A VARIATIONAL APPROACH FOR VIEWPOINT-BASED VISIBILITY MAXIMIZATION

A Dissertation

Presented to

The Academic Faculty

By

Kelvin R. Rocha

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy in the

School of Electrical and Computer Engineering

Georgia Institute of Technology

August 2008

A VARIATIONAL APPROACH FOR VIEWPOINT-BASED VISIBILITY MAXIMIZATION

Committee:

Dr. Anthony J. Yezzi Jr., Advisor
School of Electrical and
Computer Engineering
Georgia Institute of Technology

Dr. Joel R. Jackson
School of Electrical and
Computer Engineering
Georgia Institute of Technology

Dr. Allen R. Tannenbaum
School of Electrical and
Computer Engineering
Georgia Institute of Technology

Dr. Gregory Turk
College of Computing
Georgia Institute of Technology

Dr. Patricio A. Vela
School of Electrical and
Computer Engineering
Georgia Institute of Technology

Date Approved: March 26, 2008

Dedicated to:

My parents, Albania Arias and Carmelo Rocha.

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor, Dr. Anthony Yezzi, for being such a great mentor during all these years at Georgia Tech. It has been really an honor for me to work for him. Not only has he been very patient with me, but also he has allowed me to participate in many interesting projects from which I have learned a lot.

I would also like to thank Professors Allen Tannenbaum and Patricio Vela for being part of my reading committee and sharing their comments. Thanks also to Professors Joel Jackson and Gregory Turk for being part of the dissertation defense committee.

Special thanks to Professor Jerry Prince and to Blake Lucas and Aaron Carass at Johns Hopkins University for their contributions to many of the projects I have worked on.

Thanks to Professor Stefano Soatto at UCLA and Researcher Andrea Mennucci at Scuola Normale Superiore for sharing their ideas on harmonic embedding and visibility maximization.

Thanks to Professors Todd Moon, Jacob Gunther, and YangQuan Chen for helping me so much when I was in the Masters program at Utah State University. I'm particularly indebted to my Masters advisor, Professor Tamal Bose (now at Virginia Tech), for being such a great source of inspiration.

Thanks to Georgia Tech for believing in me and giving me the opportunity of being part of their excellent program. Thanks also to the Organization of American States for giving me financial support during my first two years at Georgia Tech throughout their Graduate Scholarship Program for the Americas.

I would like to sincerely thank the Instituto Politécnico Loyola and the Pontificia Universidad Católica Madre y Maestra in the Dominican Republic for giving me the opportunity of learning in such a great environment. Special thanks to Francisco Polanco and to Professors Eugenio Morel, Adriano Liranzo, Luis Enrique, Masako Saito, Marino

Martínez, René Piedra, and Rosa Elena.

Thanks to all my friends in the lab for making my life better and sharing their ideas: Christopher Alvino, Guillermo Gallego Bonet, Jeremy Jackson, Hua Li, Siddharth Manay, Ganesh Sundaramoorthi, and Namrata Vaswani.

I would also like to thank my friends: Alex Nivar, Ariel Del Rosario, Alejandro García, José Ariel Camilo, Edwin Marte, Franciso Size, Gilberto Silfa, Javier López, Jorge Quevedo, José Ramón, Simeon Sessley, Todd Humphreys, and Vladimir Del Rosario. Special thanks to the families Fernández De La Cruz, Rivera Santana, and Matos Díaz.

Thanks to my beloved wife, Leidy López, and all her family for the support they have given me during the last six years. Special thanks to Doña Yolanda, Don Pedro, Don Bolívar, and Pedrito.

Most of all, thanks to my parents, Albania and Carmelo, and their families and to my sister, Kary Alba.

TABLE OF CONTENTS

LIST OF FIGURES

SUMMARY

We present a variational method for unfolding of the cortex based on a user-chosen point of view as an alternative to more traditional global flattening methods, which incur more distortion around the region of interest. Our approach involves three novel contributions. The first is an energy function and its corresponding gradient flow to measure the average visibility of a region of interest of a surface from a given viewpoint. The second is an additional energy function and flow designed to preserve the 3D topology of the evolving surface. This latter contribution receives significant focus in this thesis as it is crucial to obtain the desired unfolding effect derived from the first energy functional and flow. Without it, the resulting topology changes render the unconstrained evolution uninteresting for the purpose of cortical visualization, exploration, and inspection. The third is a method that dramatically improves the computational speed of the 3D topology-preservation approach by creating a tree structure of the triangulated surface and using a recursion technique.

CHAPTER 1

INTRODUCTION

In this work we present a novel algorithm that maximizes the visibility of a triangulated surface with respect to a fixed external viewpoint by evolving the surface locally. We propose an energy that measures the visibility of a surface with respect to a viewpoint and then we evolve the surface by a gradient along the proposed energy. The result is a continuous unfolding of the surface with respect to the user's external viewpoint that allows the user to view the deeper self-occluded structures of the surface.

Our approach involves three novel contributions. The first is an energy function and its corresponding gradient flow to measure the average visibility of a region of interest on a surface from a given viewpoint. The second is an additional energy function and flow designed to preserve the 3D topology of the evolving surface. This latter contribution is crucial to obtain the desired unfolding effect derived from the first energy function and flow. Without it, the resulting topology changes render the unconstrained evolution uninteresting for the purpose of visualization, exploration, and inspection. The third is a method that dramatically improves the computational speed of the 3D topology-preservation approach by creating a tree structure of the 3D triangulated surface and using a recursion technique.

The proposed approach for visibility maximization could have several applications in the area of medical imaging. For instance, it could be used in human brain mapping to unfold a specific part of the cerebral cortex while introducing little distortion or to visually explore and validate the deep sulcul structures extracted by cortical surface segmentation algorithms. In addition, the proposed approach may also have a number applications in computer graphics. For example, it could be used for the visualization of complicated surfaces at a specific area and for the visual inspection of texture mappings onto complex geometries. Moreover, our novel 3D topology-preservation approach may be useful for applications not related to visibility maximization. For instance, in medical imaging, topology preservation is often used in evolution models for the purpose of segmentation.

This thesis is organized as follows. In Chapter 2 we briefly discuss the history of both the visibility and the surface flattening problem. Some of their main applications are presented together with some of the most recent works in these areas. In Chapter 3 we present the results of research we have done on partial differential equations (PDEs) techniques for cortical shape analysis. Besides having important applications in some areas of medical imaging, these techniques served as the starting point when we first considered the visibility-maximization problem. In Chapter 4, for the sake of simplicity and better insight into the 3D case, we describe our framework for visibility maximization in 2D (*i.e.*, planar contours viewed from an external point by someone positioned in the same plane). We also discuss in this chapter part of our initial investigation on the visibility-maximization problem, including the maximization of the visibility of a surface with respect to another surface. In Chapter 5 we extend our viewpoint-based visibility-maximization approach in a straight-forward manner to the more relevant 3D case and show simulations on both synthetically generated triangulated surfaces as well as triangulated surfaces extracted from segmentations of real data (cerebral cortex). In this chapter we also present a novel 3D topology-preservation method, which is crucial for our visibility application. Finally, in Chapter 6 we provide a summary of our contributions and discuss future work.

CHAPTER 2

ORIGIN AND HISTORY OF THE PROBLEM

Visibility-maximizing flows have never been attempted before. As a result, we cannot provide or discuss any previous work truly related to the main focus of this thesis. Nevertheless, in this chapter we present a brief introduction to both visibility computation and surface flattening, which are in some degree related to our work.

2.1 Visibility

In the more general definition of visibility, two points are considered to be mutually visible if the line segment that connects them is not occluded. Therefore, the problem of visibility consists of finding the point or set of points that is visible from another point or set of points.

A wide variety of visibility-related problems appear in many different research areas such as computer graphics, computer vision, computational geometry, and robotics, just to mention a few [1]. Applications of visibility include shadow computations [2], rendering [3], etching [4], object recognition [5], and many others.

The visibility problem has been of particular interest in computer graphics since the early days of the field. For instance, the classical problem of hidden surface removal (HSR), also called visible surface determination (VSD), is to determine which parts of an object are invisible from a given point, called the viewpoint, so that they need not be rendered, thereby reducing the number computations necessary to process the given object [6].

A number of HSR algorithms were developed in the late '60s and early '70s [7, 8]. Some of these algorithms, such as the one proposed by Arvo and Kirk [9], use a technique called ray shooting to compute visibility along a single ray. This approach computes the first intersection of a given ray with an object in the scene. Ray shooting is still used today in global illumination methods [10].

Another HSR algorithm, called the z-buffer, was proposed by Catmull [11, 12]. The z-buffer has been very popular and, together with its later variations, is commonly used today in interactive applications because of its simplicity and robustness [10]. In Catmull's algorithm a depth value is stored for each pixel of the image. If the point being scan-converted is not farther from the viewer than the point whose depth is currently in the z-buffer, then the new point's depth replaces the old value.

The construction of the visibility map is another type of visibility from a point problem [13]. It is used to describe the topology of the view of the scene. One of its applications includes visibility culling, a technique used to reject, in a fast way, invisible parts of the object being viewed before the actual HSR algorithm is applied. Accordingly, the computational cost of processing the image is reduced since just an estimate of the visible surface has to be analyzed. Several algorithms such as those in [14, 15, 16, 17] have been proposed to produce efficient visibility culling.

Visibility has also been used in mesh simplification algorithms so that the polygon count of a model can be reduced while maintaining the overall shape and appearance of the model, thus reducing the model storage cost and later processing time [18]. In the work proposed by Zhang and Turk [18], the visibility of a point in the surface of an object is computed as the weighted percentage of the cameras, in a surrounding sphere of cameras (camera space), from which the point is visible. The dot products between the viewing directions and the surface normal at the point are used to weight the camera space. This visibility measure is then used in a mesh simplification procedure.

Sethian [19] presented a novel idea to determine whether a point in the scene is visible from a given point by using the minimal geodesic distance between the points, that is the length of the shortest path between the two points in the presence of obstacles. In Sethian's approach, the geodesic distance, which is computed by solving the Eikonal equation, is compared to the Euclidean distance and visibility is determined by how numerically close those distances are. This approach may be suitable for applications in which the surfaces

are implicitly represented. However, Tsitsiklis [20] showed that Sethian's approach can be very expensive, computationally speaking, and prone to problems of accuracy.

Another novel approach for computing visibility within an implicit framework was presented by Tsai *et al.* [21]. In this approach, which was a continuation of the work in [22], the surface of the objects in the scene are represented as the zero level set of a function and the visibility problem is formulated as a boundary value problem for a first-order partial differential equation in such a way that a proposed algorithm is able to construct the interface separating the visible parts from the invisible ones. The approach is then extended to the case in which the source of light moves and the dynamics of shadow boundaries are analyzed.

Because of the ever increasing amount of information of 3D models, most classical visibility algorithms are not suitable for real-time applications. As a result, new approaches in different areas, such as those in [1, 14, 23, 24, 25, 26, 27], have been developed over the years to overcome this problem.

A number of excellent surveys on visibility and its application have been published [1, 8, 10, 28, 29] to which we refer the reader for a comprehensive treatment of the visibility phenomenon.

2.2 Surface Flattening

Surface flattening is a procedure in which a 3D surface is mapped into a 2D plane for the purpose of visualization and study of the given surface. If the mapping is carried out by simply projecting the 3D surface onto a viewing plane, then some parts of the surface would hide others, which would not be desirable. Instead, surface flattening attempts to unfold the 3D surface in such a way that the result can be presented in a single planar image.

The origin of the surface-flattening problem dates back to a time long before computers were invented. It had to do with the making of footwear and clothing and was, of

course, solved manually [30]. Nowadays, surface flattening is present in a number of industries and, because of its many applications, is widely studied in several fields such as computational geometry, computer graphics, cartography, computer-aided design (CAD), and medical imaging.

In the textile and footwear industry, for example, surface flattening is used to generate 2D patterns from the 3D clothing. These 2D patterns are expected to produce the original 3D shape once they are sewn together. In fact, one of the first works in surface flattening was proposed by Darwill *et al.* [31] and is related to shoe manufacturing.

Another field where surface flattening has been used for a long time is cartography, where flattening of the sphere is the main problem. Several different methods have been proposed to solve this problem [32, 33]. Some of them focus on preserving angles, while others focus on preserving distances.

In addition, surface flattening has become very popular in medical imaging because it can help to detect anatomic abnormalities. For example, several surface algorithms [34, 35, 36] have been proposed to flatten the computer tomographic (CT) images of the colon surface onto the plane so that polyps, the precursors of cancer, can be detected. Another application was proposed by Angenent *et al.* [37], where a technique is presented to flatten the brain to a sphere so that its geometry can be studied.

In general, most 3D surfaces are non-developable, that is, they cannot be flattened without deformations or cuts. As a result, surface flattening algorithms have to preserve distances and angles as much as possible to minimize distortion. Some surfaces, however, are developable and, consequently, they can be flattened (unfolded) without any distortion. Faux and Pratt [38] showed that a surface is developable when its Gaussian curvature is zero everywhere.

Many different algorithms have been proposed to flatten developable surfaces [39, 40, 41]. In fact, the first algorithms proposed in the area were tailored to solve this particular problem. This is a natural result because the origin of the problem comes from the manual

flattening of this type of surface. Other algorithms, such as those in [42, 43, 44], have been proposed to generate developable surfaces so that the resulting flattened surface has little distortion. Even though these proposed algorithms may differ in the approach they use, in all of them the user has to input two directions and a flattening origin [30].

Another group of algorithms has been proposed to address the issue of flattening non-developable surfaces. Unlike those tailored to the flattening of developable surfaces, these algorithms produce a piecewise flattening. This is the case in [45, 46, 47]. In these approaches the unfolding is stopped if the distortion error is considered unacceptable. In such cases the algorithm is restarted from another point in the 3D surface.

Regardless of the type of 3D surface to be flattened, surface flattening algorithms may differ in the actual approach they use. For instance, in the works of Ma and Lin [48] and Schartz *et al.* [49] an optimization technique is used to minimize a global metric that takes into consideration the geodesic distance between each point and its direct neighbors. This technique, however, has the problem of offering no control on the distribution of the remaining distortions. Other works, such as those by Bennis *et al.* [45] and Zigelman *et al.* in [50], use texture mapping for surface flattening. In the particular case of the work of Bennis *et al.*, the flattening of a region grows around an isoparametric curve selected by hand until a fixed distortion threshold is reached. Then, the isoparametric curves in the surfaces are mapped onto curves in the texture plane in such a way that the geodesic curvature at sample points are preserved. One of the disadvantages of this technique is the generation of surface cuts, which are undesired for the purpose of visualization. Another disadvantage is that distances are not preserved in the flattened surface.

Surface flattening can also be carried out by surface parameterization. In this approach the 3D surface is decomposed into discrete patches that are later piecewise mapped into a 2D plane in such a way that the distortions are minimized via a linear or a non-linear solver. Several methods that use this approach have been proposed [51, 52, 53, 54, 55, 56]. The main difference between them is the way in which they measure and minimize the

distortions introduced by the parameterization [57]. Some of these proposed approaches, such as those proposed by Haker *et al*. [54] and Levy *et al.* [55], use conformal mapping. Among the advantages of using conformal maps is their angle-preserving property, which allows the preservation of the local shape. Another advantage is that they are stable and easy to compute. On the other hand, conformal maps have the disadvantage of sometimes creating a lot of distortion [45].

Another technique for surface flattening is called curved planar reformation [58, 59, 60]. This simple technique extracts and then flattens a developable ruled surface. This approach has the problem of introducing discontinuities at the junction of two ruled surfaces, especially if the 3D surface is highly convoluted [61].

An interesting flattening method for triangulated surfaces was presented by Zhong and Xu [57]. In this approach, winged triangles that share the same edge are unfolded by a wrapping force field. The surface is then forced to collide with a given plane once most winged triangle pairs have been unfolded. Unlike many other surface-flattening methods, this one does not need a one-to-one mapping procedure to obtain the initial flat panel.

Hermosillo *et al.* [62] proposed normalized mean curvature flows together with a tracking framework that can be used to unfold the cerebral cortex via level set methods. This work, which is the the 3D extension of the work by Sapiro and Tannenbaum [63], provides the smoothing of a closed surface without shrinkage.

Several surface flattening methods that evolve a surface for the purpose of visualization have been proposed in the past. However, as we have mentioned, traditional surface flattening techniques are global, with no dependence on the user's viewpoint external to the surface itself, and most of these global techniques (with a few notable exceptions such as [62, 57]) do not produce an evolution of the surface itself that can be interactively halted as soon the desired level of visibility is achieved (which typically does not require a full flattening of the surface). As such, because of both the global nature and the full flattening effects of traditional techniques, much more distortion is incurred than necessary, espe-

cially if the user is interested in viewing only a localized part of the surface. Recently, a more general approach for surface unfolding was proposed by Pons *et al.* [64] in which an application-specific normal motion is used to evolve a cortical surface while a tangential motion is used to preserve its area.

Surface flattening has become a very active field and novel approaches [65, 66, 67] are often proposed. They offer new methods or improvement of old methods to perform flattening as well as to control distortion. The works in [30, 61] provide an excellent overview of the different surface-flattening approaches.

CHAPTER 3

PRELIMINARY PDE TECHNIQUES FOR CORTICAL SHAPE ANALYSIS

In this chapter we include part of the research we have done on cortical shape analysis. Specifically, we present novel techniques for the computation of correspondences between annular surfaces and for the representation of a surface via harmonic embedding. These results, which we published in [68, 84, 70], have important applications in some areas of medical imaging. They also served as the starting point for our initial consideration of the visibility-maximization problem.

3.1 Correspondence Between Annular Tissues

Many parts of the human body have an annular tissue structure composed of two or more quasi-homogeneous tissues nested within one another. For example, the cerebral cortex is composed of gray matter "sandwiched" between white matter on the inside and cerebrospinal fluid on the outside [71, 72, 73, 74]. Another example is the left ventricular myocardium, which is intrinsically annular when viewed from cross-section images such as those obtained using magnetic resonance imaging (MRI) or computed tomography (CT) [75, 76].

A variety of methods have been described and used to compute correspondences within annular regions. Most methods are ad hoc, often manually assisted, and have accuracies that are highly dependent on the boundary shapes and on the person analyzing the images. Jones *et al.* [77] proposed an approach based on solving Laplace's equation in the annular region and defining unique correspondence trajectories running between the boundaries on lines orthogonal to equipotential contours. This approach yields unique correspondences without paradoxes (see [78, 79]), and provides a strong initial basis for making correspondence and gridding of annular regions unique, accurate, and repeatable. The approach by

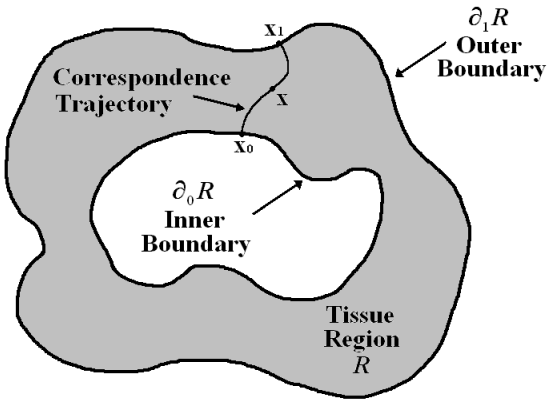

Figure 3.1: Inner and outer boundaries of tissue region R and a correspondence trajectory.

Jones *et al.* is Lagrangian, meaning that paths from one boundary to the other are explicitly traced (using numerical integration of the gradient field of Laplace's solution). Yezzi and Prince [78, 79] later proposed an Eulerian approach in which partial differential equations are solved for the desired correspondences, thereby avoiding the explicit construction or tracing of any correspondence trajectory.

Let $R \subset \mathbb{R}^n$ for $n = 2,3$ be a spatial region with a simply connected inner boundary $\partial_0 R$ and outer boundary $\partial_1 R$ (see Fig. 3.1). These boundaries have a sub-voxel resolution and are usually given as level set representations of some given functions. Now, let u be a harmonic function in R such that $u(\partial_0 R) = 0$ and $u(\partial_1 R) = 1$. The normalized gradient vector field of u coincides with the tangent vector field of the correspondence trajectories and is given by $\overrightarrow{T} = \bigtriangledown u/u$. Figure 3.2 shows the harmonic interpolant u and the tangent vector field $\overrightarrow{T}$ corresponding to the annulus between a circle and an ellipse.

There exist several numerical methods that can be used to compute the harmonic interpolant u given by $\Delta u = 0$ and subject to $u(\partial_0 R) = 0$ and $u(\partial_1 R) = 1$, where Δ is the Laplace operator [80]. In [70] we adapted the method called harmonic embedding proposed by Duci *et al.* [81] to our needs by expressing u as the linear combination of a strategically-chosen set of harmonic functions than can be easily computed. Results show that highly accurate harmonic functions can be obtained by using this novel technique.

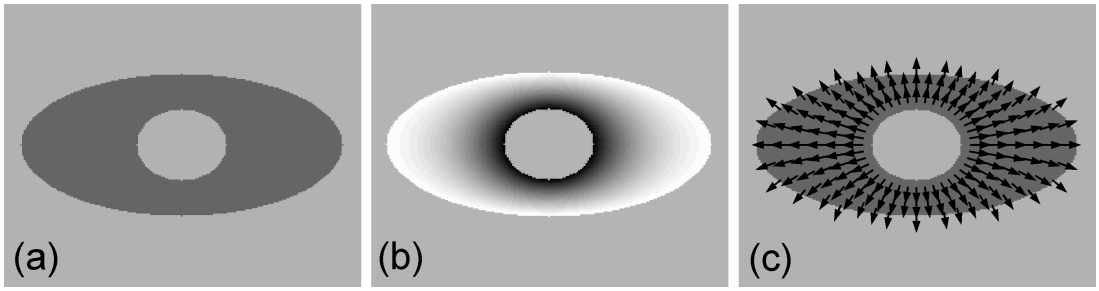

Figure 3.2: Synthetic annular region between and ellipse and a circle. (a) Region. (b) Harmonic interpolant. (c) Tangent field.

It is often useful to find the corresponding boundary points, that is, the points $\mathbf{x}_0 \in \partial_0 R$ and $\mathbf{x}_1 \in \partial_1 R$ such that the correspondence trajectory going from $\mathbf{x}_0$ to $\mathbf{x}_1$ passes through $\mathbf{x}$ (see Fig. 3.1). Since the correspondence trajectories have the property that they do not intersect each other, there is only a pair of points, $\mathbf{x}_0$ and $\mathbf{x}_1$, satisfying this condition. Having unique corresponding boundary points in both $\partial_0 R$ and $\partial_1 R$ for every grid point in R allows the generation of anatomically shaped discrete grids within the tissue region that, among other applications, can be used to subdivide the annular tissue, to create a mesh for finite element analysis, and to elaborate coordinates in which determined functions can be reported [79, 82]. Since for a given $\mathbf{x}$ in R the corresponding boundary points are just the intersections of the constructed trajectory with the inner and the outer boundaries of R, they are readily computed using the Lagrangian approach. However, this approach is computationally intensive, especially when images are large or three dimensional. That is why Yezzi and Prince [79] proposed a novel method to compute correspondences within an Eulerian framework. We now elaborate on this extension.

Let $\phi_0 : \mathbb{R}^n \rightarrow \mathbb{R}^n$ and $\phi_1 : \mathbb{R}^n \rightarrow \mathbb{R}^n$ be defined as the correspondence functions that map $\mathbf{x} \in R$ to the inner and outer boundaries of R, respectively. One of the two conditions that both ϕ_0 and ϕ_1 must satisfy is that they must remain constant along the correspondence trajectories, implying that their directional derivatives vanish along the direction given by $\overrightarrow{T}$. For $n = 3$, we can expand the correspondence functions as follows: $\phi_0(\mathbf{x}) = \left(\phi_0^x(\mathbf{x}), \phi_0^y(\mathbf{x}), \phi_0^z(\mathbf{x})\right)$ and $\phi_1(\mathbf{x}) = \left(\phi_1^x(\mathbf{x}), \phi_1^y(\mathbf{x}), \phi_1^z(\mathbf{x})\right)$. Since all of these func-

tions must remain constant along the direction given by $\overrightarrow{T}$, we have $\phi_i^j \cdot \overrightarrow{T} = 0$ for $i = 0, 1$ and $j = x, y, z$, which can be rewritten in a more compact form as $(\bigtriangledown \phi_0)\,\overrightarrow{T} = (\bigtriangledown \phi_1)\,\overrightarrow{T} = \mathbf{0}$ where $\bigtriangledown \phi_0$ and $\bigtriangledown \phi_1$ denote the Jacobian matrices of ϕ_0 and ϕ_1, respectively. In addition, each correspondence function must map a point on its own boundary to itself as well, which yields another set of conditions, given by $\phi_0(\mathbf{x}) = \mathbf{x}\ \forall \mathbf{x} \in \partial_0 R$ and $\phi_1(\mathbf{x}) = \mathbf{x}\ \forall \mathbf{x} \in \partial_1 R$. As shown by Yezzi and Prince [79], ϕ_0 and ϕ_1 can be computed by using upwind schemes. Specifically, for $\Delta x = \Delta y = \Delta z = 1$ the resulting finite difference equations are

$$\phi_0[i,j,k] \;=\; \frac{|T_x|\,\phi_0[i \mp 1, j, k] + \left|T_y\right|\,\phi_0[i, j \mp 1, k] + |T_z|\,\phi_0[i, j, k \mp 1]}{|T_x| + \left|T_y\right| + |T_z|} \tag{3.1}$$

and

$$\phi_1[i,j,k] \;=\; \frac{|T_x|\,\phi_1[i \pm 1, j, k] + \left|T_y\right|\,\phi_1[i, j \pm 1, k] + |T_z|\,\phi_1[i, j, k \pm 1]}{|T_x| + \left|T_y\right| + |T_z|}, \tag{3.2}$$

where the terms $i \pm 1$, $j \pm 1$, and $k \pm 1$ are as defined in

$$i \pm 1 = \begin{cases} i+1, T_x > 0 \\ i-1, T_x < 0 \end{cases}, j \pm 1 = \begin{cases} j+1, T_y > 0 \\ j-1, T_y < 0 \end{cases}, k \pm 1 = \begin{cases} k+1, T_z > 0 \\ k-1, T_z < 0 \end{cases}. \tag{3.3}$$

These difference equations can be efficiently solved if an iterative algorithm is used, as shown in [78].

The main advantage of the Eulerian approach for correspondences is its computational speed - several times faster than the Lagrangian approach [78, 79]. On the other hand, the Eulerian approach does not yield the highly accurate results that the Lagrangian approach yields. This is mainly due to two factors. First, in the Eulerian PDE approach for correspondences there is a loss of precision when setting up the boundary conditions for ϕ_0 and ϕ_1, as grid points are set up to be equal to their coordinate positions. This is mostly the case because both the inner and the outer boundaries exist at an inter-pixel level and rarely contain a grid point.

We may encounter another problem when solving (3.1) and (3.2) as well. Careful examination of (3.1) and (3.2) reveals that the computed values of ϕ_0 and ϕ_1 at $\mathbf{x}$ are convex combinations of ϕ_0 and ϕ_1 at grid points that are neighbors of $\mathbf{x}$. Therefore, it is possible for $\mathbf{x}$ to be mapped either far outside the boundary or far inside the boundary, depending on the boundary's curvature. This problem can be quite severe, especially for objects having highly detailed boundaries. The maps computed this way do not have pixel accuracy, much less the sub-pixel that is sometimes required in applications.

In [68] we create a natural way to blend the Lagrangian and Eulerian PDE approaches so that the resulting method yields more accurate results than the Eulerian PDE approach while requiring less computation time than the Lagrangian approach. In our approach we modify the previous Eulerian approach to obtain a new hybrid algorithm with prescribable accuracy and the minimal possible sacrifice in speed.

The first step to increase the accuracy of the Eulerian PDE approach is to improve the boundary conditions of the PDEs involved. We do this by using the Lagrangian approach to compute the values of ϕ_0 at the grid points of R located immediately next to the inner boundary. Similarly, we use the Lagrangian approach to compute the values of ϕ_1 at grid points immediately next to the outer boundary. Once we have computed these values, we use the Eulerian PDE approach to solve for ϕ_0, and ϕ_1 at the remaining grid points. In doing so, not only do we obtain more accurate values near the boundaries, but also we avoid propagating larger computational errors throughout the whole region R. Since these grid points are at most one grid away from the boundary, the explicit computation of their correspondence trajectories does not require extensive computations.

Once the initial conditions for the Eulerian approach have been improved, the next step is to guarantee that points be mapped as closely as desired to the boundaries. This can be done by making some changes to the order transversal algorithm proposed in [78]. In this algorithm points are visited in the order that they are reached by the correspondence trajectories as they flow away from the known boundary. As a result, only one full sweep

through the grid points in R is required to solve for ϕ_0, followed by one other sweep, but in a different direction, for ϕ_1.

