

Deformable Velcro Surfaces

W. Neuenschwander*, P. Fua[†], G. Székely*, and O. Kübler*

* Communication Technology

Laboratory

Swiss Federal Institute of

Technology ETH

CH-8092 Zurich, Switzerland

[†] SRI International

Artificial Intelligence Center

333 Ravenswood Avenue

Menlo Park, CA 94025, USA

Abstract

Even though methods based on the use of deformable models have become prevalent, the quality of their output depends critically on the model's initial state. The issue of initializing such models, however, has not received much attention even though it is often key to the implementation of a truly useful system.

We therefore present a new approach to segmentation of 3-Dimensional shapes that initializes and then optimizes a 3-D surface model given only the data and a very small number of 3-D seed points and corresponding surface normals. This is a valuable capability for medical, robotic and cartographic applications where such seed points can be naturally supplied. In effect, the surface model is clamped onto the object boundary in manner reminiscent of a Velcro being closed. We develop the method's mathematic framework and show results using volumetric medical data.

1 Introduction

Deformable models have emerged as a very powerful tool for semiautomated surface modeling and 3-Dimensional image segmentation in applications as diverse as medical imaging, graphics, robotics or terrain-modeling. The vast majority of deformable model approaches are optimization-based and their output critically depends on the model's initial state. Typically, the noisier the data, the smaller the radius of convergence of the method and the more critical it becomes to pick a good initial state for the model. The problem of initialization, however, has received comparatively little attention even though its solution is key to the viable implementation of a working and usable system.

In this paper, we present a method — and develop the corresponding mathematical framework — that allows a user to initialize and then optimize a 3-D surface model by supplying only a very small number of 3-D seed points and corresponding surface normals. This is a valuable capability because there are many applications in which imposing initial conditions in this manner is both easy and natural. Such applications include:

- **Segmentation of 3-D shapes from volumetric data.** The extraction of anatomical organs from medical-data can be achieved semiautomatically with very little effort on the user's part. As is the case for outlining model slices using 2-D deformable models [1], the user can specify 3-D anchor points by clicking on orthogonal 2-D slices, while the normals are provided by convolving the data with an appropriate filter. Since we need only a few points and can extract them from slices of any orientation, we believe that our method will lead to the implementation of a more generic and easier to use tool.
- **Incremental construction of a world model by a mobile robot.** Assuming that the robot has already modeled the surfaces it has previously seen, new input data such as range data can be used to extend these models by using the already modeled portions of the surfaces to constrain the reconstruction of the new ones. If one uses a facetized

representation, the normals and vertices of the existing facets provide the required seed points and normals.

- **Construction of composite models for high-resolution cartographic modeling.**

Man-made structures such as buildings or roads and natural features as ridge-lines become clearly visible on high-resolution cartographic models and must be handled explicitly. They, in turn, supply constraints that ought to be used to constrain the terrain reconstruction process so as to ensure consistency.

In this paper, we demonstrate the viability of our approach in the context of medical applications.

Our technique relies on classical elastic models that are represented as triangulated meshes and deform themselves to minimize an objective function [2]. The objective function is the sum of a data term and a regularization term. We depart from other approaches in our handling of the initialization and convergence issues. We show that a small set of 3-D seed points provides sufficient boundary conditions to solve the differential equation that governs the model's behavior in closed form, assuming that the data component of the objective function remains constant. As a result, it becomes possible to instantiate the model using these points alone by initially ignoring the data term and then propagating constraints along the surface by progressively "turning it on." In this way, we can achieve very good convergence properties even in noisy data using only a minimal set of constraints.

This paper addresses the issues of 3-D initialization and subsequent convergence. It allows the constraining of the model's optimization process in a much more natural and generic way for both semiautomated and fully automatic applications.

Our approach to surface reconstruction is the generalization of an approach to 2-D delineation we have proposed earlier [3] and review briefly, along with related methods, in the next section. We then introduce our mathematical framework, describe its implementation and finally present results using volumetric data.

2 Related Work

Some deformable-model approaches assume that the input data has been partitioned and that one target object has been selected. The corresponding data points or voxels can then be turned into attractors that induce the model’s deformation over large distances. However, partitioning is a notoriously hard problem. In most practical applications, the data contains multiple objects and may be noisy so that no initial segmentation can readily be obtained.

While this segmentation problem can be overcome by using fairly rigid models [4, 5, 6]—that is, models in which a lot of *a priori* knowledge is embedded and consequently have relatively few degrees of freedom—it rapidly becomes intractable as the model becomes more generic and the number of its degrees of freedom increases. In the presence of multiple objects, a deformable model becomes very likely to get “caught” by objects other than the one that is to be extracted.

