for vertex and fragment operations are separated, which could result in a low overall throughput. For example, when rendering a large triangle, the vertex processors are idle while the fragment processors are busy **(Al-Suyyagh, 2009)**. Therefore, in the latter generations of programmable GPUs, all the processor units are unified, which are also called shaders. The unified shader models enable to offer dynamic load balancing among varying vertex and fragment processing workloads **(Al-Suyyagh, 2009)**, which could greatly improve the throughput of GPUs. Furthermore, with the introduction of Computing Shader Model in DirectX 11, the architecture of the modern GPU is gradually evolved into an unified computing architecture, built around with massively single unified data-parallel programmable units **(Owens et al., 2008)**.

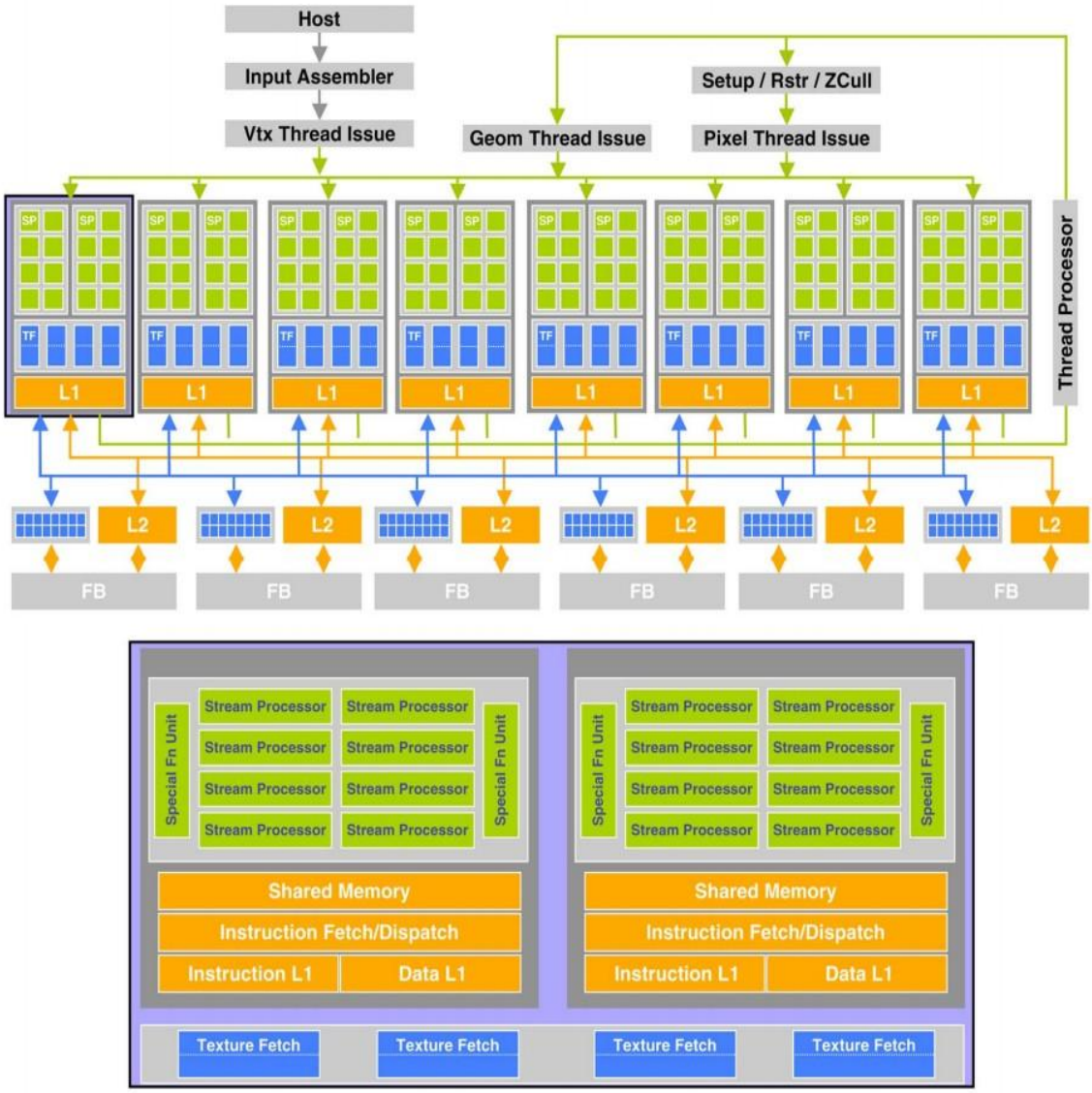


Figure 2.12: *The NVIDIA build architectures with unified, massively parallel programmable units at their cores.* **(NVIDIA, 2006)**

Figure 2.12 is the unified shader architecture of NVIDIA's flagship GPUs. As shown in the figure, the NVIDIA GeForce 8800 GTX features 16 streaming multiprocessors of 8 thread (stream) processors each. One pair of streaming multiprocessors is shown below; each contains shared instruction and data caches, control logic, a 16 KB shared memory, eight stream processors, and two special function units **(Owens et al., 2008)**. With the unified shader architecture, the majority of unified shader processors (i.e. stream processors) can be applied to processing

vertex data when an application is vertex-shader intensive. Similarly, the majority of unified shader units can be applied to pixel processing when its pixel shader is heavy **(NVIDIA, 2006)**. In other words, unified stream processors in GeForce 8800 GPUs can be regarded as general purpose floating-point processors, which can effectively process vertices, pixels, geometry or physics **(NVIDIA, 2006)**. With all the programmable power in a single stream processor, GPGPU programmers can now target that programmable unit directly, instead of dividing work across multiple hardware units **(Owens et al., 2008)**.

2.5.2 GPU Programming

The rapid increase in programmability has greatly widened the range of applications and tasks that can be implemented on modern GPUs. Generally, the programmable units of the GPU follow a single-program multiple-data (SPMD) programming model, which allows the hardware to fully exploit the application's data parallelism. Based on the SPMD model, GPU processes many elements (vertices, fragments, or geometric primitives) in parallel using the same program. Each element is independent from the other elements, and elements cannot communicate with each other **(Owens et al., 2008)**.

In the initial stage of GPU programming, the programming languages are specifically designed for graphics tasks, which are often referred to as shading languages, such as Cg/HLSL, GLSL. However, the main issue with this kind of GPU programming languages for GPU computing is that they are inherently shading languages. Computation has to be expressed in graphics terms like vertices, fragments, textures, and blending, which greatly limits the applications of GPU

computing for general purposes. Hence, it is necessary to develop more general GPU-specific programming languages for using GPU as a general-purpose computing processor, which can provide GPU functionality while hiding the GPU-specific details from the programmer **(Owens et al., 2007)**.

Buck et al. **(Buck et al., 2004)** developed Brook, a GPU programming language for performing general-purpose computation on programmable graphics hardware, which enables the use of GPU as a streaming coprocessor. In the stream programming model, data are organized as parallel streams, and computation is represented as kernels, which allows for efficient computation and data transfer **(Owens et al., 2008)**.

Then Glift **(Lefohn et al., 2006)** has been developed for defining convenient and random-access data structures on a graphics processor (GPU) to facilitate algorithmic development. As stated in **(Lefohn et al., 2006)**, Glift is an efficient, high-level abstraction with GPU random-access data structures for both graphics and GPGPU applications. GPU data structures are factored abstractly into four components: physical memory, programmable address translation, iterators, and virtual memory, which could benefit application developers to create more complex applications as well as facilitate hardware vendors to offer implementations optimized for their architecture **(Lefohn et al., 2006)**.

Scout **(Mccormick et al., 2007)** has gone one step further to provide general-purpose programming constructs as well as extensions for scientific visualization operations. In Scout environment, tools are provided for interacting with visualization results and simple code development tasks.

Both AMD and NVIDIA have developed their own GPGPU programming systems. ATI (now AMD Graphics Products Group) has developed Close To the Metal (CTM), a low-level programming interface for general-purpose computing. CTM enables developers for direct access to the native instruction set and memory of the massively parallel computational elements in modern AMD graphics cards. Compared to AMD's CTM, NVIDIA's CUDA (Computed Unified Device Architecture) **(NVIDIA, 2007)** is a higher level interface, which extends C/C++ with parallel stream processing concept to take full advantages of massively parallel architecture. CUDA exposes much more of the hardware resources than Brook. In CUDA environment, multiple levels of memory hierarchy are exposed, including fast shared memory between threads in a block, per-thread registers, board memory, and host memory. In addition, the allowing of using pointers, general loading/storing to memory within a kernel, and synchronization between threads in a thread block, make kernels in CUDA are also more flexible that those in Brook. **(Ortner, 2008)**

2.6 Summary

Based on the investigation of the essential background, it can be deduced that conventional techniques of medical volume visualization, i.e. direct volume rendering and surface rendering, are not well-suited for the tasks of 3-D vascular visualization and computer aided vascular surgery. Although the techniques of volume rendering can generate global vascular images, it is difficult to correctly infer the topology of vascular trees from the visualization, since there is no geometry representation for vascular structures. The surface rendering based techniques are prone to producing annoying visual artifacts due to the discrete

nature of radiological data. All these issues associated with conventional volume visualization can be solved by reconstructing the vascular geometry.

In fact, the accurate reconstruction of vessel geometry can not only help to achieve high visual quality of the vessel structures, but it is also essential in CADS. In addition, the geometric representation of vascular structures can also be very useful in the stage of vessel analysis, such as the visual exploration, the quantitative analysis, the interactive modification of the vessel geometry, virtual angioscopy, and so on **(Preim and Oeltze, 2007)**. Although techniques of explicit reconstruction of vascular structures can achieve a certain level of smoothness at a relatively fast speed, they are model based, and in general cannot provide an accurate approximation to the actual vascular objects. In addition, the underlying shape is expressed as polygonal meshes or in parametric forms, making it difficult to implement ramification of branching **(Bloomenthal, 1995)**. On the other hand, implicitly represented geometric objects are inherently easy to combine in various ways, which make it simple to implement ramification. However, current implicit modelling techniques for the reconstruction of vasculatures still cannot achieve a satisfactory result that meets the requirements of both accuracy and smoothness. The reconstructed vascular geometry based on convolution surfaces **(Oeltze and Preim, 2005)** is only a morphological approximation, and the quality of the resulting surfaces based on MPUIs **(Schumann et al., 2007; Schumann et al., 2008)** is generally poor and has to be improved using an additional remeshing step. In consequence, further research is therefore required to investigate implicit modelling techniques for the accurate and smooth reconstruction of vascular structures.

The major drawback of implicit surface is that it consumes greatly high processing power. The implicitly represented geometric objects require a relatively high computational cost to display, since an implicit function does not represent a surface explicitly. Hence, it would be sensible to exploit the advantages of GPU computing to achieve high performance of implicit modelling techniques for the reconstruction of vascular structures.

Chapter 3
Segmentation of Medical Data

3.1 Introduction

Segmentation plays an important role in medical image processing. The technique of segmentation is to identify the region of interest from initial datasets, which is usually an essential preliminary stage for many clinical applications, such as medical diagnosis, minimally-invasive surgery, and surgery planning. In this research, the segmentation of vessel trees from the medical datasets is crucial for the geometry reconstruction and visualization of vasculatures. So far, a great many techniques have been proposed for various image segmentation problems, ranging from threshold based methods, to pattern recognition based methods, and to deformable model based methods **(Ma et al., 2009)**. Among all these methods, the active contour method **(Kass et al., 1988)** is undoubtedly one of the most attractive approaches in the past two decades. The basic idea of active contour is to evolve the initial curve/surface to the boundaries of target objects driven by the combination of internal forces determined by the geometry of the evolving curve/surface and the external forces induced from the image **(Zhang et al., 2008)**.

Depending on the way the underlying curved contour is represented, the deformable active contour can be implemented either parametrically or implicitly **(Ma et al., 2009)**. In parametric models, active contours are explicitly represented in parametric forms

in a Lagrangian framework **(Li et al., 2005)**. On the other hand, when active contours are implemented implicitly, they are represented as a level-set **(Osher and Sethian, 1988)** function which is embedded in higher dimensional spaces and evolves according to the Partial Differential Equation (PDE) in an Eulerian framework **(Li et al., 2005)**. In general, the level-set model has several advantages over the parametric model **(Zhang et al., 2008)**. First and foremost, contours represented by level-set functions do not require any reparameterization, which is a very tough problem required to be solved for parametric model **(Antiga, 2002)**. Second, it is easy to perform shape operation and handle topological changes during evolution **(Li et al., 2005)**. Moreover, the numerical scheme is efficient and easily adapted to solve any dimensional problems **(Zhang et al., 2008)**.

Due to the remarkable advantages of implicit modelling, a large number of level-set based active contour models have been developed for medical image segmentation. These models can be categorized into three major classes: edge-based models, region-based models **(Li et al., 2007)**, and hybrid model **(Zhang et al., 2008)**. For edge-based models [e.g. geodesic active contour model **(Caselles et al., 1993; Caselles et al., 1997; Malladi et al., 1995)**], image gradient is utilized to stop evolving contours on the object boundaries **(Li et al., 2007)**. This is quite suitable for the segmentation of images with well-defined target boundaries. However, when the image contains weak parts, the active contour is likely to pass through weak target boundary and converge to an incorret result **(Zhang et al., 2008)**. On the other hand, region-based models [e.g. Chan-Vese model **(Chan and Vese, 2001)**] can solve the leakage problem using region information of the target boundary for segmentation. Unfortunately, due to the lack of boundary information, this approach may lead to inaccurate

segmentation of images with relatively well-defined target boundaries. In addition, the global nature of the region threshold is not appropriate for images with complex background **(Zhang et al., 2008)**.

In order to solve the above mentioned problems, a hybrid level-set model for medical image segmentation has been proposed in Zhang et al.'s work **(Zhang et al., 2008)**. This method integrates both region and boundary information. The boundary information can help to detect the precise location of the target object and the region information can help to prevent the boundary leakage problem. However, in this hybrid method, a preset global threshold indicating the lower bound of target object is used to specify the term concerning regional information, which is not quite appropriate, especially for medical images with huge intensity inhomogeneity. In fact, intensity inhomogeneity occurs in many medical images, such as CT or MRI **(Li et al., 2007)**. Overcoming the difficulty due to image inhomogeneity is crucial for the segmentation of vessel structures.

In this chapter, a Local Binary Fitting (LBF) energy with a kernel function **(Li et al., 2007)** is introduced into the hybrid level-set framework to accommodate. In this proposed method, the local intensity information is firstly embedded into a region-based contour model, and then incorporated into the level-set formulation of the geodesic active contour model. Compared with the global threshold based method, the use of local binary fitting energy enables the extraction of the local image information, which is essential for the segmentation of inhomogeneous images. Experimental results on 2-D and 3-D medical images will be presented to demonstrate the strengths of using locally specified dynamic thresholds in our level-set method.

3.2 Related Work

3.2.1 Geodesic Active Contour Model

The classic active contour (i.e. snakes) **(Kass et al., 1988)** is a edge–based segmentation model whose essential idea is to find a curve with minimum length that best fits the target boundary **(Yan and Kassim, 2006)**. Usually, an edge-detector that depends on the gradient of the image is utilized to stop the evolving curve on the boundary of target object **(Chan and Vese, 2001)**.

Let $C(s):[0,1]\to R^2$ be a closed planar curve, and let $I:\Omega\to R$, $\Omega\subset R^2$ be a given target image. The snake model can be described as the energy functional **(Kass et al., 1988)**

$$E(C)=\int_0^1 \alpha\left|C'(s)\right|^2+\beta\left|C''(s)\right|^2 ds-\lambda\int_0^1\left|\nabla I(C(s))\right|^2 ds \qquad (3.1)$$

Here, α, β, and λ are positive parameters. The first two terms are regarded as the internal energy to control the smoothness of the evolving curve, while the third term is the external energy that is responsible for driving the curve towards the target boundary in the image **(Caselles et al., 1997)**. By minimizing the energy functional (3.1), the curve can be located at the points of maxima $\left|\nabla I\right|$ **(Chan and Vese, 2001)**.

Starting from the classic snakes, Caselles et al. introduced the geodesic active contour model **(Caselles et al., 1997)**. The basic idea of this model is to find a minimum-length curve measured in a Riemannian space, which can be achieved by minimising the energy functional defined as **(Caselles et al., 1997)**

$$E(C) = \int_0^1 g(|\nabla I(C(s))|)|C'(s)|ds \tag{3.2}$$

where $g:[0,\infty] \to R^+$ is the decreasing function, such as $g(h) = 1/(1+ch^2)$ with c controlling the scope, and $g(|\nabla I|)$ is regarded as a general edge-detector. Note that, the energy functional of (3.2) is minimized when $g(|\nabla I(C(s))|)$ vanishes. That is, by minimizing the energy functional (3.2), we can achieve the target curve at the location of object boundary.

In the level set method, curve C is represented implicitly by $\{(x,y) \mid \phi(x,y) = 0\}$, and the evolution of the curve is given by the zero-level curve at time t of the function $\phi(t,x,y)$ **(Chan and Vese, 2001)** . Evolving the curve C in normal direction with speed V is equivalent to solve the Partial Differential Equation (PDE) **(Osher and Sethian, 1988)**

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| V \tag{3.3}$$

where $\phi(0,x,y) = \phi_0(x,y)$ defines the initial contour. Specially, the mean curvature of the curve can be regarded as the emotion speed, i.e. $V = div(\nabla\phi/|\nabla\phi|)$, where div is short for divergence operator. Then the equation becomes **(Chan and Vese, 2001)**

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| div\left(\frac{\nabla \phi}{|\nabla \phi|}\right) \tag{3.4}$$

For the geodesic active contour model, the associated curve evolution PDE in the level-set formulation can be expressed by **(Caselles et al., 1997)**

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| div\left(g\left(|\nabla I|\right) \frac{\nabla \phi}{|\nabla \phi|} \right) \tag{3.5}$$

3.2.2 Piecewise Constant Model

The piecewise constant model was proposed by Chan and Vese **(Chan and Vese, 2001)** to utilize the region information of the target object for segmentation. The energy functional of this model is defined as **(Chan and Vese, 2001)**

$$\begin{aligned} E(c_1, c_2, \phi) = {} & \lambda_1 \int_\Omega |I - c_1|^2 H(\phi) d\Omega \\ & + \lambda_2 \int_\Omega |I - c_2|^2 (1 - H(\phi)) d\Omega \\ & + \mu \int_\Omega |\nabla H(\phi)| d\Omega + \nu \int_\Omega H(\phi) d\Omega \end{aligned} \tag{3.6}$$

Here, λ_1 , λ_2 , μ , and ν are positive parameters. c_1 , c_2 are two constants approximating the image intensity inside and outside the active contour respectively, which are defined in the following way **(Chan and Vese, 2001)**

$$\begin{cases} c_1(\phi) = \dfrac{\int_\Omega H(\phi) I \, d\Omega}{\int_\Omega H(\phi) \, d\Omega} \\ c_2(\phi) = \dfrac{\int_\Omega \left(1 - H(\phi)\right) I \, d\Omega}{\int_\Omega \left(1 - H(\phi)\right) d\Omega}, \end{cases} \tag{3.7}$$

$H(\phi)$ is the Heaviside function defined as

$$H(\phi)=\begin{cases}1, if\ \phi \geq 0\\ 0, if\ \phi < 0\end{cases} \tag{3.8}$$

As can be seen from Eq. (3.6), the first two terms respectively measure the fitting energy inside and outside the active contour, whereas the last two terms regularize the length of the active contour and the area of the region inside the active contour.

In order to minimize the energy functional $E(c_1, c_2, \phi)$ with respect to ϕ, the evolution PDE in the level-set formulation for ϕ is deduced from associated Euler-Lagrange equation **(Chan and Vese, 2001)**

$$\frac{\partial \phi}{\partial t}=\delta(\phi)\left[-\lambda_1(I-c_1)^2+\lambda_2(I-c_2)^2+\mu div\left(\frac{\nabla\phi}{|\nabla\phi|}\right)-\nu\right] \tag{3.9}$$

where $\delta(\phi)$ is the Dirac function defined as the one-dimensional Heaviside function $\delta(\phi)=dH(\phi)/d\phi$.

3.2.3 Hybrid Level-set Model

This model combines the information of region and object boundary for segmentation. The proposed energy functional to be minimised is defined as **(Zhang et al., 2008)**

$$E^H(\phi)=-\alpha\int_\Omega(I-\mu)H(\phi)d\Omega+\beta\int_\Omega g(|\nabla I|)|\nabla H(\phi)|d\Omega \tag{3.10}$$

where α and β are the preset weights, and μ is the preset parameter defining the lower bound of the gray-level of the target object. The first term of Eq. (3.10) utilizes the region information for segmentation, which pushes the contours to enclose the regions with gray-levels greater than μ. The second term is equal to the geodesic active contour functional, which forces the contours to attach to the areas with high image gradients. The evolution PDE in the level-set formulation derived from the functional (3.10) can be defined as **(Zhang et al., 2008)**

$$\frac{\partial \phi}{\partial t} = \delta(\phi)\left[\alpha(I-\mu) + \beta div\left(g(|\nabla I|)\frac{\nabla \phi}{|\nabla \phi|} \right) \right] \tag{3.11}$$

Compared with other level-set based image segmentation techniques, the hybrid level-set model can achieve relatively accurate and robust segmentation results as it integrates both region and boundary information into the model. However, its performance decreases dramatically with the increase of intensity inhomogeneity in medical images due to the following reasons. Firstly, it is not an easy task to choose the appropriate global threshold μ to indicate the lower gray-level boundary of target object. The global parameter should not be either too small, which may lead to the segmentation of unwanted object, or too large, which may result in leakage of the target object. Moreover, for some inhomogeneous images, it is even impossible to define the global threshold μ for the lower boundary of the target object. This is because a certain μ may be appropriate to the gray-level of target object in some regions of the image, but it may correspond to the gray-level of the background in some other regions of the same image. Therefore, in order to tackle this problem, a local binary fitting energy with a kernel function **(Li et al., 2007)** is introduced into the hybrid level-set framework instead of using a preset global threshold to indicate the

lower bound of target object for the term referring to region information. Compared with the global threshold based method, the use of local binary fitting energy is able to extract more accurate local image information, which is essential for the segmentation of inhomogeneous images.

3.3 Local Hybrid Level-set Model

3.3.1 Local Binary Fitting Energy

LBF energy is part of the energy functional of the local binary fitting model **(Li et al., 2007)**, which utilizes the local intensity information efficiently to segment images with intensity inhomogeneity. Consider a given image $I:\Omega \to R$, in which $\Omega \subset R^n$ ($n=2$ or 3) is the image domain, and let $\phi:\Omega \to R$ be a contour in the image domain. The LBF energy for each point $\mathbf{u} \in \Omega$ can be defined as **(Li et al., 2007)**

$$E_{\mathbf{u}}^{LBF}(\phi, f_1(\mathbf{u}), f_2(\mathbf{u})) = \lambda_1 \int_{\Omega} K_\sigma(\mathbf{u}-\mathbf{v}) \left| I - f_1(\mathbf{u}) \right|^2 H(\phi(\mathbf{v}))d\mathbf{v}$$

$$+\lambda_2 \int_{\Omega} K_\sigma(\mathbf{u}-\mathbf{v}) \left| I - f_2(\mathbf{u}) \right|^2 \left(1-H(\phi(\mathbf{v}))\right) d\mathbf{v} \qquad (3.12)$$

where $\mathbf{v} \in \Omega$, and K_σ is the Gaussian kernel function with a localization property, such as $K_\sigma(\mathbf{u}) = \frac{1}{2\pi\sigma^2} e^{-|\mathbf{u}|^2/2\sigma^2}$ with a scale parameter σ. $f_1(\mathbf{u})$ and $f_2(\mathbf{u})$ are two fitting functions that fit the local intensities near the point $\mathbf{u}$, given by **(Li et al., 2007)**

$$f_1(\mathbf{u}) = \frac{K_\sigma(\mathbf{u}) * \left[H(\phi(\mathbf{u}))I(\mathbf{u})\right]}{K_\sigma(\mathbf{u}) * H(\phi(\mathbf{u}))} \qquad (3.13)$$

and

$$f_2(\mathbf{u}) = \frac{K_\sigma(\mathbf{u}) * \left[(1 - H(\phi(\mathbf{u})))I(\mathbf{u})\right]}{K_\sigma(\mathbf{u}) * (1 - H(\phi(\mathbf{u})))} \tag{3.14}$$

Due to the localization property of the kernel function $K_\sigma(\mathbf{u}-\mathbf{v})$, the contribution of the image intensity at point $\mathbf{v}$ to the LBF energy decreases to zero as the point $\mathbf{v}$ moves away from the centre point $\mathbf{u}$. Therefore, the LBF energy is dominated by the intensities of points $\mathbf{v}$ around $\mathbf{u}$. This localization property plays a key role in segmenting images with intensity inhomogeneity **(Wang et al., 2009)**.

Each $E_{\mathbf{u}}^{LBF}$ over all the centre points $\mathbf{u}$ in the image domain is integrated together to compose the entire local fitting energy functional **(Li et al., 2007)**

$$E^{LBF}(\phi, f_1, f_2) = \int_\Omega E_{\mathbf{u}}^{LBF}(\phi, f_1(\mathbf{u}), f_2(\mathbf{u}))d\mathbf{u} \tag{3.15}$$

3.3.2 Level-set Formulation of the Localized Hybrid Model

In the proposed localized hybrid model, the LBF energy is introduced as the term to represent region information instead of a preset global threshold representing region information. The proposed energy functional to be minimised is defined as

$$E^{LH}(\phi, f_1, f_2) = \alpha \int_\Omega E_{\mathbf{u}}^{LBF}(\phi, f_1(\mathbf{u}), f_2(\mathbf{u}))d\mathbf{u} + \beta \int_\Omega g(|\nabla I|)|\nabla H(\phi)|d\mathbf{u} \tag{3.16}$$

where α and β are the preset weights to balance the two terms. The first term is the LBF energy functional (refer to Section 3.3.1) for the entire image, which is regarded as the region term in our improved hybrid level-set model. The second term is equal to the geodesic active contour functional, which is just the same as the

second term of the original hybrid model. In addition to keeping the advantages of original hybrid model, the localized hybrid model is able to extract more accurate local image information by taking the LBF energy into account, which is essential for segmenting images with intensity inhomogeneity. The level-set formulation of the localized hybrid model can be expanded as

$$\begin{aligned} & E^{LH}(\phi, f_1, f_2) \\ & = \alpha\Big[\lambda_1\int_\Omega\Big(\int_\Omega K_\sigma(\mathbf{u}-\mathbf{v})\left|I - f_1(\mathbf{u})\right|^2 H(\phi(\mathbf{v}))d\mathbf{v}\Big)d\mathbf{u} \\ & +\lambda_2\int_\Omega\Big(\int_\Omega K_\sigma(\mathbf{u}-\mathbf{v})\left|I - f_2(\mathbf{u})\right|^2 (1-H(\phi(\mathbf{v})))d\mathbf{v}\Big)d\mathbf{u}\Big] \\ & +\beta\int_\Omega g(|\nabla I|)\left|\nabla H(\phi)\right|d\mathbf{u} \end{aligned} \tag{3.17}$$

By keeping $f_1(\mathbf{u})$ and $f_2(\mathbf{u})$ fixed, the associated PDE can be derived from the gradient descent flow **(Li et al., 2007)** applied to functional (3.17)

$$\frac{\partial\phi}{\partial t} = \alpha\delta(\phi)(-\lambda_1 e_1 + \lambda_2 e_2) + \beta\delta(\phi)div\left(g(|\nabla I|)\frac{\nabla\phi}{|\nabla\phi|}\right) \tag{3.18}$$

where e_1 and e_2 are defined as below **(Li et al., 2007)**

$$e_1(\mathbf{u}) = \int_\Omega K_\sigma(\mathbf{u}-\mathbf{v})\left|I - f_1(\mathbf{u})\right|^2 d\mathbf{v} \tag{3.19}$$

and

$$e_2(\mathbf{u}) = \int_{\Omega} K_{\sigma}(\mathbf{u}-\mathbf{v})\left|I - f_2(\mathbf{u})\right|^2 d\mathbf{v} \tag{3.20}$$

3.3.3 Implementation

The original Heaviside function is discontinuous and is unable to achieve smooth transition at the boundary. In practice, this problem is generally solved using a kind of smooth Heaviside function, which is usually called smooth step function or smooth unit step function. Depending on the problem of applications, different smooth step functions have been introduced to approximate smoothly the Heaviside function **(Li et al., 2006)**. More recently, a piecewise polynomial smooth unit step functions is developed and used for geometric shape design, which is regarded as a kind of generalized Heaviside unit step function **(Li and Tian, 2011)**.

In our implementation, we adopt the smooth function $H_{\varepsilon}(s)$ introduced in **(Chan and Vese, 2001)**

$$H_{\varepsilon}(s) = \frac{1}{2}\left(1 + \frac{2}{\pi}\arctan(\frac{s}{\varepsilon})\right) \tag{3.21}$$

and the corresponding Dirac function $\delta_{\varepsilon}(s)$ is defined as **(Chan and Vese, 2001)**

$$\delta_{\varepsilon}(s) = H_{\varepsilon}^{'}(s) = \frac{1}{\pi}\frac{\varepsilon}{\varepsilon^2 + s^2} \tag{3.22}$$

Then, H in Eq. (3.13) and (3.14) is replaced with H_{ε}, and δ in Eq. (3.18) is replaced with δ_{ε}. In addition, in order to avoid problems of developing shocks,

sharp or flat shapes during evolution, it is essential to reinitialize ϕ to be a signed distance function (SDF) **(Osher and Fedkiw, 2002)**. And if ϕ is a SDF, then $|\nabla\phi| = 1$. Therefore, Eq. (3.18) can be written as:

$$\begin{aligned}\frac{\partial\phi}{\partial t} &= \alpha\delta_\varepsilon(\phi)(-\lambda_1 e_1 + \lambda_2 e_2) + \beta\delta_\varepsilon(\phi)div\left(g(|\nabla I|)\nabla\phi\right) \\ &= \alpha E^{lbf}(\phi) + \beta\delta_\varepsilon(\phi)div\left(g(|\nabla I|)\nabla\phi\right)\end{aligned} \tag{3.23}$$

in which $\delta_\varepsilon(\phi)(-\lambda_1 e_1 + \lambda_2 e_2)$ is represented as $E^{lbf}(\phi)$ for short.

In the proposed model, the temporal partial derivative $\partial\phi/\partial t$ is approximated by the forward difference scheme. Then, the approximation of (3.23) can be expressed as **(Li et al., 2005)**

$$\frac{\phi^{(k+1)} - \phi^{(k)}}{\Delta t} = \alpha E^{lbf}(\phi^{(k)}) + \beta\delta_\varepsilon(\phi^{(k)})div\left(g(|\nabla I|)\nabla\phi^{(k)}\right) \tag{3.24}$$

where $\phi^{(k+1)}$ and $\phi^{(k)}$ denote the embedding function ϕ in the $(k+1)^{th}$ and k^{th} iterations respectively, and Δt is the predefined time step. The difference equation (3.24) can also be rewritten as the following iteration

$$\phi^{(k+1)} = \phi^{(k)} + \Delta t\left[\alpha E^{lbf}(\phi^{(k)}) + \beta\delta_\varepsilon(\phi^{(k)})div\left(g(|\nabla I|)\nabla\phi^{(k)}\right)\right] \tag{3.25}$$

Finally, the principal steps in the original hybrid model **(Zhang et al., 2008)** is adopted to update from $\phi^{(k)}$ to $\phi^{(k+1)}$:

1. Reinitialize $\phi^{(k)}$ to be SDF;
2. Compute the LBF energy functional $E^{lbf}(\phi^{(k)})$;
3. Update $\phi^{(k)}$ to $\phi^{(k)}{}'$ using $\phi^{(k)}{}' = \phi^{(k)} + \Delta t \alpha E^{lbf}(\phi^{(k)})$;
4. Reinitialize $\phi^{(k)}{}'$ to be SDF;
5. Update $\phi^{(k)}{}'$ to obtain $\phi^{(k+1)}$ via $\phi^{(k+1)} = \phi^{(k)}{}' + \Delta t \beta \delta_{\varepsilon}(\phi^{(k)}) div\left(g(|\nabla I|)\nabla \phi^{(k)}\right)$.

3.4 Experimental Results and Discussion

The proposed model has been applied to synthetic and real 2-D and 3-D medical images to demonstrate its effectiveness in segmenting intensity-inhomogeneity medical images. In all these experiments, the same parameters are used, namely, $\Delta t = 4.0$, $\alpha = 0.001$, $\beta = 0.5$, $\sigma = 2.0$, $\varepsilon = 1.0$, and $\lambda_1 = \lambda_2 = 1.0$.

3.4.1 Results of our Localized Hybrid Model

Figure 3.1 shows the segmentation results of the proposed approach for a synthetic image. For this image, two initial contours were placed into two target objects as in **(a)**. As can be seen from **(c)**, our localized hybrid level-set model achieves a quite satisfactory segmentation result. And **(d)** is the Level-set function ϕ coresponding to **(c)**.