Let λ be a chosen tolerance constant. As we will explain shortly, λ will provide a means to control the precision of the proposed hybrid approach. In the 2-D case, when computing $\phi_\alpha[i,j]$ (where a could be either 0 or 1) the idea consists of obtaining first the distance between the initially calculated values of $\phi_\alpha[i\pm1,j]$ and $\phi_\alpha[i,j\pm1]$. If this distance is less than λ, then we can assume that their correspondence trajectories do not diverge a lot as they approach the boundary and that the error accumulation is small, resulting in a very good precision. If, however, the distance is equal to or larger than λ, then we will need to compute $\phi_\alpha[i,j]$ using another technique. One thing we can do is just use the Lagrangian approach and follow the correspondence trajectory for this particular grid point until we reach the corresponding boundary. By doing so, we are taking full advantage of the Lagrangian approach and getting an accurate value. However, a faster way would be to follow the correspondence trajectory until it reaches a horizontal or vertical line between two grid points that have already been solved such that the distance between the two boundary maps is less than the tolerance λ. If they are close enough, we use the linear interpolation implicit in the discretized Eulerian PDE approach to estimate the value of $\phi_\alpha[i,j]$ at that point (since ϕ_α is constant along the correspondence trajectories, this is the value of ϕ_α at the original grid point); otherwise, we continue following the trajectory and doing the same procedure until we find two correspondences that are close enough according to the desired tolerance. This procedure is shown in the algorithm in Appendix A. The algorithm for computing ϕ_1 is almost the same as above with minimal differences as specified in [84].

Figure 3.3 shows some correspondence trajectories as well as a some generated grids for a myocardial MR image, evidencing the need to form curved correspondence trajectories in some parts of the annular region.

Besides computing correspondences at the same time and in a very fast way, the hybrid Eulerian-Lagrangian approach has two more important advantages. First, it terminates

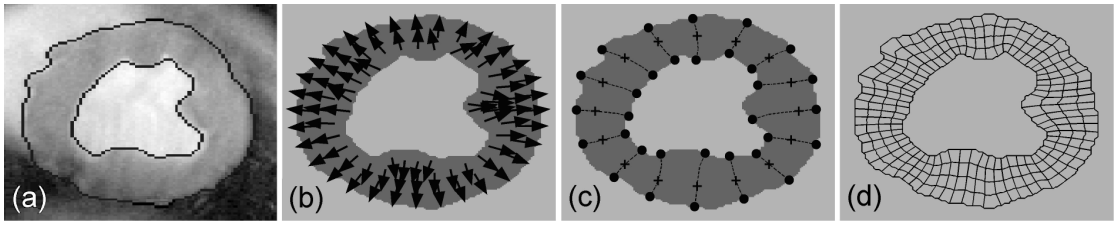

Figure 3.3: Myocardial from a short-axis MR image. (a) Endocardial and epicardial contours. (b) Tangent field. (c) Boundary correspondences for some selected points. (d) Generated grids.

automatically so we do not have to keep testing for convergence. And second, the desired tolerance λ gives us a way to control the accuracy of the computed values. If λ is very large, the hybrid Eulerian-Lagrangian approach will be nothing more than the Eulerian PDE approach with improved initial conditions, whereas if λ is 0, the hybrid Eulerian-Lagrangian approach will yield the same results as the Lagrangian approach.

3.2 Harmonic Embedding

In the previous part of this work the definition of correspondences for an annular region R was based on the gradient flow of a harmonic interpolant u that is equal to 0 in the inner boundary $\partial_0 R$ and equal to 1 in the outer boundary $\partial_1 R$. Therefore, the more accurate the estimation of u is, the better the estimations of correspondences are going to be. When computing its values, the harmonic function u can be approximated by just assigning $1s$ to the grid points on or outside $\partial_1 R$ and $0s$ to the grid points on or inside $\partial_0 R$ and then solving iteratively for grid points inside R using any appropriate iterative method such as Gauss, Gauss-Seidel, and Multigrid. This approach has the advantage of being fast and easy to implement. On the other hand, its main disadvantage is that the resulting harmonic function is most of the time different from 1 on the given outer boundary and different from 0 on the given inner boundary because those boundaries are typically inter-pixel. As a result, the harmonic function computed in this manner is slightly different from the one we really wish to find. We may overcome this problem by adapting the technique proposed

by Duci *et al.* [81], called harmonic embedding, to our specific needs. This work has already been presented in [84, 70] and is summarized below.

Let $P_0 = \left\{p_{0,i}\right\}_{i=1,\ldots,m}$ be the set of grid points in the inner boundary $\partial_0 R$ or just next to it, but not in R. Similarly, let $P_1 = \left\{p_{1,j}\right\}_{j=1,\ldots,n}$ be the set of grid points in the outer boundary $\partial_1 R$ or just next to it, but not in R. In addition, let P be the set of all grid points located in R. We now define a basis of harmonic functions, $u_{0,i}$ and $u_{1,j}$, satisfying the following conditions:

$$\begin{cases} \Delta u_{i,j}(x,y) = 0, & for\,(x,y) \in P \\ u_{i,j}(x,y) = 1, & for\,(x,y) = p_{i,,j} \\ u_{i,j}(x,y) = 0, & for\,all\,other\,P_0\,and\,P_1 \end{cases} . \tag{3.4}$$

Both $u_{0,i}$ and $u_{1,j}$ can be easily solved using any iterative procedure such as those previously mentioned. Now, let u be a linear combination of these basis functions, *i.e.,*

$$u = \sum_{i=1}^{m} \alpha_{0,i} u_{0,i} + \sum_{j=1}^{n} \alpha_{1,j} u_{1,j}, \tag{3.5}$$

where $\alpha_{i,j} \in \mathbb{R}$. By construction, u is harmonic because it is a linear combination of harmonic functions. So, the problem now consists of finding the appropriate coefficients $\alpha_{i,j}$ so that $u(\partial_0 R) = 0$ and $u(\partial_1 R) = 1$. This can be done offline by the gradient descent procedure described in [81].

The harmonic embedding representation can be easily extended to 3D contours within a spherical annulus, as shown in Figs. 3.4 and 3.5, where the 3D evolutions from the initial estimates to the target contours are depicted. The harmonic embedding representation has many others applications. Figure 3.6 shows the average shape obtained by just averaging the coefficients. In addition, convex combinations of the harmonic embedding representations can be easily computed by just doing a weighted average of the coefficients to continuously interpolate between the two shapes (Fig. 3.7).

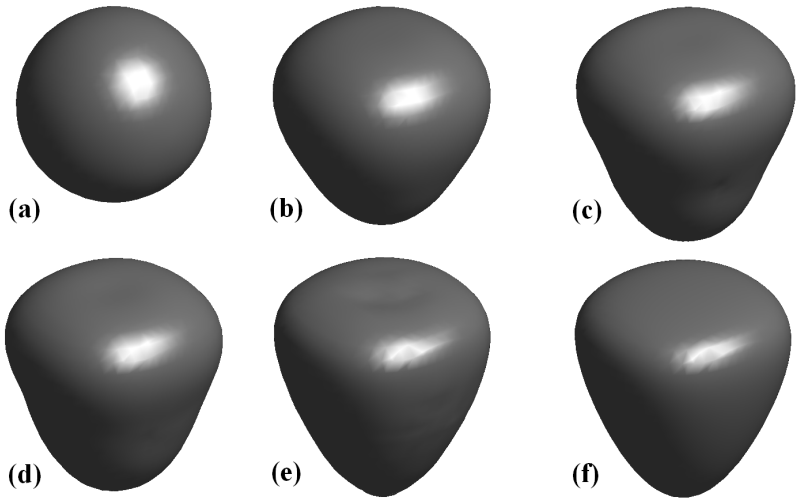

Figure 3.4: Evolution of an initial estimate to the minimum of the set-symmetric difference that is used to find the best harmonic embedding. (a) Initial contour estimate. (b)-(e) Evolutions. (f) Target contour.

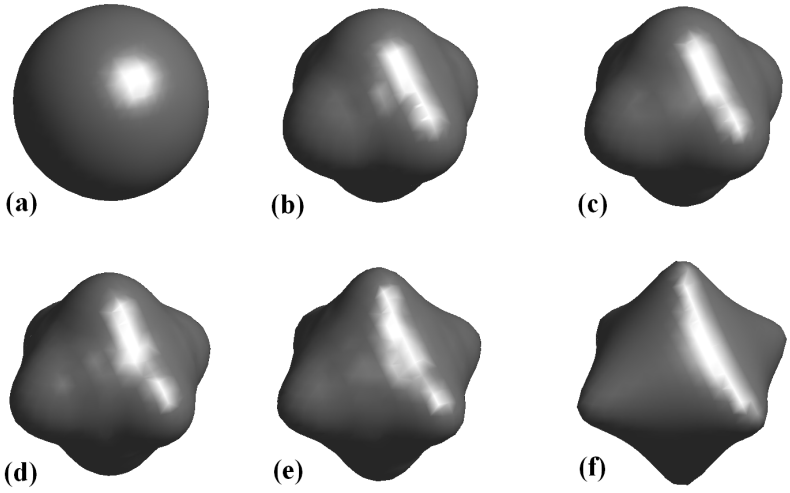

Figure 3.5: Evolution of an initial estimate to the minimum of the set-symmetric difference that is used to find the best harmonic embedding. (a) Initial contour estimate. (b)-(e) Evolutions. (f) Target contour.

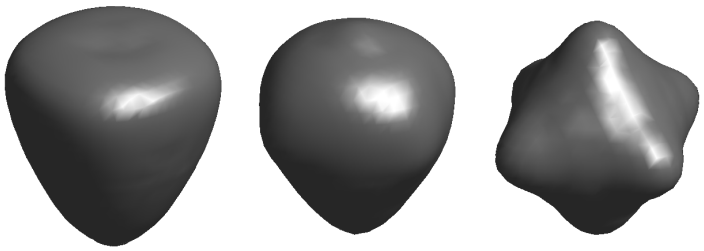

Figure 3.6: Average shape between (left) and (right) computed by averaging (center) the corresponding coefficients.

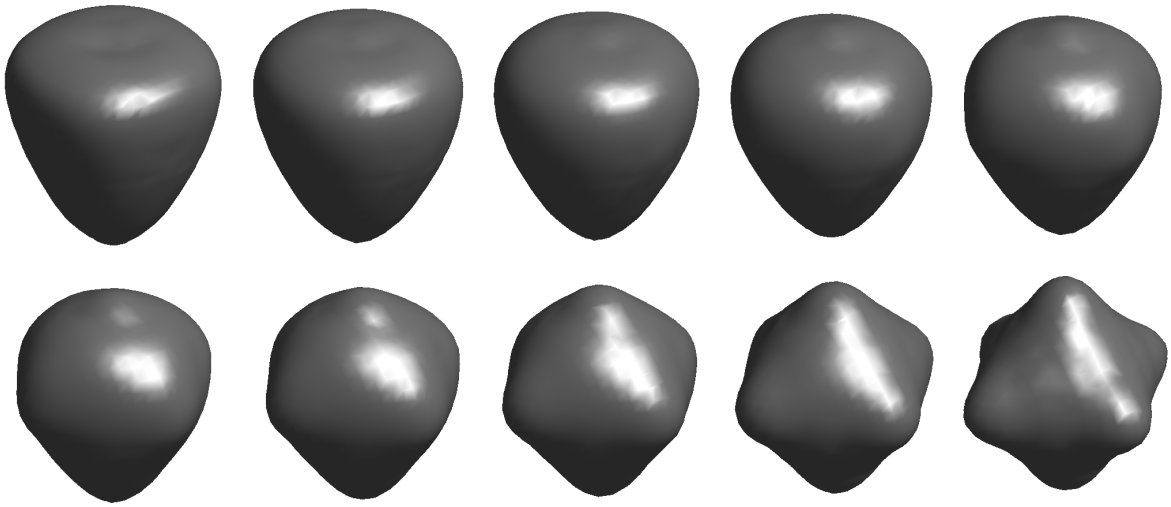

Figure 3.7: Convex combinations of the two harmonic embedding representations shown in (top left) and in (bottom right).

CHAPTER 4

PDE TECHNIQUES FOR VISIBILITY MAXIMIZATION

The object of this work is to create an algorithm that maximizes the visibility of a surface with respect to a fixed external viewpoint by evolving the surface locally in a convenient way. Indeed, we propose a novel energy that measures the visibility of a surface with respect to a fixed viewpoint and then we evolve the surface by a gradient descent along the proposed energy. One would expect that the most immediate act would be an evolution of the surface that would allow one to see the parts of the surface that are immediately occluded by the surface itself. Consequently, this work would be suitable for the exploration and visualization of complicated surfaces.

Even though the real applications of our approach are in 3D, this chapter is focused only on 2D visibility maximization so that we can get a better insight into the 3D case, which is described in the next chapter.

This chapter is divided into two sections. In the first section we describe some of the first approaches we tried for visibility maximization. Specifically, we show how we changed from trying to maximize the visibility of a contour with respect to another contour to trying to maximize the visibility of a contour with respect to a fixed viewpoint. In the second section we describe our viewpoint-based visibility-maximization approach for the case of polygons and differentiable contours. We cover in detail the geometric constraints that are needed for our proposed approach to work properly. We conclude the chapter by presenting several simulation results that we have already published in [83, 84].

4.1 Visibility Maximization

Initially, our goal was to maximize the global visibility of a contour, that is, we wanted to evolve the contour so that it becomes more visible from another surface. To do this,

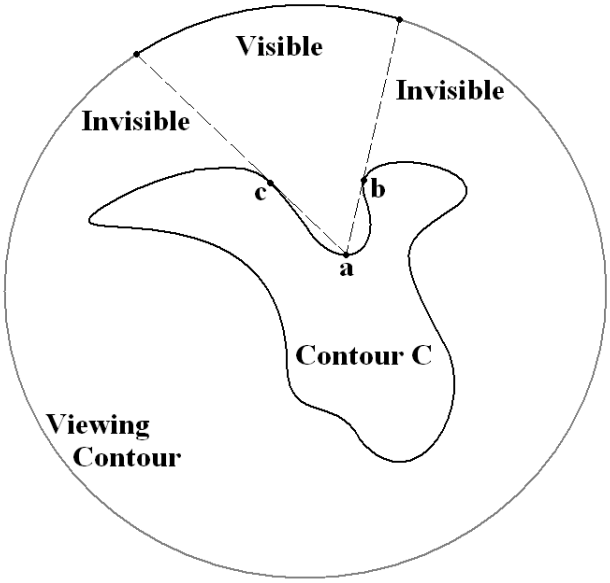

Figure 4.8: Visibility with respect to a viewing circumference.

we first found a way to measure the visibility of a point in a contour. Then, we created an energy representing the sum of the values of the visibility of each point in the surface. Accordingly, we expected that the maximization of this energy would have an unfolding effect of the given contour.

Consider Fig. 4.8, where a contour C is located inside a viewing circumference. The figure shows the arc in the viewing circumference from which the point **a** in C is visible. If we find the corresponding visible arc for other points in C, such as points **b** and **c**, then we will notice that some points can be viewed from a larger arc in the viewing circumference than others. This is natural because we cannot expect the length of the viewing arc to be the same for each point in C. In fact, that just occurs when C is also a circumference.

We can define the visibility of C with respect to a viewing circumference as the sum of the lengths of the visible arc for each point in the contour. It can be maximized by minimizing the region of the viewing circumference from which each point in C is not visible. That is, the global visibility of C with respect to the viewing circumference can be maximized by minimizing the following energy :

$$E(C) = r \oint_C \oint_{C,\hat{s}\neq s} \left(1 - \frac{\mathbf{C}(s)\cdot(\mathbf{C}(\hat{s})-\mathbf{C}(s))}{\sqrt{\|\mathbf{C}(\hat{s})-\mathbf{C}(s)\|^2 r^2 - (\mathbf{C}(s)^T J\,\mathbf{C}(\hat{s}))^2}}\right) \frac{(\mathbf{C}(\hat{s})-\mathbf{C}(s))^T J^T \mathbf{C}_{\hat{s}}(\hat{s})}{\|\mathbf{C}(\hat{s})-\mathbf{C}(s)\|^2}\, d\hat{s}\, ds, \quad (4.6)$$

where s is the arclength parameter, J is the 2D rotation matrix, and r is the radius of the viewing circumference. Unfortunately, the gradient flow along this energy is the curvature times a constant. Therefore, the evolution of a contour based on this energy will just smooth the contour, not unfold it.

Another approach for maximizing the global visibility of a contour is depicted in Fig. 4.9. This figure shows the occluding part of a contour C for a given point **a**. Accordingly, the occluding part (*i.e.*, the part of the contour that goes from point **b** to point **c**) is blocking point **a** from being visible from other parts of the contour. For each point in C, the occluding part, if any, is defined by two extreme points (points **b** and **c** in this case). In first one (point **b**), the viewing direction coming from point **a** forms a 0 degree angle with the tangent of C at point **a**. On the other hand, in the second point (**c**), the viewing direction coming from point **a** is orthogonal to the inner unit normal of C at the point **c**.

Let us define the energy

$$E(C) = \oint_C \oint_{C,\hat{s}\neq s} H((\mathbf{C}(\hat{s}) - \mathbf{C}(s))^T J^T \mathbf{C}_{\hat{s}}(\hat{s}))\, H((\mathbf{C}(\hat{s}) - \mathbf{C}(s))^T J\, \mathbf{C}_s(s))\, d\hat{s}\, ds, \qquad (4.7)$$

where H is the Heaviside function, $J\,\mathbf{C}_s(s)$ is the outward unit normal in $\mathbf{C}(s)$, and $J^T\mathbf{C}_{\hat{s}}(\hat{s})$ is the inward unit normal at $\mathbf{C}(\hat{s})$. It is easy to verify that for each point $\mathbf{C}(s)$ the integrand above is equal to 1 on the corresponding occluding part. Accordingly, this energy measures the global visibility of the contour. Thus, minimizing it would make a contour more visible as it would become convex. However, it would not unfold a complex contour.

Now, instead of trying to maximize the global visibility of a surface, we may want to maximize its visibility from a fixed viewpoint. The application of something like this would be for visualization. That is, instead of trying of generically make the complete surface more visible from an unknown viewpoint, we would evolve the surface locally so that its visibility with respect to a fixed point is maximized. This is depicted in Fig. 4.10. This figure shows the parts of the contour C that are visible from the viewpoint **P**.

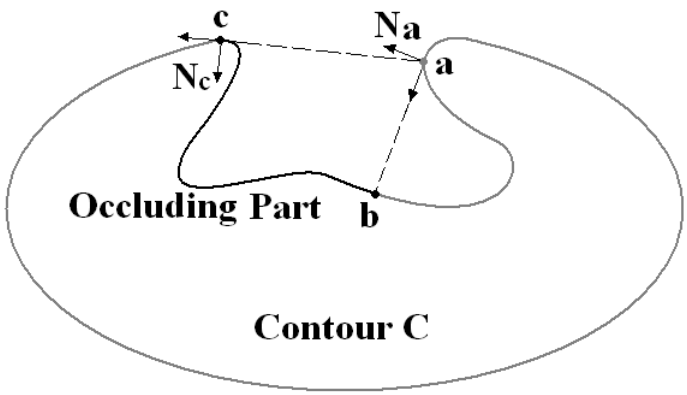

Figure 4.9: Self visibility minimization.

Specifically, the parts of C going from point**a** to point **c**, from point **d** to point **e**, and from point **f** to point **b** are the only ones that are visible from **P**. In the next section, we propose a method that evolves the surface locally so that the visibility of C from **P** is maximized. In this approach, the occluding parts of the contour move away to make visible other parts of the contour.

4.2 New Visibility Energy Functional for Viewpoint-based Unfolding: The Flux Model

The problem of maximizing the visibility of a contour with respect to a fixed viewpoint can be thought of as the problem of maximizing the flux of light that is being absorbed or received by a contour. From this perspective, we assume that the rays of light emitted by the viewpoint are neither reflected nor diffracted by the contour. Accordingly, a point is considered to be visible from the viewpoint if the line segment between the point and the viewpoint does not intersect the contour.

In Fig. 4.11 the contour C is absorbing the light coming from the viewpoint **P**, which is acting as the source of light. The part of C that goes from point **a** to point **b** and contains the points **c**, **d**, **e**, **f**, and **g** is absorbing all the rays coming from **P**. From now on, we will refer

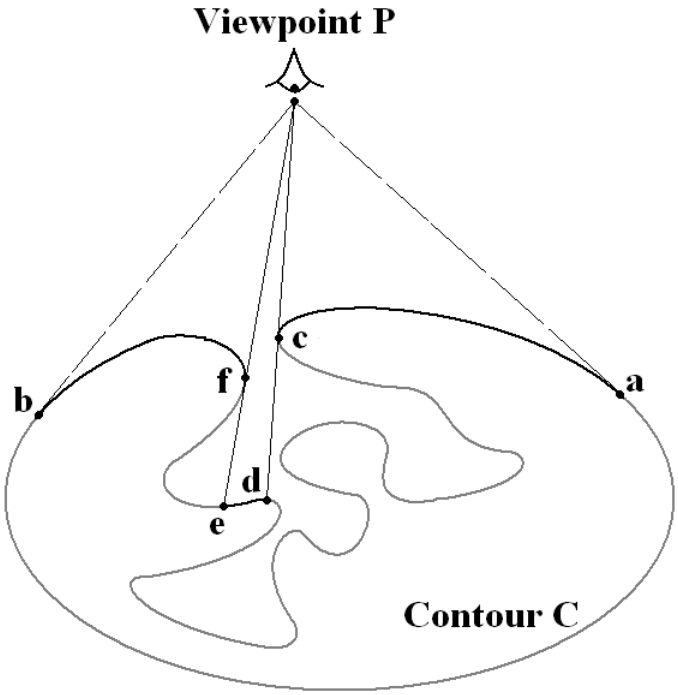

Figure 4.10: Visibility with respect to a point.

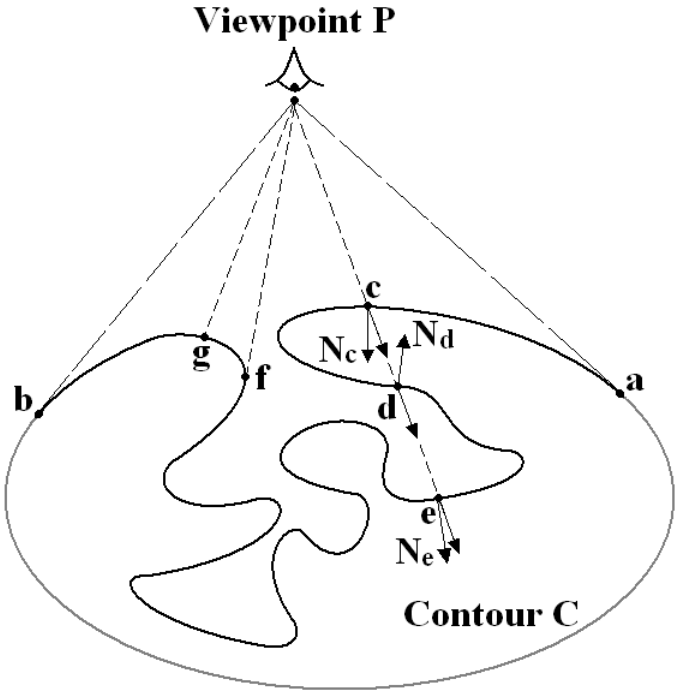

Figure 4.11: Visibility of a contour with respect to a viewpoint **P**.

to this region as the *region of interest* of the contour. In the figure we have used a natural criteria for choosing this region by finding the two points of C that are intersected by the extremities of the viewing region and then selecting the portion of C that passes through these two points. This procedure can be done automatically, but it is out of the scope of this work.

In the proposed flux model the objective is to evolve the region of interest of the contour in such a way that the average flux it receives is maximized. In order to do this, we first need a way to quantify the amount of flux received at any point in the contour. For instance, in Fig. 4.11 both points **g** and **f** receive light from **P**. However, we might argue that points in the neighborhood of point **g** are receiving more light than points in the neighborhood of point **f**. This occurs because the rays arriving at point **g** are almost perpendicular to the contour, whereas the rays arriving at point **f** are almost tangent to the contour.

We define the flux at any point in the contour as the Euclidean dot product between the unit ray that is coming from the viewpoint and the inward unit normal of the contour at the given point (see Fig. 4.12). Accordingly, the value of the flux at any point will always be between -1 and 1. This approach provides a physical interpretation of how any point in the contour is illuminated. If a point is illuminated, that is, if it is visible from the viewpoint, a positive flux indicates the degree of perpendicularity of the incoming ray. For instance, in Fig. 4.11 the flux at point **g** is greater than the one at point **c** because the ray of light is more perpendicular to the contour at the former than it is at the latter. On the other hand, if a point is not receiving any light because the ray is being blocked by the contour, a positive flux indicates how perpendicular the ray will come in if the part of the contour that is blocking the ray moves away. This is the case of point **e** in Fig. 4.11, because even though the point does not receive any light, it has a positive flux. When a point has a negative flux, like point **d** in Fig. 4.11, it means that the point is not receiving any light. The more negative the flux, the more the point will have to move to receive light. Figure 4.13 shows flux values at different points on the region of interest.

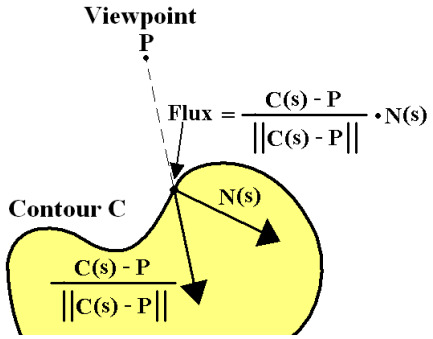

Figure 4.12: Computation of the flux of a point on a contour with respect to a viewpoint **P**.

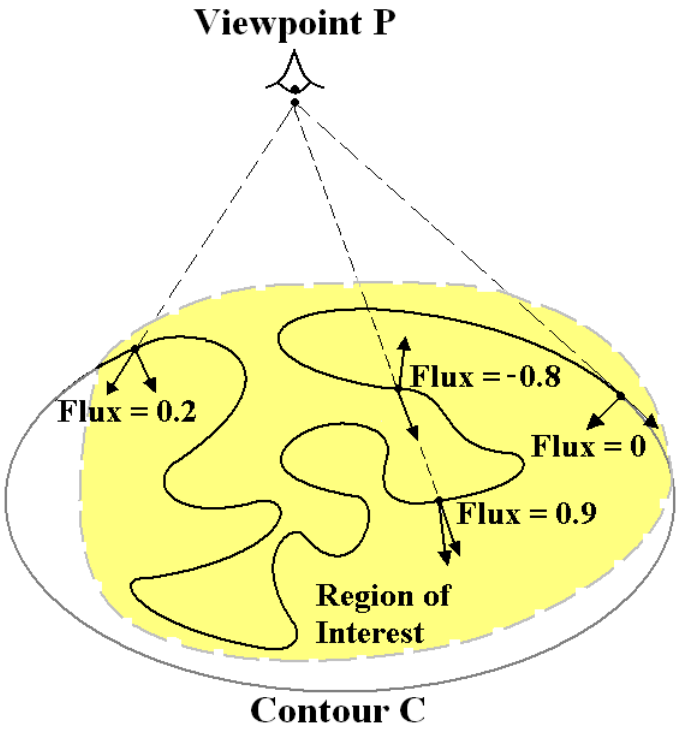

Figure 4.13: Flux value at different points on the region of interest.

This flux definition is very similar to the one proposed by Zhang and Turk [18]. In the definition proposed by Zhang and Turk, the flux is the product of the inner product as defined in the previous paragraph and a characteristic function that is 1 when a point is visible and 0 otherwise. In other words, the flux is positive for all the points that receive light from the viewpoint and 0 otherwise. Consequently, Zhang's definition of flux tells exactly which points of the contour are visible and which ones are occluded from the viewpoint. However, as we will see later, our definition is more appropriate for our purpose because, among other things, we do not have to deal with all the discontinuities of having Heaviside functions.

The average flux, which we call $E_{2D}(C)$, can be computed by integrating the flux at all the points in the region of interest of the given contour and then dividing by the total length. That is,

$$E_{2D}(C) = \frac{1}{L}\int_0^L \frac{(\mathbf{C}(s)-\mathbf{P})}{\|\mathbf{C}(s)-\mathbf{P}\|}\cdot\mathbf{N}(s)\,ds, \tag{4.8}$$

where L is the length of region of interest of the contour C. From now on, we will use the terms "average flux" and "average visibility" interchangeably.

If instead of using our flux definition in (4.8) we use the one in [18], we will get the exact average visibility that the region of interest of the contour is receiving (see Fig. 4.14). However, if we take into consideration that any point that is not visible and still has a positive flux by necessity has to have a corresponding point with negative flux (see the case of points **e** and **d** in Fig. 4.11), then the expression in (4.8) would be a good estimate of the real average visibility.

