Curve Evolution for Medical Image Segmentation

by

Liana M. Lorigo

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2000

Author ....
Department of Electrical Engineering and Computer Science
May 4, 2000

Certified by ....
Olivier D. Faugeras
Adjunct Professor of Computer Science and Engineering
Thesis Supervisor

Certified by ....
W. Eric L. Grimson
Bernard M. Gordon Professor of Medical Engineering
Thesis Supervisor

Accepted by ....
Arthur C. Smith
Chairman, Departmental Committee on Graduate Students

Curve Evolution for Medical Image Segmentation

by

Liana M. Lorigo

Submitted to the Department of Electrical Engineering and Computer Science on May 4, 2000, in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science and Engineering

Abstract

The model of geodesic curves in three dimensions is a powerful tool for image segmentation and also has potential for general classification tasks. We extend recent proofs on curve evolution and level set methods to a complete algorithm for the segmentation of tubular structures in volumetric images, and we apply this algorithm primarily to the segmentation of blood vessels in magnetic resonance angiography (MRA) images. This application has clear clinical benefits as automatic and semi-automatic segmentation techniques can save radiologists large amounts of time required for manual segmentation and can facilitate further data analysis.

It was chosen both for these benefits and because the vessels provide a wonderful example of complicated 3D curves. These reasons reflect the two primary contributions of this research: it addresses a challenging application that has large potential benefit to the medical community, while also providing a natural extension of previous geometric active contour models research.

In this dissertation, we discuss this extension and the MRA segmentation system, CURVES, that we have developed. We have run CURVES on over 30 medical datasets and have compared our cerebral segmentations to segmentations obtained manually by a neurosurgeon for approximately 10 of these datasets. In most cases, we are able to obtain a more detailed representation of the thin vessels, which are most difficult to obtain. We also discuss a novel procedure for extracting the centerlines of tubular structures and proof-of-concept experiments applying surface evolution ideas to general feature-based classification problems.

This work is a collaboration with Brigham and Women's Hospital.

Thesis Supervisor: Olivier D. Faugeras
Title: Adjunct Professor of Computer Science and Engineering

Thesis Supervisor: W. Eric L. Grimson
Title: Bernard M. Gordon Professor of Medical Engineering

Acknowledgments

I am fortunate to have had the opportunity to work with my advisors Olivier Faugeras and Eric Grimson. Olivier's excitement for the mathematical foundations of image analysis and for this project have made this project a truly enjoyable exploration which has challenged and strengthened both my mathematical skills and my understanding of the field. He introduced me to this project, and his expertise and insight have taught me about computer vision and scientific research in general. I thank him for these lessons and for his confidence in me and his constant support.

Eric's expertise, guidance, and assistance have also been invaluable over the course of this project and of my entire stay at MIT. His leadership of a variety of projects in our group has taught me about diverse computer vision methods, and his strong collaboration with Brigham & Womens Hospital has enabled the unparalleled benefit I have enjoyed from working closely with clinicians. I am grateful for these lessons, his advising, and his constant support.

I thank Ron Kikinis for teaching me about the clinical utility of this area of work and for helping to focus my attention on the aspects that are most important practically, for providing datasets and for providing this fabulous inter-disciplinary experience where I could benefit from close collaboration with medical specialists. I thank Tomas Lozano-Pérez for encouraging me to focus on the big picture of my research and for broadening the scope of my thinking about this project. I thank Carl-Fredrik Westin for assistance on so many aspects of the project, from detailed algorithm discussions to validation suggestions to data acquisition.

I thank Arya Nabavi for providing the manual segmentations used for validation in this study and for many discussions which have helped me to better understand the vasculature in the brain and to share his enthusiasm for this beautiful structure. I thank Renaud Keriven for efficient level set prototype code. I thank Dan Kacher for acquiring the multiresolution medical datasets, Yoshinobu Sato for discussing validation approaches, and Mike Leventon for providing help and distance function code. I thank Sandy Wells for discussions on this and related projects, Chris Stauffer for tracking data, and Siemens for medical datasets.

While acknowledging my advisors, readers, and collaborators, I claim any errors as my own.

Finally, I thank my colleagues at the AI Lab, INRIA, the SPL, and elsewhere who have contributed to this research through discussion and friendship. I especially thank my family – Charles, Shirley, Susan, and Lori Lorigo – for loving and supporting me in all endeavors.

This document describes research performed at the Artificial Intelligence Laboratory at the Massachusetts Institute of Technology. Support was provided by NSF Contract IIS-9610249, by NSF ERC (Johns Hopkins University agreement) 8810-274, by Navy/ONR (MURI) N00014-95-1-0600, and by NSF Contract DMS-9872228. This work was a collaboration with the Surgical Planning Laboratory at Brigham and Women's Hospital and unless otherwise noted all medical datasets are courtesy of our collaborators there. Funding also provided by NIH 1R01RR11747 and NIH 1P41RR13218. Some research was also performed at INRIA in Sophia Antipolis, France.

To my parents.

Contents

Chapter 1

Introduction

A fundamental task in computer vision is the segmentation of objects, which is the labeling of each image region as part of the object or as background. This task can also be defined as finding the boundary between the object and the background.

While no computerized methods have approached the capabilities of the human vision system, the body of automatic and semi-automatic methods known as *active contour models* has been particularly effective for the segmentation of objects of irregular shape and somewhat homogeneous intensity. After the initial presentation of the active contour approach by Kass, Witkin, and Terzopoulos in 1987, extensions abounded. Such extensions included both functional improvements which yielded increased capabilities and robustness improvements which improved the success rate of the existing approach.

Our studies began with the in-depth exploration of one group of such extensions which apply to 3D objects in 3D images, such as anatomical structures in clinical magnetic resonance imagery (MRI), and use 3D partial differential equations as an implementation mechanism. This group exhibits a number of advantages which will be explained in this dissertation.

One difficulty with these approaches and all previous related approaches, however, is their limitations in terms of presumed local object shape. In particular, it was presumed that all 3D objects were smooth, "regular" objects. This limitation is problematic for 3D objects whose shape better resembles lines or tubes than smooth solid shapes, for example. An example object is the network of blood vessels in the human body, as imaged with MR techniques. Note that we are not discussing a global notion of object shape, but rather a local notion which looks at small areas of the object.

To address this problem, we looked to theoretical research in the mathematics community which deals with a very general notion of local shapes in arbitrary dimensions. This generality is large as the number of local "shapes" in arbitrary dimensions, e.g. much greater than three, grows with the dimension of the space. We have provided an interpretation of these ideas for application to computer vision problems and also to general data representation for tasks such as classification and recognition, which are not necessarily vision-based.

To further explore these ideas, we developed a complete algorithm for the segmentation of blood vessels in magnetic resonance angiography (MRA) data using an active contours approach presuming the correct line-like local object shape. The "lines" are modeled as geodesic (local minimal length) curves; this notion will be explained in subsequent chapters. The program we developed based on our algorithm is called CURVES, and we have run it on over 30 medical datasets. We have compared its segmentations with those obtained manually by a neurosurgeon for approximately 10 of these datasets. In most cases, CURVES was able to obtain a more detailed representation of the thinnest vessels, which are the most difficult for both automatic methods and for a human expert. Because CURVES treats the tubular-shaped vessels as the 3D curves that comprise their centerlines and also because of clinical interest, we have also considered these centerlines directly and have developed an algorithm for finding them explicitly.

In short, the thesis of this research is that the model of geodesic curves in three dimensions is a powerful tool for image segmentation, and also has potential for general classification tasks. This document is an elaboration on this thesis with appropriate theoretical and empirical support. The application of MRA segmentation was chosen both for its practical importance and because the vessels provide a wonderful example of complicated 3D curves. These dual reasons reflect, respectively, the two primary contributions of this research: It addresses a challenging application that has large potential benefit to the medical community, while also providing a natural extension of previous geometric active contour models research.

1.1 Algorithm perspective

Curvature-based evolution schemes for segmentation, implemented with level set methods, have become an important approach in computer vision [20, 56, 96]. This approach uses partial differential equations to control the evolution. An overview of the superset of techniques using related partial differential equations can be found in [19]. The fundamental concepts from mathematics from which mean curvature schemes derive were explored several years earlier when smooth closed curves in 2D were proven to shrink to a point under mean curvature motion [44, 46]. Evans and Spruck and Chen, Giga, and Goto independently framed mean curvature flow of any hypersurface as a level set problem and proved existence, uniqueness, and stability of viscosity solutions [38, 22]. For application to image segmentation, a vector field was induced on the embedding space, so that the evolution could be controlled by an image gradient field or other image data. The same results of existence, uniqueness, and stability of viscosity solutions were obtained for the modified evolution equations for the case of planar curves, and experiments on real-world images demonstrated the effectiveness of the approach [17, 20].

Curves evolving in the plane became surfaces evolving in space, called *minimal surfaces* [20]. Although the theorem on planar curves shrinking to a point could not be extended to the case of surfaces evolving in 3D, the existence, uniqueness, and stability results of the level set formalism held analogously to the 2D case. Thus the method was feasible for evolving both curves in 2D and surfaces in 3D. Beyond elegant mathematics, impressive results on real-world data sets established the method as an important segmentation tool in both domains. One fundamental limitation to these schemes has been that they describe only the flow of hypersurfaces, i.e., surfaces of codimension one, where the *codimension* of a manifold is the difference between the dimensionality of the embedding space and that of the manifold.

Altschuler and Grayson studied the problem of curve-shortening flow for 3D curves [3], and Ambrosio and Soner generalized the level set technique to arbitrary manifolds in arbitrary dimension, that is, to manifolds of any codimension. They provided the analogous results and extended their level set evolution equation to account for an additional vector field induced on the space [4]. Subsequent work developed and analyzed a diffusion-generated motion scheme for codimension-two curves [95]. We herein present the first 3D geodesic active contours algorithm in which the model is a line instead of a surface, based on Ambrosio and Soner's work. Our CURVES system uses these techniques for automatic segmentation of blood vessels in MRA images.

1.2 Application perspective

1.2.1 Medical

The high-level practical goal of this research is to develop computer vision techniques for the segmentation of medical images. Automatic and semi-automatic vision techniques can potentially assist clinicians in this task, saving them much of the time required to manually segment large data sets. For this research, we consider the segmentation of blood vessels in volumetric images.

The vasculature is of utmost importance in neurosurgery and neurological study. Elaborate studies with a considerable x-ray exposure, such as multi-planar conventional angiography or spiral computed tomography (CT) with thin slices, have to be carried through to achieve an accurate assessment of the vasculature. But due to their two-dimensional character, the spatial information is lost in x-ray studies. Three-dimensional CT angiography and three-dimensional time-of flight magnetic resonance angiography (TOF-MRA) yield spatial information, but lack more subtle information. Furthermore, the three-dimensional CT needs a significant amount of contrast administration. All these studies cannot provide a spatial representation of small vessels. These vessels whose topology exhibits much variability are most important in planning and carrying out neurosurgical procedures. In planning, they provide information on where the lesion draws its blood supply and where it drains, which is of special interest in the case of vascular malformations. The surgical interest is to distinguish between the feeding vessel and the transgressing vessel which needs to be preserved. In interventional neuroradiology this information is used to selectively close the feeding vessel through the artery itself. During surgery the vessels serve as landmarks and guidelines toward the lesion. The more minute the information is, the more precise the navigation and localization of the procedures. Current representations do not yield such detailed knowledge. A more precise spatial representation is needed.

Working toward this goal, we developed the CURVES vessel segmentation system with a focus on extracting the smallest vessels. CURVES models the vessels as three-dimensional curves with arbitrary branching and uses an active contours approach to segment these curves from the medical image [75]. That is, it evolves an initial curve into the curves in the data (the vessels). It is not limited to blood vessels, but is applicable to a variety of curvilinear structures in 3D.

1.2.2 Classification

In addition to the segmentation of tubular objects in 3D images, this curve evolution algorithm can be applied to general classification and recognition problems in the field of artificial learning. In the most general case, we assume that instances of objects are specified by n parameters; that is, each instance is a point in an n-dimensional feature space.

The motivating assumption is that objects can be represented by continuous d-dimensional manifolds in that feature space, for $d < n$. The manifolds would be initialized to some given manifold, then would evolve, within the n-D space, based on positive and negative training examples until it converged to an appropriate representation of the object. The case analogous to the evolution of curves in 3D would be instances of an object in a 3D feature space which are well-modeled by 1D manifolds. This dissertation focuses on only the medical applications, but a classification experiment is also performed for proof-of-concept.

1.3 Contributions

Specifically, this dissertation makes the following contributions.

1. We have extended the geodesic active contour model, increasingly common in computer vision, to handle tubular structures correctly.

2. We have specialized the segmentation algorithm to blood vessels in volumetric MRA data, with a focus on extracting the very thin vessels, which are the most difficult.

3. We have experimented with our algorithm on over 30 medical datasets, in addition to synthetic volumes.

4. We have compared our cerebral segmentations to segmentations obtained manually by a neurosurgeon for approximately 10 of these datasets. In most cases. we are able to obtain a more detailed representation of the thin vessels.

5. To accompany our segmentation tool, we have developed a novel procedure for extracting the centerlines of tubular structures.

6. Finally, we have performed proof-of-concept experiments applying surface evolution ideas to the general feature-based classification problem.

1.4 Roadmap

This dissertation begins with a review of the mathematical techniques that are the foundation of our segmentation algorithm, including pure curve evolution, the level set representation of manifolds, and a formalism for evaluating the correctness of solutions to partial differential equations. We then review deformable models that have been used in computer vision, starting with the classic approach then comparing later approaches and modifications to the initial formulation. Chapter 4 describes the MRA data on which we have focused our experiments and discusses previous MRA segmentation approaches. CURVES is described in Chapter 5 which details its components and discusses some of the design decisions that were made. Results on simulated data and on real MRA data are presented in Chapter 6 along with comparisons to manual segmentations. Chapter 7 describes our algorithm for extracting the centerlines of tubular structures, shows examples on MRA datasets, and discusses related approaches. Chapter 8 shows our preliminary exploration of the use of manifold evolution for general data representation tasks in which this approach is applied to tracking data representing patterns of motion in a street scene. The dissertation concludes with comments on the studies described and suggestions for future work.

Chapter 2

Background Mathematics

CURVES is based on evolving a manifold, over time, according to a partial differential equation comprised of two terms. The first is a regularization force that controls the smoothness of the manifold, and the second is in image-related term that allows image data to influence the shape of the manifold. If one considers the case in which the image term is identically zero, that is, there is no image term, then the partial differential equation becomes an instance of a partial differential equation well-studied in the differential geometry community, *mean curvature flow.* CURVES then uses the *level set method* to implement this differential equation.

Both of these topics were first studied for the case of hypersurfaces, then subsequently for the more difficult case of higher codimensional manifolds. The distinction between these situations is important for an understanding of CURVES, which is an extension of previous computer vision work based on the hypersurface cases of these mathematical concepts to a higher codimensional case. Both concepts are thus described for the two cases separately.

Finally, an important benefit of the level set technique is that one can prove that it solves the given partial differential equation *in the viscosity sense.* An overview of the concept of viscosity solutions is thus provided.

2.1 Mean Curvature Flow

Mean curvature flow refers to some manifold (curve, surface, or other) evolving in time so that at each point, the velocity vector is equal to the *mean curvature vector.*

2.1.1 Hypersurfaces

Let m be a hypersurface in $\mathbb{R}^n$, $\bar{N} = \bar{N}(\bar{x})$ the normal for a given orientation (choice of inside/outside), and $H = H(\bar{x})$ the mean curvature of the manifold. All quantities are parameterized by spatial position $\bar{x}$, an $(n-1)$-tuple. The *mean curvature vector*

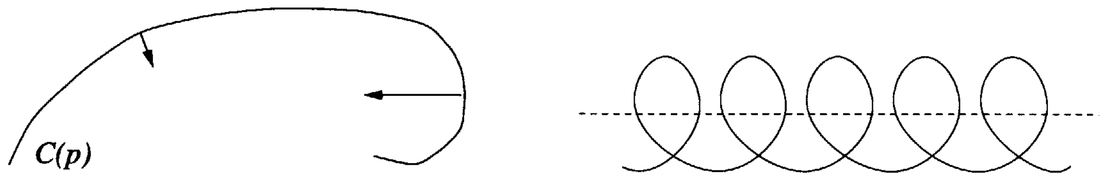


Figure 2-1: Left: A curve $C(p)$ with mean curvature vectors drawn. Right: Under curve-shortening flow for space curves, a helix (solid) shrinks smoothly to its axis (dashed).

or *mean curvature normal vector* $\bar{H}$ of m is defined locally as

$$\bar{H} = H\bar{N}$$

Now, let M be a family of hypersurfaces in $\mathbb{R}^n$ indexed by t, so $M(t)$ is a particular hypersurface. Consider t as "time", so the family describes the manifold's evolution over time. Then *mean curvature flow* is the evolution according to the equation

$$M_t = H\bar{N} \tag{2.1}$$

where M_t is the derivative of M with respect to t and an initial manifold $M(0) = M_0$ is given.

For the case of 1D curves, which are treated in this research, the mean curvature is just the usual Euclidean curvature κ of the curve. Let $C(t)$ be a family of curves indexed by time t. The evolution equation is then

$$C_t = \kappa\bar{N} \tag{2.2}$$

with initial curve $C(0) = C_0$. This motion is pictured in Figure 2-1a where a curve $C(p)$ is drawn along with mean curvature vectors at two different points. Figure 2-1b demonstrates this evolution for a helix in 3D which shrinks smoothly into its axis. Mean curvature flow for the case of 1D curves is also called *curve-shortening flow* since it is the solution, obtained by Euler-Lagrange equations (Appendix A, [37, 51]), to the problem of minimizing Euclidean curve length:

$$\min_C \int |C'(p)|dp$$

where p is the spatial parameter of the curve. That is, we now have $C = C(p,t)$ as a function of both a spatial parameter and a temporal parameter. We will write $C = C(p)$ when we are concerned only with the trajectory of the curve at a particular time t.

Work in the differential geometry community has studied the behavior of hypersurfaces evolving under mean curvature motion. A fundamental theorem in this area

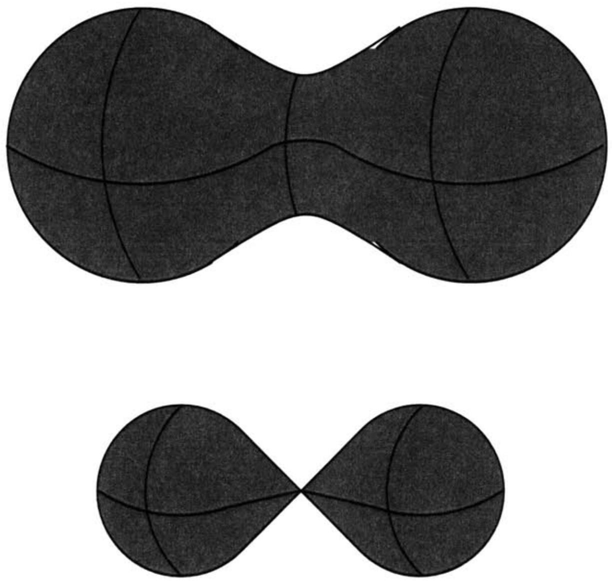


Figure 2-2: A surface in 3D that develops singularities and changes topology under mean curvature motion.

is that smooth closed curves in 2D shrink to a point under mean curvature motion [44, 46]. This theorem was proven in two steps. First, it was proved that smooth closed convex curves shrink to a point under this motion, becoming round in the limit. Second, it was proved that simple (not self-intersecting) closed planar curves become convex under this motion, thus completing the proof. For hypersurfaces of higher dimensionality, however, the behavior is not analogous. The usual counterexample is the case of a dumbbell-shaped surface in $\mathbb{R}^3$, as in Figure 2-2. Such a shape can easily be constructed so that it breaks into two pieces and develops singularities in finite time under mean curvature motion. That is, the evolution is smooth for a finite time period, but after that time period, cannot be continued in the same fashion.

One would like to continue the evolution even after the singularities develop. Presumably, the dumbbell should break into two separate pieces which should shrink smoothly. However, the representation of the shape directly as a parametrized surface cannot support this splitting. A more powerful representation is needed. The level set representation [88, 100] will be described in detail in this document as it is used in our system. The need for a representation that can handle topological changes and singularities is a major motivation for the level set methods. Similarly, the singularities that develop is one instance of the class of problems that motivated the development of *viscosity solutions*, which we will explain in section 2.3. The papers describing the level set method for mean curvature flow of arbitrary-dimensional hypersurfaces prove the correctness of the method for this problem "in a viscosity sense", which means, informally, that the solution is the smoothest approximation possible given the singularities present [38, 22].

We will see in section 3.4 that the image segmentation problem can be defined as minimizing a non-Euclidean curve length over all possible curves. The resulting curve flow equation obtained is closely related to 2.2. We will therefore be able to use much of the mathematical technology and results obtained for the mean curvature flow case for the image segmentation case. The definition of mean curvature is more complicated for manifolds that are not hypersurfaces. For completeness and because our studies have dealt with higher codimensional manifolds, we give this definition below.

2.1.2 Higher Codimension

For manifolds that are not hypersurfaces, there is not a single normal direction. Instead the dimensionality of the normal space is equal to the codimension of the manifold. Thus we no longer have a single well-defined normal direction. However, it turns out that one can compute the mean curvature normal vector uniquely as follows [102]. Let $M^n = M \subset \mathbb{R}^d$ be an n-dimensional manifold in $\mathbb{R}^d$ with codimension $d-n$. Let M_p and $M_p^\perp$ be the tangent and normal spaces, respectively, to M at point p. For any normal vector $\xi \in M_p^\perp$, we can define a mean curvature H_ξ in the direction of ξ. Like in the hypersurfaces case and as is intuitive, curvature measures the change in the tangent directions. Let $X_1, \ldots, X_n$ be vector fields that are tangent to M that are orthogonal to each other and unit length so that $X_1(p), \ldots X_n(p)$ is an orthonormal basis for M_p. We then define the ξ-dependent mean curvature as

$$H_\xi = \frac{1}{n} \sum_{i=1}^{n} \nabla'_{X_i(p)} X_i \cdot \xi$$

where $\nabla'_X(p)Y$ is the covariant derivative of the vector field Y in the direction of $X(p)$.

It turns out that there is a unique vector $\eta(p) \in M_p^\perp$ such that

$$\eta(p) \cdot \xi = H_\xi \quad \text{for all} \quad \xi \in M_p^\perp$$

and that this vector can be computed by summing the vectors obtained by multiplying each element of an orthonormal basis for $M_p^\perp$ by its direction-dependent mean curvature. Let $\nu_{n+1}, \ldots, \nu_d$ be an orthonormal basis for $M_p^\perp$, so

$$\begin{aligned}
\eta(p) &= \sum_{r=n+1}^{d} H_{\nu_r}\nu_r \\
&= \frac{1}{n}\sum_{r=n+1}^{d}(\sum_{i=1}^{n}\nabla'_{X_i(p)}X_i \cdot \nu_r)\nu_r \\
&= \frac{1}{n}\Pi[\sum_{i=1}^{n}\nabla'_{X_i(p)}X_i],
\end{aligned}$$

where Π denotes projection onto the normal space $M_p^\perp$.

As an example, let M be a 1D curve in $\mathbb{R}^3$. So the tangent space is 1-dimensional, and X_1 is the tangent field along M, and $X_1(p)$ is the tangent to M at point p. We will choose the usual unit normal and unit binormal for a space curve as the basis vectors ν_i. These vectors are defined from the tangent t according to the first two Frenet equations ([35])

$$\begin{aligned}
t' &= \kappa N \\
N' &= -\kappa t - \tau B
\end{aligned}$$

where N is the unit normal, B is the unit binormal, κ is the curvature of the curve, and τ is its torsion. Further note that the covariant derivative of the tangent field in the direction of the tangent is the usual derivative t' of the tangent.

We then compute

$$\begin{aligned}
\eta(p) &= \sum_{r=2}^{3}(\sum_{i=1}^{1}\nabla'_{X_i(p)}X_i \cdot \nu_r)\nu_r \\
&= \sum_{r=2}^{3}(\nabla'_{X_1(p)}X_1 \cdot \nu_r)\nu_r \\
&= \sum_{r=2}^{3}(t' \cdot \nu_r)\nu_r \\
&= (\kappa N \cdot N)N + (\kappa N \cdot B)B \\
&= \kappa N + (0)B \\
&= \kappa N.
\end{aligned}$$

Thus, we've shown that this definition for arbitrary dimensions does in fact reduce to the usual definition of the mean curvature vector for a curve.

The case of mean curvature flow for non-hypersurfaces has been less well-explored by the differential geometry community than that of mean curvature flow for hy-

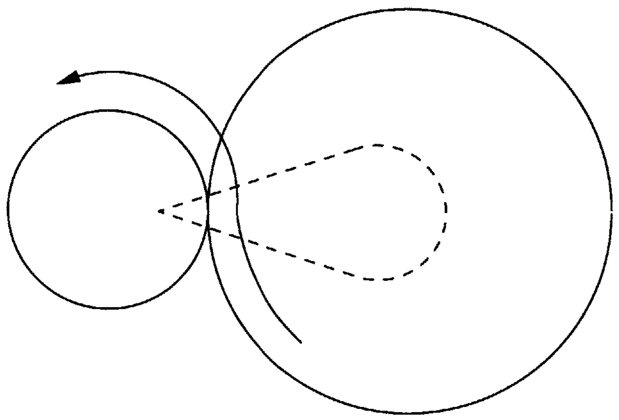


Figure 2-3: Under curve-shortening flow, the initial non-simple curve (solid) evolves into a curve (dashed) which contains a singularity.

persurfaces. The case of curve-shortening flow was studied for 3D curves [3] for the specific application of using this flow to address the limitation of planar curve flow that the curve be simple. When the initial curve is not simple then it can develop singularities after a finite time period, as pictured in Figure 2-3. In this example, the initial curve has a "figure eight" shape, and after evolving for some finite length of time, it develops a corner because the smaller circle in the figure eight shrinks to a point more slowly than does the larger circle. Altschuler and Grayson lifted the curves out of the plane by a small amount at the singular points and used the curve-shortening flow of space curves to implement the evolution of the original planar curve past these singularities.

It is the 3D version of curve-shortening flow which is most relevant to our CURVES system, as the objects to be segmented are modeled as 3D curves undergoing a motion related to curve-shortening flow. This curve flow is implemented in CURVES via level set methods, to which we now turn our attention.

2.2 Level Set Methods

Level set methods increase the dimensionality of the problem from the dimensionality of the evolving manifold to the dimensionality of the embedding space [88, 100]. For the case of a planar curve, one defines a surface which implicitly encodes the curve, then evolves that surface instead of the explicit curve. An example of a surface as an implicit representation of a curve is the signed distance function pictured in Figure 2-4. In this case, let C be the curve, and the surface u is defined so that its value at any point is the signed distance to C, with interior points having negative distance by convention. C is then, by construction, the zero level set of u. The representations are equivalent in information content since one can generate the surface from the curve and the curve from the surface.

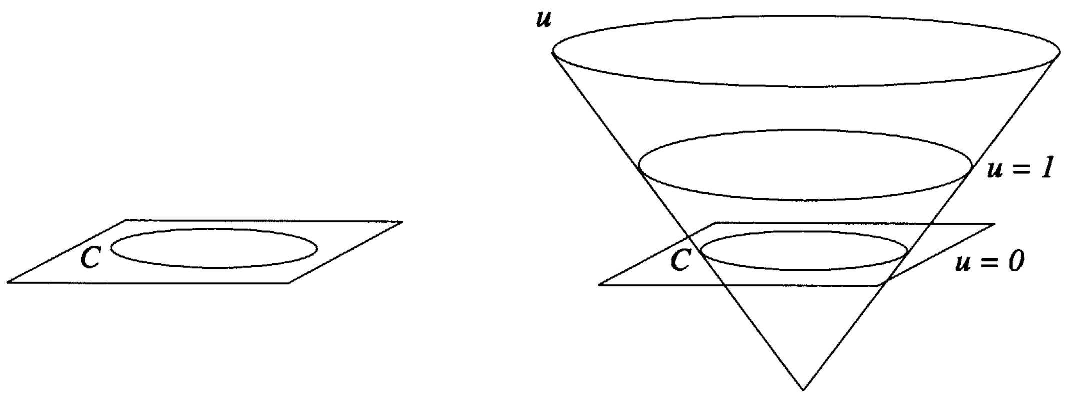


Figure 2-4: Level sets of an embedding function u, for a closed curve in $\mathbb{R}^2$.

2.2.1 Motivation

A brief discussion of the reasons why level set methods were developed is provided before a formal statement of the equivalence that underlies the methods. This discussion draws closely from [100], to which the reader is referred for further discussion.

Imagine we have some front C propagating over time. A necessary requirement for the use of the level set approach is that we care only about the front as a boundary between two regions. That is, we do not care about tangential motion of the front, but only about motion in the normal direction (Figure 2-5). As before, C is a function of both a spatial parameter and a time parameter, so we write $C = C(p, t)$ where $p \in [0, 1]$ is the spatial parameter and $t \geq 0$ is the time parameter.

We can write some equation for this motion as a partial differential equation: at each point on the curve C, the derivative of C with respect to time t is equal to a speed β times the normal $\bar{N}$

$$C_t = \beta \bar{N}, \tag{2.3}$$

with initial condition $C(\cdot, 0) = C_0(\cdot)$.

One problem with an explicit evolution of the curve is that singularities develop over time. This problem is a primary motivation for the use of level set methods. To understand this problem, first consider the case in which the initial curve can be written as a function of its spatial parameter. We will extend the intuition to functions that cannot be so written below; we discuss this case first because the derivatives are more natural.

The straightforward, explicit way to evolve the curve is to compute the normals analytically and use those normals to recompute the position of the curve at successive time steps. Consider the example of a cosine curve and assume we would like to propagate it forward in time with constant speed $\beta = 1$. This curve is plotted as the lowest curve in Figure 2-6(a). Successive curves are plotted above this curve of the evolution forward in time. The difficulty occurs at the convexity of the original curve.

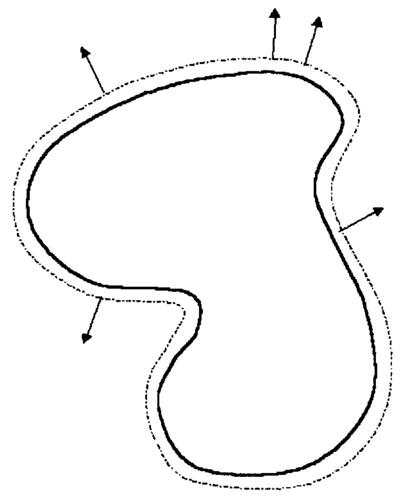


Figure 2-5: A front propagating over time, where motion is assumed to be only in the direction of the normal vectors of the front.

A singularity develops eventually, after which time the normal is ambiguous. This ambiguity implies that the evolution is no longer well-defined. Retaining all possible normal directions would cause the two "sides" of the front to intersect.

In order for the evolution to be well-defined, we need to choose one solution. The solution we choose is what is called the *entropy solution*, pictured in Figure 2-6(b). This solution is defined by the property that once a region has been crossed by the front, that region cannot be crossed again so the front cannot cross itself. One often uses the analogy of a fire moving across a field, in which case the behavior is characterized by the statement, "Once a particle is burnt, it stays burnt." This statement is referred to as the *entropy condition* and the solution it implies as the *entropy solution* because of the relation to information propagation. Information has becomes lost when the singularity develops. Since the normal is not unique at that time and location, we cannot reverse the propagation to obtain the original curve.

We now make concrete the notion of "entropy solution". For the current cosine example, if we change the speed β from $\beta = 1$ to $\beta = 1 - \varepsilon\kappa$, for some small ε and where κ is the Euclidean curvature of the front, then the singularities do not form. As ε approaches zero, the evolution approaches the entropy solution. Let $X^{\varepsilon}_{curvature}$ be the evolution (the sequence of curves) obtained by using speed term $\beta = 1 - \varepsilon\kappa$ and $X_{entropy}$ be the evolution obtained with the entropy condition. It can be proved that the limit of the $X^{\varepsilon}_{curvature}$'s as ε approaches zero is identically the entropy solution:

$$\forall t, \lim_{\varepsilon \to 0} X^{\varepsilon}_{curvature}(t) = X_{entropy}(t).$$

This limit, *i.e.* the entropy solution, is called the *viscous limit* and also the *viscosity solution* of the given evolution. Viscosity solutions will be defined from a more formal perspective in section 2.3 below. The reason this limit is called the *viscous limit* is

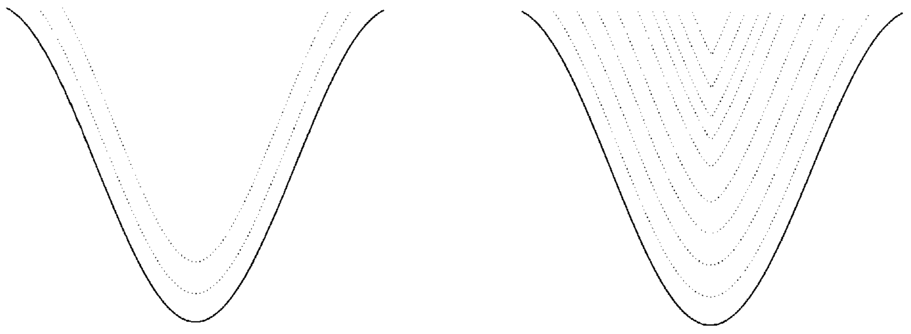


Figure 2-6: Left image: the bottom curve is the initial cosine curve and the higher curves are successive curves obtained by propagating with normal speed $\beta = 1$. Right image: continuing propagation after normal becomes ambiguous using entropy solution.

its relevance to fluid dynamics. In fluid dynamics, any partial differential equation of the form

$$u_t + [G(u)]_x = 0$$

is known as a *hyperbolic conservation law.* A simple example is the motion of compressible fluid in one dimension, described by Burger's equation:

$$u_t + uu_x = 0.$$

To indicate viscosity of the fluid, one adds a *diffusion* (spatial second derivative) term to the right hand side to obtain

$$u_t + uu_x = \varepsilon u_{xx}$$

where ε is the amount of viscosity. A well-known fact in the fluid dynamics community is that for $\varepsilon > 0$, this motion remains smooth for all time: singularities, or *shocks* as they are called in the fluid dynamics community, do not develop.

To relate the viscous limit to front propagation, we return to the idea of a curve propagating as in Figure 2-6. Let $C = C(p)$ be the evolving front and C_t the change in height of C in a unit time step. Referring to Figure 2-7, observe that the tangent at (p, C) is $(1, C_p)$ and notice that

$$\frac{C_t}{\beta} = \frac{(1 + C_p^2)^{1/2}}{1},$$

which gives the update rule

$$C_t = \beta(1 + C_p)^{1/2}$$

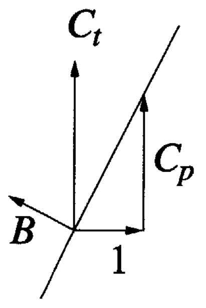


Figure 2-7: Computation of motion C_t for relation of viscosity to front propagation.

Setting the speed term to $F = 1 - \varepsilon\kappa$ and observing that $\kappa = C_{pp}/(1+C_p^2)^{1/2}$ yields an equation which relates the motion C_t of the curve C to its spatial derivatives:

$$C_t - (1+C_p^2)^{1/2} = \frac{\varepsilon C_{pp}}{(1+C_p^2)}.$$

Differentiating with respect to p and setting $u = \frac{dC}{dp}$ to be the slope of the propagating front gives an evolution equation for u:

$$u_t + [-(1+u_p^2)^{1/2}]_p = \varepsilon[\frac{u_p}{(1+u_p^2)}]_p.$$

Observe that this evolution equation is a hyperbolic conservation law. We observe that the curvature term in the speed function for a curve plays exactly the same role as the viscosity term in the evolution equation for the slope of the curve. It then follows that we can use the technology and theorems from fluid dynamics to prove that no singularities can develop in the slope of the front.

Sometimes we want to describe the motion of curves that are not expressible as functions. We thus cannot propagate them analytically. More importantly, we cannot detect singularities explicitly without the analytical form. Level set methods were developed to address exactly this issue. To summarize, they

- apply only to the situations in which we care about motion in the normal direction but not in the tangential direction,
- address problems of singularities developing during an evolution,
- evolve such fronts according to their *entropy* or *viscosity* solutions, and
- are well-suited to boundaries which are not expressible as functions.

2.2.2 Hypersurfaces

We now explain the specific equivalence that underlies the level set methods. For the example of planar curves, let $u : \mathbb{R}^2 \to \mathbb{R}$ be the signed distance function to curve

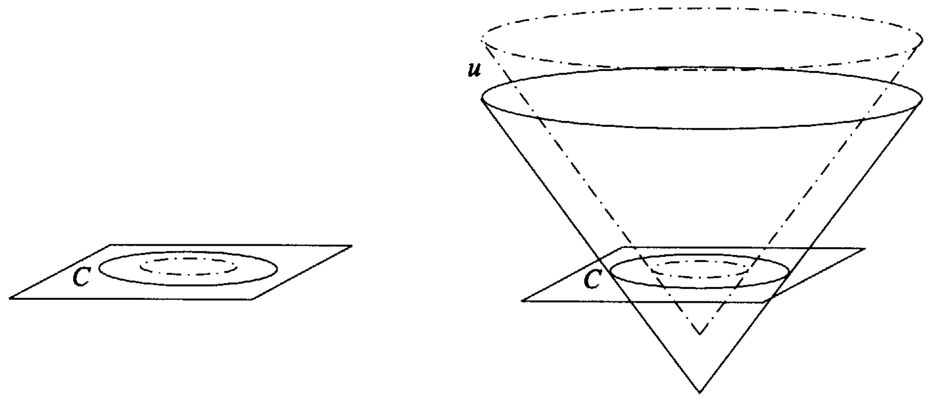


Figure 2-8: Left: the initial value of the curve C is shown as a solid curve, and a later value is shown as a dashed curve. Right: The embedding surface u at the initial time (solid) and the later time (dashed).

C as in Figure 2-4, so C is the zero level-set of u. Let C_0 be the initial curve. It is shown in [38, 22] that evolving C according to Equation 2.3 with initial condition $C(\cdot,0) = C_0(\cdot)$ for any function β, is equivalent to evolving u according to

$$u_t = \beta|\nabla u| \tag{2.4}$$

with initial condition $u(\cdot,0) = u_0(\cdot)$ and $u_0(C_0) = 0$ in the sense that the zero level set of u is identical to the evolving curve for all time. Referring to Figure 2-8 as an example, if the solid initial curve in the left image evolved to the dashed curve, the embedding surface would evolve as pictured in the right image.

Although this method may initially appear less efficient since the problem is now higher-dimensional, it has important advantages. First, it facilitates topological changes in the evolving manifold (the curve) over time. Evolving a curve directly necessarily relies on a particular explicit parameterization of the curve, which makes topological changes cumbersome, requiring special cases, whereas an implicit representation can handle them automatically as will be seen throughout this document. Parameterization is the heart of the second advantage of the implicit method as well: it is intrinsic (independent of parameterization). That is, it is an *Eulerian* formulation of the problem which updates values at fixed grid points instead of a *Lagrangian* formulation which would move the points of the curve explicitly. The Eulerian formulation is desirable for modeling curves in many applications including image segmentation since parameterizations are related only to the speed at which the curve is traversed, but not to its geometry. It is therefore problematic for a segmentation algorithm to depend on the curve's parameterization.

To see the equivalence geometrically, consider the zero level set

$$\{\Gamma \in \mathbb{R}^2 : u(\Gamma,t) = 0\}$$

Differentiating with respect to t gives

$$\nabla u \cdot \Gamma_t + u_t = 0$$

Further note that for any level set

$$\frac{\nabla u}{|\nabla u|} = -\bar{N},$$

where $\bar{N}$ is the inward-pointing normal of the level set (for the case of a planar curve embedded in a surface, this level set is a curve). Substituting,

$$-\bar{N}|\nabla u| \cdot \Gamma_t + u_t = 0.$$

We want to define the evolution u_t of u so that $\Gamma \equiv C$ for all time; that is $\Gamma(t) \equiv C(t)$. Having already initialized $\Gamma(0) \equiv C(0)$, we only need set $\Gamma_t = C_t$ to obtain

$$u_t = \beta|\nabla u|.$$

This derivation was given in [20], and is very similar to the derivation given in [87]. Moreover, it applies to hypersurfaces in any dimension [38, 22]; planar curves were used as an example for simplicity only.

It was shown that any embedding function u that is smooth in some open region of $\mathbb{R}^N$ and for some time period and whose spatial gradient does not vanish in that region and time period is a *viscosity solution* to Equation 2.4 [38, 22]. Specifically, u must be Lipschitz continuous, where *Lipschitz* means that it cannot grow unboundedly: function f is Lipschitz if

$$|f(x) - f(y)| \leq C|x - y|$$

for all x, y in the domain of f and for some constant C. It is also unnecessary to choose the *zero* level set of u: any isolevel set suffices, although the zero level set is the standard choice. For further detail, the reader is referred to [100], the primary reference for level set methods, implementation issues, and applications.