Traditional approaches typically address this issue by either imposing soft interactive constraints in the form of springs, volcanos, etc...[7] or adding energy terms that force the model to either expand or shrink [8, 9, 10, 11]. The difficulty then becomes choosing the magnitude of these terms so as to push the model out of undesirable local minima without also pushing it out of the desired one. In practice, this often turns out to be extremely difficult, if not impossible, to do reliably.

In our earlier 2-Dimensional approach [3] we bypass this problem by propagating constraints along the snake, starting from the seed points, so as to achieve convergence without having to nudge the snake using arbitrary energy terms.

The method described in this paper extends to the 3-D domain our earlier *Ziplock Snake* approach. In the 2-D case, we allow a user to delineate an image curve by merely supplying two endpoints with the mouse. The optimization then progresses from the endpoints toward the center, thereby effectively propagating edge information along the curve. The user-supplied endpoints and the automatically computed edge gradients in their vicinities serve as anchors. They are first used to compute, without using the image potential, an initial state that is ap-

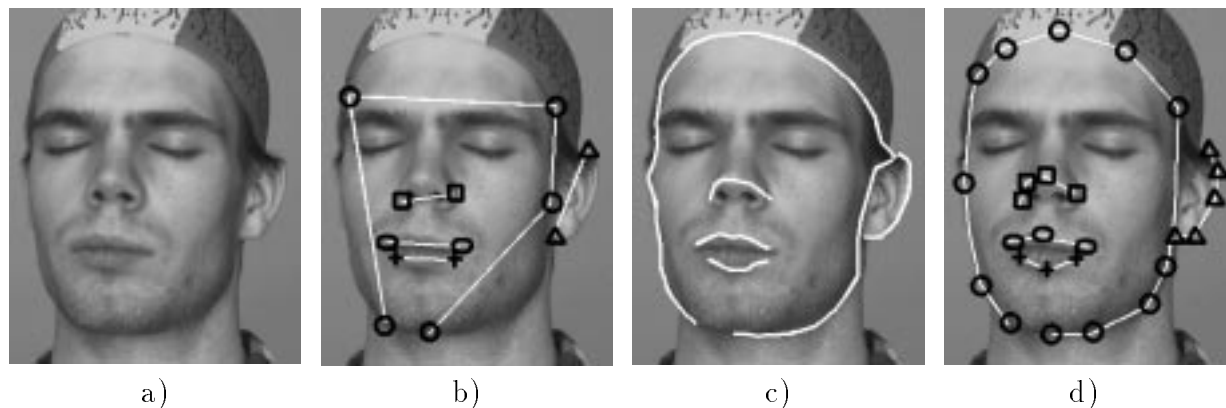


Figure 1: Outlining facial features. (a) A face image with low contrast contours. (b) Five sets of initial points, each denoted by a different symbol. Four of them are pairs of end points while the fifth is shown as a set of circles. (c) The contours delineated by Ziplock Snakes. (d) The initial delineations that must be supplied to achieve the same result using traditional snakes.

proximately correct in each anchor’s vicinity. The image term is then “turned on” progressively from the snake’s extremities toward its middle section. Still using the endpoints as anchors, the snake’s position is iteratively recomputed until the image term is fully turned on. As a result, the snake is progressively clamped onto an image contour so that it smoothly connects the two endpoints and has the right orientation at these points. To delineate convoluted curves, the user can also supply polygons whose every segment is then refined using a different ziplock-snake. In Figure 1, we show that these snakes require much less initialization work than traditional snakes [7] and converge from initial estimates that are much rougher.

3 Velcro Surfaces

We extend the *Ziplock Snake* approach to initialize and optimize a 3-D deformable surface model by supplying only a very small number of 3-D seed points and corresponding surface normals¹.

¹While supplying an initial estimate of a contour model is fairly easy and straightforward the initialization of a surface model is much more difficult. To help the the human operator, we have implemented a tool to inspect 3-D volume data and select the seed points necessary for the initialization.

An intuitive, easy and reliable way to provide model information is to interactively specify 3-D data points. As shown below, a small set of 3-D seed points provides sufficient boundary conditions to solve the differential equation that governs the model's behavior in closed form, assuming that the data component of the objective function remains constant. As a result, it becomes possible to instantiate the model using these points alone by initially ignoring the data term and then propagating constraints along the surface by progressively "turning it on". The tessellated surface behaves like a piece of Velcro that is progressively being clamped onto the surface of interest, hence the name of the surfaces.

First we present a general deformable surface model and its associated objective function together with the corresponding governing differential equation. We express the various derivatives by differential operators such that the expression for the objective function becomes independent of the chosen parameter system used to represent the model. This enables us to choose any discretization of the model which suits best the given segmentation problem. In order to approximate the differential operators we use finite difference approximations which yield typically large sets of linear equations where the system is sparse and ill-conditioned. We then address the problems encountered when solving these systems. The set of linear equations turns out to be underconstrained. We demonstrate how to overcome this problem by incorporating boundary conditions. Finally the optimization schedule that defines the gradual "turning on" of the image forces is presented. The computation of the first initial estimate of the surfaces is shown to be the very first optimization step of our scheme.