We need to point out a very important feature of the energy in (4.8). If we evolve the contour according to the gradient ascent along this energy, the points where the flux is negative would be forced to move in such a way that the flux they receive increases. By doing so, these points would make visible other points that have positive flux, but are not

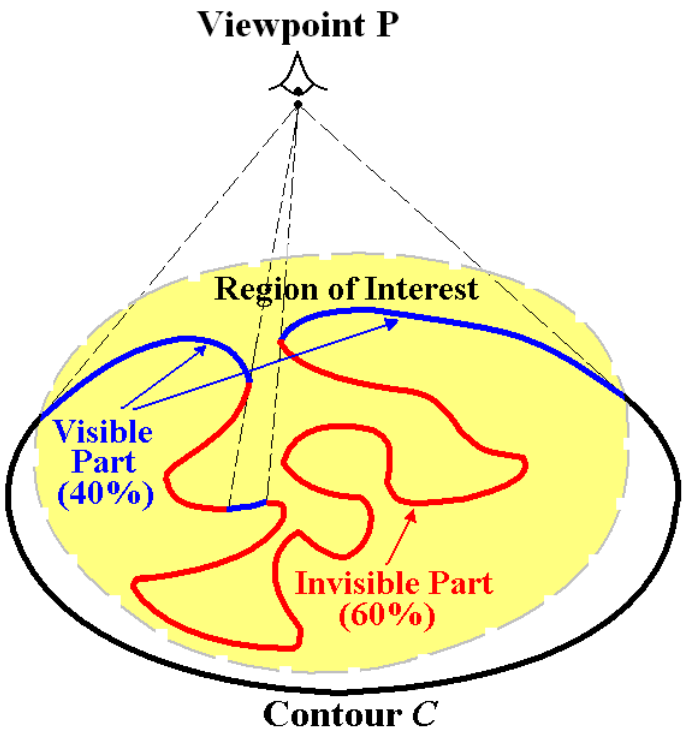

Figure 4.14: Example of a region with average visibility equal to 0.4.

visible from the viewpoint. This would generate the unfolding motion we are looking for. This effect would not occur if we use the Heaviside function as in [18] because points that are not visible simply have zero flux.

Since the flux at any point can be at most equal to 1, $E_{2D}(C)$ also has a maximum of 1. This would occur when the viewing surface coincides with an arc of a circumference that has the viewpoint $\mathbf{P}$ as its center. In this case, the rays coming from the viewpoint would have the same direction as the inward unit normal at each point and, consequently, the flux at any point of the region of interest of the contour would be equal to 1. Moreover, in this case all points would have the same flux. Therefore, maximizing the average visibility will, by itself, promote an equal distribution of light along the region of interest of the contour.

Using the Calculus of Variations we show in Appendix B that the gradient ascent for the energy in (4.8) is

$$\mathbf{C}_t = \frac{1}{L}\left(\kappa(s)\,E_{2D}(C) + \frac{1}{\|\mathbf{C}(s)-\mathbf{P}\|}\right)\mathbf{N}(s), \tag{4.9}$$

where $\kappa(s)$ is the curvature of C at the point $\mathbf{C}(s)$. This gradient shows that the problem is well-posed and that we can use our model to maximize the visibility of piece of contour *only* if the initial average visibility with respect to $\mathbf{P}$ is positive (otherwise a backwards heat flow results).

4.2.1 Polygon Case

Now consider a polygon whose region of interest has N edges with ordered vertexes $\mathbf{v}_i \in \mathbb{R}^2$ for $i \in \mathbb{Z}_n = \{1,2,\ldots,N+1\}$ and let $|C_i|$ be length of the edge C_i that goes from $\mathbf{v}_i$ to $\mathbf{v}_{i+1}$ (see Fig. 4.15). For this case the energy (4.8) that measures the average flux received by a region of interest of a differentiable contour can be rewritten as

$$E_{2D,P} = \frac{1}{L_T}\sum_{i=1}^{N} E_{2D,i}, \tag{4.10}$$

where $L_T = |C_1| + \ldots + |C_N|$ is the length of the region of interest and $E_{2D,i}$ is the total flux received by the edge C_i. In particular, $E_{2D,i}$ can be computed as follows:

$$\begin{aligned} E_{2D,i} &= \int_0^{|C_i|} \frac{\left(\mathbf{v}_i + s\frac{(\mathbf{v}_{i+1}-\mathbf{v}_i)}{|C_i|} - \mathbf{P}\right)^T J^T \frac{(\mathbf{v}_{i+1}-\mathbf{v}_i)}{|C_i|}}{\left\|\mathbf{v}_i + s\frac{(\mathbf{v}_{i+1}-\mathbf{v}_i)}{|C_i|} - \mathbf{P}\right\|} ds \\ &= \int_0^{|C_i|} \frac{(\mathbf{v}_i - P)^T J^T (\mathbf{v}_{i+1}-\mathbf{v}_i)}{|C_i|\left\|\mathbf{v}_i + s\frac{(\mathbf{v}_{i+1}-\mathbf{v}_i)}{|C_i|} - \mathbf{P}\right\|} ds \\ &= \frac{\gamma_i}{|C_i|}\left[\ln\left(|C_i| + \beta_i + \|\mathbf{v}_{i+1}-\mathbf{P}\|\right) - \ln\left(\beta_i + \|\mathbf{v}_i - \mathbf{P}\|\right)\right], \end{aligned} \tag{4.11}$$

where

$$\gamma_i = (\mathbf{v}_i - P)^T J^T (\mathbf{v}_{i+1} - \mathbf{v}_i), \tag{4.12}$$

$$\beta_i = \frac{(\mathbf{v}_{i+1} - \mathbf{v}_i)\cdot(\mathbf{v}_i - \mathbf{P})}{|C_i|}, \tag{4.13}$$

and $J = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$ is the 2D rotation matrix.

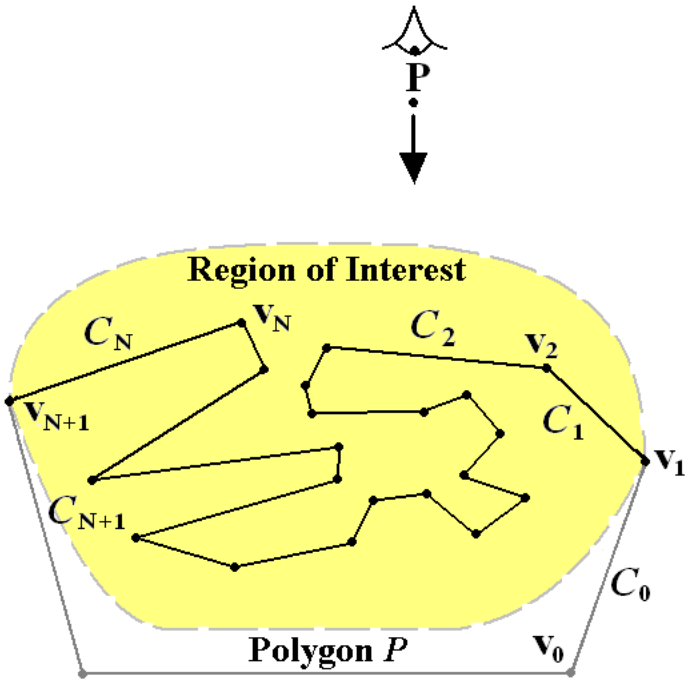

Figure 4.15: Visibility of a polygon with respect to a viewpoint **P**.

The value of $E_{2D,i}$ depends on how the rays coming from the viewpoint **P** are aligned with the corresponding unit inward unit normals. For instance, if the middle point between $\mathbf{v}_i$ and $\mathbf{v}_{i+1}$ and the value of $|C_i|$ remains constant, the maximum value for $E_{2D,i}$ would be achieved when the ray coming from **P** forms a zero degree angle with the unit inward unit normal at the middle point. This is depicted in Fig. 4.16, where $E_{2D,i}$ is greater when the edge C_i is positioned as in (a) than when it is positioned as in (b).

Taking the derivative of $E_{2D,P}$ with respect to the vertex $\mathbf{v}_k$, we get

$$\begin{aligned}\frac{\partial E_{2D,P}}{\partial \mathbf{v}_k} &= -\frac{1}{L_T^2}\sum_{i=1}^{N} E_{2D,i}\frac{\partial L_T}{\partial \mathbf{v}_k} + \frac{1}{L_T}\left(\frac{\partial E_{2D,k}}{\partial \mathbf{v}_k} + \frac{\partial E_{2D,k-1}}{\partial \mathbf{v}_k}\right) \\ &= -\frac{1}{L_T}\left(\frac{1}{L_T}\sum_{i=1}^{N} E_{2D,i}\right)\left(\frac{\mathbf{v}_k - \mathbf{v}_{k+1}}{|C_k|} + \frac{\mathbf{v}_k - \mathbf{v}_{k-1}}{|C_{k-1}|}\right)^T \\ &\quad + \frac{1}{L_T}\left(\frac{\partial E_{2D,k}}{\partial \mathbf{v}_k} + \frac{\partial E_{2D,k-1}}{\partial \mathbf{v}_k}\right), \end{aligned} \tag{4.14}$$

because the edges C_k and C_{k-1} as well as the energies $E_{2D,k}$ and $E_{2D,k-1}$ are the only expressions in $E_{2D,P}$ that depend on the value of $\mathbf{v}_k$. The partial derivatives of $E_{2D,k}$ and

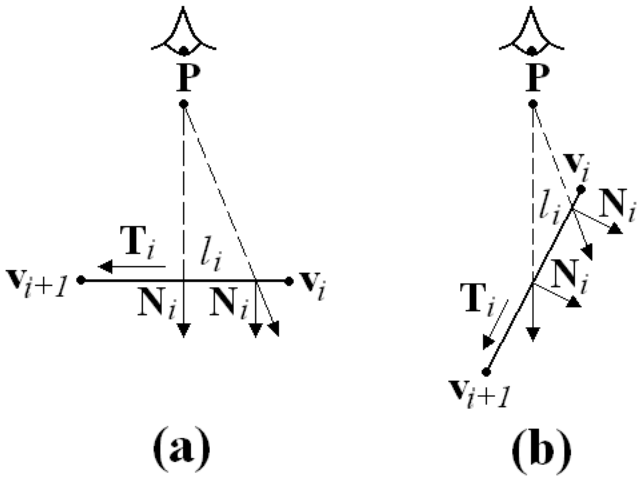

Figure 4.16: The total flux received by edge C_i is greater in (a) than in (b) because the rays coming from **P** are more aligned with the corresponding inward unit normals in (a) than they are in (b).

$E_{2D,k-1}$ with respect to $\mathbf{v}_k$ are too long to include here. They are presented in Appendix C. It follows that the motion of the vertexes to maximize the average visibility in (4.10) can be computed by

$$\frac{d\mathbf{v}_k}{dt} = \mu_{2D}\frac{\partial E_{2D,P}}{\partial \mathbf{v}_k}, \tag{4.15}$$

where μ_{2D} is a positive constant, $k \in \{1,...,N+1\}$, and t denotes an artificial time variable. Since the region of interest of the polygon goes from $\mathbf{v}_1$ to $\mathbf{v}_{N+1}$, the value of the partial derivative of $E_{2D,k-1}$ with respect to $\mathbf{v}_k$ is zero for $k = 1$. Similarly, the value of the partial derivative of $E_{2D,k}$ with respect to $\mathbf{v}_k$ is also zero for $k = N+1$.

4.2.2 Topology Preservation

When we evolve the vertexes of the region of interest of a polygon according to (4.15) to maximize the average visibility we have to make sure that the topology of the polygon does not change. In other words, we have to ensure that during the evolution the edges of the polygon do not intersect each other. To do this, we have to use a topology-preservation

method.

Several topology-preservation methods have been proposed in the past. Hans *et al.* [86, 87] presented a technique to prevent topology changes when the active contour evolution is implemented via Level Sets Methods. In this work, changes in topology at grid points are detected by deriving a condition based on the configuration of the level set function in a small neighborhood of the grid points. This method has the disadvantage of being highly dependent on the grid spacing used in the level set function. In addition, when this method is used the resulting motion may be abrupt and look unnatural.

Unal *et al.* [88] proposed another approach for topology preservation for active polygons. In this work, it is assumed that the polygon consists of a uniform charge distributed along its perimeter. Each vertex is then moved in the direction of the electrostatic force, which is computed numerically. Even though this method may prevent some topology changes, it does not prevent two adjacent sides from touching. Moreover, the flow is unstable as the number of vertexes increases and the length of the segments decreases [88].

Sundaramoorthi and Yezzi [85] have proposed a robust topology-preservation technique in which a special geometric flow is added to the original image-based curve evolution to avoid intersections. This geometric flow, which is derived from the minimization of an energy based on electrostatic principles, affects significantly the original evolution only when the contour is close to a change in topology. Unlike a curvature regularizer, when the regularizer proposed in [85] is applied to a point the resulting force depends globally on all other points of the curve. This technique, which is based on the work in [89, 90], has the advantage over the one proposed in [86, 87] of changing the original evolution in a gradual manner. Moreover, it is not restricted to level sets and can be used on any active contour implementation. Because of its robustness, the topology-preservation method on [85] is the one we use to preserve the topology of the evolving polygon. Furthermore, in the next chapter we extend it to three dimensions so that we could use it for our visibility-maximization application. In the following paragraphs we briefly present Sundaramoorthi

and Yezzi's approach for topology preservation. We refer the reader to [85] for more details.

Let $C \in \mathbb{R}^2$ be a twice-differentiable contour of length L and let $E_{2D,R}$ be the energy of an uniform charge distributed along C defined by

$$E_{2D,R}(C) = \frac{1}{2}\int\int_{C\times C}\left(\frac{1}{\|\mathbf{C}(s)-\mathbf{C}(\hat{s})\|}-\frac{1}{d_C(s,\hat{s})}\right)d\hat{s}\,ds, \tag{4.16}$$

where $d_C(s,\hat{s})$, the geodesic distance along the curve C from point $\mathbf{C}(s)$ to point $\mathbf{C}(\hat{s})$, is use to eliminate the infinite component of the first term, thereby making the energy finite. However the gradient of this energy has the property of still becoming infinitely large whenever the curve becomes close to self-intersection.

Using the Calculus of Variations, it is shown in [85] that the gradient of is given by

$$\begin{aligned}\mathbf{R}_{2D}(s) = \lim_{\varepsilon\to 0^+}\Bigg[&\int_{B_C(\varepsilon,s)}\frac{\mathbf{C}(s)-\mathbf{C}(\hat{s})}{\|\mathbf{C}(s)-\mathbf{C}(\hat{s})\|^3}\cdot\mathbf{N}(s)\,d\hat{s}\\ &+\int_{B_C(\varepsilon,s)}\frac{d\hat{s}}{\|\mathbf{C}(s)-\mathbf{C}(\hat{s})\|}\kappa(s)-\ln\left(\frac{L}{2\varepsilon}\right)\kappa(s)\Bigg]\mathbf{N}(s),\end{aligned} \tag{4.17}$$

where $B_C(\varepsilon,s)=\{C(s): d_C(s,\hat{s})>\varepsilon\}$ represents the set of all points in C except for those within a small neighborhood of $\mathbf{C}(s)$, and $\kappa(s)$ and $\mathbf{N}(s)$ represent the curvature and the inward unit normal of C at the point $\mathbf{C}(s)$, respectively. The first term in (4.17) can be regarded as the projection of the electric vector field of the charge distribution at the point $\mathbf{C}(s)$ onto the inward unit normal $\mathbf{N}(s)$. On the other hand, the second term can be regarded as the electrostatic potential of the charge distribution at the point $\mathbf{C}(s)$.

Now, let us suppose that C is evolved according to the image-based flow $\mathbf{C}_{t,original}(s)$ that is uniformly bounded. Sundaramoorthi and Yezzi [5] show that if the flow $\mathbf{R}_{2D}(s)$ in (4.17) is added to $\mathbf{C}_{t,original}(s)$, then the topology of C will be preserved during the evolution. That is, the resulting flow

$$\mathbf{C}_{t,new}(s) = \mathbf{C}_{t,original}(s)+\mu_R\mathbf{R}_{2D}(s), \tag{4.18}$$

where t is an artificial time variable and μ_R is a positive constant, preserves the topology of C. Moreover, since (4.17) is a geometric flow, this method for topology preservation is suitable for both parametric particle-based and level set implementations.

For the case of a polygon such as the one in Fig. 4.15, (4.16) becomes

$$E_{2D,R}(P) = 2\sum_i (|C_i| \ln |C_i| - |C_i|) + \frac{1}{2}\sum_{i \neq j} \int\int_{C_i \times C_j} \frac{d\hat{s}\,ds}{\left\|\mathbf{C}_i(s) - \mathbf{C}_j(s)\right\|}, \quad (4.19)$$

where the first term results from taking just the finite part of the integral $\int\int_{C\times C} \frac{d\hat{s}\,ds}{\|\mathbf{C}(s)-\mathbf{C}(\hat{s})\|}$ and discarding the infinite component. Like the energy for the differentiable contour case (4.16), the gradient of this energy only becomes infinitely large when the polygon approaches a topology change.

Let $\mathbf{R}_{2D,k}(t)$ be the gradient descend flow of (4.19) for vertex $\mathbf{v}_k$ at time t that is computed by the procedure outlined in [85] and let $\left(d\mathbf{v}_k/dt\right)_{original}$ be an original image-based vertex flow such as the one in (4.15). Sundaramoorthi and Yezzi show that the new vertex flow

$$\left(\frac{d\mathbf{v}_k}{dt}\right)_{new} = \left(\frac{d\mathbf{v}_k}{dt}\right)_{original} + \mu_R\,\mathbf{R}_{2D,k}(t), \quad (4.20)$$

has the property of preserving the topology of P during the evolution.

4.2.3 Length Constraints

If we evolve the vertexes of a polygon according to (4.20) we will notice that the length of some edges is going to increase while the length of others is going to decrease. This undesired effect occurs because the length of an edge affects the average flux the edge is receiving.

We can consider the unconstrained gradient in (4.20) as having two components: one that changes the length of an edge and the other that keeps it constant. If we keep only the part of the gradient that does not change the length of the edges, the result would be a more dramatic evolution in which the visible parts of the contour are going to be forced to

move away so that the invisible parts become visible. This behavior would be very useful for navigating and explorating complicated surfaces.

However, it is not necessary to constrain the length of all the edges of a polygon. In fact, if we do that, it could affect how the region of interest of the contour evolves. This, of course, is undesirable. Instead, we only constrain the length of the edges in the region of interest of the contour. Since we are not evolving vertexes outside the region of interest, the result will be that the extreme edges, C_0 and C_{N+1} in Fig. (4.15), are the only edges for which the length is allowed to change.

Using a similar technique as those employed in [91, 92, 93], we can constrain the length of the edges in the visible part of the contour as follows. For each of the edges in the visible contour we will have a constraint. That is, we will have the following N constraints:

$$\|\mathbf{v}_i - \mathbf{v}_{i+1}\|^2 - l_i^2 = 0, \quad \forall i \in \{1,...,N\}. \tag{4.21}$$

Taking the derivative with respect to time of this expression we get $\mathbf{J}\,\mathbf{v}_t = \mathbf{0}$, where $\mathbf{v}_t = \frac{d}{dt}\left[\mathbf{v}_1^T ... \mathbf{v}_{N+1}^T\right]^T$ and $\mathbf{J}$ is the Jacobian matrix given by

$$\mathbf{J} = \begin{bmatrix} (\mathbf{v}_1 - \mathbf{v}_2)^T & -(\mathbf{v}_1 - \mathbf{v}_2)^T & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & (\mathbf{v}_2 - \mathbf{v}_3)^T & & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & (\mathbf{v}_N - \mathbf{v}_{N+1})^T & -(\mathbf{v}_N - \mathbf{v}_{N+1})^T \end{bmatrix}. \tag{4.22}$$

Now, let $\mathbf{g}_u$ be the vector of the unconstrained gradient flow of the polygon obtained by applying (4.20) to each vertex in the region of interest of the contour. First, we find the minimum norm solution $\mathbf{x}$ to the system $\mathbf{J}\,\mathbf{x} = \mathbf{J}\,\mathbf{g}_u$ and then we subtract it from the original unconstrained gradient $\mathbf{g}_u$ so that $\mathbf{J}\,\mathbf{v}_t = \mathbf{0}$. We will use Lagrange multipliers to solve this constraint-minimization problem. Let us define $F = \mathbf{x}^T\mathbf{x} + \lambda^T(\mathbf{J}\,\mathbf{x} - \mathbf{J}\,\mathbf{g}_u)$, where λ is the vector of Lagrange multipliers. If we set to $\mathbf{0}$ the partial derivative of F with respect to

$\mathbf{x}$ and then solve for $\mathbf{x}$, we would get $\mathbf{x} = \mathbf{J}^T\mathbf{l}$, where $\mathbf{l} = -\frac{1}{2}\lambda$. Substituting this result in the system for which we want to find the minimum norm solution, we get $\mathbf{J}\mathbf{J}^T\mathbf{l} = \mathbf{J}\mathbf{g}_u$. From here we can solve $\mathbf{l}$ using the conjugate gradient method because the matrix $\mathbf{J}\mathbf{J}^T$ is symmetric positive definite. Finally, we can compute the constrained gradient $\mathbf{g}_c$ using

$$\mathbf{g}_c = \mathbf{g}_u - \mathbf{J}^T\mathbf{l}. \tag{4.23}$$

4.2.4 Simulation Results

We now present several simulation results that show the effectiveness of the proposed research on polygons. In all the simulations, the value of μ_{2D} in (4.15) was set to 1000. Also, the coefficient μ_R for the topology-preserving force in (4.20) was set to 0.02. Even though the regions of interest of the polygons are the same ones as the ones depicted in the figures below, the actual viewpoints are positioned much farther away from the polygons than the ones presented in the figures. Unless otherwise stated, all the simulations were run until convergence. In addition, all the simulations were performed in C++ on a 2.52-GHz Pentium IV computer running Windows.

Figure 4.17 depicts the evolution of a given polygon when the viewpoint is positioned as shown. The visible part of this polygon has 19 edges with a total length of 416 pixels. As can be seen in the figure, initially most of the visible part of the contour was occluded. In fact, the initial average visibility and the variance of the flux were equal to 0.21 and 0.39, respectively.

As the polygon evolved, more and more portions became visible until the point where nothing was occluded in the visible part of the polygon. Indeed, the final average visibility was virtually equal to 1 and the variance of the flux was equal to 0. The computational time for this simulation was about 1.50 seconds, with 0.70 seconds resulting from the computation of the forces for the topology preservation. Also, the final length of the visible part of the contour was 418.79 pixels, which implies that the edge lengths were preserved

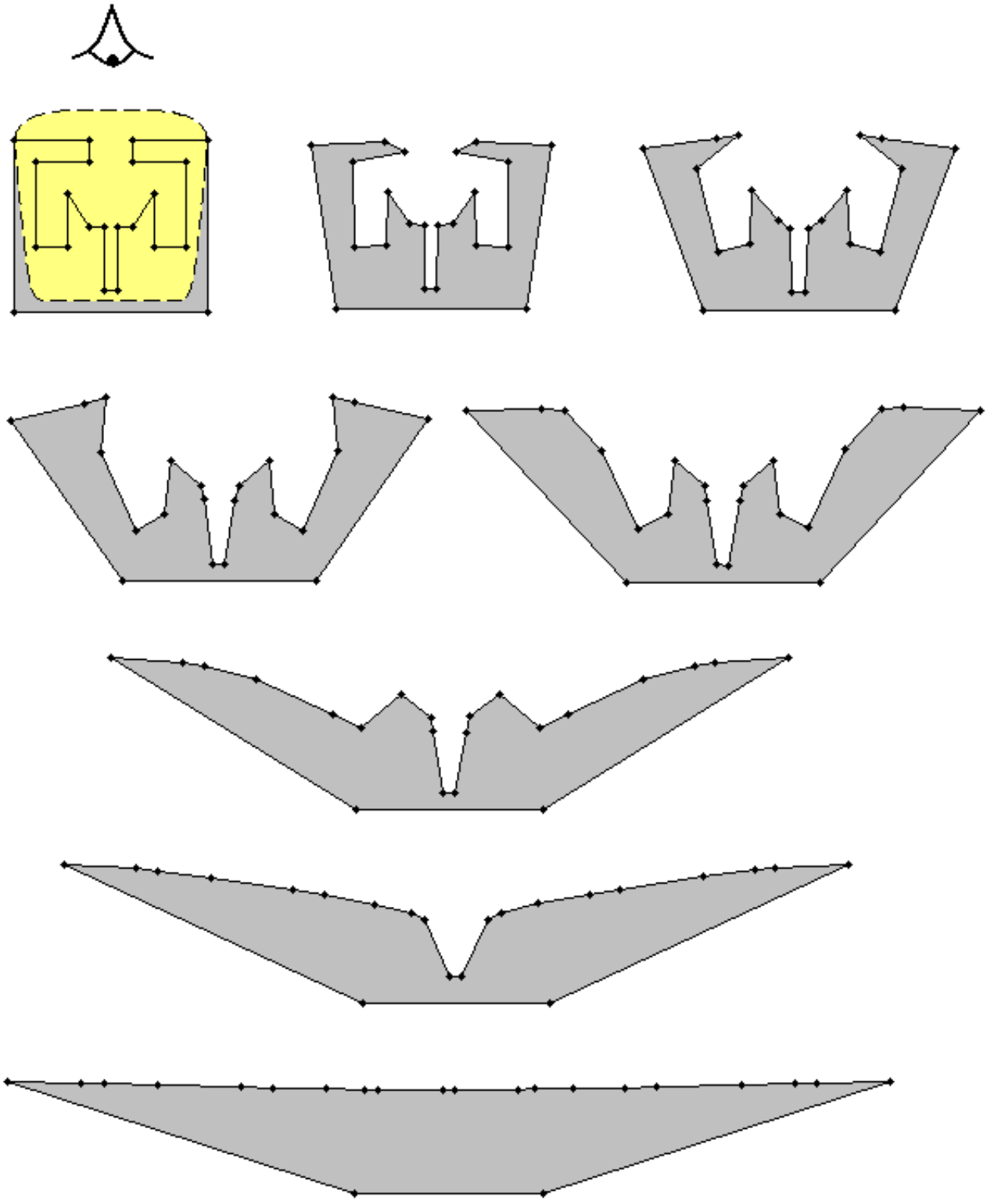

Figure 4.17: Visibility maximization for an intricate polygon.

as expected.

Another simulation result is presented in Fig. 4.18 for a 34-edge polygon that is more convoluted than the one in Fig. 4.17. Initially the polygon had an average visibility of just 0.17 with a variance of the flux of 0.69. These values are reasonable considering how convoluted the polygon is. As the proposed algorithm was applied, the polygon unfolded to make visible sections that previously were not visible. Again, at the end of the simulation the average visibility was almost equal to 1, whereas the variance of the flux was almost 0. The total length of the visible part of the polygon changed from 723.70 pixels at the beginning of the simulation to 727.60 pixels at the end of the simulation; therefore it was virtually preserved. The computational time was about 40 seconds, from which a total of 36 seconds was due to the computation of the topology-preserving forces.

The importance of length preservation can be inferred from the results shown in Fig. 4.19. In this case the proposed approach was applied to the same polygon as the one in Fig. 4.18, but this time without enforcing length constraints. The result was that the visible part of the contour shrank itself conveniently to increase the average visibility. Indeed, the length of the visible region decreased from 723.70 pixels to just 205.41 pixels as the number of vertexes decreased from 34 to 13 and the average visibility increased from 0.17 to almost 1. As a result, even though the visibility was maximized, we did not get the unfolding effect that would allow us to see the occluded sections in the visible part of the contour.

An interesting result that shows the importance of the topology-preserving forces is presented in Fig. 4.20. In this simulation we applied the proposed algorithm without the topology-preserving forces to the highly convoluted polygon of Fig. 4.18. Results show that the edges started to intersect each other after a few iterations. The changes in topology were even greater as the number of iterations increased. Comparing this result to the one in Fig. 4.18 shows the importance of having topology preservation. In the future we plan to reduce the number of computations required to compute the topology-preserving forces.