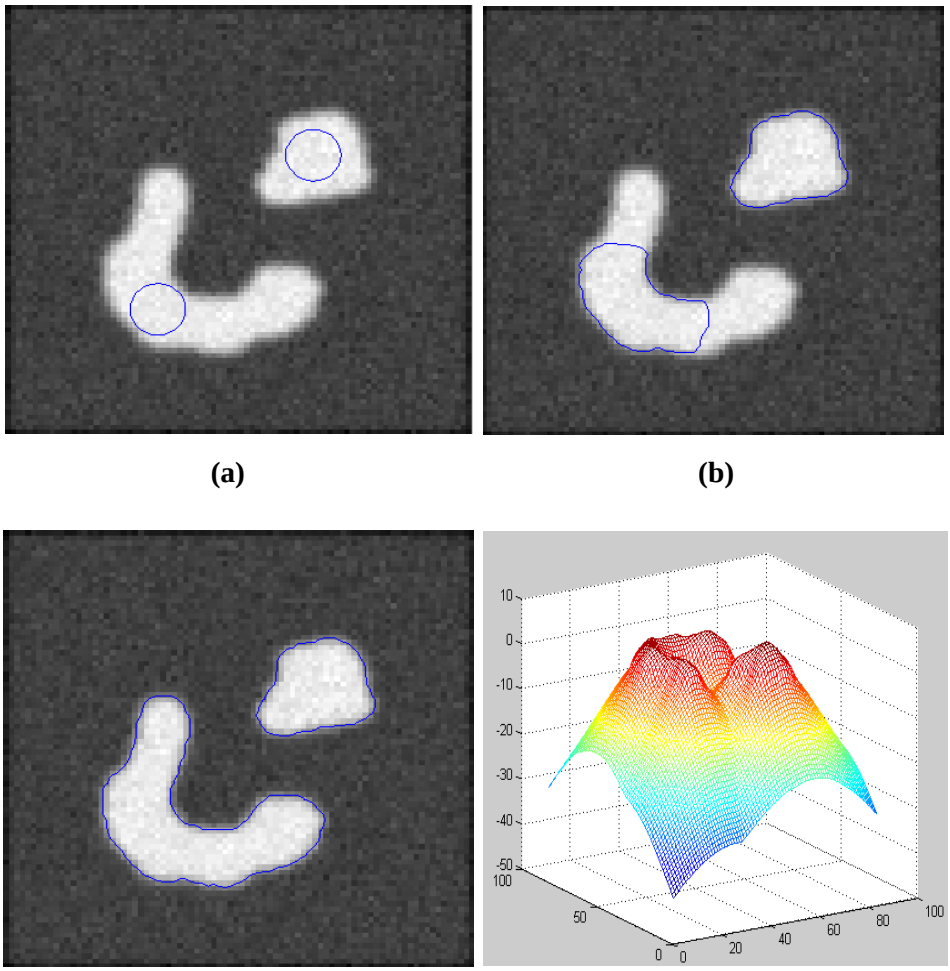
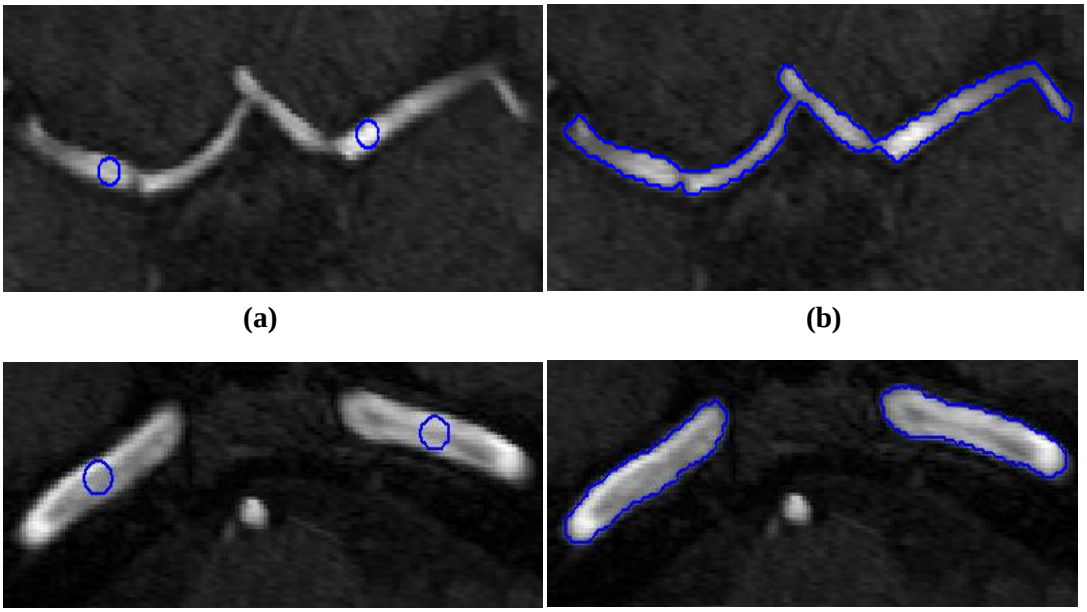


Figure 3.1: *Application to synthetic image*. **(a)** *Initial Contours,* **(b)** *10 iterations,* **(c)** *30 iterations,***(d)** *Level-set function* ϕ *coresponding to (c).*

Figure 3.2 shows the segmentation results for some 2-D slices of Magnetic Resonance Angiography (MRA) images of the human brain. As mentioned before, Magnetic Resonance (MR) images are typically intensity inhomogeneous. As can be seen from the lower row of **Figure 3.2**, the intensity of the middle region is quite lower than that of other regions of the vessel. Owning to the introduction of LBF term, the proposed method can successfully depict the vessels in these images.



(a) **(b)**

(c) **(d)**

Figure 3.2 : *Application to vessel segmentation of 2-D medical images.* **(a)** *and* **(c)** *are initial contours,* **(b)** *and* **(d)** *are final contours.*

The proposed localized hybrid model has also been applied to more than ten 3-D medical datasets for the segmentation of vasculatures. The datasets are supplied by Intelligent Bioinformatics Systems Division, Institute of Automation, the Chinese Academy of Sciences, in the format of DICOM (Digital Imaging and Communications in Medicine). In the following, we present some typical segmentation results as well as the Maximum Intensity Projections (MIPs) of the corresponding original datasets. MIPs are very useful to evaluate current 3-D angiography images since the overall shapes and paths of the vessel can be clearly visualized **(Orkisz et al., 1997).**

The first example is the segmentation of the common iliac artery for 3-D MRA images with a resolution of $384 \times 384 \times 72$ and spacing of $1.0\text{mm} \times 1.0\text{mm} \times 1.5\text{mm}$. The MIP of this dataset is shown in **Figure 3.3**, while the segmentation result using our localized hybrid technique is shown in **Figure 3.4**. Although the raw dataset has

great intensity inhomogeneity, the whole iliac artery is still successfully extracted. The second example is the segmentation of the right carotid artery for 3-D Digital Subtraction Angiography (DSA) images with a resolution of $240 \times 120 \times 160$ and spacing of $0.439\text{mm} \times 0.439\text{mm} \times 0.625\text{mm}$. The MIP of this dataset is shown in **Figure 3.5**, while the segmentation results using our localized hybrid technique are showed in **Figure 3.6**. The third example is the segmentation of the left carotid artery for 3-D DSA images with a resolution of $240 \times 120 \times 160$ and spacing of $0.439\text{mm} \times 0.439\text{mm} \times 0.625\text{mm}$. **Figure 3.7** is the MIP, and **Figure 3.8** is the segmentation result using our localized hybrid technique. As can be seen from the figures, our method can successfully extract the carotid arteries from DSA datasets.

The fourth example is the segmentation of cerebral vasculatures for 3-D MRA images with a resolution of $352 \times 448 \times 114$ and spacing of $0.49\text{mm} \times 0.49\text{mm} \times 0.80\text{mm}$. The MIP of this dataset is shown in **Figure 3.9,** while the segmentation results using our localized hybrid technique are showed in **Figure 3.10**. As can be seen from the images rendered from different viewing angles, the whole complex cerebral vasculature is successfully extracted from the 3-D MRA dataset with great intensity inhomogeneity.

Figure 3.3: *Maximum Intensity Projection (MIP) of the MRA dataset for common iliac artery.*

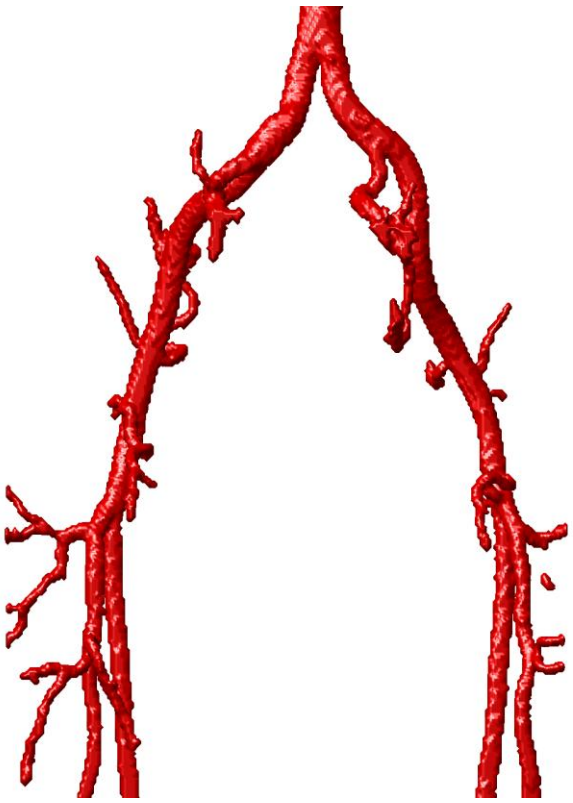


Figure 3.4: *Segmentation result of common iliac artery from the MRA dataset using our localized hybrid technique.*

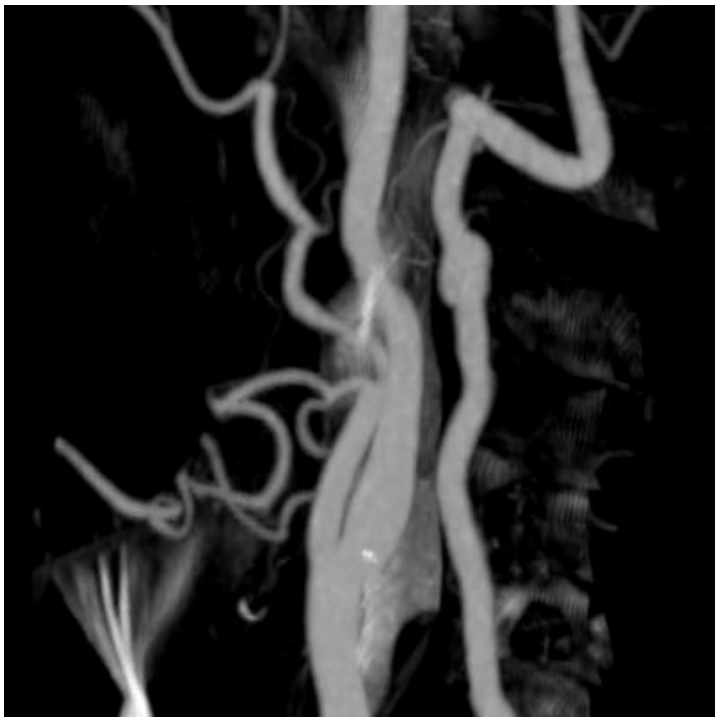


Figure 3.5: *MIP of the DSA dataset for right carotid artery.*

Figure 3.6: *Segmentation result of right carotid artery from the DSA dataset using our localized hybrid technique.*

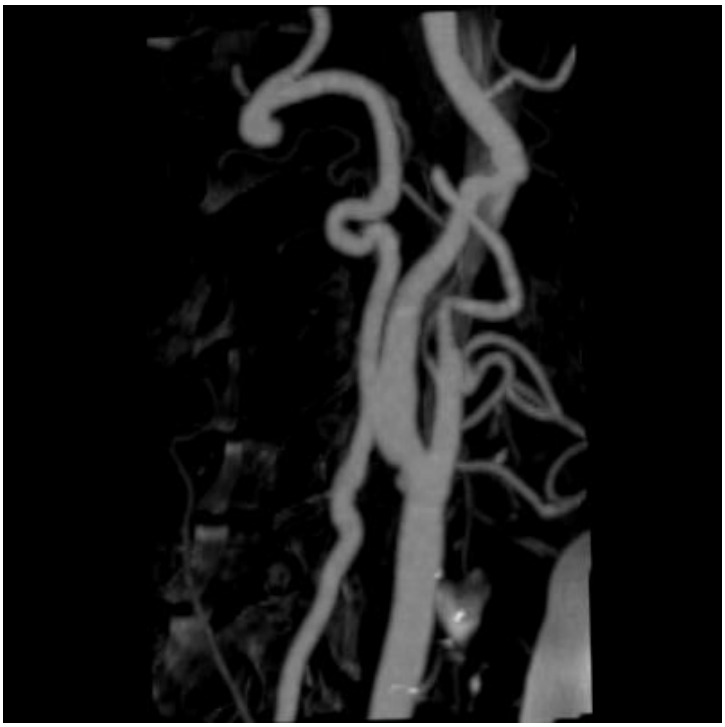


Figure 3.7: *MIP of the DSA dataset for left carotid artery.*

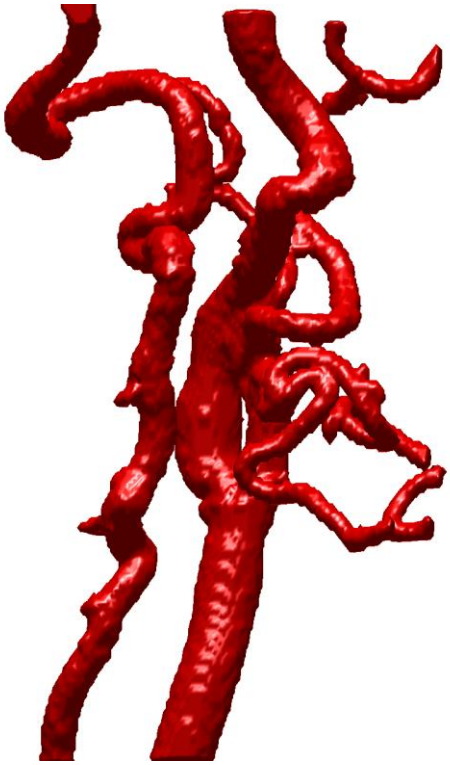
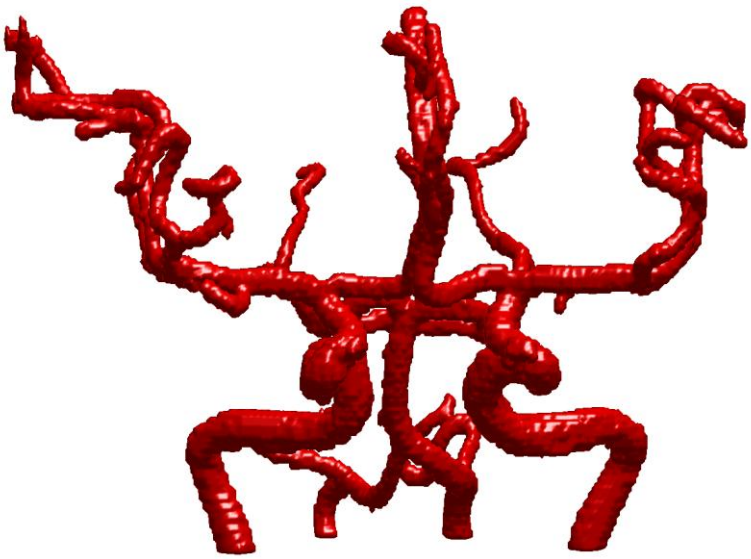


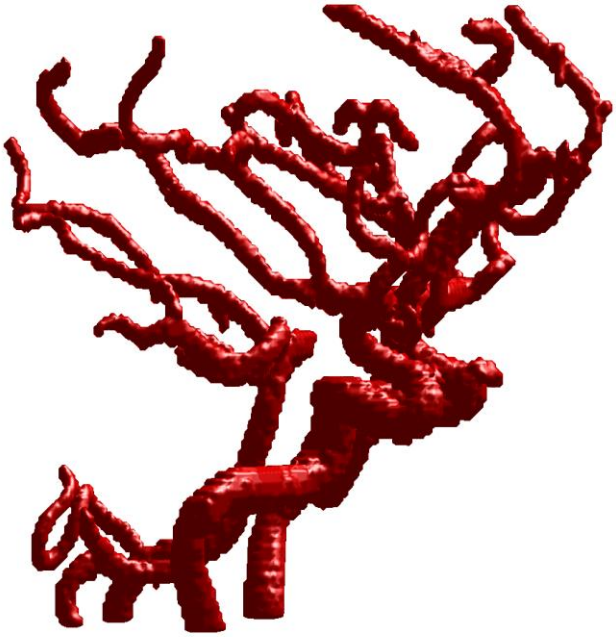
Figure 3.8: *Segmentation result of left carotid artery from the DSA dataset using our localized hybrid technique.*

Figure 3.9: *MIP of the cerebral MRA dataset.*



(a)

(b)

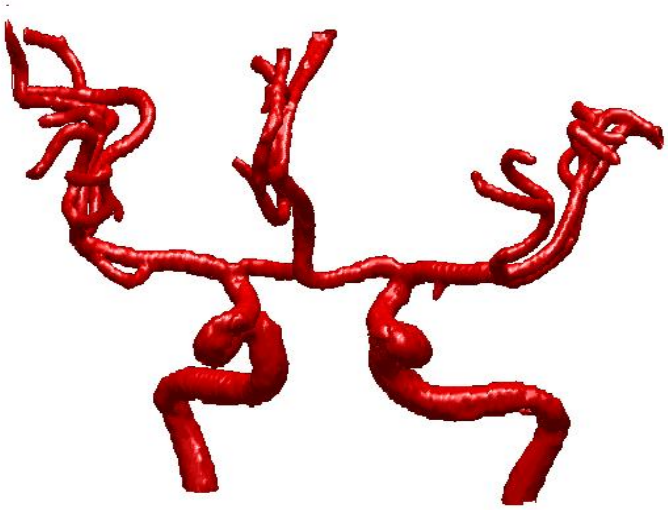


(c)

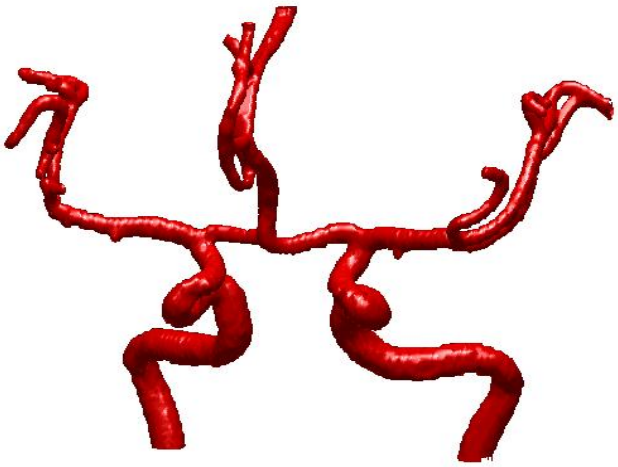
Figure 3.10 : *Segmentation results of cerebral MRA dataset using our localized hybrid technique.* **(a) (b)** *and* **(c)** *are the visualization of segmentation results rendered from different viewing angles.*

3.4.2 Comparison with the Original Hybrid Level-set Model

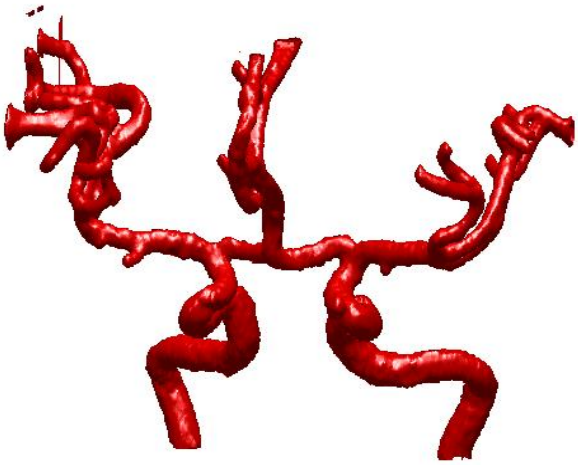
In this section, we investigate why the original hybrid model may fail in the segmentation of real medical data with intensity inhomogeneity whereas our localized model works. According to our prior knowledge, the intensity (i.e. gray level) of most vessels in the fourth dataset (i.e. cerebral MRA images) is larger than 200. However, the intensity of some small vessels and their branches can be varied, for some of them can even be lower than 100. Therefore, for the original model, it is quite difficult to predefine the global threshold μ indicating the lower bound of gray level of target vessel. If the global threshold μ were set to be 200, it would be unable to extract the small vasculatures with intensity smaller than 200; if μ were set to be 100, the segmentation result could include some irrelevant objects, which would lead to inaccurate extraction of vasculatures.



(a)



(b)

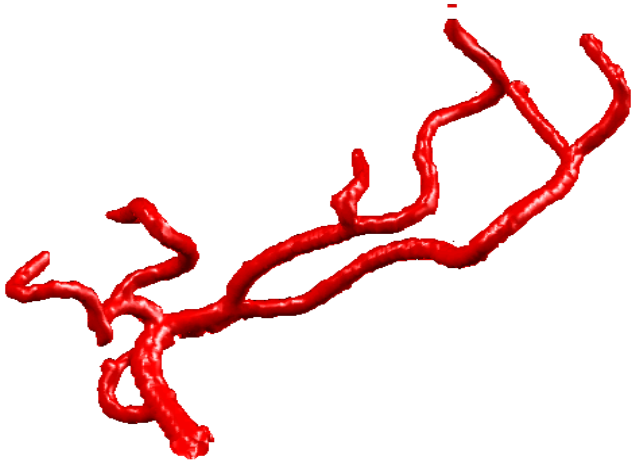


(c)

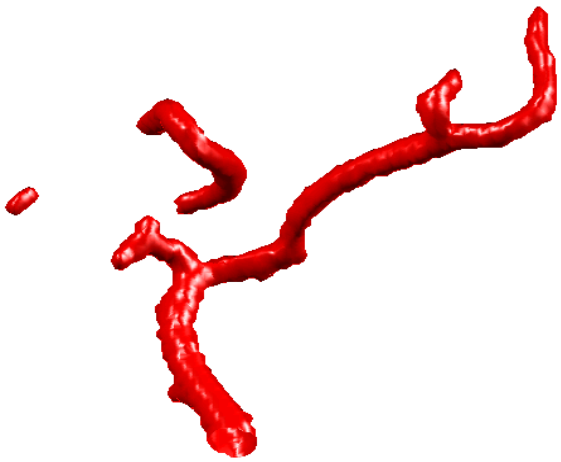
Figure 3.11: *Comparison of our localized model with the original model.* **(a)** *Part of segmentation results using our localized model with 100 iterations.* **(b)** *Part of segmentation results using the original model with 100 iterations, and* $\mu = 200$. **(c)** *Part of segmentation results using the original model with 100 iterations, and* $\mu = 100$.

For the purpose of comparison, we use the same set of parameter values (see the beginning of this section) for the common parameters of the original model and our localized model. In addition, we also set the same seed contours for both models. As

can be seen from **Figure 3.11**, our localized hybrid model can achieve a more satisfactory segmentation result of the vascular structures than the original model with different μ s.



(a)

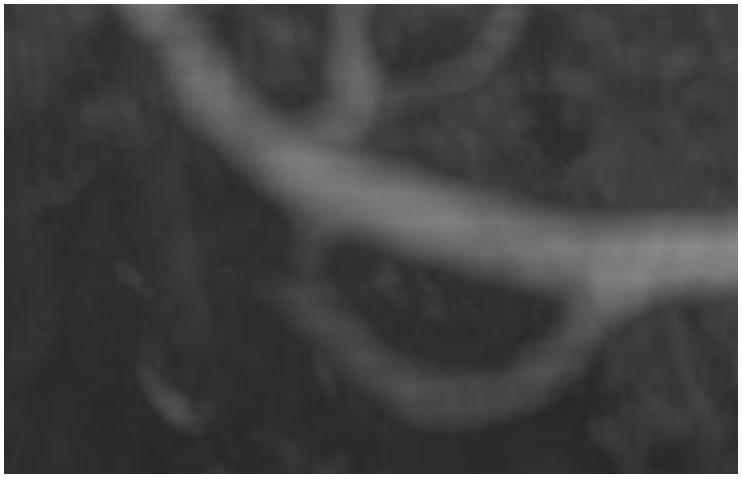


(b)

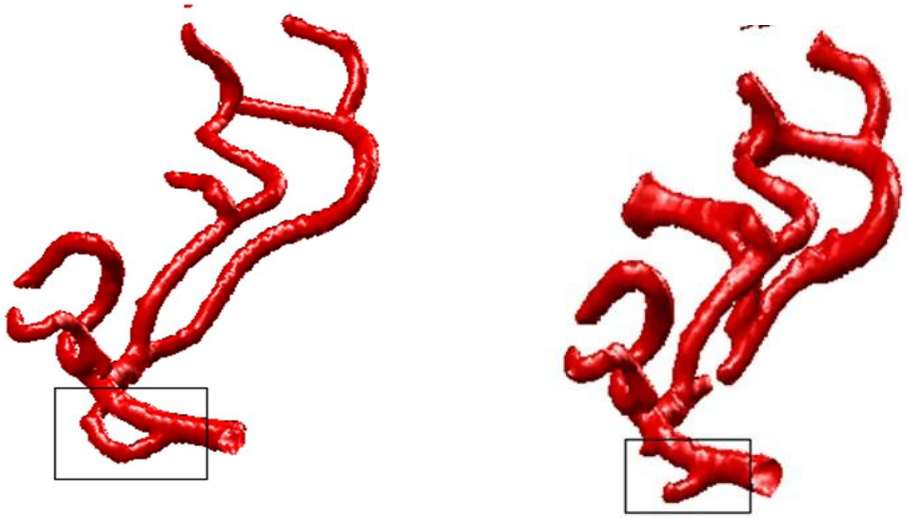
(c)

Figure 3.12 : *A detailed look at the left part of segmented vessel trees.* **(a)** *Left part of Figure 3.11(a).* **(b)** *Left part of Figure 3.11 (b).* **(c)** *Left part of Figure 3.11 (c).*

As a further comparison, **Figure 3.12** shows a detailed look at the left part of segmented vessel trees by using our localized hybrid model and the original hybrid model. As can be seen from **Figure 3.12(a)**, it is very clear that our method has the ability to extract small branches with satisfactory results. **Figure 3.12(b)** shows the segmented result of the original model with $\mu = 200$. In this case, it is unable to extract small branches with an intensity less than 200, which just provides us with the unconnected vessel tree. On the other hand, when μ is set as 100, the original hybrid model can produce a more connected vessel tree, but some of the individual branches become much thicker than expected , which can be seen clearly in **Figure 3.12(c)**. In other words, when the value of μ is set too low in the original hybrid model, it may also lead to inaccurate segmentation of vasculatures.



(a)



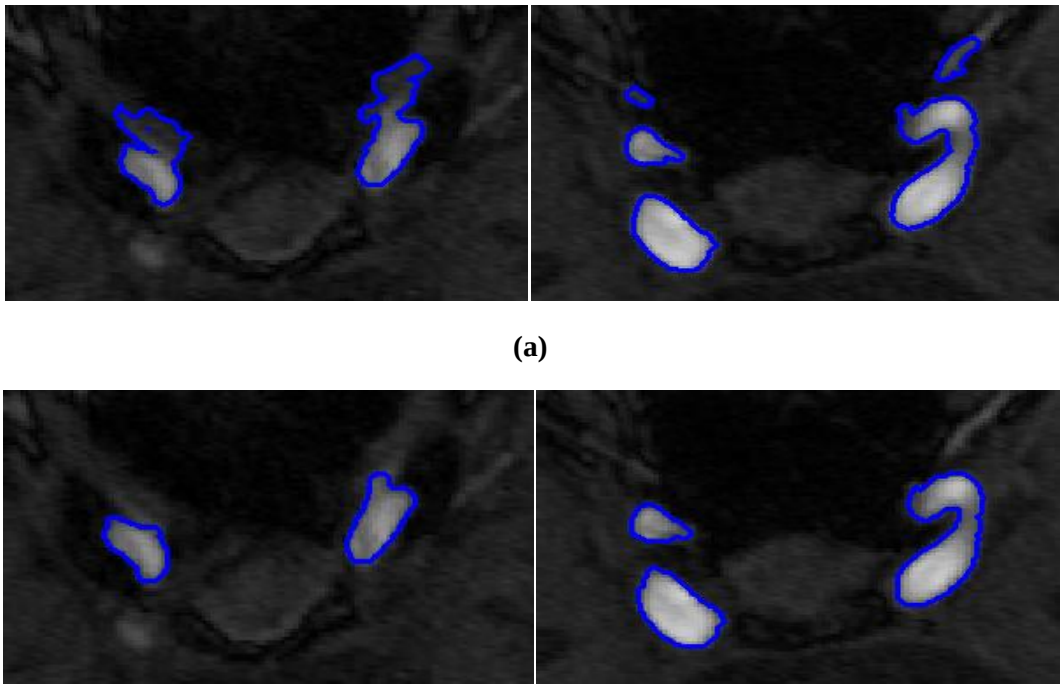
(b) **(c)**

Figure 3.13: *Detailed comparison of the segmentation of thin vessels in low contrast situations marked by the rectangle.* **(a)** *A detailed look at the MIP of the original 3-D dataset.* **(b)** *Our localized hybrid model.* **(c)** *Original model with* $\mu = 100$.

Moreover, the original hybrid model may be still unable to extract thin vessels in low contrast situations even though μ is set to be very low. As can be seen from **Figure 3.13**, **(a)** is a detailed look at the MIP of the 3-D dataset, and the rectangular region in **(b)** indicates the successful extraction of thin vessels with low contrast

using our localized model. **Figure 3.13(c)** shows the segmentation result using the original hybrid model with $\mu = 100$, which still cannot extract thin vessels.

3.4.3 Comparison with the Method Based on LBF Term



(a)

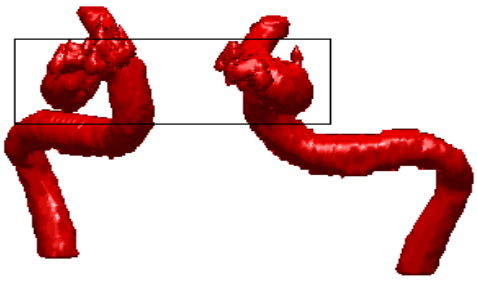
(b)

Figure 3.14: *Comparison of LBF method with our hybrid method for the segmentation of 2-D medical image.* **(a)** *Segmentation result by only using LBF term.* **(b)** *Segmentation result by our hybrid method.*

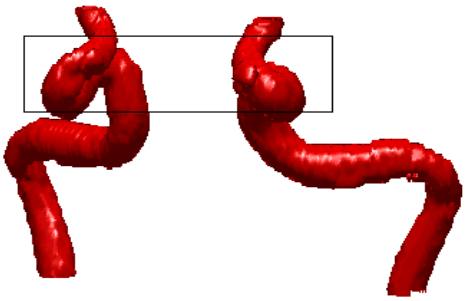
Although the LBF energy term is crucial for the accurate segmentation of medical images with intensity inhomogeneity, it still has a risk of including irrelevant regions when the contrast of gray value between object and background is not sufficient. In this case, using only the LBF energy term may lead to an inaccurate segmentation result, which can be seen in **Figure 3.14(a)**. Therefore, it is essential to combine the boundary information with the LBF energy term for the restriction of

target region. As can be seen from **Figure 3.14(b)**, our localized hybrid method can produce a more satisfactory segmentation result.

As a further comparison, **Figure 3.15** shows the segmentation results of a 3-D medical dataset by using the LBF method and our hybrid method. As can be seen from the rectangular regions of **Figure 3.15(a)**, it is very clear that the method by only using LBF term cannot achieve a satisfactory result, while our hybrid method does [see **Figure 3.15(b)**].



(a) By only using LBF term



(b) By our hybrid method

Figure 3.15: *Comparison of LBF method with our hybrid method for the segmentation of 3-D medical dataset.* **(a)** *By only using LBF term.* **(b)** *By our hybrid method.*

3.5 Summary

In this chapter, a localized hybrid level-set model for medical image segmentation is proposed. In general, the original hybrid level-set based segmentation technique cannot produce a satisfactory segmentation for medical images with high intensity inhomogeneity. To tackle this problem, a local binary fitting energy with a kernel function is introduced into the hybrid level-set framework instead of the preset global threshold for the term referring to the region information. Compared with the global threshold based method, the introduction of the local binary fitting energy into the hybrid model enables the extraction of more accurate local image information, which is essential for the segmentation of inhomogeneous images. Our proposed method has been successfully applied to over ten 3-D medical datasets for the segmentation of vasculatures. In addition, experimental results show that the localized hybrid model has great advantages over the original model in the segmentation of both 2-D and 3-D medical images with intensity inhomogeneity.

Our method can achieve satisfactory segmentation results when compared to the Maximum Intensity Projections (MIPs) of the corresponding original datasets, which is a popular technique to evaluate current 3-D angiography images since the overall shapes and paths of the vessel can be clearly visualized. On the other hand, the quantitative validation of the segmentation accuracy has not been conducted in this research, which is regarded as a substantially more difficult problem, as true surface of the actual vessel is unknown. One suggestion is to compare with the synthetic data where the true surface is known. However, the synthetic data is only a simulation of medical data, and still have some divergence with real angiography

images. Hence, quantitative validation of segmentation method still remains a challenging task.

Chapter 4
Geometry Reconstruction of Vascular Structures

4.1 Introduction

Accurate reconstruction of vessel geometry is an important task in the field of medical data visualization **(Preim and Oeltze, 2007)**. It plays a crucial role in the area of computer-aided diagnosis and computer guided minimally invasive vascular surgery, such as the diagnosis of anomalous growths and stenosis **(Joshi et al., 2008)**, and virtual angioscopy **(Bartz, 2003)**. Although the vessel structures contained in a volume dataset can be visualized using certain direct volume rendering techniques and the image generated in this way can be quite useful and suitable for the task of computer aided vascular diagnosis, in developing a computer-aided vascular surgery system, just to be able to visualize the hidden vascular structures is far from sufficient. We are required not only to see the objects but also to touch and feel them. Obviously, it can be very difficult to accurately locate and position the vascular objects directly from the image generated from direct volume rendering. This is because without an actual geometry representation for the vascular structures, it can be an extremely challenging task to register such an image with the actual vascular objects, which is a fundamental task required in the process of computer aided vascular surgery. We choose to reconstruct the vascular structures based on an explicit prior segmentation, as the isosurface extracted directly from the segmented vasculatures using surface rendering techniques

[i.e. Marching Cubes **(Lorensen and Cline, 1987)**] are usually poor, frequently exhibiting artifacts and discontinuities. The most basic requirement of a vasculature reconstruction technique is that the reconstruction geometry must be accurate. The requirement of accuracy for reconstructing a 3-D vascular tree is obviously imperative for any computer systems involving diagnosis and computer-aided surgery **(Schumann et al., 2007)**. In addition, reconstructed surfaces are expected to be of high-level smoothness and capable of being expressed in an analytical form, which is very useful in the stage of vessel analysis, such as the measurement of blood flux, vascular shape analysis, and the orientation for specific portions of the vessel structure. All these tasks could be quite difficult to be achieved based on the segmentation result it is a discrete point set **(Oeltze and Preim, 2005)**.

Due to the complex nature of vascular structures, the best way to represent the topology of vasculature is to use skeleton curves. The skeleton of an object, which is identified as the locus of the centres of maximal spheres inside the object **(Arcelli and Sanniti di Baja, 1993)**, has the ability to naturally capture important shape characteristics in three-dimensional contexts **(Faugeras, 1993)**. Thus, the skeleton-based reconstruction is regarded as the most natural option to efficiently construct the complete vascular structure **(Wu et al., 2010)**. Various skeleton-based methods have been proposed for reconstructing vasculatures from a segmented dataset. However, most of these approaches are model-based, assuming that vascular tissue has a certain regular shape, such as cylinders **(Masutani et al., 1996)**, or truncated cones **(Hahn et al., 2001)**. Although these methods can a achieve certain level of smoothness at a relatively fast speed, they are far from accurate as the model used in the reconstruction process is usually too ideal to correctly represent the actual variation

presented by the cross sections of a vascular structure. As a result, the reconstructed geometry cannot be used in a computer aided surgery system for performing computer aided minimally invasive vascular surgery. In addition, the vascular surfaces generated from most of these methods are represented either in parametric forms or as polygonal meshes, which is prone to generate discontinuities and artifacts at branching where graphics primitives are fitted together **(Preim and Oeltze, 2007).** Oeltze and Preim **(Oeltze and Preim, 2005)** have applied the convolution surfaces **(Bloomenthal and Shoemake, 1991)** to the reconstruction of vasculatures, which can achieve quite smooth vessel surfaces even at branchings. However, this method is based on an ideal model-assumption that the cross-sections of vascular structures are all circular. Actually, the cross-sections of vessels are not always circular, especially for pathologic vessels.