One limitation of this body of work, as described until this point, is the restriction to hypersurfaces (manifolds of codimension one). The examples of a planar curve and a three-dimensional surface have codimension one, but space curves have codimension two. Intuition for why the level set method above no longer holds for space curves is that there is not an "inside" and an "outside" to a manifold with codimension larger than one, so one cannot create the embedding surface u in the same fashion as for planar curves; a distance function must be everywhere positive, and is thus singular on the curve itself. The more recent discovery of level set equations for curvature-based evolution in higher codimension [4], however, overcame this limitation.

2.2.3 Higher Codimension

Ambrosio and Soner provided level set evolution equations for mean curvature flow of an arbitrary dimensional manifold in arbitrary dimensional space. Further, they gave the extension to more general curvature-based evolution which can incorporate an auxiliary externally defined vector field [4]. It is these equations that inspired our CURVES segmentation algorithm which models blood vessels as curves in 3D, which have codimension two. In particular, the auxiliary vector field equation enables the use of image information in addition to regularization. We now state these equations for the general case, as in [4].

Imagine we wish to represent and evolve some manifold $\Gamma \subset \mathbb{R}^d$ implicitly. Further assume that the codimension of Γ is $k > 1$. Let $v : \mathbb{R}^d \to [0, \infty)$ be an embedding function whose zero level set is identically Γ, that is smooth near Γ, and such that ∇v is non-zero outside Γ. For a nonzero vector $\mathbf{q} \in \mathbb{R}^d$, define

$$P_{\mathbf{q}} = I - \frac{\mathbf{q}\mathbf{q}^T}{|\,\mathbf{q}\,|^2}$$

which is the projection operator onto the plane normal to $\mathbf{q}$. Let $X = P_{\mathbf{q}} A P_{\mathbf{q}}$ for some matrix A and let

$$\lambda_1(X) \leq \lambda_2(X) \leq \ldots \leq \lambda_{d-1}(X)$$

be the eigenvalues of X corresponding to the eigenvectors *orthogonal* to $\mathbf{q}$. Notice that $\mathbf{q}$ will always be an eigenvector of X with eigenvalue 0.

Further define

$$F(\mathbf{q}, A) = \sum_{i=1}^{d-k} \lambda_i(X).$$

Assuming the general case in which the only 0 eigenvalue corresponds to the eigenvector orthogonal to $\mathbf{q}$, we can say that F is the sum of the $d - k$ smallest nonzero eigenvalues of X. It is then proved that the level set evolution equation to evolve Γ by mean curvature flow is

$$v_t = F(\nabla v(x,t), \nabla^2 v(x,t)). \tag{2.5}$$

That is, this evolution is equivalent to evolving Γ according to mean curvature flow in the sense that Γ is identical to the zero level set of v throughout the evolution.

For intuition, consider v as a distance function to Γ which thus satisfies $|\nabla v| = 1$ everywhere except at Γ, although other functions are possible. Consider an isolevel set $\Gamma_\varepsilon = \{x | v(x) = \varepsilon\}$ of v where ε is very small and positive. Then the eigenvalues of $\frac{1}{|\nabla v|}X$ that are orthogonal to ∇v are precisely the principal curvatures of the hypersurface Γ_ε, oriented by ∇v. Since Γ has codimension k, we expect Γ_ε to have

$k-1$ very large principal curvatures and $d-k$ principal curvatures related to the geometry of Γ. For the lowest dimensional case in which Γ is a curve in $\mathbb{R}^3$, Γ_ϵ is a thin tube around Γ, the larger principal curvature corresponds to the small radius of the tube, and the smaller principal curvature corresponds to the geometry of Γ. Note that the eigenvalues of $\frac{1}{|\nabla v|}X$ are exactly the eigenvalues of x scaled by $\frac{1}{|\nabla v|}$. Thus the use of X instead of $\frac{1}{|\nabla v|}X$ in Equation 2.5 is the same as using $\frac{1}{|\nabla v|}X$ and scaling by $|\nabla v|$ as in

$$v_t^{(*)} = |\nabla v| \sum_{i=1}^{d-k} \lambda_i \left(\frac{1}{|\nabla v|} X\right).$$

This alternate expression is perhaps more intuitive since the sum corresponds to curvatures so it looks like Equation 2.4.

Consider the situation in which there is an underlying vector field driving the evolution, in combination with the curvature term, so that the desired evolution equation is of the form

$$\Gamma_t = \kappa \bar{N} - \Pi \bar{d}, \tag{2.6}$$

where Γ_t gives the motion of the manifold Γ over time, κ is the mean curvature of Γ, Π is the projection operator onto the normal space of Γ (which is a vector space of dimension $d-k$) and $\bar{d}$ is a given vector field in $\mathbb{R}^d$. The evolution equation for the embedding space then becomes

$$v_t = F(\nabla v, \nabla^2 v) + \nabla v \cdot \bar{d}. \tag{2.7}$$

2.2.4 Numerical Techniques

From the equations above, one does not yet have the full story about how to implement such evolutions. An underlying theme to both the direct curve evolution and the level-set based curve evolution is that we wish to compute temporal derivatives in terms of spatial derivatives. There is some discomfort in this notion since although the two are intimately related, it is not natural to regard one as a function of the other. In order to make this construction feasible, one must be careful about how exactly the spatial gradients are computed. Specifically, one should not use values at points that correspond to future times, in which the evolution has not yet occurred. That is, information should propagate in the same direction as the evolving front. Methods for gradient computation and for overall evolution are called *upwind* if they respect this restriction on information flow. That is, they use only values that are upwind of the direction of information propagation. This stability issue arises in the computation of the gradients needed in the update equation, Equation 2.4. Sethian discusses these issues in detail in his book [100]; we will provide intuition and then state the choices used for gradient computations.

A function $f : \mathbb{R} \to \mathbb{R}$ is said to be *convex* if its second derivative is everywhere positive: $f''(x) > 0$. In higher dimensions, a function $G : \mathbb{R}^N \to \mathbb{R}$ is convex if it is smooth and all second derivatives are everywhere non-negative: $\frac{\partial^2 G}{\partial x_i \partial x_j} \geq 0$. One can also use an alternate definition that includes non-differentiable functions: f is *convex* if $f(\tau x + (1-\tau)y) \leq \tau f(x) + (1-\tau)f(y), \forall x, y \in \mathbb{R}^N, \tau \in [0,1]$. In the generalized evolution equation

$$u_t = H(x, u(x), u'(x)),$$

the term $H(x, u(x), u'(x))$ is called the *Hamiltonian.* In Equation 2.4, the Hamiltonian is $\beta|\nabla u| = \beta\sqrt{{u_x}^2 + {u_y}^2 + {u_z}^2}$, assuming 3 dimensions. We say that the speed function β in an evolution equation is *convex* if the Hamiltonian $\beta|\nabla u|$ is a convex function.

Sethian provides numerical schemes for computation of gradients for both convex and non-convex speed functions, and also for first order approximations to the gradient and for second order approximations to the gradient [100]. We review the first-order convex case here for discussion purposes. First, some notations:

The *central difference operator*, *forward difference operator*, and *backward difference operator* for first-order approximating the spatial derivative of u in the x dimension, at grid point (i, j, k), are, respectively,

$$\begin{aligned} D^x_{ijk} &= D^x_{ijk}u = \frac{u_{i+1,j,k} - u_{i-1,j,k}}{2\Delta x} \\ D^{+x}_{ijk} &= D^{+x}_{ijk}u = \frac{u_{i+1,j,k} - u_{i,j,k}}{\Delta x} \\ D^{-x}_{ijk} &= D^{-x}_{ijk}u = \frac{u_{i,j,k} - u_{i-1,j,k}}{\Delta x}. \end{aligned}$$

Taylor series expansions around x show that

$$\begin{aligned} u_x &= D^x_{ijk} + O(h^2) \\ u_x &= D^{+x}_{ijk} + O(h) \\ u_x &= D^{-x}_{ijk} + O(h), \end{aligned}$$

where h is the spacing between adjacent grid points. We thus see that the central difference scheme yields the most accurate approximation. However, when handling moving fronts, the issue of direction of information propagation is also important, so we will see that it is suboptimal in some cases.

First Order, Space Convex

An example of a convex speed function is a constant speed function applied to an embedding space u which has constant slope. Let F be any convex speed function, and assume we have the evolution equation

$$u_t + F|\nabla u| = 0.$$

The first order upwind scheme for iteratively solving this equation ([100, 88]) is then

$$u_{ijk}^{n+1} = u_{ijk}^{n} - \Delta t[\max(F_{ijk}, 0)\nabla^{+} + \min(F_{ijk}, 0)\nabla^{-}], \tag{2.8}$$

where

$$\begin{aligned}\nabla^{+} = [&\max(D_{ijk}^{-x}, 0)^2 + \min(D_{ijk}^{+x}, 0)^2 + \\ &\max(D_{ijk}^{-y}, 0)^2 + \min(D_{ijk}^{+y}, 0)^2 + \\ &\max(D_{ijk}^{-z}, 0)^2 + \min(D_{ijk}^{+z}, 0)^2]^{\frac{1}{2}},\end{aligned}$$

and

$$\begin{aligned}\nabla^{-} = [&\max(D_{ijk}^{+x}, 0)^2 + \min(D_{ijk}^{-x}, 0)^2 + \\ &\max(D_{ijk}^{+y}, 0)^2 + \min(D_{ijk}^{-y}, 0)^2 + \\ &\max(D_{ijk}^{+z}, 0)^2 + \min(D_{ijk}^{-z}, 0)^2]^{\frac{1}{2}}.\end{aligned}$$

To demonstrate the importance of choosing the correct scheme for gradient computation, we constructed a simple experiment in which a front is propagating outward at constant speed $F = 1$. Assuming u is a distance function so $|\nabla u| = 1$ everywhere, this speed function is convex because $F|\nabla u|$ is constant so all of its second derivatives are 0. In this case, we have

$$u_t + 1|\nabla u| = 0,$$

and Equation 2.8 becomes

$$\begin{aligned}u_{ijk}^{n+1} &= u_{ijk}^{n} - \Delta t[\nabla^{+}] && (2.9)\\ &= u_{ijk}^{n} - \Delta t[\max(D_{ijk}^{-x}, 0)^2 + \min(D_{ijk}^{+x}, 0)^2 + && (2.10)\\ &\quad \max(D_{ijk}^{-y}, 0)^2 + \min(D_{ijk}^{+y}, 0)^2 + && (2.11)\\ &\quad \max(D_{ijk}^{-z}, 0)^2 + \min(D_{ijk}^{+z}, 0)^2]^{\frac{1}{2}}. && (2.12)\end{aligned}$$

Imagine that each slice of our embedding surface is identical at time $t = 0$, and looks like the "Initial" solid plots in Figure 2-9. That is, the front is defined by two

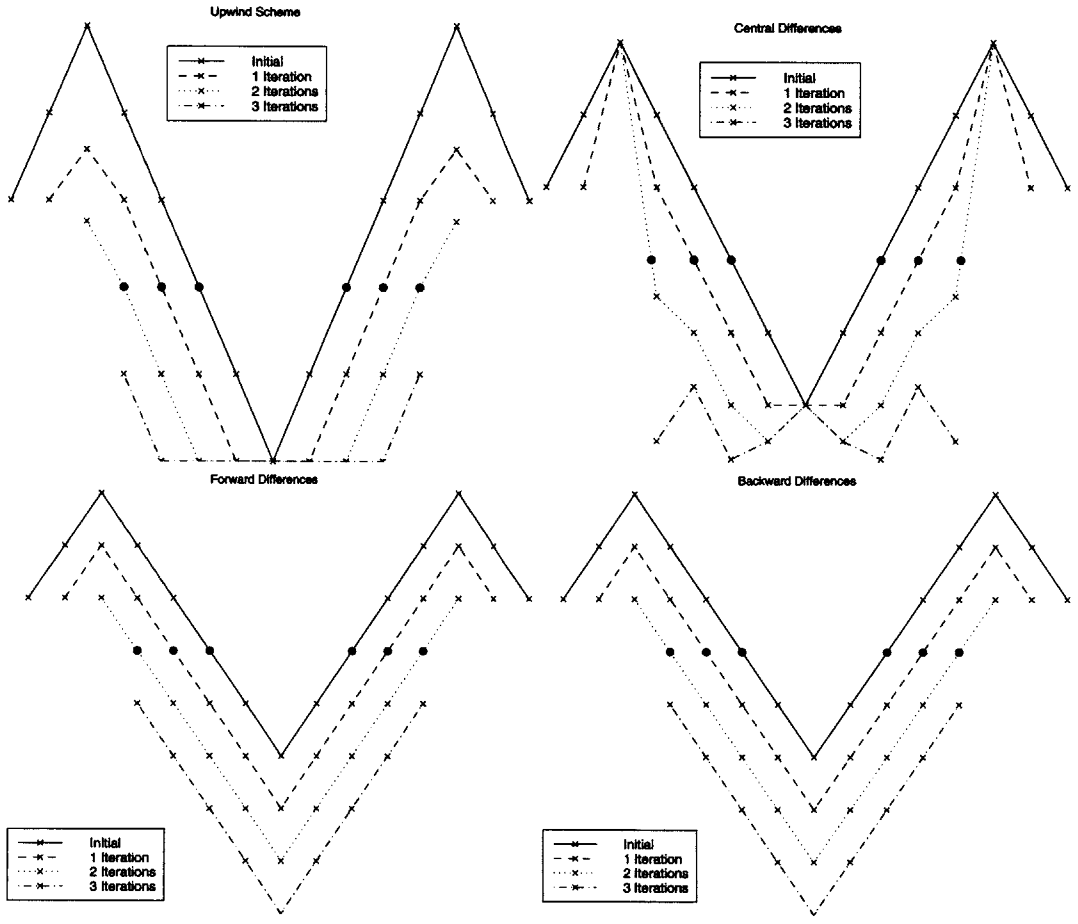


Figure 2-9: Cross-sectional slices for a front moving with constant unit speed. Behavior is shown for four different numerical methods of gradient approximation: upper left is upwind scheme, upper right is central difference scheme, lower left is forward difference scheme, and lower right is backward difference scheme.

straight lines in the direction normal to the figures, and the outward direction is the direction towards the edges of the plots. What Figure 2-9 shows is the evolution computed, for $\Delta t = 1$ and three iterations, for four difference methods of finite gradient approximation. The upper left plot shows the upwind scheme, the upper right the central difference scheme D^x_{ijk}, the lower left the forward difference scheme D^{+x}_{ijk}, and the lower right the backward difference scheme. Points on the zero level set which correspond to the contour itself moving over time are marked in black. For this particular example, $|\nabla u|$ is one everywhere initially. This implies that any one-sided difference scheme will work: the backward and forward schemes thus give the desired evolution, as does the upwind scheme away from the singularities. The upwind scheme differs from the desired behavior at exactly the singularities because it is at those points that it chooses either both or neither of the difference operators,

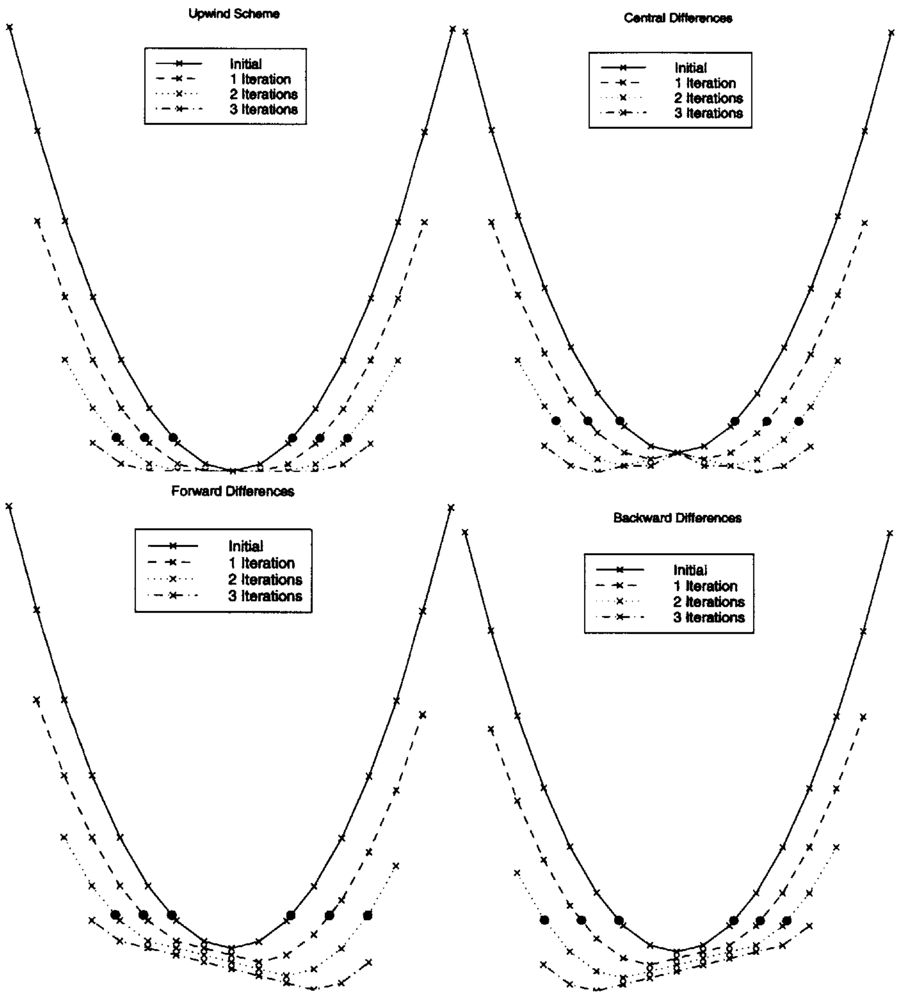


Figure 2-10: Cross-sectional slices for another front moving with constant unit speed. Upper left is upwind scheme, upper right is central difference scheme, lower left is forward difference scheme, and lower right is backward difference scheme.

so is not using exactly one of them and is thus not obtaining a slope of one. Notice that the central difference scheme, although theoretically more accurate than the one-sided schemes, gives worse behavior. It develops a local maximum at the center point of the plot, and has developed multiple undesirable local maxima by the third iteration.

Figure 2-9 shows an example in which any one-sided scheme is effective. When $|\nabla u| \neq 1$, however, one cannot simply use any one-sided scheme. In general, even if one initializes the evolution with an embedding function u such that $|\nabla u| = 1$ everywhere, the evolution will not maintain this invariant over time with a non-constant speed function. Hence, one should not use simple forward or backward differencing in general. Figure 2-10 demonstrates a case in which the initial embedding surface u is parabolic, so does not have slope one. In this case, we are still using constant speed $F = 1$. Note that we still have a convex speed function because the second derivatives are constant for a parabolic u and constant F. Observe that the forward and backward differencing schemes yield different behaviors, with the zero-level-set points becoming spaced unequally on the opposite sides of the center. This is clearly wrong. The central differencing scheme naturally maintains symmetry, but develops an undesirable singularity at the original center point. The upwind scheme is most effective here.

Hybrid Speed Functions

Imagine a speed function F is the sum of multiple speed functions each of which would require different schemes for computation of ∇u. In this case, Sethian advises to use the correct, and different, gradient computations for each term [100]. The example of a speed function which is a sum of a curvature term and a term related to an external vector field is important for the application of image segmentation, as we will see in Chapter 3.4, so we will describe that case here. This is a subpart of an example provided in [100]. Assume

$$F = F_{curv} + F_{ext},$$

where $F_{curv} = -\varepsilon\kappa$ is the curvature-related component of the speed with ε a small positive constant and κ the curvature of the front, $F_{ext} = \vec{E}(x,y)\cdot\vec{n}$ is the component that depends on an external vector field, with $\vec{n}$ the unit normal to the front. The equation for the embedding space u is then

$$u_t = \varepsilon\kappa|\nabla u| - \vec{E}(x,y)\cdot\nabla u.$$

The curvature term is based on second derivatives so uses information from both sides of the front; that is, information propagates in both directions. Thus, Sethian advocates the uses of central differences for the curvature term. The other term corresponds to a speed independent of u, so one should use an upwind scheme just as in the previous example of $F = 1$. The numerical update rule for u becomes

$$u_{ij}^{t+1} \leftarrow u_{ij}^{t} + \Delta t \begin{bmatrix} +[\varepsilon K_{ij}^{t}({D_{ij}^{x}}^{2} + {D_{ij}^{y}}^{2})^{\frac{1}{2}}] \\ \\ -\begin{array}{c}[\max(f_{ij},0)D_{ij}^{-x} + \min(f_{ij},0)D_{ij}^{+x} \\ +\max(g_{ij},0)D_{ij}^{-y} + \min(g_{ij},0)D_{ij}^{+y}]\end{array} \end{bmatrix}$$

where $\vec{E} = (f,g)$ and K_{ij}^{t} is the approximation to the curvature computed using second differences on the level set (Equation 5.10).

2.3 Viscosity Solutions

We would like to say that the curve evolution is well-defined for all time and that the specific embedding function used (usually the distance function) is a formal solution to the partial differential equation (PDE) defining the evolution. Unfortunately, we will see that this PDE does not admit a classical solution. This is because there may be singularities in the "solution" at time $t = 0$ or singularities may develop over time, in which cases we cannot have the differentiability required for a classical solution.

Instead, we will settle for a "generalized" solution that allows the development of non-differentiable points. Moreover, we will see that the distance function is such a solution. The framework used to justify this solution is that of *viscosity solutions.* This new topic in mathematics was first introduced [30] and was developed to make rigorous the idea of generalized solutions to PDE's when classical solutions do not exist. The primary reference for this topic is [29], and a collection of recent lectures by leaders in this field is found in [9].

2.3.1 Definition

The notion of viscosity solutions applies to scalar second order PDE's

$$F(x, u, Du, D^2u) = 0 \tag{2.13}$$

where $x \in \Omega$, $\Omega \subset \mathbf{R}^{\mathbf{N}}$ is open, and $u : \Omega \to \mathbf{R}$ is the unknown function we seek. Du and D^2u correspond to the first and second derivatives of u. We say "correspond" because these operators must be defined even at singularities of u so are not equal to the classical derivatives in those cases.

We further assume that F is *proper.* This means that F is nondecreasing in u and nonincreasing in D^2u:

$$F(x, s, p, X) \leq F(x, r, p, Y)$$

for $Y \leq X$, $s \leq r$ where matrix inequality is defined as $Y \leq X \iff \xi^T Y \xi \leq \xi^T X \xi$, all $\xi \in \mathbb{R}^N$. The reason that F must be proper will be stated below.

The definitions of upper and lower semicontinuous functions are needed next. Let Ω be an open subset of $\mathbf{R}^N$ and $u : \Omega \to \mathbf{R}$. Then the *upper semicontinuous envelope* u^* and *lower semicontinuous envelope* u_* of u are defined as

$$u^*(x) = \lim_{r \downarrow 0} \sup\{u(y) : y \in \Omega, |y - x| \leq r\}$$
$$u_*(x) = \lim_{r \downarrow 0} \inf\{u(y) : y \in \Omega, |y - x| \leq r\}$$

One then says that u is upper semicontinuous if $u = u^*$ and lower semicontinuous if $u = u_*$. In other words, the upper semicontinuous envelope of u is the smallest

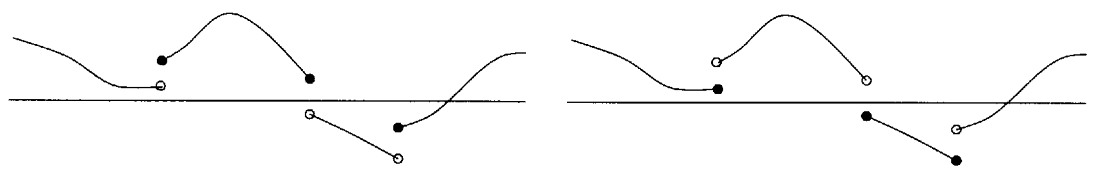


Figure 2-11: (a) An upper semicontinuous function. (b) A lower semicontinuous function.

upper semicontinuous function that is $\geq u$, and the lower semicontinuous envelope of u is the largest lower semicontinuous function that is $\leq u$.

Figure 2-11 illustrates these definitions with examples in $\mathbf{R}^1$. The first graph shows an upper semicontinuous function that is not lower semicontinuous, and the second graph shows a lower semicontinuous function that is not upper semicontinuous. Intuitively, upper semicontinuity means that at any discontinuities the function value is equal to the higher limit; lower semicontinuity means that at any discontinuities the function value is equal to the lower limit. A function that is both upper and lower semicontinuous is continuous.

We are now ready to define viscosity solutions. u is a *viscosity subsolution* of $F = 0$ in Ω if it is upper semicontinuous and for every $\varphi \in \mathbf{C}^2(\Omega)$ and local maximum point $\hat{x} \in \Omega$ of $u - \varphi$, we have $F(\hat{x}, u(\hat{x}), D\varphi(\hat{x}), D^2\varphi(\hat{x})) \leq 0$. Analogously, u is a *viscosity supersolution* of $F = 0$ in Ω if it is lower semicontinuous and for every $\varphi \in \mathbf{C}^2(\Omega)$ and local minimum point $\hat{x} \in \Omega$ of $u - \varphi$, we have $F(\hat{x}, u(\hat{x}), D\varphi(\hat{x}), D^2\varphi(\hat{x})) \geq 0$. Finally, u is a *viscosity solution* of $F = 0$ in Ω if it is both a viscosity subsolution and a viscosity supersolution (thus continuous) of $F = 0$.

2.3.2 Illustration

What does this mean? The goal is to characterize some type of solutions u to PDE's that may not be once (or twice) differentiable. That is why one introduces the φ functions which are by assumption twice differentiable. This differentiability is why we see $D\varphi$ and $D^2\varphi$ as arguments of F in the definition. To gain intuition, consider the φ's that are closest to u. Figure 2-12 provides a one-dimensional example on $\Omega = (-1, 1)$ in which $u(x) = 1 - |x|$ is shown by the solid lines. Note that this u is continuous so both semicontinuity constraints are satisfied. For the φ (dashed line) in the first figure, $\hat{x} = 0$ is a local maximum of $u - \varphi$. So, we would evaluate $F(0, u(0), D\varphi(0), D^2\varphi(0))$ for whatever F is under consideration, and if the result is ≤ 0, and if this is true for all φ's such that $u - \varphi$ has local maxima and for all such local maxima, then u is a viscosity subsolution of $F = 0$.

To determine if it is also a supersolution, we would consider all φ's such that $u - \varphi$ have local minima and all local minima. In the second graph in Figure 2-12, the local minimum is attained at 0, so we evaluate $F(0, u(0), D\varphi(0), D^2\varphi(0))$ for the given F. If this and all other local minima give results ≥ 0, then u is a viscosity

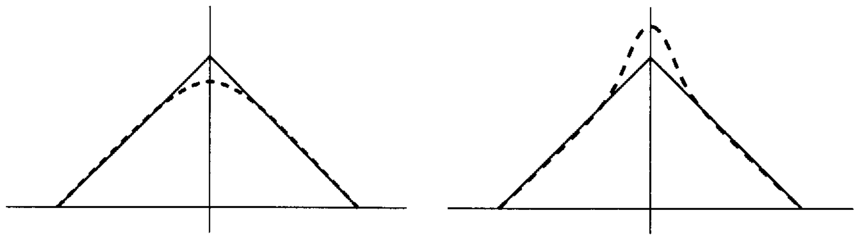


Figure 2-12: Illustrations to accompany explanation of subsolutions and supersolutions. For both examples, u is the solid line, and the φ's are the dashed lines.

supersolution of $F = 0$. If u is both a subsolution and a supersolution of $F = 0$, then it is a (viscosity) solution of $F = 0$.

The reason F must be proper is to ensure that if $u - \varphi$ has a nonnegative maximum ($\varphi - u$ has a nonpositive minimum) at $\hat{x}$, then $F(\hat{x}, \varphi(\hat{x}), D\varphi(\hat{x}), D^2\varphi(\hat{x})) \leq (\geq) F(\hat{x}, u(\hat{x}), Du(\hat{x}), D^2u(\hat{x}))$. The proof of these inequality is that F is nonincreasing in its second argument, $D\varphi(\hat{x}) = Du(\hat{x})$ since x is an extremum of $u - \varphi$, $D^2u(\hat{x}) < (>) D^2\varphi(\hat{x})$ since $\hat{x}$ is a maximum (minimum), and F is nonincreasing in its fourth argument.

To conclude, if u satisfies the above definition for a particular scalar second order PDE $F = 0$, then even though u may not be twice (or even once) differentiable, we will say that it "solves" the PDE, but the solution is in the viscosity sense instead of in the classical sense.

Chapter 3

Active Contour Models

There has been much excitement in the past decade over the development of *active contour models*, or *snakes*, for segmentation tasks in computer vision. Such deformable models iteratively update an initial contour so that the contour is attracted to true object boundaries in the image.

The era began with the model described by Kass, Witkin, and Terzopoulos in 1988 [54]. This model is now known as *classical snakes* to distinguish it from later models, including *balloons* [26] , *T-snakes* [80, 82], and *geodesic snakes* [20, 17, 56, 96, 18, 111]. The curvature-based evolution schemes on which this dissertation focuses are the last in this list, called *geodesic snakes* or *geodesic active contour models.* They are in some cases equivalent to the classical models but are implemented with more sophisticated mathematical tools which enable automatic topological flexibility and increased numerical stability. A related segmentation approach which explicitly deals with shocks is found in [105].

Although this body of methods has enjoyed use in a variety of computer vision domains, they are especially popular in medical image analysis where the smooth irregular anatomical shapes present are well-handled by the active contour approach [111]. A survey of such models in this domain is found in [81]. Likewise, our primary application is a medical application, the segmentation of blood vessels in magnetic resonance angiography data. Some other applications that have benefitted from deformable models approaches include motion tracking [12] and stereo-based depth reconstruction [40]. Also, in addition to local image gradient information, as given below for illustration, more complicated image forces have been used to drive the segmentation process, such as texture [90, 96, 73] and local orientation [96]. Most of these models were initially described for segmentation of 2D images and have been extended to the segmentation of 3D images.

Besides varying the image force, the underlying representation of the contour has been varied. Cohen and Cohen described "finite element" snakes in 2D and 3D [27]. Menet *et al* described "B-snakes" [84] which are based on B-splines constructed through the node points, and Staib and Duncan used Fourier descriptors as the parameterization for the deformable model [103].

The general snakes formulation contains the capability for user-interaction. In addition to the placement of the initial curve points, some type of energy force could be defined which would incorporate user interaction, such as pulling the snake in some pre-specified direction. Several groups have studied the specific problem of making this interaction most intuitive and effective. The *live-wire* method by Barrett and Mortensen [10] is generally considered a competing method with snakes although the two ideas are closely related and could be considered modifications of one another. The live-wire process begins with the user choosing a *seed point* on the object boundary. The software then automatically grows this point along the boundary using a minimal cost algorithm; during the growth process, the user can accept various pieces of the generated boundary and start new seed points. Thus, the procedure segments pieces of the boundary at a time, each piece representing some minimal cost path between the endpoints. Another approach to incorporating user-interaction is called *tamed snakes* [52]. This approach represents the curve at multiple scales by performing hierarchical subdivision of the curve and allowing the user to influence the segments at the various scales. It also uses the model of masses and springs attached to control vertices to update the curve. Liang, McInerney, and Terzopoulos recently proposed a software package called *United Snakes* which combines several interactive snake models and emphasizes the generality of the snake model [68].

This chapter explains the fundamentals of the classical snake model since all subsequent active contour algorithms are based on the same principles. The "balloon" extension is then described because it is used in the methods described after it. We then explain in detail the topologically adaptive snakes because this is very similar to the geodesic contour model in terms of which limitations of classical snakes it addresses, which are numerical instability and topological invariance. Finally, we describe the geodesic active contour model itself, from which our CURVES algorithm proceeds. We provide a proof of equivalence to the classical snake model for those cases in which it is indeed equivalent and discuss related segmentation schemes also based on partial differential equations that have proceeded from geodesic snakes.

3.1 Classical Snakes

A classical snake is a spline curve which minimizes an energy functional. This functional is based on two terms, one that is related of the smoothness of the curve (*internal* energy) and one that is related to image information (*external* energy). For example, internal energy may be measured by one or more derivatives of the curve and external energy may be measured by the degree to which the curve is positioned at high image gradients, which are often presumed to indicate the true boundary of the object to be segmented. The model is quite broad, however, and many other internal and external energy forces are possible. The relative weighting of the two forces is a parameter of the method. The model is called *active* because the contours are updated iteratively to minimize the pre-defined energy functional; these contours are called *snakes* because they move smoothly (*slither*) over the image during this process.

3.1.1 Energy Functional

A snake is a planar curve, represented parametrically by $C(p) = (x(p), y(p))$, where p is the spatial parameter of the curve. We will assume the curve is closed, so $p \in [0, 1]$ and $C(0) = C(1)$. The energy functional to be minimized is

$$\int_0^1 E_{internal}(C(p)) + E_{external}(C(p))dp \tag{3.1}$$

where the internal energy is given by

$$E_{internal}(C(p)) = \alpha|C'(p)|^2 + \beta|C''(p)|^2$$

where C' and C'' are the first and second order derivatives of C with respect to p. In practice, β is often set to zero, so the first-order term is the only regularizing force. Also, α and β are usually constant over the curve but could vary along the curve if desired. The external energy expression is more variable than the internal energy expression and is defined to attract the snake to whatever image features indicate the presence of the object boundary. A common choice which assumes that large intensity gradients signify the object boundary is

$$E_{external}(C(p)) = -|\nabla I(C(p))|^2 \tag{3.2}$$

where I is the intensity image. The *potential* of the shape is defined to be the external energy,

$$P(C) = E_{external}(C). \tag{3.3}$$

3.1.2 Finding a Minimum

Euler-Lagrange equations provide an expression for the gradient of an integral functional and set this gradient to zero, thus providing conditions that are true for the function that extremizes the integral. Appendix A derives the general Euler-Lagrange equations and performs the computations used in this dissertation.

The Euler-Lagrange equations for Expression 3.1 are

$$-\frac{\partial}{\partial p}(\alpha C') + \frac{\partial^2}{\partial p^2}(\beta C'') + \nabla P(C) = 0 \tag{3.4}$$

when C is a local minimum, with boundary conditions either indicating that C is closed or specifying the positions and derivatives of the endpoints. Making C a function of time t as well as spatial position p and setting the expression to the negative time derivative of C for minimization implies the following evolution equation

$$C_t = \frac{\partial}{\partial p}(\alpha C') - \frac{\partial^2}{\partial p^2}(\beta C'') - \nabla P(C), \tag{3.5}$$

with specified boundary conditions.

The original presentation described an implementation using sparse matrix methods to implement Euler steps [54]. One first defines the force

$$F(C) = \nabla P(C).$$

Second, a pentadiagonal matrix is used to compute derivatives by finite differences because the computation requires two node points on either side of a given point. One can write

$$AC = -\frac{\partial}{\partial p}(\alpha C') + \frac{\partial^2}{\partial p^2}(\beta C'')$$

where A is the pentadiagonal matrix. C_t is estimated by $\frac{C^{(t+1)}-C^{(t)}}{\tau}$ where $C^{(t)}$ and C^{t+1} are the curve at times t and $t+1$ respectively and τ is the time step parameter.

One then rewrites Equation 3.5 as

$$\begin{aligned} C^{(t+1)} - C^{(t)} &= \tau\frac{\partial}{\partial p}(\alpha C') - \tau\frac{\partial^2}{\partial p^2}(\beta C'') - \tau F(C^{(t)}) \\ &= -\tau A C^{(t+1)} - \tau F(C^{(t)}) \end{aligned}$$

when we apply the regularization constraints (multiplication by A) to the future curve points $C^{(t+1)}$ instead of to the current curve $C^{(t)}$. Finally, one obtains

$$C^{(t+1)} = (I_d + \tau A)^{-1}(C^{(t)} - \tau F(C^{(t)}))$$

where I_d denotes the identity matrix.

An illustration of the evolution of a snake for the segmentation of a simple image is shown in Figure 3-1. The snake is initialized outside of the object. In the absence of image gradients, the internal energy causes it to shrink smoothly. When high gradients are encountered, the contour converges to segment the triangle.

As an alternative to the Euler-Lagrangian solution, dynamic programming approaches have also been used to implement the snake motion [5, 21]. Such methods can potentially come closer to finding global minima and achieve an increase in efficiency.

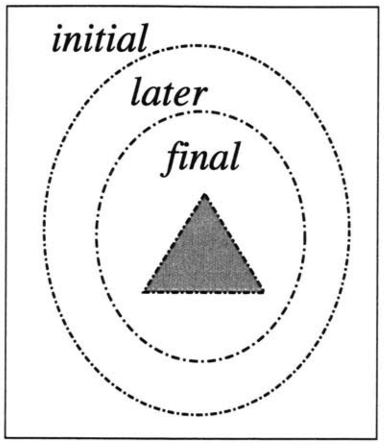


Figure 3-1: Segmentation illustration for image of triangle. The outermost ellipse is the initial contour. It is iteratively updated until it converges on the boundary of the triangle.

3.1.3 Limitations

Limitations of the classical snake formulation inspired several important extensions. Each of the following limitations is addressed by one or more of the three extensions described below.

1. **Shrinking:** Since the model is moving to increase smoothness, it will shrink to a point or line (depending on boundary conditions) in the absence of image information. This necessitates initializing the snake *outside* the object if it would need to move through any regions of zero image force, such as a region of constant intensity for the energy defined in Equation 3.2. This restriction is severe since there are some applications for which it is easier to initialize the contour *inside* the region to be segmented and to then have it evolve outward.

2. **Topology:** A second restriction of the model is the inability of the contour to change topology over the course of the evolution. For example, when segmenting an image of several nonoverlapping objects, it may be desirable to initialize one contour enclosing all of the objects, and have it evolve into multiple disjoint contours enclosing the separate objects. A second situation when topological flexibility is desired is when an object contains holes.

3. **Parametrization-Dependence:** The functional in Expression 3.1 depends on the parameterization which is unspecified. That is, the energy depends on the speed at which the curve is traversed, which is counterintuitive for the segmentation problem in which only the boundary matters, not the speed of traversal. [20] shows the following. Let $p = \phi(r)$, with $\phi : [c, d] \to [0, 1]$ and

$\phi' > 0$, be a new parameterization of the curve $C(p)$. We then observe that

$$\int_0^1 |C'(p)|^2 dp = \tag{3.6}$$

$$\int_0^1 |C'(p)|^2 (\frac{d\phi}{dr})^2 (\frac{d\phi}{dr})^{-2} dp = \tag{3.7}$$

$$\int_c^d |C'(p)|^2 (\frac{d\phi}{dr})^2 (\frac{d\phi}{dr})^{-1} dr = \tag{3.8}$$

$$\int_c^d |C'(p)\phi_r(r)|^2 (\frac{d\phi}{dr})^{-1} dr = \tag{3.9}$$

$$\int_c^d |(C \circ \phi)_r(r)|^2 (\phi_r(r))^{-1} dr \tag{3.10}$$

where subscripts indicate differentiation by the subscripted variable. This equation means that the energy is weighted by $(\phi_r(r))^{-1}$ at each point, and can thus change in an arbitrary way depending on the parameterization. Likewise, the external energy can also change arbitrarily since it is weighted by $\phi_r(r)$:

$$-\int_0^1 |\nabla I(C(p))|^2 dp = -\int_c^d |\nabla I((C \circ \phi)(r))|^2 \phi_r(r) dr. \tag{3.11}$$

This arbitrariness to the energy is problematic for segmentation algorithms which should be independent of parametrization.

A second negative effect of parameterization dependence is the inability to evolve into thin finger-like structures since the initial parameterization did not contain enough points in that region of the curve to represent fine detail.

4. **Numerical Instability:** Another difficulty is numerical instability. Node points move based on a fixed stepsize and image information, so there is no constraint to stop them from moving too close to each other which could cause numerical instability in derivative estimates as spacing between points on the boundary becomes potentially widely variable as the boundary is evolve forward in time.

5. **Stepsize:** The final limitation we mention is that there is no constraint on the amount of motion a node made undergo from one iteration to the next. That is, a point can move too far and thus move past a local minima, causing the snake to miss the desired segmentation. In general a fixed stepsize is a parameter of the algorithm, but when this fixed value is scaled by large image gradients, the resulting step may be larger than desirable.

The following three sections discuss extensions to the classical snakes model. Each addresses some subset of the limitations above. As the methods are described, their solutions to those limitations are emphasized.

3.2 Balloons

The balloon model made two changes to the force F, which became

$$F(C) = k_1\vec{N} - k\frac{\nabla P(C)}{|\nabla P(C)|} \tag{3.12}$$

where $\vec{N}$ is the unit normal to the curve, and k_1, k are constants of the same order, controlling the relative weighting between the two terms [26].

Ignoring the first term momentarily, we notice that the second term is the original force F modified to disregard the magnitude of the image-based information. k should be set so that each curve point can move at most one pixel per iteration. This modification avoids stepsize difficulties that could cause the snake to move too much, skipping over a local minimum.