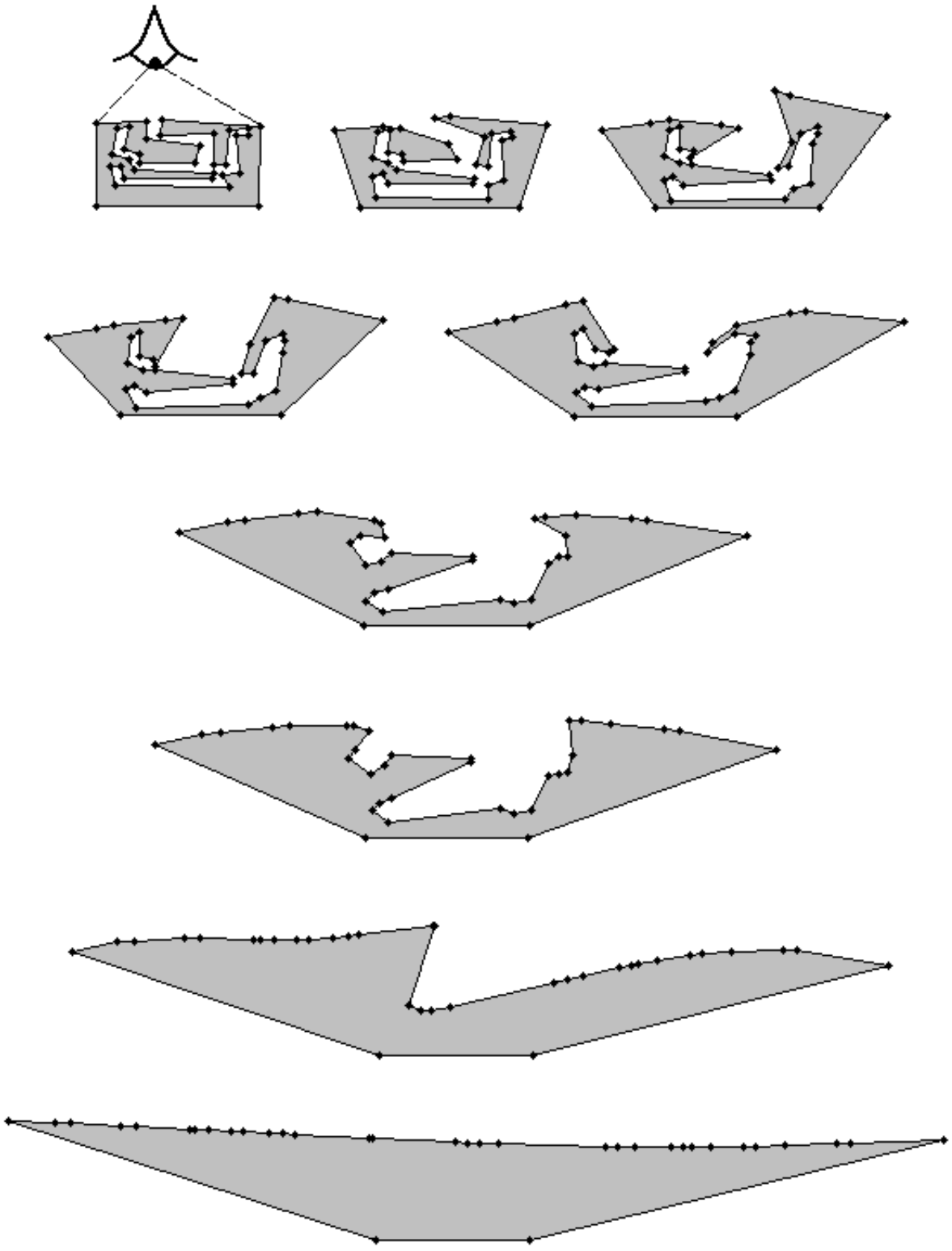

Figure 4.18: Visibility maximization for a highly convoluted polygon.

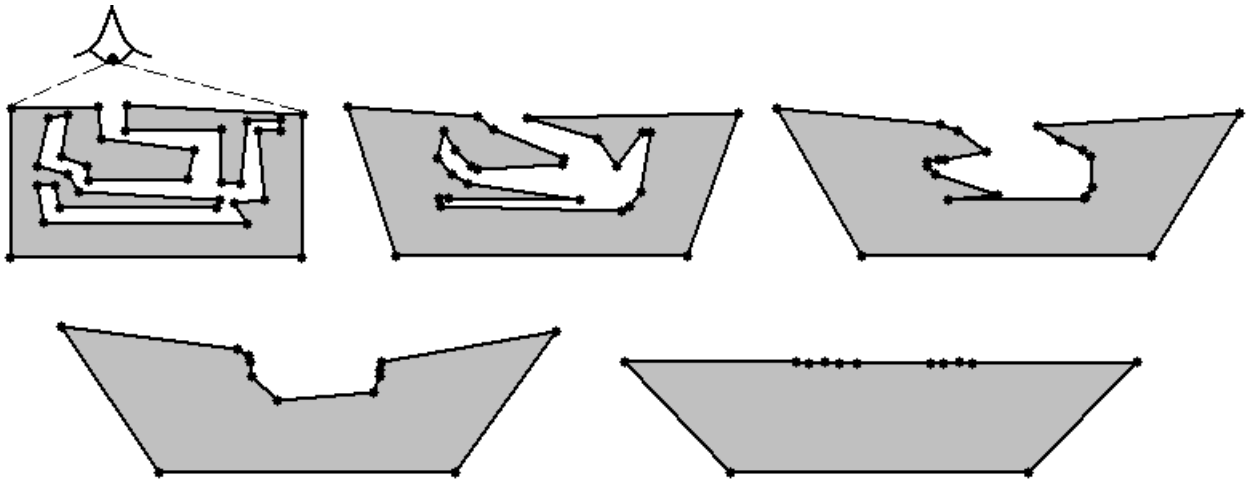

Figure 4.19: Visibility maximization of a polygon without length preservation.

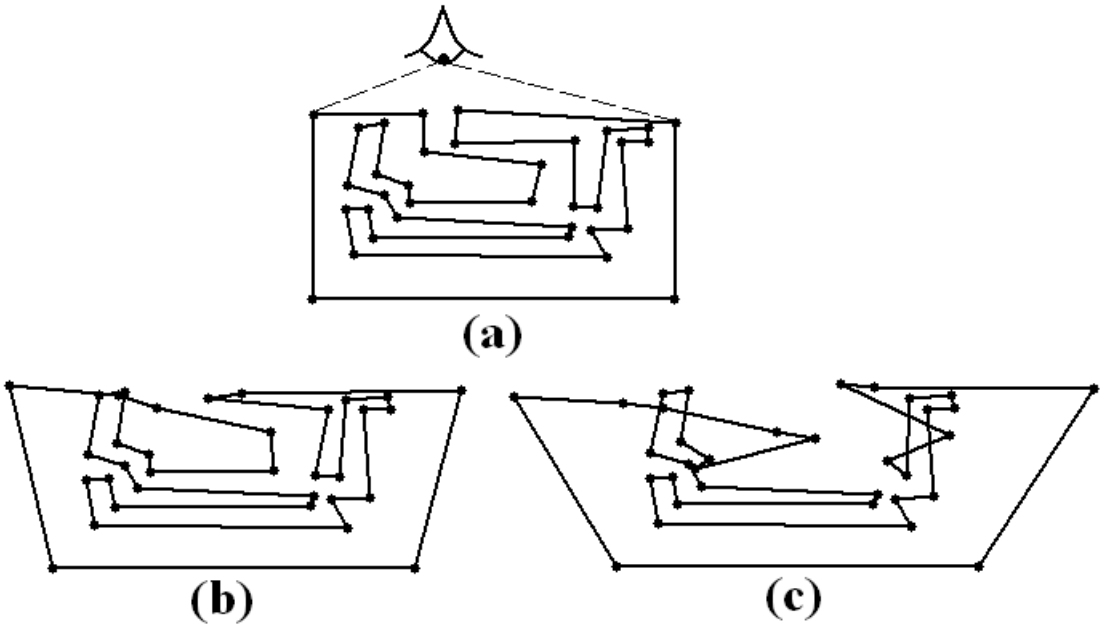

Figure 4.20: Visibility maximization of a polygon without topology preservation. (a) Initial polygon. (b) Polygon after 10 iterations. (c) Polygon after 100 iterations.

The simulation results we have presented show how effective the proposed approach can be at maximizing the average visibility of a polygon with respect to a viewpoint. Results also show that in some cases, such as the one in Fig. 4.18, the evolution can be very dramatic because of the low initial visibility. This effect can have many useful applications in computer graphics as well as in medical imaging.

CHAPTER 5

3D VISIBILITY MAXIMIZATION AND APPLICATION TO CORTICAL SHAPE UNFOLDING

In this chapter we extend our approach for 2D visibility maximization presented in the previous chapter to triangulated surfaces. That is, given a convoluted surface and an user-chosen's external viewpoint, our approach evolves the surface in such a way that it allows the user to view the deeper, self-occluded structures of the surface. Our approach has the advantage of producing little distortion in the specific region of interest as well as allowing the user to interactively determine the exact level of desired unfolding.

The proposed approach for viewpoint-based visibility maximization can have several applications in the area of medical imaging. For instance, it can be used in human brain mapping to unfold a specific part of the cerebral cortex while introducing little distortion or to visually explore and validate the deep sulcul structures extracted by cortical surface segmentation algorithms. In addition, it can have a number of applications in computer graphics. For example, it can be used for the visualization of complicated surfaces and for the visual inspection of texture mappings onto complex geometries.

The content of this chapter, which we have already submitted for publication in [98], is divided into six sections. In the first section we formulate our approach for visibility maximization for the case of differentiable surfaces. Then, in Section 2, we reformulate it for the case of triangulated surfaces. In Section 3 we extend the 2D topology-preservation method proposed by Sundaramoorthi and Yezzi [85] to triangulated surfaces. In Section 4 we propose an algorithm that improves the computational speed of the 3D topology-preservation method described in Section 3 by using a tree structure of the triangulated surface and a recursion technique. In Section 5 we show how we can preserve the area of each one of the triangular faces during the evolution by efficiently solving a linear system of equations. Finally, in Section 6, we present simulations on both synthetically created surfaces as well as cortical surfaces extracted from real data.

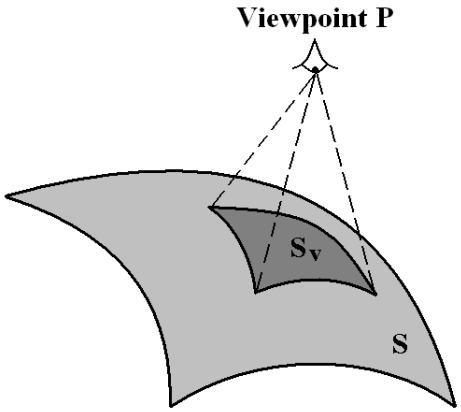

Figure 5.21: Visibility of a region of interest S_v on a 3D surface S with respect to an external viewpoint **P**.

5.1 Differentiable Surface Case

The variational approach to maximize the visibility of a contour with respect to a viewpoint that we presented in the previous chapter can be easily extended to 3D. That is, we can use a similar approach to maximize the visibility of a surface with respect to a viewpoint.

Let S be a twice-differentiable surface and let S_v be a user-chosen region of interest on S that the user would like to unfold for visualization (see Fig. 5.21). We can write the 3D version of the 2D visibility energy in (4.8) as follows:

$$E_{3D}(S_v) = \frac{1}{A_{S_v}} \int\int_{S_v} \frac{\mathbf{S}-\mathbf{P}}{\|\mathbf{S}-\mathbf{P}\|} \cdot \mathbf{N}\, dS, \tag{5.24}$$

where A_{S_v} is the area of S_v, $\mathbf{S} \in \mathbb{R}^3$ is a point in the surface, and $\mathbf{N} \in \mathbb{R}^3$ is the inward unit normal at **S**. Accordingly, $E_{3D}(S_v)$ represents the average visibility of S_v with respect to the viewpoint $\mathbf{P} \in \mathbb{R}^3$.

If we evolve S_v according to the gradient ascent of (5.24), then the points where the flux is negative would be forced to move in such a way that the flux they receive increases and becomes positive. By doing so, these points would make visible other points that already have a positive flux, but are not visible from **P**. This would generate the unfolding motion we are looking for.

Using the Calculus of Variations, it is shown in Appendix B.3 that a maximizing gradient flow for (5.24) is given by

$$\mathbf{S}_t = \frac{1}{A_{S_v}} \left(H(\mathbf{S})\, E_{3D}\,(S_v) + \frac{1}{\|\mathbf{S}-\mathbf{P}\|} \right) \mathbf{N}, \tag{5.25}$$

where $H(\mathbf{S})$ represents the mean curvature of S_v at the point $\mathbf{S}$. Like in the 2D case, this result gives us the main formulation in case we want to implement the proposed method for differentiable contours. In addition, it provides us with a mathematical criterion that tells us which part of the surface we can unfold. Specifically, this stability condition is that we must choose the portion S_v of the surface to have a positive initial average visibility with respect to $\mathbf{P}$. This may be done by enlarging or reducing the initial region S_v until this condition is satisfied. As the surface unfolds, the average visibility of S_v would increase, thereby allowing a user to select a smaller subset of S_v at later stages.

5.2 Triangulated Surface Case

Let S be now a triangulated surface and let S_v be a region of interest of S containing N triangular faces S_i, each one with area A_{S_i}. By applying (5.24) to S_v we obtain that the average visibility of S_v, which we call $E_{3D,P}$, is

$$E_{3D,P}\,(S_v) = \frac{1}{A_{S_v}} \sum_{i=1}^{N} E_{3D,i}, \tag{5.26}$$

where $A_{S_v} = A_{S_1} + \ldots + A_{S_N}$ is the total area of S_v and $E_{3D,i}$ represents the total flux received by the triangular face S_i. Accordingly, $E_{3D,i}$ is

$$E_{3D,i} = \int\int_{S_i} \frac{\mathbf{S}_i - \mathbf{P}}{\|\mathbf{S}_i - \mathbf{P}\|} \cdot \mathbf{N}_i \, dS_i, \tag{5.27}$$

where $\mathbf{S}_i$ is a point in S_i and $\mathbf{N}_i$ is the unit inward normal of the triangular face.

If $\mathbf{v}_a$, $\mathbf{v}_b$, and $\mathbf{v}_c$ are the vertexes of S_i and they are ordered counterclockwise as in Fig. 5.22, then N_i is given by

$$\mathbf{N}_i = -\frac{(\mathbf{v}_b - \mathbf{v}_a) \times (\mathbf{v}_c - \mathbf{v}_a)}{\|(\mathbf{v}_b - \mathbf{v}_a) \times (\mathbf{v}_c - \mathbf{v}_a)\|}, \tag{5.28}$$

where $\times$ is the cross product operator. Furthermore, if we parameterized S_i by using

$$\mathbf{S}_i(u,v) = \mathbf{v}_a + u\,(\mathbf{v}_b - \mathbf{v}_a) + v\,(\mathbf{v}_c - \mathbf{v}_a) \tag{5.29}$$

for $u \in [0,1]$ and $v \in [0, 1-u]$, then it is shown in Appendix B.4 that (5.27) becomes

$$E_{3D,i} = \frac{\alpha}{l_{ac}} \int_0^1 \ln\left(\frac{l_{ac}^2(1-u) + \beta + \gamma l_{ac}}{\beta + \delta\, l_{ac}}\right) du, \tag{5.30}$$

for

$$l_{ac} = \|\mathbf{v}_a - \mathbf{v}_c\|, \tag{5.31}$$

$$\alpha = -\left(\mathbf{v}_a - \mathbf{P}\right) \cdot \left[(\mathbf{v}_b - \mathbf{v}_a) \times (\mathbf{v}_c - \mathbf{v}_a)\right], \tag{5.32}$$

$$\beta = (\mathbf{v}_c - \mathbf{v}_a)\left((1-u)\mathbf{v}_a + u\,\mathbf{v}_b - \mathbf{P}\right), \tag{5.33}$$

$$\gamma = \|u\,\mathbf{v}_b + (1-u)\mathbf{v}_c - \mathbf{P}\|, \tag{5.34}$$

and

$$\delta = \|(1-u)\mathbf{v}_a + u\,\mathbf{v}_b - \mathbf{P}\|. \tag{5.35}$$

Taking the derivative of the total flux $E_{3D,i}$ received by the triangular face S_i in (5.30) with respect the vertex $\mathbf{v}_a$ we get

$$\begin{aligned} \frac{\partial E_{3D,i}}{\partial \mathbf{v}_a} &= \frac{1}{l_{ac}} \int_0^1 \ln\left(\frac{l_{ac}^2(1-u) + \beta + \gamma l_{ac}}{\beta + \delta\, l_{ac}}\right) du \left(\frac{\partial \alpha}{\partial \mathbf{v}_a} - \frac{\alpha}{l_{ac}^2}\left(\mathbf{v}_a - \mathbf{v}_c\right)\right) \\ &\quad + \frac{\alpha}{l_{ac}} \int_0^1 \left(\frac{\left(2(1-u) + \frac{\gamma}{l_{ac}}\right)(\mathbf{v}_a - \mathbf{v}_c) + \frac{\partial \beta}{\partial \mathbf{v}_a}}{l_{ac}^2(1-u) + \beta + \gamma l_{ac}} - \frac{\frac{\partial \beta}{\partial \mathbf{v}_a} + l_{ac}\frac{\partial \delta}{\partial \mathbf{v}_a} + \frac{\delta}{l_{ac}}(\mathbf{v}_a - \mathbf{v}_c)}{\beta + \delta\, l_{ac}}\right) du, \end{aligned} \tag{5.36}$$

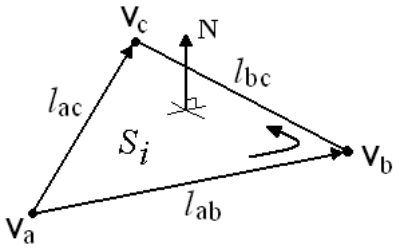

Figure 5.22: Sample triangular face S_i of a triangulated surface S.

where the partial derivatives of α, β, and δ with respect to $\mathbf{v}_a$ can be obtained from (5.32), (5.33), and (5.35), respectively (see Appendix B.4). On the other hand, if we take the derivative of the average visibility of S_v (5.26) with respect to $\mathbf{v}_a$ we obtain

$$\frac{\partial E_{3D,P}(S_v)}{\partial \mathbf{v}_a} = \frac{1}{A_{S_v}} \sum_{i=1}^{N} \left(\frac{\partial E_{3D,i}}{\partial \mathbf{v}_a} - E_{3D,P}(S_v) \frac{\partial A_i}{\partial \mathbf{v}_a} \right). \tag{5.37}$$

Using this result we have that the vertex motion for $\mathbf{v}_a$ that maximizes the average visibility of S_v with respect to the viewpoint $\mathbf{P}$ is given by

$$\frac{d\mathbf{v}_a}{dt} = \mu_{3D} \frac{\partial E_{3D,P}}{\partial \mathbf{v}_a}, \tag{5.38}$$

where μ_{3D} is a positive constant. As it was shown in the previous section, this vertex motion is stable as long as S_v has a positive average visibility with respect to the viewpoint $\mathbf{P}$ at the beginning of the evolution.

Like in the 2D case, topology changes can occur while evolving the region of interest S_v using just (5.38). These changes occur when two triangular faces intersect each other. To overcome this issue we have developed a 3D version of the work of Sundaramoorthi and Yezzi [85]. The resulting topology-preservation method is more robust than other proposed techniques such as the one by Slabaugh and Unal [94]. The details of this method have already been published in [83] and are the focus the next section.

5.3 Topology Preservation

Active surfaces, the 3D version of active contours, comprise one of the primary tools for medical image segmentation. In most medical imaging applications, the topology of the object to be segmented is known in advance. As such, a number of researchers have endeavored to incorporate various topology-preserving constraints into their evolution models for the purpose of segmentation. Some authors such as Han *et al.* [86, 87], in their work on cortical segmentation, have proposed discrete representation dependent constraints that kick in at the moment and at the location where a topology change is about to occur in order to enforce the original topology. As we saw in the previous chapter, Sundaramoorthi and Yezzi [85] recently introduced a variational method for topology preservation in active contours based on knot energies [89, 90]. Other variational approaches for topology preservation are found in the work by Shi and Karl [95], which only favors the repulsion of different connected components of the evolving curves, and in the work of Alexandrov and Santosa [96] and the recent work of Le Guyader and Vese [97], both of which are designed specifically for Level Set Methods.

Topology-preservation methods can also be applied to active polyhedron, that is, a polyhedral surface whose vertexes evolve to minimize some energy functional. In this sense, Slabaugh and Unal [94] have proposed a 3D extension of the work in [88] by adding an electric force to each vertex flow. This force is computed by creating an electric field that goes to infinity as a vertex moves towards the surface. Unfortunately, this method does not guarantee topology preservation between non-adjacent triangular faces and becomes unstable as the triangulated surface becomes finer.

For many applications, the manner in which the topology constraints are introduced is often unimportant since only the final configuration of the contour matters. Here, however, we consider an application of cortical unfolding in which the evolution itself is important to the end user who will typically wish to stop the unfolding process at any given time to obtain the desired level of unfolding. Therefore, the nature of the topology preserva-

tion should go hand-in-hand with the desired unfolding evolution and not yield undesirable transient geometric configurations that are often common when using mere topology enforcement.

The extension of the global knot-energy based topology regularizers proposed in [85] to three dimensions is conceptually straight-forward, but mathematically and computationally much more involved than the original 2D formulation. However, our effort seems to have been well justified since these types of topology-preserving energies are ideally suited to our cortical unfolding application. The resulting evolution forces, on their own, induce an unfolding effect that tends to drive the initial cortical surface towards a final spherical configuration that globally minimizes most knot energies (see [90] for the case of curves). This renders a very natural and visually pleasing global unfolding effect.

The 2D topology-preservation energy in (4.19) can be extended to 3D as follows. Let S be a triangulated surface with N triangular faces. We define the 3D topology-preservation energy as

$$E_{3D,R} = \sum_{i=1}^{N} \sum_{j=1,\, j\neq i}^{N} E_{3D,R_{i,j}}, \tag{5.39}$$

where $E_{3D,R_{i,j}}$represents the electrostatic energy between the triangular face S_i with vertexes $\mathbf{v}_a$, $\mathbf{v}_b$, and $\mathbf{v}_c$ and the triangular face S_j with vertexes $\mathbf{v}_{\hat{a}}$, $\mathbf{v}_{\hat{b}}$, and $\mathbf{v}_{\hat{c}}$. More specifically, $E_{3D,R_{i,j}}$ is defined by

$$E_{3D,R_{i,j}} = \int\int_{S_i \times S_j} \frac{1}{\left\|\mathbf{S}_i - \mathbf{S}_j\right\|^2}\, dS_i\, dS_j. \tag{5.40}$$

Accordingly, like in the 2D case, the gradient of the proposed energy in (5.39) becomes infinitely large when any two triangular faces become infinitely close. Thus preventing topology changes in the evolving triangulated surface.

Taking the derivative of $E_{3D,R}$ in (5.39) with respect to the vertex $\mathbf{v}_a$ of S_i gives us

$$\frac{\partial E_{3D,R}}{\partial \mathbf{v}_a} = \sum_{i=1}^{N} \sum_{j=1,\, j\neq i}^{N} \frac{\partial E_{3D,R_{i,j}}}{\partial \mathbf{v}_a}. \tag{5.41}$$

Furthermore, if we use the parameterization (5.29) for the triangular face S_i and the parameterization

$$\mathbf{S}_j(u,v) = \mathbf{v}_{\hat{a}} + \hat{u}\left(\mathbf{v}_{\hat{b}} - \mathbf{v}_{\hat{a}}\right) + \hat{v}\left(\mathbf{v}_{\hat{c}} - \mathbf{v}_{\hat{a}}\right) \tag{5.42}$$

for the triangular face S_j with $\hat{u} \in [0,1]$ and $\hat{v} \in [0, 1-\hat{u}]$, then it is shown in Appendix B.5 that, when S_i and S_j are non-adjacent, the derivative of $E_{3D,R_{i,j}}$ with respect to $\mathbf{v}_a$ in (5.41) becomes

$$\begin{aligned}\frac{\partial E_{3D,R_{i,j}}}{\partial \mathbf{v}_a} &= -2A_j \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} \frac{d\hat{v}\,d\hat{u}\,dv\,du}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^2} \frac{\partial}{\partial \mathbf{v}_a} \left[(\mathbf{v}_c - \mathbf{v}_b) \times \mathbf{v}_a\right]^T \mathbf{N}_i \\ &\quad -8A_iA_j \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} (1-u-v) \frac{\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^4} d\hat{v}\,d\hat{u}\,dv\,du. \end{aligned} \tag{5.43}$$

If S_i and S_j are adjacent (*i.e.*, they have at least a vertex in common), then the derivative of $E_{3D,R_{i,j}}$ with respect to $\mathbf{v}_a$ is different from (5.43) (only if $\mathbf{v}_a$ is one of the shared vertexes. It is shown in Appendix B.5 that in this case we have

$$\begin{aligned}\frac{\partial E_{3D,R_{i,j}}}{\partial \mathbf{v}_a} &= -2A_j \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} \frac{d\hat{v}\,d\hat{u}\,dv\,du}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^2} \frac{\partial}{\partial \mathbf{v}_a} \left[(\mathbf{v}_c - \mathbf{v}_b) \times \mathbf{v}_a\right]^T \mathbf{N}_i \\ &\quad -2A_i \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} \frac{d\hat{v}\,d\hat{u}\,dv\,du}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^2} \frac{\partial}{\partial \mathbf{v}_a} \left[\left(\mathbf{v}_{\hat{c}} - \mathbf{v}_{\hat{b}}\right) \times \mathbf{v}_a\right]^T \mathbf{N}_j \\ &\quad -8A_iA_j \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} (\hat{u}+\hat{v}-u-v) \frac{\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^4} d\hat{v}\,d\hat{u}\,dv\,du. \end{aligned} \tag{5.44}$$

Although the computation of (5.43) implies the numerical solution of quadruple integrals, we can reduce the number of computations by solving (5.43) explicitly *only* when the two triangular faces S_i and S_j are close enough, that is, when they are within a certain thresholded distance from each other, which is when (5.43) matters the most. On the other hand, when S_i and S_j are not considered to be close enough we can use their centroids to estimate (5.40) instead. This will make the derivative with respect to $\mathbf{v}_a$ much simpler than the expression in (5.43). Specifically , if $\bar{\mathbf{S}}_i$ and $\bar{\mathbf{S}}_j$ are the centroids of S_i and S_j,

respectively, then it is shown in Appendix B.5 that (5.43) could be approximated by

$$\begin{aligned}\frac{\partial E_{3D,R_{i,j}}}{\partial \mathbf{v}_a} &= -\frac{1}{2}A_j \frac{1}{\left\|(\mathbf{v}_a+\mathbf{v}_b+\mathbf{v}_c)-\left(\mathbf{v}_{\hat{a}}+\mathbf{v}_{\hat{b}}+\mathbf{v}_{\hat{c}}\right)\right\|^2} \frac{\partial}{\partial \mathbf{v}_a}\left[(\mathbf{v}_c-\mathbf{v}_b)\times\mathbf{v}_a\right]^T \mathbf{N}_i \\ &\quad -\frac{2}{3}A_i A_j \frac{\bar{\mathbf{S}}_i-\bar{\mathbf{S}}_j}{\left\|\bar{\mathbf{S}}_i-\bar{\mathbf{S}}_j\right\|^4}. \end{aligned} \tag{5.45}$$

Using these results together with those of the previous section, we have that the topology-preserving vertex motion that maximizes the average visibility (5.26) of the region of interest S_v with respect to the viewpoint $\mathbf{P}$ is given

$$\frac{d\mathbf{v}_a}{dt} = \mu_{3D}\frac{\partial E_{3D,P}}{\partial \mathbf{v}_a} + \mu_R \frac{\partial E_{3D,R}}{\partial \mathbf{v}_a}, \tag{5.46}$$

where μ_{3D} and μ_R are positive constants.

5.4 Recursive Computational Implementation for Topology Preservation

The topology-preservation method presented in the previous section is very robust, but at the same time is very expensive as it involves a significant amount of computations for each possible pair of triangular faces of the evolving surface. That is, we have to compute the topology-preserving forces for the vertexes of each triangular face with respect to the entire surface. Even if use (5.45) to compute the topology-preserving forces when the triangular faces are far apart, the resulting number of computations would still be large, especially considering that 3D triangulated surfaces usually have thousands of triangular faces.

We could increase the speed of the computation of the topology-preserving forces of a triangular face with respect to the rest of the surface by somehow grouping triangular faces that are far away and then using (5.45) to estimate the topology-preserving forces with respect to groups of triangular faces that are considered single units. A natural way

of doing this grouping is by creating a tree data structure, where each node of the tree represents a group of triangular faces and has a unique identification number, a centroid, an area, a list of neighbor nodes, a list of the identification number of its child nodes, and the identification number of its parent node. We can then use a recursion technique to approximate the topology-preserving forces for each triangular face of the evolving surface.

We can create a tree data structure by grouping neighbor nodes (*i.e.*, nodes that share at least one vertex) two at a time, prioritizing those neighbor nodes that are closer to each other. Every time we merge two nodes we then create a new parent node whose area is the total area of its child nodes and whose centroid is the weighted average of the centroids of its child nodes. A more detailed explanation of the merging process is described below

Merging Algorithm

1. Set all triangular faces as nodes, assign them a unique node identification number, and store the corresponding centroids and areas in each node. Tag all the nodes as UNMERGED. This set of nodes form the bottommost level of the tree.