In this chapter, a novel approach for the accurate reconstruction of vasculature along its skeleton without model-assumptions is proposed. This technique is based on an implicit surface modelling method that we develop to model generalized cylinders. In this implicit generalized cylinders modelling method, freeform cross-sections are first reconstructed implicitly using the 2-D piecewise algebraic splines **(Li and Tian, 2009)**, and then, different cross-section profiles are weighted and summed up along the skeleton using the Partial Shape Preserving (PSP) spline basis functions **(Li and Tian, 2011)**, the 1-D version of 2-D piecewise algebraic splines. In addition, the smooth piecewise polynomial blending operations **(Li, 2007)** is employed to blend the branches of implicitly constructed generalized cylinders together. In the second part of this chapter, the proposed modelling method is applied to the accurate reconstruction of vasculatures, which includes the techniques for extracting the

skeleton of the vascular structures and the contour points for specifying accurate cross-sections. Some further issues are also discussed to refine our proposed methods. Finally, the reconstruction results, comparison with other techniques, and the qualitative and quantitative validation are demonstrated at the end of this chapter.

4.2 Skeleton-based Implicit Modelling

4.2.1 Introduction

The skeleton of an object is identified as the locus of the centres of maximal spheres inside the object **(Arcelli and Sanniti di Baja, 1993).** It is available to model the covering surface of an object once the skeleton is deduced. The intuitive appeal of the skeleton-base modelling has long been noted, since the skeleton has the ability to naturally capture important shape characteristics in three-dimensional contexts **(Faugeras, 1993)**. As suggested by Bloomenthal, it is easier to work with skeletal geometry than with the more complex geometry of the corresponding surface for designing natural forms **(Bloomenthal, 1995)**.

Generally, the representation methods for skeleton-based surfaces fall into the categories of parametric and implicit **(Bloomenthal, 1995)**. For the parametric methods, a series of individual cross-sections is defined in two dimension; and then, with the aid of reference frames **(Faux and Pratt, 1979)**, the set of cross-sections is stably established along the skeleton to form the parametric generalized cylinder **(Agin, 1972)**. The parametric generalized cylinder has the advantage of producing general models, for the cross-sections do not need to be circular, and they can be variable as they progress along the skeleton **(Bloomenthal, 1995)**. However, the parametric

generalized cylinder could suffer from the problem of intersecting cross-sections when the skeleton is too curved. In addition, for parametric methods, it is quite challenging to implement the ramification of one cylinder into two or more **(Bloomenthal, 1995)**.

An implicit surface is a surface that consists of those points $P(x, y, z)$ satisfying the implicit function, $f(P) = 0$ **(Bloomenthal and Wyvill, 1990)**. With respect to the innate property of natural representing for solid objects, the combination of skeletal and implicit techniques is particularly suitable for natural form modelling **(Bloomenthal, 1995)**. One simple implicit representation for generalized cylinders is distance surface **(Faux and Pratt, 1979)**, which defines the distance from the point in 3-D space to the skeleton with associated radius. Although the distance surface is easily implemented, it may exhibit a crease when the skeleton is non-convex and create bulges where the underlying skeletal elements join **(Hornus et al., 2003)**. A solution to those problems is provided by the method of convolution surfaces **(Bloomenthal and Shoemake, 1991)**. In this technique, the field value at a point is calculated by the integrating all the contributions of all the points on the skeleton; and then, complex surfaces can be created by summing the integrals of individual field contributions of relatively simple skeletal elements **(Angelidis et al., 2002)**. Although the resulting surface is smooth even for non-convex skeletons, it is not an easy task to modulate the filter to prevent bulging at branching. Foremost, the cross-section of the convolution surface is based on the assumption of a circular shape, regardless of the variation in size with different radius. Objects in natural scenes do not always have a circular cross-section. For example, the pathologic vessels could have an irregularly shaped cross-section. Thus, the convolution surface is not appropriate for

representing objects with irregular cross-sections. The implicit swept surfaces introduced in **(Grimm, 1999; Crespin et al., 1996)** using profile curves are limited to "star shape" due to the polar definition **(Schmidt and Wyvill, 2005)**. Though some more flexible techniques have been proposed to generate an implicit sweeping surface **(Barthe et al., 2003; Schmidt and Wyvill, 2005; Seong et al., 2006)** without the shape limitation of cross-sections, the optimization for the swept shapes remains a difficult problem due to the respecting various constraints **(Azernikov, 2008)**. In general, the surface generated by sweeping a profile curve along a skeleton curve tends to have a similar shape in its cross sections. From the perspective of vascular vessel reconstruction, sweeping surfaces do not in general meet the requirements of practical needs.

In this section, we propose a technique to model implicit generalized cylinders based on 2-D piecewise algebraic splines **(Li and Tian, 2009)**, which allows the construction of generalized cylinders with arbitrary cross-sections. Our method is based on smooth blending of a set of locally constructed general cylinders corresponding to different cross-sections along a given skeleton. The implicit generalized cylinders constructed using our method can achieve any required accuracy and continuity. Moreover, the implicit surface generated in this way can have an explicit analytical representation.

4.2.2 The Proposed Method

The main idea of the proposed technique is to approximate each cross-section of a given vascular vessel with a 2-D implicit curve, represented by 2-D piecewise algebraic splines **(Li and Tian, 2009)**, which can be regarded as a 3-D cylindrical

implicit surface. Different implicit surfaces constructed corresponding to different cross-sections are then weighted and summed up along the skeleton to form a smooth implicit surface. In the case of branching, the smooth piecewise polynomial blending operations **(Li, 2007)** is employed to blend the branches of implicit surfaces together. The main steps of our proposed method are as follows:

4.2.2.1 The calculation of Frenet frame for the skeleton

The Frenet frame is an intuitive reference frame, which consists of the tangent $\mathbf{T}$, normal $\mathbf{N}$, and binormal $\mathbf{B}$ to collectively form an orthonormal basis of 3-D space. The tangent, normal, and binormal unit vectors are defined as follows **(Wikipedia, 2010)**:

- $\mathbf{T}$ is the unit vector tangent to the curve, pointing in the direction of motion.
- $\mathbf{N}$ is the derivative of $\mathbf{T}$ with respect to the arclength parameter of the curve, divided by its length.
- $\mathbf{B}$ is the cross product of $\mathbf{N}$ and $\mathbf{T}$.

Suppose $\mathbf{S}(s)$ is the parametric curve representing the skeleton of a geometric shape, the definitions of $\mathbf{T}, \mathbf{N}, \mathbf{B}$ are given by **(Wikipedia, 2010)**

$$\begin{cases} \mathbf{T}(s) = \mathbf{S}'(s) / \left|\mathbf{S}'(s)\right| \\ \mathbf{N}(s) = \mathbf{S}''(s) / \left|\mathbf{S}''(s)\right| \\ \mathbf{B}(s) = \mathbf{N}(s) \times \mathbf{T}(s) \end{cases} \tag{4.1}$$

For arbitrary point on the skeleton, its Frenet frame can be computed conveniently. As shown in **Figure 4.1**, the black curve represents the skeleton, and the red arrow,

green arrow as well as blue arrow respectively represent the tangent vector, normal vector and binormal vector at the points on the skeleton. For a curve, there might be some "bad" points without curvature, then we utilize the $\mathbf{T}$, $\mathbf{N}$, and $\mathbf{B}$ vectors of their adjacent points instead. If the curvature is always zero then the curve will be a straight line, and it would be easy to define the orthonormal basis of R^3 for a straight line.

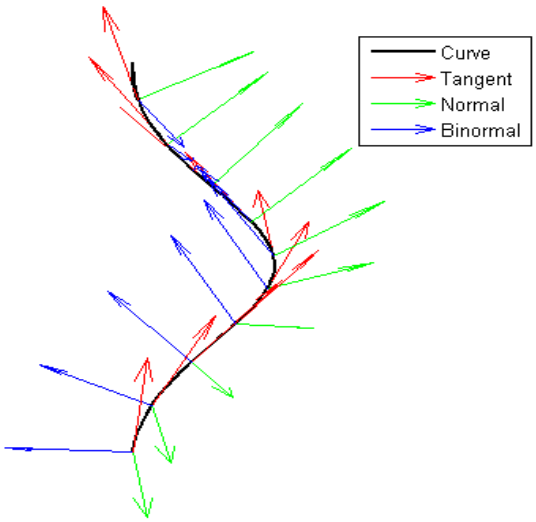


Figure 4.1: *The Frenet Frame.*

4.2.2.2 The transformation of coordinate to the Frenet-frame space

Suppose $P(x, y, z)$ is a point in 3-D space R^3, and then for a knot $\mathbf{S}(s_i)$ on the skeleton, the coordinate of P can be transformed to the orthonormal basis of R^3 constructed by the local Frenet frame defined at $\mathbf{S}(s_i)$. As shown in **Figure 4.2,** $P(x, y, z)$ is transformed to $P_f(b, n, t)$ by the following equations

$$\begin{cases} b = \left(P - \mathbf{S}(s_i)\right) \cdot \mathbf{B}(s_i) \\ n = \left(P - \mathbf{S}(s_i)\right) \cdot \mathbf{N}(s_i) \\ t = \left(P - \mathbf{S}(s_i)\right) \cdot \mathbf{T}(s_i) \end{cases} \tag{4.3}$$

where $\mathbf{B}(s_i)$, $\mathbf{N}(s_i)$ and $\mathbf{T}(s_i)$ are unit vectors representing the binormal, normal and tangent of the Frenet frame at $\mathbf{S}(s_i)$.

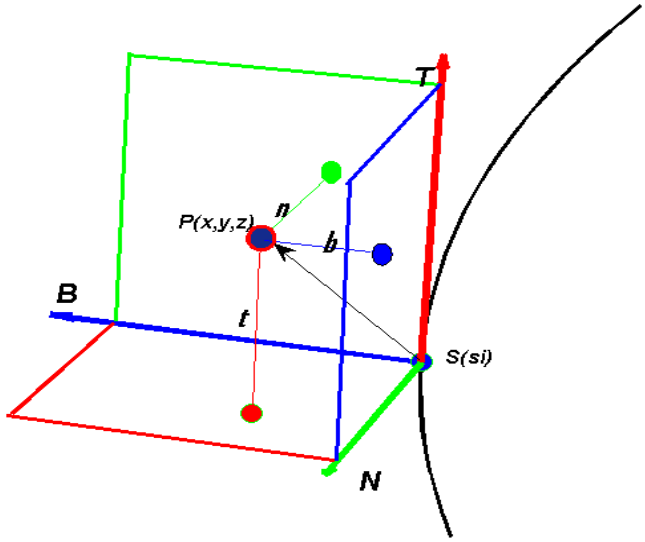
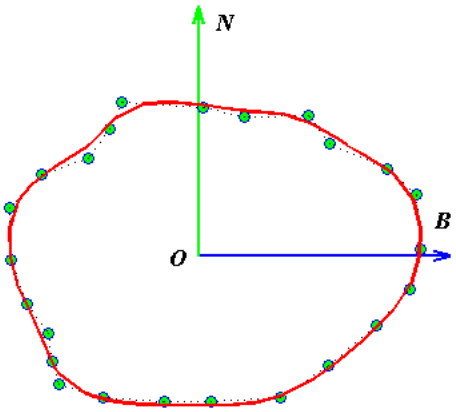


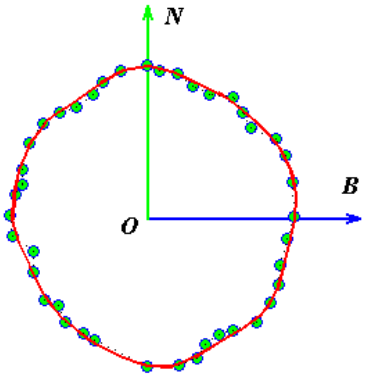
Figure 4.2: *The transformation of coordination to the local Frenet frame.*

4.2.2.3 Implicit specification of freeform cross-section by the 2-D piecewise algebraic splines

2-D piecewise algebraic splines have all the good properties of conventional 2-D tensor-product based B-spline basis functions, such as non-negativity, partition of unit and convex-hull property, which can be considered as a powerful tool for implicit shape modelling **(Li and Tian, 2009)**.



(a)



(b)

Figure 4.3: *Freeform implicit curves designed by the underlying implicit function* $B^{(2)}_{\Delta,0.2}(b,n)=0.45$.

For a simple control polygon in the Frenet frame, the corresponding implicit curve can be defined directly by $B^{(m)}_{\Delta,\delta}(b,n)=h$, where b and n respectively represent the binormal axis and normal axis in the Frenet-frame space ; Δ is the control polygon;

δ is the polygon smoothing parameter; m is the degree of smoothness of the required bivariate function; and h is the height level. The freeform cross-section can be intuitively designed by only specifying some control points in counter clockwise order and choosing the proper parameters for $B_{\triangle,\delta}^{(m)}(b,n)=h$. As discussed in **(Li and Tian, 2009)**, the height level h is generally set as 0.5, and the smaller the value of the smoothing parameter δ, the closer the implicit curve is to the underlying polygon but with less smoothness. Therefore, we set h a little smaller than 0.5 (i.e. 0.45) to obtain a smooth implicit curve closer to the underlying polygon with the same δ. **Figure 4.3** demonstrates the implicit cross-sections designed in this way. Each of these cross-sections is the contour curve of a C^2-continuous bivariate function $B_{\triangle,\delta}^{(m)}(b,n)$ with height level 0.45.

The 2-D implicit curve that defines the cross-section of the required shape at $\mathbf{S}(s_i)$, can be conveniently extruded into a 3-D implicit surface by performing a mapping from 3-D Space R^3 to local Frenet frame: $b=X(x,y,z)$, $n=Y(x,y,z)$, which is proposed in Section 4.2.2.2. Suppose the cross-section profile is defined as an implicit function $C(b,n)=B_{\triangle,\delta}^{(m)}(b,n)-h=0$, then the extruded implicit surface $F(x,y,z)$ can be directly given by

$$F(x,y,z)=C(X(x,y,z),Y(x,y,z))=B_{\triangle,\delta}^{(m)}(X(x,y,z),Y(x,y,z))-h=0 \quad (4.4)$$

which represents an implicit generalized cylinder with a freeform cross-section.

4.2.2.4 Construction of freeform implicit surface along skeleton with variable cross-sections

As shown in **(Li and Tian, 2009)**, freeform implicit surfaces can be designed as a weighted sum of a set of 2-D implicit control curves along a coordinate axis (i.e. z-axis) by spline basis functions. With the introduction of Frenet frame, different cross-section profiles can be weighted and summed up together along arbitrary skeleton, not limited to coordinate axis, to form an implicit generalized cylinder with variable cross-sections. The required generalized cylinder can be described implicitly in the Frenet-frame space by the following form

$$f(b,n,t)=\sum_{i=1}^{L} C_i(b,n)B_i(t)=0 \tag{4.5}$$

where b, n and t are defined respectively in the Frenet-frame space according to (4.3); $i=1,2,\cdots,L$, in which L is the amount of cross-sections. Supposed s_i is a list of knots for the parametric position s of skeleton $\mathbf{S}(s)$, then $C_i(b,n)$ is the implicitly defined cross-section based on the skeletal point $\mathbf{S}(s_i)$, and $B_i(t)$ is the PSP-spline basis function **(Li and Tian, 2011)** based on $\mathbf{S}(s_i)$. Suppose d is the distance between $\mathbf{S}(s_i)$ and $\mathbf{S}(s_{i+1})$, then $B_i(t)$ in (4.5) is expanded as following

$$B_i(t)=B^m_{[-d/2,d/2],\delta}\left(t\right) \tag{4.6}$$

In which the general spline basis function $B^m_{[a,b],\delta}\left(x\right)$ is defined in **(Li and Tian, 2009)**

$$B^{m}_{[a,b],\delta}(x) = H_m\left(\frac{b-x}{\delta}\right) - H_m\left(\frac{a-x}{\delta}\right) \tag{4.7}$$

where $[a,b]$ is an interval with $a \le b$; and $H_m(x)$ is the smooth unit step function introduced in **(Li et al., 2006)**.

Figure 4.4 demonstrates how the implicit function $f(b,n,t)$ is constructed as a weighted sum of a set of implicit cross-sections $C_i(b,n)$ along the skeleton **S** by spline basis functions $B_i(t)$.

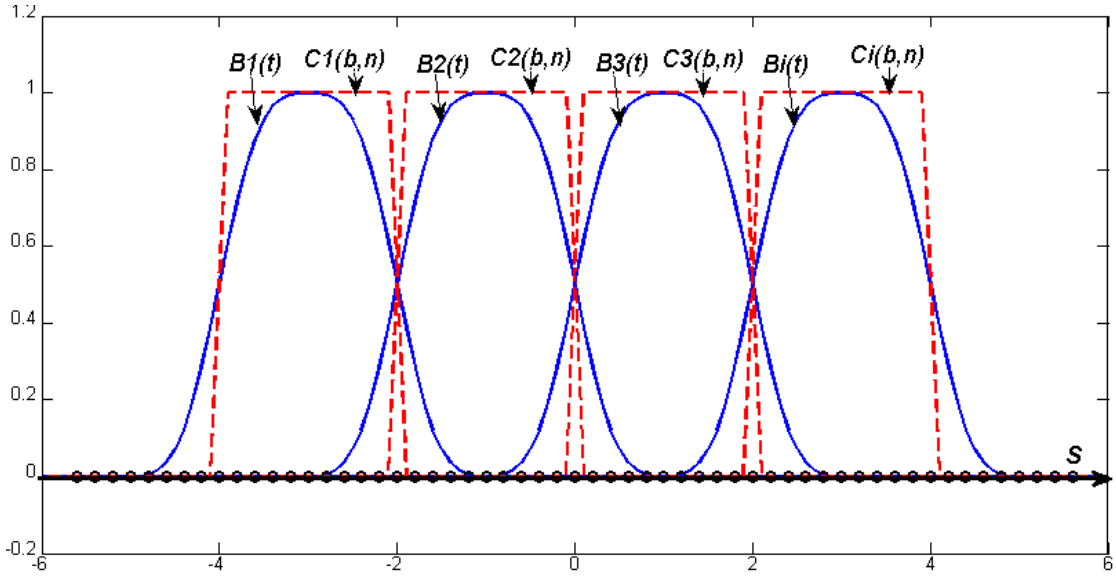
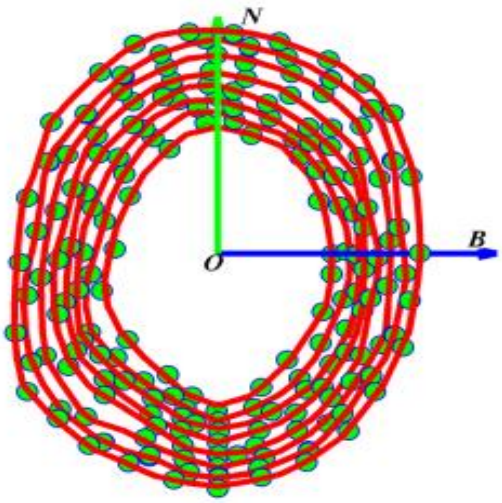
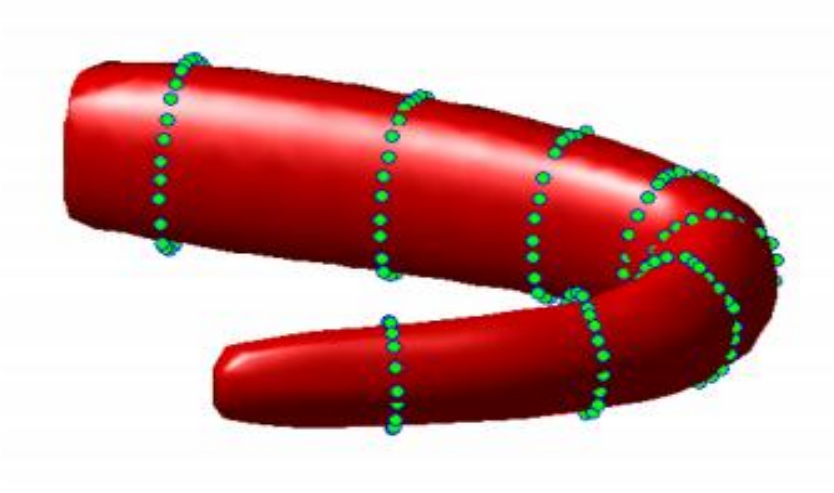


Figure 4.4: *The construction of implicit function* $f(b,n,t)$ *by weighted summing a set of implicit cross-sections* $C_i(b,n)$ *along a skeleton with spline basis functions* $B_i(t)$.

For a given skeleton **S**, once the generalized cylinder has been described in the Frenet-frame space, it is easily transformed to R^3 space by performing the mapping presented in Section 4.1.2.2: $b = X(x,y,z)$, $n = Y(x,y,z)$ and $t = Z(x,y,z)$.



(a)



(b)

Figure 4.5: *The construction of an implicit generalized cylinder by weighted summing eight variable cross-sections along a skeleton with spline basis functions.* **(a)** *The eight variable cross-sections.* **(b)** *The implicit generalized cylinder along a skeleton.*

Therefore, the generalized cylinder can be described implicitly in R^3 space by the following form

$$F(x,y,z) = f(b,n,t) = f(X(x,y,z),Y(x,y,z),Z(x,y,z)) = 0 \tag{4.8}$$

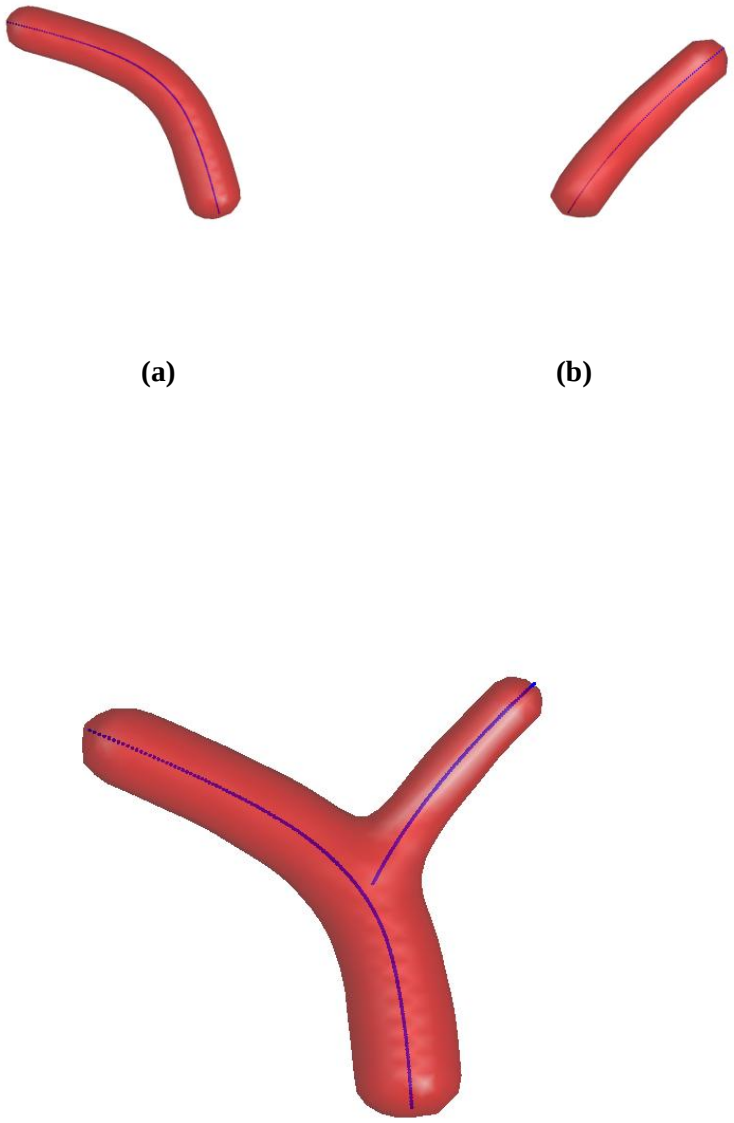
As presented in **Figure 4.5**, the implicit generalized cylinder is constructed as a weighted sum of four variable cross-sections [see **(a)**] along a skeleton with spline basis functions.

4.2.2.5 Blending for the branches of implicit shapes

The technique presented above is only appropriate when no branches exist for the given skeleton. In the case of branching, an implicit general cylinder is first constructed from each individual branch, and then all these branching implicit general cylinders can be blended together to represent the overall implicit surfaces corresponding to the entire structure. There are various ways to blend a set of implicit surfaces, but most of them are achieved with non-piecewise-polynomial shape operators and without flexibility in controlling the blending range. Since the implicit surfaces from different branches of the skeleton are piecewise polynomial, piecewise polynomial shape-blending operations are expected. In this paper, we use the extended smooth maximum function introduced in **(Li, 2007)** to blend implicit shapes constructed from different skeletal branches. The extended smooth maximum function is defined in the following form

$$\max_{n,\delta}(x,y) = \frac{1}{2}\left(x + y + |x - y|_{n,\delta}\right) \tag{4.9}$$

where $|x|_{n,\delta}$ is the smooth absolute function with the smoothness degree n and blending range-control parameter δ defined in **(Li, 2007)**.



(a) **(b)**

(c)

Figure 4.6: *Smooth blending of two implicitly defined shapes with* $\max_{n,\delta}(F_1, F_2)$.

Suppose F_1 and F_2 are two implicit shapes constructed from different skeletal elements, the smooth blending of these two implicit shapes can be achieved by the following operation

$$F_{blend} = \max_{n,\delta}(F_1, F_2) = \frac{1}{2}\left(F_1 + F_2 + \left|F_1 - F_2\right|_{n,\delta}\right) \tag{4.10}$$

Figure 4.6 demonstrates the smooth blending of two implicitly defined shapes. **(a)** and **(b)** represent two individual implicit objects, and **(c)** represents the blending result of these two objects by using the smooth maximum function $\max_{n,\delta}(F_1, F_2)$.

4.2.3 Discussion

4.2.3.1 Extruded surface along implicit extrusion path

In the case of a very curved skeleton, the distance d_i between two skeletal points should be set quite small to guarantee the smooth blending of different cross-section profiles. As an alternative, we can extrude a 2-D specified cross-section profile into 3-D implicit surfaces along the implicitly fitted curve to the skeleton. The implicit curve, that is the extrusion path, can be represented implicitly as the intersection of two implicit surfaces $F_1(x, y, z) = 0$ and $F_2(x, y, z) = 0$ [see **Figure 4.7(a)]**, then Equation 4.4 can be changed to

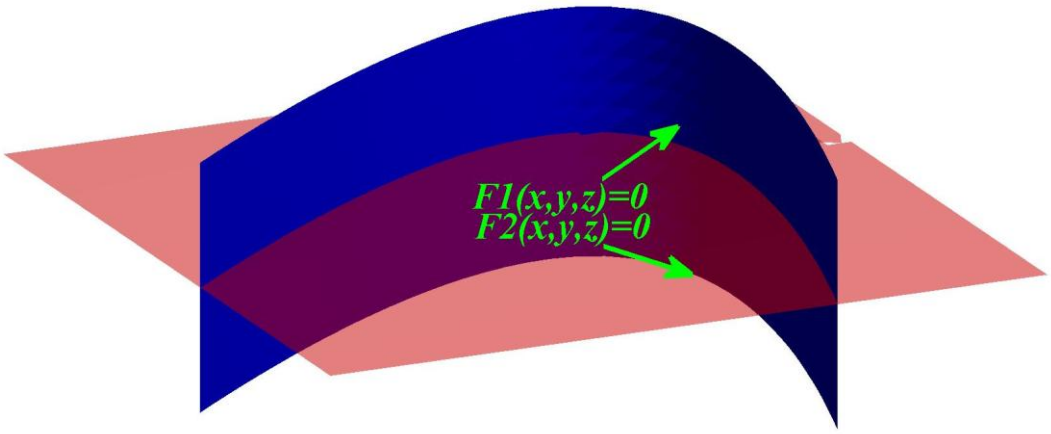
$$F(x, y, z) = C(F_1(x, y, z), F_2(x, y, z)) = B^{(m)}_{\Delta,\delta}(F_1(x, y, z), F_2(x, y, z)) - h = 0 \tag{4.11}$$

which represents the extruded implicit surface. $b = X(x, y, z)$ and $n = Y(x, y, z)$, defined in Section 4.2.2.2 can be regarded as the special situation of $F_1(x, y, z)$ and $F_2(x, y, z)$, which implies that the extrusion path is the tangent of the Frenet frame at $\mathbf{S}(s_i)$.

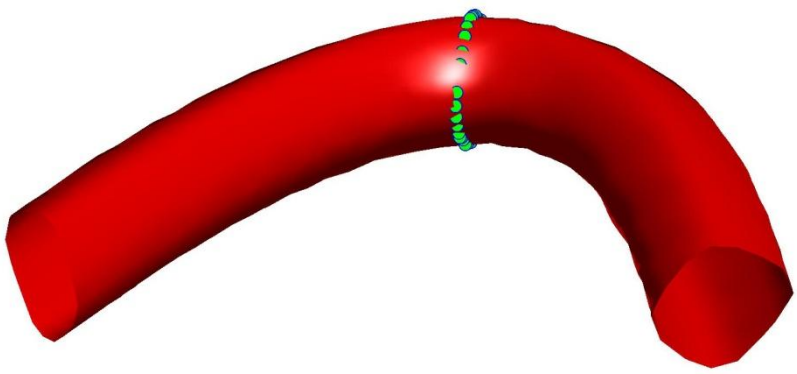
The key issue is to fit the implicit curve to the skeletal points instead of just the transformation of coordinates to the Frenet-frame space. The main steps are as follows:

1. Suppose $\mathbf{S}(s_i)$ is the centre point of a segment of skeleton, then each point on this segment can be projected on the normal plane of the Frenet frame defined at $\mathbf{S}(s_i)$. In other words, the skeletal segment is projected on the **TB**-plane of the local coordinate system defined by **T**, **B**, **N** at $\mathbf{S}(s_i)$;

2. A polynomial curve is fitted to the projected points on the **TB**-plane. In the space defined by the local coordinate system **TBN**, the curve can be implicitly represented as the intersection of the two implicit surfaces: $f_c(t, b) = 0$ and the normal plane $n = 0$;



(a)



(b)

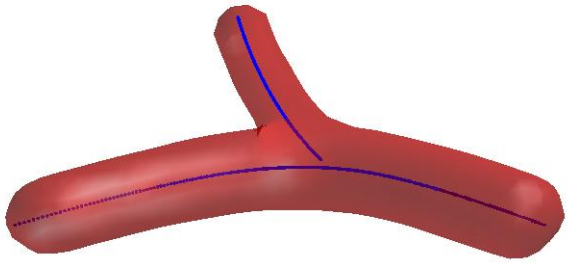
Figure 4.7: (a) *Implicit curve represented as the intersection of two implicit surfaces* $F_1(x,y,z)=0$ *and* $F_2(x,y,z)=0$. **(b)** *Extruded implicit surface along the extrusion path defined in (a). Green points indicating the 2-D cross-section profile.*

3. The two implicit surfaces defined at the local coordinate are then transformed to the world coordinate system: $F_1(x, y, z) = 0$ and $F_2(x, y, z) = 0$.

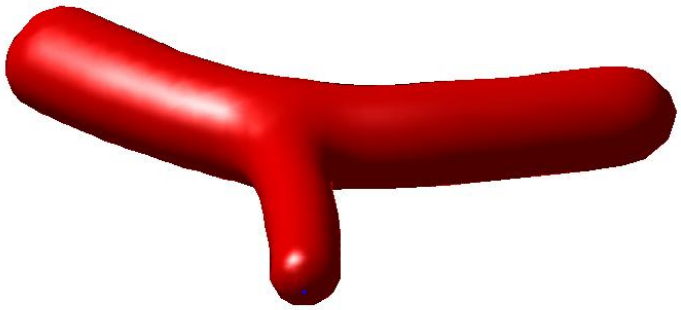
As shown in **Figure 4.7**, a 2-D cross-section profile (indicated by the green points) is extruded along the extrusion path defined in **(a)** to form the implicit surface **(b)**.

4.2.3.2 Bulge-free blending

Convolution surfaces are bulge-free for non-branching line segment skeletons due to the superposition property of convolution, but they may exhibit bulges at branching **(Oeltze and Preim, 2005)**. Our novel approach inherits the weighted summation mechanism to generate smooth surface with different cross-sections for the same skeletal element. In the case of branching, we can easily achieve bulge-free blending by utilizing the smooth blending operation $\max_{n,\delta}()$ to blend the branches of implicit surfaces for different skeletal elements. As presented in **Figure 4.8** and **Figure 4.9**, our method can achieve smooth and bulge-free blending at branchings with two and three branches. In addition, it can readily be expanded to the situation of more branches.

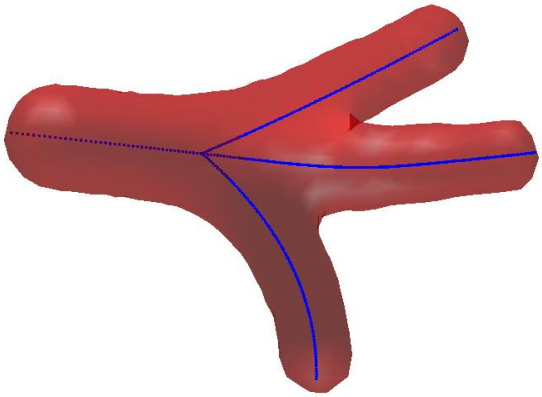


(a)

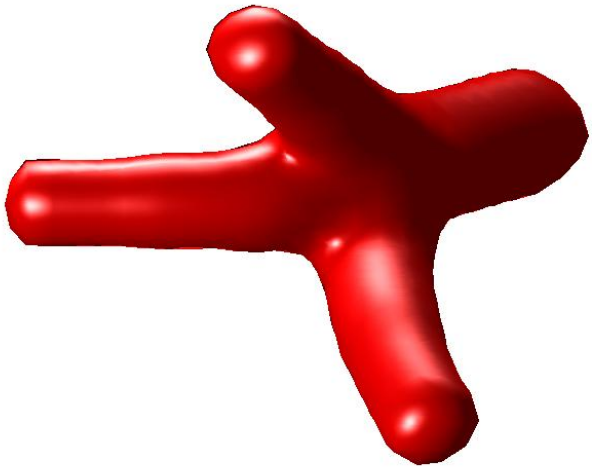


(b)

Figure 4.8: *Smooth and bulge-free blending for two branches.*



(a)



(b)

Figure 4.9: *Smooth and bulge-free blending for three branches.*

4.3 Accurate Reconstruction of Vasculatures

In this section, the method proposed in the last section is applied to the accurate reconstruction of vascular structures. Firstly, the methods for extracting the skeleton of the vascular structures and the contour points for specifying accurate cross-sections are presented. Secondly, we describe the main steps for accurate reconstruction of vascular structures. In addition, some further issues associated with our proposed method are also discussed. Finally, the reconstruction results and comparisons with other techniques are presented.

4.3.1 The Extraction of Vascular Axes

So far, various approaches have been developed to extract the skeleton of vascular structures **(Orkisz and Hernández-Hoyos, 2001; Kirbas and Quek, 2004; Tizon, 2004 ; Wink, 2004)**. Generally, these techniques can be classified into two categories **(Carrillo et al., 2005)**. In the first category, the skeleton of a binary volume can be identified by the technique of morphological thinning **(Sauret et al., 1999)**. However, this technique suffers from the problem of being sensitive to noise **(Carrillo et al., 2005)**. On the other hand, the methods based on the second category employ a step by step approach, which moves a small volume of interest, such as a parallelepiped **(Flasque et al., 2001)** or sphere **(Antiga et al., 2001)** along the vessel tree.