The first term is the more significant change to the model. It is an inflationary force (when $k_1 > 0$) enabling the snake to be initialized inside the object and to inflate to the object boundary, hence the name *balloon.* k should be slightly larger than k so an image force can overcome the inflation force. The sign of k_1 controls whether an inflationary or deflationary force is used, and in either case, the position of the initial contour can be farther from the target position and still converge correctly. An inflationary force was also included in the framework described in [106], which included many degrees of freedom and could give rise to many different equations of motion depending on the forces chosen for a particular application. The balloon model thus addresses the problems of *stepsize* and *shrinking.* In accordance with the its name, this model is generally known for its solution to the *shrinking* limitation, and the term "balloon" is usually used with reference only to the first term of Equation 3.12.

3.3 T-Snakes

A different change to the snakes model is given by *topologically adaptive snakes*, or *T-snakes* [80, 83]. This approach addresses the problems of *topology*, *parameterization-dependence* and *numerical instability.* The primary approach is to periodically reparameterize the curve by computing its intersections with a regular grid. In this way, the curve can evolve to fit far more complicated shapes, such as thin-tube-like structures, than it could under the original model. The method is also more numerically stable since the reparameterization implies that node points do not become more than a small fixed distance apart, and are approximately equidistant along the curve. The model also enables the contour to undergo topological changes such as splitting and merging over the course of the evolution.

3.3.1 Equations

A T-snake is a discrete approximation to a curve. It is a set of ordered connected nodes, where the following quantities are associated with each node. The position at time t is given by $\mathbf{x}_i(t) = [x_i(t), y_i(t)]$, stretching and bending forces are given by A_i and B_i respectively, and inflation and image forces are given by ρ_i and f_i respectively. For each node i of the snake, the equation governing its motion is

$$\gamma\dot{\mathbf{x}}_i + A_i + B_i = \rho_i + f_i \tag{3.13}$$

where $\dot{x}_i$ is the velocity of node i and γ is a parameter of the method.

The stretching term A_i attempts to keep the points equally spaced and is given by

$$A_i = a(\mathbf{x}_i - \frac{1}{2}(\mathbf{x}_{i-1} + \mathbf{x}_{i+1})).$$

The bending term B_i inhibits bending of the contour and is given by

$$B_i = b((\mathbf{x}_{i+1} - \frac{1}{2}(\mathbf{x}_i + \mathbf{x}_{i+2})) - \frac{1}{2}(\mathbf{x}_i - \frac{1}{2}(\mathbf{x}_{i+1} + \mathbf{x}_{i-1})) + (\mathbf{x}_{i-1} - \frac{1}{2}(\mathbf{x}_i + \mathbf{x}_{i-2}))).$$

ρ_i is similar to the balloon force above

$$\rho_i = q\mathbf{n}_i \tag{3.14}$$

where $\mathbf{n}_i$ is the unit normal to the curve at node i. Finally, f_i is the original image-related force

$$f_i = k\nabla P(\mathbf{x}_i) \tag{3.15}$$

where P is as defined in equation 3.3 and a, b, k are weighting constants. q has constant magnitude, but its sign depends on image information so that the contour can expand to encompass the object but can contract if leaking into the background. For notational simplicity in this discussion, we regard it as a constant.

Using a discrete approximation to derivatives, the node positions are updated according to equation 3.13 so that

$$\mathbf{x}_i^{(t+\Delta t)} = \mathbf{x}_i^{(t)} - \frac{\Delta t}{\gamma}(A_i + B_i - \rho_i - f_i) \tag{3.16}$$

3.3.2 Reparameterization

The above equations are slightly different from the original snakes formulation [54], but are not the primary difference defining the method of *T-snakes.* The primary extension is that the parametrization of the curve changes over the course of the

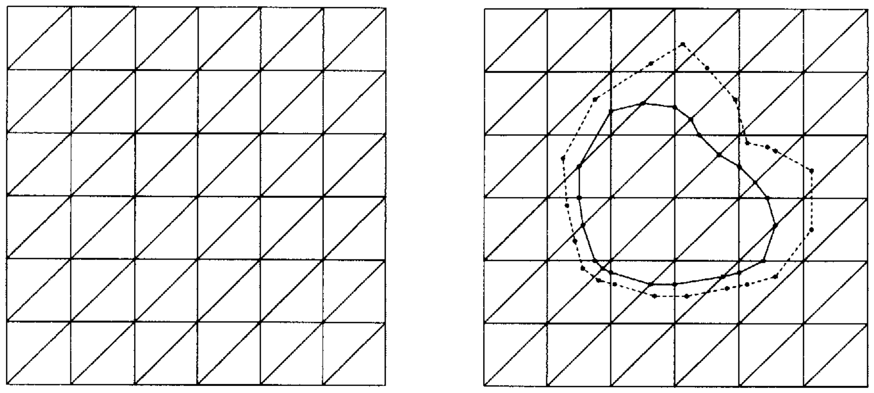


Figure 3-2: (a) Simplicial grid decomposition of $\mathbb{R}^2$. (b) Reparametrization of evolving contour. Figure reproduced from [83].

evolution. This prevents neighboring nodes from becoming too far apart while still permitting them to deform into unusual shapes, such as long thin shapes. Moreover, the particular method of reparametrization enables the topological changes which resolve that limitation of the original method.

A fixed grid is overlaid on the image, then a new parametrization is computed from a previous parametrization by computing the intersection points of contour with the grid edges. Those intersection points become the new nodes defining the new parametrization. Although both versions exist, a simplicial cell decomposition is usually used to partition space instead of the usual nonsimplicial Cartesian grid. In a *simplicial cell decomposition*, the cells are open *simplices*, where an n-D *simplex* is the generalization of a tetrahedral region of space to n-D. An equivalent definition is that an n-D simplex is the simplest geometrical object of dimension n. Hence, a 2D simplex is a triangle. A simplicial cell decomposition is also called a *triangulation*. The benefit of a simplicial decomposition over a nonsimplicial decomposition is that only the vertices of the *simplicial* decomposition are needed to unambiguously represent an implicitly defined object boundary, without special machinery for disambiguation.

This reparametrization is illustrated in Figure 3-2, which is reproduced from [83]. The first figure shows a simplicial decomposition of $\mathbb{R}^2$. The second figure illustrates the use of this decomposition in reparametrizing the curve. The initial contour is indicated by the solid line, defined by a set of node points which lie on the edges of the simplicial decomposition. Each node point is then moved according to the t-snakes update equation, and the points no longer lie on the edges of the grid. The curve is reparametrized so that the intersections of the new contour (dotted line) with the grid become the node points for the next iteration.

3.3.3 Topological Changes

Lists of all interior vertices of the grid are maintained as the contour moves over the grid. The contour is thus implicitly represented by the information about which points are inside the contour and which are outside the contour. This representation is similar to the level set representation described in section 2.2 in that interior and exterior vertices correspond, respectively, to negative and positive values of the distance function in the level set framework. Thus, it can also handle topological changes automatically. We will see the full level set method used in another extension to the classical snakes model next.

3.4 Geodesic Snakes

The geodesic snakes are another extension to the basic active contour model. They are based on concepts from the differential geometry, partial differential equations, and fluid dynamics communities. In addition to a different form of the energy function, they provide a novel implicit boundary representation which avoids some of the limitations of classical snakes. In particular, this representation provides an elegant solution to the problems of *topology*, *parameterization-dependence*, and *numerical instability.*

A *geodesic* curve is a minimal distance curve: the shortest path between any two points on the curve is exactly the portion of the curve connecting those two points. The idea is to formulate the minimization problem 3.1 as the minimization of the length of the curve, where "length" is computed according to some non-Euclidean image-related metric. Recall from section 2.1 that to minimize Euclidean arclength

$$\int |C'(p)|dp$$

one evolves the curve according to

$$\vec{C}_t = \kappa \vec{N}. \tag{3.17}$$

One would like to define a similar minimization so that the corresponding evolution equation achieves image segmentation [20, 17, 56, 78, 79].

3.4.1 Curve Evolution Equation

The task of finding the curve that best fits the boundary of the object is posed as a minimization problem over all closed planar curves $C(p) : [0, 1] \to \mathbb{R}^2$. The objective function is

$$\int_0^1 g(|\nabla I(C(p))|)|C'(p)|dp$$

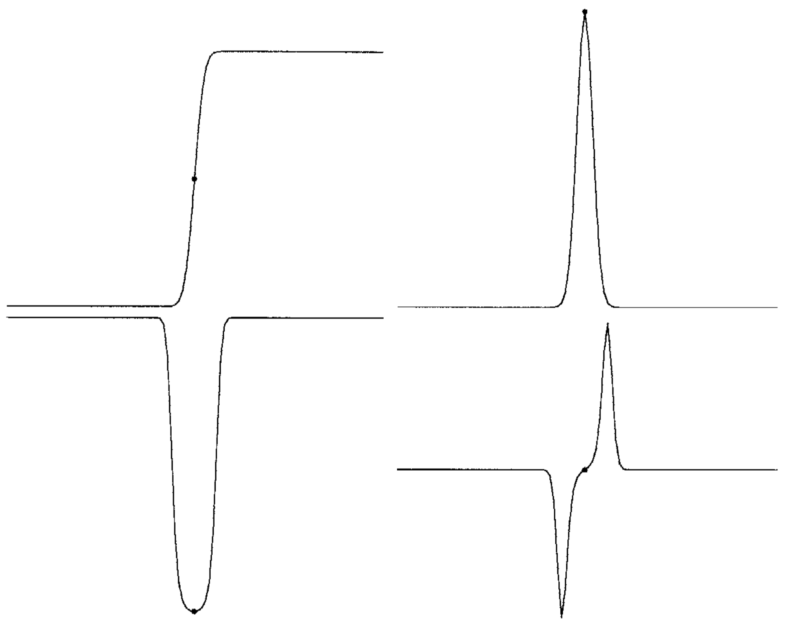


Figure 3-3: The evolution equation near an edge in a 1D image. (a) The intensity profile of the 1D image $\hat{I}$. (b) Its gradient $\nabla \hat{I}$. (c) The weighting function $g = \frac{1}{1+|\nabla \hat{I}|}$. (d) Its gradient ∇g.

where $I : [0,a] \times [0,b] \to [0,\infty)$ is the image and $g : [0,\infty) \to \mathbb{R}^+$ is a strictly decreasing function such that $g(r) \to 0$ as $r \to \infty$ [20, 17, 56]. That is, we are looking for the minimal distance curve where distance is weighted by the function g which acts on intensity gradients, so that the curve is attracted to intensity edges in the image. For example, $g(|\nabla I|) = \frac{1}{1+|\nabla I|^2}$ is a common choice because it can be computed easily.

To minimize this objective function by steepest descent, consider C to be a function of time t as well as spatial parameter p. Then one can compute the Euler-Lagrange equation of the objective function to determine the evolution of the curve, i.e., its derivative with respect to time. This yields the curve evolution equation

$$\vec{C}_t = g(|\nabla I|)\kappa\vec{N} - (\nabla g(|\nabla I|)\cdot\vec{N})\vec{N} \tag{3.18}$$

where κ is the Euclidean curvature and $\vec{N}$ is the unit inward normal. Note that we have obtained an equation similar to the original mean curvature flow equation (Equation 2.2). Both equations find geodesic curves (curves of minimal length), but the original equation uses the Euclidean metric and the new equation uses a weighted distance.

We call the attention of the reader to the two terms present in Equation 3.18. Previous evolution schemes have used only the first term [18]. That is, the curve-shortening flow is weighted by some function of the image gradient to attract the curve to regions of interest in the image. In the geodesic snakes curve evolution equation, however, the second term which follows from the use of a non-Euclidean distance metric gives improved behavior. This improvement lies in the fact that $-\nabla g$ gives the direction toward the minimizer of g. Thus, the addition of the term, which is zero at the minimizer of g, provides additional influence on the curve in the correct direction. Figure 3-3 illustrates this situation. The leftmost graph shows the intensity profile of a test 1D image $\hat{I}$. This profile was generated by convolving an ideal step edge with a Gaussian filter. The next graph shows its gradient, $\nabla\hat{I}$, whose maximum occurs at the sharpest point on the profile of $\hat{I}$. Third is the g function, with $g = \frac{1}{1+|\nabla\hat{I}|}$; of course its minimum is attained at the point which maximizes $\nabla\hat{I}$. Note that in an ideal situation, this minimum would be equal to zero, so the first term in Equation 3.18 would be zero. The final graph, ∇g, shows that the second term would also be zero at this extremal point, thus causing the evolution to converge. Also observe that near this point, $-\nabla g$ gives the direction in which the boundary estimate should move to extremize g.

3.4.2 Equivalence to Classical Snakes

The geodesic snake formulation is equivalent to the classical snakes formulation with a couple parameters fixed [20]. First, choose $E_{external}(C) = \lambda g(|\nabla I(C)|)^2$ where g is constrained as above and fix $\beta = 0$ to obtain the energy expression

$$E(C) = \alpha \int_0^1 |C'(p)|^2 dp + \lambda \int_0^1 g(|\nabla I(C(p))|)^2 dp \tag{3.19}$$

where α and λ are free parameters that control the trade-off between smoothness and edge proximity. That is, we now have

$$E_{internal} = \alpha |C'(p)|^2$$

and

$$E_{external} = \lambda g(|\nabla I(C(p))|)^2$$

Maupertuis' Principle states conditions sufficient for an energy minimization to be equivalent to finding a geodesic curve in some metric space. This principle is used to obtain the length-minimization (geodesic) expression that is equivalent to the energy equation 3.19.

Before stating the principle, we define the terms from physics that are needed [37]. Specifically, we use the Lagrangian and Hamiltonian formalism of mechanics. This means that we need a *Lagrangian* $\mathcal{L}(p, C, C')$ which is the sum of a kinetic energy term and a potential energy term. For our purposes, we call the internal energy the "kinetic energy" and the external energy the "potential energy". Note that these choices are only for the Lagrangian-based derivation of the equivalence; they do not reflect dynamic or mechanical properties or relations of the snakes' energy terms. So, the Lagrangian is

$$\mathcal{L} = E_{internal} + E_{external}$$

Once a Lagrangian has been defined, the *Hamiltonian H* is given by

$$H = \frac{\partial \mathcal{L}}{\partial C'} C' - \mathcal{L}.$$

This formalism is used to write down equations for a variety of types of motion in the mechanics field, and we now use it to relate geodesic snakes back to classical snakes. We next compute it for our Lagrangian $\mathcal{L}$. We have

$$\frac{\partial \mathcal{L}}{\partial C'} = 2\alpha |C'| \frac{C'}{|C'|} = 2\alpha C',$$

so

$$H = 2\alpha C' \cdot C' - \alpha |C'|^2 - \lambda g^2$$

which reduces to

$$\begin{aligned} H &= \alpha |C'|^2 - \lambda g(|I(C)|)^2 && (3.20) \\ &= E_{internal} - E_{external} && (3.21) \\ & && (3.22) \end{aligned}$$

Note the dual nature of the Hamiltonian and the Lagrangian: the Hamiltonian is the difference of the summands of the Lagrangian. This relationship holds for a large class of motion equations.

For similarity to the literature and for simplicity of the theorem below, we denote $\frac{\partial \mathcal{L}}{\partial C'}$ by q, so $q = 2\alpha C'$. To avoid carrying around the "2", we make a change of variables: $m = 2\alpha$ and $q = mC'$. That is, our Hamiltonian is written as

$$H = \frac{q^2}{2m} - \lambda g(|I(C)|)^2.$$

We are now ready to state the theorem on which the equivalence relies.

Theorem 3.1 *(Maupertuis' Principle) Curves $C(p)$ in Euclidean space which*

1. *are extremal corresponding to the Hamiltonian $H = \frac{q^2}{2m} + U(C)$ and*
2. *which have a fixed energy level E_0*

are geodesics with respect to a new metric. Specifically, they minimize

$$\int_0^1 \sqrt{2m(E_0 - U(C))}|C'(p)|dp$$

For our Hamiltonian, we have

$$U(C) = -\lambda g(|I(C)|)^2.$$

The fixed energy level E_0 refers to a fixed value of the Hamiltonian. By Equation 3.20 we saw that this is the difference between the internal energy and the external energy, so

$$E_0 = E_{internal} - E_{external}.$$

Next, we need to specify the fixed energy level E_0 required to apply Maupertuis' Principle. This parameter incorporates both the parameterization of C (which until now was arbitrary) as well as α and λ, since $E_0 = E_{internal} - E_{external}$. In the absence of other information, and for simplicity, the method chooses $E_0 = 0$ which means that the internal and external energy components are equal; that is, they make equal contributions to the total energy level.

We have now obtained that minimizing the energy equation 3.19 is equivalent to finding the (geodesic) curve which minimizes

$$\int_0^1 \sqrt{2m\lambda g(|\nabla I(C(p))|)^2}|C'(p)|dp$$

which is equivalent to minimizing

$$\int_0^1 g(|\nabla I(C(p))|)|C'(p)|dp$$

since m and λ are constants.

Thus, the geodesic snake formulation is equivalent to the classical snake formulation with the restrictions of $\beta = 0$ and of equal contributions from internal and external energy components. This discussion follows [20].

3.4.3 Level Set Implementation

A level set scheme as described in section 2.2 is used in order to make this flow intrinsic to the curve. Define $u : [0, a] \times [0, b] \rightarrow \mathbb{R}$ to give the signed distance from any image point to the curve $C(\cdot)$. Then u is evolved instead of C, which is identically the zero level-set of u, according to the method described in section 2.2. Choosing $\beta = g\kappa - (\nabla g \cdot \vec{N})$ as in Equation 3.18 and noting that $\kappa = \mathbf{div}(\frac{\nabla u}{|\nabla u|})$ gives

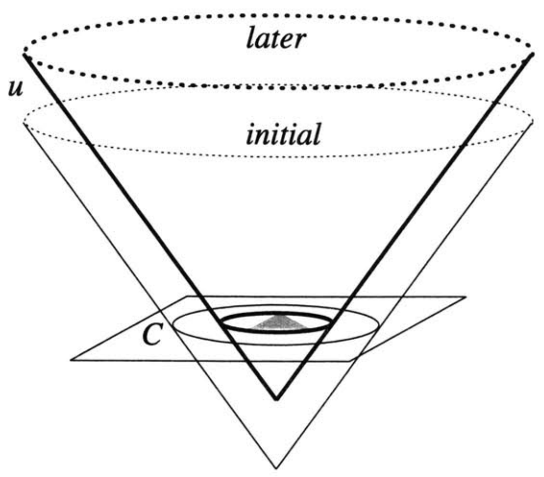


Figure 3-4: Level set implementation of curve evolution for segmentation of triangle image.

the evolution equation

$$u_t = g|\nabla u|\mathbf{div}(\frac{\nabla u}{|\nabla u|}) + \nabla g \cdot \nabla u. \tag{3.23}$$

This behavior is illustrated in Figure 3-4. An additional term c can be used either to increase the speed of the flow or to force the contour to flow outward, similar to the balloon force in [26] described above, to yield

$$u_t = g|\nabla u|(c + \mathbf{div}(\frac{\nabla u}{|\nabla u|})) + \nabla g \cdot \nabla u.$$

Notice that we again have a second term $\nabla g \cdot \nabla u$ not seen in previous surface evolution schemes. This term causes the same desirable behavior as illustrated in Figure 3-3 and described above in the context of planar curves.

We summarize the approach of the geodesic snakes method and its extensions: The evolutions are related to curve-shortening flow. Instead of Euclidean distance, however, a metric is used which is based on image information. In this way the geodesic curve is one that optimizes some salient image features. The curve evolution equation follows directly from the objective function, assuming gradient descent. Finally, the computation is lifted one dimension so that some embedding function u is evolved instead of the curve, which can be retrieved at any time by extracting a level-set of u.

3.4.4 Minimal Surfaces

The extension to surfaces in three dimensions is straightforward and is called *minimal surfaces* [18]. Consider the case of a surface S evolving in $\mathbb{R}^3$. This is important for image segmentation applications in which the image data is volumetric, such as segmenting the brain in magnetic resonance imagery.

In the absence of external image information, minimizing area A where

$$A = \int\int da$$

is achieved by mean curvature motion

$$\frac{\partial S}{\partial t} = \mathbf{H}\vec{N}$$

where da is an area element, $\mathbf{H}$ is the mean curvature of the surface, and $\vec{N}$ is its inward unit normal. The incorporation of an image term gives a weighted area

$$A_w = \int\int g(I)da$$

where $I : [0,a] \times [0,b] \times [0,c] \rightarrow [0,\infty)$ is the 3D image and $g : [0,\infty) \rightarrow \mathbb{R}^+$ is a strictly decreasing function such that $g(r) \rightarrow 0$ as $r \rightarrow \infty$. Computation of the Euler-Lagrange equations yields the surface evolution equation

$$\frac{\partial S}{\partial t} = (g\mathbf{H} - \nabla g \cdot \vec{N})\vec{N}.$$

For a level set implementation, let $v : \mathbb{R}^3 \rightarrow \mathbb{R}$ be the signed distance function to the evolving surface S with negative distances inside S and positive distances outside S. The level set update equation is then

$$v_t = g(I)|\nabla v|\mathbf{div}(\frac{\nabla v}{|\nabla v|}) + \nabla g \cdot \nabla v.$$

For the same purposes as in the lower dimensional case, one can again incorporate an additional term c to obtain

$$v_t = g(I)|\nabla v|(c + \mathbf{div}(\frac{\nabla v}{|\nabla v|})) + \nabla g \cdot \nabla v.$$

3.4.5 Other Partial Differential Equations for Segmentation

These segmentation schemes belong to the superset of work using partial differential equations for a variety of image analysis tasks including image enhancement and restoration [59, 58], motion segmentation [59], stereo reconstruction [40], optical flow [63, 6], and others. This general approach is applicable to any image task which is local in nature, by the local nature of the differential equations. An overview of the use of partial differential equations in image processing and analysis is provided in [19].

3.4.6 Extensions

There have been many extensions to the geodesic snake model. Sapiro has provided the natural extension to the case of vector-valued image data, such as in the case of color images, texture images (in which one wishes to consider some original image's responses to a set of texture filters as separate "response images", and multi-modal medical images [96]. Yezzi, Tsai, and Willsky [112] have viewed the image segmentation problem from a statistical standpoint, using both global and local properties to partition the image into a predetermined number of classes. In particular, they define energy functionals based on intensity mean, variance, and texture, and still use a level set implementation. Paragios and Deriche have extended the geodesic snakes segmentation strategy to detection and tracking of moving objects [89] and also to the segmentation of static images according to texture information [90]. Gomes and Faugeras have provided an extension in which the distance function is forced to remain a distance function throughout the evolution [45]. The importance of this property of the evolution will be discussed in further detail in relation to our CURVES system in section 5.3. Another fundamental change to the framework has been the incorporation of an area-minimization force, in addition to the usual length-minimization force [101].

One difficulty with the local nature of the active contours framework that there is no mechanism to incorporate global shape information. Several researchers have addressed this problem: Leventon *et al.* has incorporated learned shape and intensity priors into the level set segmentation framework [65, 64] and Guo and Vemuri have incorporated generator functions which represent the object shape with a small number of parameters [49].

3.5 Discussion

Note that both the t-snakes and the geodesic snakes address the same limitations in the original snake model: *topology*, *parametrization-dependence*, and *numerical instability*. A potential benefit of the t-snakes approach over the geodesic snakes approach is a more natural capability for user-interaction in the t-snakes approach, in which the explicit representation of the boundary makes interactive control or influence straightforward. One potential problem with the level set representation used in geodesic snakes is that because there is only an implicit representation of the curve, it is difficult to interact with it. Cohen and Kimmel, however, have provided a method for interacting with the evolving level set representation in which the user can fix the endpoints of the curve [25].

In contrast, an advantage of the geodesic snakes over the t-snakes is that the representation is more elegant and does not require the repeated reparametrization and computing of the new node points. Further, the fact that no specific parametrization is used in the geodesic snakes method enables proofs and understanding about the behavior of the evolution which are not possible when the energy function is dependent on the changing parametrization; this problem is presumably lessened in the t-snakes case since although the parametrization changes, the distance between points is bounded.

It is also important to note that the t-snakes' representation of lists of interior and exterior points is very similar to the level set representation which distinguishes these two groups based on the sign of the distance function. That is, although the underlying ideas and the directions from which the problems were approached are different, the resulting representations are very similar. In both cases, they are used to calculate the true boundary at some step (either throughout the evolution in the t-snakes case or when the boundary is requested for visualization in the geodesic snakes case) and provide the basis for the topological flexibility.

All of the deformable models discussed, by construction, are vulnerable to local minima. This is a result of the iterative evolution, which only considers local information. This aspect of the methods makes it difficult to understand their convergence properties, such as how many iterations are required for convergence, if convergence is guaranteed at all, for what class of images is convergence possible, and whether or not the contour at convergence indeed corresponds to the true object boundary. Some work has incorporated a statistical framework [112], but this remains an area for future work.

Chapter 4

MRA Segmentation

To study deformable models for image segmentation and also the temporal evolution of 3D curves, we have chosen the application of segmenting tubular structures in volumetric medical imagery. We have addressed the specific task of segmenting blood vessels in MRA data with a focus on segmenting the thinnest vessels because these are most difficult and because the fine spatial detail they provide is important in surgical planning. This chapter provides a description of this type of image data and of previous approaches to segmenting it.

4.1 MRA Data

The two standard types of MR angiography imaging are "Time of Flight" (TOF) and "Phase Contrast" (PC-MRA). Both obtain volumetric images in which bloods vessels appear bright against a dark background. We have dealt primarily with PC-MRA images, so it is of that technique that we will provide an overview. Many of the same segmentation techniques can be applied to both types of images, since the appearance is very similar. When reviewing previous approaches, we will not distinguish between the two. Computed Tomography Angiography (CTA) is a different imaging modality also used to visualize blood vessels. These images have a very different appearance, although some similar segmentation approaches may be possible.

PC-MRA measures the velocity of blood flow at each spatial location in the imaged volume. One thus obtains bright voxels in locations of blood vessels and dark voxels elsewhere. Moreover, the brightness of the imaged vessels is greater for vessels carrying faster blood flow than for those carrying slower motion. The velocity at each spatial location is obtained for the x, y, and z directions. Thus, one obtains a 3-vector at each point indicating this velocity. For each dataset, a *venc* or *velocity encoding* parameter must be set. This is the maximum velocity that can be correctly measured. For, example, if a venc of 60 cm/second is selected, speeds between -60 cm/second and 60 cm/second will be reported correctly. The valid velocity range is mapped to a circle and the measurement can be considered as an angle. Zero degrees

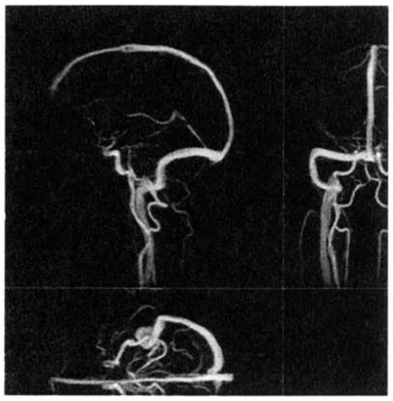


Figure 4-1: Maximum intensity projection of a phase-contrast MRA image of blood vessels in the brain.

is mapped to zero cm/second, angles approaching 180 degrees are mapped to values approaching 60 cm/second, and angles approaching -180 degrees are mapped to values approaching -60 cm/second. Speeds whose magnitude is greater than the venc, however, will "wrap" around the circle. For our example, a speed of 70 cm/second will wrap so it appears as a speed of -50 cm/second. Choosing a lower venc, such as 30 cm/second, provides better visualization of the thin vessels in which blood flows slowly since a larger portion of the dynamic range of the image is devoted to them. The expense is that the fast blood flow in the centers of large vessels will be measured incorrectly due to wrapping.

Although one obtains these three volumes for the x, y, and z velocity components, one generally uses only what is called the *magnitude image.* Also a 3D volume, the magnitude image is obtained by computing the magnitude of the (x,y,z) vectors at each voxel and then usually scaling that magnitude by an anatomical magnetic resonance image. That scaling has the effect of removing noisy vectors that may appear outside the anatomical structure, in regions that would be black (zero intensity) in the anatomical MR image. Other scaling terms can also be used. In general, when we say "MRA image," we refer to a magnitude image which consists of a scalar intensity value at each point with high intensities indicating blood flow.

An advantage of magnetic resonance vascular imaging over x-ray vascular imaging such as CTA is that MR techniques often do not require the injection of any contrast agent into the bloodstream of the patient. If a contrast agent is used, the amount injected is often small.

One such PC-MRA image is pictured in Figure 4-1. Again, blood vessels appear as bright curve-like patterns which may be noisy and have gaps. What is shown is a "maximum intensity projection". The stack of slices is collapsed into a single image for viewing by performing a projection through the stack that assigns to each pixel in the projection the brightest voxel over all slices. This image shows projections along three orthogonal axes. The typical size of cerebral datasets we saw was 60 slices, each of dimensions 256x256 voxels. For these datasets, each voxel had dimensions $.9375 \times .9375 \times 1.5$ cm^3. These datasets are considered quite high resolution, and many other datasets have larger voxel sizes.

4.2 Thresholding

Thresholding is one possible segmentation approach in which voxels are labeled as vessels if their intensity is above some threshold and as background otherwise. This approach works adequately on the larger vessels. The problem arises in detecting the small vessels, and that is the objective of our work. Thresholding cannot be used for the small vessels for several reasons. The voxels may have an intensity that is a combination of the intensities of vessels and background if the vessel is only partially inside the voxel. This sampling artifact is called *partial voluming.* Other imaging conditions can cause some background areas to be as bright as other vessel areas, complicating threshold selection. Finally, the images are often noisy, and methods using local contextual information can be more robust.

A statistical approach has been proposed which retains the basic thresholding approach but employs a fully automatic method for choosing thresholds [109, 24]. This method assumes Gaussian [109] or Rician [24] intensity distributions for background and for vessel intensities, then uses the expectation maximization (EM) algorithm to find appropriate thresholds that will be used to classify the voxels.

4.3 Multiscale Filtering, Local Structure

Multiscale filtering has been proposed for the segmentation of curvilinear structures in 3D medical images such as vasculature in MRA images [97, 62, 42, 72]. The primary application addressed is the segmentation of vasculature in MRA images. This method involves convolving the image with Gaussian filters at multiple scales and analyzing the eigenvalues of the Hessian matrix at each voxel in the image to determine the local shape of the structures in the image. Multiscale analysis has a wide range of applications, and the method of normalizing the outputs of the filters of the various scales in order to determine the best response over the scales is studied in the relatively new field of scale-space theory [69].

We now discuss this theory for the specific application of detecting structures which have local shape properties of interest, such as cylindrical blood vessels; this discussion follows [97]. The Hessian can be used for line or cylinder detection and enhancement according to the following observations. Let $I : [0,a] \times [0,b] \times [0,c] \rightarrow [0,\infty)$ be the volumetric image (e.g., an MRA image). The Hessian matrix of I at each point $\bar{x} = (x, y, z)$ is given by

$$\nabla^2 I(\bar{x}) = \begin{bmatrix} I_{xx}(\bar{x}) & I_{xy}(\bar{x}) & I_{xz}(\bar{x}) \\ I_{yx}(\bar{x}) & I_{yy}(\bar{x}) & I_{yz}(\bar{x}) \\ I_{zx}(\bar{x}) & I_{zy}(\bar{x}) & I_{zz}(\bar{x}) \end{bmatrix}$$

Instead of using the image directly, one first convolves it with a Gaussian filter $G(\bar{x}; \sigma_f)$ of sigma σ_f to obtain smoother gradients which improve the estimation of local structure in the presence of image noise or non-smooth edges. The Hessian of this filtered image is equivalent to convolving the original image with the second derivatives of the Gaussian filter, for example

$$I_{xx}(\bar{x}; \sigma_f) = \{\frac{\partial^2}{\partial x^2} G(\bar{x}; \sigma_f)\} * I(\bar{x}).$$

To understand how the eigenvalues of this matrix express the structures in I, consider an image L containing a bright line parallel to the z axis with Gaussian cross-sectional intensity profile of width σ_r. That is,

$$L(\bar{x}; \sigma_r) = \exp(-\frac{(x - x_0)^2 + (y - y_0)^2}{2{\sigma_r}^2})$$

where (x_0, y_0) is the center of the line. Let $\nabla^2 L(\bar{x}; \sigma_r, \sigma_f)$ be the corresponding Hessian matrix combined with the Gaussian convolution, and let $\lambda_1(\bar{x}), \lambda_2(\bar{x}), \lambda_3(\bar{x})$ be its eigenvalues with $\lambda_1(\bar{x}) > \lambda_2(\bar{x}) > \lambda_3(\bar{x})$. Observe that both λ_2 and λ_3 have the same minimum at $x = x_0$, $y = y_0$, and $\sigma_r = \sigma_f$, assuming σ_f is fixed. They have the same minimum there because they correspond to two eigenvectors that lie in the plane perpendicular to the line, whose relative strengths are equal because the line is radially symmetric; they would differ for a ribbon-like model that lacked this symmetry. Further, at this location, λ_1 is zero because it corresponds to the curvature of the line, which is zero. It follows that the conditions indicating a bright line are

$$\lambda_1 \approx 0$$

and

$$\lambda_2 \approx \lambda_3 \ll 0.$$

Further analysis of the eigenvalues [62] has provided more specific information about structure within the image. Such analysis can be performed for other structures as well, such as spheres and thin sheets.

Various vessel segmentation algorithms have been proposed following such eigenvalue analyses. Some methods use the output of the multiscale filter directly to define a new image in which curvilinear structures are brightened and bright voxels corresponding to speckle noise and planar structures such as skin are darkened [97, 42, 72]. This enhanced image is visualized directly [42], thresholded [97], or segmented using an active contour method [72]. Other methods use the eigenvalues so obtained to define a candidate set of voxels which could correspond to the centerlines of vessels [62, 61]. Multiscale response functions are evaluated at each of these voxels to determine the likelihood that the voxel is a vessel of various diameters. The maximal response over all choices of diameters (scales) is retained at each voxel, and a surface model of the entire vascular structure is reconstructed from knowledge of centerlines and diameters. A final method which obtains segmentations by thresholding a filtered MRA image uses anisotropic diffusion to remove noise without removing small vessels [60, 91].

A related approach for segmentation and enhancement is the estimation of local structure by tensors that are created from the outputs of quadrature filters or other local filters [108]. These filters are sensitive to signals at various orientations, and the tensors indicate the degree to which each location is planar, line-like, and sphere-like.

4.4 Intensity Ridges, Centerlines

A different multiscale approach based on medial axes uses the fact that the centerlines of the vessels appear brightest to detect these centerlines as intensity ridges of the image [7]. The width of a vessel is then determined by a multiscale response function. This algorithm has been used in conjunction with 2D/3D registration to incorporate information from a pair of x-ray angiograms [16, 15]. Another algorithm that extracts centerlines explicitly uses differential geometry and treats the 3D MRA image as a hypersurface of 4D space in which extrema of curvature correspond to vessel centerlines [92].

4.5 Deformable Models

Instead of looking for centerlines, an alternate approach to segmentation is to look for the boundary directly. Deformable models, as described in Chapter 3, have been applied to 3D vascular segmentation. In these approaches, an initial boundary estimate is deformed iteratively to optimize an energy function which depends both on image information and on the smoothness of the surface. Two such approaches are *minimal surfaces* [18] and *t-surfaces* [83]. Our work follows closely from the minimal surfaces approach and we compare results obtained with our algorithm to those obtained by a variant of the minimal surfaces approach in section 6.5.

Frangi *et al.* have recently used their multiscale filtering method [42] within the framework of a deformable model approach to yield a complete segmentation system [41]. They use two deformable models: the first is a B-spline curve which represents the central axis of the vessels, the second is a B-spline surface which represents the vessel walls. An overview of the use of deformable models in medical image analysis is found in [81].

4.6 Morphology

A recent vessel segmentation algorithm was proposed that is based on dilation or erosion of a 3d volume [36]. It was presented with application to segmentation of vessels in a 3D tomography scan of a liver. The method is based on the concept of *simple points* which are defined as a point in an object (assume 3D object for this case) whose deletion does not change the topology of the object. Two dual methods are proposed. In the first, the object is initialized to consist of only a single point known to be inside the true object, then that object is dilated successively by the addition of adjacent simple points. In the second, the background is segmented with the same method, and then the object is defined as the complement of the background. Voxel intensities give priorities on points so that they are added in order of decreasing (increasing) intensity; this ordering is crucial to the algorithm. In both cases, an intensity threshold is uses to stop the growing so that no points beneath (above) that threshold are added. This algorithm relies on the assumption that the structure to be segmented has a known topology, and can be regarded as a thresholding based algorithm that maintains the topological constraint. The calculation of simple points has also been applied to 3D skeletonization [11], which we will discuss in section 7.

Chapter 5

CURVES

CURVES is our technique for segmentation of MRA images based on 3D curve evolution. The name *CURVES* is derived from the description "CURve evolution for VESsel segmentation." It is an implementation of the equations described in section 5.1 and operates on clinically-obtained images. The previous geodesic active contour models described only the flow of *hypersurfaces*: manifolds whose co-dimension is one. We extend these techniques to the evolution of manifolds of higher co-dimension. For our case of curves in three dimensions, the curves have dimension one and co-dimension two.

The high-level structure of the algorithm is the iterative modification of volume v to move closer to the final segmentation at each step. Specifically, the CURVES system takes in an MRA dataset or other similar volume as input. It uses this dataset both to generate an initial volumetric distance function v_0 and to drive the evolution. The most important component of the system is the evolution routine which repeatedly modifies this volume v according to the partial differential equation that corresponds to our customized geodesic active contours model. Periodically, it is necessary to reinitialize the evolving volume to be a true distance function to its current zero level set. This is necessary because the evolution equation modifies the volume so that it is no longer a distance function. At convergence or when desired, the zero level set is extracted from v for visualization of the segmentation. This framework is illustrated in Figure 5-1.

The curve evolution equation we use follows directly from an energy-minimization problem statement. Following the description of this equation, we discuss the primary components of the CURVES system including the choice to model the curves as tubes. When embedding that curve evolution in the evolution of a volume, we make a non-traditional choice for incorporating the image information. The calculation of distance functions is important to the system and we discuss two types of distance function estimation algorithms. We discuss the use of curvatures instead of eigenvalues in the regularization force and describe a modification to the image force that forces the orientation of the detected boundary to coincide with the orientation of

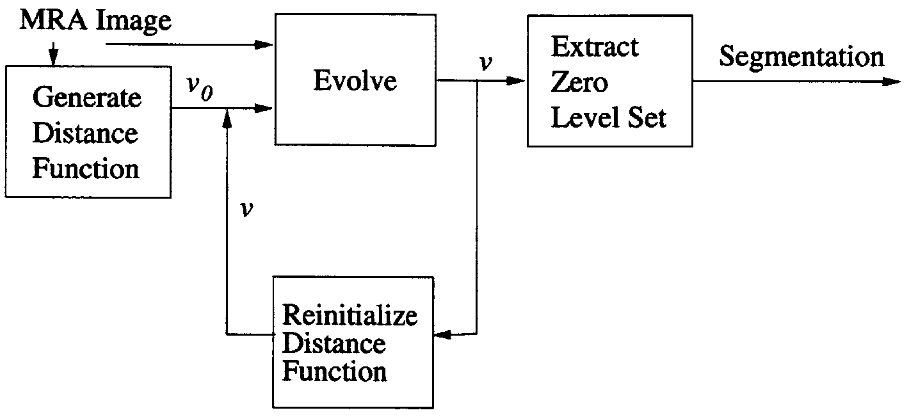


Figure 5-1: Overview of segmentation system. v is the evolving volume whose zero level set is the current segmentation estimate. An initial volume v_0 is generated and passed to the "Evolve" routine along with the image data, where it is evolved according to the partial differential equation derived from the energy minimization formulation. Periodically, v is reinitialized to be a distance function to its zero level set. Finally, the zero level set is extracted from v for visualization of the segmentation.

the intensity gradients in the image. Finally, we review the Marching Cubes isolevel surface extraction algorithm used to visualize the vascular segmentation.

5.1 Evolution Equation

For the case of 1D structures in 3D images, we wish to minimize

$$\int_0^1 g(|\nabla I(C(p))|)|C'(p)|dp \tag{5.1}$$

where $C(p) : [0,1] \to \mathbb{R}^3$ is the 1D curve, $I : [0,a] \times [0,b] \times [0,c] \to [0,\infty)$ is the image, and $g : [0,\infty) \to \mathbb{R}^+$ is a strictly decreasing function such that $g(r) \to 0$ as $r \to \infty$ (analogous to [20]). For our current implementation, we use $g(r) = exp(-r)$ because it works well in practice. Another possible choice is $g(|\nabla I|) = \frac{1}{1+|\nabla I|^2}$.