3.1 Deformable Surface Models

In contrast to 2-D active contour models where arc length provides a natural parameterization, 2-D manifolds as used for 3-D deformable models pose a complex, topology and shape dependent parameterization problem. Two basic difficulties of using explicit surface parameterization are:

- There is no easy way to evenly distribute the grid points across the surface.

- In many cases regular neighborhood relations cannot be assured on the surface grid.

Since we aim at using finite differences to approximate the governing equations we propose to use no specific parameterization. Note, that this can be justified only if the model's objective function can be written in a parameter invariant form as shown below. Since we make no assumption about how the model is tessellated, we use a notation which avoids specific parameter systems. We develop a mathematical kernel of the deformable surface framework which is invariant under different shape and model topologies. When adapting the framework to a specific task the only problem left is to specify proper boundary conditions. We define a generalized deformable surface model as

$$\begin{aligned} \vec{v} : \Omega \subset \mathbb{R}^2 &\rightarrow \mathbb{R}^3 \\ \boldsymbol{\omega} &\mapsto \vec{v}(\boldsymbol{\omega}) = (v^{(1)}(\boldsymbol{\omega}), v^{(2)}(\boldsymbol{\omega}), v^{(3)}(\boldsymbol{\omega})) \end{aligned} \quad (1)$$

where $\boldsymbol{\omega}$ is a suitable parameterization and $v^{(1)}, v^{(2)}$ and $v^{(3)}$ are the corresponding coordinate functions of the surface. The model \vec{v} has to be viewed as a body evolving in a 3-D force field which is associated with the external image. The deformation of the model is governed by an energy expression of the form

$$E(\vec{v}) = E_I(\vec{v}) + E_D(\vec{v}) . \quad (2)$$

where the image energy is defined as:

$$E_I(\vec{v}) = \Leftrightarrow \iint_{\Omega} P(\vec{v}(\boldsymbol{\omega})) d\boldsymbol{\omega} \quad (3)$$

where $P(\vec{v})$ depends on the task at hand. When dealing with MR images, we want to detect object boundaries and take $P(\vec{v})$ to be:

$$P(\vec{v}(\boldsymbol{\omega})) = |\nabla I(\vec{v}(\boldsymbol{\omega}))| \quad (4)$$

where I denotes either the image itself or the image convolved by a Gaussian kernel. When dealing with stereo imagery we define $P(\vec{v})$ in the manner described in [12].

The convex regularization term $E_D(\vec{v})$ is introduced to overcome problems associated with the minimization of the potential energy. Let $\omega = (\omega_1, \omega_2) \in \Omega$ be an appropriate set of parameters. We reformulate (see Appendix) the standard expression [13]

$$E_D(\vec{v}) = \iint_{\Omega} \tau(\omega) \left[\left| \frac{\partial \vec{v}}{\partial \omega_1} \right|^2 + \left| \frac{\partial \vec{v}}{\partial \omega_2} \right|^2 \right] + \rho(\omega) \left[\left| \frac{\partial^2 \vec{v}}{\partial \omega_1^2} \right|^2 + 2 \left| \frac{\partial^2 \vec{v}}{\partial \omega_1 \partial \omega_2} \right|^2 + \left| \frac{\partial^2 \vec{v}}{\partial \omega_2^2} \right|^2 \right] d\omega \quad (5)$$

to get E_D in a notation invariant to the parameterization:

$$E_D(\vec{v}) = \sum_{i=1}^3 \iint_{\Omega} \varphi(\omega) \left\{ \tau(\omega) |\nabla v^{(i)}|^2 + \rho(\omega) \left[(\Delta v^{(i)})^2 \Leftrightarrow 2H(v^{(i)}) \right] \right\} d\omega \quad (6)$$

where $\rho(\omega) = 1 \Leftrightarrow \tau(\omega)$ and the functions φ and τ control surface cohesion and tension. In theory, φ and τ can be arbitrary functions, which makes the model very flexible. However, φ and τ should be chosen such that the surface has homogeneous properties except, possibly, for very specific and clear-cut locations. In the implementation described below, we take the surface cohesion parameter, φ , to be uniformly equal to 1 and the tension parameter, τ , to be a constant between 0 and 1 supplied by the user.