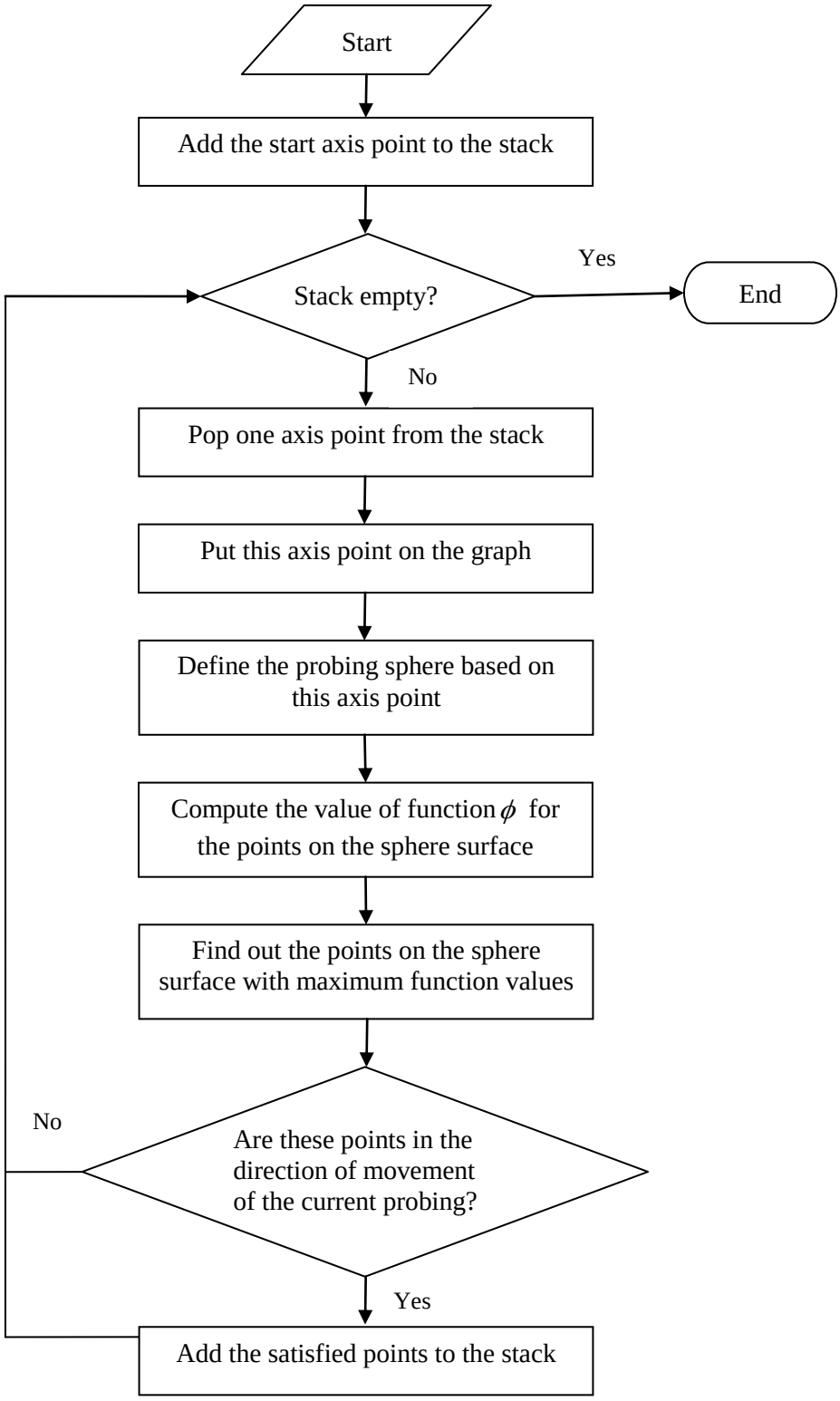


Figure 4.10: *Flow chart for the iterative process of detecting axis points.*

Our method falls into the second category, but there are some main differences. With the introduction of Signed Distance Function (SDF) **(Osher and Fedkiw, 2002)**, the moving sphere along the vessel tree can easily and automatically detect bifurcations and predict the location of next point. The main notion of our method is that, once the raw medical data has been segmented, the zero set of the embedding function ϕ can be used to represent the vessel contour $C = \{\mathbf{p} \mid \phi(\mathbf{p}) = 0\}$, where $\mathbf{p} \in R^3$; then the fast marching method **(Sethian, 1999)** is employed to initialize ϕ to be SDF, which guarantees that the closer the point to the vascular axe, the bigger the value of function ϕ; under the condition of SDF, the probing sphere can easily detect the skeleton by moving towards to the point on the sphere with a maximum value of function ϕ. **Figure 4.10** presents the flowchart of the iterative process for detecting axis points.

The main steps of the iterative process are as follows:

- Add the axis point taken from the stack to the graph representing the vascular tree axis.
- Define the probing sphere based on the current axis point and compute the value of function ϕ for the points on the sphere surface.
- Find out the points on the sphere surface with maximum function values and test them whether they are in the direction of movement of the probing sphere.
- Push the satisfying points into the stack to be processed later.

This process begins with a user defined point and finishes when there are no more points in the stack to be processed.

The process of moving the probing sphere is demonstrated in **Figure 4.11**, where blue points represent the detected axis points; C stands for the current axis point representing the barycentre of the probing sphere; and $N1$ stands for the next axis point which satisfies the following two conditions:

1) It has the maximum value of function ϕ on the surface of the current sphere.
2) It is in the direction of movement of the current probing sphere.

In **Figure 4.12**, the situation of bifurcation is presented. $N1$ stands for the first next axis point which has the first maximum value of function ϕ on the surface of the current sphere; and $N2$ stands for the second next axis point which has the second maximum value of function ϕ on the surface of the current sphere.

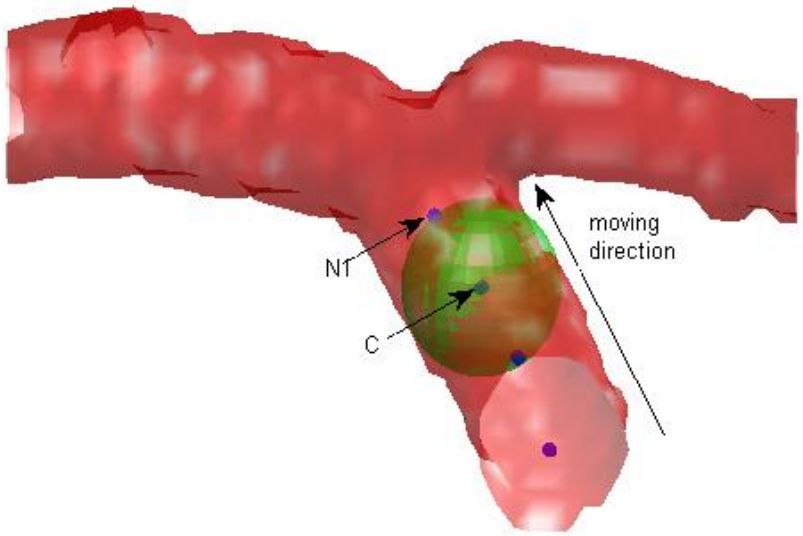


Figure 4.11: *The moving process of probing sphere.*

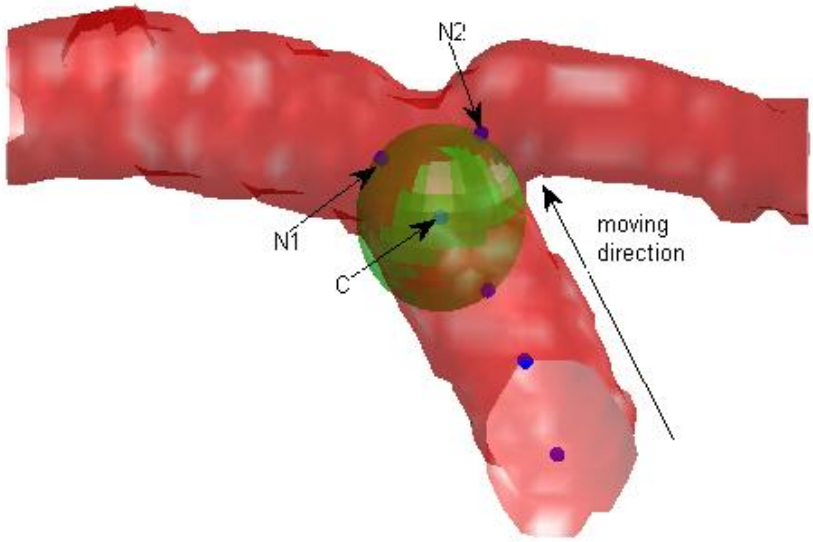
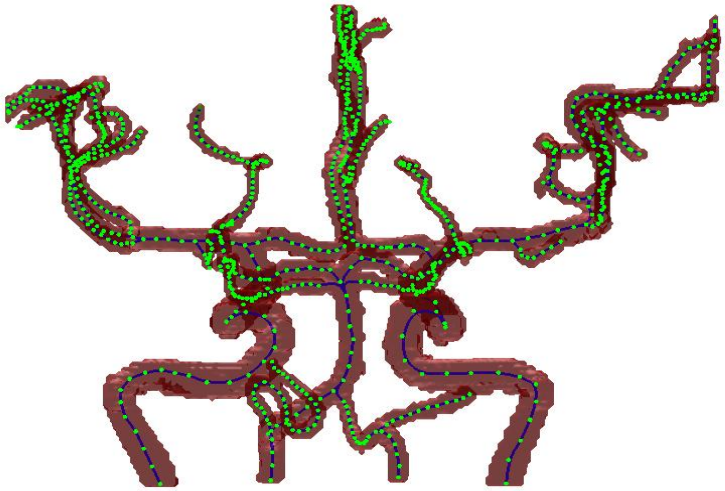
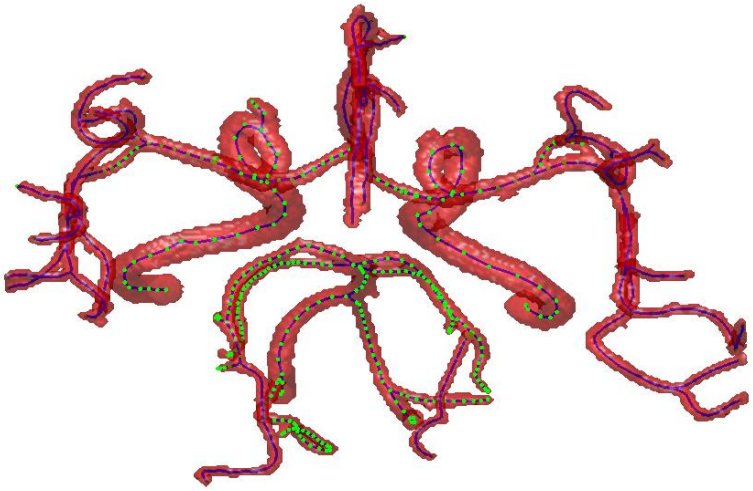


Figure 4.12: *The moving process of probing sphere with bifurcation.*

Figure 4.13, **Figure 4.14**, **Figure 4.15**, and **Figure 4.16** demonstrate some typical examples of skeleton extraction of vessel trees segmented from 3-D medical datasets. Green points represent the extracted axis points, and smooth blue curves represent the skeletons approximated from the axis points by spline function.

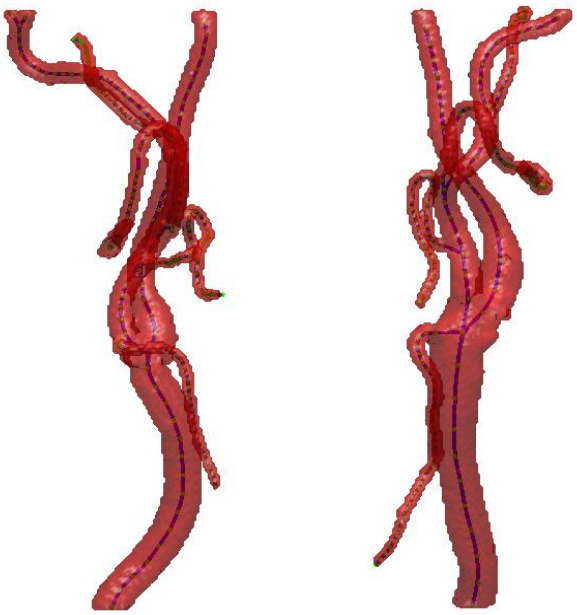


(a)

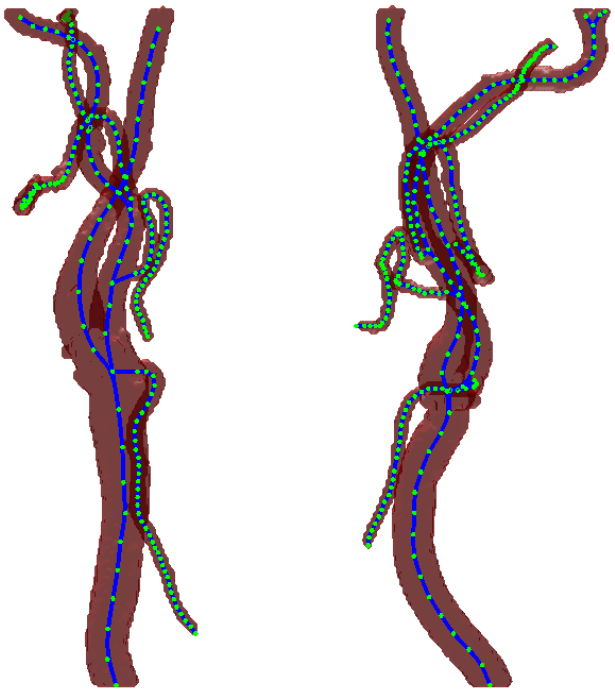


(b)

Figure 4.13: *Skeleton extraction of segmented vessel tree from MRA cerebral dataset.*



(a)



(b)

Figure 4.14: *Skeleton extraction of segmented vessel tree from CTA carotid artery dataset.*

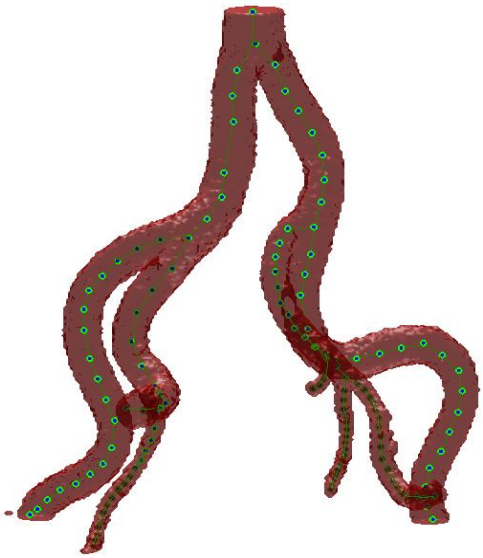


Figure 4.15: *Skeleton extraction of segmented vessel tree from MRA abdominal aorta dataset.*

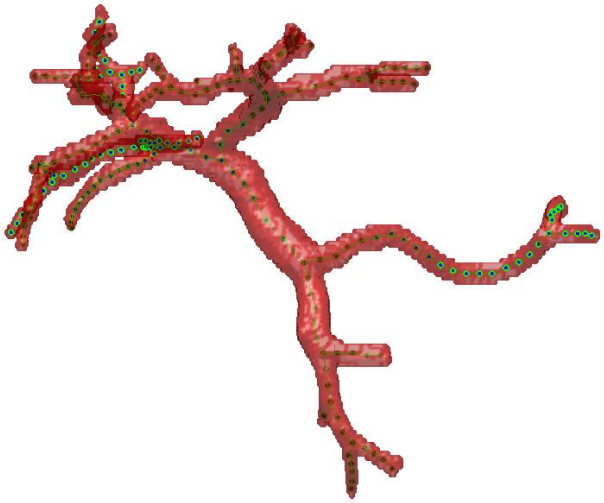


Figure 4.16: *Skeleton extraction of segmented vessel tree from liver portal vein dataset.*

4.3.2 The Extraction of Control Points for the Specification of Accurate Cross-sections

Before constructing the implicit generalized cylinder for the vasculatures, we need to extract the control points for specifying accurate cross-sections. Suppose $\mathbf{S}(s_i)$ is a skeletal point, then the steps for extracting the control points for specifying the cross-section based on $\mathbf{S}(s_i)$ are as follows:

1. Define a rectangle based on the local coordinate system (see **Figure 4.17**).

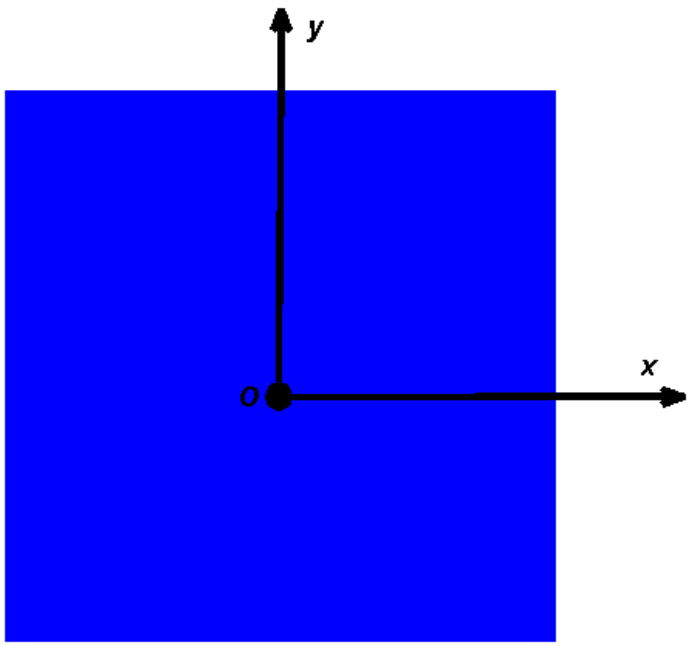


Figure 4.17: *Defined rectangle on the local coordinate system.*

2. Transform the coordinate of the rectangle to its world coordinate system, and intersect with the vessel surface (see **Figure 4.18**). The steps of transformation are as follows:

a. Rotate the local z-axis to make coincident with the Tangent unit vector $\mathbf{T}(s_i)$ on the Frenet frame at $\mathbf{S}(s_i)$;

b. Translate the local original point to the position of $\mathbf{S}(s_i)$;

c. Rotate the local x-axis and y-axis to make coincident with the Binormal unit vector $\mathbf{B}(s_i)$ and Normal unit vector $\mathbf{N}(s_i)$ on the Frenet frame at $\mathbf{S}(s_i)$ respectively.

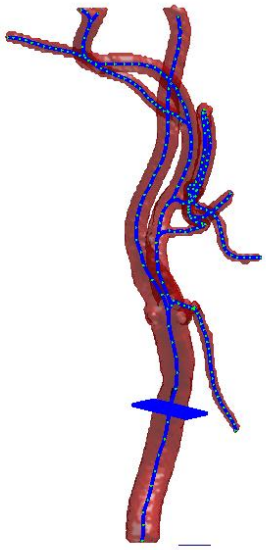


Figure 4.18: *The intersection of the defined rectangle with the vessel surface.*

3. Map the intensity value of the segmented vessel into the defined rectangle. As shown in **Figure 4.19**, green pixels with intensity value larger than zero, represent the region inside the vessel, while blue pixels with intensity value smaller than zero, represent the region outside the vessel.

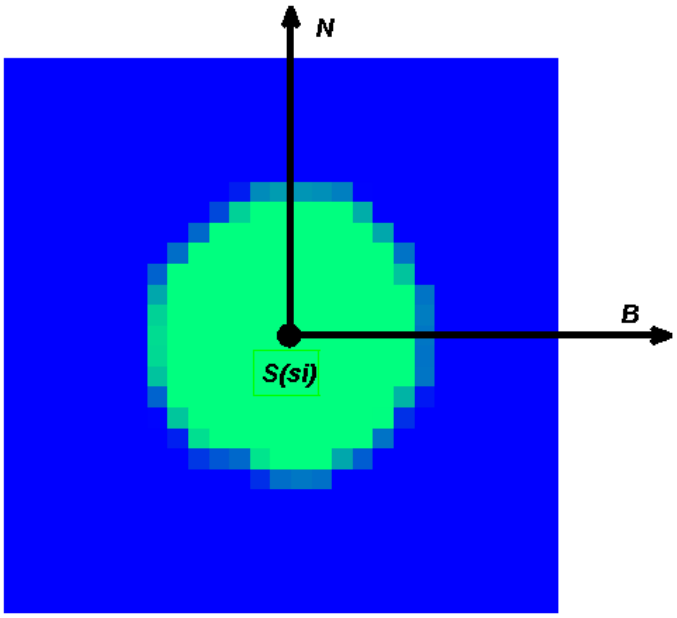


Figure 4.19: *Mapping the intensity value of the segmented vessel into the rectangle.*

4. Extract the contour points with zero intensity value. As shown in **Figure 4.20**, the red points represent the extracted contour points with zero intensity value.

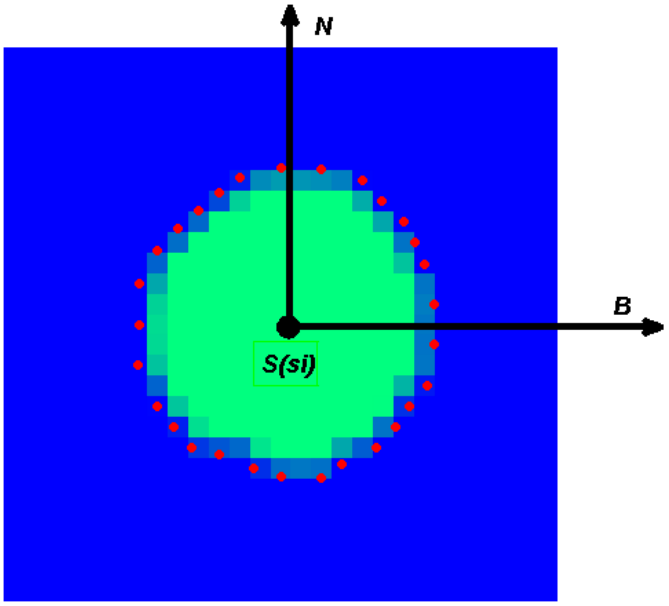


Figure 4.20: *The extraction of contour points with zero intensity value.*

4.3.3 Reconstruction Steps of Vascular Structures

Once the skeleton of the vessel structure has been extracted and steps for extracting the contour points for specifying the cross-section of vessel have been presented, we can employ our proposed implicit generalized cylinder to model the vasculatures accurately. The modelling steps are as follows:

1. Suppose $\mathbf{s}_j(s)$ is a skeletal branch of the vessel tree and $P(x, y, z)$ is an arbitrary point in 3-D space R^3 containing the skeleton $\mathbf{s}_j(s)$, for each control point $\mathbf{s}_j(s_i)$ on the skeleton, the coordinates of P are transformed to the orthonormal basis of R^3 constructed by the local Frenet frame at $\mathbf{S}(s_i)$ (please refer to Section 4.2.2.2).

2. For each knot $\mathbf{s}_j(s_i)$ on skeleton, we employ the method proposed in Section 4.3.2 to extract the control points for specifying the cross-section based on $\mathbf{s}_j(s_i)$.

3. Based on the control points, the smooth curve of cross-section C_i is implicitly specified by the 2-D piecewise algebraic splines **(Li and Tian, 2009)** (please refer to Section 4.2.2.3). As shown in **Figure 4.21**, a green smooth curve is specified by the extracted red contour points.

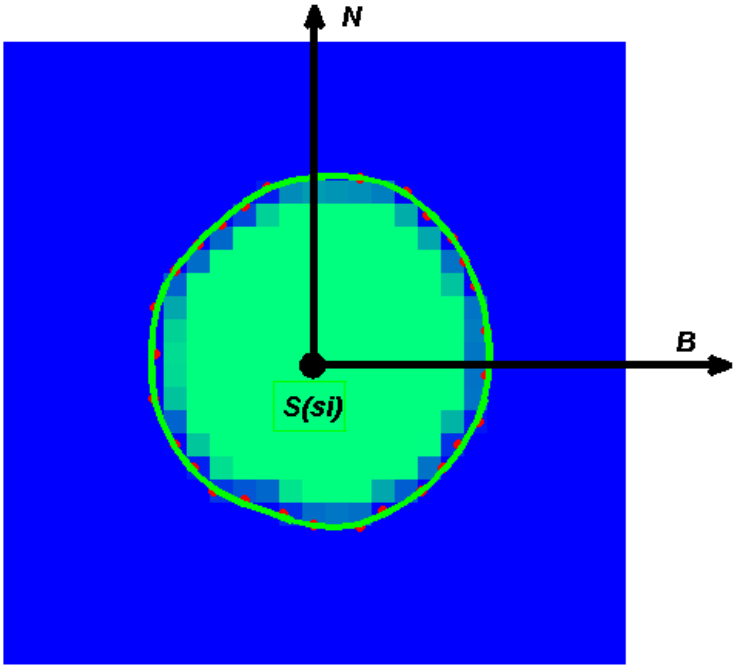
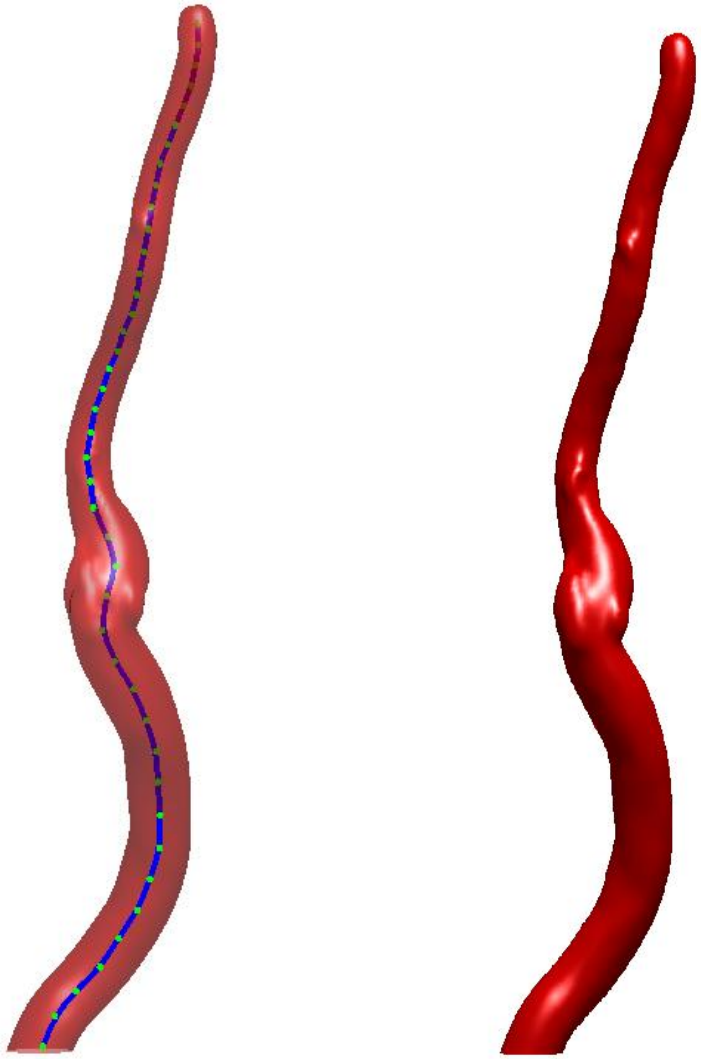


Figure 4.21 : *The smooth implicit curve specified by the extracted contour points.*

4. Different cross-section profiles specified at corresponding skeletal points $\mathbf{S}_j(s_i)$ are now weighted and summed up together along the skeleton $\mathbf{S}_j(s)$, to form an implicit generalized cylinder representing the vessel branch (please refer to Section 4.2.2.4). **Figure 4.22** demonstrates a smoothly reconstructed result of vessel surface along the first skeletal branch $\mathbf{S}_1(s)$: **(a)** is the translucent surface with vessel skeleton and; **(b)** is the opaque vessel surface.

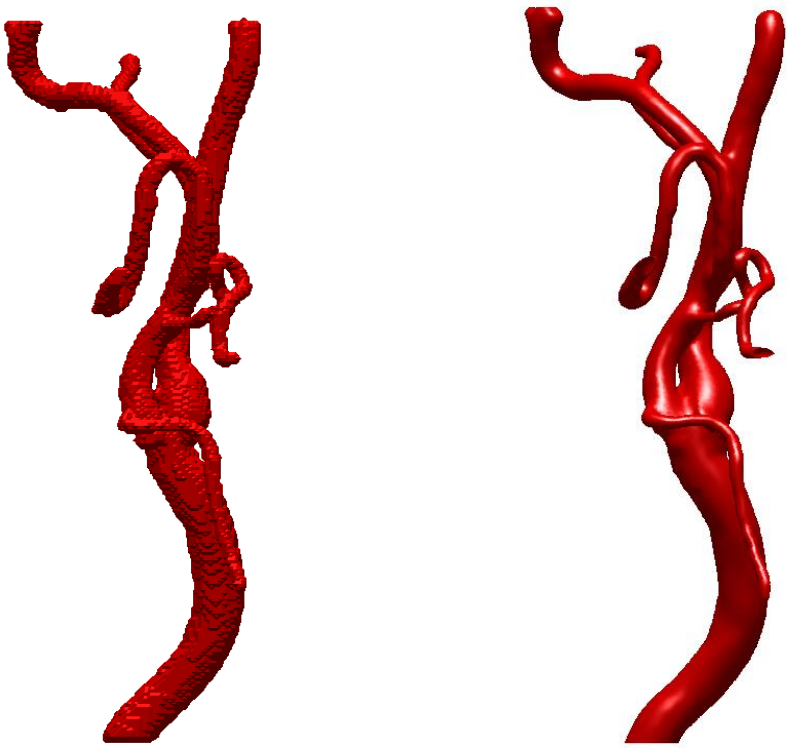


(a) **(b)**

Figure 4.22: *The smooth reconstruction result of vessel surface along the first skeletal branch* $\mathbf{S}_1(s)$. **(a)** *The translucent surface with vessel skeleton.* **(b)** *The opaque vessel surface.*

5. Finally, different branches of the implicitly defined vessel structures from different skeletal branches $\mathbf{S}_j(s)(j=1,2,3,\cdots,N)$ are blended together by using the smooth maximum function $\max_{n,\delta}(x,y)$ to construct the complete

vascular tree. As presented in **Figure 4.23**, **(a)** is the segmented result of vessel structures, and **(b)** is the reconstruction result with our method from **(a)**.



(a) **(b)**

Figure 4.23: (a) *The segmented result of vessel structures*. **(b)** *The reconstruction result with our method from (a).*

4.3.4 Further Implementation Details

4.3.4.1 Determination of the side length for the intersecting rectangle

As mentioned in Section 4.3.1, when the embedding function ϕ of the vessel dataset is initialized to be SDF, the function value on the point inside the vessel represents the minimum distance from that point to the vessel surface. Suppose $\mathbf{S}(s_i)$ is a

skeletal point, and C_i is the cross-section defined based on $\mathbf{S}(s_i)$. Generally, $\mathbf{S}(s_i)$ is the barycentre or at least close to the barycentre of C_i, since it has the local maximum function value which has been recorded during the process of skeleton extraction. Then the pre-calculated function value $\phi(\mathbf{S}(s_i))$ can be approximately regarded as the radius R of the cross-section C_i. The side length for the intersecting rectangle defined based on C_i is usually set as $3*\phi(\mathbf{S}(s_i))+1$, which is long enough to ensure that the rectangle can cover the whole cross-section (see **Figure 4.24**).

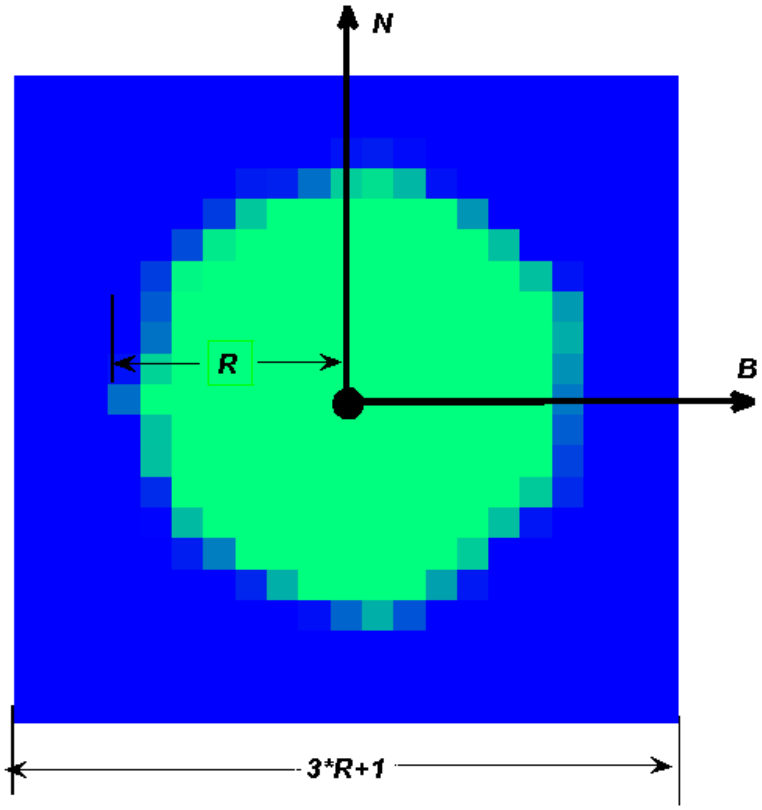
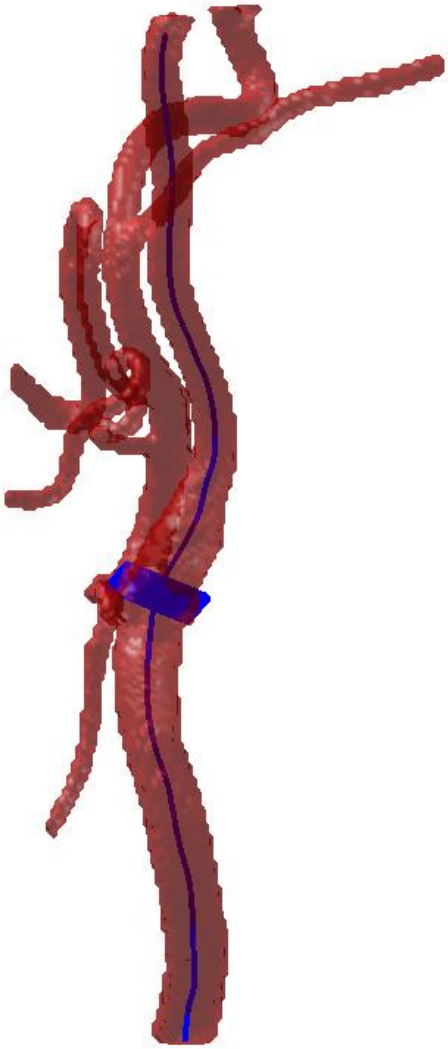


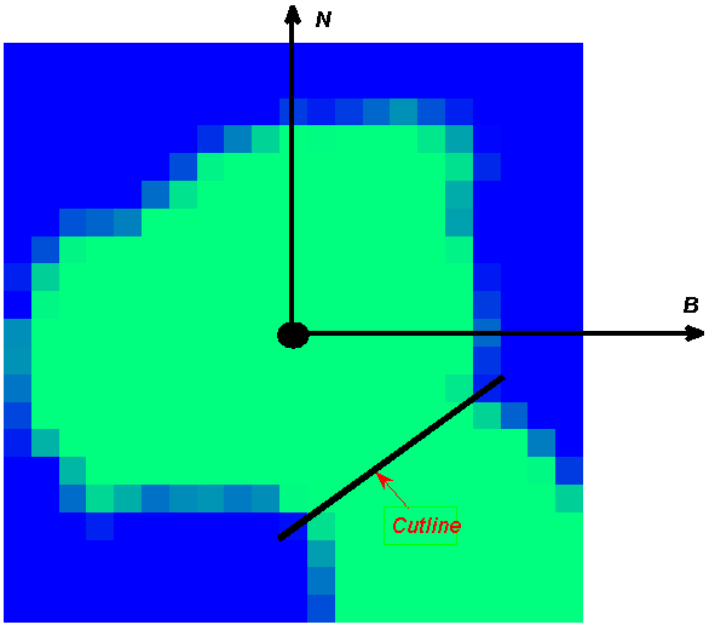
Figure 4.24: *The side length for the intersecting rectangle.*

If the cross section of the vessel exceeds the area of the defined rectangle, we suppose that the rectangle is intersecting with more than one vessel branch. As shown in **Figure 4.25(b)**, the green area above the cutline is the cross-section of the

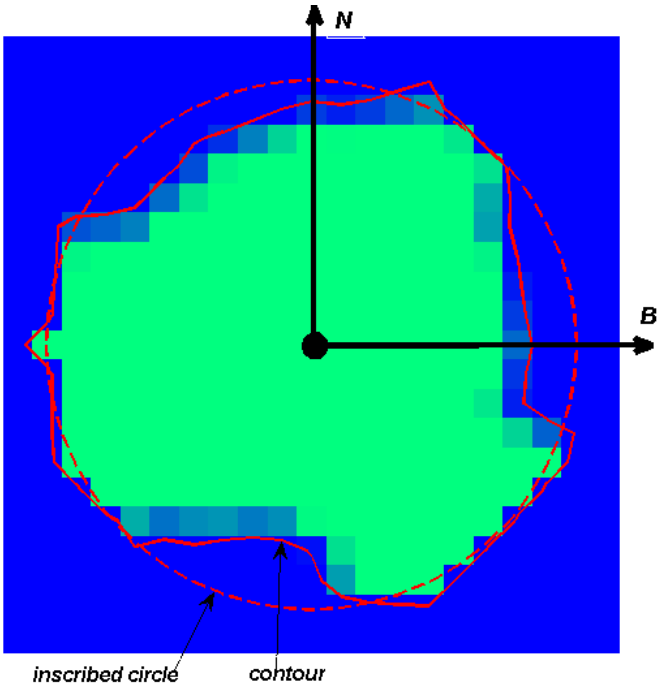
intended branch, and the green area below the cutline is part of the cross-section of the unintended branch. In this case, in order to extract the contour points, we drop some of the target points outside the inscribed circle of the rectangle to ensure the target area is closed [see **Figure 4.25(c)]**. Most of the dropped target points are in the area of the cross-section of the unintended branch. Although the target area may still include a certain part of the unintended branch, this does not affect the reconstruction result, since the area will be blended with the vessel structure constructed from the corresponding branch.