By computing the Euler-Lagrange equations, we find that the following holds at local minima:

$$\begin{aligned} 0 &= g\kappa\vec{N} - \Pi(\nabla g) && (5.2)\\ &= g\kappa\vec{N} - g'\Pi(\mathbf{H}\frac{\nabla I}{|\nabla I|}), && (5.3)\end{aligned}$$

where $\mathbf{H}$ is the Hessian of the intensity function and Π is the projection operator onto the normal plane to C. Initially, one would expect to set the temporal derivative of the curve C_t to the quantity on the right hand side of Equation 5.3 for gradient

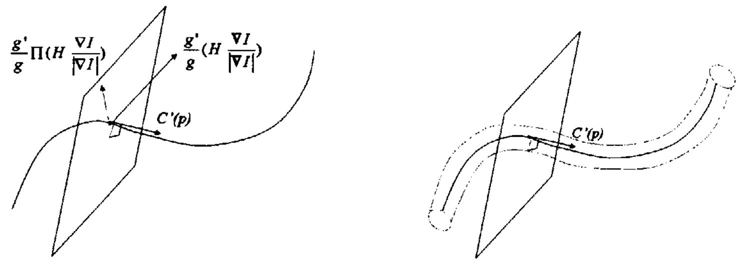

$\frac{g'}{g}\Pi(H\frac{\nabla I}{|\nabla I|})$
$\frac{g'}{g}(H\frac{\nabla I}{|\nabla I|})$
C'(p)
C'(p)

Figure 5-2: (a) The tangent to C at p, the normal plane, the image-based vector, and its projection onto the normal plane. (b) ε-level set method.

descent minimization. The update rule of Ambrosio and Soner in Equation 2.7 ([4]) requires, however, that the motion equation for C_t be the product of curvature and the normal vector, offset by an auxiliary vector field. In order to achieve this form, we divided Equation 5.3 by g before setting C_t to the quantity on the right hand side. This modification is valid since g is constrained to never attain a value of zero; it is expected to have effects on the speed of the evolution, but not on the local minimum obtained. We thus obtain the motion equation

$$\vec{C}_t = \kappa\vec{N} - \frac{g'}{g}\Pi(\mathbf{H}\frac{\nabla I}{|\nabla I|}). \tag{5.4}$$

The second term in Equation 5.4 is illustrated in Figure 5-2(a). In the notation of Equation 2.7, the auxiliary vector field is

$$\vec{d} = \frac{g'}{g}\mathbf{H}\frac{\nabla I}{|\nabla I|},$$

and the update equation for the embedding space v is

$$v_t = \lambda(\nabla v(x,t), \nabla^2 v(x,t)) + \frac{g'}{g}\nabla v(x,t)\cdot\mathbf{H}\frac{\nabla I}{|\nabla I|}. \tag{5.5}$$

where $\lambda(\nabla v(x,t), \nabla^2 v(x,t))$ is the smaller nonzero eigenvalue of $P_{\nabla v}\nabla^2 v P_{\nabla v}$ analogous to the F function in Section 2.2.3. We note that for $g(r) = exp(-r)$, we have $\frac{g'}{g} = -1$ identically. We retain the more general expression in terms of g in our equations because they are applicable to other choices of g as well.

We make some comments on the division by g involved in moving from Equation 5.3 to Equation 5.4. The evolution equations used in [18] do not have this division by g. Thus, they obtain the advantageous behavior described in Section 3.4.1 and illustrated with Figure 3-3. That is, the g function approaching zero causes the $g\kappa\vec{N}$ term to go to zero, stopping motion from this term, and the ∇g term also goes to zero at the steepest point on the gradient, stopping motion from this term as well. In the CURVES situation of Equation 5.4, the first term $\kappa\vec{N}$ is independent of the image I (recall that $g = g(I)$ is a function of the image) so it will not go to zero on sharp image gradients. In this sense, when the curve stops evolving, we have a balance between the two terms, but we do not have the more stable situation of both terms going to zero separately. We ran preliminary experiments in which we did not divide the right hand side by g, that is, in which $\vec{C}_t = g\kappa\vec{N} - g'\Pi(\mathbf{H}\frac{\nabla I}{|\nabla I|})$ and thus $v_t = g\lambda(\nabla v(x,t), \nabla^2 v(x,t)) + g'\nabla v(x,t) \cdot \mathbf{H}\frac{\nabla I}{|\nabla I|}$, but performance suffered in comparison to the original rules of Equation 5.4 and Equation 5.5. Large-scale analysis of the reasons for the change in performance has not been done. We currently retain Equation 5.4 and Equation 5.5 due to empirically superior segmentation results.

In summary, Ambrosio and Soner's work has provided the basis for the use of level set methods to segment 1D structures in 3D. Equation 5.5 will become the main body of in the CURVES segmentation algorithm.

5.2 ε-Level Set Method

Initial experiments required that the evolving volume be a distance function to the underlying curve; however, it was not clear how to robustly extract the zero level set or even evolve those points since the distance function was singular exactly there. Moreover, the projection operator P_q is defined only for non-zero vectors q, so the method is undefined at $\nabla v = \vec{0}$, which is the curve itself, and is numerically unstable near the curve. For this reason, we developed the *ε-Level Set Method* which defines a thin tube of radius ε around the initial curve, then evolves that tube instead of the curve (Figure 5-2(b)). ε does not denote a fixed value here, but means only that the evolving shape is a "tubular" surface of some unspecified and variable nonzero width. Thus, we are now evolving surfaces similar to minimal surfaces [18], but that follow the motion of the underlying curve so they do not regularize against the high curvatures found in thin cylindrical structures such as blood vessels and bronchi. In addition to being more robust than the straightforward method of using a 3D distance function to curves, this method better captures the geometry of such structures, which have nonzero diameter.

We stress that this is an approximation to evolving the underlying curve but is not equivalent. If we were to constrain the width of the tube to remain constant along the tube, it would be equivalent; however, allowing the image to attract local surface areas independently causes the width to vary, so the tube is no longer as isolevel set of the distance function to its centerline.

5.3 Locality of Image Information

For geodesic snakes of any dimensionality and codimensionality one must compute some curvature and some external image-related term at each point on the higher-dimensional manifold (the surface in the case of a planar curve, the volume in the

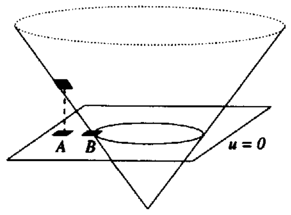


Figure 5-3: To evolve a point on the distance function, CURVES chooses image information from A instead of B.

case of a space curve). For each of these terms, one can use the values defined at the particular location or those defined at the closest point on the zero level set (Figure 5-3). Traditional level set segmentation methods use the image term from the closest point on the level set, but compute the curvature term locally [20, 17, 56]. The reason is that the curvature term is defined locally, and the level-set-equivalence relation says that indeed one should use that local curvature. The image-term, conversely, is not defined locally if one regards the problem as evolving the curve directly. One must, then, "invent" an image term at those points off the zero level set. The choice that best implies a smooth evolution is the use of the image term at the nearest point on the zero level set. This choice keeps the evolving "distance function" as close to a true distance function as possible without modifying the curvature term. Alternative formulations keep the evolving manifold a distance function throughout the evolution [45, 113] using image or other information from the object boundary as well as curvature information from the boundary only; no local information is used at all.

The CURVES method, however, uses the image term at each location on the higher dimensional manifold instead of propagating the image data off the current zero level set. This choice was made to enable the evolving surface to be attracted to edge gradients that are not on the current surface. For example, if there are two neighboring tubes in the image and the curve or surface is initialized near one, CURVES can capture the other tube; the traditional method cannot. However, this also means that the CURVES method is not equivalent to explicit Lagrangian evolution, which would not find the second tube. The reason that neither an explicit evolution nor a traditional level set evolution would find the second tube is that they are stopped by the local minimum found at the outline of the single tube. CURVES is thus less sensitive to initialization than previous level set methods are.

This choice also has implications for the need to reinitialize the evolving higher-dimensional manifold to be a distance function. In general, all of the level sets are evolving toward the same local minima implied by the image force in traditional methods; they thus become increasingly close together so the manifold is no longer a distance function [45]. In the case of CURVES, the local image much more severely invalidates the distance function constraint than does the general motion of all of the level sets to the local minima. This invalidation is due to the use of potentially

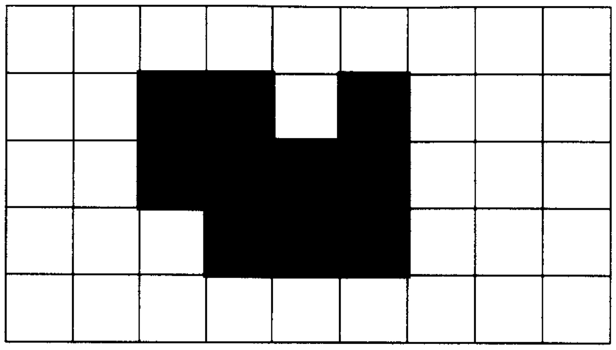


Figure 5-4: A binary image of an object to which we would like to build an L_1 distance map.

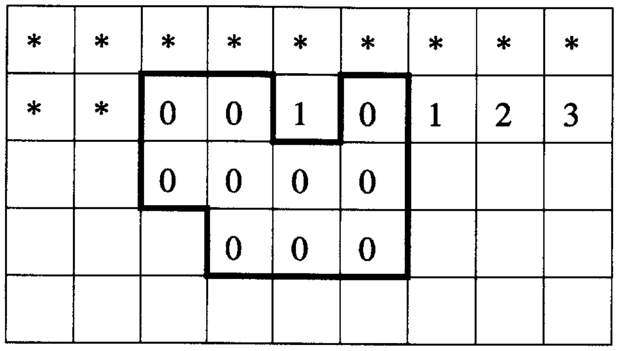
very different image values used to update the "distance function" in a small region.It follows that CURVES requires far more reinitializations than the traditional method. However, if we wish the image information off the zero level set to affect the evolution, we cannot reinitialize too frequently. For example, reinitializing after every step in the evolution is equivalent to using only the image information on the zero level set, since the reinitialization maintains only those values, updating all other values to be their distance to the zero level set.

5.4 Banding

Instead of evolving the entire volume, we evolve only the portion of the volume within a narrow band of the zero level set (the current surface). Normally, we set the band to include voxels that are 4 to 6 voxels away from the surface. This aspect of the implementation does not have the same meaning as "banding" in previous geodesic active contour methods where the image data on the zero level set is propagated throughout the band [2, 78, 23]. We simply mean that only those points are evolved. Note that, unlike these other methods, CURVES is sensitive to the width chosen for the band since image values therein are indeed used in the evolution.

5.5 Computing Distance Functions

Recall from Section 2.2 that a *distance function* to a given object boundary is a function whose value at each point is the distance from that point to the object boundary. A *signed distance function* is a distance function with the modification that each distance is labeled as either positive or negative. For our purposes, we will use negative labels to denote the interior of the object and positive labels to denote the exterior of the object. Although we are interested in the *signed* distance function computation, we will usually have prior labels indicating "interior" versus "exterior",



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

Figure 5-5: State of the distance function $d(x, y)$ after passing the mask M_1 over the first two rows only. All unmarked pixels have infinite distance at current estimate, and have not been visited. *'s indicate pixels that have been visited but still have infinite estimated distance.

so it is the distances themselves with which we concern ourselves. Observe that the distances must increase with slope one as one moves away from the object in a normal direction. Letting v be the volume in question, this constraint is written as

$$|\nabla v| = 1, \tag{5.6}$$

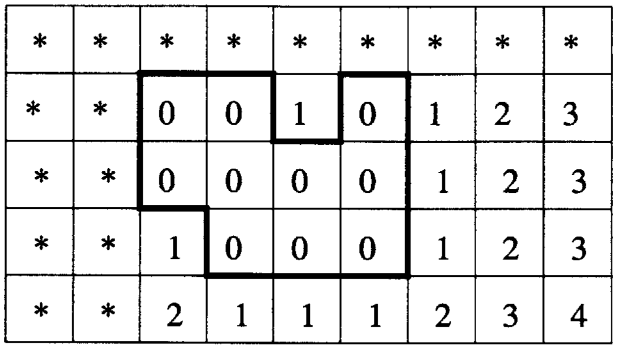
which is known as the *Eikonal equation.* The step of generating distance functions to object boundaries is crucial to our segmentation algorithm and is potentially very slow, so we experimented with various algorithms searching for the most efficient method that gave an approximation to the distance function that was adequate for our purposes.

5.5.1 Bi-directional Propagation

A common method of distance function computation in computer vision is to use what is called the *chamfer method* [14]. A very readable discussion of this approach is given in [94]. The family of efficient algorithms for computing an approximation to the distance function to which this method belongs is called *bi-directional propagation.* Assume that we would like to compute the distance function defined by the L_1 distance to the object in the 2D images in Figure 5-4. In $\mathbb{R}^n$, the L_1 distance between $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$ and $\hat{x} = (\hat{x}_1, \ldots, \hat{x}_n) \in \mathbb{R}^n$ is defined as

$$|x - \hat{x}|_1 = \sum_{i=1}^{n} |x_i - \hat{x}_i|.$$

The idea of the these methods is to create a mask that will be passed over the image (two times) to propagate the distances throughout the image. For the L_1 distance, the mask M is



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

Figure 5-6: State of the distance function $d(x, y)$ after one complete pass of M_1 over image. *'s indicate infinite estimated distance.

$$\begin{array}{ccc} & 1 & \\ 1 & 0 & 1 \\ & 1 & \end{array}$$

which is divided into two masks, M_1:

$$\begin{array}{ccc} & 1 & \\ 1 & 0 & - \\ & - & \end{array}$$

and M_2

$$\begin{array}{ccc} & - & \\ - & 0 & 1 \\ & 1 & \end{array}$$

The distance function $d(x, y)$ will be initialized for each location (x, y) according to

$$d(x,y) = \begin{cases} 0 & \text{if } (x,y) \text{ in object} \\ \infty & \text{otherwise} \end{cases}$$

Then the mask M_1 will be passed over the image in left-to-right, top-to-bottom order, so that at each location (x, y) the update rule is

$$d(x,y) \leftarrow \min \left(\begin{array}{cc} & d(x,y-1)+1, \\ d(x-1,y)+1, & d(x,y) \end{array} \right),$$

where x is increasing in the left-to-right direction and y is increasing in the top-to-bottom direction. Figure 5-5 shows the intermediate distance function resulting from passing M_1 over only the first two rows of the image. Those pixels marked with a "*" were visited, but still retain their initial distance estimate of ∞. Figure 5-6 shows the distance function after one complete pass of M_1 over the image.

The next step of the algorithm is to pass M_2 over the image in the opposite order of the ordering used with M_1. That is, right-to-left and bottom-to-top ordering for

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

Figure 5-7: State of the distance function $d(x, y)$ after passing M_1 and M_2 over the image. This is the final distance function.

this example. The update rule for the M_2 mask is

$$d(x,y) \leftarrow \min \left(\begin{array}{cc} d(x,y), & d(x+1,y)+1, \\ d(x,y+1)+1 & \end{array} \right),$$

Figure 5-7 shows the result of this step, which is also the final distance function. Thus, the entire distance function has been computed in only two passes over the image, which gives time complexity that is constant in the number of pixels.

For this case of the L_1 distance, bi-directional propagation gives an exact distance function. However, for the Euclidean, or L_2, distance, an exact solution is not possible. Instead, the method gives an approximate answer whose accuracy depends on the mask used. The simplest approximation is the use of the mask

$$\begin{array}{ccc} \sqrt{2} & 1 & \sqrt{2} \\ 1 & 0 & 1 \\ \sqrt{2} & 1 & \sqrt{2} \end{array}$$

It is bi-directional propagation with this 3×3 mask that is called the *chamfer method*, although sometimes the term is used loosely to include any bi-directional propagation algorithm. A better approximation than that obtained with the chamfer mask is obtained with the larger mask

$$\begin{array}{ccccc} & \sqrt{5} & & \sqrt{5} & \\ \sqrt{5} & \sqrt{2} & 1 & \sqrt{2} & \sqrt{5} \\ & 1 & 0 & 1 & \\ \sqrt{5} & \sqrt{2} & 1 & \sqrt{2} & \sqrt{5} \\ & \sqrt{5} & & \sqrt{5} & \end{array}$$

For numerical reasons, one may prefer integer estimates, such as the mask

$$\begin{array}{ccccc} & 225 & & 225 & \\ 225 & 141 & 100 & 141 & 225 \\ & 100 & 0 & 100 & \\ 225 & 141 & 100 & 141 & 225 \\ & 225 & & 225 & \end{array}$$

which is the above mask scaled by 100, then rounded. This mask approximates the true L_2 distance to within 2% [94]. For further discussion of such masks, the reader is referred to [66].

These masks extend naturally to 3D distance function computation. The drawback is the error inherent in the method when used for Euclidean distances and also the inability to construct a distance function to a surface defined with subvoxel accuracy, both of which are needed for our application.

An extension to bi-directional propagation which increases its accuracy is the storing, for each pixel location in the image, of the pixel location on the object from which the distance was calculated. That is, it is the pixel that is "closest" to the object; "closest" is in quotes because it is in the sense of this distance function approximation, not in the sense of an exact computation. This is only a constant-time expense in efficiency, since this location value is propagated at the same time as the distance itself. Once the forward and reverse passes are complete, one performs a final pass over the data during which exact distance values are computed between each pixel location and these points from the object that are associated with them. While still not providing an exact measure, this method provides significantly higher accuracy. Even higher accuracy can be obtained by computing exact distances to the associated object points at each step in the propagation.

Related methods which compute Euclidean distances and still require a small number of images scans have been described and explored by Danielsson [33]. A subsequent method applies morphological operators to compute the exact Euclidean distance at a limited number of pixels for efficiency [32]. These methods overcome the approximation error of simply adding distances, but retain the limitation to distances to only discrete pixel (voxel) locations: the chamfer method and its variations are effective when the object boundary exactly coincides with discrete pixel locations. For our implicit representation, however, this is not the case. In particular, we have a 3D volume with a value at every point, and the goal is to reset that volume to be the signed distance function to its current zero level set which is an implicitly defined surface of subvoxel precision.

A potentially more precise but inefficient approach to obtain the unsigned distances would be to extract the zero level set explicitly using Marching Cubes [71], which would generate the zero level set as a list of triangles. The unsigned distance could then be computed for each voxel by computing the distances from that voxel to each of the triangles and selecting the minimum over those distances. Regarding the subsequent choice of sign for each voxel, *i.e*, whether it is interior or exterior, an analytical calculation using surface normals of the triangles could be used. This may yield ambiguous results near edges or vertices of the triangles, and in general there is not a known stable method for determining this sign. For our problem, however, one knows that the sign of each voxel should remain unchanged from the previous step, since the zero level set should not move. Thus, we can just copy the signs from the previous volume.

It would be inefficient to compute the distances from each voxel to each triangle for complicated surfaces which would have large numbers of triangles. However, an efficient hybrid method combining the triangle vertices approach with bi-directional propagation could also be constructed. In such a method, one would use the triangle vertices to generate precise distance function values in the voxels that are within one voxel of the triangle surface model. One would then use bi-directional propagation to efficiently propagate these distances throughout the remainder of the volume.

This use of the triangles is likely more precise than the usual bi-directional method of reducing the object boundary to a discrete binary representation. However, it is dependent on parameters of the Marching Cubes method, such as allowed angles between triangles, number of triangles returned, and other tolerances. Moreover, it also has the disadvantage of being a piecewise planar approximation to the true boundary instead of the true boundary itself.

Bi-directional propagation is very efficient, but it is only an approximation to the true distance. It turns out that for the very thin and complicated structures we wish to segment, such as blood vessels, the error in this approximation has a significant negative affect on segmentation performance. Because some structures may be as thin as only one voxel in width at some places, the quantization error of rounding to nearest voxels and other error due to summing of discrete distances can be significant enough to "break" those structures. We need a more precise method that does not perform this inexact rounding to either voxels or triangle vertices. Specifically, we would like a precise smooth solution satisfying Equation 5.6. We naturally turn to the use of partial differential equations to obtain this property.

5.5.2 PDE Method

The partial differential equation

$$v_t = 1 - |\nabla v| \tag{5.7}$$

was proposed by Rouy *et. al.* [93], where it is assumed that $v \geq 0$ on some region Ω^+ and $v = 0$ on its boundary $\partial\Omega^+$. One repeatedly applies Equation 5.7 to $\Omega^+ - \partial\Omega^+$ until convergence, at which time $|\nabla v| = 1$. The difficulty with this method is the requirement that the zero level set be known. Sussman *et. al.* thus modified Equation 5.7 to apply to the entire domain of v, without the zero level set being known explicitly [104]. Let v_0 be the initial function which we would like to reset to a distance function to its zero level set. Their PDE evolves v into a (signed) distance function with the property that the zero level set of v remains the same as that of v_0 over the evolution. This property is achieved by the incorporation of the sign of the previous values into the PDE. Their new PDE is

$$v_t = sgn(v_0)(1 - |\nabla v|) \tag{5.8}$$

where sgn is the sign function $sgn(x) = \frac{x}{\sqrt{x^2}}$ for any nonzero scalar x and $sgn(0) = 0$ by definition. They suggest smoothing the sign function $sgn(v_0)$ by replacing it with

$$sgn_\varepsilon(v_0) = \frac{v_0}{\sqrt{v_0^2 + \varepsilon^2}} \tag{5.9}$$

for numerical reasons.

One difficulty with Equation 5.8, however, is that it is not robust to slight inaccuracies in those values of v_0 near the zero level set. These inaccuracies can cause those values of v_0 to have the wrong sign, thus causing the zero level set to move slightly, which is undesirable. The smoother sign function proposed in Equation 5.9 can lessen the problem, but still can result in numerical instabilities near this boundary. Further, it is heuristic and depends on the choice of ε.

Sethian proposed a different method of creating such a distance function which does not have this difficulty at the border [99]. This method computes the positive (exterior) distances by propagating the zero level set of v_0 forward with speed $\beta = 1$ in the normal direction:

$$S_t = 1\vec{N_{in}}$$

where $\vec{N_{in}}$ is the inward pointing normal and S is the surface to be evolved, or

$$S_t = 1\vec{N_{out}}$$

where $\vec{N_{out}}$ is the outward pointing normal. Because we choose the interior of the object to have negative distance and the exterior to have positive distance within our level set implementation, $|\nabla v|$ gives the outward pointing normal of the isolevel surface. Thus, our level set update rule comes from the equation in terms of $\vec{N_{out}}$, yielding

$$v_t = -|\nabla v|.$$

One then stores the time at which this front passes each voxel; that time gives the distance of the voxel to the zero level set of v_0. Note that this level set is an implicit representation of the evolving front, so the contour is never extracted explicitly. Since we will need multiple time steps to move this front (the zero level set of the volume) and to store the given time step at which it crosses each grid point, we do not actually write to the volume v which is used for the segmentation evolution. Instead, we copy v to some temporary volume w when we wish to reinitialize it to be a distance function to its zero level set. It is w that is then updated by

$$w_t = -|\nabla w|.$$

And when the zero level set crosses a grid point, the time at which the crossing

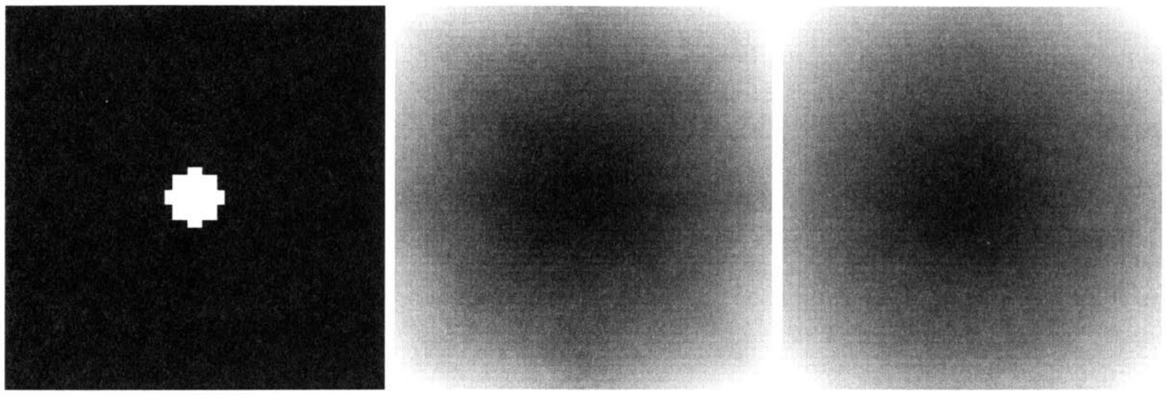


Figure 5-8: A white object on a black background, followed by two different signed distance functions to the object. The first (center image) is computed with a chamfer map, the second (rightmost image) with the level set method. We thank Mike Leventon for code used to compute the chamfer map and Renaud Keriven for code used to compute the PDE map.

occurred is written to v, at that grid point, as this gives the distance of that grid point to the zero level set of v. A final detail is that we actually need two volumes w_1 and w_2 for this evolution, since the need for gradients prevents performing the updates "in place". That is, each iteration must compute values from one and write them to the other. (This is likewise necessary in the segmentation update of v.) To compute the negative (interior) distances, one repeats the procedure with normal speed $\beta = -1$:

$$w_t = |\nabla w|.$$

Keriven [55] notes that the advantages of this method are that it does not move the zero level set, it can be implemented in large part with the fast methods described in [1], and it can be truncated naturally at some pre-specified distance from the boundary when distances are only required in a narrow band around the object. It is the implementation of the $\beta = +/-1$ evolution by Keriven [55] that we use in CURVES.

We experimented with the chamfer method and the variant of the chamfer method that uses the triangle vertices because these versions are faster than our $\beta = +/-1$ implementation. We found, however, that the error in the chamfer approximations caused CURVES to find worse segmentations than it found with the implementation of [55]. Figure 5-8 illustrates this situation. It shows two distance functions to a roughly circular object. The leftmost image is the object, and the center image is the distance function created with the basic chamfer map using the 3×3 mask. We acknowledge Mike Leventon for code used to compute the chamfer map. The rightmost image is the smoother distance function created with the PDE method. Notice the rays that appear to extend from the circle toward the corners in the chamfer map. These inaccuracies in the distance map can be problematic for algorithms that require high accuracy. Thus, we retained the PDE method which produces slower but higher quality distance functions.

In conclusion, our reinitialization of the distance function is itself implemented with a level set method. To obtain the positive distances, the surface is propagated outward at constant speed of 1, and the distance at each point is determined to be the time at which the surface crossed that point. A second step propagates the surface inward to obtain the negative distances analogously. We reinitialize the distance function every 3 to 5 steps; this is much more frequently than previous level set techniques for reasons discussed in Section 5.3.

5.6 Regularization Term

For computational efficiency and because of numerical instability of the gradient computations and thus the evolution equation near $\nabla v = \vec{0}$, we remark that the level sets of the function v flow in the direction of the normal with velocity equal to the sum of their smaller principal curvature (scaled by $|\nabla v|$; see Section 2.2.3) and the dot product of ∇v with the image-based vector field $\vec{d}$. Therefore, we compute the smaller curvature p_2 directly from v instead of as an eigenvalue of $P_{\nabla v}\nabla^2 v P_{\nabla v}$, as in

$$\lambda(\nabla v(x,t), \nabla^2 v(x,t)) = |\nabla v| p_2.$$

p_2 is computed from the mean and Gaussian curvatures of the surface by solving a quadratic equation. The mean curvature κ_M and Gaussian curvature κ_G are available directly from the volume v according to

$$2\kappa_M = \nabla \cdot \frac{\nabla v}{|\nabla v|} = \frac{\left\{ \begin{array}{c} (v_{yy}+v_{zz})v_x^2 + (v_{xx}+v_{zz})v_y^2 + (v_{xx}+v_{yy})v_z^2 \\ -2v_xv_yv_{xy} - 2v_xv_zv_{xz} - 2v_yv_zv_{yz} \end{array} \right\}}{(v_x^2+v_y^2+v_z^2)^{3/2}} \tag{5.10}$$

and

$$\kappa_G = \frac{\left\{ \begin{array}{c} v_x^2(v_{yy}v_{zz}-v_{yz}^2) + v_y^2(v_{xx}v_{zz}-v_{xz}^2) + v_z^2(v_{xx}v_{yy}-v_{xy}^2) \\ +2[v_xv_y(v_{xz}v_{yz}-v_{xy}v_{zz}) + v_yv_z(v_{xy}v_{xz}-v_{yz}v_{xx}) + v_xv_z(v_{xy}v_{yz}-v_{xz}v_{yy})] \end{array} \right\}}{(v_x^2+v_y^2+v_z^2)^2}. \tag{5.11}$$

These equations are given in [100, 107]. We then use the definitions that the Gaussian curvature is the product of the two principal curvatures and the mean curvature is their average to compute the p_2 by solving a quadratic equation.

5.7 Image Term

To control the trade-off between fitting the surface to the image data and enforcing the smoothness constraint on the surface, we add an *image scaling* term ρ to Equation 5.5 to obtain

$$v_t = \lambda(\nabla v(x,t), \nabla^2 v(x,t)) + \rho \frac{g'}{g} \nabla v(x,t) \cdot \mathbf{H} \frac{\nabla I}{|\nabla I|}. \qquad (5.12)$$

ρ is set by the user or can be pre-set to a default value. Empirically, we have found it useful for ρ to be set so that the image-related term $\rho \frac{g'}{g} \nabla v \cdot \mathbf{H} \frac{\nabla I}{|\nabla I|}$ and the curvature-related term $\lambda(\nabla v, \nabla^2 v)$ are the same order of magnitude.

The second modification to the image term is more fundamental and controls the direction of the angle between the surface normal and the image intensity gradient. Because vessels appear brighter than the background, we weight the image term by the cosine of the angle between the inward normal to the surface and the gradient of the image. This cosine is given by the negated dot product of the respective gradients of v and I, so the update equation becomes

$$v_t = \lambda(\nabla v, \nabla^2 v) - \rho(\frac{\nabla v}{|\nabla v|} \cdot \frac{\nabla I}{|\nabla I|}) \frac{g'}{g} \nabla v \cdot \mathbf{H} \frac{\nabla I}{|\nabla I|}. \qquad (5.13)$$

This dot product is negated because ∇v gives the outward normal and we want the inward normal. For example, if the inward surface normal and the image gradient point in the same direction, then the brighter region is inside the surface and the darker region is outside; the angle between the vectors is 0, whose cosine is 1, so the image term is fully counted. However, if they point in opposite directions, the negative weighting prevents the evolving vessel walls from being attracted to image gradients that point in the opposite direction.

It should be noted that this change to the image force can also be viewed as an approach toward increasing the tolerance to image noise. We can rewrite the last term in Equation 5.13 so that the outer product matrix $\frac{\nabla v \nabla v^T}{|\nabla v|^2}$ is multiplied by ∇I. The result is the projection of ∇I onto the direction of ∇v. At stages in the evolution where the surface is approaching the correct segmentation result, this has the effect of removing noisy components of ∇I that may be orthogonal to the true intensity gradients.

5.8 Numerical Issues

With regards to the numerical issues involved in computing the gradients for level set evolutions (Section 2.2.4), we make different choices for the two level set techniques used in CURVES.

For the segmentation evolution, Equation 5.5, there are two terms as in the ex-

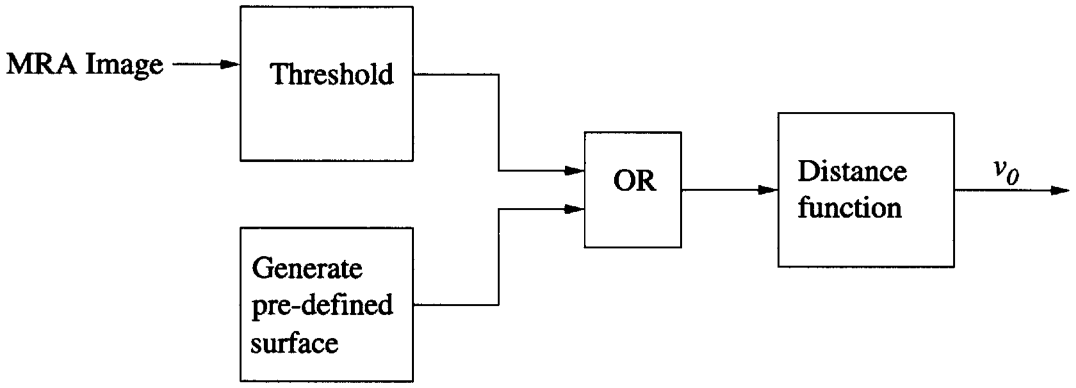


Figure 5-9: Illustration of initialization part of CURVES.

ample in Section 2.2.4. The first depends on curvature, so should be computed with central differences because information propagates in all directions off the curve. The theory says that we should use an upwind scheme for the image-dependent term since the direction is dependent only on an external force. In practice, however, we have found central differences to yield smoother and better segmentations. The reason is that central differences are more accurate than the upwind scheme because they look at a larger neighborhood for gradient computation, in the absence of singularities. It is these singularities that motivate the use of the upwind scheme. For smooth areas in the distance function v, such as the neighborhood around the zero level set, the upwind scheme is not necessary. Thus, for empirical reasons, we use central differences for this second term in Equation 5.5 as well.

The evolution equation that is used to reinitialize the distance function as described in Section 5.5.2, however, is a simple convex external speed function, so we do use the upwind method described in Section 2.2.4. We do not experience the smoothness difficulties which warranted the central differencing method in the segmentation evolution because the speed function is much smoother in this case due to the absence of the image term. For this evolution, the upwind scheme does in fact achieve smooth and accurate distance function. We thank Renaud Keriven for the code used to compute these reinitializations [55].

5.9 Initial Surface

Like all active contour and surface approaches, CURVES begins with an initial boundary estimate. Because our segmentation surface is represented implicitly by a distance function, we must start with an initial function. This initial volume is created in one of two ways. The usual approach is to threshold the raw dataset to obtain an initial boundary estimate, then to build the distance function to this surface. One may also manually edit the initialization to remove noise artifacts if desired. The second approach is to simply provide a prior arbitrarily constructed surface and build a distance function to it. Figure 5-9 illustrates this process. Our early experiments used this second approach to test the capture range of the segmentation algorithm. For example, we initialized the algorithm with a vertical bar in the center of the volume as shown in Section 6.3.

5.10 Convergence Detection

Convergence of the algorithm is detected when the volume of the segmented region changes less than some specified percentage of total volume, across a specified number of iterations. We usually choose .01 for this percentage and 10 or 20 for the number of iterations. One can expect to achieve similar behavior using a smaller percentage over a smaller number of iterations or using a larger percentage over a larger number of iterations.

Due to the trade-off between evolving the "distance function" to fit image information and reinitializing it to be a true distance function, the volume does not converge to a distance function to its zero level set. That is, the image force is repeatedly driving it away from a distance function, and the reinitialization step returns it to a distance function periodically. Thus, what we wish to capture is convergence of the zero level set surface, given a particular frequency of reinitializations. This is the reason we do not check at successive steps, but instead across a larger number of steps.

We experimented with variations on this protocol. The first experiment was to add a loop on top of this checking protocol which would require some number Y of convergence checks to be satisfied before declaring convergence. We tested Y values of three and four. We observed with this protocol that the evolution could enter an infinite loop: the segmentation would expand and contract slightly due to the image-reinitialization tradeoff, so would never determine convergence. One could of course decrease the frequency of checking, but loops are possible for any frequency. This is an undesirable aspect of the algorithm. At the current time we only use the single convergence check ($Y = 1$) for empirical reasons. This raises an interesting question for future study: can we guarantee that successive steps indeed decrease the "energy" of the evolving manifold, as defined by the energy functional in (5.1) or a related expression? Note that it may not be possible to use the fundamental geodesic active contours energy function (5.1) directly since we use image information off the curve.

5.11 Marching Cubes

Whenever we would like to view the segmentation, the zero level set is extracted using *Marching Cubes* [71]. This is a popular method for extracting isolevel surfaces from 3D data in the graphics and visualization communities. It operates on the

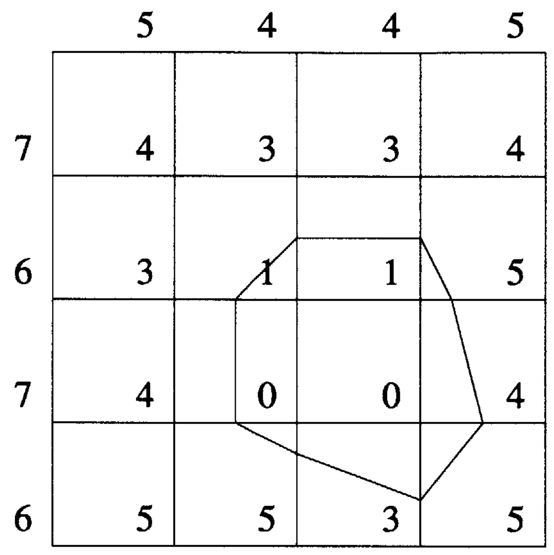


Figure 5-10: The 2-isolevel contour of a 2D array.

principle that for each cell (the cubes whose vertices are neighboring voxels), there is only a finite number of ways the contour can pass through that cell. These ways are differentiated according to which of the eight vertices of the cell are inside the contour and which are outside. There are thus $2^8 = 256$ ways the contour can pass through a cell. These ways are called the topological *states* of the cell, and once the topological state of a cell is determined, the exact contour through that cell can be computed by linear interpolation of the values at the vertices.

The 2D analog of Marching Cubes is called *Marching Squares.* We describe Marching Squares first to facilitate understanding of Marching Cubes. Marching Squares is a technique for producing isolevel curves of a 2D array of values, such as an image or a 2D distance function (Figure 5-10). In this case, a cell is a square whose vertices have values given by the 2D array. Analogous to Marching Cubes, it treats each cell independently. Because a cell has 4 vertices, there are $2^4 = 16$ topological states a cell can exhibit as shown in Figure 5-11. Note that two of the states exhibit an ambiguity about where to draw the contour because there is not enough information to decide; in general one choice is chosen arbitrarily. Each state can be represented with a 4-bit index, which will be used to index into a table giving the proper contours to draw (modulo the array values, which must be interpolated) for each state. For a pre-selected value at which the contour is desired, Marching Squares does the following.

1. Choose a cell.

2. Examine the value of each vertex in the cell to see if it is above or below the value. This tells if it is inside or outside the contour. Note that it does not

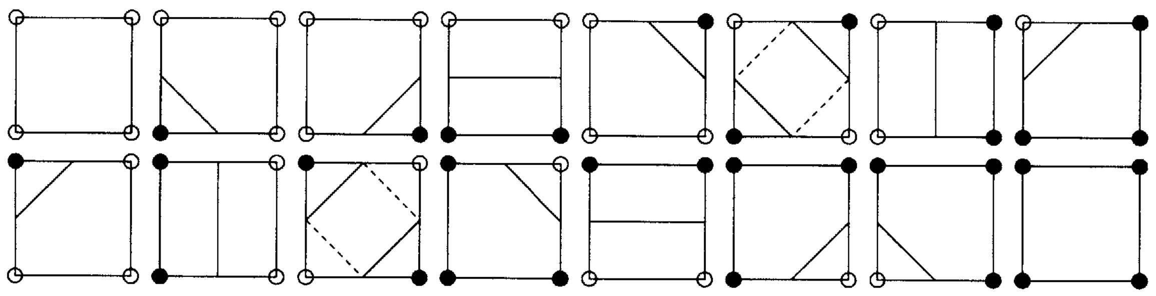


Figure 5-11: The 16 topological states a cell can have according to Marching Squares. Each square represents a state: filled-in circles indicate that the vertex is inside the contour, empty circles indicate that the vertex is outside the contour. The lines within the squares indicate where to place the contour, although the values at the vertex will indicate exactly where along the particular edges it should be placed. The dotted lines indicate an ambiguity in this choice.

matter whether higher/lower indicates inside/outside or vice versa, but this must be defined *a priori.*

3. Encode the inside/outside information in a 4-bit index.

4. Use this index to probe the table to obtain a rule for how to draw the contour.

5. Using his rule, interpolate the specific array values at the vertices to draw the precise contour.

This method has the advantage of being easy to implement and efficient, due to the table and the ability to perform only one pass over the data. One drawback is that many disconnected contour segments are produced since each cell is processed separately. In practice, one uses a post-processing step to connect these segments.

The 3D version, Marching Cubes generates isosurfaces analogously. In this case, the cells are cubes with eight vertices, so there are 256 possible states. This large number can be reduced using rotational and reflective symmetries to the 15 states pictured in Figure 5-12. In this figure, black circles indicate vertices that are inside the isosurface. The triangles within the cubes indicate where to place the triangles that comprise the isosurface, although the values at the vertices will indicate exactly where along the particular edges they should be placed. Like in the Marching Squares case, there are some ambiguities. Here, the six states with ambiguities are indicated with a "*". Unfortunately, resolving the ambiguities is more difficult for the 3D case than for the 2D case. In the 2D case, either interpretation is acceptable in the sense that the contour still remains connected. In the 3D case, however, some choices may lead to holes in the isosurface. This difficulty can be addressed by a post-processing decision step, by the use of tetrahedra (which avoid ambiguity since they are simplices) instead of cubes, or by the addition of several "complementary states" to the illustrated 15

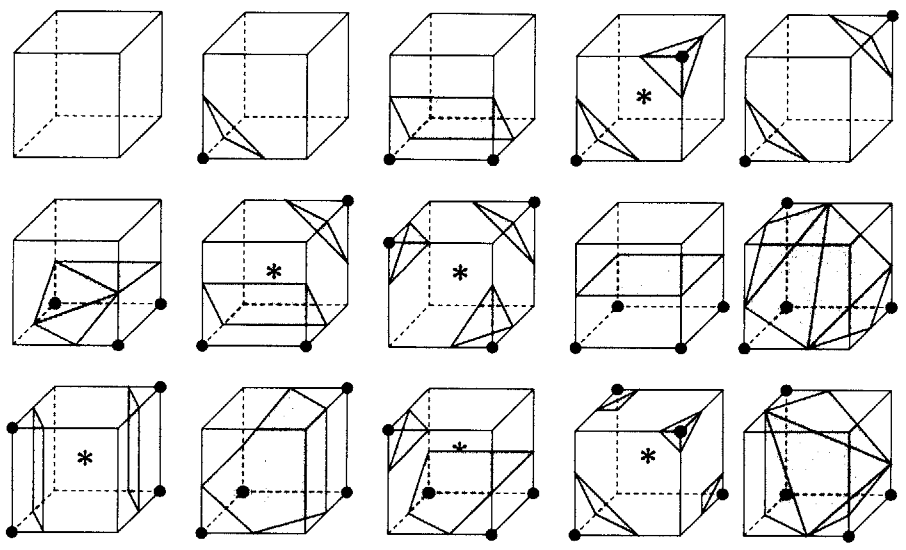


Figure 5-12: The 15 topological states for Marching Cubes, which have been reduced from the original 256 states by symmetry. Black circles indicate vertices that are inside the isosurface. The triangles within the cubes indicate where to place the triangles that comprise the isosurface, although the values at the vertices will indicate exactly where along the particular edges they should be placed. There is ambiguities in the 6 states marked with an asterisk "*".

states. The reader is referred to [98] for further detail and discussion. In CURVES, we use the implementation of Marching Cubes available with the *Visualization Toolkit (VTK)* [98]. This set of visualization libraries provides a wide range of visualization functionality and is available without cost from `http://www.kitware.com/vtk.html`. We use it for all rendering in the CURVES system.