The minimization of the model's total energy $E(\vec{v}) = E_I(\vec{v}) + E_D(\vec{v})$ can be achieved by solving the corresponding Euler differential equation. From variational calculus [14] it is well known that if $\vec{v}(\omega)$ minimizes $E = E_D + E_I$ and is sufficiently regular, that is at least $C^4(\Omega)$, then it must be a solution of the set of three coupled Euler differential equations

$$\begin{aligned} \Leftrightarrow \tau \Delta v^{(1)} + (1 \Leftrightarrow \tau) \Delta^2 v^{(1)} &= \Leftrightarrow \frac{\partial P}{\partial v^{(1)}} \\ \Leftrightarrow \tau \Delta v^{(2)} + (1 \Leftrightarrow \tau) \Delta^2 v^{(2)} &= \Leftrightarrow \frac{\partial P}{\partial v^{(2)}} \\ \Leftrightarrow \tau \Delta v^{(3)} + (1 \Leftrightarrow \tau) \Delta^2 v^{(3)} &= \Leftrightarrow \frac{\partial P}{\partial v^{(3)}} \end{aligned} \quad (7)$$

where $v^{(1)}$, $v^{(2)}$, and $v^{(3)}$ represent the three degrees of freedom of each vertex, as defined by Equation (1).

These differential equations do not have a unique solution in the absence of boundary conditions. Note, however, that the choice of useful boundary conditions is quite limited if a classical solution of Equation (7) is to be obtained. Different topologies of the surface will also affect the nature of the solutions. The chosen notation is sufficiently general to deal with these cases.

3.2 Discretization of the Surface

We assume that the surface is topologically equivalent to a sphere since this is the case of most practical interest in medical image analysis. The problem of open surfaces can be handled similarly by taking special care at the boundaries. Since we approximate the governing equation using finite difference operators we aim at defining an appropriate strategy to produce a discrete mesh of the surface. The commonly used polar coordinate system creates two poles where the sampling of the surface in their neighborhood is much higher than at the “equator”. To avoid creating poles and achieve a more homogeneous distribution of the grid points we use a triangular tessellation. This is valid since the formulation we use does not require a specific parameterization of the surface. Therefore the surface is defined in a general way by a set of N vertices

$$\vec{v} = \{\vec{v}_i\}_{1 \leq i \leq N} = \left\{ \begin{array}{c} \left(v_i^{(1)} \right) \\ \left(v_i^{(2)} \right) \\ \left(v_i^{(3)} \right) \end{array} \right\}_{1 \leq i \leq N} = \{\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \mathbf{v}^{(3)}\} \quad (8)$$

Since the tessellated model consists of a set of connected points the mesh can be interpreted as an adjacency graph which we call \mathcal{G} . Each node on this graph has a particular number of neighboring nodes which is usually not the same for every node.

In order to compute the solution of the Equation (7) we define two discrete operators \mathcal{L}_1

and \mathcal{L}_2 which approximate the Laplacian Δ of Equation (7) and its square Δ^2 respectively. A discrete convolution of the weighted sum $\Leftrightarrow\tau\mathcal{L}_1 + (1 \Leftrightarrow\tau)\mathcal{L}_2$ with the adjacency graph \mathcal{G} which represents the triangulated surface, yields the discrete counter-part of Equation (7).

$$\Leftrightarrow\tau \mathcal{L}_1 * \mathbf{v} + (1 \Leftrightarrow\tau) \mathcal{L}_2 * \mathbf{v} = \Leftrightarrow\nabla P(\mathbf{v}) \quad (9)$$

where \mathbf{v} stands either for $\mathbf{v}^{(1)}, \mathbf{v}^{(2)}$ or $\mathbf{v}^{(3)}$ and \mathcal{L}_1 denotes the discrete Laplacian operator whereas its square \mathcal{L}_2 is obtained by convolution of appropriate \mathcal{L}_1 -operators.

We define the discrete Laplacian operator \mathcal{L}_1 and the operator \mathcal{L}_2 in accordance to boundary value problems in potential theory (see [14]). The value of the function at one particular grid point is given as the weighted average of the function values at the immediately adjacent grid points. In principle the weighting must reflect the local geometry around the grid point in question, but at present we calculate the weights only taking the local mesh topology into account. This is an acceptable approximation as long as the shape of a facet does not become too degenerated. Figure 2 shows the two discrete operators \mathcal{L}_1 and \mathcal{L}_2 for a mesh with hexagonal structure.

Since the deformation energy E_D in Equation (6) is quadratic and decouples the coordinates of the surface, Equation (9) can be written as a system of three linear equations

$$\begin{aligned} K \mathbf{v}^{(1)} &= \Leftrightarrow \frac{\partial P}{\partial v^{(1)}} \Big|_{(\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \mathbf{v}^{(3)})} \\ K \mathbf{v}^{(2)} &= \Leftrightarrow \frac{\partial P}{\partial v^{(2)}} \Big|_{(\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \mathbf{v}^{(3)})} \\ K \mathbf{v}^{(3)} &= \Leftrightarrow \frac{\partial P}{\partial v^{(3)}} \Big|_{(\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \mathbf{v}^{(3)})} \end{aligned} \quad (10)$$

which are coupled by the ‘‘image forces’’ depending on the surface’s location $\vec{\mathbf{v}} = (\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \mathbf{v}^{(3)})$, making the System (10) semi-linear.