2. For each node tagged as UNMERGED, find the UNMERGED nodes that are neighbors and store their node identification numbers in the list of neighbor nodes.

3. Compute the distance between each node and each one of its neighbors and add the information to a heap sorted by the distances.

4. Grab the pair of neighbor nodes from the top of the current heap, that is, the two closest neighbors. If any of the nodes is tagged as MERGED, then go to Step 4, otherwise tag them as MERGED and merge them by creating a parent node whose area is the sum of the two node areas and whose centroid is the weighted average of the two centroids. Create a list of child nodes in the parent node and store the parent identification number in each of the child nodes. Tag the parent node as UNMERGED. Go to Step 4.

5. Create a new heap sorted by distances with the elements of the previous heap that have one node tagged as UNMERGED.

6. Grab the pair of neighbor nodes from the top of new heap and add the node tagged as UNMERGED to the list of child nodes of the parent node of the node tagged as MERGED. Tag the UNMERGED node as MERGED and update the area, the centroid, and the list of child nodes of the parent node. Store the number identification of the parent node in the child node. Stop if the number of UNMERGED nodes is equal to one; else go to Step 2.

Storing the parent identification number in its child nodes is very useful when finding the neighbor nodes in Step 2. That is, if we want to find neighbors of a parent node we just need

to find the neighbors its child nodes. Then, the identification number of the parent nodes of those neighbors are the identification numbers of the neighbors of the given parent node.

Figure 5.23(a) depicts a tree structure for a surface containing nine triangular faces. As can be seen, in the bottommost level of the tree we have the original nine triangular faces of the surface. After going from Step 1 to Step 6 we have the first level of new parent nodes (*nodes* 10 to 13). We can see that even though most of the parent nodes have only two child nodes, it is still possible to have more child nodes. That is the case of *node* 11, which has three child nodes (*nodes* 3 to 5). The reason for this is that even though we want to have two child nodes per parent node, this is not always possible due to the structure of the triangulation of the surface. After going from Step 1 to Step 6 another time we have only two new parent nodes (*nodes* 14 and 15). Finally, after going from Step 1 to Step 6 for the last time, the algorithm produces only one new parent node, *node* 16, which is called the root node. It comprises all the triangular faces of the surface.

We can exploit the tree structure for the estimation of the topology-preserving forces by using recursion as follows. Say for instance that we want to compute the topology-preserving forces for the vertexes of a particular triangular face with respect to a node that does not contain the triangular face and that has two or more child nodes and many more nodes below it. One thing we can do is to estimate the topology-preserving forces by using (5.45) and the centroids and areas of both the triangular face and the parent node. We can do exactly the same thing with each one of the child nodes. We can then compare the results to determine whether or not more accuracy is needed. If the average relative error of the result obtained using only the parent node with respect to the sum of the results obtained by using each child node is less than or equal to some chosen value, then we can consider the approximation to be good enough and, as a result, we can assume that the sum the topology-preserving forces with respect to the child nodes is a good approximation of the topology-preserving forces with respect to all the triangular faces below the parent node. If instead, the average relative error is greater than the chosen value, then we have to

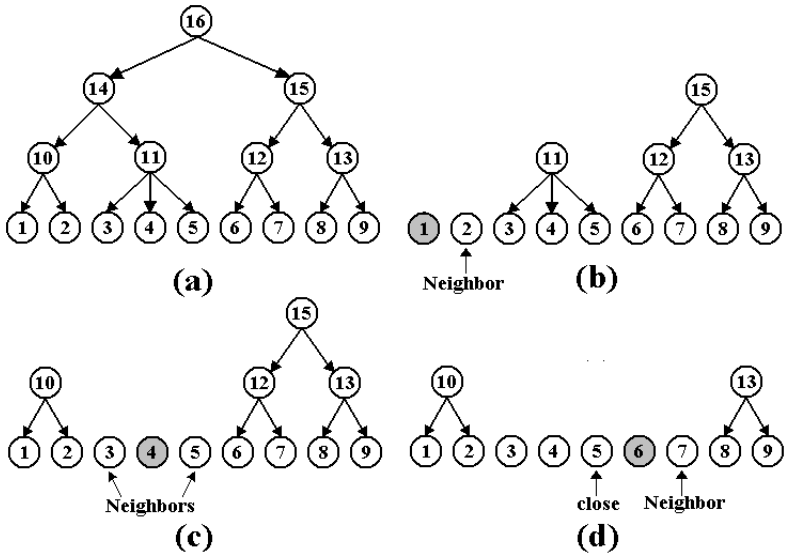

Figure 5.23: Changes in the tree structure. (a) Original tree structure. (b) - (c) Resulting tree structures when computing the topology-preserving forces with respect to the *nodes* 1, 4, and 6, respectively.

repeat the procedure again with each one of the child nodes, where this time they would be considered as parent nodes of some other child nodes.

This recursion technique gives us a very powerful resource as we are estimating topology-preserving forces with respect to a group of triangular faces at a time instead of a single triangular face at a time. Moreover, it automatically decides when more accuracy is needed.

We can also increase the accuracy of the computation of the topology-preserving forces by explicitly computing them when they matter the most, that is, when two triangular faces are very close or when and they are neighbors. For those cases we would use (5.43) and (5.44) as needed. The resulting algorithm, which is fast and at the same time accurate, is described in more detail below.

Fast Algorithm to Compute Topology-Preserving Forces

For each triangular face of the surface do the following:

1. Tag all nodes in the tree as UNVISITED, except for the selected triangular face, which is in the bottommost level of the tree.

2. For each one of the nodes in the bottommost level of the tree that are within a thresholded distance from the selected triangular face or are neighbors of the selected triangular face compute the accurate topology-preserving forces for its vertexes by using (5.43) and (5.44), respectively. Tag those nodes as VISITED.

3. For each one of the nodes tagged as VISITED follow their paths up to the root node, the topmost node, and tag as VISITED all the parent nodes in the paths.

4. Select the UNVISITED node with the highest identification number and compute the topology-preserving forces for the vertexes of the selected triangular face with respect to this node and all the nodes below it by using recursion and the estimation formula in (5.45). Add the results to the overall topology-preserving forces for the vertexes of the selected triangular face. Tag all these nodes as VISITED.

5. Stop if all nodes are tagged as VISITED; else go to Step 4.

Figure 5.23 shows several examples of how this algorithm works. In Fig. 5.23(b) it is shown the resulting tree when computing the topology-preserving forces of the surface with respect to *node* 1, which is shown in gray. In this case we just need to compute the topology-preserving forces accurately once, for *node* 2, which is the only neighbor node that *node* 1 has. We then have to use recursion to compute the topology-preserving forces starting with *node* 15. Of course, in the case of *node* 11, it makes more sense just to compute the topology-preserving forces with respect to each of the *nodes* 3, 4, and 5 by using (5.45). Figs. 5.23(c) and 5.23(d) show the resulting tree structures when *node* 4 and

node 6 are selected. As can be seen, different tree structures can result. However, they can be easily obtained from Fig. 5.23(a) by deleting the nodes that are in the paths from the neighbor nodes of the selected node to the root node (*node* 16) and from the nodes that are close enough to the selected node to the root node.

As it is going to be seen in the last section of this chapter, the improvement in speed by using a tree structure and a recursion technique can be enormous.

5.5 Area Preservation

For a region of interest S_v on a triangulated surface S to have an average visibility equal to 1 with respect to a viewpoint, it has to have a visibility equal to 1 at every point of each one of its triangular faces. Similar to the 2D polygon case, it is easy to see that this only occurs when all triangular faces collapse to a single point. To overcome this problem we need to maximize the average visibility and, at the same time, maintain the area of each one of the triangular faces of S_v constant. This can be done by applying the same technique we used in the previous chapter to preserve the lengths of the evolving polygon. That is, for the evolution we will only use the component of the gradient that does not change the area. As we mentioned in Chapter 4, a similar technique has been employed before in [91, 92, 93]. We described it below for our specific case.

Let $\Psi = \{\mathbf{v}_1, \ldots, \mathbf{v}_M\}$ be the ordered set of the M vertexes of S_v. To maintain a constant area for each of the N triangular faces S_i of S_v during the evolution we need to satisfy the following N constraints:

$$\frac{1}{2}\left\|\left(\mathbf{v}_{i,b}-\mathbf{v}_{i,a}\right)\times\left(\mathbf{v}_{i,c}-\mathbf{v}_{i,a}\right)\right\|^2 - A_i^2 = 0, \tag{5.47}$$

where $i = \{1, \ldots, N\}$ and $\mathbf{v}_{i,a}$, $\mathbf{v}_{i,b}$, and $\mathbf{v}_{i,c}$ represent the corresponding vertexes of S_i in Ψ.

Using Lagrange multipliers one can obtain

$$V_{t,c} = V_{t,u} - J^T \mathbf{l}, \tag{5.48}$$

where $V_{t,u}$ is the vector of unconstrained gradient flow obtained by applying (5.46) to each vertex of S_v, J is the Jacobian matrix of the N constraints in (5.47), $V_{t,c}$ is the vector of constrained gradient flow that we will use to evolve S_v, and $\mathbf{l}$ is the minimum norm solution vector to the system

$$JJ^T \mathbf{l} = JV_{t,u}. \tag{5.49}$$

Since the matrix JJ^T is symmetric positive definite, then (5.49) can be quickly computed using the conjugate gradient method. Moreover, since J is sparse (*i.e.*, most of its elements are equal to zero), the matrix multiplication JJ^T can be computed and stored efficiently.

5.6 Application to Cortical Unfolding

In this section we present some simulation results that show how the proposed approach for visibility maximization can be used for local unfolding.

Figure 5.24 depicts the evolution of a 3D synthetic surface when the visibility is maximized and, at the same time, the topology is preserved. The initial visibility of this surface was 0.2, whereas by the end of the simulation it was very close to 1.

Topology preservation plays a very important role when the surface for which the visibility is going to be maximized is very convoluted. This is the case of the evolution shown in Fig. 5.25, where a region of a cortex is evolved over 150 iterations so that the visibility with respect to a viewpoint located just in front of the region is maximized. This region consisted of about 1,500 triangular faces. Using the method of topology preservation as described in Section 3 took about 20 minutes per iteration on a 2.53 GHz computer running Windows. However, using the fast algorithm described in Section 4 reduced the

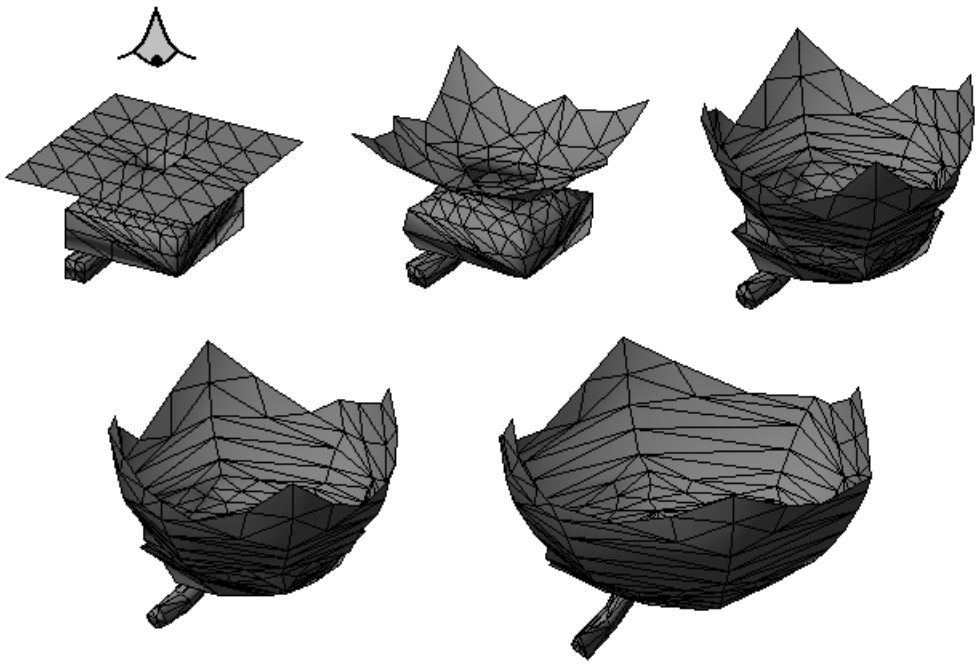

Figure 5.24: Unfolding of a synthetic triangulated surface using the proposed visibility-maximization approach.

computational time to just 5 second per iteration.

Figures 5.26 and 5.27 show additional evolutions in which one can clearly see how the selected regions unfold to become more visible from the external viewpoints.

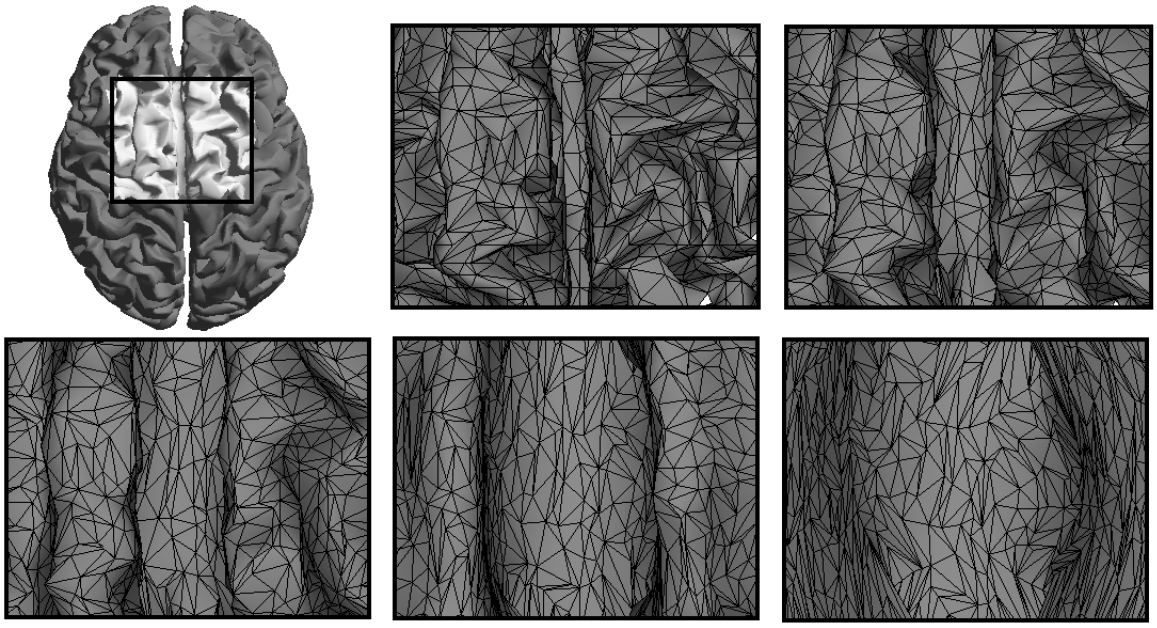

Figure 5.25: Unfolding of a region of a cortex using visibility maximization.

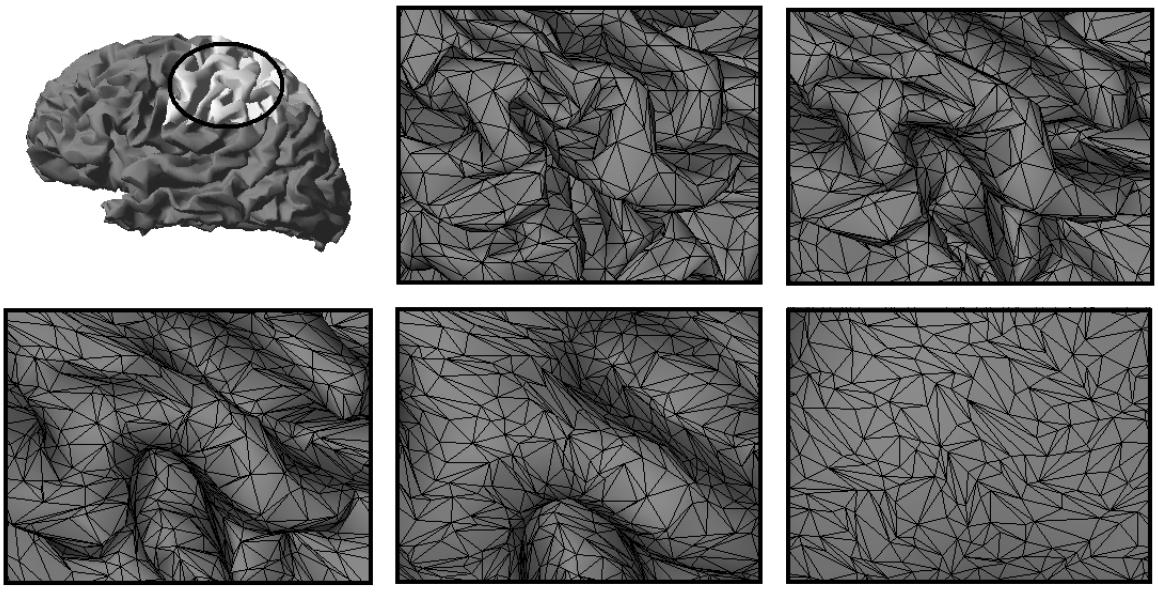

Figure 5.26: Unfolding of a region on the right side of a cortex.

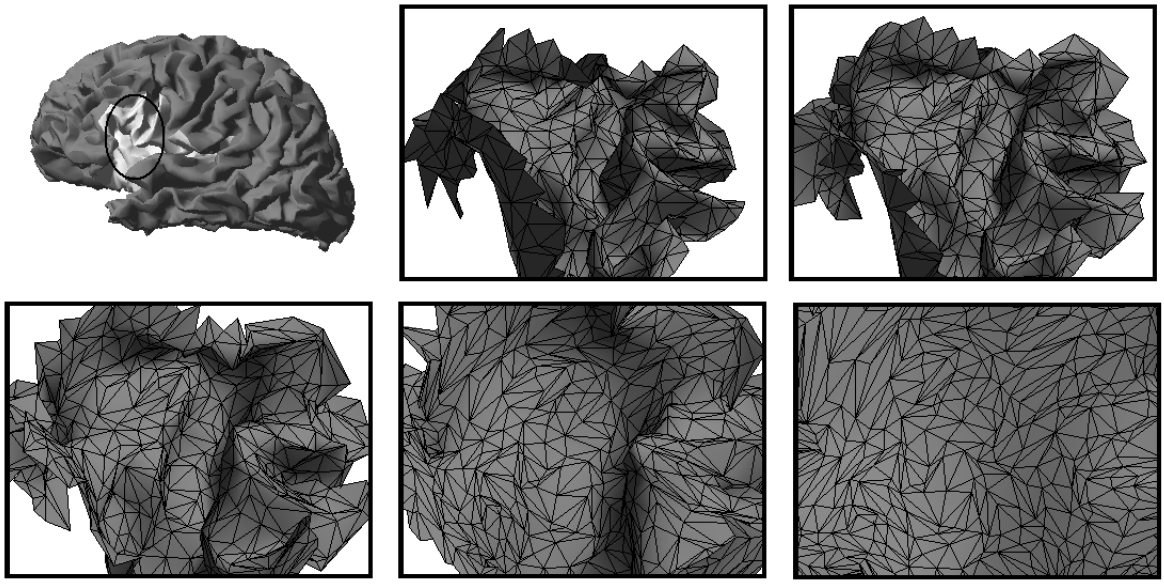

Figure 5.27: Unfolding of a region of a cortex where one can clearly see how the surface unfolds.

CHAPTER 6

CONCLUSION

We have presented the novel idea of visibility maximization by defining a viewpoint-based visibility energy and creating a three dimensional generalization of knot-energy type forces for topology preservation. Simulation results show that the gradient flow of these combined energy terms yields a useful localized unfolding of triangulated surfaces specially tailored to the user's viewpoint. We believe this method, compared with more traditional global flattening techniques, may be very useful for more customized inspection of 3D surfaces, such as cortical surface segmentations.

The fact that visibility maximization is a new concept leaves the door open to new applications and extensions of the proposed approach. For instance, the idea of visibility maximization with respect to a viewpoint can be easily extended to visibility maximization with respect to a line or even to a surface by simply adding a line or surface integral to our original visibility formulation.

Besides creating a robust method for 3D topology preservation that can be used for many others active polyhedron applications beyond visibility maximization, we have also been able to make a significant increase in computational speed for the proposed 3D topology-preservation method by using a tree structure and a recursion technique. This makes the proposed method more suitable for real-time applications.

There is still room for improvement in terms of speed for the proposed visibility-maximization approach. For example, we could use a method similar to the one we used to speed up the computation of the topology-preserving forces to speed up the computation of the visibility-maximizing forces. In addition, it is possible that a different merging algorithm to create the tree structure could improve the speed of the topology-preservation approach. Moreover, the use of adaptive time steps could improve the speed of the convergence of the evolving surface.

APPENDIX A: HYBRID EULERIAN-LAGRANGIAN ALGORITHM

Hybrid Eulerian-Lagrangian Algorithm to compute ϕ_0

1. Initially tag all grid points in R as UNVISITED.

2. Use the Lagrangian approach to compute the values of ϕ_0 at grid points in R adjacent to the boundary $\partial_0 R$ (grid points with at least 1 neighbor outside R) and re-tag them as SOLVED.

3. Use (3.1) to compute the value of ϕ_0 at grid points in R next to the points already tagged as SOLVED, tag them as VISITED, and put them into a heap sorted by the values of u.

4. Grab the grid point from the top of the current heap of VISITED points (*i.e.*, the grid point with the smallest value of u). Remove this point from the heap and tag it as SOLVED.

5. If the distance between the correspondence values of ϕ_0 at the neighboring grid points used in (6) is less than the desired tolerance λ, then compute ϕ_0 using (3.1) and go to Step 7.

6. Follow the correspondence trajectory at the current grid point until it intersects the horizontal or vertical grid line between two grid points tagged as SOLVED and located 1 cell away from each other. If the distance between the values of ϕ_0 at these new grid points is greater than or equal to λ, then go to Step 6, else compute ϕ_0 using linear interpolation and assign the resulting value to ϕ_0 of the original grid point of Step 5.

7. Update the values of ϕ_0 using (3.1) for whichever neighbors of this grid point are not yet tagged as SOLVED. If any of these neighbors are currently tagged as UNVISITED, re-tag them as VISITED and add them to the current heap of VISITED grid points.

8. Stop if all points in R have been tagged SOLVED, else go to Step 4.

APPENDIX B: DERIVATIONS

B.1 Gradient Ascent for the Average Visibility Energy of a 2D Differentiable Contour

Let C be a twice-differentiable contour and let $\mathbf{P}$ be a viewpoint located outside C. We define the average visibility of a region of interest of C with respect to $\mathbf{P}$ as

$$E_{2D}(C) = \frac{1}{L}\int_0^L \frac{\mathbf{C}(s)-\mathbf{P}}{\|\mathbf{C}(s)-\mathbf{P}\|}\cdot\mathbf{N}(s)\,ds, \tag{50}$$

where $\mathbf{N}(s)$ is the inward unit normal of C at $\mathbf{C}(s)$ and where we have assumed that the region of interest has length L. Taking the derivative with respect to time t of this energy we get

$$\begin{aligned}
E'_{2D}(C) &= \frac{d}{dt}\left[\frac{1}{L}\int_0^L \frac{\mathbf{C}(s)-\mathbf{P}}{\|\mathbf{C}(s)-\mathbf{P}\|}\cdot\mathbf{N}(s)\,ds\right] \\
&= \int_0^L \frac{\mathbf{C}(s)-\mathbf{P}}{\|\mathbf{C}(s)-\mathbf{P}\|}\cdot\mathbf{N}(s)\,ds\,\frac{d}{dt}\frac{1}{L} + \frac{1}{L}\frac{d}{dt}\int_0^L \frac{\mathbf{C}(s)-\mathbf{P}}{\|\mathbf{C}(s)-\mathbf{P}\|}\cdot\mathbf{N}(s)\,ds \\
&= -\frac{1}{L^2}\int_0^L \frac{\mathbf{C}(s)-\mathbf{P}}{\|\mathbf{C}(s)-\mathbf{P}\|}\cdot\mathbf{N}(s)\,ds\,\frac{dL}{dt} + \frac{1}{L}\frac{d}{dt}\int_0^L \frac{\mathbf{C}(s)-\mathbf{P}}{\|\mathbf{C}(s)-\mathbf{P}\|}\cdot J^T\mathbf{C}_s(s)\,ds \\
&= -\frac{1}{L}E_{2D}(C)\,\frac{d}{dt}\int_0^L ds + \frac{1}{L}\frac{d}{dt}\int_0^1 \frac{\mathbf{C}(p)-\mathbf{P}}{\|\mathbf{C}(p)-\mathbf{P}\|}\cdot J^T\mathbf{C}_p(p)\,dp \\
&= \frac{1}{L}E_{2D}(C)\int_0^L \mathbf{C}_t(s)\cdot\kappa(s)\mathbf{N}(s)\,ds + \frac{1}{L}\int_0^1 \left(\frac{\mathbf{C}(p)-\mathbf{P}}{\|\mathbf{C}(p)-\mathbf{P}\|}\right)_t\cdot J^T\mathbf{C}_p(p) \\
&\quad + \frac{\mathbf{C}(p)-\mathbf{P}}{\|\mathbf{C}(p)-\mathbf{P}\|}\cdot J^T\mathbf{C}_{tp}(p)\,dp,
\end{aligned}$$

where J is the inversion matrix and $E_{2D}(C)$ is the average visibility as defined in (50). Now, using integration by parts, and taking into consideration that $\mathbf{C}_t$ is zero at the boundary points of the region of interest, we get

$$\begin{aligned}
E'_{2D}(C) &= \frac{1}{L}E_{2D}(C)\int_0^L \mathbf{C}_t(s)\cdot\kappa(s)\mathbf{N}(s)\,ds \\
&\quad + \frac{1}{L}\int_0^1 \frac{1}{\|\mathbf{C}(p)-\mathbf{P}\|}\left[\mathbf{C}_t(p) - \left(\mathbf{C}_t(p)\cdot\frac{\mathbf{C}(p)-\mathbf{P}}{\|\mathbf{C}(p)-\mathbf{P}\|}\right)\frac{\mathbf{C}(p)-\mathbf{P}}{\|\mathbf{C}(p)-\mathbf{P}\|}\right]\cdot J^T\mathbf{C}_p(p)\,dp
\end{aligned}$$

$$-\frac{1}{L}\int_0^1 \left(\frac{\mathbf{C}(p)-\mathbf{P}}{\|\mathbf{C}(p)-\mathbf{P}\|}\right)_p \cdot J^T \mathbf{C}_t(p)\,dp$$

$$\begin{aligned}
= \quad & \frac{1}{L}E_{2D}(C)\int_0^L \mathbf{C}_t(s)\cdot\kappa(s)\mathbf{N}(s)\,ds \\
& +\frac{1}{L}\int_0^1 \frac{1}{\|\mathbf{C}(p)-\mathbf{P}\|}\left(\mathbf{C}_t(p)\cdot\frac{J(\mathbf{C}(p)-\mathbf{P})}{\|\mathbf{C}(p)-\mathbf{P}\|}\right)\frac{J(\mathbf{C}(p)-\mathbf{P})}{\|\mathbf{C}(p)-\mathbf{P}\|}\cdot J^T\mathbf{C}_p(p)\,dp \\
& -\frac{1}{L}\int_0^1 \frac{1}{\|\mathbf{C}(p)-\mathbf{P}\|}\left[\mathbf{C}_p(p)-\left(\mathbf{C}_p(p)\cdot\frac{\mathbf{C}(p)-\mathbf{P}}{\|\mathbf{C}(p)-\mathbf{P}\|}\right)\frac{\mathbf{C}(p)-\mathbf{P}}{\|\mathbf{C}(p)-\mathbf{P}\|}\right]\cdot J^T\mathbf{C}_t(p)\,dp,
\end{aligned}$$

where we have used the fact that the unit vectors $\frac{\mathbf{C}(p)-\mathbf{P}}{\|\mathbf{C}(p)-\mathbf{P}\|}$ and $\frac{J(\mathbf{C}(p)-\mathbf{P})}{\|\mathbf{C}(p)-\mathbf{P}\|}$ form an orthogonal basis. Rearranging terms, we obtain

$$\begin{aligned}
E_{2D}^{'}(C) \quad = \quad & \frac{1}{L}E_{2D}(C)\int_0^L \mathbf{C}_t(s)\cdot\kappa(s)\mathbf{N}(s)\,ds \\
& -\frac{1}{L}\int_0^L \mathbf{C}_t(s)\cdot\frac{1}{\|\mathbf{C}(s)-\mathbf{P}\|}\left(\frac{\mathbf{C}(s)-\mathbf{P}}{\|\mathbf{C}(s)-\mathbf{P}\|}\cdot\mathbf{C}_s(s)\right)\frac{J(\mathbf{C}(s)-\mathbf{P})}{\|\mathbf{C}(s)-\mathbf{P}\|}\,ds \\
& -\frac{1}{L}\int_0^1 \frac{1}{\|\mathbf{C}(p)-\mathbf{P}\|}\left(\mathbf{C}_p(p)\cdot\frac{J(\mathbf{C}(p)-\mathbf{P})}{\|\mathbf{C}(p)-\mathbf{P}\|}\right)\frac{J(\mathbf{C}(p)-\mathbf{P})}{\|\mathbf{C}(p)-\mathbf{P}\|}\cdot J^T\mathbf{C}_t(p)\,dp
\end{aligned}$$

$$\begin{aligned}
= \quad & \frac{1}{L}E_{2D}(C)\int_0^L \mathbf{C}_t(s)\cdot\kappa(s)\mathbf{N}(s)\,ds \\
& -\frac{1}{L}\int_0^L \mathbf{C}_t(s)\cdot\frac{1}{\|\mathbf{C}(s)-\mathbf{P}\|}\left(\mathbf{C}_s(s)\cdot\frac{\mathbf{C}(s)-\mathbf{P}}{\|\mathbf{C}(s)-\mathbf{P}\|}\right)\frac{J(\mathbf{C}(s)-\mathbf{P})}{\|\mathbf{C}(s)-\mathbf{P}\|}\,ds \\
& +\frac{1}{L}\int_0^L \frac{1}{\|\mathbf{C}(s)-\mathbf{P}\|}\left(\frac{\mathbf{C}(s)-\mathbf{P}}{\|\mathbf{C}(s)-\mathbf{P}\|}\cdot\mathbf{C}_t(s)\right)\frac{J(\mathbf{C}(s)-\mathbf{P})}{\|\mathbf{C}(s)-\mathbf{P}\|}\cdot\mathbf{C}_s(s)\,ds
\end{aligned}$$

$$\begin{aligned}
= \quad & \frac{1}{L}E_{2D}(C)\int_0^L \mathbf{C}_t(s)\cdot\kappa(s)\mathbf{N}(s)\,ds \\
& -\frac{1}{L}\int_0^L \mathbf{C}_t(s)\cdot\frac{1}{\|\mathbf{C}(s)-\mathbf{P}\|}\left(\mathbf{C}_s(s)\cdot\frac{\mathbf{C}(s)-\mathbf{P}}{\|\mathbf{C}(s)-\mathbf{P}\|}\right)\frac{J(\mathbf{C}(s)-\mathbf{P})}{\|\mathbf{C}(s)-\mathbf{P}\|}\,ds \\
& +\frac{1}{L}\int_0^L \mathbf{C}_t(s)\cdot\frac{1}{\|\mathbf{C}(s)-\mathbf{P}\|}\left(\mathbf{C}_s(s)\cdot\frac{J(\mathbf{C}(s)-\mathbf{P})}{\|\mathbf{C}(s)-\mathbf{P}\|}\right)\frac{\mathbf{C}(s)-\mathbf{P}}{\|\mathbf{C}(s)-\mathbf{P}\|}\,ds.
\end{aligned}$$