5.12 Pre-processing, post-processing

Before running CURVES on medical datasets, the datasets are usually smoothed by a small isotropic Gaussian since the algorithm inherently requires some smoothness of gradients. The sigma normally used is 0.75mm in each dimension, where a typical dataset has voxel dimensions of .9375 x .9375 x 1.5mm^3.

We post-process the segmentations to remove any surface patches whose surface area is less than some threshold (a parameter of the method) to eliminate patches corresponding to noise in the original data. Alternately, the user can base the thresholding on volume of connected components instead of surface area, removing all structures whose volume is less than some threshold. This step can remove "noise" that may have been incorrectly segmented or can enable the user to focus only on the largest connected structures.

Chapter 6

Results

We have run CURVES on approximately 30 medical datasets and on synthetic volumes as well. Initial experiments on simple tubular structures were conducted to understand the behavior of the curve evolution differential equation. Once medical experiments were started, our first results demonstrated the potential of the method to operate effectively on such complicated structures in the presence of imaging noise. Results were compared visually to simple thresholding of the raw data to see in which cases the curve evolution approach is most important [75]. With further experimentation, the algorithm was refined to have the improved performance we currently observe [74]. While experimenting, we simultaneously explored various validation options. Prior to showing results, this chapter explains various validation options, why it is difficult to obtain quantitative performance measures, and what validation strategies we chose for this research. We then provide images showing results of the CURVES segmentation system, first on synthetic data, then on medical datasets. Medical image segmentations are validated against simple thresholding and against manual segmentations by a neurosurgeon.

6.1 Validation Issues

It is difficult to quantify the performance of an automatic segmentation algorithm whose output is as complicated as a vessel tree. The primary difficulty is in obtaining *ground truth.* The ideal ground truth would be a correct labeling of each voxel as "vessel" or "not vessel", and also a surface model showing the exact vessel wall to increase precision in areas of partial voluming. The first reason this is difficult to obtain is the complexity of the vasculature. An image of a cerebral vessel model obtained by *plastination* is shown in Figure 6-1. Plastination is a process in which a glue-like substance in injected into the vessels of a cadaver. This substance is carried throughout the vessels in the anatomical region of interest. After it hardens, the anatomical area is caused to disintegrate, so that the only remaining structure is the hardened glue in the shape of the vessels. This figure shows the huge number of

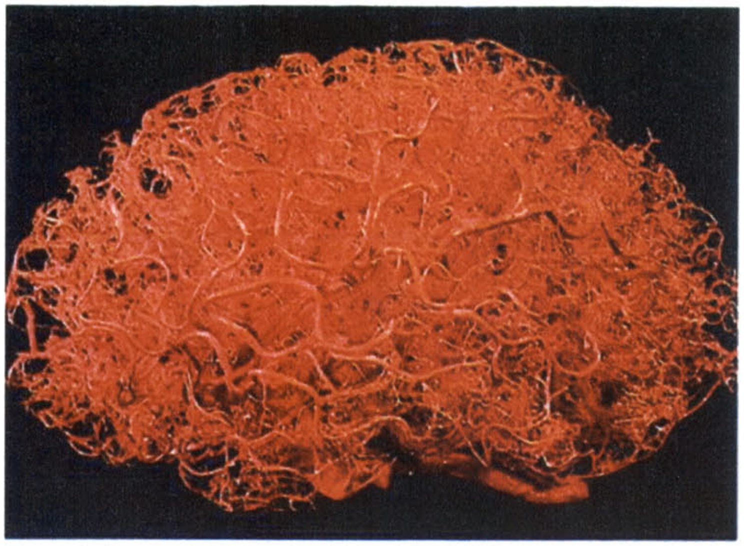


Figure 6-1: Plastination model of cerebral vessels in a human brain. Image reproduced from [114] with permission from Zlokovic, Keck School of Medicine, University of Southern California.

vessels and intricate 3D arrangement and branching pattern.

The availability of such knowledge about the locations of the vessels, although unwieldy, raises the possibility of constructing an atlas of the vessels to use as ground truth in validation experiments. One could register the segmented model to the atlas then calculate a score of what percentage of the total vasculature was obtained. This idea, however, is precluded by the large variability of these structures across individuals. For example the Circle of Willis, a structure connecting the major cerebral arteries, is not fully connected in a large fraction of people. Yasargil found percentages of occurrence for the major cerebral vessels by studying cadavers [110], but incidence rates for all vessels are not known.

Other difficulties that are potentially easier to overcome, but would still need to be addressed, are the question of how to represent the vessels and the related question of exactly what to measure once a registration is achieved. Possibilities for the latter include the presence of pre-specified "important" vessels, the lengths and widths of vessels, the number of small vessels, and overall surface area or volume agreement. An additional source of error is the scanning modality. Since the algorithm does not operate on the vessels directly, but rather on an image acquired by a scanner, this transformation from anatomy to image should be modeled. Some vessels will not appear in the image, while phase wrapping due to velocity encoding (for the case of PC-MRA) will cause others to appear in the wrong geometric locations and at the wrong intensities.

Another area of discussion is the question of vascular malformations. These segmentation algorithms will often be applied to pathological cases. In some cases the goal will be to detect and perform calculations on the particular pathology, such as an arteriovenous malfunction (AVM) in which an artery and a vein connect and thus short-circuit the blood flow or an aneurysm in which a portion of an artery has a potentially large balloon-like bulge. These pathologies, however, may confound the registration algorithm and also the atlas-based comparison.

Since we reject the atlas validation strategy at present for the reasons above, we consider other possibilities for validation. The first is to have an expert radiologist, neurosurgeon, or neuroradiologist (for the case of the cerebral vessels) manually label each voxel in a dataset "vessel" or "not vessel". The most obvious drawback to this option is the large amount of time required for this task. Typical MRA datasets acquired clinically at Brigham and Womens Hospital are $256 \times 256 \times 60$ voxels, and one may prefer to use super-resolution datasets which would be even larger to obtain more precise validation. The time required for an expert to carefully segment such a dataset is on the order of a few hours. This is not feasible for testing on a large number of cases. Other drawbacks to this approach include the variability for a common dataset of the same expert's segmentation over multiple trials and the variability of segmentations by multiple experts, both of which are significant.

Another method that has been used in the research community is the use of a vascular phantom for which ground truth vessel locations are know. R.G. Shelley Lt., North York, Ontario is a company which manufactures and sells a variety of vascular phantoms (`http://www.simutec.com`, [39, 85, 8, 50]). Such phantoms contain one or more tubes made of material designed to mimic vessel walls. "Blood-mimicking fluid" can then be pumped through these tubes while they are imaged in a magnetic resonance scanner, to obtain a volumetric image similar to that of actual blood vessels. Frangi *et. al.* used a Shelley phantom in recent vessel segmentation studies [41]. One limitation is that such phantoms do not contain the extra-vascular tissues that interfere with the imaging process in real anatomy. The phantom image is thus often "cleaner" than is realistic which makes it easier to segment. Shelley has used of bags filled with saline solution surrounding the tubes to simulate this effect. The fundamental limitation of this validation strategy even with an excellent phantom, however, is that it does not measure the algorithm's performance on the actual datasets on which it is to be applied, but only on necessarily simpler structures.

Also used in the literature has been the visual comparison of segmentations of volumetric MRA scans with digital subtraction angiography (DSA) images [15, 97]. Also called *x-ray angiograms*, DSA images are 2D images that show projections of 3D vascular structure. They are used because they exhibit a much higher level of detail than do MRA images. They are more invasive than MR imaging since a significant level of contrast agent must be injected into the subject, so one would not use them in healthy subjects for validation experiments only. However, there are corresponding MR and DSA images for a large number of patients, so availability is not problematic. The approach used in [15] is to register the MRA image to the DSA image, project the centerlines and branching points of the segmentation of the MRA image onto the DSA

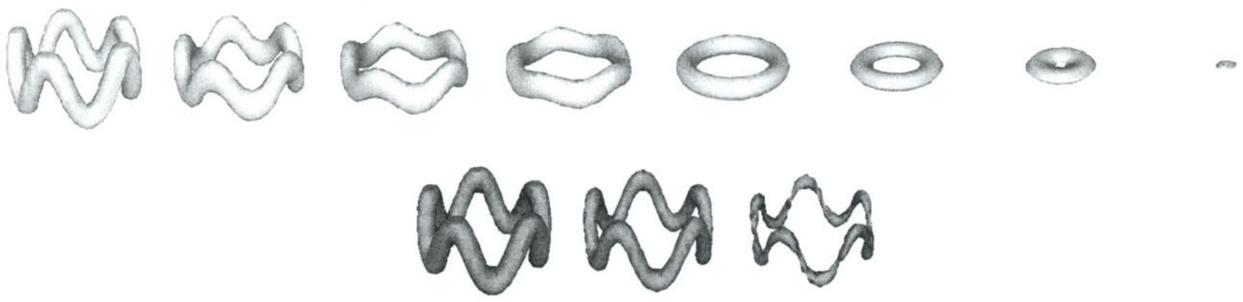


Figure 6-2: Top row demonstrates the tubular object evolving to smooth the underlying curve, as in CURVES. Notice that the bumps are first smoothed out until the shape approximates a torus, then the torus shrinks to a point. Bottom row demonstrates the shape evolving under codimension-one flow. The high curvatures corresponding to the small radius of the tube cause the shape to become thinner until it disappears without perceptibly affecting the geometry of the underlying curve.

image, then ask an expert to verify that the branching locations are indeed correctly localized. This strategy does adequately quantify the degree to which branches are correctly detected, but there are many other attributes one wishes to validate, such as the presence of very thin vessels and the vessel widths. Further, it is dependent on the quality of the registration between the two datasets and on the quality of the DSA image and is restricted to arteries since veins do not appear in DSA images.

We propose another strategy for ground truth determination to overcome some of these problems. We acquired two scans of a single subject at different resolutions. The scans were acquired without moving the patient so they are registered by default. The idea is to then use a segmentation of the higher resolution dataset as ground truth by which to evaluate segmentations of the lower resolution dataset. This strategy is discussed in more detail in section 6.6 below.

We also ran compared to segmentations acquired in a different manner. This manner involves an interactive MRA segmentation tool currently used clinically at Brigham and Womens Hospital, considered "state of the art" for vessel segmentations in clinical use. This tool enables the expert to generate segmentations in much less time than would be required by a fully manual method, but the segmentations generated are not completely reflective of the true vasculature. This protocol is described in detail in section 6.6.2 where comparison results obtained with it are shown. In particular, we compared our segmentations to those obtained by a neurosurgeon with this tool for approximately 10 datasets.

The following sections describe experiments on simple tube-like structures that were used in early stages of development, then other simulation results. Following the simulation discussion, the experiments on medical datasets are provided.

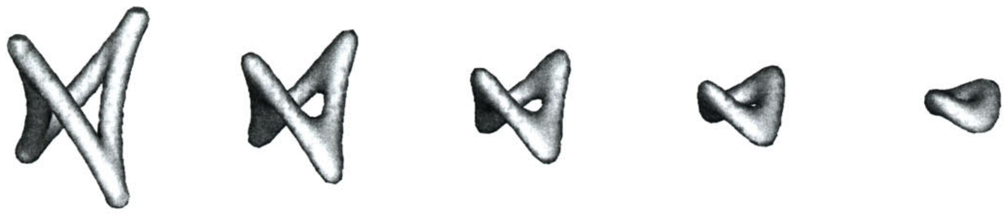


Figure 6-3: A tubular shape with corners under codimension-two flow.

6.2 Simulation

We constructed many toy shapes to understand the behavior of the codimension-two evolution and the image force. We first show regularization only applied to synthetic tubular shapes. We then show a toy shape evolving by both regularization and image-attraction. Finally, we discuss the trade-off between image weighting and the frequency of reinitializing the evolving volume to be a true distance function, using a synthetic example for illustration.

6.2.1 Tubular Objects Under Codimension-Two Flow

We generated a tubular shape whose centerline is a sinusoidal function along a circle. The first row of Figure 6-2 demonstrates the behavior of this shape undergoing smoothing based on the curvature of its centerline. This regularization is the "codimension-two" force we use in CURVES since we want shapes to behave like their centerlines where regularization is concerned. These images were produced by using CURVES with the image term set to zero. Notice the bumps are first smoothed out until the shape approximates a torus, then the torus shrinks to a point. This is the behavior of the circular 1D centerline would exhibit under curve-shortening flow. The second row shows the behavior of traditional (codimension-one) mean curvature flow in which the regularization is based on the mean curvature of the surface. In this case, the high curvatures corresponding to the small radius of the tube cause the shape to become thinner until it disappears without perceptibly affecting the geometry of the underlying curve. We emphasize that our novel regularization force is one of the primary differences between our work and previous surface evolution segmentation methods which use the mean curvature force [18].

Figure 6-3 shows another tubular shape undergoing codimension-two flow. Again, the figures were produced by evolving the shape in CURVES with the image term set to zero, so only the regularization force was used. This shape has sharp corners, and we see that although the 1D centerline of the shape is singular at those corners, the codimension-two force simply uses the smaller principal curvature of the surface at these points. This procedure has the advantage of enabling the evolution to proceed in a natural way to smooth out these corners and the disadvantage of causing the tube to become fatter at these corners over the course of the evolution. The fattening is

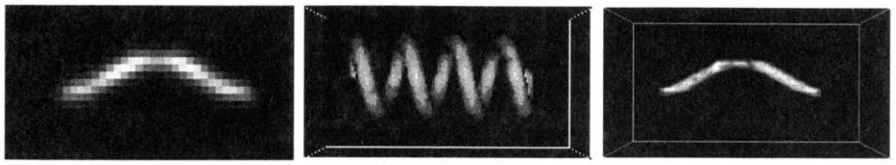


Figure 6-4: Evolving helix under mean curvature flow with additional vector field: target curve, initial level set, level set after evolution with endpoints constrained.

caused by the different curvature estimates obtained on the inside and outside of the corners. This disadvantage exists for all tubular shapes, but is especially prominent for corners where the interior curvature estimate is very high. As expected, the tube shrinks according to its underlying centerline, modulo this non-constant change in width.

6.2.2 Incorporation of Image Data

We illustrate our early image-force experiments with the evolution of another tubular shape. We wanted to evolve a helical tube into a cosine-shape tube. Recall that a helix shrinks to its centerline under usual curve-shortening flow. In this case, however, we are using the full image-based CURVES evolution equation with an underlying image of a bright cosine curve, so we would expect the helix to evolve smoothly into this cosine shape. Indeed, this is the result, as pictured in Figure 6-4. The underlying volumetric image data is shown first as a maximum intensity projection. This volume was generated by drawing a cosine curve in the volume, then smoothing with a Gaussian filter. We chose to evolve an initial curve which was very far from the target, such as the helix (second image), instead of using thresholding to obtain the initial curve, which would give a close guess. The result of the evolution is shown in the rightmost image. For this experiment, we made a modification to the algorithm. Because the tube is open, it has endpoints at which both principal curvatures are large, and neither of those curvatures corresponds to the geometry of the centerline. Thus, it is inappropriate to use the specified regularization force at these points. We modified the algorithm accordingly to perform at check at each point based on the ratio of the principal curvatures to determine which points were endpoints, at which no regularization would be performed; that is, the regularization force was set to zero. This modification did not cause the points to remain motionless because of the image force, the motion of neighboring points, and the smoothness of the distance function, but it did prevent high curvatures from being used incorrectly for regularization. In our later experiments with MRA data after default parameter setting and initialization procedures were in place, we found this modification to be unnecessary since the image weighting and the gradient-directionality term are strong enough to counteract the inappropriate regularization. This modification is thus not used in CURVES.

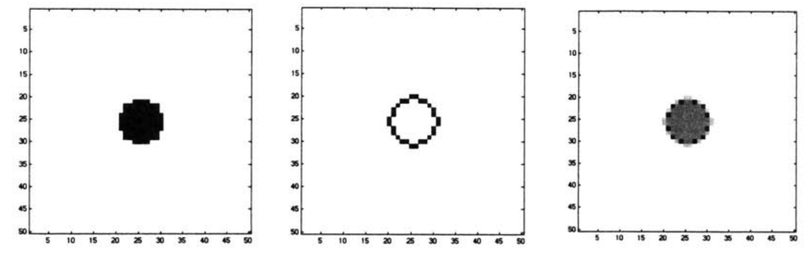


Figure 6-5: Simple segmentation example for which ground truth is known. The structure to be segmented was a cylinder. Cross-sectional slices are shown. The first image is the boundary computed analytically, to resolution precision. The second image is the boundary found by CURVES. The third image is the two overlaid.

A different experiment used thresholding for initialization instead of a pre-specified shape. A dark image containing a bright cylinder of Gaussian intensity profile was created. A ground truth segmentation was defined to be all voxels inside the tube defined by the largest intensity gradient magnitudes. This ground truth was computed analytically. The CURVES segmentation using default parameter settings, including thresholding to obtain the initial boundary estimate, did indeed find this segmentation to within rasterization error, as demonstrated by the images of cross-sectional slices in Figure 6-5. The first image is the boundary computed analytically, to resolution precision. The second image is the CURVES segmentation. The third image is the two overlaid.

The Gaussian intensity profile is a common and adequate approximation to the intensity profile of vessels under MRA imaging. This profile, combined with the feature of our algorithm of convergence on maximal gradient magnitudes implies that the detected boundaries can be inside the true vessel boundaries since the gradient maxima are generally inside the vessel walls in this imaging modality. However, for modalities such as CT that do not have this profile, gradient magnitude does not have this difficulty. The first row of Figure 6-6 shows the CURVES segmentation of a Gaussian profile cylinder from a lengthwise slice. Notice that even when initialized (by thresholding) outside of the bright vessel-like structure, CURVES converges to the sharpest gradients. The second row shows an example with a flatter profile constructed by smoothing a step edge. In this case, the sharpest gradients better approximate the object boundary. As we will see in real MRA experiments below, this "thinning" effect is problematic for the wider vessels whose intensities are brightest on the centerlines and diminish toward the vessel walls with this profile. The small vessels' profiles only show one or two voxels brighter than the background (the diameter of the vessel), so this troublesome profile is prevented by the image resolution. We expect that this problem could be addressed by extremizing measures other than image gradients but have not explored them.

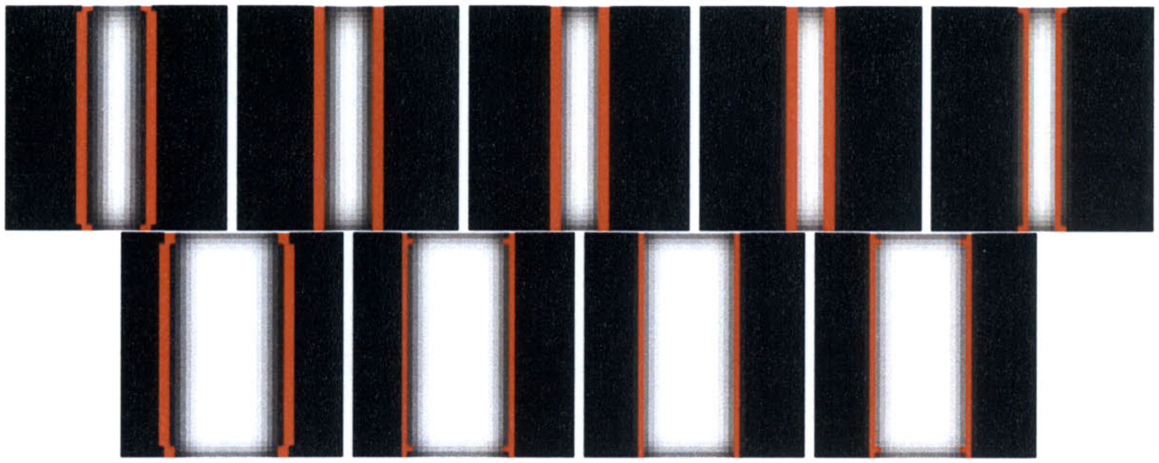


Figure 6-6: First row: Gaussian intensity profile. Second row: smoothed step profile. In both cases, the current boundary estimate is overlaid in red and the initialization boundary is shown first followed by the boundaries after several steps until convergence.

6.2.3 Reinitialization and Image Weighting

The observed behavior on clinical datasets as well as on toy examples was that one could reinitialize much less frequently if the image term had relatively low weight. This is intuitive since the image term destroys the distance function, and the degree to which v is not a distance function is monotonically related to the weighting of this image term. Further, when v becomes more irregular, the evolution becomes noisy and is not well behaved; one must reinitialize to maintain smooth behavior. It is desirable for efficiency reasons to reinitialize as infrequently as possible since the procedure is computationally expensive. However, a low image weight may slow the evolution or cause faint structures to be missed entirely, thus a high image weight is also often desirable. We constructed a simulation experiment to enable better understanding of the trade-off between infrequent reinitialization and increased image weighting.

Using the same synthetic cylinder as in Figure 6-5, we ran the evolution with default parameters, using thresholding for initialization, to determine the valid range of trade-offs for the image weighting parameter ρ versus the frequency of reinitialization of the distance function. Figure 6-7 illustrates the structure, ground truth, and CURVES' performance. The upper left image is a cross-section of the cylinder, showing the Gaussian intensity profile. The upper right image is the gradient magnitude image, and the lower left is the set of points with highest gradient magnitude which are the "ground truth" boundary points. With adequate settings for image weighting and reinitialization frequency, CURVES indeed converges to the true segmentation, as illustrated in the lower right image in which the CURVES segmentation is overlaid on the ground truth boundary points. Note that a forced reinitialization always occurs when the zero level set gets too close to the border of the narrow

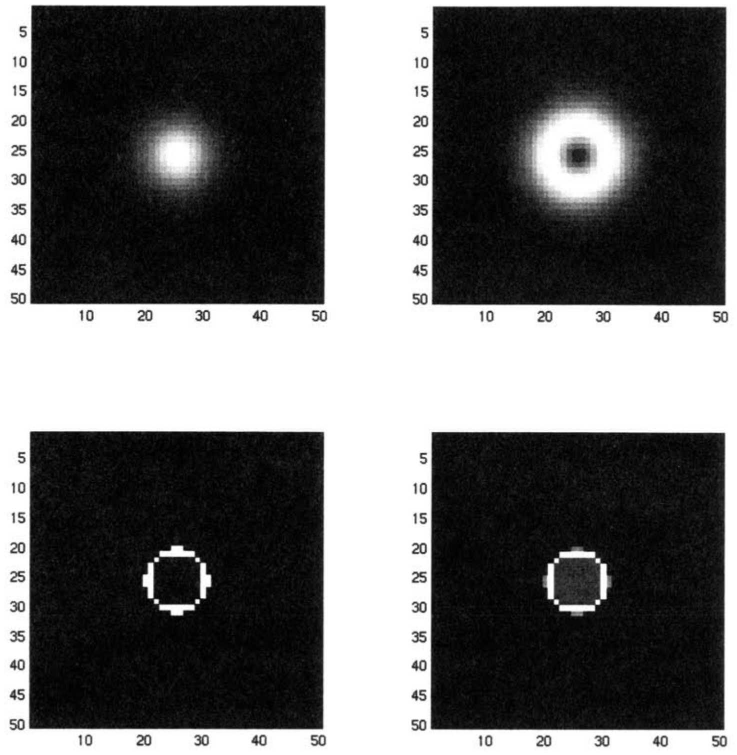


Figure 6-7: Simple segmentation example for which ground truth is known. The structure to be segmented was a cylinder. Cross-sectional slices are shown. The first image is the Gaussian intensity profile of the cylinder, the second is the gradient magnitudes, the third are those points with highest gradient magnitude, and finally those points overlaid on the CURVES segmentation, showing that CURVES did converge to the correct segmentation.

band, and the parameter controlling reinitialization frequency is in addition to this forced reinitialization. Figure 6-8 plots the largest number of steps permitted between parameter-reinitializations for which the answer found was correct versus the image scaling term. A larger value of this term indicates more image weight, proportionally. The figure shows that for an image term less that one, one can reinitialize infrequently, but for an image weight over 2.5, one must reinitialize every step. We would interpret this to mean that surface is moving too much at each step, so one should either decrease the coefficients such as ρ or the step size. Note that these measurements are dependent on the intensity properties of the image, so and image with a different distribution of gradients would give different results. Thus, one can regard this graph as showing an example of the shape of this trade-off, but the numbers will vary for different types of datasets.

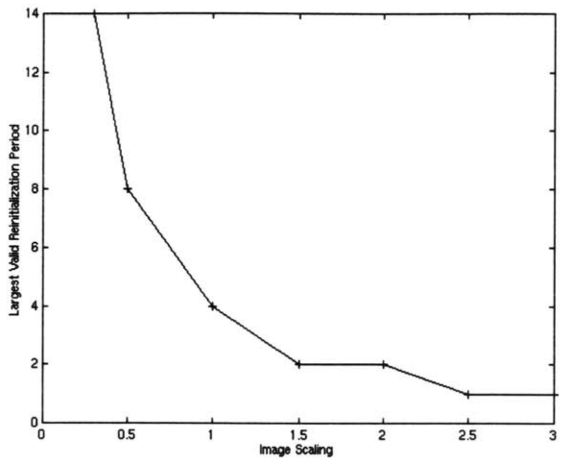


Figure 6-8: The largest number of steps permitted between parameter-reinitializations for which the answer found was correct versus the image scaling term. The experiment was performed for the data pictured in Figure 6-7.

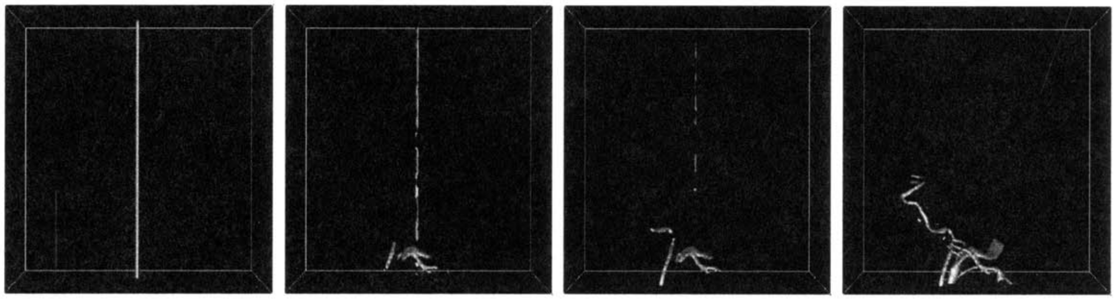


Figure 6-9: Illustration of a vertical bar evolving to segment vasculature.

6.3 Medical Evolution Sequences

We next move to medical datasets. The datasets we normally operate on are PC-MRA, as described in Section 4.1, and a typical dataset size in $256 \times 256 \times 60$ voxels. Image size, imaging artifacts, and complexity of the anatomical structures make these experiments more difficult than the synthetic shapes above. Before showing final segmentations and validation comparisons for various types of datasets, we illustrate the evolution over time of the boundary estimate for these types of datasets. These sequences are shown to give the reader an understanding of the behavior and run-time performance of the system before analyzing the final segmentations. As stated, we have two options for how to initialize the segmentations: a pre-specified shape or one obtained by thresholding the input dataset. We show two sequences, one obtained with each method. Figure 6-9 shows the evolution of a pre-specified shape, a vertical bar: the initial bar is shown, followed by subsequent steps in the evolution for the segmentation of an MRA dataset. Notice that the image attraction force can pull the

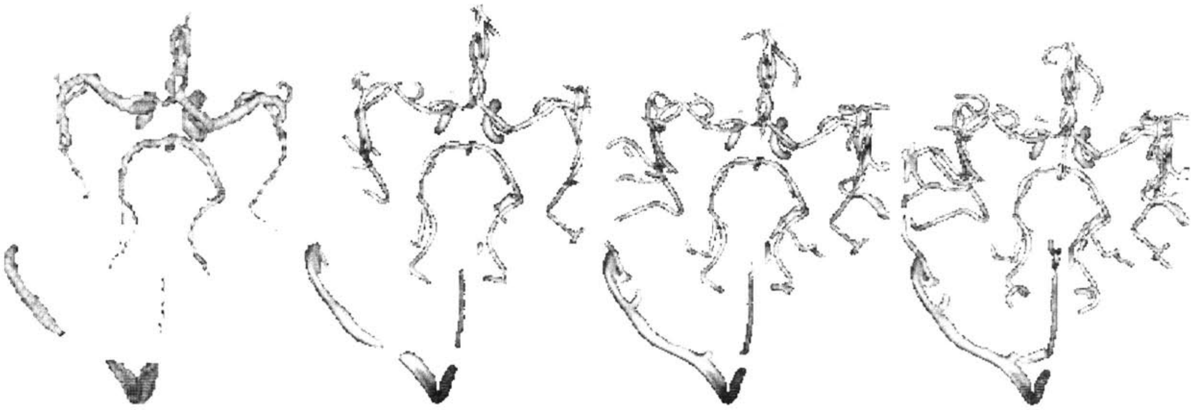


Figure 6-10: Surface evolution over time: initialization, followed by successive boundary estimates.

surface toward true vessels even when the starting point is not close to the vessels. In practice, however, it is beneficial to start with the better estimates that are available by thresholding. Figure 6-10 illustrates the system's behavior when initialized by thresholding the dataset. This dataset was also a cerebral MRA scan and the surface models are shown from an axial viewpoint. The initial surface is obtained by thresholding the raw dataset, then CURVES evolution produces the subsequent images. Notice how branching occurs automatically with either initialization. Again, in practice the thresholding initialization approach is better for both speed and performance, so it is the default in our system and was used in the remainder of the experiments described in this chapter.

6.4 Aorta

Figure 6-11 shows the segmentation of a contrast-enhanced MRA image of an aorta courtesy of Siemens. This dataset was acquired on a Siemens scanner at New York University. Voxel dimensions are $1.75 \times 1.75.39\text{mm}^3$ and a contrast agent was used. The image size of the volume shown is $110 \times 256 \times 46$ voxels. The segmentation is shown from two orthogonal viewpoints. For each viewpoint, the maximum intensity projection of the raw data is shown first, followed by the CURVES segmentation. Notice that CURVES is able to obtain many thin vessels, which are the most difficult to segment.

6.5 Lung CT, Comparison to Mean Curvature

For comparison purposes, we have created a version of the CURVES program which uses the codimension-one regularization force, the mean curvature of the surface,

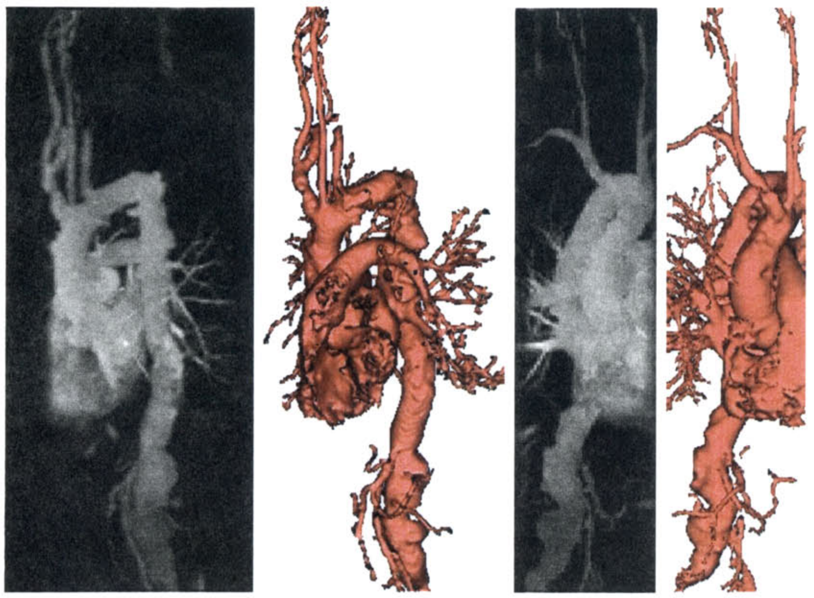


Figure 6-11: Segmentation of a contrast-enhanced aorta MRA image, courtesy of Siemens, acquired on a Siemens scanner at New York University. From each viewpoint is shown the maximum intensity projection of the raw data and the CURVES segmentation.

as in previous level set segmentation schemes [18]; otherwise, all parameter settings were identical to those used in the CURVES experiment. That is, this version of CURVES is not the minimal surfaces algorithm because CURVES uses a different image force that is obtained locally in the image and that is also weighted by a gradient-directionality term, but it differs from CURVES only in the regularization force.

Experiments with this version have shown that the magnitude of the effect of the modification on the resulting segmentation depends on the trade-off chosen between the regularization term and the image term in the underlying evolution equation. (This trade-off is specified by ρ in Equation 5.13.) When this trade-off is chosen so that the image information is weighted much more heavily than the smoothing term, the effect is negligible. This was the case for the highest quality MRA image we had, such as shown in Section 6.6.2. However, for CT data and lower quality MRA images, the effect can be significant.

Figure 6-12 shows the CURVES segmentation of bronchi in a lung CT dataset compared to the codimension-one segmentation, for the same parameter settings. The first image is the maximum intensity projection of a subregion of the dataset in which the bronchial tubes appear bright. They appear bright in CT because blood vessels cause a bright signal in CT and they run along the bronchi. The second and third images are the CURVES and codimension-one segmentations respectively, and the fourth is a combination image of the two segmentations. Notice that the codimension-

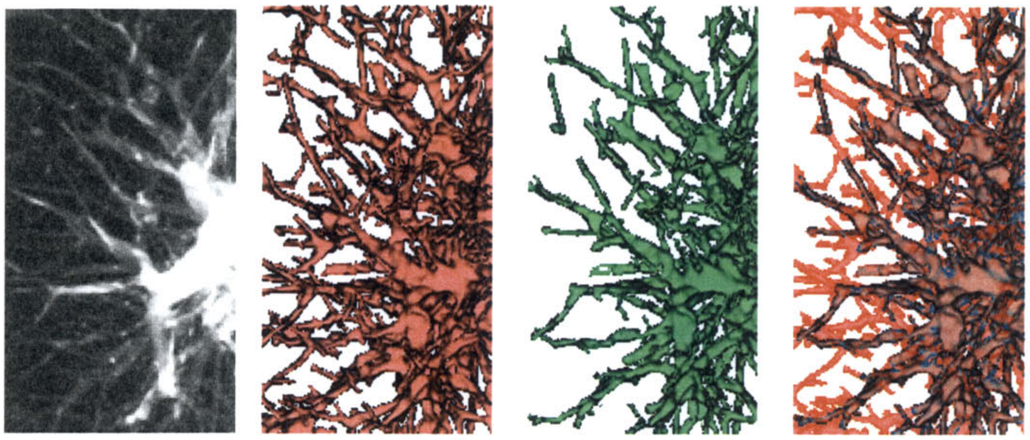


Figure 6-12: Segmentation of bronchi in volumetric CT image of lung. Maximum intensity project of raw data is shown, followed by CURVES segmentation (red) and codimension-one segmentation (green). Final image is combination image of the two segmentation images, showing that CURVES method captures more fine detail of the bronchial structure.

two regularization force in CURVES does indeed allow the segmentation of more thin structures than does the codimension-one force. This effect is intuitive because the codimension-one algorithm incorporates a smoothness constraint which acts to prevent high curvatures anywhere on the resultant surface, which is inappropriate for the segmentation of thin tubular structures which must have high curvatures corresponding to their small radii. Conversely, the regularization force in CURVES allows this high curvature, regularizing against only the curvature of the underlying one-dimensional curve.

6.6 Cerebral Vessel Validation

One would like a quantitative measure of CURVES performance in comparison to some baseline or to other methods or as an absolute score of performance. For this reason, we developed a multi-resolution validation strategy in which a pair of registered MRA images are acquired at multiple scales and a segmentation of the higher resolution scan is used as ground truth for the experiments on the lower resolution scan.

We thus acquired several multi-resolution cerebral MRA images, some with a .5T magnet and some with a 1.5T magnet. We selected an area containing the Circle of Willis because of its connections to important vessels. The Circle of Willis is a vascular structure which loops around the brainstem and connects to the anterior, middle and posterior cerebral arteries, the primary arteries supplying the brain. This situation has the unusual benefit that damage to any one of these three arteries will not compromise blood flow to the brain; moreover, blood can flow in either direction in the circle.

In the .5T magnet, we acquired three pairs of images of this region of healthy subjects. A velocity encoding (venc) of 60cm/sec was used. Slice thicknesses of 1.7mm and 0.8mm were obtained for the low and high resolution scans respectively. A 256x256 pixel image was acquired for each slice in the high resolution scan, and a 256x128 pixel image was acquired for each slice in the low resolution scan. The low resolution slices were interpolated to 256x256 but the true resolution was 256x128. 28 slices were acquired for the low resolution scan and 60 for the high resolution scan. The same venc, resolutions, and slice thicknesses were used for the 1.5T scans, of which we acquired for two pairs of healthy subjects, except the field of view was increased to obtain 40 slices and 84 slices for the low and high resolution scans respectively.

A concern of ours in using this strategy is that although the higher resolution dataset can presumably show more detail, each voxel represents a smaller number of hydrogen atoms, so is less likely to give a reliable reading. Empirically, however, we found that the high resolution scans did indeed appear much better than th low resolution scans for the 1.5T images and that the pairs of scans looked very similar for the .5T images.

Besides our multiresolution validation experiments, it was these datasets that a neurosurgeon segmented manually for us with the use of an interactive computer program that works as follows. The neurosurgeon interactively chooses a threshold that is used to binarize the MRA dataset: all voxels brighter than that threshold are labeled as vessel, while all others are discarded. A "connectivity" program then partitions the set of labeled voxels into connected components. Each connected component appears in a distinct color on the user interface. The surgeon looks at individual slices and clicks on colored regions that correspond to vasculature. All connected components so chosen are stored as the final manual segmentation. The surgeon may choose to manually edit the segmentations depending on the errors present and the requirements of the clinical situation. Note that segmentations obtained with this process can often contain image artifacts and miss small vessels since the process is based on thresholding which has these difficulties. We will discuss the manual segmentations and our comparisons to them following the discussion of the multiresolution experiments.

6.6.1 Multiresolution Validation

There are several parameters of the multiresolution validation method that need to be specified. The first is how to construct the "ground truth" segmentation from the higher resolution scan. Possibilities include CURVES and the expert interactive method described in section 6.6.2; we will show experiments with both choices. Second is what to measure about the the segmentations of the lower resolution scan: we chose the scoring protocol used by Sato in [97] in which ROC-like curves are plotted.

Receiver operating characteristic (ROC) curves show performance of detection systems for which the only free parameter is a single decision threshold on some quantity related to the probability of the signal being correct. Such systems have the property that one can always choose a threshold high enough so that the probability of both false positives and true positives is zero; conversely one can always choose a threshold below the valid range so that the probability of false positives and true positives is one. So, each such algorithm sweeps out a curve over the (p_f, p_t) space, where p_f is the probability of false positives and p_t is the probability of true positives. $(0,0)$ and $(1,1)$ are contained in the curves for all algorithms. The ideal case is $p_f = 0$ and $(p_t) = 1$, so to compare different algorithms, one sees which yields a curve that comes closest to the $(0,1)$ location in this space. For our experiments, we wish to compare the performance of CURVES to that of simple thresholding on the raw data. Our needs do not fit the requirements for ROC curves in two aspects, so we will generate similar "ROC-like" plots instead.