The stiffness matrix K is singular. Therefore the system of the three linear Equations (10)

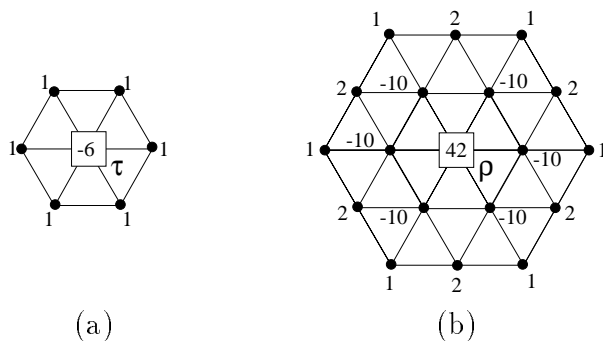


Figure 2: Finite difference operators on a hexagonal grid. (a) Discretization of the Laplacian \mathcal{L}_1 . The different numbers denote the weights used for the corresponding filter mask. (b) Discretization of the square of the Laplacian $\mathcal{L}_2 = \mathcal{L}_1 * \mathcal{L}_1$. The index τ denotes the membrane element whereas the index ρ denotes the thin plate element.

cannot be solved directly. The singularity of the matrix stems from the fact that we did not stipulate any boundary conditions when solving the differential Equation (7). These are the same problems we encountered when solving the corresponding Euler equation for ordinary snakes.

3.3 Incorporation of Boundary Conditions

Setting appropriate boundary conditions raises the question of how to derive the necessary boundary values and of what kind they should be. To facilitate the interaction with deformable models, we have developed an easy to use tool which fits into the mathematical framework of deformable models. It allows the human operator to supply a few 3-D “anchor-points” with reasonable ease. The local surface orientation can then be calculated using the directional information of some 3-D gradient operators as e.g. the 3-D Canny edge detector. This is by far one of the simplest and intuitive ways to provide the necessary information.

Assuming a set of anchor points and surface normals are given, a solution for the three homogeneous equations (thin plate problem without external forces)

$$\Delta^2 v^{(1)} = 0 \quad , \quad \Delta^2 v^{(2)} = 0 \quad , \quad \Delta^2 v^{(3)} = 0 \quad (11)$$

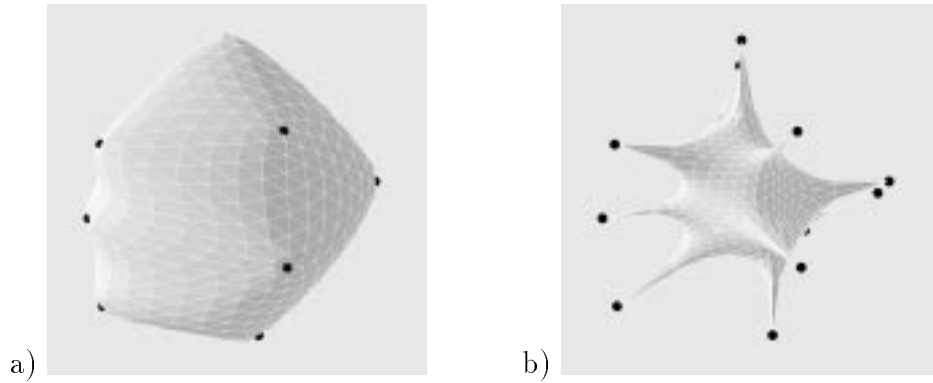


Figure 3: a) Surface model constrained to pass all fix points and have the according surface normals at their locations (for $\tau = 0$). b) Surface model constrained to pass all fix points for $\tau = 1$.

can be computed. Specifying boundary conditions for isolated points of deformable surfaces in principle leads to the theory of weak solutions and the associated mathematical framework for the minimization problem. The solution of this set of Equations (11) belongs to a Sobolev space and is therefore a *weak solution*. It is a smooth interpolation surface through all points that is as close to being spherical as possible. Figure 3a) exemplifies such a solution. Taking the tension parameter $\tau = 0$ and solving the set of three homogeneous equations

$$\Delta v^{(1)} = 0 \quad , \quad \Delta v^{(2)} = 0 \quad , \quad \Delta v^{(3)} = 0 \quad (12)$$

for a specific set of boundary points only, yields a solution depicted in Figure 3b). At the locations of the anchor-points the surface is still continuous but not differentiable anymore and therefore not a *classical* solution of Equation (12). For a detailed discussion on *weak* solutions of differential equations we refer the reader to [15].