Continuing we get

$$\begin{aligned}
E_{2D}^{'}(C) \quad = \quad & \frac{1}{L}E_{2D}(C)\int_0^L \mathbf{C}_t(s)\cdot\kappa(s)\mathbf{N}(s)\,ds \\
& +\frac{1}{L}\int_0^L \mathbf{C}_t(s)\cdot\frac{1}{\|\mathbf{C}(s)-\mathbf{P}\|}\left[\left(\mathbf{C}_s(s)\cdot\frac{J(\mathbf{C}(s)-\mathbf{P})}{\|\mathbf{C}(s)-\mathbf{P}\|}\right)\frac{\mathbf{C}(s)-\mathbf{P}}{\|\mathbf{C}(s)-\mathbf{P}\|}\right.
\end{aligned}$$

$$- \left(\mathbf{C}_s(s) \cdot \frac{\mathbf{C}(s) - \mathbf{P}}{\|\mathbf{C}(s) - \mathbf{P}\|} \right) \frac{J(\mathbf{C}(s) - \mathbf{P})}{\|\mathbf{C}(s) - \mathbf{P}\|} \Bigg] ds$$

$$= \frac{1}{L} E_{2D}(C) \int_0^L \mathbf{C}_t(s) \cdot \kappa(s) \mathbf{N}(s)\, ds + \frac{1}{L} \int_0^L \mathbf{C}_t(s) \cdot \frac{1}{\|\mathbf{C}(s) - \mathbf{P}\|} J^T \mathbf{C}_s(s)\, ds$$

$$= \int_0^L \left\langle \mathbf{C}_t(s), \frac{1}{L} \left(\kappa(s) E_{2D}(C) + \frac{1}{\|\mathbf{C}(s) - \mathbf{P}\|} \right) \mathbf{N}(s) \right\rangle ds.$$

Therefore, the gradient flow that maximizes the average visibility of the region of interest of the contour C with respect to the viewpoint $\mathbf{P}$ is

$$\mathbf{C}_t(s) = \frac{1}{L} \left(\kappa(s) E_{2D}(C) + \frac{1}{\|\mathbf{C}(s) - \mathbf{P}\|} \right) \mathbf{N}(s). \tag{51}$$

Since $\mathbf{P}$ is located outside C, $\|\mathbf{C}(s) - \mathbf{P}\|$ is nonzero as long as the evolving contour does not touch $\mathbf{P}$.

B.2 Derivatives Used in the Flux Model

Partial derivatives of the energies $E_{2D,k}$ and $E_{2D,k-1}$ with respect to the vertex $\mathbf{v}_k$:

$$\begin{aligned}
\frac{\partial E_{2D,k}}{\partial \mathbf{v}_k} &= \frac{1}{|C_k|}\left[\ln\left(|C_k|+\beta_k+\|\mathbf{v}_{k+1}-\mathbf{P}\|\right)-\ln\left(\beta_k+\|\mathbf{v}_k-\mathbf{P}\|\right)\right]\left(\frac{\partial \gamma_k}{\partial \mathbf{v}_k}\right. \\
&\left.-\frac{\gamma_k}{|C_k|^2}\left(\mathbf{v}_k-\mathbf{v}_{k+1}\right)^T\right) \\
&+\frac{\gamma_k}{|C_k|}\left[\frac{1}{|C_k|+\beta_k+\|\mathbf{v}_{k+1}-\mathbf{P}\|}\left(\frac{1}{|C_k|}\left(\mathbf{v}_k-\mathbf{v}_{k+1}\right)^T+\frac{\partial \beta_k}{\partial \mathbf{v}_k}\right)\right. \\
&\left.-\frac{1}{\beta_k+\|\mathbf{v}_k-P\|}\left(\frac{\partial \beta_k}{\partial \mathbf{v}_k}+\frac{\left(\mathbf{v}_k-\mathbf{P}\right)^T}{\|\mathbf{v}_k-\mathbf{P}\|}\right)\right] \qquad (52)
\end{aligned}$$

$$\begin{aligned}
\frac{\partial E_{2D,k-1}}{\partial \mathbf{v}_k} &= \frac{1}{|C_{k-1}|}\left[\ln\left(|C_{k-1}|+\beta_{k-1}+\|\mathbf{v}_k-\mathbf{P}\|\right)-\ln\left(\beta_{k-1}+\|\mathbf{v}_{k-1}-\mathbf{P}\|\right)\right]\left(\frac{\partial \gamma_{k-1}}{\partial \mathbf{v}_k}\right. \\
&\left.-\frac{\gamma_{k-1}}{|C_{k-1}|^2}\left(\mathbf{v}_k-\mathbf{v}_{k-1}\right)^T\right) \\
&+\frac{\gamma_{k-1}}{|C_{k-1}|^2}\left[\frac{1}{|C_{k-1}|+\beta_{k-1}+\|\mathbf{v}_k-\mathbf{P}\|}\left(\frac{1}{|C_{k-1}|}\left(\mathbf{v}_k-\mathbf{v}_{k-1}\right)^T+\frac{\partial \beta_{k-1}}{\partial \mathbf{v}_k}+\frac{\left(\mathbf{v}_k-\mathbf{P}\right)^T}{\|\mathbf{v}_k-\mathbf{P}\|}\right)\right. \\
&\left.-\frac{1}{\beta_{k-1}+\|\mathbf{v}_{k-1}-\mathbf{P}\|}\left(\frac{\partial \beta_{k-1}}{\partial \mathbf{v}_k}\right)\right] \qquad (53)
\end{aligned}$$

Auxiliary Derivatives:

$$\begin{aligned}
\frac{\partial \beta_k}{\partial \mathbf{v}_k} &= \frac{\partial}{\partial \mathbf{v}_k}\frac{\left(\mathbf{v}_{k+1}-\mathbf{v}_k\right)^T\left(\mathbf{v}_k-\mathbf{P}\right)}{|C_k|}=-\frac{\beta_k}{|C_k|^2}\left(\mathbf{v}_k-\mathbf{v}_{k+1}\right)^T+\frac{\left(\mathbf{v}_{k+1}-2\mathbf{v}_k+\mathbf{P}\right)^T}{|C_k|} \\
\frac{\partial \beta_{k-1}}{\partial \mathbf{v}_k} &= \frac{\partial}{\partial \mathbf{v}_k}\frac{\left(\mathbf{v}_k-\mathbf{v}_{k-1}\right)^T\left(\mathbf{v}_{k-1}-\mathbf{P}\right)}{|C_{k-1}|}=-\frac{\beta_{k-1}}{|C_{k-1}|^2}\left(\mathbf{v}_k-\mathbf{v}_{k-1}\right)^T+\frac{\left(\mathbf{v}_{k-1}-\mathbf{P}\right)^T}{|C_{k-1}|} \\
\frac{\partial \gamma_k}{\partial \mathbf{v}_k} &= \frac{\partial}{\partial \mathbf{v}_k}\left[\left(\mathbf{v}_k-\mathbf{P}\right)^T J^T\left(\mathbf{v}_{k+1}-\mathbf{v}_k\right)\right]=\left(\mathbf{v}_{k+1}-\mathbf{P}\right)^T J \\
\frac{\partial \gamma_{k-1}}{\partial \mathbf{v}_k} &= \frac{\partial}{\partial \mathbf{v}_k}\left[\left(\mathbf{v}_{k-1}-\mathbf{P}\right)^T J^T\left(\mathbf{v}_k-\mathbf{v}_{k-1}\right)\right]=\left(\mathbf{v}_{k-1}-\mathbf{P}\right)^T J^T
\end{aligned}$$

B.3 Gradient Ascent for the Average Visibility Energy of a 3D Differentiable Surface

Let S be a twice-differentiable surface of area A and let $\mathbf{P}$ be a viewpoint located outside S. We define the average visibility of S with respect to $\mathbf{P}$ as

$$E_{3D}(S) = \frac{1}{A}\int\int_S \frac{\mathbf{S}-\mathbf{P}}{\|\mathbf{S}-\mathbf{P}\|}\cdot\mathbf{N}\,dS, \tag{54}$$

where $\mathbf{N}$ is the inward unit normal at the surface point $\mathbf{S}$. Let $u \in [0,1]$ and $v \in [0,1]$ be parameterization variables over which the integral above will be evaluated. Using this parameterization (54) becomes

$$\begin{aligned} E_{3D}(S) &= \frac{1}{A}\int_0^1\int_0^1 \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\cdot\frac{\mathbf{S}_v\times\mathbf{S}_u}{\|\mathbf{S}_v\times\mathbf{S}_u\|}\,\|\mathbf{S}_u\times\mathbf{S}_v\|\,du\,dv \\ &= \frac{1}{A}\int_0^1\int_0^1 \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\cdot(\mathbf{S}_v\times\mathbf{S}_u)\,du\,dv. \end{aligned} \tag{55}$$

Taking the derivative of (55) with respect to time we get

$$\begin{aligned} E'_{3D}(S) &= \frac{d}{dt}\left[\frac{1}{A}\int_0^1\int_0^1 \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\cdot(\mathbf{S}_v\times\mathbf{S}_u)\,du\,dv\right] \\ &= \int_0^1\int_0^1 \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\cdot(\mathbf{S}_v\times\mathbf{S}_u)\,du\,dv\,\frac{d}{dt}\frac{1}{A} \\ &\quad +\frac{1}{A}\int_0^1\int_0^1 \frac{d}{dt}\left[\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\cdot(\mathbf{S}_v\times\mathbf{S}_u)\right]du\,dv \\ &= -\frac{1}{A^2}\int_0^1\int_0^1 \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\cdot(\mathbf{S}_v\times\mathbf{S}_u)\,du\,dv\,\frac{d}{dt}A \\ &\quad +\frac{1}{A}\int_0^1\int_0^1 \left[\left(\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)_t\cdot(\mathbf{S}_v\times\mathbf{S}_u)+\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\cdot(\mathbf{S}_v\times\mathbf{S}_u)_t\right]du\,dv \\ &= -\frac{E_{3D}(S)}{A}\frac{d}{dt}\int\int_S dS+\frac{1}{A}\int_0^1\int_0^1 \frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\Big[\mathbf{S}_t \end{aligned}$$

$$
\begin{aligned}
&- \left(\mathbf{S}_t \cdot \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \right) \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \Bigg] \cdot (\mathbf{S}_v \times \mathbf{S}_u)\, du\, dv \\
&+ \frac{1}{A} \int_0^1 \int_0^1 \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \cdot (\mathbf{S}_{vt} \times \mathbf{S}_u + \mathbf{S}_v \times \mathbf{S}_{ut})\, du\, dv
\end{aligned}
$$

$$
\begin{aligned}
= \; & -\frac{E_{3D}(S)}{A} \int_0^1 \int_0^1 \frac{d}{dt} \|\mathbf{S}_u \times \mathbf{S}_v\|\, du\, dv + \frac{1}{A} \int_0^1 \int_0^1 \frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \Bigg[\mathbf{S}_t \\
&- \left(\mathbf{S}_t \cdot \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \right) \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \Bigg] \cdot (\mathbf{S}_v \times \mathbf{S}_u)\, du\, dv \\
&+ \frac{1}{A} \int_0^1 \int_0^1 \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \cdot (\mathbf{S}_{vt} \times \mathbf{S}_u)\, du\, dv \\
&+ \frac{1}{A} \int_0^1 \int_0^1 \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \cdot (\mathbf{S}_v \times \mathbf{S}_{ut})\, du\, dv
\end{aligned}
$$

$$
\begin{aligned}
= \; & -\frac{E_{3D}(S)}{A} \int_0^1 \int_0^1 \frac{(\mathbf{S}_u \times \mathbf{S}_v) \cdot (\mathbf{S}_{ut} \times \mathbf{S}_v + \mathbf{S}_u \times \mathbf{S}_{vt})}{\|\mathbf{S}_u \times \mathbf{S}_v\|}\, du\, dv \\
&+ \frac{1}{A} \int_0^1 \int_0^1 \frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \Bigg[\mathbf{S}_t \\
&- \left(\mathbf{S}_t \cdot \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \right) \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \Bigg] \cdot (\mathbf{S}_v \times \mathbf{S}_u)\, du\, dv \\
&+ \frac{1}{A} \int_0^1 \int_0^1 \left(\mathbf{S}_u \times \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \right) \cdot \mathbf{S}_{tv}\, du\, dv \\
&- \frac{1}{A} \int_0^1 \int_0^1 \left(\mathbf{S}_v \times \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \right) \cdot \mathbf{S}_{tu}\, du\, dv. \qquad (56)
\end{aligned}
$$

Now using integration by parts in (56) and then simplifying we obtain

$$
\begin{aligned}
E'_{3D}(S) = \; & \frac{E_{3D}(S)}{A} \int_0^1 \int_0^1 \mathbf{S}_t \cdot \left(\mathbf{S}_v \times \frac{\mathbf{S}_u \times \mathbf{S}_v}{\|\mathbf{S}_u \times \mathbf{S}_v\|} \right)_u du\, dv \\
&+ \frac{E_{3D}(S)}{A} \int_0^1 \int_0^1 \mathbf{S}_t \cdot \left(\frac{\mathbf{S}_u \times \mathbf{S}_v}{\|\mathbf{S}_u \times \mathbf{S}_v\|} \times \mathbf{S}_u \right)_v du\, dv + \frac{1}{A} \int_0^1 \int_0^1 \frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \Bigg[\mathbf{S}_t \\
&- \left(\mathbf{S}_t \cdot \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \right) \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \Bigg] \cdot (\mathbf{S}_v \times \mathbf{S}_u)\, du\, dv \\
&- \frac{1}{A} \int_0^1 \int_0^1 \mathbf{S}_t \cdot \left[\mathbf{S}_{uv} \times \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} + \mathbf{S}_u \times \left(\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \right)_v \right] du\, dv \\
&+ \frac{1}{A} \int_0^1 \int_0^1 \mathbf{S}_t \cdot \left[\mathbf{S}_{vu} \times \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} + \mathbf{S}_v \times \left(\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \right)_u \right] du\, dv
\end{aligned}
$$

$$
= \frac{E_{3D}}{A} \int_0^1 \int_0^1 \mathbf{S}_t \cdot \left[\mathbf{S}_{vu} \times \frac{\mathbf{S}_u \times \mathbf{S}_v}{\|\mathbf{S}_u \times \mathbf{S}_v\|} + \mathbf{S}_v \times \left(\frac{\mathbf{S}_u \times \mathbf{S}_v}{\|\mathbf{S}_u \times \mathbf{S}_v\|} \right)_u \right] du\, dv
$$

$$
\begin{aligned}
&+\frac{E_{3D}}{A}\int_0^1\int_0^1 \mathbf{S}_t\cdot\left[\frac{\mathbf{S}_u\times\mathbf{S}_v}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\times\mathbf{S}_{uv}+\left(\frac{\mathbf{S}_u\times\mathbf{S}_v}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\right)_v\times\mathbf{S}_u\right]dudv\\
&+\frac{1}{A}\int_0^1\int_0^1 \frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\left[\mathbf{S}_t\right.\\
&\left.-\left(\mathbf{S}_t\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right]\cdot(\mathbf{S}_v\times\mathbf{S}_u)\,dudv\\
&-\frac{1}{A}\int_0^1\int_0^1 \mathbf{S}_t\cdot\left[\mathbf{S}_u\times\left(\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)_v\right]dudv\\
&+\frac{1}{A}\int_0^1\int_0^1 \mathbf{S}_t\cdot\left[\mathbf{S}_v\times\left(\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)_u\right]dudv
\end{aligned}
$$

$$
\begin{aligned}
=\quad &\frac{E_{3D}(S)}{A}\int_0^1\int_0^1 \mathbf{S}_t\cdot\left[\mathbf{S}_v\times\left(\frac{\mathbf{S}_u\times\mathbf{S}_v}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\right)_u-\mathbf{S}_u\times\left(\frac{\mathbf{S}_u\times\mathbf{S}_v}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\right)_v\right]dudv\\
&+\frac{1}{A}\int_0^1\int_0^1 \frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\left[\mathbf{S}_t\right.\\
&\left.-\left(\mathbf{S}_t\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right]\cdot(\mathbf{S}_v\times\mathbf{S}_u)\,dudv\\
&-\frac{1}{A}\int_0^1\int_0^1 \mathbf{S}_t\cdot\left\{\mathbf{S}_u\times\frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\left[\mathbf{S}_v\right.\right.\\
&\left.\left.-\left(\mathbf{S}_v\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right]\right\}dudv\\
&+\frac{1}{A}\int_0^1\int_0^1 \mathbf{S}_t\cdot\left\{\mathbf{S}_v\times\frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\left[\mathbf{S}_u\right.\right.\\
&\left.\left.-\left(\mathbf{S}_u\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right]\right\}dudv
\end{aligned}
$$

$$
\begin{aligned}
=\quad &\frac{E_{3D}(S)}{A}\int_0^1\int_0^1 \mathbf{S}_t\cdot\left\{\mathbf{S}_v\times\frac{1}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\left[(\mathbf{S}_u\times\mathbf{S}_v)_u\right.\right.\\
&\left.\left.-\left((\mathbf{S}_u\times\mathbf{S}_v)_u\cdot\frac{\mathbf{S}_u\times\mathbf{S}_v}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\right)\frac{\mathbf{S}_u\times\mathbf{S}_v}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\right]\right\}dudv\\
&-\frac{E_{3D}(S)}{A}\int_0^1\int_0^1 \mathbf{S}_t\cdot\left\{\mathbf{S}_u\times\frac{1}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\left[(\mathbf{S}_u\times\mathbf{S}_v)_v\right.\right.\\
&\left.\left.-\left((\mathbf{S}_u\times\mathbf{S}_v)_v\cdot\frac{\mathbf{S}_u\times\mathbf{S}_v}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\right)\frac{\mathbf{S}_u\times\mathbf{S}_v}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\right]\right\}dudv\\
&+\frac{1}{A}\int_0^1\int_0^1 \frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\mathbf{S}_t\cdot(\mathbf{S}_v\times\mathbf{S}_u)\,dudv\\
&-\frac{1}{A}\int_0^1\int_0^1 \frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\left(\mathbf{S}_t\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\cdot(\mathbf{S}_v\times\mathbf{S}_u)\,dudv\\
&-\frac{1}{A}\int_0^1\int_0^1 \frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\mathbf{S}_t\cdot(\mathbf{S}_u\times\mathbf{S}_v)\,dudv\\
&+\frac{1}{A}\int_0^1\int_0^1 \frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\mathbf{S}_t\cdot\left[\left(\mathbf{S}_v\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\mathbf{S}_u\times\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right]dudv
\end{aligned}
$$

$$+\frac{1}{A}\int_0^1\int_0^1 \frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\mathbf{S}_t\cdot(\mathbf{S}_v\times\mathbf{S}_u)\,du\,dv \tag{57}$$
$$-\frac{1}{A}\int_0^1\int_0^1 \frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\mathbf{S}_t\cdot\left[\left(\mathbf{S}_u\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\mathbf{S}_v\times\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right]du\,dv.$$

Taking into consideration that we can decompose the vectors $(\mathbf{S}_u\times\mathbf{S}_v)_u$ and $(\mathbf{S}_u\times\mathbf{S}_v)_v$ as a linear combination of the orthogonal set of unit vectors $\left\{\frac{\mathbf{S}_u}{\|\mathbf{S}_u\|},\frac{\mathbf{S}_v}{\|\mathbf{S}_v\|},\frac{\mathbf{S}_u\times\mathbf{S}_v}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\right\}$, it follows that:

$$\begin{aligned}
E'_{3D}(S) &= \frac{E_{3D}(S)}{A}\int_0^1\int_0^1\mathbf{S}_t\cdot\left\{\mathbf{S}_v\times\frac{1}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\left[\left((\mathbf{S}_u\times\mathbf{S}_v)_u\cdot\frac{\mathbf{S}_u}{\|\mathbf{S}_u\|}\right)\frac{\mathbf{S}_u}{\|\mathbf{S}_u\|}\right.\right.\\
&\left.\left.+\left((\mathbf{S}_u\times\mathbf{S}_v)_u\cdot\frac{\mathbf{S}_v}{\|\mathbf{S}_v\|}\right)\frac{\mathbf{S}_v}{\|\mathbf{S}_v\|}\right]\right\}du\,dv\\
&-\frac{E_{3D}(S)}{A}\int_0^1\int_0^1\mathbf{S}_t\cdot\left\{\mathbf{S}_u\times\frac{1}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\left[\left((\mathbf{S}_u\times\mathbf{S}_v)_v\cdot\frac{\mathbf{S}_u}{\|\mathbf{S}_u\|}\right)\frac{\mathbf{S}_u}{\|\mathbf{S}_u\|}\right.\right.\\
&\left.\left.+\left((\mathbf{S}_u\times\mathbf{S}_v)_v\cdot\frac{\mathbf{S}_v}{\|\mathbf{S}_v\|}\right)\frac{\mathbf{S}_v}{\|\mathbf{S}_v\|}\right]\right\}du\,dv\\
&+\frac{3}{A}\int_0^1\int_0^1\frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\mathbf{S}_t\cdot(\mathbf{S}_v\times\mathbf{S}_u)\,du\,dv\\
&-\frac{1}{A}\int_0^1\int_0^1\frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\mathbf{S}_t\cdot\left\{\left[(\mathbf{S}_v\times\mathbf{S}_u)\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right]\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right\}du\,dv\\
&+\frac{1}{A}\int_0^1\int_0^1\frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\mathbf{S}_t\cdot\left[\mathbf{S}_u\times\left(\mathbf{S}_v\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right]du\,dv\\
&-\frac{1}{A}\int_0^1\int_0^1\frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\mathbf{S}_t\cdot\left[\mathbf{S}_v\times\left(\mathbf{S}_u\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right]du\,dv
\end{aligned}$$

$$\begin{aligned}
&= \frac{E_{3D}(S)}{A}\int_0^1\int_0^1\mathbf{S}_t\cdot\left((\mathbf{S}_u\times\mathbf{S}_v)_u\cdot\frac{\mathbf{S}_u}{\|\mathbf{S}_u\|^2}\right)\frac{\mathbf{S}_v\times\mathbf{S}_u}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\,du\,dv\\
&+\frac{E_{3D}(S)}{A}\int_0^1\int_0^1\mathbf{S}_t\cdot\left((\mathbf{S}_u\times\mathbf{S}_v)_v\cdot\frac{\mathbf{S}_v}{\|\mathbf{S}_v\|^2}\right)\frac{\mathbf{S}_v\times\mathbf{S}_u}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\,du\,dv\\
&+\frac{1}{A}\int_0^1\int_0^1\left\langle\mathbf{S}_t,\frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right\{3\,(\mathbf{S}_v\times\mathbf{S}_u)\\
&-\left[(\mathbf{S}_v\times\mathbf{S}_u)\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right]\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\\
&+\mathbf{S}_u\times\left(\mathbf{S}_v\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\\
&-\mathbf{S}_v\times\left(\mathbf{S}_u\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\left\}\right\rangle du\,dv
\end{aligned}$$

$$
\begin{aligned}
= \quad & \frac{E_{3D}(S)}{A}\int_0^1\int_0^1 \mathbf{S}_t\cdot\left((\mathbf{S}_{uu}\times\mathbf{S}_v+\mathbf{S}_u\times\mathbf{S}_{vu})\cdot\frac{\mathbf{S}_u}{\|\mathbf{S}_u\|^2}\right)\frac{\mathbf{S}_v\times\mathbf{S}_u}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\,du\,dv \\
& +\frac{E_{3D}(S)}{A}\int_0^1\int_0^1 \mathbf{S}_t\cdot\left((\mathbf{S}_{uv}\times\mathbf{S}_v+\mathbf{S}_u\times\mathbf{S}_{vv})\cdot\frac{\mathbf{S}_v}{\|\mathbf{S}_v\|^2}\right)\frac{\mathbf{S}_v\times\mathbf{S}_u}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\,du\,dv \\
& +\frac{1}{A}\int_0^1\int_0^1 \left\langle \mathbf{S}_t, \frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\left\{3\left(\mathbf{S}_v\times\mathbf{S}_u\right)\right.\right. \\
& -\left[(\mathbf{S}_v\times\mathbf{S}_u)\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right]\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \\
& +\mathbf{S}_u\times\left(\mathbf{S}_v\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \\
& \left.\left.-\mathbf{S}_v\times\left(\mathbf{S}_u\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right\}\right\rangle du\,dv. \qquad (58)
\end{aligned}
$$