(a)



(b)

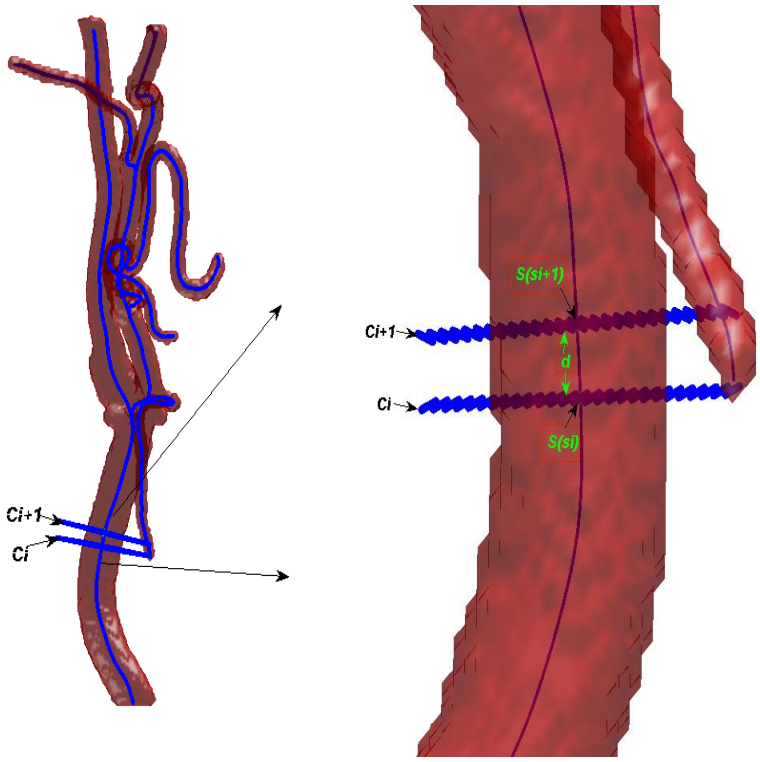


(c)

Figure 4.25: *The modification of the target area when the cross section of the vessel exceeds area of the defined rectangle.* **(a)** *The intersection of the defined rectangle with the vessel surface with branchings.* **(b)** *The green area above the cutline is the cross-section of the intended branch, and the green area below the cutline is part of the cross-section of the unintended branch.* **(c)** *Dropping some of the target points outside inscribed circle of the rectangle to ensure the target area is closed.*

4.3.4.2 Determination of the interval between the two adjacent defined cross-sections

Generally, for real vasculatures, the difference between two cross-sections defined on the two adjacent approximated skeletal points is quite small; thus, it is unnecessary to define the cross-section for every approximated skeletal point. Therefore, in our method, we choose a list of knots s_i for the parametric position s of skeleton $\mathbf{S}(s)$ to define the cross-section. The interval d between two adjacent knots is proportional to the radius of the current defined cross-section and inversely proportional to the curvature of the skeleton at the current knot. Generally, d is set as half the radius of current profile. As shown in **Figure 4.26**, **(a)** is the overview of the two adjacent cross-sections intersecting with the vessel surface; and **(b)** is a detailed look at the two adjacent cross-sections, in which C_i and C_{i+1} represent the two adjacent cross-sections; the green points $\mathbf{S}(s_i)$ and $\mathbf{S}(s_{i+1})$ represent the two adjacent sample skeletal points at interval d .



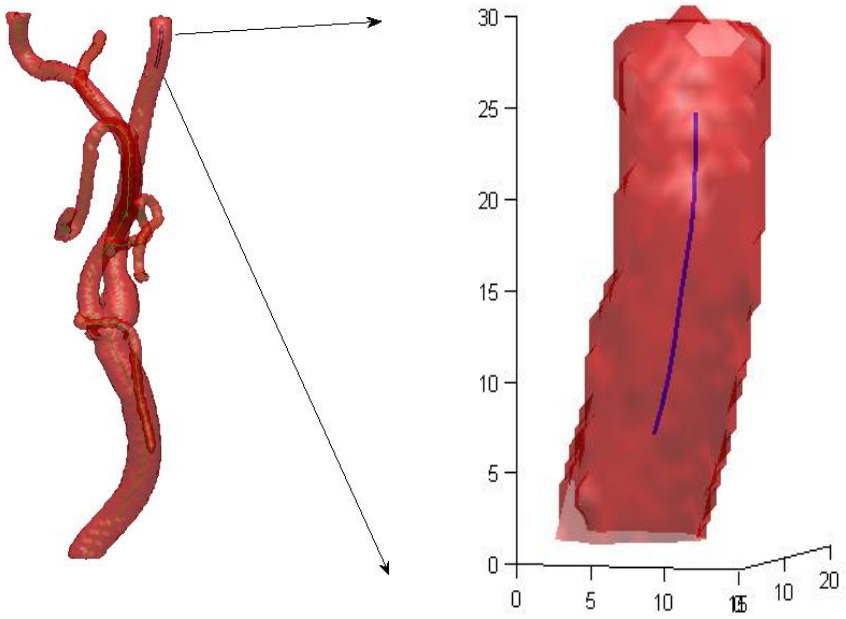
(a) **(b)**

Figure 4.26: (a) *The overview of the two adjacent cross-sections.* **(b)** *A detailed look at the two adjacent cross-sections.*

4.3.4.3 Subdivision for the volume along the skeleton

When a skeletal branch is long, it is necessary to partition the space along the skeleton for the purpose of accelerating the computation. In other words, the long skeleton is divided into several sub-skeletons, and for each sub-skeleton, an axis-aligned bounding box is computed in voxel coordinates. As presented in **Figure 4.27**, **(a)** is a sub-skeleton subdivided from the whole skeleton and **(b)** is its corresponding axis-aligned bounding box. Then for each sub-skeleton, its local implicit function is constructed within its corresponding axis-aligned bounding box, and mapped to the world coordination system. Finally, all of the local implicit functions are summarized together to form the global implicit function representing the vessel

structure reconstructed from the whole skeletal branch. More details will be presented in **Chapter 5**.



(a) **(b)**

Figure 4.27: (a) *A sub-skeleton subdivided from the whole skeleton.* **(b)** *The corresponding axis-aligned bounding box.*

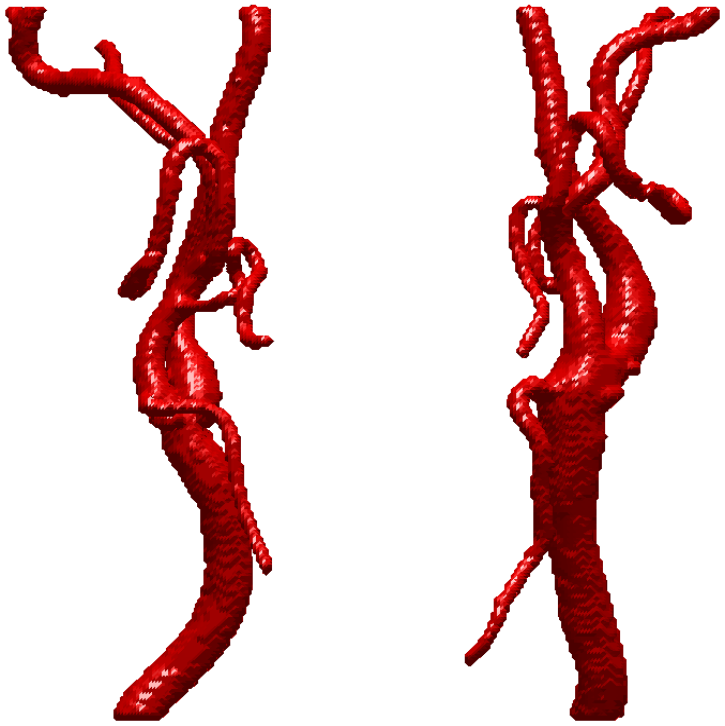
4.3.5 Results and Discussion

4.3.5.1 Reconstruction results based our method

Our proposed technique has been applied to over ten medical datasets. In the following, we present some typical reconstruction results, which represent the characteristic vascular structures of human vasculature system. The first example is the reconstruction of the carotid artery for 3-D CT angiography (CTA) images with a resolution of $512\times512\times206$ and spacing of $0.52\text{mm}\times0.52\text{mm}\times0.63\text{mm}$. As presented in **Figure 4.28**, **(a)** is the isosurface rendering of segmentation result of carotid artery, and **(b)** is the corresponding reconstructed result with our method. The

second example is the reconstruction of cerebral vasculatures for the 3-D magnetic resonance angiography (MRA) images with a resolution of $352\times448\times114$ and spacing of $0.49mm\times0.49mm\times0.80mm$. **Figure 4.29(b)(c)(d)** demonstrate the reconstruction results of MRA cerebral vessels using the segmented data shown in **Figure 4.29(a)**. The third example is the reconstruction of the abdominal aorta for 3-D MRA images with a resolution of $512\times512\times310$ and spacing of $0.70mm\times0.70mm\times0.63mm$. As shown in **Figure 4.30**, **(a)** is the isosurface rendering of segmentation result of abdominal aorta, and **(b)** is the reconstructed result using our method. The last example is the reconstruction of the segmented liver portal vein using a medical dataset obtained from the public resource (http://www.ircad.fr/softwares/3-Dircadb/3-Dircadb1/index.php), with a resolution of $512\times512\times151$ and spacing of $0.78mm\times0.78mm\times1.60mm$. As presented in **Figure 4.31, (a)** is the isosurface rendering of segmented liver portal vein, and **(b)** is the corresponding reconstructed result with our method.

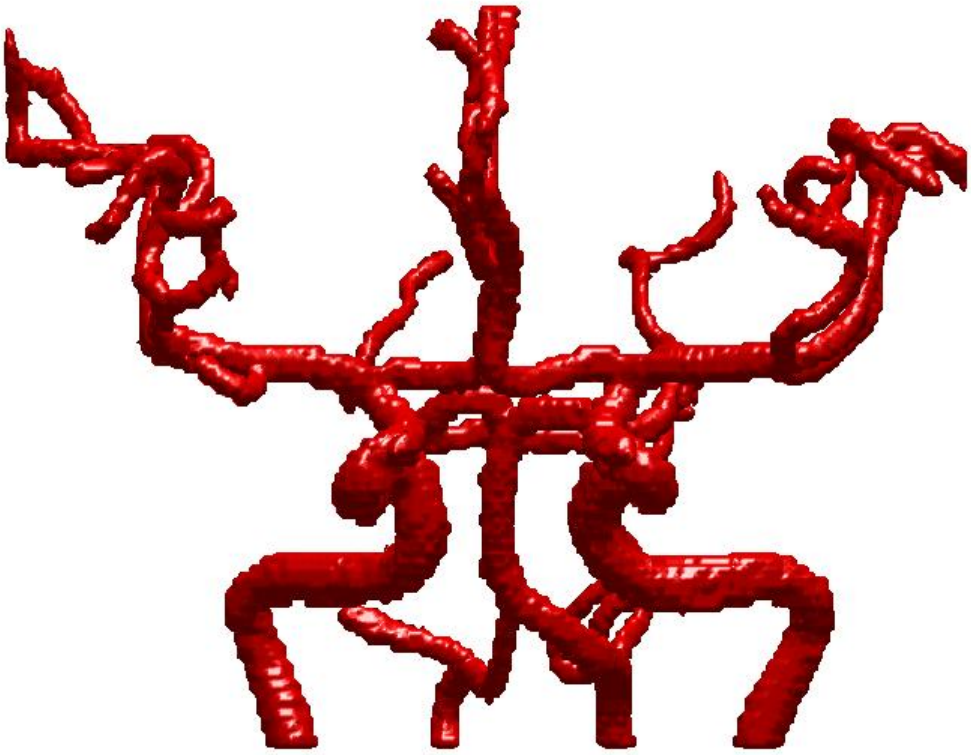
As can be seen from the figures, a visual comparison between our reconstruction results and the segmentation results gives us the first evidence that our proposed method can correctly represent the morphology and topology of vascular structures. In addition, very thin branches (**Figure 4.28**) and curved, complex structures (**Figure 4.29** and **Figure 4.31**) can be reconstructed faithfully using our method.



(a)

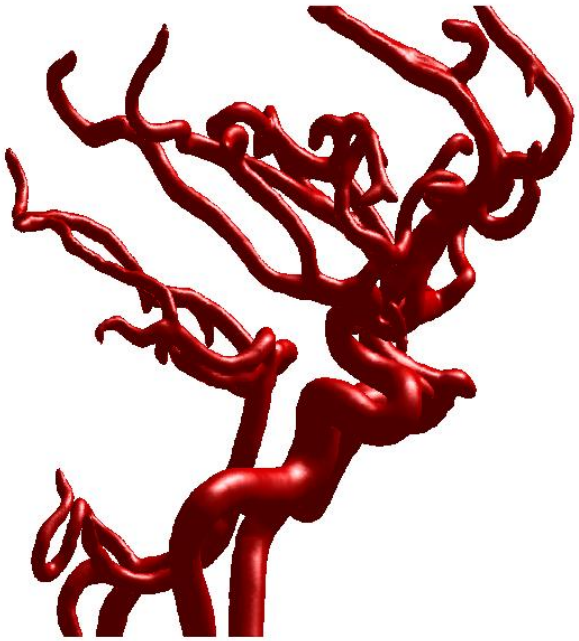
(b)

Figure 4.28: *The reconstructed result of CTA carotid artery.* **(a)** *The isosurface rendering of segmentation result.* **(b)** *The reconstruction result using our method.*



(a)

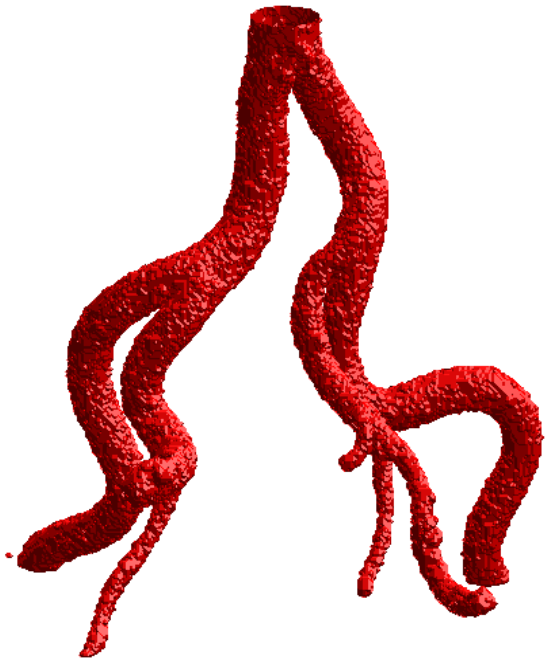
(b)



(c)

(d)

Figure 4.29: *The reconstructed results of MRA cerebral vessels.* **(a)** *The isosurface rendering of segmentation result.* **(b)(c)(d)** *The reconstruction result using our method.*



(a)

(b)

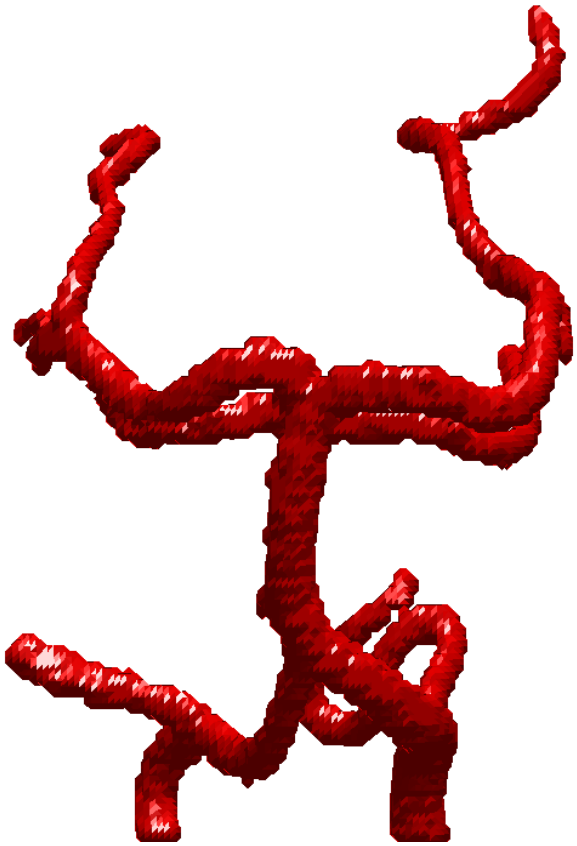
Figure 4.30: *The reconstruction of MRA abdominal aorta.* **(a)** *The isosurface rendering of segmentation result.* **(b)** *The reconstruction result using our method.*

(a)

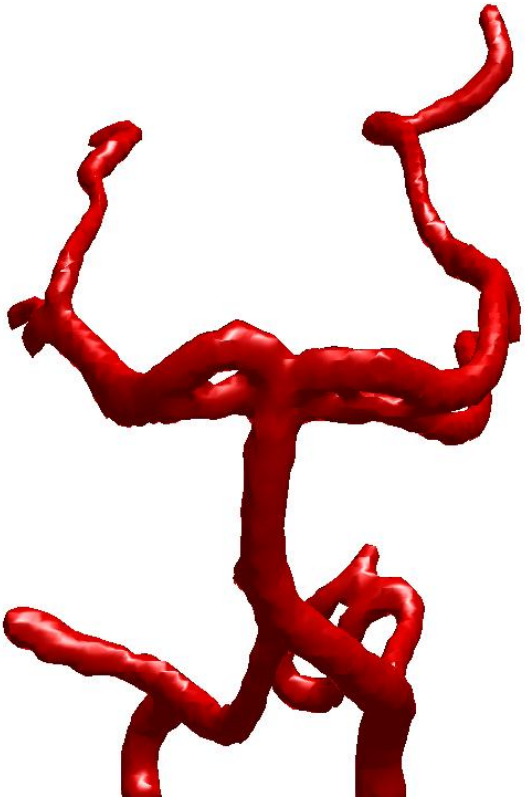
(b)

Figure 4.31: *The reconstruction of liver portal vein.* **(a)** *The isosurface rendering of segmentation result.* **(b)** *The reconstruction result using our method.*

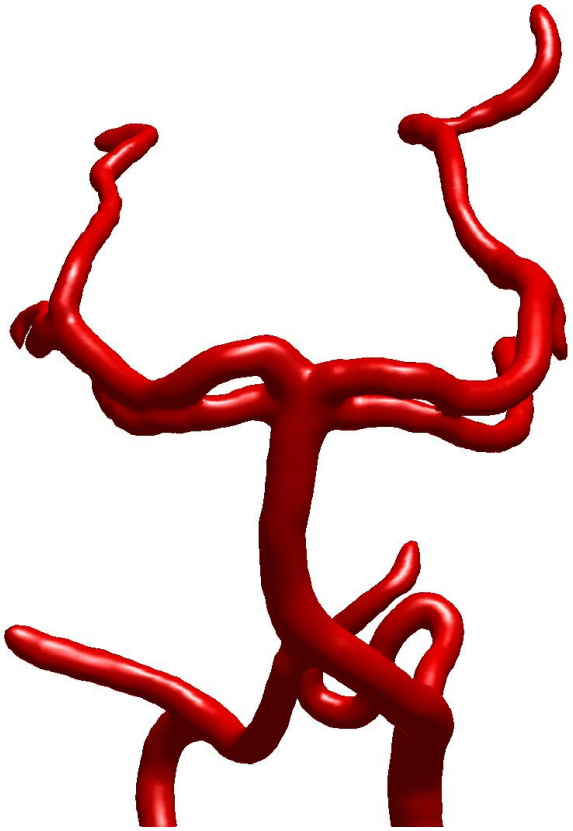
For further comparison, **Figure 4.32** shows a detailed look at the reconstruction of the MRA cerebral vessels. As presented in the figure, the isosurface rendering of segmentation result **(a)** suffers from strong aliasing artifacts like staircases. It therefore diverges considerably from the real vessels and might hamper the visual interpretation of the vessel surface **(Schumann et al., 2007)**. Compared to the isosurface rendering of segmentation result, the method based on MPUI can construct a smooth surface with a certain visual quality **(b)**. However, due to the complex and fine nature of most vascular trees, the quality of the resulting surfaces is still not good enough and has to be improved using an additional remeshing step **(Schumann et al., 2008)**, based on either parameterization, or fitting of subdivision surfaces **(Boubekeur et al., 2006)**, which subsequently increases the effort required from the reconstruction process and inevitably introduces further errors. In contrast, our approach can achieve superior visual quality and produce smooth transitions at branchings **(c)**. Indeed, our method can guarantee as high a geometric continuity as the reconstruction method with convolution surfaces **(Oeltze and Preim, 2005)** does. Furthermore, we can achieve more accurate vessel surfaces since our method is without model assumptions. Generally, our method can achieve quite smooth and accurate vessel surfaces. Compared with the direct visualization of the segmented result, the vascular structures reconstructed by our method are a more faithful approximation to the real vessels.



(a)



(b)



(c)

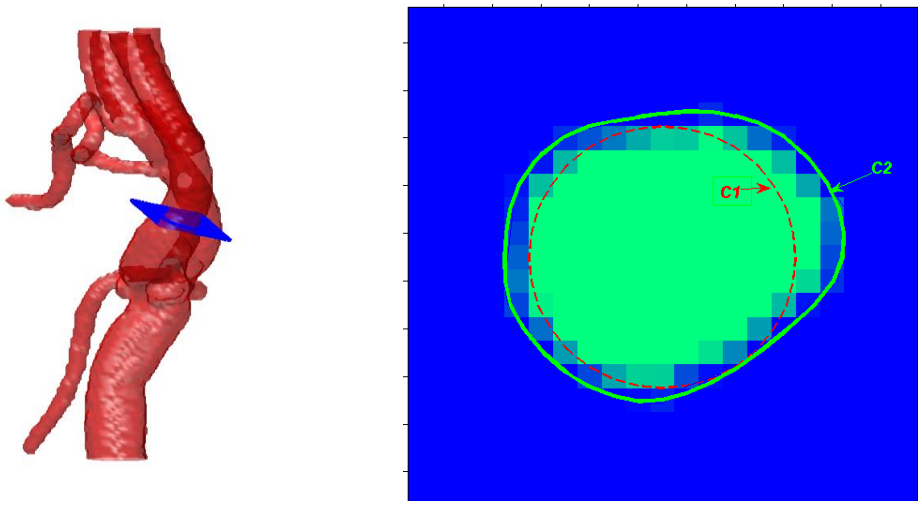
Figure 4.32: *A detail look at the reconstruction of the MRA cerebral vessels.* **(a)** *The isosurface rendering of segmentation result.* **(b)** *The reconstruction result using MPUI.* **(c)** *The reconstruction result using our method.*

4.3.5.2 Validation

The validation of any reconstruction technique is crucial for its clinical applications **(Oeltze and Preim, 2005)**. The underlying data (segmented vessel dataset) used for validating the proposed technique is generated with the vessel segmentation methods. Our goal is to reconstruct a smooth vessel surface from a segmented point set and verify how accurately it can approximate the "original" vessel surface.

1. Qualitative validation

Generally, our proposed approach has two key ingredients to guarantee requirements of both accuracy and smoothness. First and foremost, the cross-sections of the vessel are freely specified by 2-D implicit splines, without any model assumption, such as circular or elliptical shapes. The freeform specification of the vessel cross-sections is the essential requirement for computer aided vessel diagnosis, since the cross-sections of vascular geometry are not always circular, especially for those pathologic vessels.



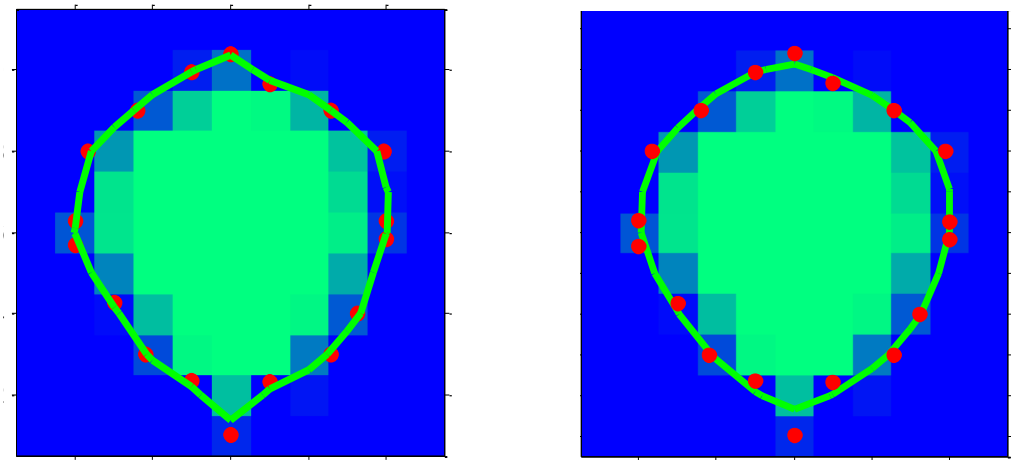
(a) **(b)**

Figure 4.33: **(a)** *The extraction of vessel cross-section from real medical data.* **(b)** *The specification of cross-section contour by circular shape and our method.*

As can be seen from **Figure 4.33**, the technique based on our method can represent the cross-section of vessel more accurately. In fact, the implicit curve constructed using 2-D implicit splines can achieve any preset precision, as long as the smoothing parameter δ is sufficiently small. In addition, by choosing the parameter of continuous degree m, the implicit curve can achieve any required continuity $C^{m-1}(m \geq 2)$. In

Figure 4.33(b), the dashed red circle $C1$ represents the circular contour of cross-section, and the solid green curve $C2$ represents the accurate and smooth contour of cross-section specified by our method.

However, this does not mean that the smoothing parameter δ should be set as small as possible. Because for the same resolution, if δ is set too small, the constructed curve as well as the surface would not be as smooth as the real vessel surface in visual inspection, especially in the case of reconstructing from a noisy data set. **Figure 4.34** demonstrates the two different implicit curves specified using a 2-D piecewise algebraic spline $B^{(3)}_{\Delta,\delta}(x, y)$ with the same set of contour points extracted from the vessel cross-section. The implicit curve presented in **(a)** is constructed with $\delta = 0.4$, which is much closer to the "original" point sets, but with less smoothness. Although the smoothness can be improved by increasing the spatial resolution, it would cost much more time to display the cruve/surface. On the other hand, the implicit curve presented in **(b)** is constructed with $\delta = 0.8$, which is not so close to the "original" point sets as, but it is much smoother than that of curve in **(a)** displaying on the same resolution to represent the real vessel cross-section. Therefore, to some extent, a trade off needs to be made between the faithfulness of the reconstructed surface to the "original" point sets and the smoothness of its appearance. Generally, the δ is set as 0.8, which can guarantee a reasonable trade-off between accuracy and smoothness.



(a) $\delta = 0.4$ **(b)** $\delta = 0.8$

Figure 4.34: *Two different implicit curves specified using 2-D piecewise algebraic spline* $B^{(3)}_{\Delta,\delta}(x,y)$ *with the same set of coutour points extracted from the vessel cross-section.*

The other key ingredient is that the partial shape preserving (PSP) spline basis functions **(Li and Tian, 2011)** is employed to smoothly combine the collection of implicit control surfaces. As stated in **(Li and Tian, 2011)**, the PSP-spline basis function has all the advantages of the conventional B-spline technique, and it is a kind of shape-preserving spline, which can preserve the original cross-section surfaces as much as possible. In addition, it is flexible to adjust the smoothing parameter, which can be used for controlling the blending range, to achieve a satisfactory reconstruction result. **Figure 4.35** shows the blending of a set of cross-sections $C_i(x,y)$ with different PSP-spline basis functions $B_i(z)$. As shown in the figure, from left to right, the controlling parameter δ in $B_i(z)$ is changed from 0.5 to 1.1 with an increasing interval 0.2.

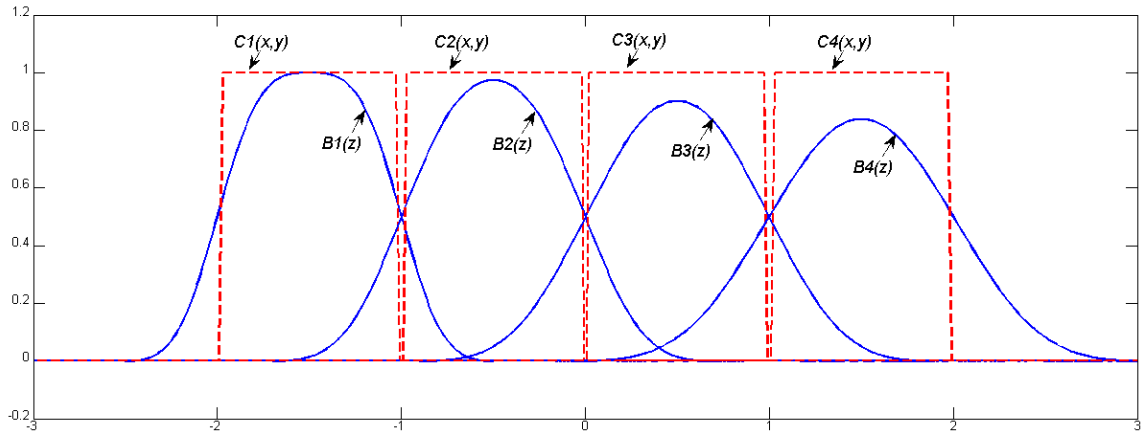


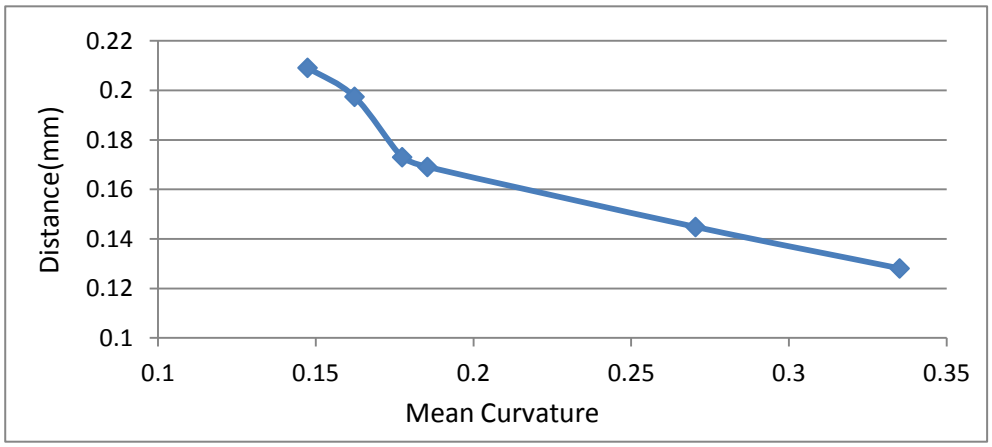
Figure 4.35: *The blending of a set of implicit cross-sections* $C_i(x,y)$ *with different PSP-spline basis functions* $B_i(z)$.

2. Quantitative validation

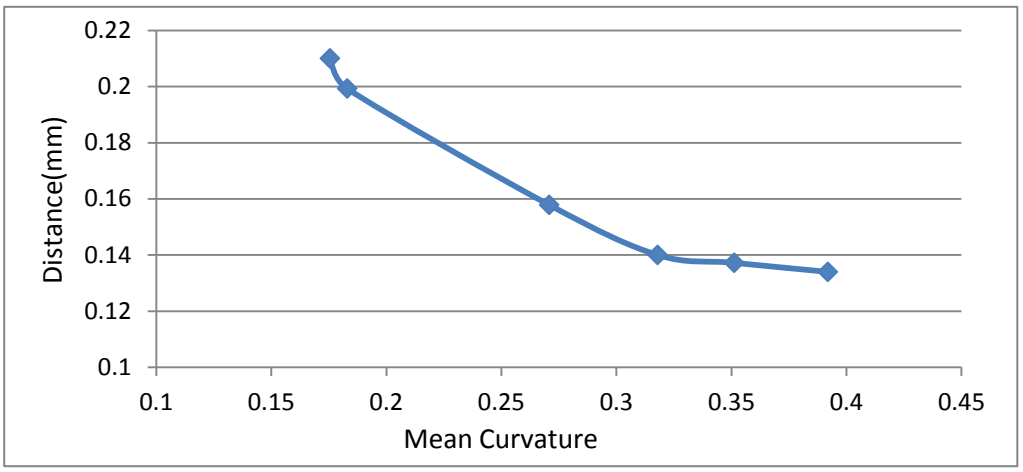
Besides visual inspection, quantitative validation analysis is necessary for estimating whether the underlying data (segmented vessel dataset) are faithfully represented with a certain smoothness **(Oeltze and Preim, 2005)**. For judging the accuracy of reconstruction, we analyse the distances between segmented data and our reconstructed surface. The comparison is realized in the following way: for each vertex on the isosurface of the segmentation result, the Euclidean distance to its closest point on our reconstructed surface is calculated. For estimating the smoothness, we examine the mean curvature of each vertex on the reconstructed surface, since the mean curvature has the ability of getting insight to the degree of smoothness of the surface **(Dong and Wang, 2005)**.