Given a total number of M user-supplied anchor-points P_i ($4 \leq i \leq M$, non-coplanar) and the normal vector at their locations, the system of equations reduces to

$$K^* \cdot \mathbf{v}^* = F_{\mathbf{v}^*}^* \quad (13)$$

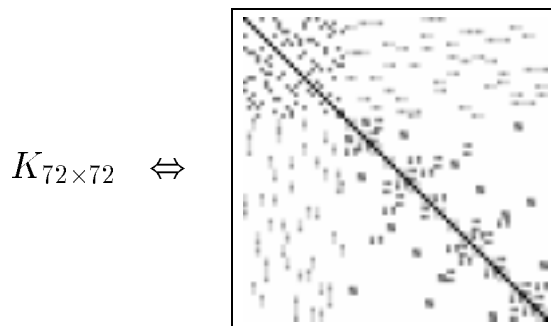


Figure 4: A typical sparse and symmetric stiffness matrix K for a model containing of 72 free nodes. Gray shaded squares denote non-zero entries in K .

where \mathbf{v}^* stands for either $\mathbf{v}^{(1)*}$, $\mathbf{v}^{(2)*}$ or $\mathbf{v}^{(3)*}$ the reduced vectors of the three coordinate functions, and K^* an $(N \Leftrightarrow M) \times (N \Leftrightarrow M)$ sparse matrix that is now invertible. Since we aim at segmenting closed 3-D objects we seek an initial model that extends in all three cardinal directions. This can be achieved by selecting at least 4 points which are non-coplanar such that a tetrahedron can be spanned through them otherwise the surface would be a flat sheet (3 points), a line (2 points) or a single point. The maximal number of seed points is not limited. Of course, since $F_{\mathbf{v}^*}^*$ depends on the surface's current position, the system in Equation (13) is still only semi-linear and cannot, in general, be solved in closed form.

3.4 Ill-conditioned sparse linear systems

The stiffness matrix K of Equation (10) for a surface model with 72 free nodes is shown in Figure 4. The gray shaded squares denote non-zero entries. In real applications models commonly consist of several thousand vertices leading to sparse stiffness matrices K with millions of entries. The solution of such a system of equations, however, can be computed efficiently using iterative linear solvers such as the ones provided by the package PILS [16]. It is important to note that a very sophisticated package should be used to solve these systems. There are many algorithms such as *Jacobi's method*, *Conjugate Gradient*, *BiConjugate Gradient*, *BiConjugate Gradient Squared*, *Orthomin*, *Generalized Minimum Residual Iteration* etc. that have been developed in the past and are available in various numeric computation packages. However,

since the stiffness matrix K is ill-conditioned, a powerful pre-conditioner is necessary before we can start solving for the unknowns. Some of the algorithms we tried out suffer from numerical instabilities even though the matrix was pre-conditioned. Therefore the numerical package one uses to solve the system of Equation (13) should fulfill the following requirements:

- It can deal with large matrices having a sparse structure.
- The pre-condition algorithm should be powerful.
- All the arithmetic operations have to be done in a numerically stable fashion.

In order to compare the performance we have used two different software packages, namely

1. PILS, implemented by [16].
2. Sparse, public domain software provided by [17].

The two packages use completely different algorithms but both of them yield satisfactory results.

3.5 Initialization

To successfully optimize the surface, we must start from an initial shape that is approximately correct in the neighborhood of the selected anchor-points. We are now left with the task of computing a surface which passes through the user-specified seed points and has the correct orientation at their locations. There are many possible ways of doing that. One way for example would be to use Bézier or cubic B-Spline surface patches to derive the desired interpolation. However, these approaches don't fit quite into the presented mathematical framework. Using the same strategy as for the *Ziplock Snake* the easiest way to achieve the desired result is to solve the homogeneous equations that correspond to the system of Equation (7). As discussed in Section 3.1, we take τ to be constant, and the homogeneous system becomes

$$\Leftrightarrow \Delta v + (1 \Leftrightarrow \tau) \Delta^2 v = 0 \tag{14}$$

where v stands for either $v^{(1)}, v^{(2)}$ or $v^{(3)}$. The initial surface we compute is a solution of Equation (14) that satisfies the set \mathcal{B} of boundary conditions defined by the 3-D anchor-points P_i and the corresponding surface normals \vec{n}_{P_i} . Therefore \mathcal{B} can be written as

$$\begin{aligned} \mathcal{B} = \quad & \{ \vec{v}_1 = P_1, \quad \vec{N}_{\vec{v}_1} = \vec{n}_{P_1}, \\ & \vec{v}_2 = P_2, \quad \vec{N}_{\vec{v}_2} = \vec{n}_{P_2}, \\ & \quad \quad \quad \vdots \quad \quad \quad \vdots \\ & \vec{v}_M = P_M, \quad \vec{N}_{\vec{v}_M} = \vec{n}_{P_M} \} \end{aligned} \tag{15}$$

Where $\vec{N}_{\vec{v}_1} = \vec{n}_{P_1}$ for example denotes that the discrete surface has to have a normal vector \vec{N} at the vertex $\vec{v}_1 = (x_1, y_1, z_1)^T$ pointing in the direction \vec{n}_{P_1} . This yields a set of linear equations which has to be solved for the $v^{(1)} \Leftrightarrow v^{(2)} \Leftrightarrow v^{(3)}$ components of all the grid points. By construction, this solution will pass through the anchor-points and have the specified normal vectors there, and be close to the final surface near these points. It can therefore serve as an initial surface for the subsequent minimization of the energy functional.