The first aspect in which our situation differs from standard ROC analyses is that we obtain a complicated segmentation for each run of the algorithm, instead of a single "yes" or "no" response. For each segmentation, we compute a *true positive fraction* T_p and a *false positive fraction* F_p as defined by Sato *et al.* [97]. In parts of this analysis, we use *skeleton* points of the vessels instead of the entire segmentations for robustness of the comparisons. The skeletonization algorithm used, however, returns a very large number of points so the "skeleton" is often thicker than the true centerlines. In fact there was approximately a 3:1 ratio between the number of segmentation points and the number of skeleton points. The skeletonization algorithm used is based on the removal of *simple points* from the binary segmentation [11, 76]. This thinning algorithm will be discussed in Section 7.5.2. Let N_{td} be the number of ground truth vessel skeleton points that are detected, N_t the total number of ground truth vessel skeleton points, N_{fd} the number of detected points not included in the ground truth segmentation, and N_d the total number of detected points. Then,

$$T_p = \frac{N_{td}}{N_t}, \qquad F_p = \frac{N_{fd}}{N_d}.$$

This definition can be stated equivalently as follows. If G is the ground truth segmentation, G_s the skeleton of the ground truth segmentation, A the automatic segmentation to be evaluated, and A_s the skeleton of the automatic segmentation to be evaluated, then

$$T_p = \frac{|G_s \cap A|}{|G_s|}, \qquad F_p = \frac{|\bar{G} \cap A|}{|A|},$$

where $\bar{G}$ denotes those points not in G and $|S|$ denotes the cardinality of some set S.

In the plots of thresholding, each point on the curve corresponds to a different threshold. The CURVES curves, however, exhibit the second aspect in which our situation differs from standard ROC analyses. There is not a "decision threshold" in the CURVES algorithm. Instead, the most meaningful parameter to vary is ρ, the relative weighting between image fidelity and surface smoothness. Each is point on a CURVES ROC-like curve then is the (T_f, T_p) value for a differ choice of ρ. This is a more important difference from usual ROC curves than just definition. It means that CURVES will not produce ROC curves of the usual shape; for example, they will generally not contain the points $(0, 0)$ and $(1, 1)$. They also are not guaranteed to be monotonically increasing in T_p and T_f as usual plots are. We would expect an increase in ρ to increase T_p and T_f since higher image weight implies less smoothing and smoothing is related to shrinking the size of the surface model. Empirically, we have found that increasing ρ normally causes larger segmentations. However, since the image term depends on the local image information which will vary as the segmentation evolves into different image regions, we cannot expect this increase in size to be proportional to ρ or even monotonic, it is only the expected case.

One of the segmentations of a 1.5T dataset used in this experiment is pictured in Figure 6-10. More images of the segmentations and raw datasets used will be shown in Section 6.6.2. Figures 6-13 and 6-14 show ROC-like plots for two different experiments. The same dataset, a low resolution 1.5T scan (Figure 6-16 with corresponding high resolution scan in Figure 6-15 which shows the manual segmentation used as ground truth), is used in both cases. Segmentations of it are computed with CURVES for various values of ρ and by thresholding for various thresholds. The difference is how the ground truth segmentation is computed. Figure 6-13 shows values for when a CURVES segmentation of the corresponding high resolution scan is used as ground truth, and Figure 6-14 shows values for when an expert's "manual" segmentation is used as ground truth, where the "manual" procedure is the interactive thresholding-based procedure described above.

These plots demonstrated that this experiment was extremely biased toward the choice of ground truth. If the higher resolution manual segmentations, which contain a thresholding step, were used, then the thresholding method appears superior to the CURVES method. However, when the higher resolution CURVES segmentations are used, then CURVES appears much better. Thus, one cannot use this set of experiments to make conclusive statements about the advantages of the CURVES method since an unbiased ground truth estimate was not available. Two observations, however, can be made. The first is that there is consistency between how CURVES (and thresholding) behaves on images take at higher and lower scan resolutions. One would expect this behavior, since the MRA datasets have a similar appearance in terms of intensities and image gradients. The second is that there is symmetry between the CURVES case and the manual/thresholding case in that each option gave a more desirable ROC-like curve when it was chosen as the ground truth, with the benefit appearing stronger in the CURVES case than the thresholding case. Due to the limited nature of the conclusions we can draw from this experiment, we chose to compare segmentations directly against manual segmentations. The following section describes and show these comparisons.

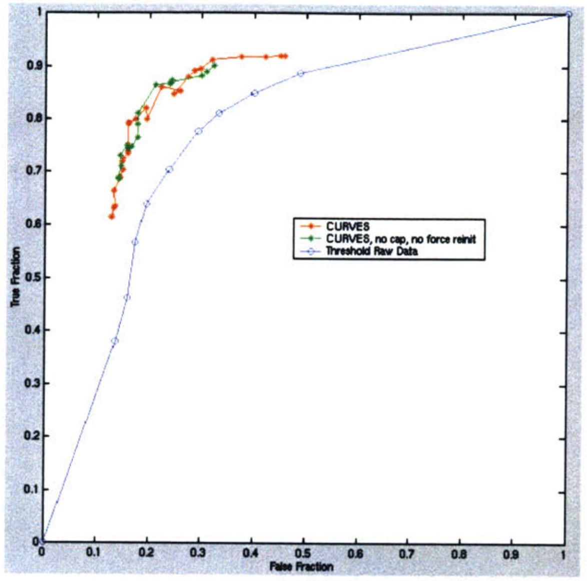


Figure 6-13: Ground truth is CURVES segmentation of high resolution scan. Scan on which experiments were run is corresponding low resolution scan. Green and red show two variations of the CURVES algorithm. Notice that when this particular "ground truth" is used, CURVES appears to significantly outperform thresholding (blue).

6.6.2 Manual Validation

One specific practical motivation for our work is the use of surface models of cerebral vasculature as an aid in neurosurgical planning and procedure, especially in the context of the image-guided surgery program at our institution [47]. Currently the vessel models are obtained with the thresholding-based manual procedure described above. The first drawback of this method is the expert user-interaction required, the second is that the thresholding step implies that all regions of image "noise" which adjoin vasculature are incorrectly labeled as vessel and small thin vessels which may appear broken or disconnected from larger structures will often be omitted. Thus, our goal is to reduce user interaction while increasing the ability to segment thin vessels.

A neurosurgeon has segmented our multi-resolution image image pairs for us using this procedure. We show comparisons for eight such datasets of both high and low image resolution and of both high (1.5T) and low (0.5T) magnet strength. All datasets are PC-MRA on healthy subjects without contrast agent. In all cases, each image slice was 256×256 voxels, but the number of slices varied across the datasets. The image size of 256×256 was cropped to approximately 110×125 to remove empty borders before processing for efficiency. In all cases, all slices were used; that is, the z dimension was not cropped. The voxel x and y dimensions were 1.171875mm, and but the z dimension (slice thickness) varied. Each of the following figures shows a single dataset from three orthogonal viewpoints. For each viewpoint in each figure, the maximum intensity image of the raw data is shown first, followed by the CURVES

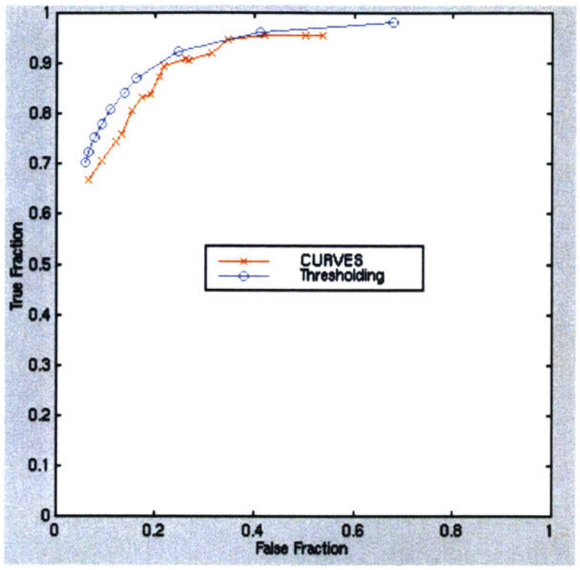


Figure 6-14: Ground truth is manual segmentation of high resolution scan. Scan on which experiments were run is corresponding low resolution scan. Notice that now simple thresholding appears to outperform CURVES since the "manual" method of segmentation is closely related to thresholding.

segmentation in red, the manual segmentation in blue, and a combination image illustrating the differences between the segmentations. In all cases the voxel-based manual segmentations are displayed as surfaces which were smoothed for better appearance. The smoothing parameters used are the same that are used as a default in surgical planning cases at our institution.

The first comparison is shown in Figure 6-15. This dataset was acquired with a 1.5T magnet and has 84 slices each of 0.8mm thickness. Notice that CURVES is able to capture much more of the thin vessels than is the manual procedure which is based on simple thresholding. One negative aspect of CURVES performance on this example is that some large vessels such as the middle cerebral arteries and the superior sagittal sinus appear too thin in the CURVES segmentation. This artifact occurs because CURVES places the vessel boundary at the location of sharpest intensity gradient. If the vessel intensity profile in MRA is assumed to be Gaussian, then the true vessel wall should be placed farther out along the tails of the Gaussian than at the points of sharpest gradients. This problem does not occur in CT data below which does not have a Gaussian profile, and future work will explore the modification of the objective function dependent on the imaging modality used. Figure 6-16 shows the corresponding lower resolution scan, also acquired with a 1.5T magnet, but with 40 slices of 1.714284mm thickness.

Figure 6-17 shows another 1.5T dataset also containing 84 slices each of 0.8mm thickness. This dataset contains considerable pulsatile flow artifacts which appear as a bright horizontal area surrounding the middle cerebral arteries. For this example, the thresholding-based manual method must include much of this "noise" in order to also obtain the thin vessels; since CURVES depends on intensity gradients it is better able to distinguish those arteries from the surrounding region, without losing the small vessels. Figure 6-18 shows the corresponding lower resolution scan, also acquired with a 1.5T magnet, but with 40 slices of 1.714284mm thickness; notice the flow artifacts again present.

Figure 6-19 and Figure 6-20 show corresponding 0.5T datasets containing 60 and 28 slices of 0.8mm and 1.1714284mm thicknesses respectively. Observe that this lower magnet strength yields less detailed images. Again, notice that CURVES is able to extract additional detail at the thinnest vessels. Figure 6-21 and Figure 6-22 show another pair of corresponding 0.5T images of containing 60 and 28 slices of 0.8mm and 1.1714284mm thicknesses respectively. Notice in these four datasets that CURVES obtains more thin arteries which are most clearly visible in the centers of the images in the middle (sagittal) viewpoint.

6.7 Radii Estimation

One would like to make a variety of size-related measurements of the vascular surface models for diagnosis, change-detection as multiple scans are acquired for a single patient over time, and for study purposes. Total volume and surface area estimates are directly computable from the segmentations. Another important measurement is vessel width. This measurement is important in determining the progression of vascular malformations, in selecting blood flow patterns to be modified in surgical procedures, and in diagnosing and quantifying vascular disease.

With a direct triangulated surface model representation of a segmentation, vessel width would be difficult to estimate. Realistic data would not yield tubular structures smooth enough so that the vertices of the triangles could be used directly in the curvature estimate. Curvature is computed from second derivatives, and the irregular shapes or vessels would imply that the use of such points in in a second order computation would be so noisy as to be unusable. A more sophisticated approach would be needed, such as fitting cylindrical tubes to the vessels or applying significant smoothing to the surface model to increase the stability of the second derivative calculations.

By the nature of the CURVES algorithm, however, vessel width estimations are automatically available. Of the two surface curvatures, CURVES computes the two principal curvatures of the surface from the 3D distance function (via Equation 5.10 and Equation 5.11) and uses the smaller in the segmentation procedure, but the larger curvature can also be useful as it corresponds to the radii of the vessels. We color-code the segmentation result according to the larger curvature at each point on the surface in Figure 6-23a. In this image of a partial segmentation, the colorscale is continuous from blue to green to red, with blue indicating a radius of curvature $\leq$ 1mm and red indicating a radius of curvature $\geq$ 2mm. Intermediate curvatures are green. Notice that for a ribbon-like vessel, the flatter sides shows a large radius, and the sharply curved edges show a small radius. The curvatures output by our evolution have been smoothed by a 3x3x3 filter prior to coloring the surface. A color-coded segmentation of the aorta images from Figure 6-11 is shown in Figure 6-23b. We reiterate that cylinders are not fit globally, but only local curvature properties are used, so the color can vary between adjacent regions where the structure is not perfectly cylindrical.

To measure the accuracy of the radii estimated by this method, we constructed synthetic volumes. We constructed distance functions corresponding to tubes of varying radii. These distance functions were $40 \times 40 \times 40$ isotropic voxels, with the tube centered in the volume and extending the length of the volume, parallel to the y-axis. We loaded these distance functions into CURVES and computed the vessel radii according to the distance function as described above. That is Equations 5.10 and 5.11 were used for the Gaussian and mean curvature, then the larger principal curvature was computed from those curvatures. Its inverse was used as the computed vessel radii.

Since there is numerical difficulty with computing curvatures from discrete points, we experimented with various levels of smoothing of the larger principal curvatures. The larger principal curvatures are available for all isolevel surfaces, *i.e.*, for the entire volume, even though the only ones used in the validation and in the coloring pictures are those on the zero level set. Since they are defined for the whole volume, we can smooth this volume. We define one "iteration of smoothing" as convolving the volume once with a normalized [1 4 1] filter in each of the three dimensions.

Figure 6-24 shows the accuracy of the estimates for a variety of tube widths and a variety of levels of smoothing. Tube radii was varied from one to ten millimeters, assuming each voxel's dimension was 1mm^3. The identity plot is shown for comparison: it is the ideal situation. For each experimental case, the (larger principal) curvatures were computed for the entire tube, then they were averaged over the middle portion of the tube; the endpoints were excluded to control against any errors introduced by boundary conditions where the mean and Gaussian curvatures are not defined. Plots are shown for zero, three, and ten iterations of smoothing, for the various widths. For this experiment, some smoothing does increase the accuracy of the estimates, but too much smoothing decreases the accuracy. The correct level of smoothing also may be dependent on the tube width. In practice, several iterations of smoothing are applied before generating the color-coded segmentations of real datasets.

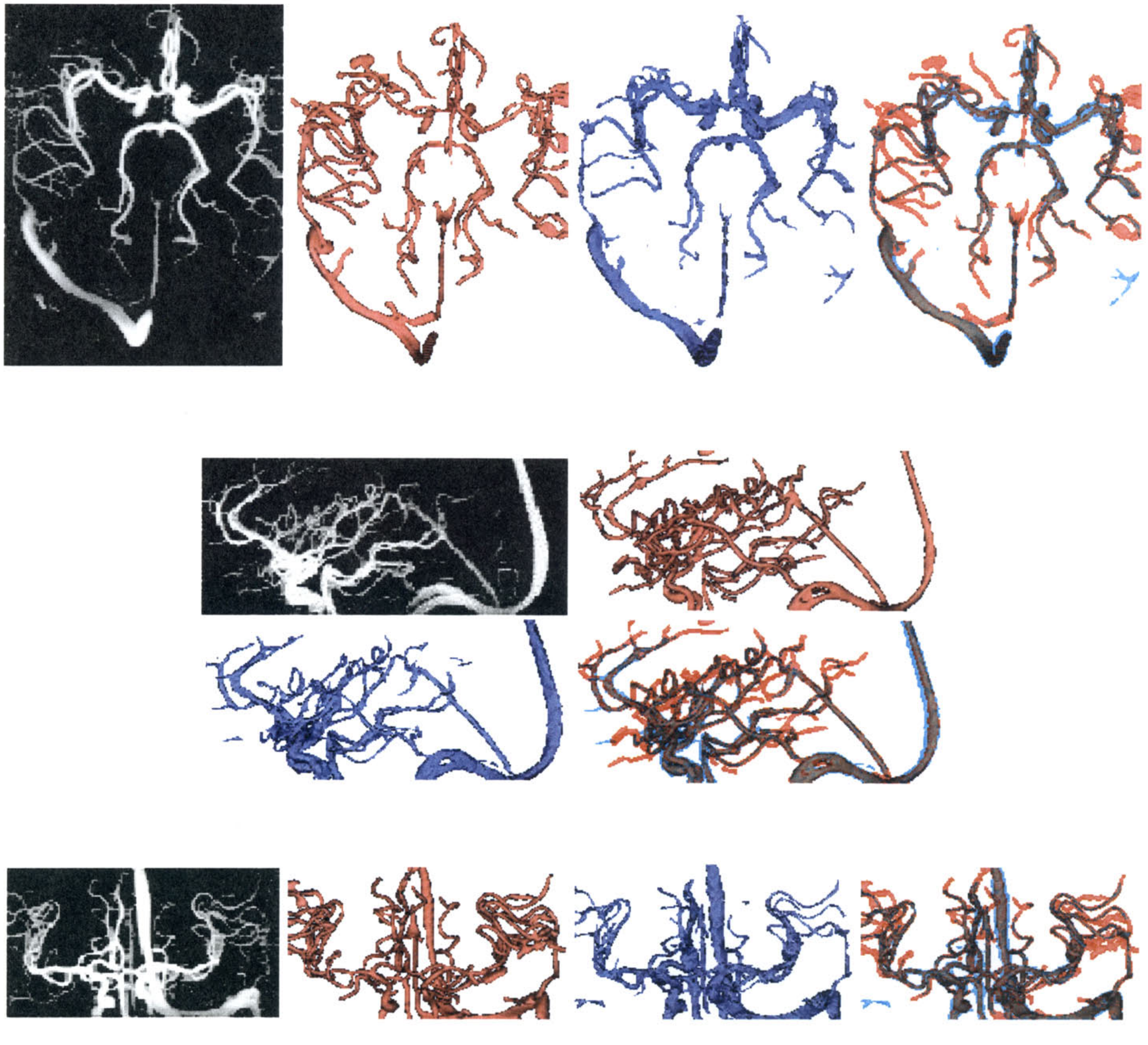


Figure 6-15: Dataset 1, Maximum intensity projection of raw data, CURVES segmentation (red), manual segmentation (blue), and combined image, from axial, sagittal, and coronal viewpoints.

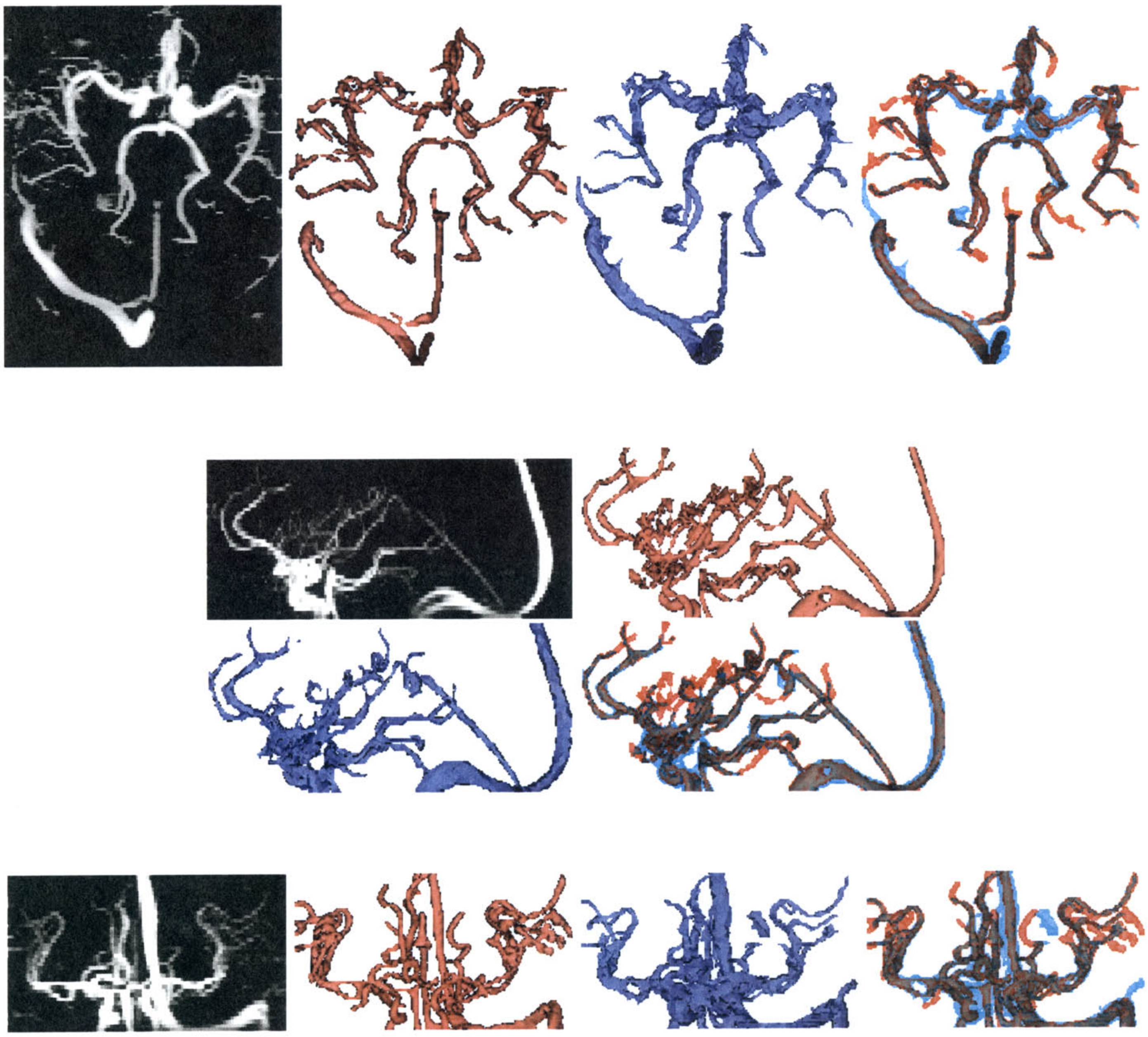


Figure 6-16: Dataset 2, Maximum intensity projection of raw data, CURVES segmentation (red), manual segmentation (blue), and combined image, from axial, sagittal, and coronal viewpoints.

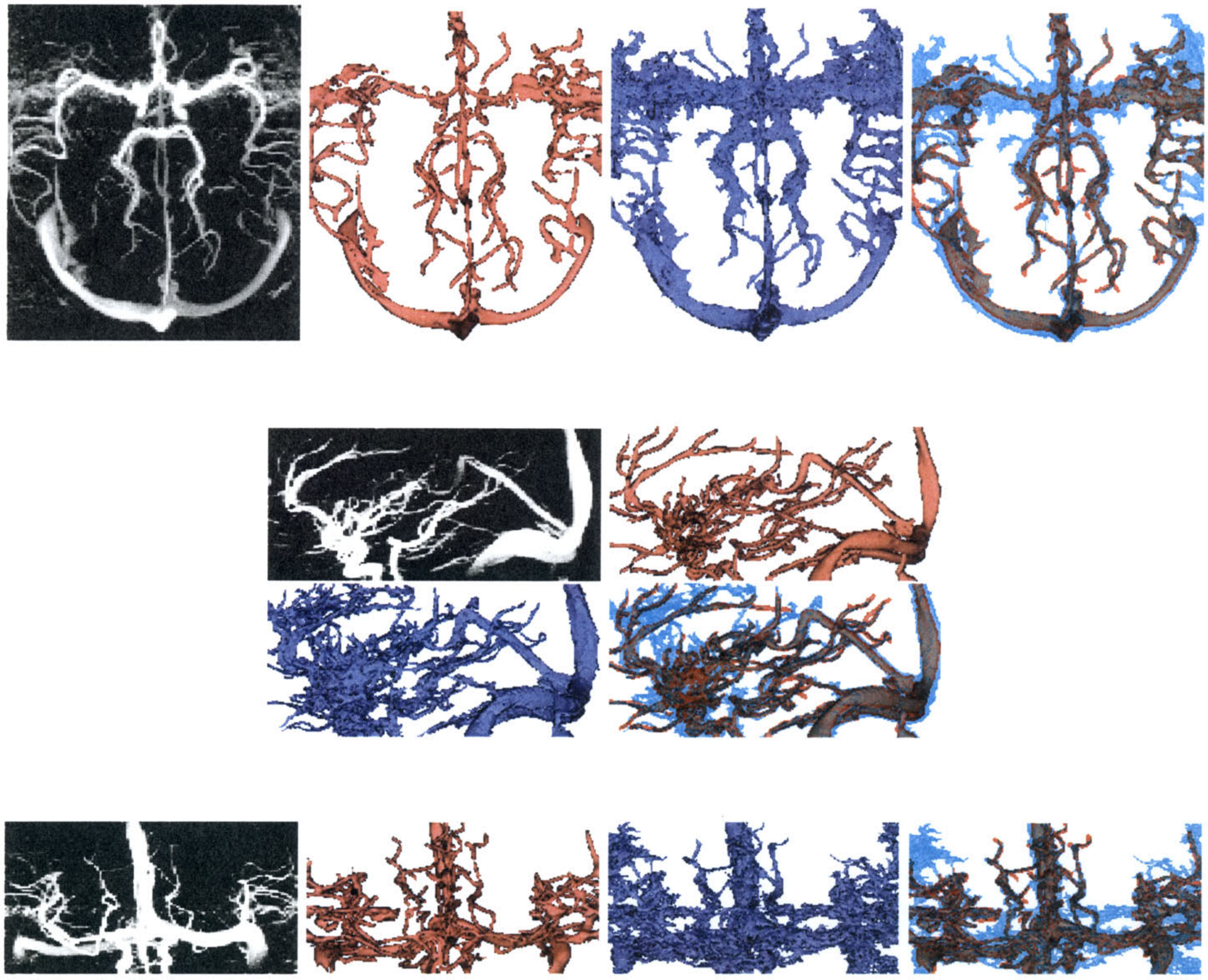


Figure 6-17: Dataset 3, Maximum intensity projection of raw data, CURVES segmentation (red), manual segmentation (blue), and combined image, from axial, sagittal, and coronal viewpoints.

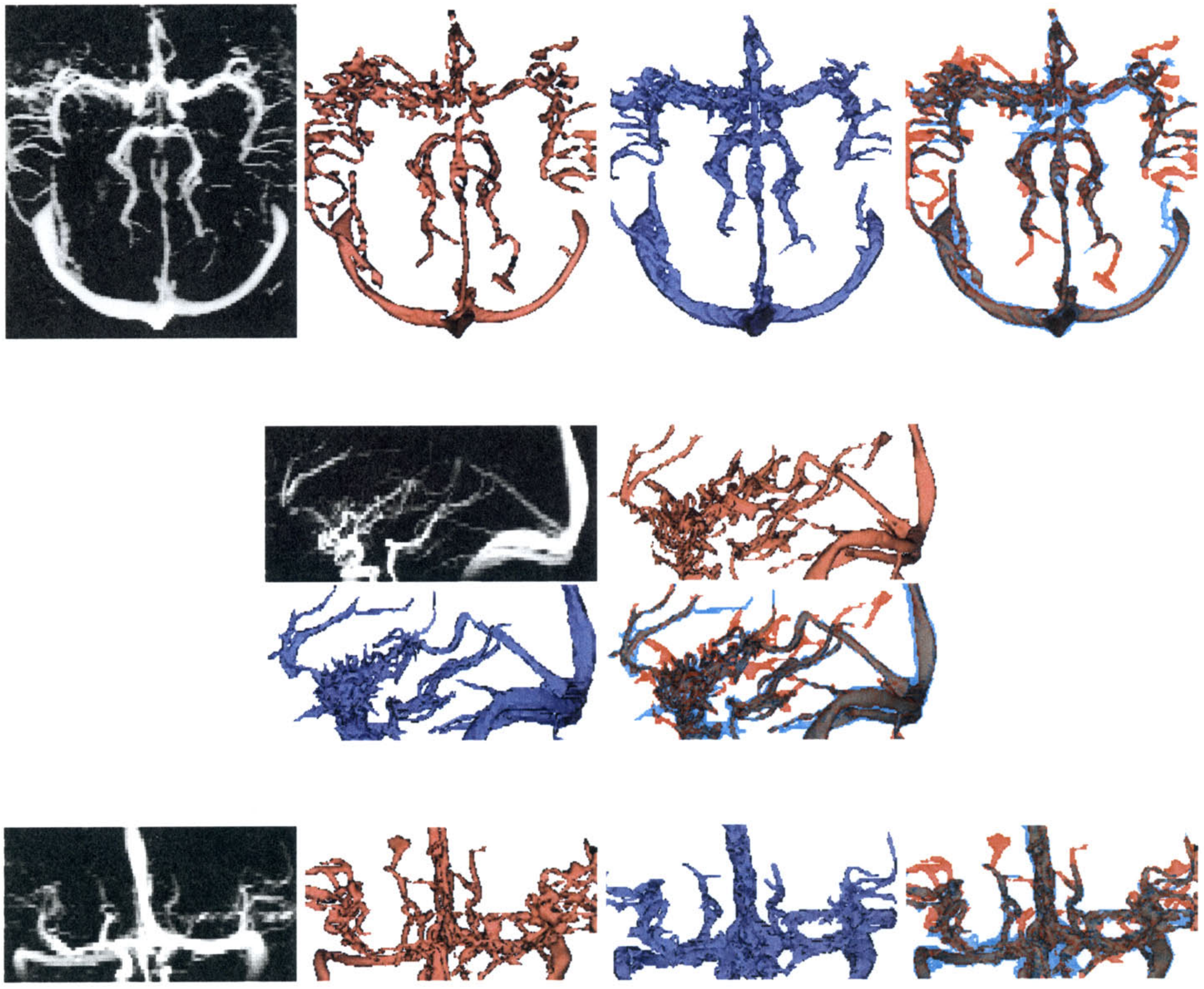


Figure 6-18: Dataset 4, Maximum intensity projection of raw data, CURVES segmentation (red), manual segmentation (blue), and combined image, from axial, sagittal, and coronal viewpoints.

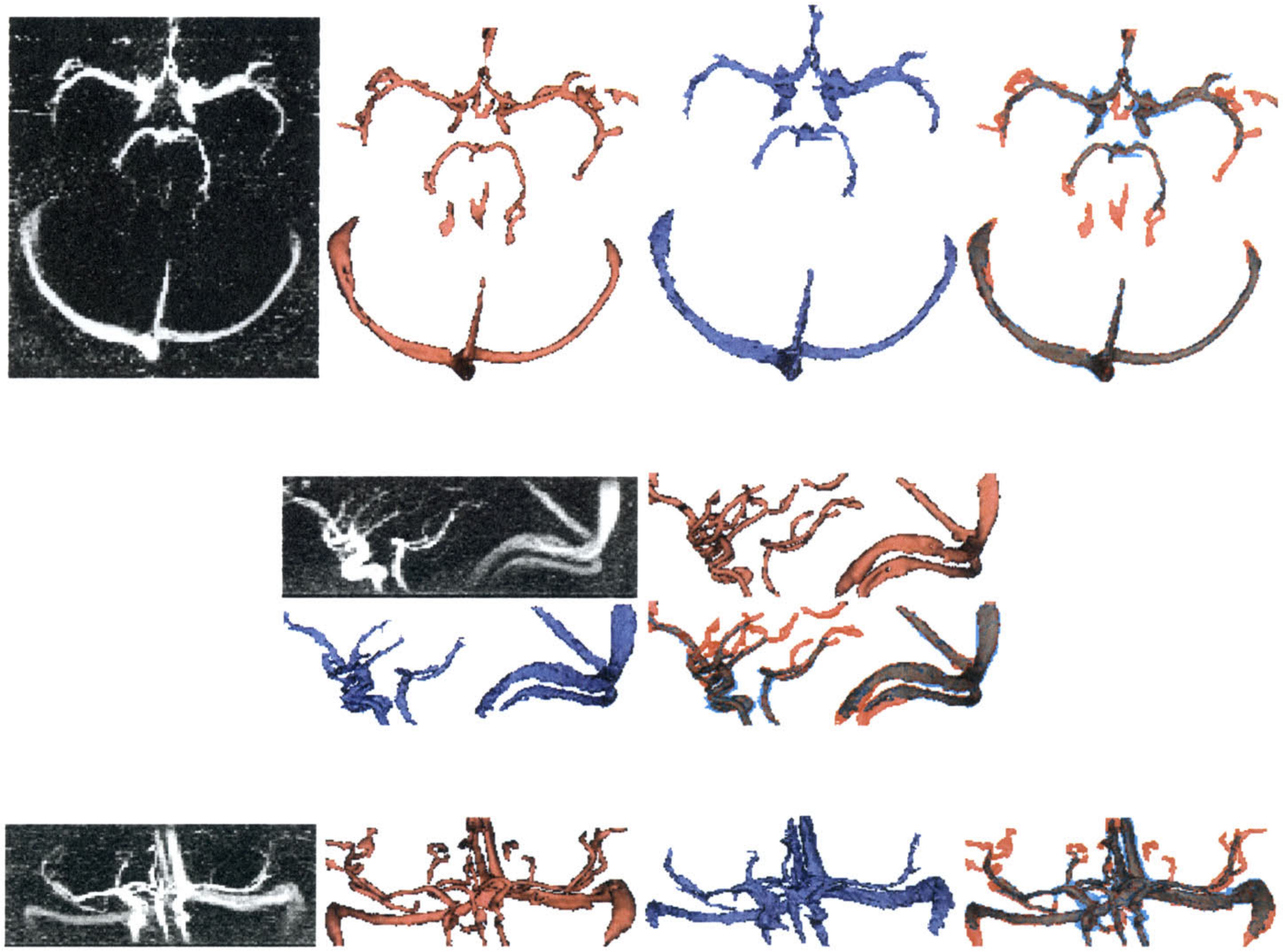


Figure 6-19: Dataset 5, Maximum intensity projection of raw data, CURVES segmentation (red), manual segmentation (blue), and combined image, from axial, sagittal, and coronal viewpoints.

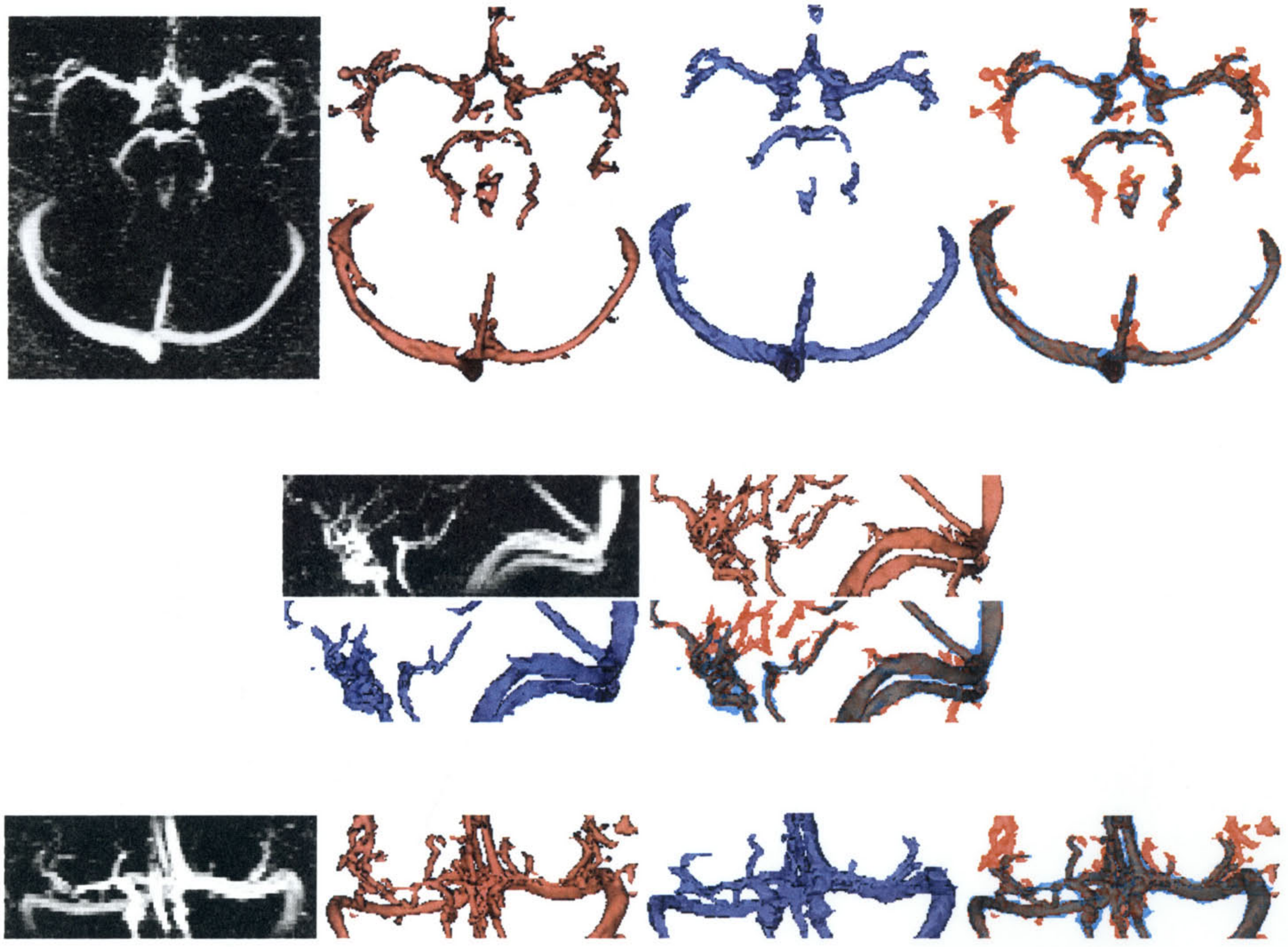


Figure 6-20: Dataset 6, Maximum intensity projection of raw data, CURVES segmentation (red), manual segmentation (blue), and combined image, from axial, sagittal, and coronal viewpoints.

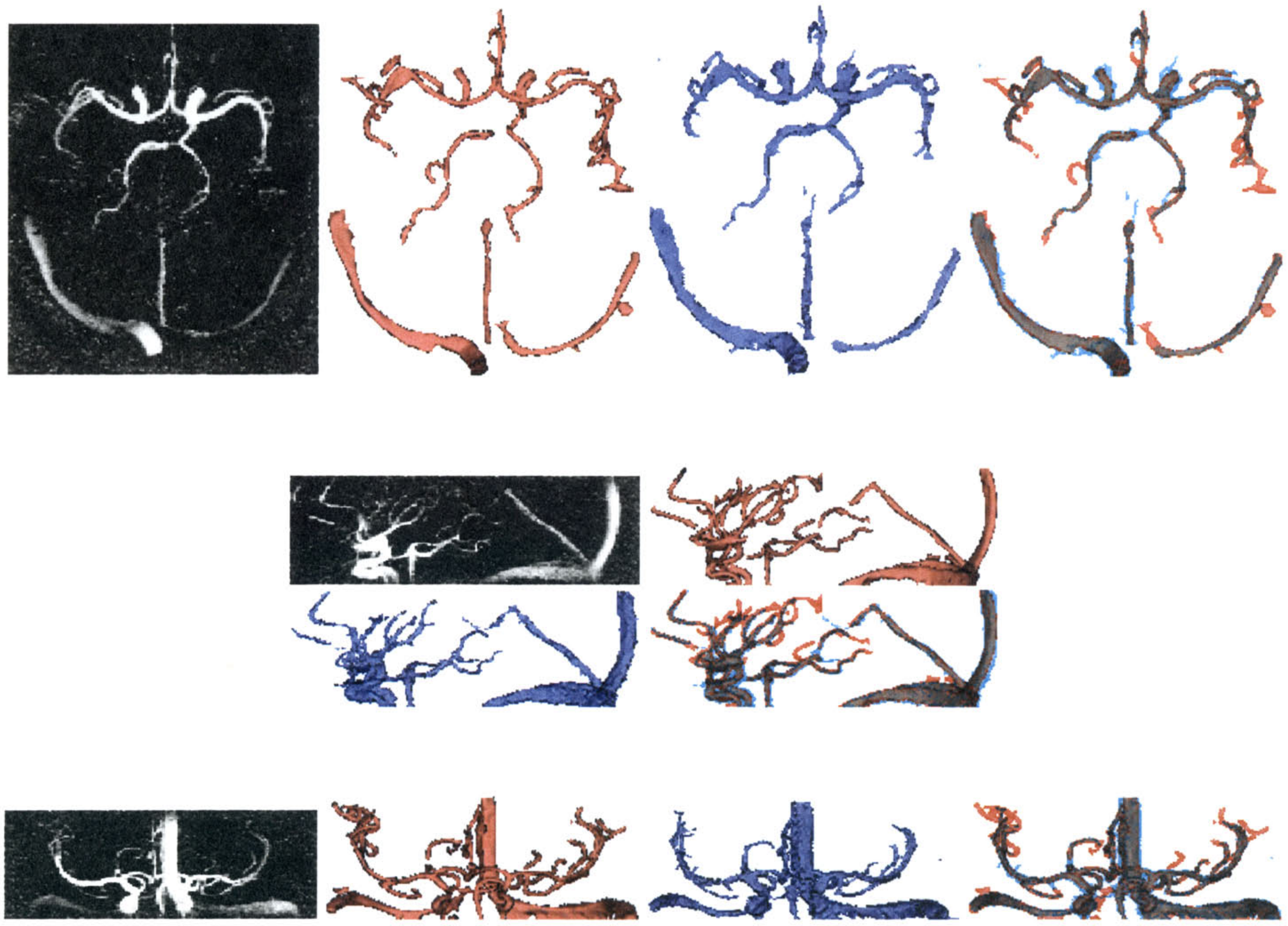


Figure 6-21: Dataset 7, Maximum intensity projection of raw data, CURVES segmentation (red), manual segmentation (blue), and combined image, from axial, sagittal, and coronal viewpoints.