Curve in **Figure 4.36** illustrates the joint distribution of the measured distances and mean curvatures for our experimental datasets. The vertical axis represents the mean distances of each vertex on the isosurface of the segmentation result to our

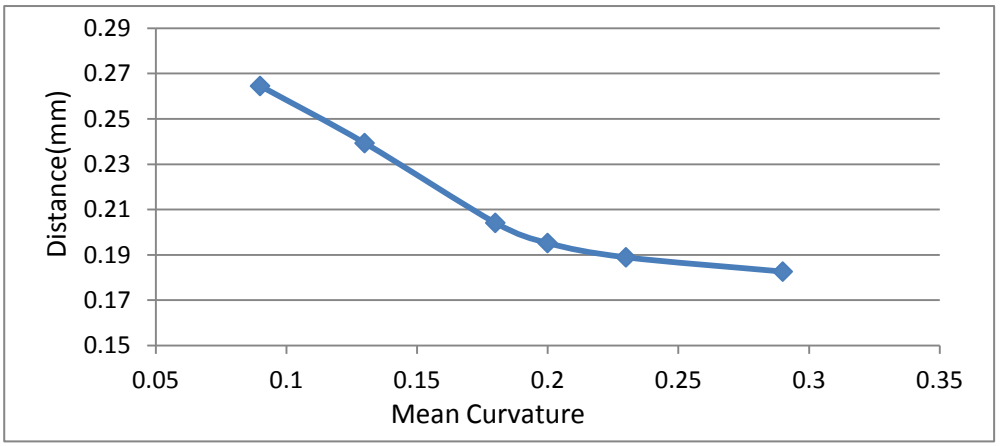
reconstructed surface; and the horizontal axis represents the average of the unsigned mean curvatures (AUMC) for all vertices on the reconstructed surface. As demonstrated in the curves, the minimum deviation can be as small as $0.12mm$ with certain smoothness (the AUMC is less than 0.4), and the maximum deviation cannot be bigger than $0.26mm$ even the level of smoothness is set quite high. As discussed in the last section, the accuracy and the smoothness of the reconstructed surface from the discrete point sets needs to be balanced. Generally, our method can achieve highly accurate reconstructed surface, of which the mean distance is generally as small as $0.20mm$, which is much smaller than the mean distance ($0.39mm$) presented in **(Oeltze and Preim, 2005)** and the median of the deviations ($0.30mm$) presented in **(Schumann et al., 2008)**. Furthermore, the average deviation is much less than the mean diagonal voxel size ($1.28mm$) of the used datasets ($0.96mm$ for the CTA carotid artery dataset, $1.05mm$ for the MRA cerebral vessel dataset, $1.17mm$ for the MRA abdominal aorta dataset and $1.94mm$ for the segmented liver portal vein). In addition, the surfaces are reconstructed with quite satisfied smoothness, not only in visual inspection but also the quantitative analysis (the AUMC is generally as small as 0.207).



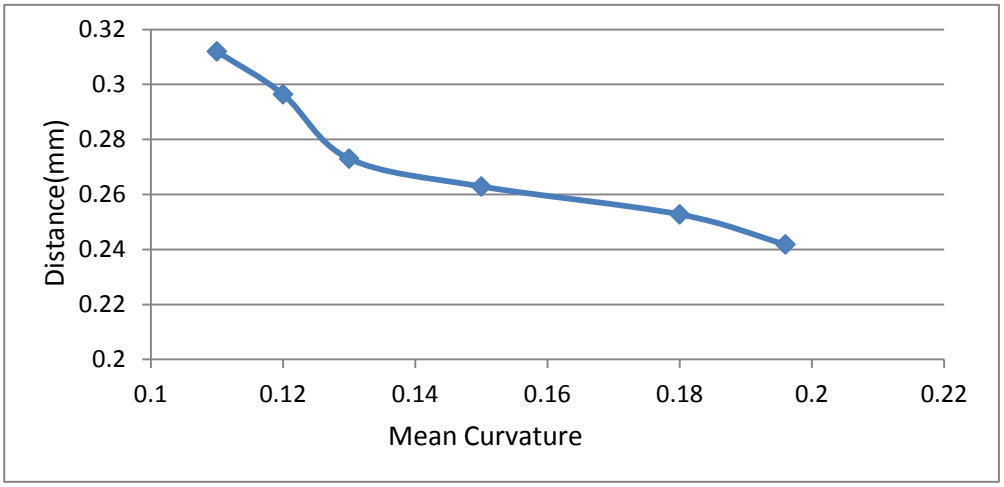
(a)



(b)



(c)



(d)

Figure 4.36: *Quantitative analysis for our reconstructed surfaces. The correlation between the measured distances and mean curvatures for: the carotid artery constructed from CTA dataset* **(a),** *the cerebral vessels constructed from MRA dataset* **(b),** *the abdominal aorta constructed from MRA dataset***(c),** *the liver portal vein from binary segmentation result* **(d).**

The quantitative comparison has also been conducted between the surfaces reconstructed using our method and that based on MPUI method. As have been shown in **Table 4.1**, the average distances and AUMC corresponding to the surfaces built from our method are much smaller than that for the surfaces reconstructed based on the MPUI method for each experimental dataset. In other words, our method can achieve more accurate and smoother vessel structures than the MPUI based method.

Table 4.1: *Quantitative comparison between our reconstructed surfaces and that based on MPUI method. Distances are given in millimetres. The accuracy is measured by calculating the average distances between the segmentation result to the reconstructed surface, and the smoothness is estimated by measuring the average unsigned mean curvature (AUMC) of the reconstructed surface.*

Datasets	**MPUI based method**		**Our method**	
	Average Distance	**AUMC**	**Average Distance**	**AUMC**
CTA carotid artery	**0.27**	**0.29**	**0.17**	**0.21**
MRA cerebral vessels	**0.27**	**0.35**	**0.16**	**0.28**
MRA abdominal aorta	**0.44**	**0.27**	**0.21**	**0.19**
Binary segmentation result of Liver portal vein	**0.37**	**0.20**	**0.27**	**0.15**

4.3.5.3 Reconstruction for pathological vasculatures

Our technique has also been applied to the reconstruction of pathological vasculatures from the data sets of patient with vessel disease. The first example is the reconstruction of a carotid artery with stenosis for 3-D CTA images with a resolution of $512 \times 512 \times 100$ and spacing of $0.48\text{mm} \times 0.48\text{mm} \times 0.65\text{mm}$ (see **Figure 4.37**). As can be seen from the rectangle area, our algorithm can faithfully represent the stenosis of the carotid artery and achieve superior visual quality.

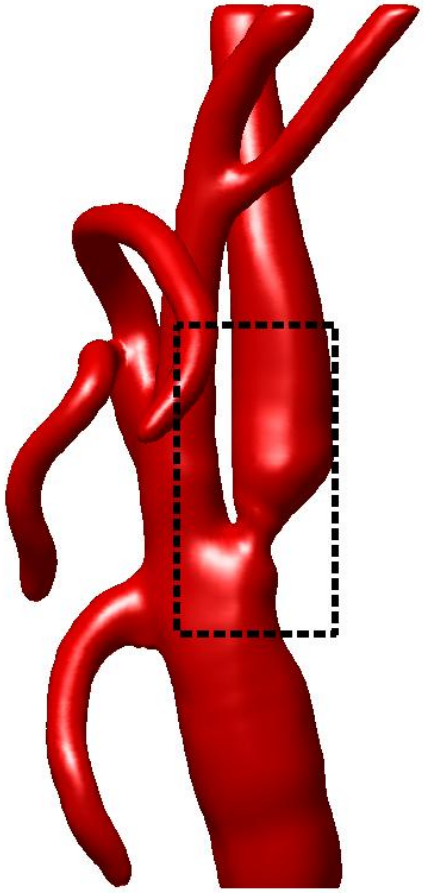


Figure 4.37: *The reconstruction of carotid artery with stenosis.*

Another example is the reconstruction of a peripheral artery with aneurysm for 3-D CTA images with a resolution of $512 \times 512 \times 100$ and spacing of $0.83\text{mm} \times 0.83\text{mm} \times 1.0\text{mm}$ (see **Figure 4.38**). As shown in the rectangle area, our technique can accurately reconstruct the peripheral artery with aneurysm in the form of implicit functions, which would be helpful to perform the quantitative analysis for the pathological vessel.

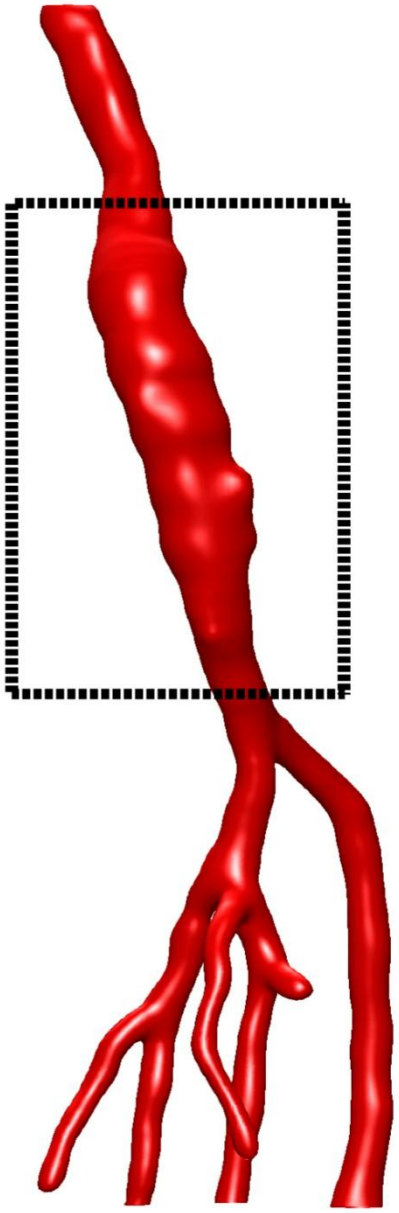


Figure 4.38: *The reconstruction of peripheral artery with aneurysm.*

4.4 Summary

This chapter has presented a novel technique for the accurate reconstruction of vasculatures without model-assumptions, which is underpinned by a skeleton-based

implicit generalized cylinder modelling method. With the proposed method, an implicit surface with extremely high smoothness and accuracy can be reconstructed from a given segmented dataset. The visual inspection of the experimental results illustrate that our reconstructed vasculatures are faithful to the actual vessel structures. In addition, qualitative and quantitative analysis has been conducted to validate the accuracy and smoothness of the reconstructed results.

The implicitly reconstructed continuous geometry of vasculatures not only allows to achieve high-quality visualization, but can also be very useful for the vessel shape analysis. Briefly, our reconstructing method has three main advantages over other vascular surface reconstruction techniques. Firstly, based on the extracted contour points, the cross-section of the vessel can be accurately defined by the 2-D piecewise algebraic splines, and need not to be either a circle or similar shape. Secondly, the modelling results of our method can achieve any required smoothness since we inherit the weighted summation mechanism to generate a surface with different cross-sections for the same skeletal element, and utilize a smooth blending operation to blend the branches of implicit surfaces for different skeletal elements. Thirdly, the reconstructed shapes based on our method have the superiority of scalability. That is, the reconstructed vasculatures can be rendered at any specified resolution, as there is an analytical representation for the resulting object.

Chapter 5
GPU Accelerating Techniques for Rendering Implicitly Represented Vasculatures

5.1 Introduction

When the vascular structures are represented as implicit functions, the display of reconstructed vasculatures becomes a problem of rendering implicit surfaces. Since an implicit function does not represent a surface explicitly, that is, the points on the surface cannot be generated by substitution **(Bloomenthal, 1995)**, implicitly represented geometric objects require a relatively high computational cost to display. As a consequence, it is necessary to investigate the techniques of hardware acceleration for achieving higher performance of rendering implicitly represented objects.

As discussed in Chapter 2, modern GPUs have a highly distributed architecture with massively parallel processors and memories, which makes them not only very efficient in processing graphics information, but also as a sort of general purpose graphics processor for effectively processing a range of complex computational tasks. A broad range of applications and tasks have been successfully implemented on modern GPUs, including the areas of signal and image processing, global illumination, geometric computing, databases and data mining, physically based

simulation **(Owens et al., 2007)**. In particular, the highly parallel structures of the GPU have made graphics hardware an attractive target for computer vision applications. For instance, in **(Yang and Pollefeys, 2005)**, a real-time system for stereo depth extraction from multiple images has been implemented on GPUs, which can perform more than four times faster than a comparable CPU-based commercial system. A real-time system for computing and visualizing motion on 640×480 25 Hz 2-D image sequences using graphics hardware has been presented in **(Strzodka and Garbe, 2004)**. Three 3-D reconstruction algorithms **(Xu and Mueller, 2005)** have been implemented on programmable graphics hardware, which can achieve high-quality floating-point 128^3 reconstructions from 80 projections in time frames from seconds to tens of seconds. Thus, by taking advantage of modern high performance GPUs, the computational time of rendering implicitly represented vascular structures can be reduced significantly.

In the rest of this chapter, the techniques of GPGPU programming and implicit surface rendering are first investigated; then a GPU accelerating technique is devised and applied to achieve high performance of rendering our constructed implicit functions representing vasculatures. The relevant performance analysis is conducted at the end of this chapter.

5.2 Related Work

5.2.1 GPGPU Programming and CUDA

The modern programmable GPUs are initially designed for processing graphics tasks using shader languages such as Cg, GLSL and HLSL. Over the past few years,

graphics hardware has evolved into a kind of general-purpose GPU (GPGPU), which is capable of executing increasingly costly and complex algorithms for general purposes.

The highly parallel programmability has increased the GPGPU's attractiveness as a platform for science and engineering applications, such as computational visualization in medicine, computational finance, bio-informatics and life sciences. Various programming environments have been developed for GPGPU computing **(Strengert et al., 2008)**, such as Brook **(Buck et al., 2004)**, Glift **(Lefohn et al., 2006)**, Scout **(Mccormick et al., 2007)**, and so on. Nowadays, NVIDIA's CUDA (Compute Unified Device Architecture) **(NVIDIA, 2007)** is probably the most commonly used high-level GPGPU language.

CUDA is a general purpose parallel computing architecture, which enables developers to write programs to solve complex computational problems by leveraging the parallel compute engine in NVIDIA GPUs. From the application developer's perspective, a typical CUDA program contains two phases that are executed on either the host (CPU) or the device (GPU) **(Kirk and Hwu, 2010)**. The phase that exhibits no or little data parallelism is implemented in host code, and the other phase that exhibits a large amount of data parallelism is implemented in the kernel (device) code. Generally, the developer supplies a single source program encompassing both host and kernel code, which is separated by the NVIDIA C Complier (NVCC). The host code is compiled with the host's standard C/C++ compilers and runs as an ordinary process; while the kernel code is compiled by the NVCC and executed on the GPU device. The data transformation between the host and device is implemented in the host code via API calls **(Stone et al., 2008)**.

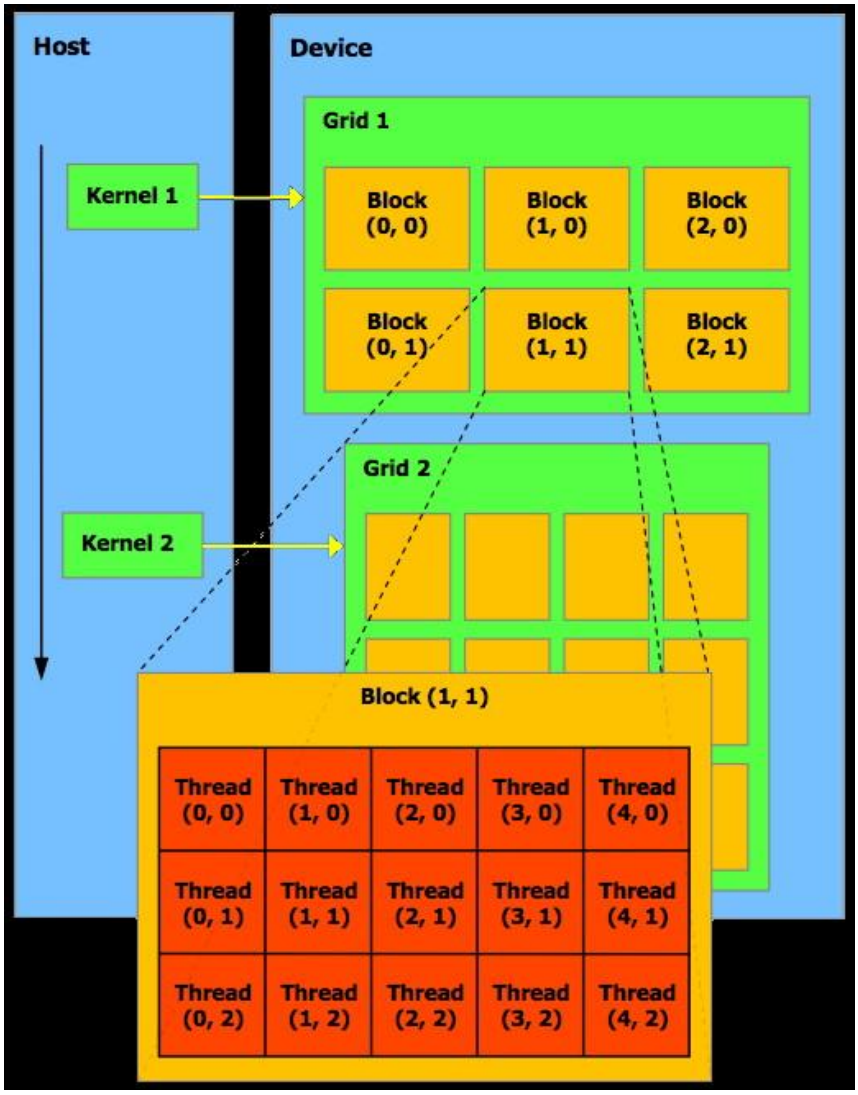


Figure 5.1 : *CUDA thread model.***(NVIDIA, 2007)**

Threads in CUDA environment are extremely lightweight threads. A single device may run thousands of threads in parallel, where each thread executes a given kernel. A number of threads can be bundled in a thread block, and then a number of thread blocks can be bundled in a grid, which is assigned to a kernel (see **Figure 5.1**). When a kernel is invoked by the host application, it is executed as a grid of parallel threads **(Kirk and Hwu, 2010)**. Then the dimensions of the grid and blocks containing the threads have to be specified, which can be one, two or three dimensional. Furthermore, a list of parameters may be passed to a kernel, including input and

output stream recalling the stream programming model. Since each thread in a grid of thread blocks executes the same kernel, parallelization is achieved by processing different elements of the input stream **(Ortner, 2008)**.

5.2.2 Rendering Techniques of Implicit Surfaces

Generally, there are two ways of rendering implicit surfaces. One approach is to render directly by ray tracing **(Hart, 1993)** or ray casting **(De Cusatis Jr et al., 1999)** . In this kind of approaches, a ray from an eye is defined for each pixel of an image, and then the intersection between the ray and the isosurface is computed. A comparatively high-quality image can be achieved, since the shading colour of each pixel on the image is calculated based on geometric optics simulation **(Kanai et al., 2006)**. The main challenge is how to efficiently find the first ray-surface intersection point when the implicit surface function is too complex or is not given in an explicit form **(Liktor, 2008)**. In addition, this kind of technique has a high computational cost, since it has to traverse the entire dataset each time when an image is rendered. Thus, the GPU-accelerated ray-tracing of implicit surface has been investigated for different types of surfaces **(Fryazinov and Pasko, 2007)**, such as the ray-tracing of implicit surfaces defined by radial based functions **(Corrigan and Dinh., 2005)**; the rendering of quadratic implicit surfaces on GPU **(Lessig, 2004)**; the effective ray-tracing on GPU for objects represented by CSG-trees **(Toledo and Levy, 2004)**; and the ray-casting of discrete isosurfaces **(Hadwiger et al., 2005)**.

Although ray-tracing based methods can produce high quality images, they are unable to create concrete surface representation, which would be useful for performing non-imaging operations on the represented objects, such as moving from

one surface point to another, positioning objects upon a surface and defining texture along the surface **(Bloomenthal, 1995)**. The other approach for displaying an implicit surface is to approximate the surface using a polygonal mesh, which can be produced using methods such as Marching Cubes **(Lorensen and Cline, 1987)** or Marching Tetrahedrons **(Gueziec and Hummel, 1995)**. In this kind of approaches, an implicit function is firstly sampled and evaluated on a regular grid. Then based on the pre-define grid, the zero-set contour is extracted and assembled into polygons. This can require high computational cost for the conversion of a functionally specified implicit surface to a polygonal approximation, especially when the function is quite complex or the resolution of the grids is high. However, polygonization is still considered to be more efficient than ray-tracing when visualizing an implicit surface **(Bloomenthal, 1995)**, because most graphics hardware is extremely efficient at processing polygons and the polygonization of implicit surfaces is a direct way of exploiting this advantage of GPUs. Some research has been done regarding the GPU-accelerated polygonization of implicit surfaces. In **(Kipfer and Westermann, 2005)**, the isosurface extraction process was re-formulated in a way that reduced both numerical computations and memory access operations by considering intrinsic features of recent GPUs. Johansson and Carr **(Johansson and Carr, 2006)** improved existing GPU-based isosurface acceleration by caching the topology on the graphics card and using a vertex program to perform geometric interpolation. Tatarchuk *et al.* **(Tatarchuk et al., 2008)** extracted the isosurfaces from medical datasets in polygonal form by taking advantage of Microsoft DirectX 10 based on GPUs.

5.3 Implementation

In developing a computer aided vascular surgery system, we are required not only to see vessel structure but also to feel or touch it. In this situation, the concrete surface representation of the actual vascular geometry is essential. Hence, we choose the technique of polygonization to rendering implicitly represented vasculatures.

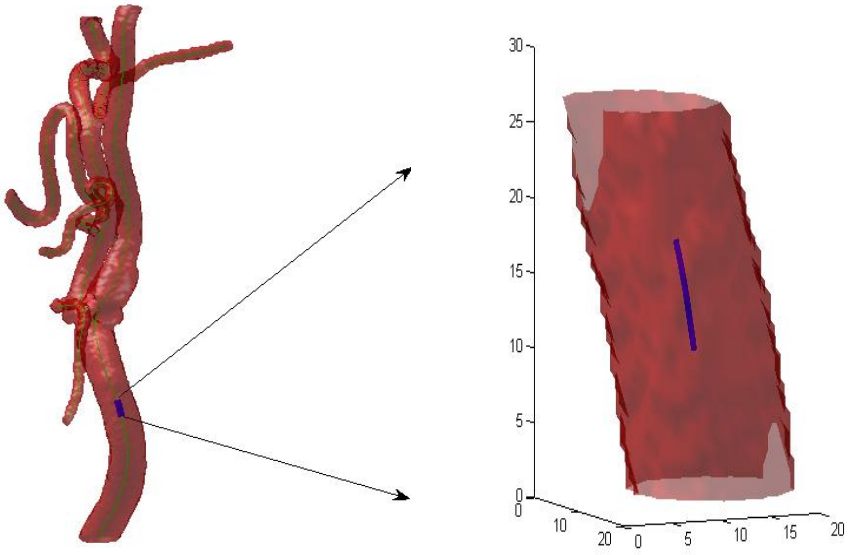


Figure 5.2： *The construction of axis-aligned bounding box.*

In our reconstruction technique, the vessel tree is represented as a global implicit function $F(x,y,z)$, which is the combination of hundreds of local implicit functions $f_i(x,y,z)(i=1,2,3,...,L)$. The evaluations of all of the local implicit functions in certain regular grid cells could require considerable computation. Fortunately, though each local implicit function is defined in the global domain, its influence is restricted within a certain range. Thus, for each local implicit function, its corresponding axis-aligned bounding box ($AABB$) is computed in voxel

coordinates (see **Figure 5.2**). Then the evaluation of the local $f_i(x, y, z)$ can be performed just within its $AABB_i$, not the global extent, which could greatly reduce the computational time. Finally, all of the $AABBs$ containing the evaluated function values are combined together to construct the global grid cell volume with sampled function-values for polygonalization (see **Figure 5.3**).

Each local implicit function $f_i(x, y, z)$ contains a 2-D piecewise algebraic spline function $B_{\Delta,\delta}^{(m)}(x, y)$, which is a complex piecewise polynomial and requires about $\mathrm{n}\times\left(20+40\times\mathrm{m}^2+36\times\mathrm{m}^3\right)$ operations for computation, where n is the vertex number of the control polygon and m is the degree of smoothness of the required bivariate function. Suppose the number of voxels in an $AABB$ is K, then there would be $\mathrm{K}\times\mathrm{n}\times\left(20+40\times\mathrm{m}^2+36\times\mathrm{m}^3\right)$ operations for computing the 2-D piecewise algebraic spline function $B_{\Delta,\delta}^{(m)}(x, y)$, which could cost several seconds for current CPU. Fortunately, the evaluation of an implicit function in an $AABB$ can be done on the GPU, as it contains substantial data-parallelism. The function value at any voxel depends only on the values of sample points, but no elements of $f_i(x, y, z)$ depend on any other elements. That is, all elements of $f_i(x, y, z)$ can be evaluated independently in a parallel way. Therefore, it is appropriate to utilize the parallel structures of modern GPUs for the concurrent evaluation of the local implicit function on each voxel in its corresponding $AABB$.

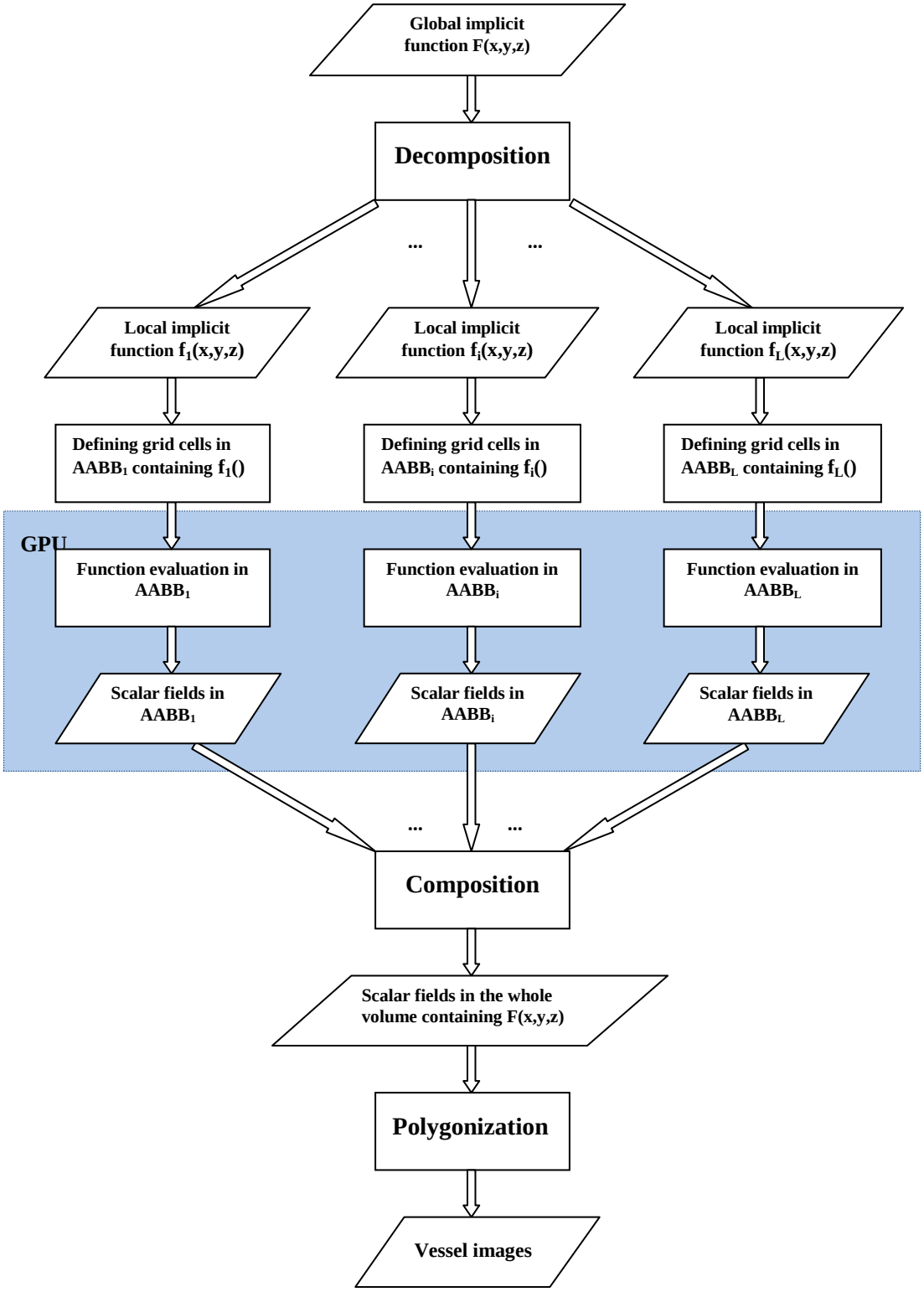


Figure 5.3: *Flowchart for the rendering of our constructed implicit functions representing the vessel structure.*

In our approach, we employ CUDA as our accelerating tool, as its single program, multiple-data (SPMD) programming model enables us to easily implement data-parallel algorithms for high-performance applications. Each voxel in the $AABB$ has the same evaluated formula for function value. Thus, for each voxel, CUDA can assign a thread for its function evaluation. All of the threads can be executed concurrently on the GPU, which can greatly improve the performance.

As shown in **Figure 5.4** function *Evaluation()* is the main program in host code. The main function first allocates the data matrices on the device and performs I/O to read input data from CPU. It also initializes the thread number for each block. And then, it invokes the kernel function, *ker_Evaluation()* to perform function evaluation on GPU. Finally, it performs I/O to transfer the resulted matrix back to the host and free the allocated matrices. **Figure 5.5** shows the kernel function *ker_Evaluation()*, which specifies the code to be executed by all threads of a parallel phase. Since all threads execute the same kernel code, the thread indices are required to distinguish different threads and direct themselves towards the particular parts of the data structure they are designated to work on **(Kirk and Hwu, 2010)**. As illustrated in **Figure 5.1**, threads executing on NVIDIA GPUs are generally organized into a two-level hierarchy. At the top level, each kernel creates a single grid, which consists of many thread blocks. All thread blocks must have the same number of threads organized in the same manner, and then each thread block is in turn organized as an array of threads numbering up to 512 **(Kirk and Hwu, 2010)**. Finally, threads within a block are organized into warps of 32 threads. Each warp executes in SIMD fashion. In our kernel function, one thread performs the function evaluation for each voxel.

```
void Evaluation(float *x, float *y, float * ctrPoints, float *delta, float *n,// input
            float *FF, //output
            int numPoints,int numVoxels)
{
             float *x_gpu, *y_gpu,*FF_gpu,*ctrPoints_gpu;
              /* Create an input and output data array on the GPU */
            cudaMalloc((void **) &x_gpu, sizeof(float)*numVoxels);
            cudaMalloc((void **) &y_gpu, sizeof(float)*numVoxels);
            cudaMalloc((void **) &FF_gpu, sizeof(float)*numVoxels);
            cudaMalloc((void **) &ctrPoints_gpu, sizeof(float)*2*numPoints);
            cudaMemcpy(x_gpu, x, sizeof(float)*numVoxels, cudaMemcpyHostToDevice);
            cudaMemcpy(y_gpu, y, sizeof(float)*numVoxels, cudaMemcpyHostToDevice);
            cudaMemcpy(ctrPoints_gpu, ctrPoints, sizeof(float)*2*numPoints, cudaMemcpyHostToDevice);

             int threads = 128;
            dim3 grid((numVoxels / threads)+1, 1, 1);
             // get around maximum grid size of 65535 in each dimension
             if (grid.x > 65535) {
                        grid.y = grid.x / 32768;
                        grid.x = 32768;
              }
             /* Call function on GPU */
             ker_Evaluation<<<grid, threads>>>(x_gpu,y_gpu,FF_gpu,ctrPoints_gpu, numPoints,delta[0], n[0]);
              /* copy result back to host */
             cudaMemcpy( FF, FF_gpu, sizeof(float)*numVoxels, cudaMemcpyDeviceToHost);

             cudaFree(FF_gpu);
             cudaFree(x_gpu);
             cudaFree(y_gpu);
             cudaFree(ctrPoints_gpu);
}
```

Figure 5.4: *The main program in host code.*

```
__global__ void ker_Evaluation(float* x_gpu, float* y_gpu,float* FF_gpu, float * ctrPoints_gpu,int
numPoints, float delta, float n)
{
        unsigned int blockId = __umul24(blockIdx.y, gridDim.x) + blockIdx.x;
        unsigned int i = __umul24(blockId, blockDim.x) + threadIdx.x;

        FF_gpu[i]=0.0;
        float x0,y0,x1,y1,xDir;
        for (int j=0;j<numPoints-1;j++)
        {
                x0=ctrPoints_gpu[j];
                y0=ctrPoints_gpu[j+numPoints];
                x1=ctrPoints_gpu[j+1];
                y1=ctrPoints_gpu[j+1+numPoints];
                xDir=x0-x1;
                FF_gpu[i]=FF_gpu[i]+((xDir > 0) ? 1 : ((xDir < 0) ? -1 : 0))*
                        LineSeg_imp(x_gpu[i],y_gpu[i], x0, y0, x1, y1, delta, n);
        }
        return;
}
```

Figure 5.5： *The kernel function in device code.*

5.4 Performance Analysis

Our method has been carried out on an AMD Athlon CPU 4600+, 2.41 GHz system with 2.00 GB RAM and NVIDA Geforce 8600 GT Graphics Card. The vessel branch shown in **Figure 5.6** is represented as a global implicit function $F(x, y, z)$, which is the combination of 40 local implicit functions $f_i(x, y, z)(i = 1, 2, 3, ..., 40)$. As stated in Section 3, $f_i(x, y, z)$ is a kind of complex piecewise polynomial and

requires considerable operations for computation. The evaluation of $f_i(x, y, z)$ in its $AABB_i$ could cost several seconds for current CPU, especially when the resolution of the $AABB$ is relatively high. In this example, the reconstruction time (including the time of point extraction, construction of implicit functions, evaluations of function values, and generation of polygons) executed totally on CPU could be 167.28 seconds, in which the cost of evaluating function values can be as high as 157.01 seconds. On the other hand, we can implement the function evaluations in parallel on GPU via CUDA, which takes only 4.06 seconds to evaluate all of the local implicit functions with the same resolutions. The total time has been reduced to 16.21 seconds, which is only one tenth of that totally based on CPU.