3.6 Supplying Anchor-Points

An effective mechanism for supplying anchor-points has to take into account the type of input that a user can provide with reliability and facility. While satisfactory solutions are available for 2-D deformable models, initializing becomes a truly difficult problem in the 3-Dimensional case. It is still possible, however, to provide individual surface locations or small surface patches and their normals with reasonable ease. Visual inspection of 3-D data sets simultaneously from the three cardinal directions is supported by appropriate user interfaces and allows to identify surface points interactively. On a technically more advanced level, surface detectors like 3-D generalizations of the Canny edge detector will yield patches of high reliability if a strict quality selection is performed. In addition, they allow the reliable computation of local surface normals

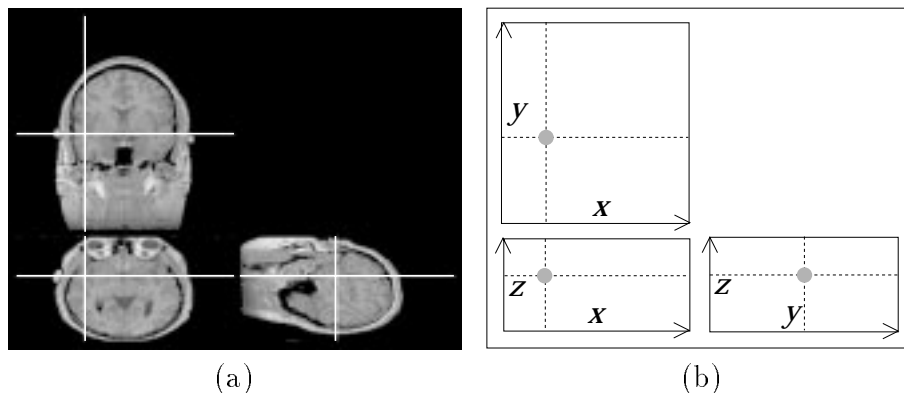


Figure 5: Interface for the seed point selection. (a) The browser allows a user to scan through a 3-D data stack (3-D MR data of human head) and inspect arbitrary slices from three different views. (b) Schematic overview of the interface.

provided that we focus on points of good surface definition.

In order to supply the necessary 3-D seed points to initialize the surface, we have built a special interface. The tool depicted by Figure 5(a) is a browser based on multi-planar reconstruction techniques that allows the human operator to inspect arbitrary slices in a 3-D data stack from three different views. The tool simultaneously updates the three orthogonal planes that cross each other at the actual 3-D cursor position, which is shown as cross-hair on each of the planes. The position of the cursor can be changed on any of the planes. In this fashion, one can always select a plane for position determination, that is far from being parallel to the target surface's local tangent plane. The image at the top shows the $x \leftrightarrow y \leftrightarrow$ plane whereas the two images below depicts the $z \leftrightarrow x \leftrightarrow$ and $z \leftrightarrow y \leftrightarrow$ planes. During inspection, the user selects the seed-points at clearly visible edges. The location of the chosen points are then optimized to insure accurate position on an edge.

When the user is satisfied with the set of chosen points he has to construct the first rough model. From the cloud of points the operator has selected a convex hull is generated. The algorithm splits the polyhedron into Delaunay-tetrahedra. Note, there will be only a few points lying on the surface of the convex hull polyhedron. Since all the selected points have to lie on the surface, a sculpturing phase must follow the selection process. This deletion of tetrahedra

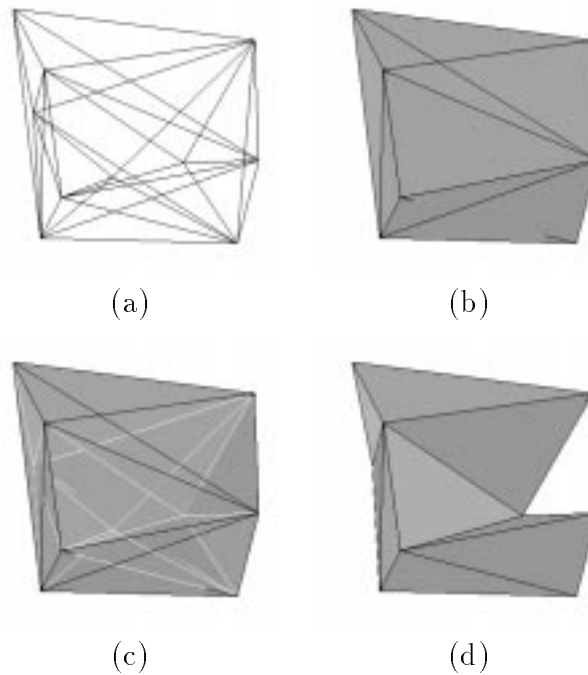


Figure 6: Computer aided interactive modeling. (a) Wire frame model of the convex hull of 9 user supplied seed points. The model consisting of 17 tetrahedra is hard to interpret. (b) Opaque view of the model. (c) By changing the model's transparency, points lying inside the convex hull become visible. This facilitates significantly the task of deleting particular tetrahedra. (d) Final polyhedral model which serves as boundary conditions for the Euler differential equation.