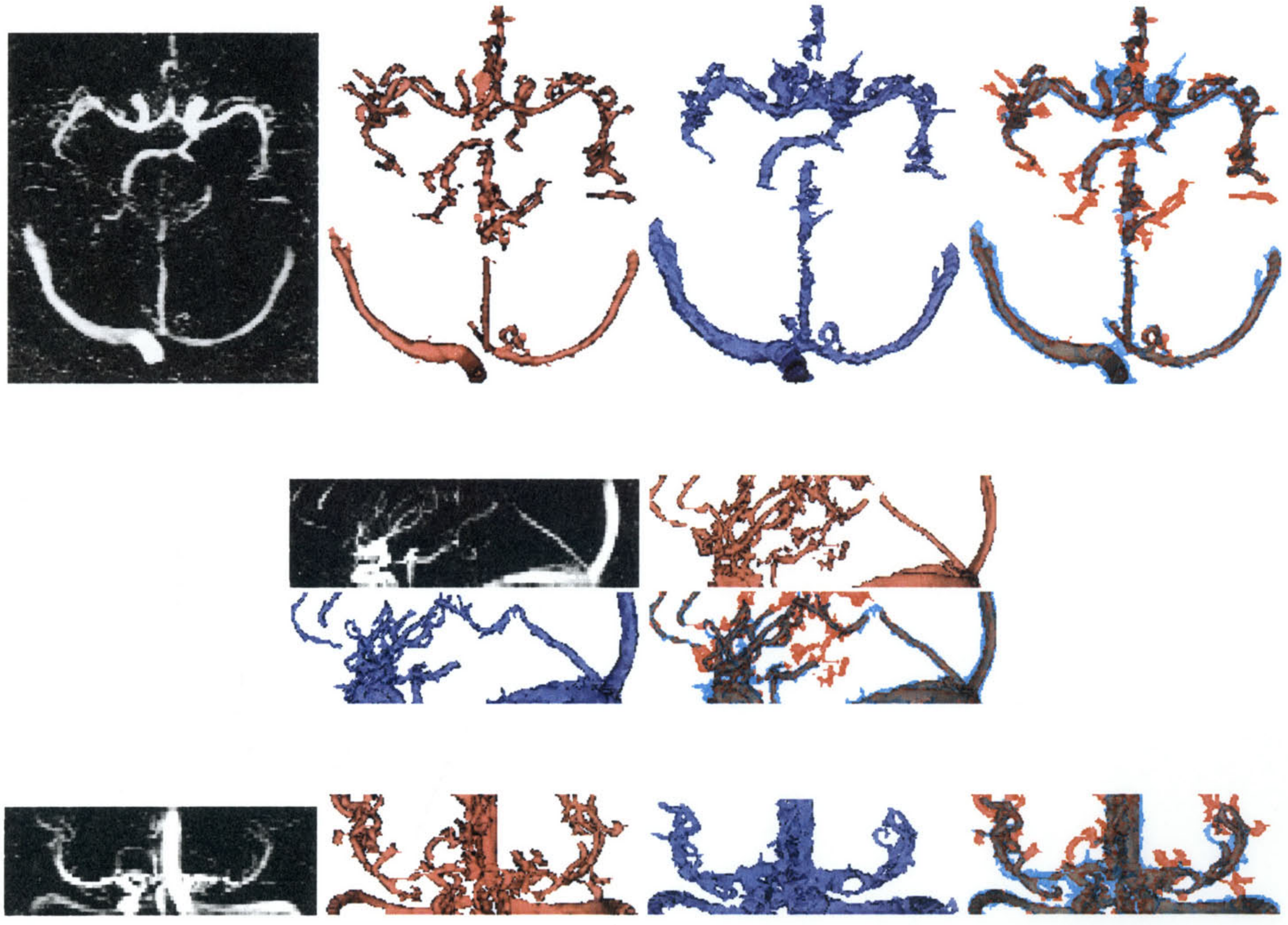
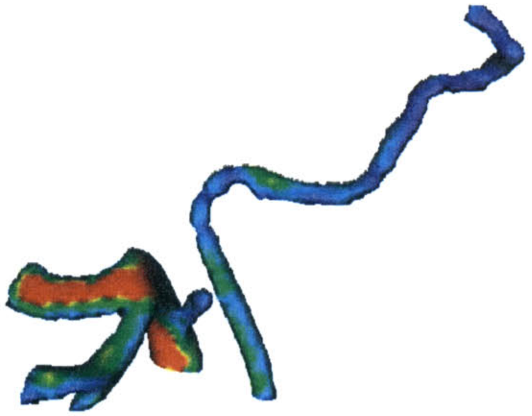
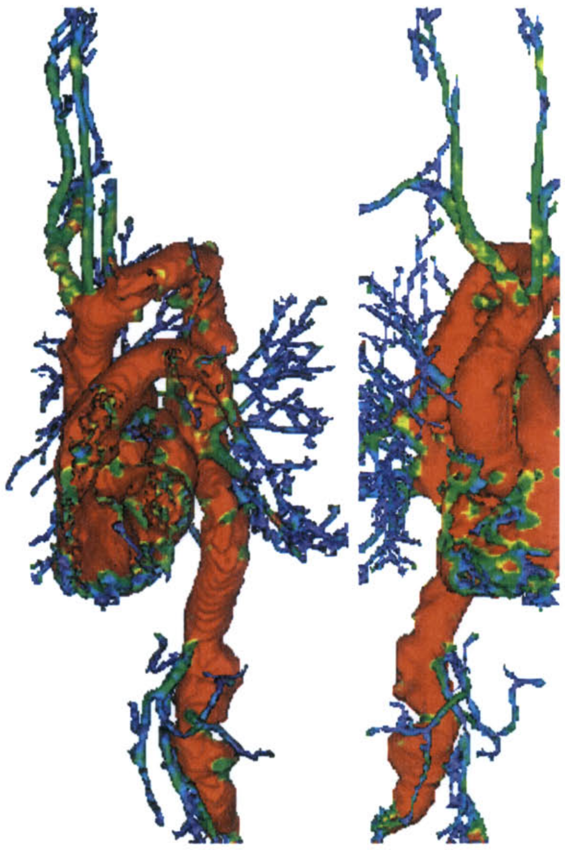


Figure 6-22: Dataset 8, Maximum intensity projection of raw data, CURVES segmentation (red), manual segmentation (blue), and combined image, from axial, sagittal, and coronal viewpoints.



(a)



(b)

Figure 6-23: Estimation of local radii of curvature of vessels. (a) In this image of a partial segmentation, the colorscale is continuous from blue to green to red, with blue indicating a radius of curvature $\leq$ 1mm and red indicating a radius of curvature $\geq$ 2mm. (b) The aorta MRA dataset from which this segmentation was made is courtesy of Siemens, acquired on a Siemens scanner at New York University. Again, the colors range from blue to green to red in order of increasing radius (numerical values not computed for this example). Two orthogonal viewpoints are shown.

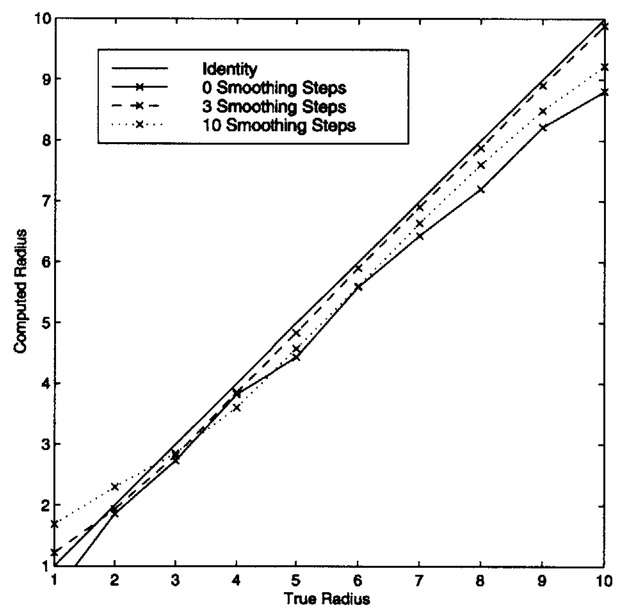


Figure 6-24: The accuracy of radii estimate versus tube widths and levels of smoothing. The identity plot (ideal case) is shown for comparison. Tube radii was varied from one to ten millimeters, assuming each voxel's dimension was 1mm^3. Plots are shown for zero, three, and ten iterations of smoothing, for the various widths.

Chapter 7

Centerline Extraction

The task of finding centerlines in blood vessels has important practical motivations. First, for large blood vessels, surgeons may perform endoscopic procedures in which an instrument would be inserted into and along the vessel. Vessel centerlines, or *skeletons*, are important in planning trajectories for such surgeries. Second, centerlines provide a more concise representation of the vasculature than the segmentation result itself. This representation can potentially be used in registering vasculature structures to those in an atlas for labeling, those taken at previous times for change-detection, and those of other patients for comparison. It can also be useful in building a topological model of the vasculature for study purposes.

In addition to these separate applications, skeletonization methods became important for our validation studies on the CURVES segmentation algorithm. It is possible to produce more meaningful quantitative comparisons if one compares thinned versions of vascular structures instead of the structures themselves because often one is less interested in obtaining the vessel wall perfectly than in acquiring the underlying topological structure. Further, if vessel width is incorrect by a small amount, the overall change in volume of the structure is large; one may not want a validation protocol to penalize more heavily for this error than for missing thin structures entirely.

Skeletonization in both 2D and 3D is a widely explored topic in computer vision. Some references for the general case are [13, 67, 77]. For the case of application to medical datasets, one should also refer to the "cores" approach [43]. Recent work which deals with branching and connecting separate pieces of the skeleton, related to our work below, is found in [70]. A system more related to ours which looks for ridges in the distance function and smooths them with an active contour model was developed for endoscopic applications [31].

We here present a novel method that was inspired by the presence of the distance function, available from CURVES. In general, the skeleton of a volumetric region such as the vessels is a two-dimensional surface. However, for many vascular applications, one would like the one-dimensional skeleton that best approximates this surface. That

$\boxed{-4}$	-2	0	1	2	3	2	0	-2
0	-2	-1	0	1	2	1	0	$\boxed{-4}$
2	1	0	$\boxed{-2}$	1	0	-2	-2	-1
4	2	1	-1	-1	-2	$\boxed{-4}$	-2	0
5	4	0	$\boxed{-2}$	0	2	0	1	3

Figure 7-1: An example 2D grid, near a distance function. The local minima are indicated with black boxes.

is, we want the skeleton obtained to be one voxel thick. We have designed our skeletonization algorithm to meet this constraint. In short, our algorithm, *Hierarchical Shortest Path (HSP),* finds an initial sparse set of disconnected points known to be on the centerline. These points are then linked together in a hierarchical manner, with increasingly large portions of the centerline structure detected at each iteration. After describing our algorithm, we compare it to two related thinning algorithms with which we also experimented.

7.1 Initial Centerline Points

Our centerline-finding algorithm is initialized with a set of disconnected points. Consider the signed distance function defined so that the segmentation result (the vasculature surface) is the zero level set and voxels inside the surface are defined to have negative distance. Our centerline extraction algorithm uses those points which have negative distance and are also local minima of the distance function to initialize the skeleton. Figure 7-1 shows an example on a 2D grid. The values are a rough approximation to a distance function, and the local minima are indicated with black boxes. Intuitively, these points must belong to the skeleton because they are inside the vascular tubes and are as far as possible away from the vascular walls.

The next step is to connect these points to give trajectory and topology information. This task is performed by applying Dijkstra's shortest path algorithm [34] repeatedly on larger and larger segments of the centerline structure and connecting the segments by the resulting paths, until the entire structure is connected. Again, the initial centerline structure is defined as the set of local minima points found as in Figure 7-1.

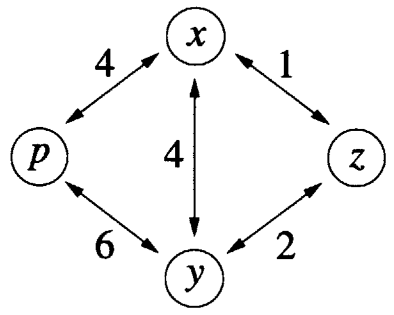


Figure 7-2: Illustration of Dijkstra's shortest path algorithm.

7.2 Dijkstra's Shortest Path Algorithm

Dijkstra's shortest path algorithm finds the minimal distance from one vertex p in a graph G to all other vertices in G. This problem is called the *single-source shortest-paths problem* [28]. Let $G = (V, E)$ be the graph, where a graph is a pair of a set V of vertices and a set E of edges. Each edge $(u, v) \in E$, with $u, v \in V$, has an associated real-valued weight given by the function $w : E \rightarrow \mathbb{R}$. The algorithm keeps track of a subset $S \subset V$ for which the distances to p are known. Initially, this set is null: $S = \emptyset$. The algorithm also stores an upper bound on the distance for each vertex which is initialized to ∞ for each vertex except p, for which it is initialized to 0.

These upper bounds are used as the *keys* for a *priority queue* Q which stores all of the vertices whose distances to p are not yet known. A *priority queue* is a data structure for which each entry has an associated *priority*, indicated by a real-valued *key*. Such a data structure also includes an efficient means of modifying an entry's key and of extracting the entry with the minimal key, which is assumed to have the highest priority. Further detail on priority queues is available in [28]. In Dijkstra's algorithm, the *keys* in the priority queue are the estimated distances to the vertices from p. That is, p has a key of 0 and the other vertices have a key of ∞ at the start.

Dijkstra's algorithm proceeds by repeatedly selecting the vertex u from Q whose key (distance from p) is least. This key is assumed to be the correct distance to u from p, so u is inserted into S. The algorithm then looks at all vertices v that are adjacent to u, *i.e.*, for which $(u, v) \in E$. For each such v, the key of v (the estimated distance from p) is replaced by the minimum of it and the sum of the key of u and the weight of (u, v), $w((u, v))$. Intuitively, this weight is the distance from u to v; if the distance from p to u (which is known to be correct) plus the distance from u to v is less than the upper bound on the distance from p to v, then we wish to lower that upper bound accordingly. After this procedure is performed for each v, the next u is chosen, and the process repeats until Q is empty.

Pseudocode for this algorithm is [28]:

```
Dijkstra($G = (V, E), w, p$)
    Initialize keys;
```

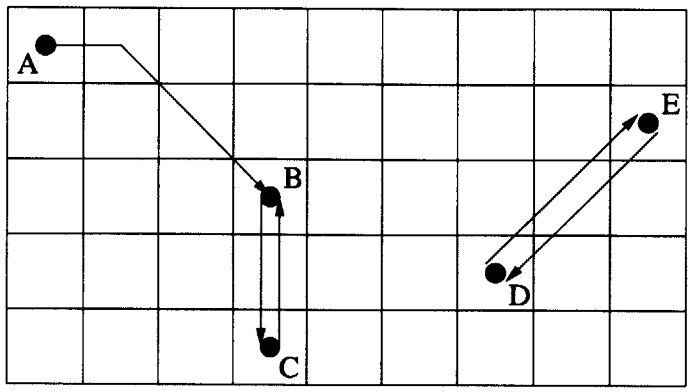


Figure 7-3: The initial nodes are indicated with black dots. The paths found by the first iteration of HSP are shown with arrows.

```
$S \leftarrow \emptyset$;
$Q \leftarrow V$;
while $Q \neq \emptyset$
    do $u \rightarrow$ Extract-Min($Q$)
        $S \leftarrow S \cup \{u\}$
        for each vertex $v$ such that $(u, v) \in E$
            do $key(v) \leftarrow \min(key(v), key(u) + w((u, v)))$
```

For a sample algorithm execution, refer to Figure 7-2. For this example, we will assume that edge weights are symmetric. That is, $(u, v) \in E \iff (v, u) \in E$ and for all $(u, v) \in E$, $(u, v) = (v, u)$. This property is displayed with the two-headed arrows in Figure 7-2. This symmetry is also the case in our vessel-centerlines application. On this graph, the algorithm terminates after the following behavior.

- After zero steps, $S = \emptyset, key(p) = 0, key(x) = key(y) = key(z) = \infty$.
- 1 step, $S = \{p\}, key(x) = \min(\infty, 0 + 4) = 4, key(y) = \min(\infty, 0 + 6) = 6$.
- 2 steps, $S = \{p, x\}, key(y) = \min(6, 4 + 4) = 6, key(z) = \min(\infty, 4 + 1) = 5$.
- 3 steps, $S = \{p, x, z\}, key(y) = \min(6, 5 + 2) = 6$.
- 4 steps, $S = \{p, x, z, y\}$.

7.3 Hierarchical Shortest Path

Now that the shortest-path algorithm is understood, we specify the HSP algorithm. For each local minima point, first pass of the HSP algorithm finds the closest other local minima point and stores the path between them. Consider the 2D example pictured in Figure 7-3. The paths found on this first pass are shown in solid arrows.

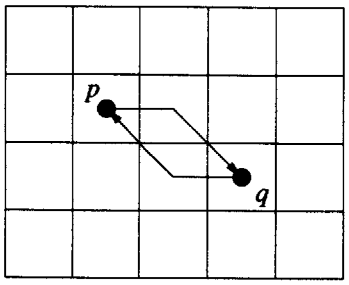


Figure 7-4: Paths (p, q) and (q, p) contain different points. We thus discard one of them when generating the cluster to maintain the requirement that the centerline be exactly one voxel thick.

Notice that points B and C find each other, as do points D an E, while point A finds point B. We next generate clusters of points, where points belong to the same cluster if they belong to the same connected component. Connected components are defined by these paths and their endpoints. After one pass of HSP, we have two clusters. The first cluster consists of points A, B, and C, and also the paths between A and B and between B and C. The second cluster consists of points D and E and also the path from D to E. Note that if a path (p, q) is included, we do not include the path (q, p). Since we do not restrict the ordering of points considered in building the paths, it is possible for (p, q) to contain different points that (q, p) (Figure 7-4). If we added both paths, then the resulting structure could be wider that one voxel, which violates our requirements for the centerlines.

The second path operates on the newly-constructed clusters. For each cluster, it finds the closest other cluster, and stores the path between them. The closest cluster is defined according to the distances from all points in the current cluster to all points in the cluster being checked for proximity. That is, the distance between two clusters is the minimum distance between any point of the first cluster and any point of the second cluster. Figure 7-5 shows the clusters after the first pass of the algorithm. The first cluster is indicated with white circles, the second with black circles. Shortest paths are found from point F in the first cluster to point D in the second cluster, and vice versa, as indicated in the figure with arrows. Notice that the point of connection, F, in the first cluster was not one of the original local minima points. This illustrates the fact that once a cluster has been formed, all points are considered equally; it is irrelevant which ones were in the paths and which were original points. The next step is to generate higher level clusters. In this case, since the original points are now fully connected, we only have one cluster (Figure 7-6) so the algorithm terminates.

If the points were not fully connected, the process of alternately finding paths and clustering would continue until either there was only one cluster or there was no path between remaining clusters. The latter could happen if the vessel structure contained disconnected vessel trees. We reiterate that in all searches, the only points valid for including in a path are those inside the vessel wall, as indicated by the sign of the

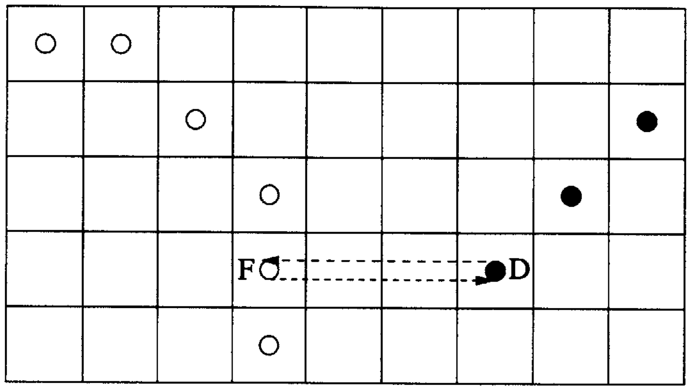


Figure 7-5: The two clusters after the first pass of the algorithm: one marked with white circles, the other black. The paths found for each cluster are shown with arrows.

distance function to the vessel walls. Further, the datasets on which we apply HSP are 3D; 2D was used for illustration only.

Formally, the HSP algorithm specification is the following.

- Input:
  1. graph $G = (V, E)$ of vertices and edges,
  2. partition of vertices $V = P \dot{\cup} S$ into *primary* and *secondary* vertices, and
  3. weight function $w : E \rightarrow \mathbb{R}$ on edges.
- Procedure: Iterate the following 2-step sequence until convergence.
  1. Shortest-Path Step: For each $p \in P$, calculate

     $$path^*(p) = \min_{q \in P-p} SP(p, q)$$

     where $SP(p, q)$ is the shortest path ([34]) from p to q. If no path exists, define $path^*(p) = p$.
  2. Cluster Step: We now have *clusters*, each of which is comprised of two primary nodes and the *path**'s of secondary nodes connecting them. Alternatively, some clusters may contain more than two primary nodes if the same node was found by more than one node in the previous step. Let C be the set of all clusters. Then update the graph as follows.
     (a) $P_{t+1} \leftarrow C$
     (b) E is re-defined as follows. $\forall (p, s) \in P_{t+1} \times S$, $\exists (p, s) \in E_{t+1}$ if $\exists r \in p$ such that $(r, s) \in E_t$. w is re-defined for nodes p in P_{t+1} as

     $$w(p, s) = \min_{r \in P_{t+1}, (r,s) \in E_t} w(r, s).$$

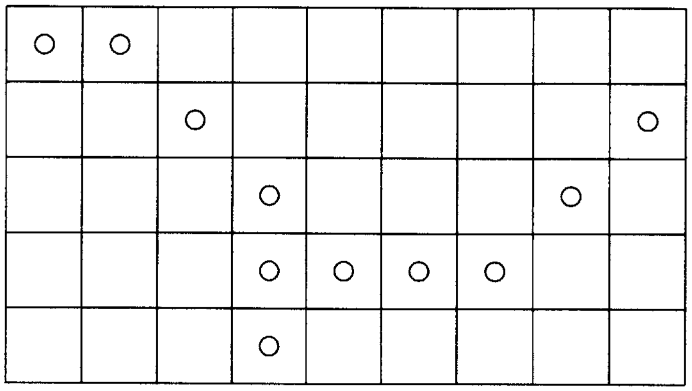


Figure 7-6: The final cluster of this run of the HSP algorithm. This is the detected centerline.

There is no change to $w(s,t)$ where $s,t \in S$.

- Output: P.

We make two comments on the algorithm. First, at convergence, the number of primary nodes $|P|$ is the number of disconnected subgraphs in the initial graph G that contain at least one primary node $p \in P$. Second, one could update S in the second step of the procedure to remove any node that is contained in any cluster. This is not necessary, however, since such a node would never be found by the shortest path algorithm: when that node is encountered, the cluster containing it would be encountered, which is itself a primary node, so the shortest path algorithm would stop because a primary node was reached. Further note that the distance function information is not used after the initial local minima points are detected, except that its sign is used as a mask indicating which voxels are inside the object and which are outside it. This mask is used as a constraint on the paths found between the points.

Our HSP algorithm follows a similar structure as general *hierarchical clustering* algorithms in computer science [53]. Such algorithms can be classified as either *divisive* or *agglomerative*, depending on whether they build the clusters in a coarse-to-fine order (divisive) or a fine-to-coarse order (agglomerative). Our HSP algorithm is thus agglomerative. General agglomerative hierarchical clustering algorithms cluster a fixed set of data points into clusters at increasing levels, where the clusters at a higher level are groups of clusters from a previous level. Our HSP algorithm has this intuition, but differs in that the points that are not in the initial set of local minima points, but are in the final cluster(s) are not known *a priori*. Instead, more such additional points are found at each level of the hierarchy, *i.e.*, at each iteration. It still, however, is helpful to see the connection of our algorithm to this broad and well-studied class of algorithms. The reader is referred to [53] for information on general clustering algorithms. For information on shortest path algorithms, the reader is referred to [28].

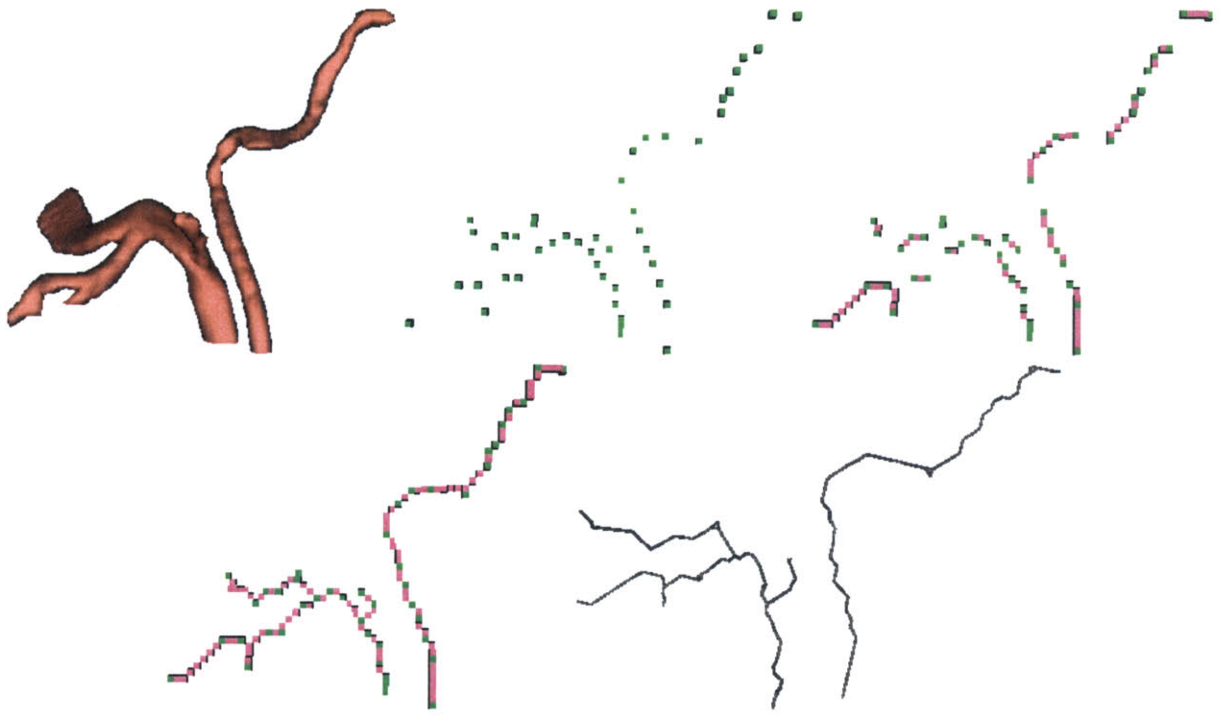


Figure 7-7: Illustration of skeletonization procedure on a partial segmentation of a cerebral MRA dataset. First row shows segmentation, local minima of distance function, and skeletal structure after one iteration. Second row shows final skeletal structure in original display format and also in line segment format at a slight rotation.

7.4 Application to MRA data

For this application, we initialize the *primary nodes* P to be the negative local minima extracted from the distance function and the *secondary nodes* S to be all other voxels inside the vascular walls. We define edges E between nodes adjacent in the 3D volume and weight the edges with the Euclidean distance between the corresponding voxels; we use 26-connectedness which means a point is *adjacent* to each of the 26 points in the smallest cube around that point, so weights are either 1, $\sqrt{2}$, or $\sqrt{3}$. We then run the hierarchical shortest path on this graph.

In more detail, the graph $G = (V, E)$ for our application is constructed as follows. Let v be the distance function volume, and recall that the vessel wall is defined implicitly as the zero level set of v, and those voxels inside the vessel wall have negative values and those outside have positive values. We thus insert a node into V for each voxel with a negative value of v. Assuming 26-connectedness, there is an edge in E between each pair of vertices whose corresponding voxels are adjacent. The weight of each edge is the distance between the voxels: 1, $\sqrt{2}$, or $\sqrt{3}$. The partition of V into sets of primary and secondary nodes $V = P \dot{\cup} S$ is given by the local minima of the distance function. Those local minima comprise P and the remaining points

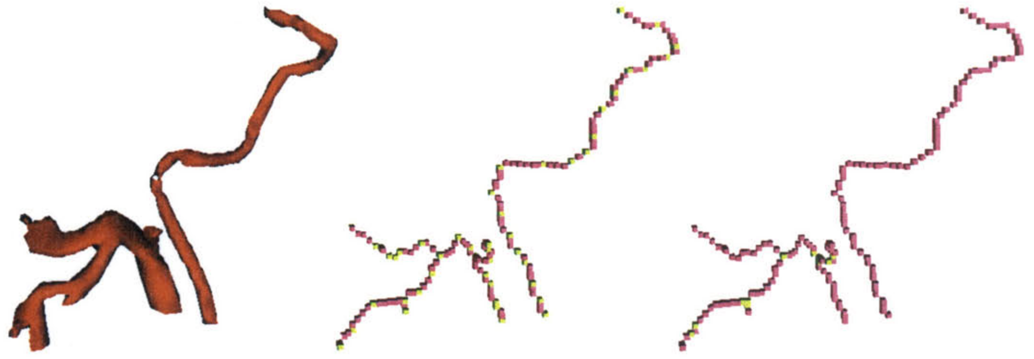


Figure 7-8: a. Surface model of vessels in a human neck. b. Skeleton model with local minima of distance function in yellow and connections obtained by hierarchical shortest path in magenta. c. Same skeleton with branch points only shown in yellow.

inside the vessel wall comprise S. Then for each primary node, we run Dijkstra's shortest path algorithm. We do not run it until all shortest paths are found, however. That is, we do not continue until Q is empty. Instead, we terminate the algorithm when one of the vertices extracted from Q is a primary node. The path to this vertex is the answer for the given starting point. All paths are then clustered into new primary nodes, and the hierarchical algorithm continues.

An example on a partial vasculature segmentation is shown in Figure 7-7. The initial local minima are shown in green, and the paths detected by hierarchical shortest path are shown in purple. We can display the skeletal structures connected line segments to emphasize the fact that the structures are indeed one voxel thick, as also shown in the last image where a line segment is drawn between each pair of neighboring voxels in the set of HSP skeleton points. The fact that the resulting path is one-voxel in diameter implies that branching points can be detected trivially as voxels with more than two neighbors in the centerline structure. This topological information is important for endoscopic applications, registration, labeling, and other purposes. Figure 7-8 shows an example in which the skeletonization result is displayed with branch points indicated. Figure 7-9 shows a skeletonization example on a full cerebral MRA dataset using the line segment display option.

7.5 Comparison to Other Methods

We experimented with two other techniques used for thinning and skeletonization. The first uses extrema of the distance function, like ours. The second is based on topological constraints. Neither guarantees a one-voxel-thick skeleton, which our HSP algorithm does, but they both have advantages.

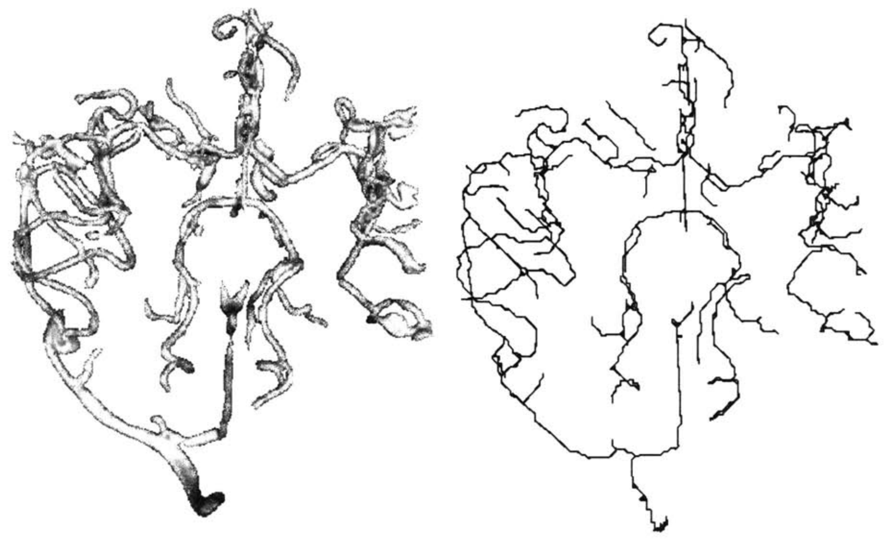


Figure 7-9: Centerlines for a cerebral vascular surface model, shown as a line-drawing.

7.5.1 Singular-ness of Distance Function

Gomes *et. al.* use the singular-ness of the distance function to estimate skeletons of 3D surfaces [45]. That is, the singular points correspond to skeleton points of the 3D object. They define eight derivative operators as follows. Let v be the 3D distance function, then define $D_x^+ v = v(i+1,j,k) - v(i,j,k)$ and $D_x^- v = v(i,j,k) - v(i-1,j,k)$. Define expressions for D_y and D_z analogously. The eight derivative operators are then

$$\begin{aligned} D^1 v &= (D_x^+ v, D_y^+ v, D_z^+ v), && (7.1) \\ D^2 v &= (D_x^+ v, D_y^+ v, D_z^- v), && (7.2) \\ &\dots && (7.3) \\ D^8 v &= (D_x^- v, D_y^- v, D_z^- v). && (7.4) \\ & && (7.5) \end{aligned}$$

Let $\overline{Dv} = \frac{1}{8}\Sigma_i D_i v$ be the average of the values of those eight operators.

Singular-ness s is calculated according to the dot products between the individual operators, normalized, and the average, also normalized.

$$s = \sum_i \left(\frac{D^i v}{\mid D^i v \mid} \cdot \frac{\overline{Dv}}{\mid \overline{Dv} \mid} \right)^2 \tag{7.6}$$

One then thresholds the volume based on s: if s is below some threshold the point is labeled as a skeleton point, else it is not. The intuition is that small dot products correspond to large angle differences, which correspond to large variation in derivative estimates, which means we are likely to be at a singularity. Singularities of the

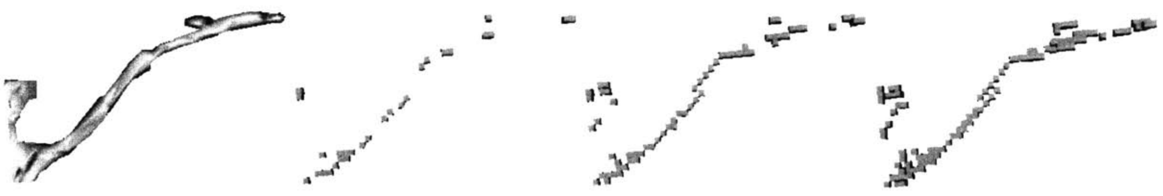


Figure 7-10: A surface model of blood vessels segmented from and MRA image, followed by the points labeled as "skeleton points" according to the definition in [45], for three different thresholds.

distance function are skeleton points of the surface.

Gomes *et. al.* demonstrate smooth-looking skeletons for 3D surfaces that are the segmentation of the cortex. We applied this method to our vascular models to observe its performance on very thin structures. Figure 7-10 shows the skeleton points found for a sample branching structure, for several different thresholds of s. Since the structure is very thin, increasing the threshold to obtain a connected structure causes much of the vessels to be included. In general, the method does not allow one to obtain a connected list of points that indicate a 1D curve in 3D space.

7.5.2 Simple Points

Bertrand and Malandain discuss a geometric concept called *simple points* [11, 86]. Assuming an object is indicated by a binary image, a *simple point* is a point whose deletion does not change the topology of the object. That is, it does not change the number of cavities, connected components, and holes in the object. One can then define skeletonization algorithms based on the removal of simple points from the object.

Assume we are using 26-connectivity for the object X and let $N^*_{26}(x)$ be the set of 26 neighbors of x. We must then use 6-connectivity for the background $\bar{X}$, which means that points of the background are only said to be *adjacent* to those points directly above, below, in front of, behind, left, or right of them; this set is called $N^*_6(x')$ for a point $x' \in \bar{X}$. We cannot use 26-connectivity for both because this would enable objects to pass through each other; 6-connectivity for both would unnecessarily break the object into too many pieces. Finally, 18-connectivity considers as adjacent to x all points of the cube around x except for the 8 corners; this set is called $N^*_{18}(x)$.

The following definitions are stated in [11]. Let x be the point we are currently checking to determine if it is simple or not. Let $NCC_a(Y)$ be the number of connected components of Y adjacent to x. The *topological number* of a point x in an object X, using 26-connectivity, is defined as

$$T_{26}(x, X) = NCC_a[X \cap N^*_{26}(x)]$$

and the topological number using 6-connectivity is

$$T_6(x, X) = NCC_a[X \cap N_{18}^*(x)].$$

In words, for the 26-connectivity, the topological number of x is the number of connected components (in the object) in the cube around x that remain if x is removed from the object.

Then a point x in an object X is a *26-simple point* if and only if

$$T_{26}(x, X) = 1 \qquad \text{and} \tag{7.7}$$

$$T_6(x, \bar{X}) = 1. \tag{7.8}$$

There can only be one connected component of the object in the cube around x, since x is in the object and all other points of the cube are adjacent to x. So the first half of this definition says that the removal of x does not cause there to be more than one connected component; that is, it is not the case that only x is holding the component together. The second half of the definition says that x is not simple if its removal creates a hole, in the case of $T_6(x, \bar{x}) = 0$, nor if its removal deletes a 6-cavity of X or creates a 6-hole of $\bar{X}$. We refer the reader to [11] for explanation of the latter situation.

Malandain and Bertrand provide the following algorithm for determining if a point x is simple or not [76]. Let NC_a^n be an indicator which is one if the number of connected components adjacent to the point x under consideration is one and zero otherwise, where n is the connectivity used. That is, $n = 26$ or $n = 6$ depending on which topological number statement we are testing. Let E_n be the set of object (or background for $n = 6$ case) points in the considered neighborhood, *i.e.*, $E_{26} = X \cap N_{26}^*(x)$ and $E_6 = \bar{X} \cap N_{18}^*(x)$. Let S be the list of points adjacent to x that need to be considered, *visited* the array that tells whether or not a point in E_n has been considered, and N the number of points in E_n that have been considered. T is a temporary list that is used when updating S, since this update cannot be performed "in place". The following algorithm will be applied for both $n = 26$ and $n = 6$ to check Equation 7.7 and Equation 7.8 respectively.

```
• Choose some $i \in E_n$. $S \leftarrow \{i\}$; $T \leftarrow \emptyset$.
• $visited(i) \leftarrow$ true; $visited(y) \leftarrow$ false for all $y \in E_n - \{i\}$
• $N \leftarrow 1$
• while $S \neq \emptyset$ do
    – for all $y \in S$ do
        ∗ for all $z$ $n$-adjacent to $y$ with $visited(z) =$ false
```

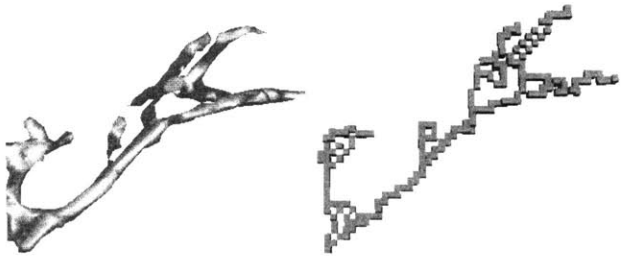


Figure 7-11: A surface model of blood vessels segmented from and MRA image, followed by the points that remain after removing simple points.

```
            · $N \leftarrow N + 1$
            · $visited(z) \leftarrow$ true
            · $T \leftarrow T \cup \{z\}$
      – $S \leftarrow T; T \leftarrow \emptyset$
• if $N = |E_n|$ then $NC_a^n = 1$ else $NC_a^n = 0$.
```

An increase in efficiency is possible by checking the following conditions before running the above algorithm. If there is no $i \in E_n$ n-adjacent to x, then x **is not** a simple point. If there is a single point of $\bar{X}$ 6-adjacent to x, then x **is** a simple point. If there are exactly two points $\bar{X}$ 6-adjacent to x, then x **is not** simple if the two points are opposite, and x **is** simple if they are 18-adjacent to each other and the common neighbor is in $\bar{X}$. If these conditions do not apply, one runs the above algorithm for NC_a^{26} and NC_a^6 in accordance with Equation 7.7 and Equation 7.8.

To construct a complete thinning algorithm from this simple points characterization algorithm, it remains only to specify the order in which the points are to be checked for removal. The thinned structure depends on this order, so it is necessary to choose an ordering consistent with knowledge about the object. For use on our vessel segmentations, we based the order on the intensity of the MRA image so the faintest point in the object is the next to be removed if it is a simple point. The intuition for this strategy is that the centerlines of the vessels are brightest, with intensity falling off monotonically with distance from the centerline. We also prohibit the removal of end points from the structure to avoid shortening the vessels. Our thinning algorithm is thus:

- Input the binary image with vessel points X indicated.
- Sort these vessel points in order of increasing intensity, so the faintest will be checked first for removal.
- For each $p \in X$, if p is a simple point and is not an endpoint, $X \leftarrow X - \{p\}$.