Figure 5.6: *An example of reconstructed vessel branch.*

For further demonstrating the strength of the concurrent computation on GPU via CUDA, some cases of the exact function-evaluation time based on CPU and GPU are listed. As illustrated in **Figure 5.7**, the horizontal axis represents the voxel number of the $AABB$ containing the implicit function and the vertical axis represents the time for evaluating the implicit function. The blue columns indicate the computing time implemented on CPU, which is generally 50 times as high as that implemented on GPU (indicated by red columns). Especially when the resolution of $AABB$ is high, the computing time executed on CPU could be as much as 10 seconds, while that based on GPU is only 0.2 seconds. Hence, the performance of rendering the implicit functions representing vessel structure can be greatly improved based on GPU via CUDA.

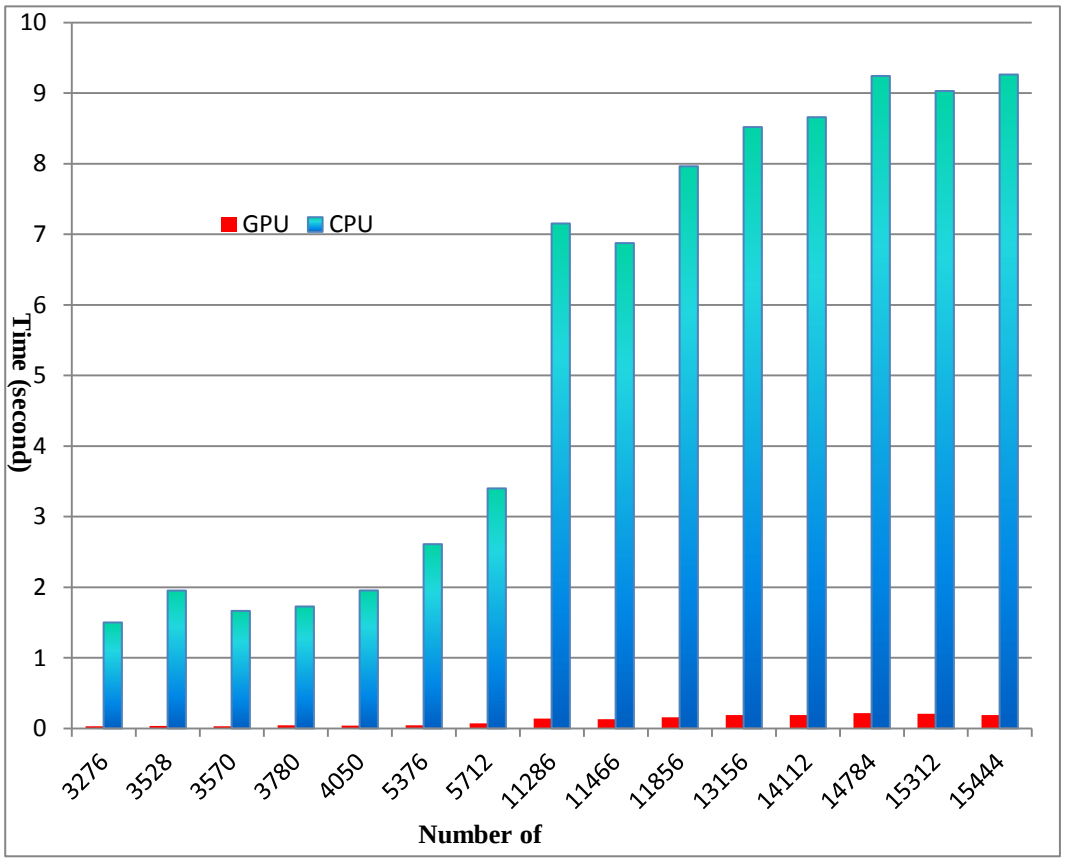


Figure 5.7: *The comparison of function-evaluation time based on CPU and GPU.*

To illustrate the advantages of using GPU, **Table 5.1** presents the information concerning the complexity of the resulting models and the reconstruction times of our implicit surfaces based on CPU and GPU. The reconstruction time includes the time of point extraction, construction of implicit functions, evaluation of scalar values, and generation of patch faces. As presented in the last column of the table, by taking advantage of programmable GPU via CUDA, the reconstruction time of our implicit surfaces is greatly reduced, and can be only one tenth of that implemented totally on CPU.

Table 5.1: *Performance measurements for the implicit surface reconstruction from anatomic vascular structures carried out on an AMD Athlon CPU 4600+, 2.41 GHz system with 2.00 GB of RAM and NVIDA Geforce 8600 GT Graphics Card.*

Datasets	**Resolutions**	**Number of triangles**	**Reconstruction time (second)**	
			Totally based on CPU	**Based on CPU and GPU**
CTA carotid artery	512×512×206	133116	847.18	86.37
MRA cerebral vessels	352×448×114	157018	743.42	143.99
MRA abdominal aorta	512×512×310	155224	1063.39	165.30
Binary segmentation result of Liver portal vein	512×512×151	112980	1025.21	189.52

5.5 Summary

There are various ways to display an implicit surface. In this research, the technique of polygonization is used to render our implicitly represented vasculatures. With our technique, the reconstructed vessel tree is represented as hundreds of implicit functions, which could require considerable time for polygonization. Fortunately, CUDA enables us to easily implement parallel computing on programmable GPUs, which can greatly reduce the polygonization time of implicit functions. Experimental results demonstrate that the performance of rendering our implicitly represented vasculatures has been greatly improved by exploiting advantage of modern GPUs via CUDA.

Chapter 6
Virtual Angioscopy based on Implicit Vasculatures

6.1 Introduction

Virtual endoscopy is one of the most active areas in medical data visualization. It is a non-invasive diagnosis technique and has no direct deleterious effects on patients **(Bartz, 2005)**. It utilizes computers to process 3-D image datasets to provide simulated perspective view inside specific hollow organs **(Geiger and Kikinis, 1994; Rubin, 1996)**, which are similar or equivalent to those produced by standard endoscopic procedures **(Wickham, 1994)** . This technique has been applied to virtual colonscopy **(Hong et al., 1997; Bartrolí 2001)**, bronchoscopy **(Ferretti et al., 1996)**, ventriculoscopy **(Auer and Auer, 1998; Bartz et al., 1999a)**, and so on. Virtual angioscopy is a specialized virtual endoscopy technique for exploring the human vascular system **(Davis et al., 1996; Beier et al., 1997; Gobbetti et al., 1998)**, which generates an interactive environment for vascular examination from a point of view inside the vessels **(Bartz et al., 1999a)**. Virtual fly-through of vascular structures is a useful technique for educational purposes and some diagnostic tasks, as well as intra-operative navigation and intervention planning **(Preim and Oeltze, 2007)**. In such a virtual visualization system, it is usually essential to combine detailed perspective view inside the vascular structure with an overview of the vessel tree.

Virtual angioscopy requires a relatively high visual quality of perspective view inside the dataset for the purposes of training and diagnosis. One of the common approaches for the visualization of a virtual angioscopy is surface rendering, yielding images close to a real endoscopy. However, the direct application of surface rendering algorithms (i.e. Marching Cubes) **(Lorensen and Cline, 1987)** to the segmented vasculatures may suffer from the typical diamond-shape artifacts caused by trilinear interpolation **(Bartz, 2005).** Therefore, the smooth and accurate reconstruction of vascular tree is crucial for virtual angioscopy. In addition, the navigation of the virtual camera along the vessel structures is another key issue, which requires the operation of collision avoidance. This problem can be easily solved when the vasculatures are represented as implicit surface, since the reconstructed vessel is a connatural implicit volume, which is a favourite kind of geometric object when performing collision detection.

In the rest of this chapter, the major issues associated with virtual angioscopy are firstly investigated, including perspective rendering techniques and navigation paradigms. Then we implement a virtual angioscoy system based on our implicitly reconstructed vascular geometry, which can achieve high quality of perspective views inside the vessel as well as freely interactive navigation. Experiments are carried out to demonstrate the strengths of our technique.

6.2 Related work

6.2.1 Perspective Rendering Techniques

Generally, the rendering techniques for virtual endoscopy fall into two categories: surface rendering and volume rendering **(Bartz, 2005)**. Surface-based rendering techniques typically extract surfaces by fitting geometric primitives, such as polygons or patches, to constant-value contour surfaces in volumetric datasets **(Elvins, 1992)**. It has been widely applied to virtual endoscopy because of its high rendering speed. In addition, the extracted surface, containing the space-relation information, is very useful for the navigation of the virtual camera. Many current virtual endoscopy systems, such as VESA **(Lorensen et al., 1995)**, VirEn **(Nain et al., 2001)**, 3-D Slicer **(Bruckner, 2003)** and FreeFlight **(Vining et al., 1997)**, are based on the surface rendering technique. However, the general visual quality based on surface rendering is comparatively poor **(Bartz, 2005)**. In particular, the direct isosurface rendering for the segmented dataset may suffer from the diamond-shape artifacts caused by trilinear interpolation **(Bartz, 2005)**. Therefore, the smooth and accurate reconstruction of segmented results is crucial for surface rendering-based virtual endoscopy.

Compared to surface rendering, direct volume rendering is a technique for directly displaying 3-D sampled datasets in the form of 2-D projection, without the intermediate geometric primitive representations used in surface-based rendering techniques **(Elvins, 1992)**. Generally, the volume rendering techniques, such as ray casting **(Tuy and Tuy, 1984)**, splatting **(Westover, 1990)**, and shear warp **(Lacroute and Levoy, 1994)**, can generate higher visual quality results for virtual endoscopy.

However, the frame rate they can achieve is relatively low, which requires accelerating algorithms for real time rendering **(Bartz, 2005)**. Although the texture mapping-based volume rendering **(Cabral et al., 1994)** can achieve a relatively high frame rate, it needs the additional support of high-performance graphics hardware. Another drawback of volume rendering method is that, the generated image does not provide the spatial relation information, which is essential for the navigation of the virtual camera. There are several virtual endoscopy systems based on volume rendering, such as VI VoxelView **(Serlie et al., 2001)** and CRS4 **(Beier et al., 1997)**. However, in the case of virtual angioscopy, when the vascular structures are rather small, the direct application of volume rendering techniques is prone to produce severe aliasing artifacts **(Schumann et al., 2007)**. In addition, as stated in **(Nakajima et al., 2007)**, the volume rendering technique is not suitable for visualizing small anatomic structures, such as subependymal arteries, cranial nerves, and other lesions.

6.2.2 Navigation Paradigms

Besides rendering, the camera navigation paradigm is another key problem required to be solved for the development of a virtual endoscopy system **(Bartz, 2005)**. The interaction of the user to control the movement of the virtual camera inside simulated hollow organs is not an easy task. It requires operations of mapping the input device movements to camera parameter modifications **(Vilanova et al., 1999)**.

Various virtual endoscopy techniques can be roughly classified into three classes: automatic navigation, manual or free navigation and guided navigation **(Hong et al., 1997)**. With automatic navigation, the user defines a certain number of key frames camera parameters are specified. With pre-defined key frames, smooth camera

movements can be calculated automatically. The drawback of automatic navigation is the lack of interactivity. In other words, irrelevant regions cannot be easily skipped since there is limited user interaction **(Bartz, 2005)**. By manual navigation, the user can completely control all parameters of the virtual camera without any constraints. However, in order to keep the virtual camera inside the organs, collision avoidance is needed, which requires costly query operations. In addition, in contrast to automatic navigation, free navigation requires a highly interactive rendering technique to avoid a significant lag between interaction and rendering **(Bartz, 2005)**.

The guided navigation is between the two previous methods; it combines user guidance with an efficient collision-avoidance scheme **(Bartz, 2005)**. The user can control over the camera parameters but some constraints are added, such as keeping the position of the camera to the optimal path. For instance, Vilanova et al. **(Vilanova et al., 1999)** described a guided navigation system where the location of the camera was fixed to a pre-computed path, while the camera orientation could be selected freely. Another typical system is Hong et al.'s virtual voyage **(Hong et al., 1997)**, which employed several potential fields and kinematic rules to guide the virtual camera along the human colon. Although this kind of technique can achieve a higher frame rate because of the efficient collision-avoidance scheme, the generation of potential fields has the potential risk of changing the morphology of the anatomic structures. On the other hand, based on implicitly modelled vasculatures, collision avoidance for the camera navigation of virtual angioscopy can be easily solved without additional generation of potential fields, for the constructed model is a connatural implicit volume, which is a favourite kind of geometric object when performing collision detection **(Li and Tian, 2009)**.

6.3 Implementation

For our virtual angioscopy system, we basically follow the standard processing pipeline **(Bartz et al., 1999b)** (see **Figure 6.1**). Firstly, segmentation techniques are employed to extract the vessel structures from standard 3-D medical datasets, such as CT or MRA images. Secondly, in order to achieve high visualization quality, we adopt our skeleton-based implicit spline (SIS) modelling technique proposed in **Chapter 4** to accurately and smoothly reconstruct the vasculatures from the segmented discrete vascular surface points. Based on the implicit vascular geometry, the virtual camera can be automatically or freely navigated along the vascular tree and acquire a high-quality perspective view inside the vessel structures.

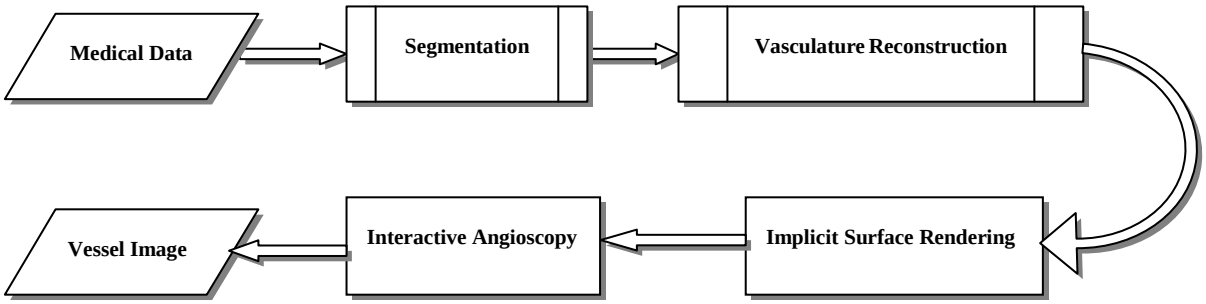


Figure 6.1 : *Pipeline for our virtual angioscopy system.*

As discussed in the above section, virtual angioscopy requires high visual quality of perspective views inside the dataset. The general surface rendering techniques for segmented datasets are not sufficient to generate high-quality visualization. On the other hand, by using the SIS modelling technique, the segmented vasculatures can be represented as an analytical implicit function. Based on this implicit function, we

can render high-quality vessel surfaces with any required continuity and smoothness. Besides the high-quality perspective views, we can also achieve accurate cross sections, since our method is without model assumptions.

For the navigation of camera, we can achieve efficient collision avoidance without the additional generation of potential fields, since the SIS method can guarantee to define an implicit volume by replacing the equality of Eq. 4.8 with an inequality:

$$F(x,y,z) = f(b,n,t) = f(X(x,y,z),Y(x,y,z),Z(x,y,z)) \geq 0 \qquad (6.1)$$

That is, the vasculatures are represented as a global implicit function $F(x,y,z)$. When $F(x,y,z)=0$, it represents the vessel surfaces; when $F(x,y,z)>0$, it represents implicit volume inside the vessel structures; and when $F(x,y,z)<0$, it represents the implicit volume outside the vessel structures. The implicit volume is a favourite kind of geometric object when performing collision detection **(Li and Tian, 2009)**. When the vasculatures are modelled as implicit volume, one can tell directly whether a point lies inside or outside the vasculatures and the problem of collision detection can easily be solved **(Lin and Gottschalk, 1998)**. In other words, $F(x,y,z)$ is a kind of signed distance function (SDF) **(Osher and Fedkiw, 2002)**, which guarantees that the closer the point to the vascular axe, the bigger the value of function $F(x,y,z)$.

Figure 6.2 illustrates the flowchart of the manual navigation of virtual camera inside the vasculature. By operating the 3-D mouse, the next camera position can be easily

calculated. And then, the implicit function value: $F(\text{nextCamPos})$ is tested. If $F(\text{nextCamPos}) > 0$, we suppose the new pre-computed camera position is inside the vasculature, and the camera is moved to the new pre-computed position and the current camera position is updated. Otherwise, the new pre-computed position is outside the vasculature, which requires the re-operation of the 3-D mouse.

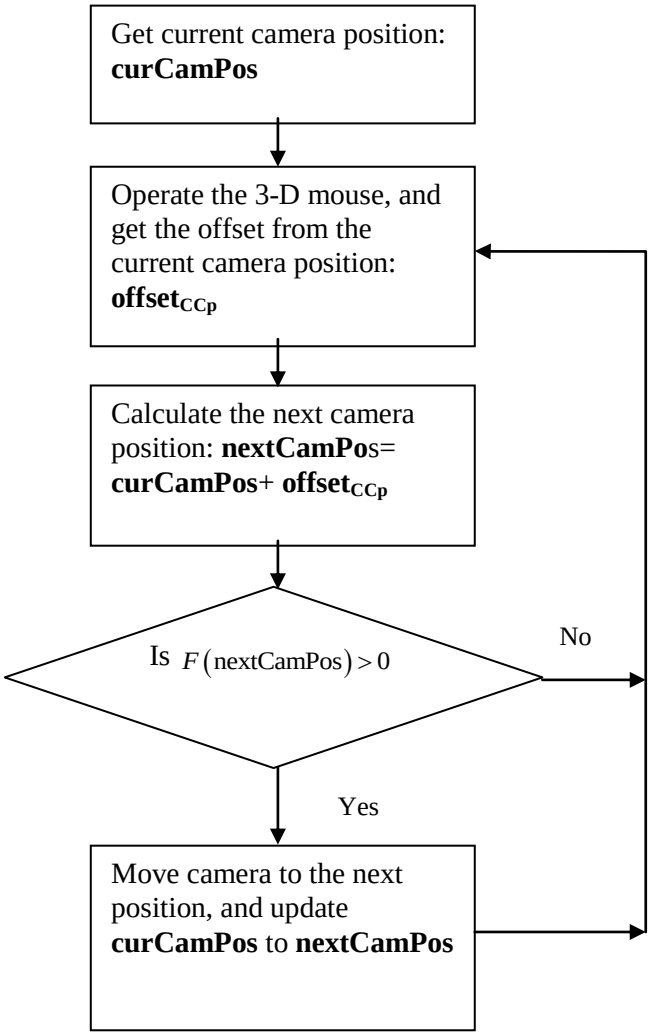


Figure 6.2 : *The flowchart of the manual navigation of virtual camera.*

6.4 Results

Based on the reconstruction of vasculatures using SIS, we can easily examine the interior of vascular system using the technique of virtual endoscopy.

The skeleton of the vessel has been extracted during the process of reconstruction (see **Figure 6.3**). Thus, it is convenient to implement the automatic navigation mode of virtual angisocopy using the pre-extracted skeleton as camera path (see **Figure 6.4**). The camera moves along the skeleton, and the target point is set to a fixed distance ahead on the skeleton so that the camera can smoothly follow each vessel segment.

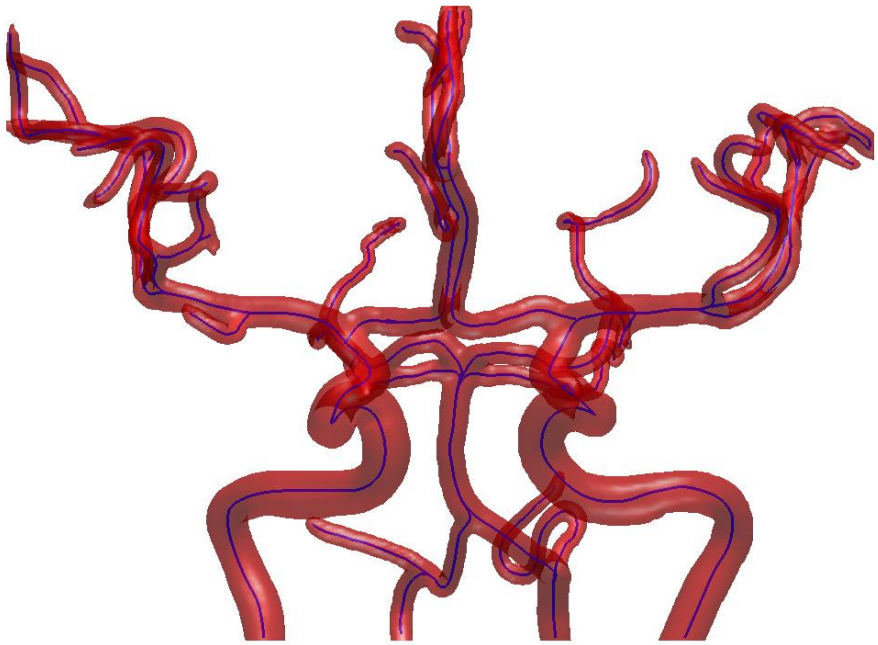
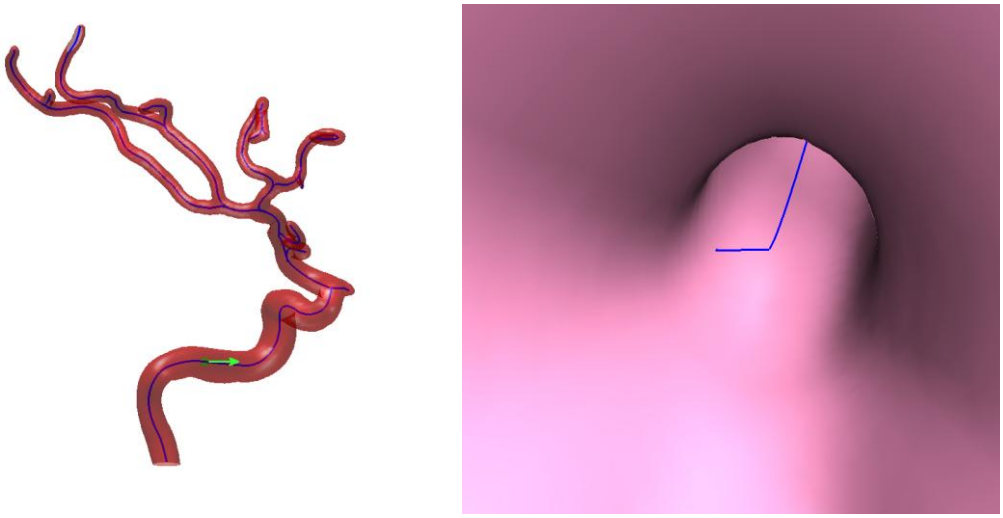
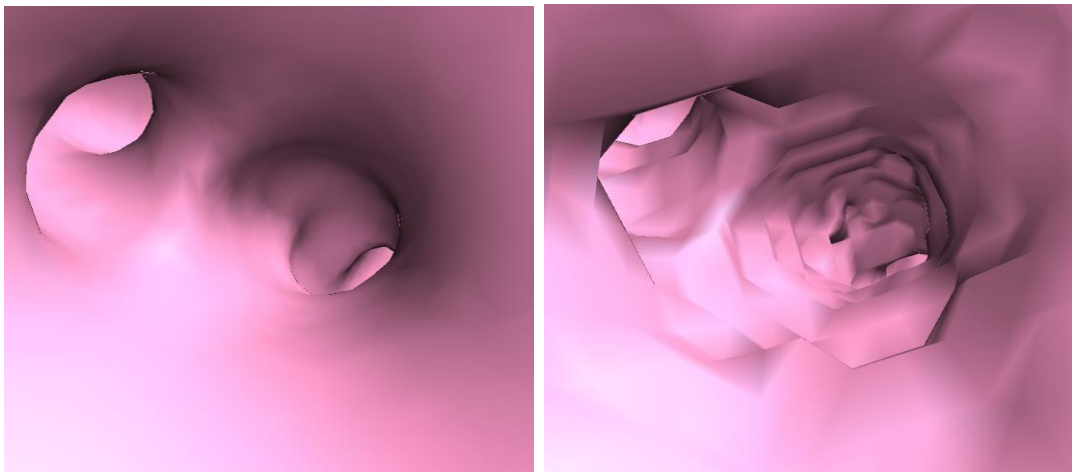


Figure 6.3: *The reconstructed vasculatures with extracted skeleton.*



(a) **(b)**

Figure 6.4 : *Virtual Angisocopy using the pre-extracted skeleton as camera path.* **(a)** *3-D overview of reconstructed vessel tree with skeleton (Green arrow represents the current position and direction of virtual camera).* **(b)** *Perspective view inside the vasculatures (Blue line represents the ongoing camera path).*

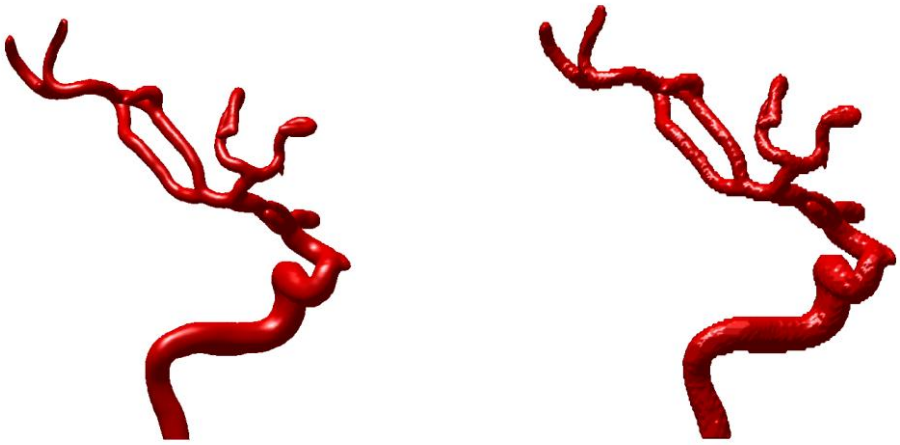


(a) **(b)**

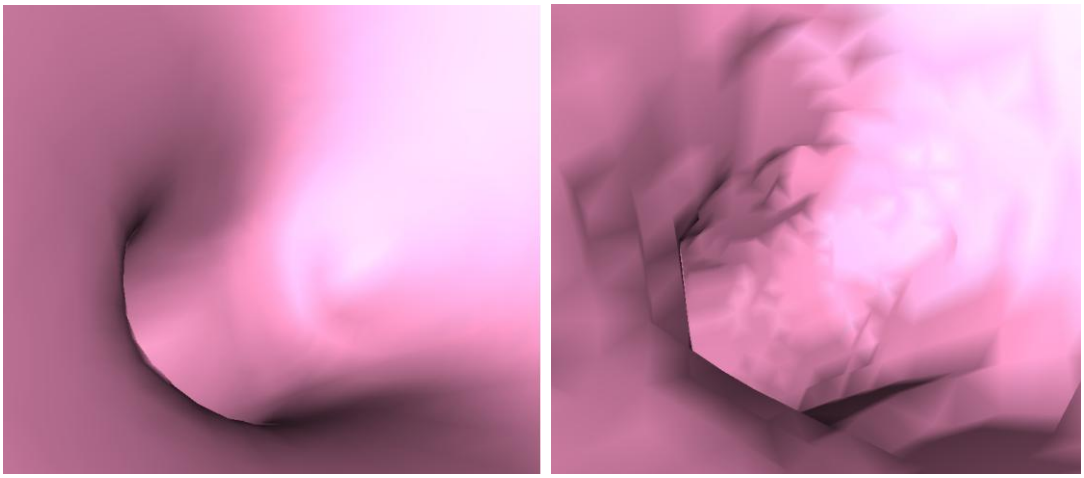
Figure 6.5: *Comparison of rendering quality between our reconstructed result and segmented result.* **(a)** *The perspective view inside the vessel based on our implicit modelling of vasculatures.* **(b)** *The perspective view inside the vessel based on direct application of Marching Cubes to the segmented vasculatures.*

As discussed in Section 6.3, compared to the direct application of surface rendering algorithms (i.e. Marching Cubes) to the segmented vasculatures [see **Figure 6.5(b)]**, based on the our SIS modelling of vasculatures, the surface rendering methods can achieve much higher quality perspective views as well as accurate cross-sections (without model assumptions) [see **Figure 6.5(a)]**, which is suitable for training purposes and diagnosis tasks.

For the free or guided navigation mode of virtual angisocopy, the generation of potential fields is required for achieving efficient collision avoidance. However, the generation of potential fields has the potential risk of changing the morphology of the anatomic structures. For instance, the potential field coding the distance to the vascular surface is generated for the navigation of virtual camera, which has changed the surface morphology of vessel. As can been seen from **Figure 6.6(b),** the vessel surface is not as smooth as that before the generation of the distance field. Accordingly, the changes of vessel surface morphology would greatly influence the visual effect of perspective rendering inside the vasculature [see **Figure 6.6(d)**]. On the other hand, as stated in Section 6.2.2, efficient collision avoidance can be easily achieved based on the constructed implicit function $F(x,y,z)$, without the additional generation of potential fields. In addition, by taking advantage of the 3-D mouse, the position and orientation of the camera can be freely controlled, without the limitations of traditional input devices with limited degrees of freedom. The users can achieve an immersive experience of flexible and effective exploration of the interior of the vascular tree, since they can easily and freely move the camera closer to an object and inspect the region of interest by changing the viewpoint.



(a) **(b)**

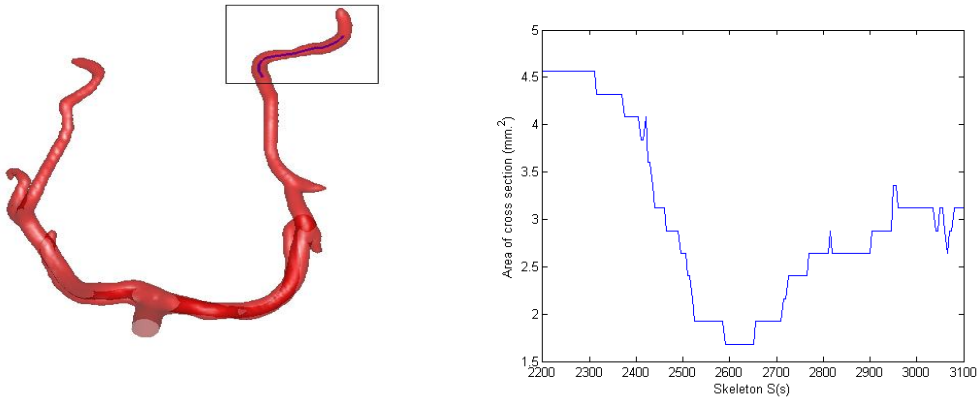


(c) **(d)**

Figure 6.6: *The side effect of generating distance field.* **(a)** *and* **(b)** *are the surface morphology of vessel before and after generating the distance field.* **(c)** *and* **(d)** *are perspective views inside the vasculatures based on (a) and (b)***.**

For the purpose of diagnosing vessel stenosis, the area quantification of cross section is necessary, since visual observation is not adequate for accurate diagnosis. For instance, as shown in **Figure 6.7(a)**, the vessel segment in the rectangle is suspected to suffer stenosis from a general overview of the vasculature. Based on our system, accurate diagnosis can be easily achieved by exploring along the

skeleton of the vessel segment to acquire the area distribution curve of the cross sections [see **Figure 6.7(b)]**. The horizontal axis is the parametric curve **S(s)** representing the skeleton of the vessel; and the vertical axis represents the area of cross section for the corresponding skeleton. For a normal vessel, according to the surgeon's opinion, the areas of the cross sections should in general change gradually along a branch skeleton; while in this case, the areas of the cross sections change sharply, which is an indicator of stenosis.



(a) **(b)**

Figure 6.7: *The diagnosis of vessel stenosis.* **(a)** *The overview of the vessel suspected to suffer stenosis.* **(b)** *Quantitative analysis for the cross sections by calculating the area distribution curve.*

6.5 Summary

Due to the complex nature of vascular structures, it is essential to combine detailed views of the inner structures with an overview of the anatomic structures. Virtual angioscopy technique provides us with an interactive environment for exploring the human vascular system. The smooth and accurate reconstruction of vascular trees is

crucial for virtual angioscopy, as it requires a relatively high visual quality of perspective view inside the dataset for the purposes of training and diagnosis. In this chapter, a virtual angioscopy system based on implicitly reconstructed vasculatures using SIS is presented. Compared to the direct application of surface rendering technique to the segmented datasets, our method can achieve much higher visualization quality as well as accurate cross-sections. In addition, interactive camera navigation can be easily implemented as it is very simple to perform camera-vascular wall collision detection, since the vascular geometry is represented as implicit surfaces.

The use of virtual angioscopy has several benefits. First and foremost, compared to real endoscopy, it is non-invasive or at least minimally invasive, and can provide insights into some vessel parts that might pose difficulty of access to current medical procedures. It can be easily applied to educational purposes, providing unusual insights into the vessels of living patients. In addition, it has the potential benefits for non-invasive evaluation of vascular diseases **(Louisa et al., 2010)**. In our virtual angioscopy system, the free navigation mode allows high-precision manual analysis of the vasculature under various viewing angles and better dynamic localisation of abnormalities. Furthermore, the area quantification of cross section can be easily achieved for the purpose of diagnosing vessel stenosis.

Chapter 7
Conclusions

7.1 Thesis Summary

3-D reconstruction and visualization of vasculatures is one of the most important tasks involved in the development of computer-aided vascular diagnosis and surgery system. It plays a crucial role in many clinical applications, including the diagnosis of stenosis and aneurysm, virtual angosicopy, vascular surgery simulation and planning, and hemodynamics study. Our work focuses on the development of advanced techniques for accurate reconstruction and visualization of vascular structures. This is accomplished with a set of techniques associated with the analysis of 3-D angiography images, including segmentation, modelling, reconstruction, exploration and rendering of vascular structures. The main contributions of this thesis are as follows:

- A localized hybrid level-set model for the segmentation of vasculatures has been proposed. In our model, a local binary fitting energy with a kernel function is introduced into the hybrid level-set framework instead of the preset global threshold for the term referring to the region information. Compared with the global threshold based method, the introduction of local binary fitting energy into the hybrid model enables the extraction of more accurate local image information, which is very essential for enhancing the efficiency and

effectiveness of the segmentation of inhomogeneous images. The proposed method has been successfully applied to over ten 3-D medical datasets for the segmentation of vasculatures. Experimental results show that the localized hybrid model has great advantages on the segmentation of both 2-D and 3-D medical images with intensity inhomogeneity.

- A set of techniques associated with the accurate geometry reconstruction of vasculatures have been accomplished. We firstly present a technique for modelling implicit generalized cylinders, which can construct an accurate and $C^{k-1}(k \geq 2)$ continuous implicit surface in an analytical form. Then, the proposed modelling method has been applied to the accurate geometry reconstruction of vasculatures, which has been accomplished with the methods for extracting the skeleton of the vascular structures and the contour points for specifying accurate cross-sections. The experimental results and validation have shown that our method can achieve accurate, faithful and smooth vascular structures. The continuous geometry of vasculature is represented in the form of implicit functions, which not only allows us to achieve better visualization, but can also be very useful for vessel shape analysis. Briefly, our technique has three main advantages over other techniques for reconstructing vasculatures. First of all, the reconstructed surface is a kind of isosurface and can achieve extremely high smoothness and accuracy to the segmented discrete vascular surface points. Secondly, the modelling result of our method is an implicit volume, and can be directly applied to virtual angioscopy. Thirdly, the modelling objects based on our method are easy to calculate, for there is an analytical representation for the resulting shape.

- GPU acceleration techniques have been investigated and implemented for the rendering of our implicitly represented vasculatures. CUDA is utilized as the accelerating tool to implement data-parallel algorithms on programmable GPUs, which has greatly improved the performance of rendering our implicitly represented vasculatures.