Since $(\mathbf{S}_u\times\mathbf{S}_{vu})\cdot\mathbf{S}_u=(\mathbf{S}_u\times\mathbf{S}_u)\cdot\mathbf{S}_{vu}=0$ and $(\mathbf{S}_{uv}\times\mathbf{S}_v)\cdot\mathbf{S}_v=-(\mathbf{S}_v\times\mathbf{S}_v)\cdot\mathbf{S}_{uv}=0$, we obtain

$$
\begin{aligned}
E'_{3D}(S) \quad = \quad & \frac{E_{3D}(S)}{A}\int_0^1\int_0^1 \mathbf{S}_t\cdot\left((\mathbf{S}_{uu}\times\mathbf{S}_v)\cdot\frac{\mathbf{S}_u}{\|\mathbf{S}_u\|^2}+(\mathbf{S}_u\times\mathbf{S}_{vv})\cdot\frac{\mathbf{S}_v}{\|\mathbf{S}_v\|^2}\right)\frac{\mathbf{S}_v\times\mathbf{S}_u}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\,du\,dv \\
& +\frac{1}{A}\int_0^1\int_0^1 \left\langle \mathbf{S}_t, \frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\left\{3\left(\mathbf{S}_v\times\mathbf{S}_u\right)\right.\right. \\
& -\left[(\mathbf{S}_v\times\mathbf{S}_u)\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right]\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \\
& +\mathbf{S}_u\times\left(\mathbf{S}_v\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \\
& \left.\left.-\mathbf{S}_v\times\left(\mathbf{S}_u\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right\}\right\rangle du\,dv
\end{aligned}
$$

$$
\begin{aligned}
= \quad & \frac{E_{3D}(S)}{A}\int_0^1\int_0^1 \mathbf{S}_t\cdot\left(\frac{\mathbf{S}_{uu}}{\|\mathbf{S}_u\|^2}\cdot(\mathbf{S}_v\times\mathbf{S}_u)+\frac{\mathbf{S}_{vv}}{\|\mathbf{S}_v\|^2}\cdot(\mathbf{S}_v\times\mathbf{S}_u)\right)\frac{\mathbf{S}_v\times\mathbf{S}_u}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\,du\,dv \\
& +\frac{1}{A}\int_0^1\int_0^1 \left\langle \mathbf{S}_t, \frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\left\{3\left(\mathbf{S}_v\times\mathbf{S}_u\right)\right.\right. \\
& -\left[(\mathbf{S}_v\times\mathbf{S}_u)\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right]\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \\
& +\mathbf{S}_u\times\left(\mathbf{S}_v\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \\
& \left.\left.-\mathbf{S}_v\times\left(\mathbf{S}_u\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right\}\right\rangle du\,dv. \qquad (59)
\end{aligned}
$$

Continuing we get

$$
\begin{aligned}
E'_{3D}(S) &= \frac{E_{3D}(S)}{A}\int_0^1\int_0^1 \mathbf{S}_t\cdot\left[\left(\frac{\mathbf{S}_{uu}}{\|\mathbf{S}_u\|^2}+\frac{\mathbf{S}_{vv}}{\|\mathbf{S}_v\|^2}\right)\cdot\frac{\mathbf{S}_v\times\mathbf{S}_u}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\right]\frac{\mathbf{S}_v\times\mathbf{S}_u}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\|\mathbf{S}_u\times\mathbf{S}_v\|\,du\,dv \\
&\quad +\frac{1}{A}\int_0^1\int_0^1\left\langle \mathbf{S}_t,\frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\left\{3\left(\mathbf{S}_v\times\mathbf{S}_u\right)\right.\right. \\
&\quad -\left[\left(\mathbf{S}_v\times\mathbf{S}_u\right)\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right]\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \\
&\quad +\mathbf{S}_u\times\left(\mathbf{S}_v\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \\
&\quad \left.\left.-\mathbf{S}_v\times\left(\mathbf{S}_u\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right\}\right\rangle du\,dv.
\end{aligned}
$$

$$
\begin{aligned}
&= \frac{E_{3D}(S)}{A}\int_0^1\int_0^1 \mathbf{S}_t\cdot\left(\kappa_u+\kappa_v\right)\frac{\mathbf{S}_v\times\mathbf{S}_u}{\|\mathbf{S}_u\times\mathbf{S}_v\|}\|\mathbf{S}_u\times\mathbf{S}_v\|\,du\,dv \\
&\quad +\frac{1}{A}\int_0^1\int_0^1\left\langle \mathbf{S}_t,\frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\left\{3\left(\mathbf{S}_v\times\mathbf{S}_u\right)\right.\right. \\
&\quad -\left[\left(\mathbf{S}_v\times\mathbf{S}_u\right)\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right]\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \\
&\quad +\mathbf{S}_u\times\left(\mathbf{S}_v\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \\
&\quad \left.\left.-\mathbf{S}_v\times\left(\mathbf{S}_u\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right\}\right\rangle du\,dv.
\end{aligned}
$$

$$
\begin{aligned}
&= \frac{1}{A}\int\int_S\langle\mathbf{S}_t,2H(\mathbf{S})E_{3D}(S)\mathbf{N}\rangle\,dS \\
&\quad +\frac{1}{A}\int_0^1\int_0^1\left\langle \mathbf{S}_t,\frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\left\{3\left(\mathbf{S}_v\times\mathbf{S}_u\right)\right.\right. \\
&\quad -\left[\left(\mathbf{S}_v\times\mathbf{S}_u\right)\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right]\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \\
&\quad +\mathbf{S}_u\times\left(\mathbf{S}_v\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} \\
&\quad \left.\left.-\mathbf{S}_v\times\left(\mathbf{S}_u\cdot\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right)\frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right\}\right\rangle du\,dv, \qquad (60)
\end{aligned}
$$

where $H(\mathbf{S})$ is the mean curvature at $\mathbf{S}$.

In order to further simplify the last two lines in (60) we need to do an additional analysis. Let $\mathbf{a},\mathbf{b}\in\mathbb{R}^3$ be unit vectors such that the set $\{\mathbf{a},\mathbf{b},\mathbf{c}\}$ form an orthogonal basis with

$\mathbf{c} = \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}$. We can then express the the vectors $\mathbf{S}_u$, $\mathbf{S}_v$ as

$$\mathbf{S}_u = (\mathbf{S}_u \cdot \mathbf{a})\,\mathbf{a} + (\mathbf{S}_u \cdot \mathbf{b})\,\mathbf{b} + (\mathbf{S}_u \cdot \mathbf{c})\,\mathbf{c} \tag{61}$$

and

$$\mathbf{S}_v = (\mathbf{S}_v \cdot \mathbf{a})\,\mathbf{a} + (\mathbf{S}_v \cdot \mathbf{b})\,\mathbf{b} + (\mathbf{S}_v \cdot \mathbf{c})\,\mathbf{c}. \tag{62}$$

Now, let us choose $\mathbf{a}$ and $\mathbf{b}$ such that $\mathbf{a} \times \mathbf{b} = \mathbf{c}$, then $\mathbf{b} \times \mathbf{c} = \mathbf{a}$ and $\mathbf{c} \times \mathbf{a} = \mathbf{b}$. Using these results we can express the terms in (60) as functions of $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$ as follows:

$$\begin{aligned}
\mathbf{S}_v \times \mathbf{S}_u &= [(\mathbf{S}_v \cdot \mathbf{a})\,\mathbf{a} + (\mathbf{S}_v \cdot \mathbf{b})\,\mathbf{b} + (\mathbf{S}_v \cdot \mathbf{c})\,\mathbf{c}] \times [(\mathbf{S}_u \cdot \mathbf{a})\,\mathbf{a} + (\mathbf{S}_u \cdot \mathbf{b})\,\mathbf{b} + (\mathbf{S}_u \cdot \mathbf{c})\,\mathbf{c}] \\
&= [(\mathbf{S}_v \cdot \mathbf{b})\,(\mathbf{S}_u \cdot \mathbf{c}) - (\mathbf{S}_v \cdot \mathbf{c})\,(\mathbf{S}_u \cdot \mathbf{b})]\,\mathbf{a} + [(\mathbf{S}_v \cdot \mathbf{c})\,(\mathbf{S}_u \cdot \mathbf{a}) - (\mathbf{S}_v \cdot \mathbf{a})\,(\mathbf{S}_u \cdot \mathbf{c})]\,\mathbf{b} \\
&\quad + [(\mathbf{S}_v \cdot \mathbf{a})\,(\mathbf{S}_u \cdot \mathbf{b}) - (\mathbf{S}_v \cdot \mathbf{b})\,(\mathbf{S}_u \cdot \mathbf{a})]\,\mathbf{c},
\end{aligned} \tag{63}$$

$$\begin{aligned}
\left[(\mathbf{S}_v \times \mathbf{S}_u) \cdot \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right] \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} &= ((\mathbf{S}_v \times \mathbf{S}_u) \cdot \mathbf{c})\,\mathbf{c} \\
&= [(\mathbf{S}_v \cdot \mathbf{a})\,(\mathbf{S}_u \cdot \mathbf{b}) - (\mathbf{S}_v \cdot \mathbf{b})\,(\mathbf{S}_u \cdot \mathbf{a})]\,\mathbf{c},
\end{aligned} \tag{64}$$

$$\begin{aligned}
\mathbf{S}_u \times \left(\mathbf{S}_v \cdot \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right) \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} &= \mathbf{S}_u \times (\mathbf{S}_v \cdot \mathbf{c})\,\mathbf{c} \\
&= [(\mathbf{S}_u \cdot \mathbf{a})\,\mathbf{a} + (\mathbf{S}_u \cdot \mathbf{b})\,\mathbf{b} + (\mathbf{S}_u \cdot \mathbf{c})\,\mathbf{c}] \times (\mathbf{S}_v \cdot \mathbf{c})\,\mathbf{c} \\
&= (\mathbf{S}_u \cdot \mathbf{b})\,(\mathbf{S}_v \cdot \mathbf{c})\,\mathbf{a} - (\mathbf{S}_u \cdot \mathbf{a})\,(\mathbf{S}_v \cdot \mathbf{c})\,\mathbf{b},
\end{aligned} \tag{65}$$

and

$$\begin{aligned}
\mathbf{S}_v \times \left(\mathbf{S}_u \cdot \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right) \frac{\mathbf{S}(u,v)-\mathbf{P}}{\|\mathbf{S}(u,v)-\mathbf{P}\|} &= \mathbf{S}_v \times (\mathbf{S}_u \cdot \mathbf{c})\,\mathbf{c} \\
&= [(\mathbf{S}_v \cdot \mathbf{a})\,\mathbf{a} + (\mathbf{S}_v \cdot \mathbf{b})\,\mathbf{b} + (\mathbf{S}_v \cdot \mathbf{c})\,\mathbf{c}] \times (\mathbf{S}_u \cdot \mathbf{c})\,\mathbf{c} \\
&= (\mathbf{S}_v \cdot \mathbf{b})\,(\mathbf{S}_u \cdot \mathbf{c})\,\mathbf{a} - (\mathbf{S}_v \cdot \mathbf{a})\,(\mathbf{S}_u \cdot \mathbf{c})\,\mathbf{b}.
\end{aligned} \tag{66}$$

Now, we can simplify (60) as shown below

$$
\begin{aligned}
E'_{3D}(S) &= \frac{1}{A}\int\int_S \langle \mathbf{S}_t, 2H(\mathbf{S})E_{3D}(S)\mathbf{N}\rangle\, dS \\
&\quad +\frac{1}{A}\int_0^1\int_0^1 \left\langle \mathbf{S}_t, \frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\{3[(\mathbf{S}_v\cdot\mathbf{b})(\mathbf{S}_u\cdot\mathbf{c})-(\mathbf{S}_v\cdot\mathbf{c})(\mathbf{S}_u\cdot\mathbf{b})]\mathbf{a}\right. \\
&\quad +3[(\mathbf{S}_v\cdot\mathbf{c})(\mathbf{S}_u\cdot\mathbf{a})-(\mathbf{S}_v\cdot\mathbf{a})(\mathbf{S}_u\cdot\mathbf{c})]\mathbf{b}+3[(\mathbf{S}_v\cdot\mathbf{a})(\mathbf{S}_u\cdot\mathbf{b})-(\mathbf{S}_v\cdot\mathbf{b})(\mathbf{S}_u\cdot\mathbf{a})]\mathbf{c} \\
&\quad -[(\mathbf{S}_v\cdot\mathbf{a})(\mathbf{S}_u\cdot\mathbf{b})-(\mathbf{S}_v\cdot\mathbf{b})(\mathbf{S}_u\cdot\mathbf{a})]\mathbf{c}+(\mathbf{S}_u\cdot\mathbf{b})(\mathbf{S}_v\cdot\mathbf{c})\mathbf{a}-(\mathbf{S}_u\cdot\mathbf{a})(\mathbf{S}_v\cdot\mathbf{c})\mathbf{b} \\
&\quad \left.-(\mathbf{S}_v\cdot\mathbf{b})(\mathbf{S}_u\cdot\mathbf{c})\mathbf{a}+(\mathbf{S}_v\cdot\mathbf{a})(\mathbf{S}_u\cdot\mathbf{c})\mathbf{b}\}\right\rangle\, du\,dv \\
&= \frac{1}{A}\int\int_S \langle \mathbf{S}_t, 2H(\mathbf{S})E_{3D}(S)\mathbf{N}\rangle\, dS \\
&\quad +\frac{1}{A}\int_0^1\int_0^1 \left\langle \mathbf{S}_t, \frac{1}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\{2[(\mathbf{S}_v\cdot\mathbf{b})(\mathbf{S}_u\cdot\mathbf{c})-(\mathbf{S}_v\cdot\mathbf{c})(\mathbf{S}_u\cdot\mathbf{b})]\mathbf{a}\right. \\
&\quad +2[(\mathbf{S}_v\cdot\mathbf{c})(\mathbf{S}_u\cdot\mathbf{a})-(\mathbf{S}_v\cdot\mathbf{a})(\mathbf{S}_u\cdot\mathbf{c})]\mathbf{b} \\
&\quad \left.+2[(\mathbf{S}_v\cdot\mathbf{a})(\mathbf{S}_u\cdot\mathbf{b})-(\mathbf{S}_v\cdot\mathbf{b})(\mathbf{S}_u\cdot\mathbf{a})]\mathbf{c}\}\right\rangle\, du\,dv \\
&= \frac{1}{A}\int\int_S \langle \mathbf{S}_t, 2H(\mathbf{S})E_{3D}(S)\mathbf{N}\rangle\, dS+\frac{1}{A_{S_v}}\int_0^1\int_0^1 \left\langle \mathbf{S}_t, 2\frac{\mathbf{S}_v\times\mathbf{S}_u}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\right\rangle du\,dv \\
&= \frac{1}{A}\int\int_S \langle \mathbf{S}_t, 2H(\mathbf{S})E_{3D}(S)\mathbf{N}\rangle\, dS \\
&\quad +\frac{1}{A}\int_0^1\int_0^1 \left\langle \mathbf{S}_t, \frac{2}{\|\mathbf{S}(u,v)-\mathbf{P}\|}\frac{\mathbf{S}_v\times\mathbf{S}_u}{\|\mathbf{S}_v\times\mathbf{S}_u\|}\right\rangle \|\mathbf{S}_v\times\mathbf{S}_u\|\, du\,dv \\
&= \frac{1}{A}\int\int_S \langle \mathbf{S}_t, 2H(\mathbf{S})E_{3D}(S)\mathbf{N}\rangle\, dS+\frac{1}{A_{S_v}}\int\int_{S_v}\left\langle \mathbf{S}_t, \frac{2}{\|\mathbf{S}-\mathbf{P}\|}\mathbf{N}\right\rangle dS \\
&= \int\int_S \left\langle \mathbf{S}_t, \frac{2}{A}\left(H(\mathbf{S})E_{3D}(S)+\frac{1}{\|\mathbf{S}-\mathbf{P}\|}\right)\mathbf{N}\right\rangle dS. \qquad (67)
\end{aligned}
$$

Therefore, a gradient flow that maximizes the average visibility of the surface S with respect to the viewpoint $\mathbf{P}$ is

$$
\mathbf{S}_t = \frac{1}{A}\left(H(\mathbf{S})E_{3D}(S)+\frac{1}{\|\mathbf{S}-\mathbf{P}\|}\right)\mathbf{N}. \qquad (68)
$$

Since $\mathbf{P}$ is located outside S, then $\|\mathbf{S}-\mathbf{P}\|$ is nonzero as long as the evolving contour does not touch $\mathbf{P}$. Moreover, this gradient flow is stable if the initial value of the average visibility, E_{3D}, is positive.

B.4 Visibility of a Triangular Face

Let $\mathbf{v}_a$, $\mathbf{v}_b$, and $\mathbf{v}_c$ be the vertexes of the triangular face S_i of area A_i in a given triangulated surface. Let also assume that the vertexes are ordered counterclockwise so that the inward unit normal of S_i could be computed without ambiguity (see Fig. 5.22). It follows that the inward unit normal at every point in S_i can be computed by

$$\begin{aligned} \mathbf{N}_i &= -\frac{(\mathbf{v}_b-\mathbf{v}_a)\times(\mathbf{v}_c-\mathbf{v}_a)}{\|(\mathbf{v}_b-\mathbf{v}_a)\times(\mathbf{v}_c-\mathbf{v}_a)\|} \\ &= -\frac{\mathbf{v}_a\times\mathbf{v}_b+\mathbf{v}_b\times\mathbf{v}_c+\mathbf{v}_c\times\mathbf{v}_a}{\|\mathbf{v}_a\times\mathbf{v}_b+\mathbf{v}_b\times\mathbf{v}_c+\mathbf{v}_c\times\mathbf{v}_a\|}, \end{aligned} \tag{69}$$

where $\times$ is the cross product operator.

The total visibility $E_{3D,i}$ of the triangular face S_i with respect to the viewpoint $\mathbf{P}$ can be written as the surface integral

$$E_{3D,i} = \int\int_{S_i} \frac{(\mathbf{C}(s)-\mathbf{P})}{\|\mathbf{C}(s)-\mathbf{P}\|}\cdot\mathbf{N}_i\, dS. \tag{70}$$

We can parameterize the triangular face S_i as

$$S_i(u,v) = \mathbf{v}_a + u(\mathbf{v}_b-\mathbf{v}_a) + v(\mathbf{v}_c-\mathbf{v}_a), \tag{71}$$

for $u\in[0,1]$ and $v\in[0,1-u]$. Using this parameterization, (70) becomes

$$\begin{aligned} E_{3D,i} &= \int_0^1\int_0^{1-u} \frac{\mathbf{v}_a+u(\mathbf{v}_b-\mathbf{v}_a)+v(\mathbf{v}_c-\mathbf{v}_a)-\mathbf{P}}{\|\mathbf{v}_a+u(\mathbf{v}_b-\mathbf{v}_a)+v(\mathbf{v}_c-\mathbf{v}_a)-\mathbf{P}\|}\cdot\mathbf{N}_i \left\|\frac{\partial S_i}{\partial u}\times\frac{\partial S_i}{\partial v}\right\| dv\,du \\ &= \int_0^1\int_0^{1-u} \frac{\mathbf{v}_a-\mathbf{P}}{\|\mathbf{v}_a+u(\mathbf{v}_b-\mathbf{v}_a)+v(\mathbf{v}_c-\mathbf{v}_a)-\mathbf{P}\|}\cdot\mathbf{N}_i\,\|(\mathbf{v}_b-\mathbf{v}_a)\times(\mathbf{v}_c-\mathbf{v}_a)\|\, dv\,du \qquad (72) \\ &= -\int_0^1\int_0^{1-u} \frac{(\mathbf{v}_a-\mathbf{P})\cdot((\mathbf{v}_b-\mathbf{v}_a)\times(\mathbf{v}_c-\mathbf{v}_a))}{\sqrt{l_{ac}^2v^2+2(\mathbf{v}_c-\mathbf{v}_a)\cdot(\mathbf{v}_a+u(\mathbf{v}_b-\mathbf{v}_a)-\mathbf{P})v+\|\mathbf{v}_a+u(\mathbf{v}_b-\mathbf{v}_a)-\mathbf{P}\|^2}}\, dv\,du, \end{aligned}$$

where $l_{ac} = \|\mathbf{v}_a - \mathbf{v}_c\|$. Continuing we get

$$\begin{aligned} E_{3D,i} &= -\frac{(\mathbf{v}_a-\mathbf{P})\cdot((\mathbf{v}_b-\mathbf{v}_a)\times(\mathbf{v}_c-\mathbf{v}_a))}{l_{ac}} \int_0^1 \left[\ln\left(l_{ac}v + \frac{(\mathbf{v}_c-\mathbf{v}_a)\cdot(\mathbf{v}_a+u(\mathbf{v}_b-\mathbf{v}_a)-\mathbf{P})}{l_{ac}}\right.\right. \\ &\quad \left.\left. + \|\mathbf{v}_a+u(\mathbf{v}_b-\mathbf{v}_a)+v(\mathbf{v}_c-\mathbf{v}_a)-\mathbf{P}\|\right)\right]_0^{1-u} \\ &= \frac{\alpha}{l_{ac}}\int_0^1 \ln\left(l_{ac}(1-u)+\frac{\beta}{l_{ac}}+\gamma\right) - \ln\left(\frac{\beta}{l_{ac}}+\delta\right) du \\ &= \frac{\alpha}{l_{ac}}\int_0^1 \ln\left(\frac{l_{ac}^2(1-u)+\beta+\gamma l_{ac}}{\beta+\delta\, l_{ac}}\right) du, \end{aligned} \tag{73}$$

for $\alpha = -(\mathbf{v}_a-\mathbf{P})\cdot(\mathbf{v}_a\times\mathbf{v}_b+\mathbf{v}_b\times\mathbf{v}_c+\mathbf{v}_c\times\mathbf{v}_a)$, $\beta = (\mathbf{v}_c-\mathbf{v}_a)\cdot((1-u)\mathbf{v}_a+u\,\mathbf{v}_b-\mathbf{P})$, $\gamma = \|u\,\mathbf{v}_b+(1-u)\mathbf{v}_c-\mathbf{P}\|$, and $\delta = \|(1-u)\mathbf{v}_a+u\,\mathbf{v}_b-\mathbf{P}\|$.

Taking the derivative of (73) with respect to the vertex $\mathbf{v}_a$ we get

$$\begin{aligned} \frac{\partial E_{3D,i}}{\partial \mathbf{v}_a} &= \frac{\partial}{\partial \mathbf{v}_a}\left[\frac{\alpha}{l_{ac}}\int_0^1 \ln\left(\frac{l_{ac}^2(1-u)+\beta+\gamma l_{ac}}{\beta+\delta\, l_{ac}}\right) du\right] \\ &= \int_0^1 \ln\left(\frac{l_{ac}^2(1-u)+\beta+\gamma l_{ac}}{\beta+\delta\, l_{ac}}\right) du \frac{\partial}{\partial \mathbf{v}_a}\left[\frac{\alpha}{l_{ac}}\right] + \frac{\alpha}{l_{ac}}\frac{\partial}{\partial \mathbf{v}_a}\int_0^1 \ln\left(\frac{l_{ac}^2(1-u)+\beta+\gamma l_{ac}}{\beta+\delta\, l_{ac}}\right) du \\ &= \int_0^1 \ln\left(\frac{l_{ac}^2(1-u)+\beta+\gamma l_{ac}}{\beta+\delta\, l_{ac}}\right) du \left(\frac{1}{l_{ac}}\frac{\partial \alpha}{\partial \mathbf{v}_a} - \frac{\alpha}{l_{ac}^3}(\mathbf{v}_a-\mathbf{v}_c)\right) \\ &\quad + \frac{\alpha}{l_{ac}}\int_0^1 \left(\frac{\beta+\delta\, l_{ac}}{l_{ac}^2(1-u)+\beta+\gamma l_{ac}}\right)\frac{\partial}{\partial \mathbf{v}_a}\left[\frac{l_{ac}^2(1-u)+\beta+\gamma l_{ac}}{\beta+\delta\, l_{ac}}\right] du \\ &= \frac{1}{l_{ac}}\int_0^1 \ln\left(\frac{l_{ac}^2(1-u)+\beta+\gamma l_{ac}}{\beta+\delta\, l_{ac}}\right) du \left(\frac{\partial \alpha}{\partial \mathbf{v}_a} - \frac{\alpha}{l_{ac}^2}(\mathbf{v}_a-\mathbf{v}_c)\right) \\ &\quad + \frac{\alpha}{l_{ac}}\int_0^1 \left(\frac{\beta+\delta\, l_{ac}}{l_{ac}^2(1-u)+\beta+\gamma l_{ac}}\right)\left(\frac{\left(2(1-u)+\frac{\gamma}{l_{ac}}\right)(\mathbf{v}_a-\mathbf{v}_c)+\frac{\partial \beta}{\partial \mathbf{v}_a}}{\beta+\delta\, l_{ac}}\right. \\ &\quad \left. - \frac{l_{ac}^2(1-u)+\beta+\gamma l_{ac}}{(\beta+\delta\, l_{ac})^2}\left(\frac{\partial \beta}{\partial \mathbf{v}_a} + l_{ac}\frac{\partial \delta}{\partial \mathbf{v}_a} + \frac{\delta}{l_{ac}}(\mathbf{v}_a-\mathbf{v}_c)\right)\right) du \\ &= \frac{1}{l_{ac}}\int_0^1 \ln\left(\frac{l_{ac}^2(1-u)+\beta+\gamma l_{ac}}{\beta+\delta\, l_{ac}}\right) du \left(\frac{\partial \alpha}{\partial \mathbf{v}_a} - \frac{\alpha}{l_{ac}^2}(\mathbf{v}_a-\mathbf{v}_c)\right) \end{aligned}$$

$$+\frac{\alpha}{l_{ac}}\int_0^1\left(\frac{\left(2(1-u)+\frac{\gamma}{l_{ac}}\right)(\mathbf{v}_a-\mathbf{v}_c)+\frac{\partial\beta}{\partial\mathbf{v}_a}}{l_{ac}^2(1-u)+\beta+\gamma l_{ac}}-\frac{\frac{\partial\beta}{\partial\mathbf{v}_a}+l_{ac}\frac{\partial\delta}{\partial\mathbf{v}_a}+\frac{\delta}{l_{ac}}(\mathbf{v}_a-\mathbf{v}_c)}{\beta+\delta\, l_{ac}}\right)du, \tag{74}$$

where the partial derivatives of α, β, and δ with respect to $\mathbf{v}_a$ can be easily obtained as follows:

$$\begin{aligned}\frac{\partial\alpha}{\partial\mathbf{v}_a} &= \frac{\partial}{\partial\mathbf{v}_a}\left[-(\mathbf{v}_a-\mathbf{P})\cdot(\mathbf{v}_a\times\mathbf{v}_b+\mathbf{v}_b\times\mathbf{v}_c+\mathbf{v}_c\times\mathbf{v}_a)\right]\\ &= -\left(((\mathbf{v}_b-\mathbf{v}_a)\times(\mathbf{v}_c-\mathbf{v}_a))+\frac{\partial}{\partial\mathbf{v}_a}\left[(\mathbf{v}_c-\mathbf{v}_b)\times\mathbf{v}_a\right]^T(\mathbf{v}_a-\mathbf{P})\right)\end{aligned} \tag{75}$$

$$\begin{aligned}\frac{\partial\beta}{\partial\mathbf{v}_a} &= \frac{\partial}{\partial\mathbf{v}_a}\left[(\mathbf{v}_c-\mathbf{v}_a)\cdot((1-u)\mathbf{v}_a+u\,\mathbf{v}_b-\mathbf{P})\right]\\ &= (1-u)(\mathbf{v}_c-\mathbf{v}_a)-((1-u)\mathbf{v}_a+u\,\mathbf{v}_b-\mathbf{P})\\ &= -(2(1-u)\mathbf{v}_a+u\,\mathbf{v}_b-(1-u)\mathbf{v}_c-\mathbf{P})\end{aligned} \tag{76}$$

$$\begin{aligned}\frac{\partial\delta}{\partial\mathbf{v}_a} &= \frac{\partial}{\partial\mathbf{v}_a}\|(1-u)\mathbf{v}_a+u\,\mathbf{v}_b-\mathbf{P}\|\\ &= \frac{(1-u)}{\delta}((1-u)\mathbf{v}_a+u\,\mathbf{v}_b-\mathbf{P}).\end{aligned} \tag{77}$$

B.5 3D Topology Preservation

Non-Adjacent Triangular Faces

Let S_i and S_j be two non-adjacent triangular faces of a triangulated surface with set of vertexes $\{\mathbf{v}_a, \mathbf{v}_b, \mathbf{v}_c\}$ and $\{\mathbf{v}_{\hat{a}}, \mathbf{v}_{\hat{b}}, \mathbf{v}_{\hat{c}}\}$, respectively. The minimization of the energy

$$E_{3D,R_{i,j}} = \int\int_{S_i \times S_j} \frac{1}{\|\mathbf{S}_i - \mathbf{S}_j\|^2} dS_i dS_j \tag{78}$$

would avoid intersections between the two non-adjacent triangular faces, therefore preventing topology changes during the evolution of the surface

If we use the parameterizations

$$\mathbf{S}(u,v) = \mathbf{v}_a + u(\mathbf{v}_b - \mathbf{v}_a) + v(\mathbf{v}_c - \mathbf{v}_a) \tag{79}$$

$$\mathbf{S}(\hat{u},\hat{v}) = \mathbf{v}_{\hat{a}} + \hat{u}\left(\mathbf{v}_{\hat{b}} - \mathbf{v}_{\hat{a}}\right) + \hat{v}(\mathbf{v}_{\hat{c}} - \mathbf{v}_{\hat{a}}) \tag{80}$$

where $u \in [0,1]$, $v \in [0, 1-u]$ and $\hat{u} \in [0,1]$, $\hat{v} \in [0, 1-u]$, then (78) becomes

$$\begin{aligned} E_{3D,R_{i,j}} &= \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} \frac{\|\mathbf{S}_u \times \mathbf{S}_v\| \, \|\mathbf{S}_{\hat{u}} \times \mathbf{S}_{\hat{v}}\|}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^2} d\hat{v}\, d\hat{u}\, dv\, du \\ &= \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} \frac{\|(\mathbf{v}_b - \mathbf{v}_a) \times (\mathbf{v}_c - \mathbf{v}_a)\| \, \left\|\left(\mathbf{v}_{\hat{b}} - \mathbf{v}_{\hat{a}}\right) \times (\mathbf{v}_{\hat{c}} - \mathbf{v}_{\hat{a}})\right\|}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^2} d\hat{v}\, d\hat{u}\, dv\, du \\ &= 4A_i A_j \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} \frac{1}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^2} d\hat{v}\, d\hat{u}\, dv\, du. \end{aligned} \tag{81}$$

Now taking the derivative with respect to $\mathbf{v}_a$ of the expression above we get

$$\begin{aligned} \frac{\partial E_{3D,R_{i,j}}}{\partial \mathbf{v}_a} &= \frac{\partial}{\partial \mathbf{v}_a}\left[4A_i A_j \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} \frac{1}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^2} d\hat{v}\, d\hat{u}\, dv\, du\right] \\ &= 2A_j \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} \frac{d\hat{v}\, d\hat{u}\, dv\, du}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^2} \frac{\partial}{\partial \mathbf{v}_a} \|(\mathbf{v}_b - \mathbf{v}_a) \times (\mathbf{v}_c - \mathbf{v}_a)\| \\ &\quad + 4A_i A_j \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} \frac{\partial}{\partial \mathbf{v}_a} \frac{d\hat{v}\, d\hat{u}\, dv\, du}{\|(1-u-v)\mathbf{v}_a + u\mathbf{v}_b + v\mathbf{v}_c - \mathbf{S}(\hat{u},\hat{v})\|^2} \\ &= -2A_j \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} \frac{d\hat{v}\, d\hat{u}\, dv\, du}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^2} \frac{\partial}{\partial \mathbf{v}_a} [(\mathbf{v}_c - \mathbf{v}_b) \times \mathbf{v}_a]^T \mathbf{N}_i \end{aligned}$$

$$-8A_iA_j\int_0^1\int_0^{1-u}\int_0^1\int_0^{1-\hat{u}}(1-u-v)\frac{\mathbf{S}(u,v)-\mathbf{S}(\hat{u},\hat{v})}{\|\mathbf{S}(u,v)-\mathbf{S}(\hat{u},\hat{v})\|^4}d\hat{v}\,d\hat{u}\,dv\,du, \quad (82)$$

where $\mathbf{N}_i$ is the inward unit normal of the triangular face S_i.