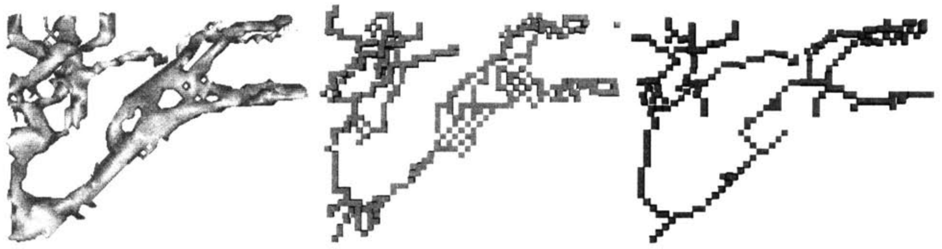


Figure 7-12: A surface model of blood vessels segmented from and MRA image, followed by the points that remain after removing simple points, then by the skeleton obtained with our Hierarchical Shortest Path skeletonization method.

Another option for the ordering of the points would be to use the distance function defined by our segmentation algorithm. Recall that all vessel points will have negative distances by definition, so the order would be to remove those whose magnitude of distance is least. We would expect performance to be similar to the use of image intensity.

Figure 7-11 shows a vascular segmentation obtained by CURVES and the result of applying our simple points-based thinning algorithm to it. A cube is drawn for each non-simple point in the skeleton. It appears that our implementation of these ideas retains too many points. However, it is still useful in cutting down the number of points from the original number of object points in a meaningful way, for use in validation for example. Figure 7-12 shows another example for which we also ran our Hierarchical Shortest Path skeletonization algorithm. Notice that the HSP method only is able to guarantee a one-voxel-thick skeleton, which is desirable for vascular applications. However, it is possible that the HSP method could miss segments of the vasculature if no initial local minima point was found within particular segments. Our simple points-based algorithm retains all of the original structure.

Dokládal *et. al.* propose two dual approaches to segmentation of vascular structures using this definition of simple points [36]. That is, they do not use it to thin an existing segmentations but use it to generate the segmentation from 3D images. Their algorithms operate on volumetric CT scans of the liver, in which vessels appear brighter than the surrounding tissues. Their first algorithm requires initialization with a point inside the vessel structure (which can presumably be obtained fully automatically). The algorithm then proceeds to iteratively grow this initial structure by adding adjacent simple points. Again, the order of insertion is important, and points are added in order of brightest to dimmest. The algorithm also restricts points to be brighter than some threshold before adding them; this condition provides the termination criterion for the algorithm. The dual approach is to initialize with a non-vessel point, *i.e.*, a point in the background of the image. One then reconstructs the background of the image by iteratively adding adjacent simple points until no simple points that are dimmer than the threshold exist. The fundamental idea behind this pair of algorithms is that the vascular structure should have a fixed and pre-specified topology: it is a single simply connected component. Constraining the growing procedures to simple points enforces this specification.

Chapter 8

Application to General Classification Problem

A very different application of the evolution of high codimensional manifolds is for general data representation and classification problems. Each instance in a training set would be represented by a point in a high dimensional feature space which is the embedding space for the manifolds. One would then evolve manifolds in this feature space, and the resulting manifolds would be a concise, meaningful representation of the objects in the training set. One could classify new instances by their distances to the manifolds, and one could analyze variability within object classes by the shapes of the manifolds. One could further use this representation to detect anomalies in datasets, for example, when new instances do not lie on or near existing manifolds.

8.1 Approach

Specifically, we create an N-dimensional image where N is the number of features used to describe an instance of an object and where each positive instance corresponds to brightness in the feature space and each negative instance darkness. We then use intensity gradients and regularization to control the evolution so that the manifold is attracted to large intensity gradients. For the case of 1-D manifolds in 3-D space, this will result in tubular structures (analogous to blood vessels) in which the interior is contained in the manifold and the exterior is not.

For classification and recognition, the probability of an instance being the object is high inside the manifold, lower as the distance from the manifold increases, and highest at the centerlines of the manifold. The relative widths can also be used to express variance of an object: wider tubes indicate higher variance in the corresponding object.

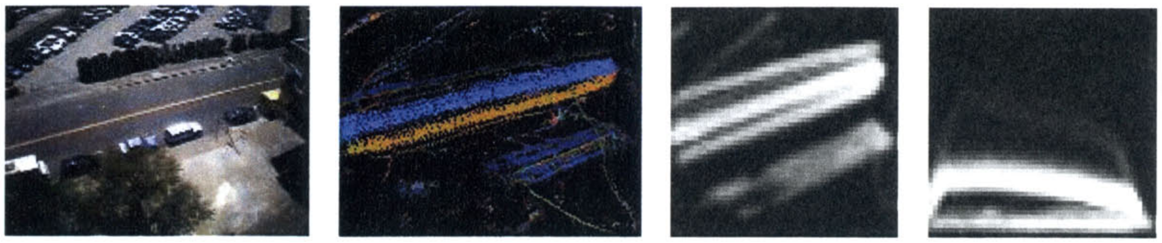


Figure 8-1: a. The scene. b. Motion trajectories for tracked objects: cars, pedestrians, bicycles, and trucks. c. Projection onto x-y space. d. Projection onto x-s space: the lowest white segment corresponds to people, the second and brightest segment to cars, and the highest and faintest segment to trucks which have the largest size (s) values.

8.2 Experiments

For proof-of-concept, we applied this approach to data points acquired by Chris Stauffer at the MIT AI Laboratory as part of the video surveillance and monitoring project. He has built a tracker that detects moving objects in the scene [48]. The first image in Figure 8-1 shows an area of a street in front of the laboratory. Moving objects that occur in this scene include cars, trucks, bicycles, and pedestrians. The second image in this figure shows locations at which these objects were detected. We wish to obtain a geometric description of the patterns of motion occurring in this scene which could be used for detecting anomalies that may be detected by the tracker. In addition to anomaly detection, the description could be useful as a rich and compact representation of activity within the scene. An important requirement of the representation is that it accurately separate out the different types of instances of motion, such as cars, trucks, bikes, and pedestrians while obtaining smooth manifolds for each.

For this experiment, we evolve 1D manifolds in a 3D feature space, where the three dimensions correspond to x, y (of centroid), and size (s) of an image region that was detected by the motion tracker. We obtained a list of 30,880 such triples and initialized the 3D space (resolution = 50x50x50) as described above. We smoothed the volume for compatibility with the gradient-based approach. Figure 8-1 the volume we created from this dataset in maximum intensity projection from two orthogonal viewpoints. The first viewpoint is like the previous images, with the x axis horizontal and the y axis vertical; notice the motion trajectories appearing as bright lines or smears. The second viewpoint shows the x axis horizontally and the size axis vertically, so pedestrians correspond to the lowest line or smear, then bikes, cars, and trucks in order of increasing image height in this projection. Notice that the truck line is faint as there were comparatively few trucks in the dataset.

Manifolds produced by CURVES are displayed in Figured 8-2. Notice that the manifolds are smooth while still showing potential for cleanly separating out the different classes of instances.

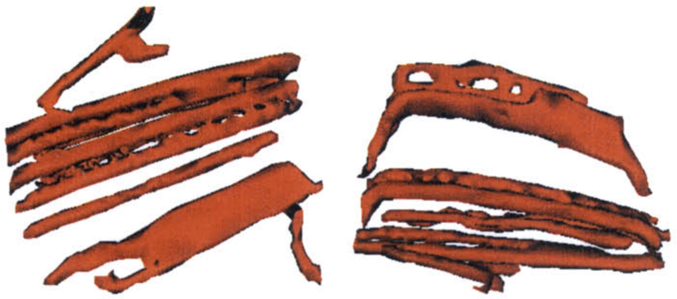


Figure 8-2: Manifolds produced by CURVES, from orthogonal viewpoints.

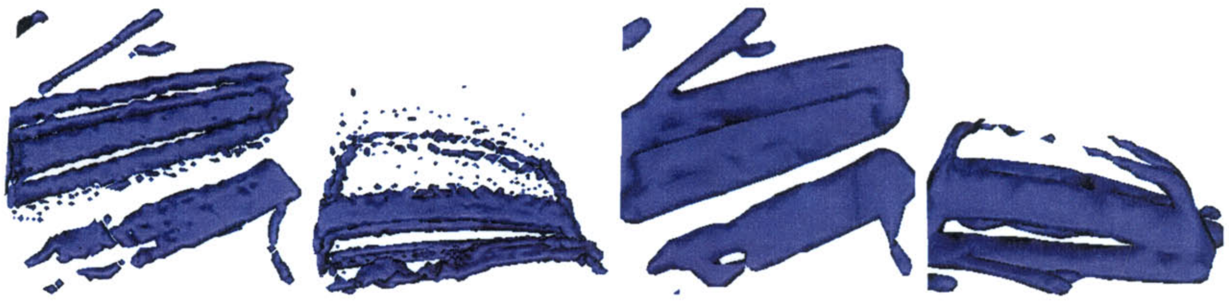


Figure 8-3: First two images: Surface models produced by thresholding non-smoothed data, too noisy to use as a geometric description. Final two images: Structures obtained by thresholding smoothed data. Note that it is not possible to obtain the fainter lines (such as the trucks in the X-S view) while simultaneously separating the lanes of traffic into different structures, which is possible with the CURVES method.

We compared this evolution approach to simple thresholding on the initialized 3-D space. The first two images in Figure 8-3 show results of thresholding the raw data: gaussian blur was not used. The final two images show results of thresholding the Gaussian-blurred dataset, which was the input to the evolution algorithm. Observe that in order to obtain most of the structure by thresholding, one either needs to blur together the different object classes or settle for a noisy representation, both of which are undesirable. Thus, these images support the use of our evolution technique to potentially produce a more robust and useful geometric class than considering only the raw data.

Chapter 9

Conclusions

We have studied the mathematics of manifolds undergoing under curvature-based evolution and the related computer vision techniques for segmentation. Moreover, we have contributed to a further understanding of the connection between these two topics by exploring the new area of regularization dependent on a manifold which is not a hypersurface. Using these ideas, we have developed a novel algorithm for the segmentation of tubular structures in medical images. We have customized our algorithm for the segmentation of vessels in MRA datasets because of the clinical importance of this application. During algorithm development, we worked with surgeons and radiologists to understand this problem and the types of segmentations that are most useful clinically so that our algorithm could be clinically useful. To demonstrate potential clinical benefit, we have run our algorithm on over 30 medical datasets and have compared our segmentations to manual segmentations when possible. In addition to the segmentation algorithm, we developed a centerline extraction algorithm that operates on the vascular models because our interaction with clinicians showed us the importance of such a tool. Finally, we performed preliminary experiments to show the wide applicability of high-dimensional manifold evolution, beyond the medical domain. In particular, we showed an application to data representation for general classification an recognition tasks.

This work suggests a number of challenging studies in the computer vision and medical image analysis fields, including

- Varying the image weighting function g in Expression 5.1, perhaps making it dependent on quantities other than the gradient magnitude in addition to experimenting with different functions;
- Exploring the use of the combination of Marching Cubes isosurface generation and very smooth distance functions to create surface models as smooth as those returned by CURVES for other applications;
- Developing a labeled atlas for the blood vessels such as that described by Koller *et al.* in [57] and the incorporation of the raw 3D MRA flow vectors (Section 4.1)

into the segmentation algorithm for increased performance;

- Exploring further the evolution of true 1D curves with particular application to the extraction of vascular centerlines instead of evolving tubes and finding the skeletons separately;
- And exploring the use of manifold evolution for data representation (Chapter 8) for more complicated situations of higher dimensions and sparse data.

Such studies would provide practical benefit in addition to further understanding of the theory of manifold evolution and related areas.

Appendix A

Euler-Lagrange Equations

The calculus of variations is the branch of calculus in which the variables of the expressions or equations are functions instead of single-valued parameters. A primary topic in this area is finding extrema of expressions that depend on functions. We review this calculation because it is used in minimizing the energy functional upon which the active contours approach is based. For further details, we refer the reader to [51] and [37]. The explanation that follows is based on that in [51], with customization to the energy-minimization problem used in geodesic snakes and in CURVES.

Consider the integral

$$E = \int_0^1 \Phi(p, C, C')dp$$

where C is an unknown function that depends on p, and Φ is a function that depends on C and its derivative C', also unknown. Further assume that C is a closed curve, so $C(0) = C(1)$, and that we wish to minimize E over all possible C. We will call E the "energy" of C. The goal is to characterize the C at which a minima is attained according to the property that the gradient of E is zero there. To minimize E by gradient descent, we follow that gradient to modify an initial curve C_0. We emphasize that by construction, only local minima are found. No global search is performed.

Assume that C is a function that minimizes E. This implies that small variations to C should not change E significantly, and should increase it. Let $\eta(p)$ be a test function. Consider adding $\epsilon\eta(p)$ to $C(p)$ for small ϵ. Assume that $\eta(0) = \eta(1)$ for consistency with C. If E varied linearly with ϵ, then we could subtract $\epsilon\eta(p)$ from $C(p)$ to decrease the energy, which contradicts the assumption that C minimizes E. Therefore E cannot vary linearly with ϵ, so it must increases with ϵ^2. Since E cannot vary linearly with ϵ at $C(p)$, we must have that the first derivative of E with respect to ϵ at $C(p)$, equivalently, "at $\epsilon = 0$, is zero. That is, for all test functions $\eta(p)$,

$$\left.\frac{\partial E}{\partial \epsilon}\right|_{\epsilon=0} = 0.$$

Modifying C by $\epsilon\eta$, we obtain the energy equation

$$E = \int_0^1 \Phi(p, C + \epsilon\eta, C' + \epsilon\eta')dp.$$

In order to derive the Euler-Lagrange equations, one expands $\Phi(p, C + \epsilon\eta, C' + \epsilon\eta')$

in Taylor series around $\epsilon = 0$ to obtain

$$\Phi(p, C + \epsilon\eta, C' + \epsilon\eta') = \tag{1}$$

$$\Phi(p, C, C') + \frac{\partial}{\partial C}\Phi(p, C, C')\epsilon\eta(p) + \frac{\partial}{\partial C'}\Phi(p, C, C')\epsilon\eta'(p) + e \tag{2}$$

where e contains second-order and higher powers of ϵ.

The energy equation is now

$$E = \int_0^1 (\Phi + \epsilon\eta(p)\Phi_C + \epsilon\eta'(p)\Phi_{C'} + e)dp.$$

By above, the derivative of E with respect to ϵ, evaluated at $\epsilon = 0$, must be zero in order for C to solve the minimization. Thus, we differentiate E with respect to ϵ, choose $\epsilon = 0$, and set the result to zero:

$$0 = \int_0^1 (\eta(p)\Phi_C + \eta'(p)\Phi_{C'})dp = \int_0^1 \eta(p)\Phi_C dp + \int_0^1 \eta'(p)\Phi_{C'} dp. \tag{3}$$

We then use integration by parts to remove reference to $\eta'(p)$:

$$\int_0^1 \eta'(p)\Phi_{C'}dp = [\eta(p)\Phi_{C'}]_0^1 - \int_0^1 \eta(p)\frac{\partial}{\partial p}\Phi_{C'}dp.$$

Because $\eta(0) = \eta(p)$ and $C(0) = C(1)$, we have that $[\eta(p)\Phi_{C'}]_0^1 = 0$. Substituting into Equation 3, we obtain

$$0 = \int_0^1 \eta(p)\Phi_C dp - \int_0^1 \eta(p)\frac{\partial}{\partial p}\Phi_{C'}dp = \int_0^1 \eta(p)(\Phi_C - \frac{\partial}{\partial p}\Phi_{C'})dp.$$

For this to hold for all test functions $\eta(p)$, then

$$\Phi_C - \frac{\partial}{\partial p}\Phi_{C'} = 0. \tag{4}$$

Equation 4 is called the *Euler-Lagrange equation* or the *Euler equation* for the given minimization problem. The Euler-Lagrange equations for the situation where the integrand contains higher order derivatives of C [51, 37] are as follows. At a local minima of

$$E = \int_0^1 \Phi(p, C, C', C'', \ldots)dp,$$

one finds that

$$\Phi_C - \frac{\partial}{\partial p}\Phi_{C'} + \frac{\partial^2}{\partial p^2}\Phi_{C''} - \ldots = 0. \qquad (5)$$

After one obtains this condition for a local minima, one makes C a function of time t as well as spatial position p and sets the gradient to the negative time derivative of C to yield the evolution equation. We next provide the calculations of Euler-Lagrange equations for the problems of minimizing Euclidean curve length and non-Euclidean curve length, as in geodesic snakes.

Curve-Shortening Flow

To minimize curve length

$$\int_0^1 |C'(p)|dp,$$

we set $\Phi(p, C, C') = |C'(p)|$. Differentiating,

$$\Phi_C = 0$$

$$\Phi_{C'} = \frac{C'(p)}{|C'(p)|}$$

$$\frac{\partial}{\partial p}\Phi_{C'} = \frac{\partial}{\partial p}\left(\frac{C'(p)}{|C'(p)|}\right).$$

Let s denote arclength of C, $\vec{t}$ the unit tangent vector of C, and $\vec{N}$ the unit normal vector of C. Then the curvature κ of C is defined according to the Frenet equation

$$\frac{\partial \vec{t}}{\partial s} = \kappa \vec{N}.$$

Observe that $\frac{C'(p)}{|C'(p)|} = \vec{t}$ is the unit tangent to C. The chain rule gives

$$\frac{\partial}{\partial p}\left(\frac{C'(p)}{|C'(p)|}\right) = \frac{\partial}{\partial s}\left(\frac{C'(p)}{|C'(p)|}\right)\frac{\partial s}{\partial p}.$$

$\frac{\partial s}{\partial p}$ is the speed of the parametrization p, which is also equal to $|C'(p)|$. Thus,

$$\frac{\partial}{\partial p}\left(\frac{C'(p)}{|C'(p)|}\right) = \kappa \vec{N}|C'(p)|$$

We find that the Euler-Lagrange equation is

$$0 = -\kappa\vec{N}|C'(p)|$$

Dividing by $|C'(p)|$ and setting the temporal derivative of C to the negative of this quantity for minimization gives Equation 2.2

$$\bar{C}_t = \kappa\bar{N}. \tag{6}$$

Geodesic Active Contours

To minimize

$$\int_0^1 g(|\nabla I(C(p))|)|C'(p)|dp,$$

we set $\Phi(p, C, C') = g(|\nabla I(C(p))|)|C'(p)|$. Differentiating,

$$\Phi_C = |C'(p)|g'\frac{\nabla I}{|\nabla I|}H$$

$$\Phi_{C'} = g\frac{C'(p)}{|C'(p)|}$$

$$\frac{\partial}{\partial p}\Phi_{C'} = g\frac{\partial}{\partial p}\left(\frac{C'(p)}{|C'(p)|}\right) + \frac{C'(p)}{|C'(p)|}\frac{\partial}{\partial p}(g).$$

By the chain rule application above,

$$\frac{\partial}{\partial p}\left(\frac{C'(p)}{|C'(p)|}\right) = \kappa\vec{N}|C'(p)|$$

We find that the Euler-Lagrange equation is

$$0 = |C'(p)|g'\frac{\nabla I}{|\nabla I|}H - g\kappa\vec{N}|C'(p)| - \frac{C'(p)}{|C'(p)|}\frac{\partial}{\partial p}(g)$$

The final term is a vector in the direction of the tangent to the curve. It would give an amount to move the curve in the direction of the tangent. Since we only care about the curve as a boundary between regions of the image, and do not care about the parametrization of the curve, we can ignore this motion. Thus, that term is omitted. We also remove the tangential motion from $g'\frac{\nabla I}{|\nabla I|}H$ by replacing it with its component in the direction of the normal to the curve. We thus obtain

$$0 = |C'(p)|((g'\frac{\nabla I}{|\nabla I|}H) \cdot \vec{N})\vec{N} - g\kappa\vec{N}|C'(p)|.$$

Equivalently,

$$0 = ((g'\frac{\nabla I}{|\nabla I|}H) \cdot \vec{N})\vec{N} - g\kappa\vec{N}. \tag{7}$$

Observe that $g'\frac{\nabla I}{|\nabla I|}H = \nabla g$ and write

$$0 = (\nabla g \cdot \vec{N})\vec{N} - g\kappa\vec{N}$$

To achieve this minima by an iterative evolution using gradient descent, we set the time derivative of C to the negative of this expression to obtain:

$$C_t = g\kappa\vec{N} - (\nabla g \cdot \vec{N})\vec{N}.$$

Bibliography

[1] D. Adalsteinsson and J.A. Sethian. A fast level set method for propagating interfaces. *Journal of Computational Physics*, 118:269–277, 1995.

[2] D. Adalsteinsson and J.A. Sethian. The fast construction of extension velocities in level set methods. *Journal of Computational Physics*, 148:2–22, 1999.

[3] S. Altschuler and M. Grayson. Shortening space curves and flow through singularities. *Journal of Differential Geometry*, 35:283–298, 1992.

[4] Luigi Ambrosio and Halil M. Soner. Level set approach to mean curvature flow in arbitrary codimension. *J. of Diff. Geom.*, 43:693–737, 1996.

[5] A. Amini, T. Weymouth, and R. Jain. Using dynamic programming for solving variational problems in vision. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 9:855–867, 1990.

[6] G. Aubert, R. Deriche, and P. Kornprobst. Computing optical flow via variational techniques. *SIAM Journal on Applied Mathematics*, to appear, 2000.

[7] S. Aylward, S.M. Pizer, E. Bullitt, and D. Eberly. Intensity ridge and widths for 3d object segmentation and description. In *IEEE Proc. Workshop Mathematical Models Biomedical Image Analysis*, pages 131–138, 1996.

[8] C.J.G. Bakker, R.M. Hoogeveen, and M.A. Viergever. Construction of a protocol for measuring blood flow by two-dimensional phase-contrast MRA. *Annals of Biomedical Engineering*, 9:119–127, 1999.

[9] M. Bardi, M.G. Crandall, L.C. Evans, H.M. Soner, and P.E. Souganidis. *Lecture Notes in Mathematics: Viscosity Solutions and Applications.* Springer-Verlag, Berlin Heidelberg, 1997.

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

[11] G. Bertrand and G. Malandain. A new characterization of three-dimensional simple points. *Pattern Recognition Letters*, 15:169–175, 1994.

[12] Andrew Blake and Michael Isard. *Active Contours.* Berlin, Heidelberg New York: Springer-Verlag, 1998.

[13] H. Blum. A transformation for extracting new descriptors of shape. In *Models of the perception of Speech and Visual Form.* MIT Press, Cambridge, MA, 1967.

[14] G. Borgefors. Distance transformations in digital images. *CVGIP: Image Understanding*, 34:344–371, 1986.

[15] E. Bullitt, S. Aylward, A. Liu, J. Stone, S.K. Mukherji, C. Coffey, G. Gerig, and S. Pizer. 3d graph description of the intracerebral vasculature from segmented MRA and tests of accuracy by comparison with x-ray angiograms. In *Int'l Conf. Information Processing in Medical Imaging*, pages 308–321. Springer-Verlag, 1999.

[16] E. Bullitt, A. Liu, S. Aylward, and S. Pizer. Reconstruction of the intracerebral vasculature from mra and a pair of projection views. In *Int'l Conf. Information Processing in Medical Imaging*, pages 537–542. Springer-Verlag, 1997.

[17] V. Caselles, F. Catte, T. Coll, and F. Dibos. A geometric model for active contours. *Numerische Mathematik*, 66:1–31, 1993.

[18] V. Caselles, R. Kimmel, G. Sapiro, and C. Sbert. Minimal surfaces based object segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 9(4):394–398, 1997.

[19] V. Caselles, J.M. Morel, G. Sapiro, and A. Tannenbaum. Introduction to the special issue on partial differential equations and geometry-driven diffusion in image processing and analysis. *IEEE Transactions on Image Processing*, 7(3):269–273, 1998.

[20] Vicent Caselles, Ron Kimmel, and Guillermo Sapiro. Geodesic active contours. *Int'l Journal Comp. Vision*, 22(1):61–79, 1997.

[21] Sharat Chandran, Tsukasa Maejima, and Sanae Miyaziki. Global minima via dynamic programming: Energy minimizing active contours. *SPIE*, 1570:391–402, 1991.

[22] Y.G. Chen, Y. Giga, and S. Goto. Uniqueness and existence of viscosity solutions of generalized mean curvature flow equations. *J. Differential Geometry*, 33:749–786, 1991.

[23] D.L. Chopp. Computing minimal surfaces via level set flow. *Journal of Computational Physics*, 106:77–91, 1993.

[24] A. Chung and J.A. Noble. Statistical 3d vessel segmentation using a Rician distribution. In *Proc. Medical Image Conference and Computer Assisted Interventions (MICCAI)*, pages 82–89, 1999.

[25] L. D. Cohen and R. Kimmel. Global minimum for active contours models: A minimal path approach. In *Proc. IEEE Conf. Comp. Vision and Pattern Recognition*, pages 666–673, 1996.

[26] Laurent D. Cohen. On active contour models and balloons. *CVGIP: Image Understanding*, 53(2):211–218, 1991.

[27] Laurent D. Cohen and I. Cohen. Finite element methods for active contour models and balloons for 2d and 3d images. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 15(11):1131–1147, 1993.

[28] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Algorithms for clustering data.* The MIT Press, 1992.

[29] M.G. Crandall, H. Ishii, and P.L. Lions. User's guide to viscosity solutions of second order partial differential equations. *Bulletin of the American Math. Society*, 27:1–67, 1992.

[30] M.G. Crandall and P.L. Lions. Viscosity solutions of Hamilton-Jacobi equations. *Trans. American Math. Society*, 277:1–42, 1983.

[31] Olivier Cuisenaire. *Distance transformations: fast algorithms and applications to medical image processing.* Ph.D. Thesis, UCL, Louvain-la-Neuve, Belgium, 1999.

[32] Olivier Cuisenaire and Benot Macq. Fast euclidean morphological operators using local distance transformation by propagation. In *Proc. Int'l Conf. Image Processings and its Applications*, pages 856–860, 1999.

[33] P-E. Danielsson. Euclidean distance mapping. *Computer Graphics and Image Processing*, 14:227–248, 1980.

[34] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[35] M. P. DoCarmo. *Differential Geometry of Curves and Surfaces.* Prentice-Hall, 1976.

[36] P. Dokládal, C. Lohou, L. Perroton, and G. Bertrand. Liver blood vessel extraction by a 3-d topological approach. In *Proc. Medical Image Conference and Computer Assisted Interventions (MICCAI)*, pages 98–105. Springer-Verlag, Berlin Heidelberg, 1999.

[37] B.A. Dubrovin, A.T. Fomenko, and S.P. Novikov. *Modern Geometry - Methods and Applications I.* Springer-Verlag, New York, 1984.

[38] L.C. Evans and J. Spruck. Motion of level sets by mean curvature: I. *Journal of Differential Geometry*, 33:635–681, 1991.

[39] R. Fahrig, H. Nikolov, A.J. Fox, and D.W. Holdsworth. A three-dimensional cerebrovascular flow phantom. *Medical Physics*, 26(8):1589–1599, 1999.

[40] Olivier D. Faugeras and Renaud Keriven. Variational principles, surface evolution, PDEs, level set methods, and the stereo problem. *IEEE Trans. Image Processing*, 7(3):336–344, 1998.

[41] A. Frangi, W. J. Niessen, R.M. Hoogeveen, Th. Van Walsum, and M. A. Viergever. Quantitation of vessel morphology from 3D MRA. In *Proc. Medical Image Conference and Computer Assisted Interventions (MICCAI)*, pages 358–367, 1999.

[42] A. Frangi, W. J. Niessen, K. L. Vincken, and M. A. Viergever. Vessel enhancement filtering. In *Proc. Medical Image Conference and Computer Assisted Interventions (MICCAI)*, pages 130–137, 1998.

[43] D.S. Fritsch, D. Eberly, S.M. Pizer, and M.J. McAuliffe. Stimulated cores and their applications in medical imaging. In *Int'l Conf. Information Processing in Medical Imaging*, pages 365–368. Springer-Verlag, 1995.

[44] M. Gage and R.S. Hamilton. The heat equation shrinking convex plane curves. *J. of Differential Geometry*, 23:69–96, 1986.

[45] J. Gomes and O. Faugeras. Reconciling distance functions and level sets. In *Proc. Int'l Conf. Scale-Space*, pages 70–81, 1999.

[46] M. Grayson. The heat equation shrinks embedded plane curves to round points. *J. of Differential Geometry*, 26:285–314, 1987.

[47] W.E.L. Grimson, G.J. Ettinger, T. Kapur, M.E. Leventon, W.M. Wells III, and R. Kikinis. Utilizing segmented MRI data in image-guided surgery. *Int'l Journal of Pattern Recognition and Artificial Intelligence*, 1996.

[48] W.E.L. Grimson, C. Stauffer, R. Romano, and L. Lee. Using adaptive tracking to classify and monitor activities in a site. In *Proc. IEEE Conf. Comp. Vision and Pattern Recognition*, pages 22–29, 1998.

[49] Yanlin Guo and Baba C. Vemuri. Hybrid geometric active models for shape recovery in medical images. In *Int'l Conf. Information Processing in Medical Imaging*, pages 112–125. Springer-Verlag, 1999.

[50] R.M. Hoogeveen, C.J.G. Bakker, and M.A. Viergever. Limits to the accuracy of vessel diameter measurement in MR angiography. *Journal of Magnetic Resonance Imaging*, 8:1228–1235, 1998.

[51] B. K. P. Horn. *Robot Vision.* MIT Press, Cambridge, MA, 1986.

[52] J. Hug, C. Brechbühler, and G. Székely. Tamed snake: A particle system for robust semi-automatic segmentation. In *Proc. Medical Image Conference and Computer Assisted Interventions (MICCAI)*, pages 106–115, 1999.

[53] Anil K. Jain and Richard C. Dubes. *Algorithms for clustering data.* Prentice-Hall, Inc., Englewood Cliffs, NJ, 1988.

[54] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *Int'l Journal Comp. Vision*, 1(4):321–331, 1988.

[55] Renaud Keriven. *Equations aux Dérivées Partielles, Evolutions de Courbes et de Surfaces et Espaces d'Echelle: Applications á la Vision par Ordinateur.* Ph.D. Thesis, L'École Nationale des Ponts et Chaussées, France, 1997.

[56] A. Kichenassamy, A. Kumar, P. Olver, A. Tannenbaum, and A. Yezzi. Gradient flows and geometric active contour models. In *Proc. IEEE Int'l Conf. Comp. Vision*, pages 810–815, 1995.

[57] Th. Koller, G. Gerig, and G. Székely. Modelling and interaction for superior analysis of anatomical structures in medical imaging. TR 156, Swiss Federal Institute of Technology (ETH), Communication Technology Laboratory, Image Science, 1995.

[58] P. Kornprobst, R. Deriche, and G. Aubert. Image coupling, restoration, and enhancement via PDE's. In *Int'l Conf. Pattern Recognition*, pages 458–461, 1997.

[59] P. Kornprobst, R. Deriche, and G. Aubert. Image sequence restoration: a PDE based coupled method for image restoration and motion segmentation. In *Proc. European Conf. Comp. Vision, Lect. Notes Comp. Sci. 800*, pages 548–562, 1998.

[60] K. Krissian, G. Malandain, and N. Ayache. Directional anisotropic diffusion applied to segmentation of vessels in 3d images. In *Proc. Int'l Conf. Scale-Space*, pages 345–348, 1997.

[61] K. Krissian, G. Malandain, N. Ayache, R. Vaillant, and Y. Trousset. Model based multiscale detection of 3d vessels. *Proc. IEEE Conf. Comp. Vision and Pattern Recognition*, pages 722–727, 1998.

[62] Karl Krissian, Grégoire Malandain, and Nicolas Ayache. Model based detection of tubular structures in 3d images. Technical Report 3736, INRIA, Sophia Antipolis, France, 1999.

[63] Arun Kumar, Allen R. Tannenbaum, and Gary J. Balas. Optical flow: a curve evolution approach. *IEEE Trans. Image Processing*, 5(4):598–610, 1996.

[64] Michael Leventon, Olivier Faugeras, W. E. L. Grimson, and William Wells. Level set based segmentation with intensity and curvature priors. In *IEEE Proc. Workshop Mathematical Models Biomedical Image Analysis*, 2000.

[65] Michael Leventon, W. E. L. Grimson, and Olivier Faugeras. Statistical shape influence in geodesic active contours. *Proc. IEEE Conf. Comp. Vision and Pattern Recognition*, 2000.

[66] F. Leymarie and M.D. Levine. Faster raster scan distance propagation on the discrete rectangular lattice. *CVGIP: Image Understanding*, 55(1):84–94, 1992.

[67] F. Leymarie and M.D. Levine. Simulating the grassfire transform using and active contour model. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 14(1):56–75, 1992.

[68] J. Liang, T. McInerney, and D. Terzoloulos. Interactive medical image segmentation with united snakes. In *Proc. Medical Image Conference and Computer Assisted Interventions (MICCAI)*, pages 116–127. Springer-Verlag, Berlin Heidelberg, 1999.

[69] T. Lindeburg. Feature detection with automatic scale selection. *Int'l Journal Comp. Vision*, 30(2):79–116, 1998.

[70] T.L. Liu, D. Geiger, and R.V. Kohn. Representation and self-similarity of shapes. In *Proc. IEEE Int'l Conf. Comp. Vision*, pages 1129–1135, 1999.

[71] W.E. Lorensen and H.E. Cline. Marching cubes: a high resolution 3d surface construction algorithm. In *Proceedings of the SIGGRAPH '87 Conference*, volume 21, pages 163–170, Anaheim, California, 1987.

[72] C. Lorenz, I.-C. Carlsen, T.M. Buzug, C. Fassnacht, and J. Weese. A multiscale line filter with automatic scale selection based on the hessian matrix for medical image segmentation. In *Proc. Int'l Conf. Scale-Space*, pages 152–163, 1997.

[73] Liana M. Lorigo, Olivier Faugeras, W.E.L. Grimson, Renaud Keriven, and Ron Kikinis. Segmentation of bone in clinical knee MRI using tecture-based geodesic active contours. In *Proc. Medical Image Conference and Computer Assisted Interventions (MICCAI)*, pages 1195–1204, 1998.

[74] Liana M. Lorigo, Olivier Faugeras, W.E.L. Grimson, Renaud Keriven, Ron Kikinis, Arya Nabavi, and Carl-Fredrik Westin. Codimension-two geodesic active contours. In *Proc. IEEE Conf. Comp. Vision and Pattern Recognition*, 2000.

[75] Liana M. Lorigo, Olivier Faugeras, W.E.L. Grimson, Renaud Keriven, Ron Kikinis, and Carl-Fredrik Westin. Co-dimension 2 geodesic active contours for MRA segmentation. In *Int'l Conf. Information Processing in Medical Imaging*, pages 126–139. Springer-Verlag, 1999.

[76] G. Malandain and G. Bertrand. Fast characterization of 3d simple points. *Int'l Conf. Pattern Recognition*, 3:232–235, 1992.

[77] G. Malandain and S. Fernández-Vidal. Euclidean skeletons. *Image and Vision Computing*, 16:317–327, 1999.

[78] R. Malladi, J.A. Sethian, and B.C. Vemuri. Evolutionary fronts for topology-independent shape modeling and recovery. In *Proc. European Conf. Comp. Vision, Lect. Notes Comp. Sci. 800*, pages 3–13, 1994.

[79] R. Malladi, J.A. Sethian, and B.C. Vemuri. Shape modeling with front propagation: a level set approach. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 17:158–175, 1995.

[80] T. McInerney and D. Terzopoulos. Topologically adaptable snakes. In *Proc. IEEE Int'l Conf. Comp. Vision*, pages 840–845, 1995.

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

[82] T. McInerney and D. Terzopoulos. Medical image segmentation using topologically adaptable surfaces. In *Proc. First Joint Conf. Computer Vision, Virtual Reality, and Robotics in Medicine and Medical Robotics and Computer-Assisted Surgery (CVRMed-MRCAS)*, pages 23–32. Springer-Verlag, Berlin, 1997.

[83] T. McInerney and D. Terzopoulos. Topology adaptive snakes. *Medical Image Analysis*, 1999.

[84] S. Menet, P. Saint-Marc, and G.G. Medioni. Active contour models: Overview, implementation and applications. In *IEEE Transactions on Systems, Man, and Cybernetics*, pages 194–199, 1990.

[85] J.A. Moore, D.A. Steinman, D.W. Holdsworth, and C.R. Ethier. Accuracy of computational hemodynamics in complex geometries reconstructed from MRI. *Annals of Biomedical Engineering*, 27(1):32–41, 1999.

[86] D.G. Morgenthaler. Three dimensional simple points: serial erosion, parallel thinning, and skeletonization. Technical Report TR-1005, Computer Vision Laboratory, Computer Science Center, Univ. of Maryland, College Park, 1981.

[87] W. Mulder, S.J. Osher, and J.A. Sethian. Computing interface motion in compressible gas dynamics. *Journal Computational Physics*, 100, 1992.

[88] S. Osher and J. Sethian. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulation. *Journal of Computational Physics*, 79(1):12–49, 1988.

[89] Nikos Paragios and Rachid Deriche. Geodesic active contours and level sets for detection and tracking of moving objects. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 1999.

[90] Nikos Paragios and Rachid Deriche. Geodesic active regions for supervided texture segmentation. In *Proc. IEEE Int'l Conf. Comp. Vision*, 1999.

[91] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 12(7):629–639, May 1990.

[92] V. Prinet, O. Monga, C. Ge, L.X. Sheng, and S.D. Ma. Thin network extraction in 3d images: application to medical angiograms. *Int'l Conf. Pattern Recognition*, 3:386–390, 1996.

[93] E. Rouy and A. Tourin. *SIAM Journal of Numerical Analysis*, 29(3), 1992.

[94] William James Rucklidge. *Efficient computation of the minimum Hausdorff distance for visual recognition.* Ph.D. Thesis, Cornell University, Ithaca, NY, 1995.

[95] S.J. Ruuth, B Merriman, J. Xin, and S. Osher. Diffusion-generated motion by mean curvature for filaments. Technical Report 98–47, UCLA Computational and Applied Mathematics, November 1998.

[96] Guillermo Sapiro. Vector-valued active contours. In *Proc. IEEE Conf. Comp. Vision and Pattern Recognition*, pages 680–685, 1996.

[97] Y. Sato, S. Nakajima, N. Shiraga, H. Atsumi, S. Yoshida, T. Koller, G. Gerig, and R. Kikinis. Three-dimensional multi-scale line filter for segmentation and visualization of curvilinear structures in medical images. *Medical Image Analysis*, 2(2):143–168, 1998.

[98] W. Schroeder, K. Martin, and B. Lorensen. *The Visualization Toolkit: an object oriented approach to 3D graphics.* Prentice Hall PTR, Upper Saddle River, New Jersey, 1998.

[99] J. Sethian. Curvature flow and entropy conditions applied to grid generation. *Journal of Computational Physics*, 115:440–454, 1994.

[100] J. A. Sethian. *Level Set Methods.* Cambridge University Press, 1996.

[101] K. Siddiqi, Y. B. Lauzière, and A. Tannenbaum. Area and length minimizing flows for shape segmentation. *IEEE Trans. Image Processing*, 7(3):433–443, March 1998.

[102] Michael Spivak. *A Comprehensive Introduction to Differential Geometry*, volume 1–5. Publish or Perish, Berkeley, CA, 1979. Second edition.

[103] L.H. Staib and J.S. Duncan. Boundary finding with parametrically deformable contour models. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 14(11), 1992.

[104] M. Sussman, P. Smereka, and S.J. Osher. A level set method for computing solutions to incompressible two-phase flow. *Journal of Computational Physics*, 114:146–159, 1994.

[105] H. Tek and B. Kimia. Image segmentation by reaction-diffusion bubbles. In *Proc. IEEE Int'l Conf. Comp. Vision*, pages 156–162, 1995.

[106] D. Terzopoulos, A. Witkin, and M. Kass. Constraints on deformable models: recovering 3d shape and nonrigid motion. *Artificial Intelligence*, 36(1):91–123, 1988.

[107] Jean-Philippe Thirion. New feature points based on geometric invariants for 3d image registration. *Int'l Journal Comp. Vision*, 18(2):121–137, 1996.

[108] Carl-Fredrik Westin, Abhir Bhalerao, Hans Knuttson, and Ron Kikinis. Using local 3D structure for segmentation of bone from computer tomography images. In *Proc. IEEE Conf. Comp. Vision and Pattern Recognition*, pages 794–800, 1997.

[109] D. Wilson and J.A. Noble. Segmentation of cerebral vessels and aneurysms from MR angiography data. In *Int'l Conf. Information Processing in Medical Imaging*, pages 423–428. Springer-Verlag, 1997.

[110] M. Gazi Yasargil. *Microneurosurgery.* Thieme Medical Publishers, Incorporated, 1987.

[111] Anthony Yezzi, Satyanad Kichenassamy, Arun Kumar, Peter Olver, and Allen Tannenbaum. A geometric snake model for segmentation of medical imagery. *IEEE Trans. Medical Imaging*, 16(2):199–209, 1997.

[112] Anthony Yezzi Jr., Andy Tsai, and Alan Willsky. A statistical approach to snakes for bimodal and trimodal imagery. In *Proc. IEEE Int'l Conf. Comp. Vision*, pages 898–903, 1999.

[113] H-K. Zhao, T. Chan, B. Merriman, and S. Osher. A variational level set approach to multiphase motion. *Journal Computational Physics*, 127:179–195, 1996.

[114] B.V. Zlokovic and M.L.J. Apuzzo. Strategies to circumvent vascular barriers of the central nervous system. *Neurosurgery*, 43(4):877, 1998.