- A virtual angioscopy technique based on our vasculature geometry reconstructed using skeleton-based implicit surfaces is developed. The highly accurate implicit representation of the vasculature not only makes it possible to achieve high visual quality of perspective view inside the vessel structures, but also make the implementation of an interactive virtual angioscopy a much easier task, as the issue of collision detection of virtual camera with vascular objects can be easily solved when the vasculature is represented in implicit form.

There are also some limitations to this study. For the segmentation of 3-D angiography images, our developed method can achieve satisfactory results when compared to the Maximum Intensity Projections (MIPs) of the corresponding original datasets. However, the quantitative validation of segmentation accuracy has not been conducted in this research, which still remains a challenging task, since true surface of the actual vessel is unknown. The proposed skeleton-based modelling technique is very powerful for the modelling of tubular-like objects. However, it would be not appropriate for the modelling of non-tubular-like objects, whose skeleton is difficult to identify. For our GPU accelerating technique, just part of the reconstruction method is implemented on GPU. Although the computational time has been greatly reduced, higher computing performance can be still achieved by implementing all the steps of the reconstruction method on GPU.

7.2 Future Work

As the skeleton of an object has the ability to naturally capture important shape characteristics in three-dimensional contexts, it is intuitively appropriate to represent and model the tubular like objects based on the skeleton. The proposed skeleton-based implicit modelling technique can construct and model the tubular objects as continuous implicit surfaces in analytical forms, with the advantage of convenient implementation of ramifications for branching. In this thesis, we have applied the developed modelling technique to the accurate reconstruction of vascular structures and achieved satisfactory results. It can also be directly applied to the geometry reconstruction of other tubular-like organs, such as the bronchus and colons.

To investigate its clinical applications, collaboration with physicians is essential. This is because computer scientists who master the techniques for reconstructing and visualizing medical images may have little or no knowledge about medicine. To develop CADS to assist physicians for minimally invasive vascular surgery or diagnosis, the involvement o f physicians should be fully considered. In our future work, we will focus on the clinical trial with the collaboration of physicians, and continually improve our techniques to reach a state of maturity. In addition, the integration of our proposed techniques into current computer-aided vascular surgery systems is also a worthwhile endeavour.

Additional attention will be also given to the development of registration techniques for the vasculature structures. This is because, in developing a computer-aided vascular surgery system, the registration between the pre-operative vessel trees and the actual vascular objects is an essential task. In addition, registration is necessary

for many other tasks, including the fusion of the images of the same patient taken with different modalities, such as CT and MRI; combining information of the same patient before, during, and after surgery, such as pre-operative MRI and intra-operative video from a microscope or an endoscope; performing a statistical study of patient data **(Taylor and Joskowicz, 2002).**

Another task that is also worthy of investigation in the future would be to apply the reconstructed vessel geometry to the simulation of Computational Fluid Dynamics (CFD). A highly accurate and smooth vessel geometry representation is crucial for CFD to avoid numerical instabilities and guarantee correct simulation results **(Schumann et al., 2008)**. Simulations of the blood flow are curtail for conducting the quantification of hemodynamic conditions in arteries **(Taylor and Humphrey, 2009)**. Furthermore, in the area of predictive medicine, the computational methods for modelling blood flow could offer an opportunity to design optimal treatments for individuals based on predicted outcomes **(Taylor and Humphrey, 2009)**. CFD simulation would also enable device manufacturers to evaluate endovascular devices in more realistic models of circulation, which could result in several advantages, such as reduced development costs, safer designs, and shorter time-to-market **(Taylor and Steinman, 2010)**.

Furthermore, the interactive modification of patient-specific vascular geometry deserves more attention, as it is the essential link between the 3-D reconstructed vascular model and surgical planning **(Pekkan et al., 2008)**. The interactive shape editing tools enable the users to edit, deform and blend the geometry of the patient-specific vascular models. These edited models can then be exported for patient-specific CFD simulation to analyse the outcomes of different pre-operative

procedures, which is crucial for optimal surgical decision making. In other words, the interactive modification techniques of patient-specific vascular models could enable us to envision large clinical studies based on the limited datasets, which is vital for assessing the validity of predictive patient-specific CFD simulations **(Pekkan et al., 2008)**.

Bibliography

Agin, G. J., 1972. *Representation and Description of Curved Objects.* Memo AIM-173, Stanford Artificial Intelligence Report. Stanford University.

Al-Suyyagh, A., 2009. *GPU Computing-The Ascend of the Coprocessor* [Online]. Available: http://www.abandah.com/gheith/Courses/CPE731_F09/Research_Projects/1_Report.pdf [Accessed 18 May 2011].

Angelidis, A., Jepp, P., and Cani, M.-P., 2002. Implicit modeling with skeleton curves: controlled blending in contact situations. *In:* International Conference on Shape Modeling, 17-22 May 2002, Alberta, Canada. IEEE Computer Society, 137-144.

Antiga, L., 2002. *Patient-specific modelling of geometry and blood flow in large arteries.* Thesis (PhD), Politecnico di Milano University.

Antiga, L., Eneiordache, B., Remuzzi, G., and Remuzzi, A., 2001. Automatic generation of glomerular capillary topological organization. *Microvascular Research*, 62, 346-354.

Arcelli, C. and Sanniti Di Baja, G., 1993. Euclidean skeleton via centre-of-maximal-disc extraction. *Image andVision Computing*, 11(3), 163-173.

Auer, D. P. and Auer, L. M., 1998. Virtual Endoscopy - A New Tool for Teaching and Training in Neuroimaging. *International Journal of Neuroradiology*, 4, 3-14.

Azernikov, S., 2008. Sweeping solids on manifolds. *In:* ACM Symposium on Solid and Physical Modeling, 2-4 June 2008, New York. ACM, 249-255.

Bade, R., Haase, J., and Preim, B., 2006. Comparison of Fundamental Mesh Smoothing Algorithms for Medical Surface Models. *In: Simulation und Visualisierung*, 2-3 March 2006, Magdeburg. SCS Publishing House, 289-304.

Barthe, L., Dodgson, N. A., Sabin, M. A., Wyvill, B., and Gaildrat, V., 2003. Two-imensional potential fields for advanced implicit modeling operators. *Computer Graphics Forum*, 22(1), 23-33.

Bartrolí A. V., 2001. *Visualization Techniques for Virtual Endoscopy.* Thesis (PhD), Vienna University of Technology.

Bartz, D., 2003. *Virtual Endoscopy of the Vascular System (Virtual Angioscopy)* [Online]. Available: http://www.gris.uni-tuebingen.de/people/staff/bartz/proj/endos/angioscopy.html [Accessed 30 July 2009].

Bartz, D., 2005. Virtual Endoscopy in Research and Clinical Practice. *Computer Graphics Forum*, 24(1), 111-126.

Bartz, D., Skalej, M., Welte, D., Straßer, W., and Duffner, F., 1999a. A Virtual Endoscopy System for the Planning of Endoscopic Interventions in the Ventricle System of the Human Brain. *In:* BiOS'99: Biomedical Diagnostics, Guidance and Surgical Assist Systems, 9 July 1999a. SPIE, 91-100.

Bartz, D., Straßer, W., Skalej, M., and Welte, D., 1999b. Interactive Exploration of Extra and Intracranial Blood Vessels. *In:* IEEE Visualization, 24-29 October 1999b, San Francisco. IEEE, 389-392.

Beier, J., Diebold, T., Vehse, H., Biamino, G., Fleck, E., and Felix, R., 1997. Virtual Endoscopy in the Assessment of Implanted Aortic Stents. *In:* Computer Assisted Radiology, June 1997, Berlin. 183-188.

Bloomenthal, J., 1995. *Skeletal design of natural forms.* Thesis (PhD), The University of Calgary.

Bloomenthal, J., Bajaj, C., Blinn, J., Cani-Gascuel, M., Rockwood, A., Wyvill, B., and Wyvill, G., 1997. *Introduction to implicit surfaces,* San Francisco, California, Morgan Kaufmann Publishers, Inc.

Bloomenthal, J. and Shoemake, K., 1991. Convolution Surfaces. *Computer Graphics (Proc. of ACM SIGGRAPH 1991)*, 25, 251-256.

Bloomenthal, J. and Wyvill, B., 1990. Interactive techniques for implicit modeling. *Computer Graphics*, 24(2), 109-116.

Bornik, A., Reitinger, B., and Beichel, R., 2005. Reconstruction and Representation of Tubular Structures using Simplex Meshes. *In:* Winter School of Computer Graphics, 31 January - 4 February 2005, Plzen - Bory. 61-65.

Boskamp, T., Hahn, H., Hindennach, M., Zidowitz, S., Oeltze, S., and Preim, B., 2005. Geometrical and structural analysis of vessel systems in 3D medical image datasets. *Medical Imaging Systems Technology*, 5, 1-60.

Boubekeur, T., Heidrich, W., Granier, X., and Schlick, C., 2006. Volume-Surface Trees. *Computer Graphics Forum*, 25(3), 399-406.

Bourke, P., 1997. *Polygonising a Scalar Field Using Tetrahedrons* [Online]. Available: http://paulbourke.net/geometry/polygonise/ [Accessed 8 August 2011].

Bruckner, S., 2003. *Efficient volume visualization of large medical datasets.* Thesis (Master), Technical University of Vienna.

Buck, I., Foley, T., Horn, D., Sugerman, J., Fatahalian, K., Houston, M., and Hanrahan, P., 2004. Brook for GPUs: streamcomputing on graphics hardware. *ACM Trans. Graph.*, 23(3), 777-786.

Cabral, B., Cam, N., and Foran, J., 1994. Accelerated Volume Rendering and Tomographic Reconstruction Using Texture Mapping Hardware. *In:* ACM Symposium on Volume Visualization, 17-18 October 1994, Washington. ACM, 91-98.

Carrillo, J. F., Orkisz, M., and Hernández Hoyos, M., 2005. Extraction of 3D vascular tree skeletons based on the analysis of connected components evolution. *In:* 11th International Conference on Computer Analysis of Images and Patterns, 5-8 September 2005, Versailles. Springer, 604-611.

Carvalho, B. M., 2003. *Cone-Beam Helical CT Virtual endoscopy: Reconstruction, Segmentation and Automatic Navigation.* Thesis (PhD), University of Pennsylvania.

Caselles, V., Catte, F., Coll, T., and Dibos, F., 1993. A geometric model for active contours in image processing. *Numer.Math.*, 66, 1-31.

Caselles, V., Kimmel, R., and Sapiro, G., 1997. Geodesic active contours. *International Journal of Computer Vision*, 22, 61-79.

Chan, T. and Vese, L., 2001. Active contours without edges. *IEEE Transactions on Image Processing*, 10(2), 266-277.

Chernyaev, E. V., 1995. *Marching Cubes 33: Construction of Topologically Correct Isosurfaces* [Online]. Available: http://wwwinfo.cern.ch/asdoc/psdir/mc.ps.gz. [Accessed 8 June 2011].

Corrigan, A. and Dinh., H. Q., 2005. Computing and Rendering Implicit Surfaces Composed of Radial Basis Functions on the GPU. *In:* International Workshop on Volume Graphics, 2005.

Crespin, B., Blanc, C., and Schlick, C., 1996. Implicit sweep objects. *Computer Graphics Forum*, 15(3), 165-174.

Davis, C. P., Ladds, M. E., Romanowski, B. J., Wildermuth, S., Kopflioch, J. F., and Debatin, J. F., 1996. Human Aorta: Preliminary Results with Virtual Endoscopy Based on Three-dimensional MR Imaging Data Sets. *Radiology*, 199(1), 37-40.

De Cusatis Jr, A., De Figueiredo, L. H., and Gattas, M., 1999. Interval methods for raycasting implicit surfaces with affine arithmetic. *In:* XII Brazilian Symposium on Computer Graphics and Image Processing 17-20 October 1999, Sao Paulo. 65-71.

Dong, C. and Wang, G., 2005. Curvatures estimation on triangular mesh. *J Zhejiang Univ SCI*, 6A(Suppl. I), 128-136.

Drebin, R. A., Carpenter, L., and Hanrahan, P., 1988. Volume Rendering. *Computer Graphics*, 22(4), 65-74.

Elvins, T. T., 1992. A survey of algorithms for volume visualization. *Computer Graphics ACM Siggraph Quarterly*, 26(3), 194-201.

Faugeras, O., 1993. *Three-Dimensional Computer Vision: A Geometric Viewpoint*, Cambridge, MA, MIT Press.

Faux, I. D. and Pratt, M. J., 1979. *Computational Geometry for Design and Manufacture*, New York, Halsted Press.

Felkel, P., Fuhrmann, A., Kanitsar, A., and Wegenkittl, R., 2002. Surface Reconstruction of the Branching Vessels for Augmented Reality Aided Surger. *In:* The Analysis of Biomedical Signals and Images Proceedings (BIOSIGNAL) 2002. 252-254.

Felkel, P., Wegenkitt, L. R., and Buhler, K., 2004. Surface Models of Tube Trees. *In:* Computer Graphics International, 16-19 June 2004, Crete. 70-77.

Ferretti, G. R., Vining, D. J., Knoplioch, J., and Coulomb., M., 1996. Tracheobronchial Tree: Three-Dimensional Spiral CT with Bronchoscopic Perspective. *Journal of Computer Assisted Tomography*, 20(5), 777-781.

Flasque, N., Desvignes, M., Constans, J.-M., and Revenu, M., 2001. Acquisition, segmentation and tracking of the cerebral vascular tree on 3D magnetic resonance angiography images. *Medical Image Analysis*, 5(3), 173-183.

Frangi, A. F., Niessen, W. J., Vincken, K. L., and Viergever, M. A., 1998. Multiscale vessel enhancement filtering. *In:* Medical Image Computing and

Computer-Assisted Intervention (MICCAI), 11-13 October 1998, Cambridge MA. Springer, 130-137.

Fryazinov, O. and Pasko, A., 2007. GPU-based real time FRep ray casting. *In:* GraphiCon'2007, International Conference on Computer Graphics and Vision, 2007, Moscow State University, Moscow. 69-74.

Fuchs, H., Kedem, Z. M., and Uselton, S. P., 1977. Optimal Surface Reconstruction from Planar Contours. *Communications of the ACM*, 20(10), 693–702.

Geiger, B. and Kikinis, R., 1994. Simulation of endoscopy. *In:* AAAI Spring Symposium Series: Applications of Computer Vision in Medical Images Processing, 1994. Stanford University, 138-140.

Gerig, G., Koller, T., Székely, G., Brechbühler, C., and Kübler, O., 1993. Symbolic description of 3d structures applied to cerebral vessel tree obtained from MR angiography volume data. *In:* Information Processing in Medical Imaging, 14-18 June 1993, Arizona. Springer, 94-111.

Gobbetti, E., Pili, P., Zorcolo, A., and Tuveri, M., 1998. Interactive Virtual Angioscopy. *In:* IEEE Visualization, 18-23 October 1998, Research Triangle Park, NC. IEEE Computer Society, 435-438.

Grimm, C., 1999. Implicit Generalized Cylinders using Profile Curves. *In Implicit Surfaces 99*, 33-41.

Gueziec, A. and Hummel, R., 1995. Exploiting Triangulated Surface Extraction using Tetrahedral Decomposition. *IEEE Transactions on Visualisation and Computer Graphics*, 1(4), 328-342.

Hadwiger, M., Sigg, C., Scharsach, H., Bühler, K., and Gross., M., 2005. Real-Time Ray-Casting and Advanced Shading of Discrete Isosurfaces. *Computer Graphics Forum*, 24(3), 303-312.

Hahn, H. K., Preim, B., Selle, D., and Peitgen, H., 2001. Visualization and Interaction Techniques for the Exploration of Vascular Structures. *In:* IEEE Visualization, 21-21 October 2001, San Diego. IEEE Computer Society, 395-402.

Hart, J. C., 1993. Ray Tracing Implicit Surfaces. *Siggraph 93 Course Notes*, (25), 1-15.

Herman, G. T. and Liu, H. K., 1979. Three-dimensional display of Human Organs from Computed Tomograms. *Computer Graphics and Image Processing*, 9(1), 1-21.

Hohne, K., Pflesser, B., and Pommert, A., 2000. A Realistic Model of the Inner Organs from the Visible Human Data. *In:* Medical Image Computing and Computer-Assisted Intervention, 11-14 October 2000, Pittsburgh. Springer, 776-785.

Hong, L., Muraki, S., Kaufman, A., Bartz, D., and He, T., 1997. Virtual voyage: Interactive navigation in the human colon. *In:* ACM SIGGRAPH Conference Proceedings, 3-8 August 1997, Los Angeles. 27-34.

Hornus, S., Angelidis, A., and Cani, M.-P., 2003. Implicit Modelling Using Subdivision curves. *Visual Comput.*, 19(2-3), 94-104.

Johansson, G. and Carr, H., 2006. Accelerating marching cubes with graphics hardware. *In:* Proceedings of the 2006 conference of the Center for Advanced Studies on Collaborative research, 2006. 39.

Joshi, A., Qian, X., Dione, D. P., Bulsara, K. R., Breuer, C. K., Sinusas, A. J., and Papademetris, X., 2008. Effective visualization of complex vascular structures using a non-parametric vessel detection method. *IEEE Transactions on Visualization and Computer Graphics*, 14(6), 1603-1610.

Kanai, T., Ohtake, Y., Kawata, H., and Kase, K., 2006. Gpu-based rendering of sparse low-degree implicit surfaces. *In:* the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast Asia, 29 November - 2 December 2006, Kuala Lumpur. ACM, 165-171.

Kass, M., Witkin, A., and Terzopoulos, D., 1988. Snakes: active contour models. *International Journal of Computer Vision*, 1, 321-331.

Kaufman, A., 1991. *Volume Visualization*, Los Alamitos, IEEE Computer Society.

Kaufman, A., 1997. *Volume Visualization: Principles and Advances.* SIGGRAPH Course Notes.

Keppel, E., 1975. Approximating Complex Surfaces by Triangulation of Contour Lines. *IBM Journal of Research and Development*, 19(1), 2-11.

Kipfer, P. and Westermann, R., 2005. GPU construction and transparent rendering of iso-surfaces. *In:* Proceedings of Vision, Modeling and Visualization, 16-18 November 2005, Erlangen. 241-248.

Kirbas, C. and Quek, F., 2004. A Review of Vessel Extraction Techniques and Algorithms. *ACM Computing Surveys*, 36(2), 81-121.

Kirk, D. and Hwu, W.-M., 2010. *Programming Massively Parallel Processors: A Hands-on Approach*, Elsevier.

König, A. H. and Gröller, E., 2001. *3D Medical Visualization: Breaking the Limits of Diagnostics and Treatment* [Online]. Available: http://www.ercim.org/publication/Ercim_News/enw44/koenig.html [Accessed 24 July 2009].

Lacroute, P. and Levoy, M., 1994. Fast Volume Rendering Using a Shear-Warp Factorization of the Viewing Transformation. *In:* ACM SIGGRAPH, 24 - 29 July 1994, Orlando. ACM, 451-458.

Lefohn, A. E., Sengupta, S., Kniss, J., Strzodka, R., and Owens, J. D., 2006. Glift: Generic, efficient, random-access GPU data structures. *ACM Trans. Graph.*, 25(1), 60-99.

Lesage, D., Angelini, E. D., Bloch, I., and Funka-Lea, G., 2009. A review of 3D vessel lumen segmentation techniques: Models, features and extraction schemes. *Medical Image Analysis*, 13, 819-845.

Lessig, C., 2004. *Interaktives Ray-Tracing und Ray-Casting auf programmierbaren Grafikkarten.* Thesis (Bachelor), Bauhaus University Weimar.

Li, C., Kao, C., Gore, J., and Ding, Z., 2007. Implicit active contours driven by local binary fitting energy. *In:* IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), 17-22 June 2007, Minneapolis. 1-7.

Li, C., Xu, C., Gui, C., and Fox, M. D., 2005. Level Set Evolution Without Re-initialization: A New Variational Formulation. *In:* IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), 20-26 June 2005, San Diego. 430-436.

Li, Q., 2007. Smooth Piecewise Polynomial Blending Operations for implicit shapes. *Computer Graphics forum*, 26(2), 157-171.

Li, Q., Griffiths, J. G., and Ward, J., 2006. Constructive implicit fitting. *Comput. Aided Geom. Des.*, 23(1), 17-44.

Li, Q. and Tian, J., 2009. 2D piecewise algebraic splines for implicit modeling. *ACM Transactions on Graphics*, 28(2), 1-13.

Li, Q. and Tian, J., 2011. Partial Shape-Preserving Splines. *Comput. Aided Des.*, 43, 394-409.

Liktor, G., 2008. Ray tracing implicit surfaces on the GPU. *Computer Graphics and Geometry*, 10(3), 36-53.

Lin, M. and Gottschalk, S., 1998. Collision detection between geometric models: A survey. *In:* IMA Conference on Mathematics of Surfaces, August 1998. 37-56.

Lorensen, W., Jolesz, F., and Kikinis, R., 1995. The exploration of cross-sectional data with a virtual endoscope. *In:* MORGA, R. S. A. K. (ed.) *Interactive Technology and New Medical Paradigms for Health Care.*

Lorensen, W. E. and Cline, H. E., 1987. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. *In:* ACM SIGGRAPH, July 1987, Anaheim. ACM, 163-169.

Louisa, N., Bruguiereb, E., Kobeiterb, H., Desgrangesa, P., Allairea, E., Kirschc, M., and Becquemina, J. P., 2010. Virtual Angioscopy and 3-Dimensional Navigation Findings of the Aortic Arch After Vascular Surgery. *European Journal of Vascular and Endovascular Surgery*, 40(3), 340-347.

Ma, Z., Tavares, J., and Jorge, R., 2009. *A review on the current segmentation algorithms for medical images* [Online]. Available: http://paginas.fe.up.pt/~tavares/downloads/publications/artigos/IMAGAPP_2009_ZM_JT_RJ.pdf [Accessed 24 September 2009].

Malladi, R., Sethian, J. A., and Vemuri, B. C., 1995. Shape modeling with front propagation: a level set approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2), 158-175.

Masutani, Y., Masamune, K., and Dohi, T., 1996. Region-Growing-Based Feature Extraction Algorithm for Tree-Like Objects. *In:* Visualization in Biomedical Computing, 22-25 September 1996, Hamburg. Springer, 161-171.

Mccormick, P., Inman, J., Ahrens, J., Mohd-Yusof, J., Roth, G., and Cummins, S., 2007. Scout: A Data-Parallel Programming Language for Graphics Processors. *Parallel Comput.* , 33(10-11), 648-662.

Nain, D., Haker, S., Kikinis, R., and Grimson, W., 2001. An interactive virtual endoscopy tool. *In:* Workshop on Interactive Medical Image Visualization and Analysis, 14-17 October 2001, Utrecht. 55-60.

Nakajima, N., Wada, J., Miki, T., Haraoka, J., and Hata, N., 2007. Surface Rendering-based Virtual Intraventricular Endoscopy: Retrospective Feasibility Study and Comparison to Volume Rendering-based Approach. *NeuroImage* 37(Suppl 1), 89-99.

Nvidia, 2006. *Technical Brief: NVIDIA GeForce 8800 GPU Architecture Overview*. NVIDIA Corporation.

Nvidia, 2007. *CUDA Programming Guide.* NVIDIA Corporation.

Oeltze, S. and Preim, B., 2005. Visualization of Vascular Structures with convolution surfaces: Method, Validation and Evaluation. *IEEE Transactions on Medical Imaging*, 25(4), 540-548.

Oentoro, A., 2009. *A system for computer-assisted surgery with intraoperative CT imaging.* Thesis (Master), Queen's University

Ohtake, Y., Belyaev, A., Alexa, M., Turk, G., and Seidel, H., 2003. Multilevel Partition of Unity Implicits. *ACM Transactions on Graphics*, 22, 463-470.

Orkisz, M. and Hernández-Hoyos, M., 2001. Models for 3D vascular image analysis. *J Med Informatics Technol*, 2(1), 13-22.

Orkisz, M. M., Bresson, C., Magnin, I. E., Champin, O., and Douek, P. C., 1997. Improved vessel visualization in MR angiography by nonlinear anisotropic filtering. *Magnetic Resonance in Medicine*, 37, 914-919.

Ortner, T., 2008. *GPGPU Raytracing in Real-Time.* Master Thesis (Master), University of Hull.

Osher, S. and Fedkiw, R., 2002. *Level Set Methods and Dynamic Implicit Surfaces,* New York, Springer.

Osher, S. and Sethian, J. A., 1988. Fronts propagating with curvature dependent speed: algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 79, 12-49.

Owens, J. D., Houston, M., Luebke, D., Green, S., Stone, J. E., and Phillips, J. C., 2008. GPU computing. *Proceedings of the IEEE*, 96(5), 879-899.

Owens, J. D., Luebke, D., Govindaraju, N., Harris, M., Krüger, J., Lefohn, A. E., and Purcell, T., 2007. A Survey of General-Purpose Computation on Graphics Hardware. *Computer Graphics Forum*, 26(1), 80-113.

Pasko, G. I., Pasko, A. A., and Kunii, T. L., 2005. Bounded blending for function-based shape modeling. *IEEE Computer Graphics and Applications*, 25(2), 36-45.

Pawasauskas, J., 1997. *Volume Visualization with Ray Casting* [Online]. Available: http://web.cs.wpi.edu/~matt/courses/cs563/talks/powwie/p1/ray-cast.htm [Accessed 3 August 2009].

Pekkan, K., Whited, B., and Kanter, K., 2008. Patient-specific surgical planning and hemodynamic computational fluid dynamics optimization through free-form haptic anatomy editing tool (SURGEM). *Med Biol Eng Comput*, 46, 1139-52.

Peters, M. T., 2000. Image-guided surgery: From X-rays to virtual reality. *Computer Methods in Biomechanics and Biomedical Engineering*, 4(1), 27-57.

Pommert, A., Bomans, M., and Höhne, K. H., 1992. Volume visualization in magnetic resonance angiography. *IEEE Computer Graphics and Application*, 12(5), 12-13.

Preim, B. and Oeltze, S., 2007. 3D Visualization of Vasculature: An Overview. *Visualization in Medicine and Life Science*, 39-59.

Qian, X., Brennan, M., Dione, D., Dobrucki, L., Jackowski, M., Breuer, C., Sinusas, A., and Papademetris, X., 2008. A non-parametric vessel detection method for complex vascular structures. *Medical Image Analysis*, 13(1), 49-61.

Reuter, P., 2003. *Reconstruction and Rendering of Implicit Surfaces from Large Unorganized Point Sets.* Thesis (PhD), LABRI Universite Bordeaux.

Robb, R. A., 1996. *Virtual (Computed) endoscopy: Development and evaluation using the visible human datasets* [Online]. Available: http://www.nlm.nih.gov/research/visible/vhp_conf/robb/robb_pap.htm#1.com [Accessed 28 July 2009].

Rubin, G. D., 1996. Perspective volume rendering of CT and MR images: Applications for endoscopic imaging. *Radiology*, 199, 321-330.

Sarah, F. and Frisken, G., 1998. Constrained elastic surface nets: Generating smooth surfaces from binary segmented data. *In:* International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI), 11-13 October 1998, Cambridge MA. Springer, 888-898.

Sauret, V., Goatman, K. A., Fleming, J. S., and Bailey, A. G., 1999. 3D topology and morphology of branching networks using computed tomography (CT) - application to the airway tree. *In:* Medical Image Understanding and Analysis, July 1999, Oxford.

Schmidt, R. and Wyvill, B., 2005. Generalized sweep templates for implicit modeling. *In:* The 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia, 29 November - 2 December 2005, Dunedin. ACM, 187-196.

Schumann, C., Neugebauer, M., Bade, R., Peitgen, H.-O., and Preim, B., 2008. Implicit vessel surface reconstruction for visualization and CFD simulation. *International Journal of Computer Assisted Radiology and Surgery*, 2(5), 275-286.

Schumann, C., Oeltze, S., Bade, R., Preim, B., and Peitgen, H.-O., 2007. Model-free surface visualization of vascular trees. *In:* IEEE/Eurographics Symposium on Visualization,, 23-25 May 2007, Linköping. 283-290.

Seong, J.-K., Elber, G., and Kim, M.-S., 2006. Trimming local and global self-intersections in offset curves/surfaces using distance maps. *Computer-Aided Design*, 38(3), 183-193.

Serlie, I., Vos, F., Gelder, R. V., Stoker, J., Truyen, R., Gerritsen, F., Nio, Y., and Post, F., 2001. Improved visualization in virtual colonoscopy using image-based rendering. *In:* Data Visualization (Proceedings of Symposium on Visualization), 28-30 May 2001, Ascona. 137-146.

Sethian, J. A., 1999. *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*, Cambridge University.

Shirley, P. and Tuckman, A., 1991. A Polygonal Approximation to Direct Scalar Volume Rendering. *Computer Graphics*, 24(5), 63-70.

Singh, A., Goldgof, D., and Terzopoulos, D., 1998. *Deformable Models in Medical Image Analysis*, IEEE Computer Society.

Stone, S. S., Haldar, J. P., Tsao, S. C., Hwu, W.-M. W., Sutton, B. P., and Liang, Z.-P., 2008. Accelerating advanced MRI reconstructions on GPUs. *Journal of Parallel and Distributed Computing*, 68(10), 1307-1318.

Straka, M., 2006. *Processing and Visualization of Peripheral CT-Angiography Datasets.* Thesis (PhD), Vienna University of Technology.

Strengert, M., Müller, C., Dachsbacher, C., and Ertl, T., 2008. CUDASA: Compute Unified Device and Systems Architecture. *In:* Eurographics 2008 Symposium on Parallel Graphics and Visualization, 14-15 April 2008, Crete. 49-56.

Strzodka, R. and Garbe, C., 2004. Real-time motion estimation and visualization on graphics cards. *In:* IEEE Visualization, 10-15 October 2004, Austin. IEEE Computer Society, 545-552.

Tatarchuk, N., Shopf, J., and Decoro, C., 2008. Advanced interactive medical visualization on the GPU. *Journal of Parallel and Distributed Computing*, 68(10), 1319-1328.

Taubin, G., 1995. A signal processing approach to fair surface design. *In:* ACM SIGGRAPH, 6-11 August 1995, Los Angeles. 351-358.

Taylor, C. A. and Humphrey, J. D., 2009. Open problems in computational vascular biomechanics: hemodynamics and arterial wall mechanics. *Computer Methods in Applied Mechanics and Engineering*, 198, 3514-3523.

Taylor, C. A. and Steinman, D. A., 2010. Image-Based Modeling of Blood Flow and Vessel Wall Dynamics: Applications, Methods and Future Directions. *Annals of Biomedical Engineering*, 38(3), 1188-1203.

Taylor, R. H. and Joskowicz, L., 2002. Computer-integrated surgery and medical robotics. *In:* KUTZ, M. (ed.) *Standard Handbook of Biomedical Engineering and Design.* McGraw-Hill.

Tizon, X., 2004 *Algorithms for the analysis of 3D Magnetic Resonance Angiography Images.* Thesis (PhD), Acta Universitatis Agriculturae Sueciae.

Toledo, R. and Levy, B., 2004. Extending the graphic pipeline with new GPU-accelerated primitives. *In:* Tech report, INRIA, 2004.

Tuy, H. K. and Tuy, L. T., 1984. Direct 2-D Display of 3-D Objects. *IEEE Computer Graphics & Applications*, 4(10), 29-33.

Vega-Higuera, F., Hastreiter, P., Fahlbusch, R., and Greiner, G., 2005. High performance volume splatting for visualization of neurovascular data. *In:* IEEE Visualization, 23-28 October 2005, Minneapolis. IEEE Computer Society, 271 - 278.

Vega-Higuera, F., Sauber, N., Tomandl, B., Nimsky, C., Greiner, G., and Hastreiter, P., 2003. Enhanced 3D-Visualization of Intracranial Aneurysms Involving the Skull Base. *In:* Medical Image Computing and Computer-Assisted Intervention (MICCI), 15-18 November 2003, Montréal. Springer, 256-263.

Vilanova, A., König, A., and Gröller, E., 1999. VirEn: A Virtual Endoscopy System. *Journal Machine Graphics & Vision*, 8(3), 469-487.

Vining, D., Stelts, D., Ahn, D., Hemler, P., Ge, Y., Hunt, G., Siege, C., Mccorquodale, D., Sarojak, M., and G.Ferretti, 1997. FreeFlight: A virtual endoscopy system. *In:* First Joint Conference, Computer Vision, Virtual

Reality and Robotics in Medicine and Medical Robotics and Computer-Assisted Surgery, 19-22 March 1997, Grenoble. Springer, 413-416.

Vollmer, J., Mencel, R., and Mueller, H., 1999. Improved laplacian smoothing of noisy surface meshes. *Computer Graphics Forum*, 18(3), 131-138.

Wang, L., Li, C., Suna, Q., Xiaa, D., and Kao, C. Y., 2009. Active contours driven by local and global intensity fitting energy with application to brain MR image segmentation. *Computerized Medical Imaging and Graphics*, 33(7), 520-531.

Westover, L., 1990. Footprint Evaluation for Volume Rendering. *Computer Graphics*, 24(4), 367-376.

Whitaker, R., Breen, D., Museth, K., and Soni, N., 2001. Segmentation of biological datasets using a level-set framework. *In:* Volume Graphics, 21-22 June 2001, New York. 249-263.

Who, 2011. *Cardiovascular diseases (CVDs)* [Online]. Available: http://www.who.int/mediacentre/factsheets/fs317/en/index.html [Accessed 18 August 2011].

Wickham, J. E. A., 1994. Minimally invasive surgery: Future developments. *BMJ*, 308, 193-196.

Wikipedia, 2010. *Frenet-Serret formulas* [Online]. Available: http://en.wikipedia.org/wiki/Frenet%E2%80%93Serret_formulas [Accessed June 6 2010].

Wikipedia, 2011. *Computer-assisted surgery* [Online]. Available: http://en.wikipedia.org/wiki/Computer-assisted_surgeryWikipedia [Accessed Octorber 18 2011].

Wink, O., 2004. *Vessel axis determination for diagnosis and treatment,* Amsterdam, The Netherlands, Utrecht University.

Wu, X., Luboz, V., Krissian, K., Cotin, S., and Dawson, S., 2010. Segmentation and reconstruction of vascular structures for 3D real-time simulation. *Medical Image Analysis*, 15(1), 22-34.

Xu, F. and Mueller, K., 2005. Accelerating popular tomographic reconstruction algorithms on commodity PC graphics hardware. *IEEE Trans. on Nuclear Science*, 52(3), 654-663.

Yagel, R. and Kaufman, A., 1992. Template-based volume viewing. *Computer Graphics Forum*, 11(3), 153-167.

Yan, P. and Kassim, A. A., 2006. Segmentation of volumetric MRA images by using capillary active contour. *Med. Image Anal*, 10 (3), 317-329.

Yang, R. and Pollefeys, M., 2005. A versatile stereo implementation on commodity graphics hardware. *Real-Time Imaging*, 11(1), 7-18.

Zhang, Y., Matuszewski, B. J., Shark, L. K., and Moore, C. J., 2008. Medical Image Segmentation Using New Hybrid Level-Set Method. *In:* The Fifth International Conference BioMedical Visualization: Information Visualization in Medical and Biomedical Informatics, July 2008, London. 71-76.

Zhukov, L., Museth, K., Breen, D., Whitaker, R., and Barr, A., 2003. Level set modeling and segmentation of DT-MRI brain data. *Journal of Electronic Imaging*, 12(1), 125-133.