Non-Adjacent Triangular Faces (Approximation)

When two non-adjacent triangular faces are far apart we can approximate (81) to reduce the number of computations. One way to do this is by using the centroids of each triangular face in (78). That is,

$$\begin{aligned} E_{3D,R_{i,j}} &= 4A_iA_j\int_0^1\int_0^{1-u}\int_0^1\int_0^{1-\hat{u}}\frac{1}{\left\|\bar{\mathbf{S}}_i-\bar{\mathbf{S}}_j\right\|^2}d\hat{v}\,d\hat{u}\,dv\,du \\ &= \frac{4A_iA_j}{\left\|\bar{\mathbf{S}}_i-\bar{\mathbf{S}}_j\right\|^2}\int_0^1\int_0^{1-u}dv\,du\int_0^1\int_0^{1-\hat{u}}d\hat{v}\,d\hat{u} \\ &= \frac{A_iA_j}{\left\|\bar{\mathbf{S}}_i-\bar{\mathbf{S}}_j\right\|^2}. \end{aligned} \quad (83)$$

Therefore we have

$$\begin{aligned} \frac{\partial E_{3D,R_{i,j}}}{\partial \mathbf{v}_a} &= \frac{\partial}{\partial \mathbf{v}_a}\frac{A_iA_j}{\left\|\bar{\mathbf{S}}_i-\bar{\mathbf{S}}_j\right\|^2} \\ &= \frac{1}{2}\frac{A_j}{\left\|\bar{\mathbf{S}}_i-\bar{\mathbf{S}}_j\right\|^2}\frac{\partial}{\partial \mathbf{v}_a}\left\|(\mathbf{v}_b-\mathbf{v}_a)\times(\mathbf{v}_c-\mathbf{v}_a)\right\| \\ &\quad -\frac{2}{3}A_iA_j\frac{\bar{\mathbf{S}}_i-\bar{\mathbf{S}}_j}{\left\|\bar{\mathbf{S}}_i-\bar{\mathbf{S}}_j\right\|^4} \\ &= -\frac{1}{2}A_j\frac{1}{\left\|(\mathbf{v}_a+\mathbf{v}_b+\mathbf{v}_c)-\left(\mathbf{v}_{\hat{a}}+\mathbf{v}_{\hat{b}}+\mathbf{v}_{\hat{c}}\right)\right\|^2}\frac{\partial}{\partial \mathbf{v}_a}\left[(\mathbf{v}_c-\mathbf{v}_b)\times\mathbf{v}_a\right]^T\mathbf{N}_i \\ &\quad -\frac{2}{3}A_iA_j\frac{\bar{\mathbf{S}}_i-\bar{\mathbf{S}}_j}{\left\|\bar{\mathbf{S}}_i-\bar{\mathbf{S}}_j\right\|^4}. \end{aligned} \quad (84)$$

Adjacent Triangular Faces

Let S_i and S_j be two adjacent triangular faces with set of vertexes $\{\mathbf{v}_a,\mathbf{v}_b,\mathbf{v}_c\}$ and $\left\{\mathbf{v}_a,\mathbf{v}_{\hat{b}},\mathbf{v}_{\hat{c}}\right\}$, respectively. Thus S_i and S_j have $\mathbf{v}_a$ as a common vertex. If we use the parameterizations

$$\mathbf{S}(u,v) = \mathbf{v}_a+u(\mathbf{v}_b-\mathbf{v}_a)+v(\mathbf{v}_c-\mathbf{v}_a) \quad (85)$$

$$\mathbf{S}(\hat{u},\hat{v}) \;=\; \mathbf{v}_a + \hat{u}\left(\mathbf{v}_{\hat{b}} - \mathbf{v}_a\right) + \hat{v}\left(\mathbf{v}_{\hat{c}} - \mathbf{v}_a\right), \tag{86}$$

where $u \in [0,1]$, $v \in [0,1-u]$ and $\hat{u} \in [0,1]$, $\hat{v} \in [0,1-u]$, then (78) becomes

$$\begin{aligned}
E_{3D,R_{i,j}} &= \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} \frac{\|\mathbf{S}_u \times \mathbf{S}_v\| \, \|\mathbf{S}_{\hat{u}} \times \mathbf{S}_{\hat{v}}\|}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^2} d\hat{v}\, d\hat{u}\, dv\, du \\
&= \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} \frac{\|(\mathbf{v}_b - \mathbf{v}_a) \times (\mathbf{v}_c - \mathbf{v}_a)\| \, \left\|\left(\mathbf{v}_{\hat{b}} - \mathbf{v}_a\right) \times \left(\mathbf{v}_{\hat{c}} - \mathbf{v}_a\right)\right\|}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^2} d\hat{v}\, d\hat{u}\, dv\, du \\
&= 4A_iA_j \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} \frac{1}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^2} d\hat{v}\, d\hat{u}\, dv\, du.
\end{aligned} \tag{87}$$

Now taking the derivative with respect to $\mathbf{v}_a$ of the expression above we get

$$\begin{aligned}
\frac{\partial E_{3D,R_{i,j}}}{\partial \mathbf{v}_a} &= \frac{\partial}{\partial \mathbf{v}_a}\left[4A_iA_j \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} \frac{1}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^2} d\hat{v}\, d\hat{u}\, dv\, du\right] \\
&= 2A_j \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} \frac{d\hat{v}\, d\hat{u}\, dv\, du}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^2} \frac{\partial}{\partial \mathbf{v}_a} \|(\mathbf{v}_b - \mathbf{v}_a) \times (\mathbf{v}_c - \mathbf{v}_a)\| \\
&\quad + 2A_i \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} \frac{d\hat{v}\, d\hat{u}\, dv\, du}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^2} \frac{\partial}{\partial \mathbf{v}_a} \left\|\left(\mathbf{v}_{\hat{b}} - \mathbf{v}_a\right) \times \left(\mathbf{v}_{\hat{c}} - \mathbf{v}_a\right)\right\| \\
&\quad + 4A_iA_j \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} \frac{\partial}{\partial \mathbf{v}_a} \frac{d\hat{v}\, d\hat{u}\, dv\, du}{\|(\hat{u}+\hat{v}-u-v)\mathbf{v}_a + u\mathbf{v}_b + v\mathbf{v}_c - \hat{u}\mathbf{v}_b - \hat{v}\mathbf{v}_c\|^2} \\
&= -2A_j \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} \frac{d\hat{v}\, d\hat{u}\, dv\, du}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^2} \frac{\partial}{\partial \mathbf{v}_a} \left[(\mathbf{v}_c - \mathbf{v}_b) \times \mathbf{v}_a\right]^T \mathbf{N}_i \\
&\quad - 2A_i \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} \frac{d\hat{v}\, d\hat{u}\, dv\, du}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^2} \frac{\partial}{\partial \mathbf{v}_a} \left[\left(\mathbf{v}_{\hat{c}} - \mathbf{v}_{\hat{b}}\right) \times \mathbf{v}_a\right]^T \mathbf{N}_j \\
&\quad - 8A_iA_j \int_0^1 \int_0^{1-u} \int_0^1 \int_0^{1-\hat{u}} (\hat{u}+\hat{v}-u-v) \frac{\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})}{\|\mathbf{S}(u,v) - \mathbf{S}(\hat{u},\hat{v})\|^4} d\hat{v}\, d\hat{u}\, dv\, du,
\end{aligned} \tag{88}$$

where $\mathbf{N}_j$ is the inward unit normal of the triangular face S_j.

REFERENCES

[1] F. Durand, 3D Visibility: Analysis Study and Applications. Ph.D. Thesis, University J. Fourier, Grenoble, France, July 1999.

[2] A. Woo, P. Poulin, and A. Fournier, "A survey of shadow algorithms," *IEEE Computer Graphics and Applications*, vol. 10, no. 6, pp. 13-32, November 1990.

[3] A. Hertzmann, D. Zorin, "Illustrating smooth surfaces," in *Proc. SIGGRAPH'00*, pp. 517-526, July 2000.

[4] J. A. Sethian and D. Adalsteinsson, "An overview of level set methods for etching, deposition, and lithography development," *IEEE Transactions on Semiconductor Manufacturing*, vol. 10, no. 1, pp. 167-184, February 1997.

[5] A. R. Pope, "Model-based object recognition: A survey of recent research," Technical Report TR-94-04, University of British Columbia, Department of Computer Science, January 1994.

[6] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes, Computer Graphics: Principles and Practice. Reading, MA: Addison-Wesley Publishing Co, 2nd ed., 1990.

[7] A. Appel, "Some Techniques for shading machine renderings of solids," in *Spring Joint Computer Conf.*, vol. 32, pages 37-45, 1968.

[8] I. E. Sutherland, R. A. Schumacker, and R. F. Sproull, "A characterization of ten hidden-surface algorithms," *Computing Surveys*, vol. 6, no. 1, pp. 1-55, March 1974.

[9] J. Arvo and D. Kirk, "A survey of ray tracing acceleration techniques," *An Introduction to Ray Tracing*, Academic Press, pp. 201-262, 1989.

[10] D. Cohen-Or, Y. L. Chrysanthou, C. T. Silva, and F. Durand, "A survey of visibility for walkthrough applications," *IEEE Transactions on Visualization and Computer Graphics*, vol. 9, no. 3, pp. 412-431, July 2003.

[11] E. E. Catmull, A Subdivision Algorithm Computer Display of Curved Surfaces. Ph.D. thesis, University of Utah, December 1974.

[12] E. E. Catmull, "Computer display of curved surfaces," in *Proc. of the IEEE Conf. on Computer Graphics, Pattern Recognition, and Data Structure*, pp. 11-17, 1975.

[13] A. J. Stewart and T. Karkanis, "Computing the approximate visibility map, with applications to form factors and discontinuity meshing," in *Proc. of the Ninth Eurographics Workshop on Rendering*, pp. 57-68, June 1998.

[14] J. H. Clark, "Hierarchical geometric models for visible surface algorithms," *Communications of the ACM*, vol. 19, no. 10, pp. 547-554, October 1976.

[15] B. Garlick, D. Baum, and J. Winget, "Parallel algorithms and architectures for 3D image generation," *SIGGRAH'90 Course Notes*, vol. 28, pp. 239-245, 1990.

[16] S. Coorg and S. Teller, "Real-time occlusion culling for models with large occluders", in *Proc. of the Symp. on Interactive 3D Graphics*, pp. 83-90, April 1997.

[17] S. Kumar, D. Manocha, W. Garrett, and M. Lin, "Hierarchical back-face computation," *Computer & Graphics*, vol. 23, no. 5: 681-692, October 1999.

[18] E. Zhang and G. Turk, "Visibility-Guided Simplification," in *Proc. of the IEEE Visualization Conf.*, pp. 267-274, October 2002.

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

[20] J. N. Tsitsiklis, "Efficient algorithms for globally optimal trajectories," *IEEE Transactions on Automatic Control*, vol. 40, no. 9, pp. 1528-1538, September 1995.

[21] Y.-H. R. Tsai, L.-T. Cheng, S. Osher, P. Burchard, G. Sapiro, "Visibility and its dynamics in a PDE based implicit framework," *Journal of Computational Physics*, vol. 199, no. 1, pp. 260-290, September 2004.

[22] Y.-H. R. Tsai, L.-T. Cheng, S. Osher, P. Burchard, G. Sapiro, "Dynamic visibility in an implicit framework," UCLA CAM report, vol 2, no. 6, 2002.

[23] C. B. Jones, "A new approach to the 'hidden line' problem," *Computer Journal*, vol. 14, no. 3, pp. 232-237, August 1971.

[24] D. J. Meagher, "Efficient synthetic image generation of arbitrary 3-D objects," in *Proc. IEEE Computer Society Conf. on Pattern Recognition and Image Processing*, pp. 473-478, June 1982.

[25] J. M. Airey, J. H. Rohlf, and F.P. Brooks, "Towards image realism with interactive update rates in complex virtual building enviroments," *Computer Graphics*, vol. 24, no. 2, pp. 41-50, March 1990.

[26] S. J. Teller and C. H. Séquin, "Visibility preprocessing for interactive walkthroughs," in *Proc. of SIGGRAPH'91*, pp. 61-69, July 1991.

[27] N. Greene, M. Kass, and G. Miller, "Hierarchical Z-buffer visibility," in *Proc. SIGGRAPH'93*, pp. 231-238, August 1993.

[28] C. W. Grant, Visibility Algorithms in Image Synthesis. Ph.D. thesis, University of California, Davis, 1992.

[29] J. Bittner, P. Wonka, "Visibility in computer graphics," *Journal of Enviroment and Planning B: Planning and Design*, vol. 30, no. 5, pp. 719-725, September 2003.

[30] S. Kolmanic and N. Guid, "FlattGen: teaching tool for surface flattening," *Computer Applications in Engineering Education*, vol. 14, no. 2, pp. 106-119, August 2006.

[31] A. N. Darvill, D. E. Williams, and B. E. Taylor, "Method and apparatus for deriving the net form for shoe uppers," BP 1375 822 Patent Office, London, 1974.

[32] P. Richardus and R. K. Adler, Map projections for geodesists, cartographers and geographers. Amsterdam: North-Holland, January 1972.

[33] F. Pearson, B. Nordenskjold, and T. J. Lam, Map Projections: Theory and Applications. CRC Press, March 1990.

[34] S. Haker, S. Angenent, A. Tannenbaum, and R. Kikinis, "Nondistorting flattening maps and the 3-D visualization of colon CT images," *IEEE Transactions on Medical Imaging*, vol. 19, no. 7, pp. 665-670, July 2000.

[35] A. V. Bartrolí, R. Wegenkittl, A. König, E. Gröller, and E. Sorantin, "Virtual colon flattening," *VisSym'01 Joint Eurographics - IEEE TCVG Symp. on Visualization*, pp. 127-136, May 2001.

[36] W. Hong, M. Jin, and A. Kaufman, "Conformal virtual colon flattening," in *Proc. ACM Symp. on Solid and Physical Modeling*, pp. 85-94, June 2005.

[37] S. Angenent, S. Haker, A. Tannenbaum, and R. Kikinis, "On the Laplace-Beltrami operator and brain surface flatenning," *IEEE Transactions on Medical Imaging*, vol.18, no. 8, pp. 700-711, August 1999.

[38] I. D. Faux and M. J. Pratt, Computational Geometry for Design and Manufacture. Ellis Harwood, Chichester: Prentice Hall Europe, January 1979.

[39] B. Gurunathan and S. G. Dhande, "Algorithms for development of certain classes of ruled surfaces," *Computer & Graphics*, vol. 11, no. 2, pp. 105-112, 1987.

[40] J. Lang and O. Röschel, "Developable (1, n)-Bézier surfaces," *Computer-Aided Geometric Design*, vol. 9, no. 4, pp. 291-298, September 1992.

[41] H. Pottmann and G. Farin, "Developable rational Bézier and B-spline surfaces," *Computer-Aided Geometric Design*, vol. 12, no. 5, pp. 513-531, August 1995.

[42] J. C. Clements, "A computer system to derive developable hull surfaces and tables offsets," *Marine Technology*, vol. 18, no. 3, pp. 227-233, July 1981.

[43] M. C. Gan, S. T. Tan, and K. W. Chan, "Generating and flattening of developable surfaces," in *Proc. ASME Design Technical Conf.*, vol. 69-71, pp. 359-366, September 1994.

[44] H. Pottmann and J. Wallner, "Approximation algorithms for developable surfaces," *Computer-Aided Geometric Design*, vol. 16, no. 6, pp. 539-556, June 1999.

[45] C. Bennis, J.-M. Vézien, and G. Iglésias, "Piecewise surface flattening for nondistorted texture mapping," *Computer Graphics*, vol. 25, no. 4, pp. 237-246, July 1991.

[46] J. Hoschek, "Approximation of surfaces of revolution by developable surfaces," *Computer Aided Design*, vol. 30, no. 10, pp. 757-763, September 1998.

[47] S. Kolmanic and N. Guid, "The flattening of arbitrary surfaces by approximation with developable stripes," in *Proc. Seventh Workshop on Geometric Modelling: Fundamentals and Applications*, pp. 35-46, October 2000.

[48] S.-D. Ma and H. Lin, "Optimal texture mapping," in *Proc. EUROGRAPHICS'88*, pp. 421-428, September 1988.

[49] E. L. Schartz, B. Merker, E. Wolfson, and A. Shaw, "Computational neuroscience: applications of computer graphics and image processing to two and three modelling of functional architecture of visual cortex," *IEEE Computer Graphics and Applications*, vol. 8, no. 4, pp. 13-28, July 1988.

[50] G. Zigelman, R. Kimmel, and N. Kiryati, "Texture mapping using surface flattening via multidimensional scaling," *IEEE Transactions on Visualization and Computer Graphics*, vol. 8, no. 2, pp. 198-207, April 2002.

[51] M. S. Floater and K. Hormann, "Recent advances in surface parameterization," in *Multiresolution in Geometric Modelling*, pp. 259-284, 2003.

[52] M. Eck, T. Derose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle, "Multiresolution analysis of arbitrary meshes," in *Proc. of SIGGRAPH 95*, pp. 173-182, August 1995.

[53] M. S. Floater, "Parametrization and smoth approximation of surface triangulations," *Computer-Aided Geometric Design*, vol. 14, no. 3, pp. 231-271, April 1997.

[54] S. Haker, S. Angenent, A. Tannebaum, R. Kikinis, G. Sapiro, M. Halle, "Conformal surface parameterization for texture mapping," *IEEE Transactions on Visualization and Computer Graphics*, vol. 6, no. 2, pp. 181-189, April 2000.

[55] B. Levy, S. Petitjean, N. Ray, and J. Maillot, "Least squares conformal maps for automatic texture atlas generation," *ACM Transactions on Graphic*, vol. 21, no. 3, pp. 362-371, July 2002.

[56] S. Yoshizawa, A. Belyaev, and H. P. Seidel, "A fast and simple stretch-minimizing mesh parameterization," in *Proc. Shape Modeling International*, pp. 200-208, July 2004.

[57] Y. Zhong and B. Xu, "A physically based method for triangulated surface flattening," *CAD Computer-Aided Design*, vol. 38, no. 10, pp. 1062-1073, October 2006.

[58] S. He, R. Dai, B. Lu, C. Cao, H. Bai, and B. Jing, "Medial axis reformation: a new visualization method for CT angiography," Academic Radiology, vol. 8, pp. 726-733, August 2001.

[59] O. Figueiredo and R. D. Hersch, "Parallel unfolding and visualization of curved surfaces extracted from large three-dimensional volumes," *Journal of Electronic Imaging*, vol. 11, no. 4, pp. 423-433, October 2002.

[60] A. Kanitsar, R. Wegenkittl, D. Fleishmann, and M.E. Gröller, "Advanced curved planar reformation: flattening of vascular structures," in *Proc. IEEE Visualization 2003*, pp. 43-50, October 2003.

[61] L. Saroul, O. Figueiredo, and R. D. Hersch, "Distance preserving flattening of surface sections," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 1, pp. 26-35, January 2006.

[62] G. Hermosillo, O. Faugeras, and J. Gomes, "Cortex unfolding using level set methods," Technical Report No 3663, INRIA, April 1999.

[63] G. Sapiro and A. Tannenbaum, "Area and length preserving geometric invariant scalespaces," PAMI, vol. 17, no. 1, pp. 67-72, January 1995.

[64] J.-P. Pons, R. Keriven, and O. Faugeras, "Area preserving cortex unfolding," MICCAI, 376-383, 2004.

[65] J. McCartney, B. K. Hinds, and B. L. Seow, "The flattening of triangulated surfaces incorporating darts and gussets," *Computer-Aided Design*, vol. 31, no. 4, pp.249-260, April 1999.

[66] J. McCartney, B. K. Hinds, and K. W. Chong, "Pattern flattening for orthotropic materials," *Computer-Aided Design*, vol. 37, no. 6, pp. 631-644, May 2005.

[67] C. C. L. Wang, K. Tang, and B. M. L. Yeung, "Freeform surface flattening based on fitting a woven mesh model," *Computer-Aided Design*, vol. 37, no. 8, pp. 799-814, July 2005.

[68] K. Rocha, A. Yezzi, and J. Prince, "A hybrid Eulerian-Lagrangian approach for thickness, correspondence, and gridding of annular tissues," *Lecture Notes in Computer Science*, vol. 3765, pp. 72-81, October 2005.

[69] K. Rocha, A. Yezzi, and J. Prince, "A hybrid Eulerian-Lagrangian approach for thickness, correspondence, and gridding of annular tissues," to be published on *IEEE Transactions on Medical Imaging*.

[70] A. Duci, A. Yezzi, S. Soatto, and K. Rocha, "Harmonic embedding for linear shape analysis," *Journal of Mathematical Imaging and Vision*, vol. 25, no. 3, pp. 341-352, October 2006.

[71] D. MacDonald, N. Kabani, D. Avis, and A. C. Evans, “Automated 3-D extraction of inner and outer surfaces of cerebral cortex from MRI,” *NeuroImage*, vol. 12, no. 3, pp. 340-356, September 2000.

[72] C. C. Henery and T. M. Mayhew, “The cerebrum and cerebellum of the fixed human brain: efficient and unbiased estimates of volumes and cortical surface areas,” *Journal of Anatomy*, vol. 167, pp. 167-180, December 1989.

[73] G. Paxinos, The Human Nervous System. San Diego, CA: Academic Press, January 1990.

[74] C. Economo and G. N. Koskinas, Die Cytoarchitektonik der Himrinde des Erwachsenen. Berlin: Springer, 1925.

[75] J. Park, D. Metaxas, and L. Axel, “Analysis of left ventricular wall motion based on volumetric deformable models and MRI-SPAMM,” *Medical Image Analysis*, vol. 1, no.1, pp. 53-71, March 1996.

[76] A. H. Aletras, R. S. Balaban, and H. Wen, “High-resolution strain analysis of the human heart with fast-DENSE,” *Journal of Magnetic Resonance*, vol. 140, no. 1, pp. 41-57, September 1999.

[77] S. E. Jones, B. R. Buchbinder, and I. Aharon, “Three-dimensional mapping of the cortical thickness using Laplace’s equation,” *Human Brain Mapping*, vol. 11, pp. 12-32, 2000.

[78] A. J. Yezzi Jr. and J. L. Prince, “An Eulerian PDE approach for computing tissue thickness,” *IEEE Transactions on Medical Imaging*, vol. 22, no. 10, pp. 1332-1339, October 2003.

[79] A. Yezzi and J. L. Prince, "A PDE Approach for Thickness, Correspondence, and Gridding of Annular Tissues", in *Proc. European Conf. on Computer Vision (ECCV)*, Copenhagen, 2002.

[80] W. Press, B. Flannery, S. Teukolsky, and W. Vetterling, Numerical Recipes: The Art of Scientific Computing. Cambridge, U.K.: Cambridge Univ. Press, 1988.

[81] A. Duci, A. Yezzi, S. K. Mitter, and S. Soatto, “Shape Representation via Harmonic Embedding,” in *Proc. International Conf. on Comp. Vision*, vol. 1, pp. 656-663, October 2003.

[82] S. M. Pizer, D. Eberly, D. Fritsch, and B. S. Morse, “Zoom-invariant vision of figural shape: The mathematics of cores,” *Computer Vision and Image Understanding*, vol. 69, no. 1, pp. 55-71, January 1998.

[83] K. R. Rocha, G. Sundaramoorthi, and A. J. Yezzi, “3D topology preserving flows for viewpoint-based cortical unfolding,” Proceedings IEEE Computer Society Workshop on Mathematical Methods in Biomedical Image Analysis, October 2007.

[84] K. R. Rocha, A. J. Yezzi, A. Mennucci, and J. L. Prince, "Viewpoint-based maximizing flows," Proceeding MICCAI Workshop on Interaction in Medical Image Analysis and Visualization, November 2008.

[85] G. Sundaramoorthi and A. Yezzi, "More-than-topology-preserving flows for active contours and polygons," in *Proc. of the 10th IEEE International Conf. on Computer Vision, ICCV 2005,* pp. 1276-1283, October 2005.

[86] X. Han, C. Xu, D. Tosun, and J. L. Prince, "Cortical surface reconstruction using a topology preserving geometric deformable model," in *Proc. of MMBIA*, pp. 213-220, 2001.

[87] X. Han, C. Xu, and J. L. Prince, "A topology preserving level set method for geometric deformable models," in *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 25, no. 6, pp. 755-768, 2003.

[88] G. Unal, A. Yezzi, and H. Krim, "Information-theoretic active polygons for unsupervised texture segmentation," in *International Journal of Computer Vision,* vol. 62, no. 3, pp. 199-220, 2005.

[89] J. O'Hara, "Energy of a knot," in *Topology,* vol. 30, pp. 241-247, 1991.

[90] A. Abrams, J. Cantarella, J. Fu, M. Ghomi, and R. Howard, "Circles minimize most knot energies," in *Topology,* vol. 42, pp. 381-394, 2003.

[91] A. Witkin and D. Baraff, Physically Based Modelling: Principles and Practice. SIGGRAPH'97 Course Notes, pp. 1-13, 1997.

[92] J. H. Cantarella, E. D. Demaine, H. N. Iben, and J. F. O'Brian, "An energy-driven approach to linkage unfolding," in *Proc. of the Twentieth Annual Symp. on Computational Geometry*, pp. 134-143, June 2004.

[93] H. N. Iben, J. R. O'Brien, and E. D. Demaine, "Refolding planar polygons," in *Proc. of the Twenty-Second Annual Symp. on Computational Geometry*, pp. 71-79, June 2006.

[94] G. Slabaugh and G. Unal, "Active Polyhedron: Surface evolution theory applied to deformable meshes," in *Proc. of the IEEE Computer Society Workshop on Mathematical Methods in Biomedical Analysis*, pp. 84-91, 2005.

[95] Y. Shi and W. C. Karl, "Differentiable minimum shape distance for incorporating topological priors in biomedical imaging," in *IEEE International Symposium on Biomedical Imaging*, pp. 1247-1250, 2004.

[96] O. Alexandrov and F. Santosa, "A topology-preserving level set method for shape optimization," in *Journal of Computational Physics*, vol. 204, pp. 121-130, 2005.

[97] C. Le Guyader and L. Vesse, "Self-repelling snakes for topology segmentation models," Technical Report, UCLA, 2007.

[98] K. R. Rocha, G. Sundaramoorthi, A. J. Yezzi, and J. L. Prince, "3D topology-preserving flows for viewpoint-based cortical unfolding," Submitted to the *International Journal on Computer Vision*, January 2007.