is presently performed manually using a 3-D graphical interface. Figure 6(a) illustrates a wire frame model of the convex polyhedron showing the tessellation into tetrahedra. Since this wire frame display is hard to interpret the user can adjust the transparency of the model. Figure 6(b) shows an opaque view of the same model. By adjusting the transparency, points still lying inside the model become visible like in Figure 6(c). Therefore it becomes easier to select the next tetrahedron to delete. Figure 6(d) depicts the final concave model having all the selected seed points as vertices. This sculpturing process guarantees that the deformable surface passes every seed point since this first rough model serves as boundary condition for the Euler differential equation.

3.7 The Optimization Procedure

The optimization strategy in 3-D is similar to the one we discussed for the *Ziplock Snake* in Section 2: We start the optimization of the energy term by defining the initial surface as the solution of the homogeneous differential system of Equation (14). At this stage the surface “feels” absolutely no external image forces. Therefore this first initial estimate minimizes the deformation energy only under the constraint of the boundary conditions. During the ongoing iterative optimization process the image potential P is taken into account progressively for all surface vertices, starting from the various anchor points. From each anchor point a region of *active* vertices starts to grow. The union of all vertices for which both the image term E_I and E_D is optimized build the *active* set \mathcal{A} . The rest of the vertices are called *passive* and build the set \mathcal{P} . As the optimization proceeds the set \mathcal{A} of active vertices grows and will eventually cover the whole surface until no passive points will be left over ($\mathcal{P} = \emptyset$). We call the border between the two regions a “force front”. This propagation of the force front is illustrated by Figure 7 where the black vertices denote the different anchor points. The light shaded node points denote the *active* surface vertices; facets with no explicitly drawn vertices are *passive*. Note, that the illustration is intended to show the concept of force front propagation. No external image forces are applied to the model in this case. The propagation mechanism is fairly simple up to now. Each front moves further by one vertex if the average displacement of the whole surface between two successive optimization steps is smaller than half a voxel. This strategy can be improved by moving every force front depending only on the motion of its own interior active vertices. A further refinement would be to allow non-concentric wave propagation. This would have the effect that the front moves faster across stable surface regions and propagates slower in areas of strong image forces. The whole optimization procedure can be divided into five basic steps which are depicted in Table 1.

Since the three systems of linear Equations (10) are only semi linear, they have to be solved using an iterative procedure. To enforce stability we introduce a viscosity term $\gamma(\omega, t)$ similar

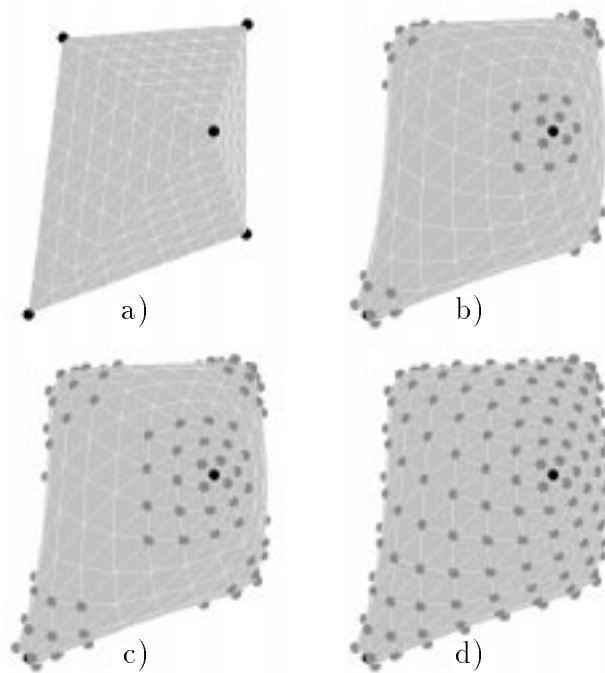


Figure 7: Propagation of the force boundaries. a) First polyhedral initialization performed by the user. b)-d) Solution of the Euler differential equation. At every seed point a force front expands on the surface. The black dots denote the various fix points while the light shaded vertices denote the active mesh nodes. Note, no external forces were turned on for the purpose of this illustration.

to the one used by traditional snakes and iteratively solve the equation

$$(K^* + \gamma^{[t]} \mathbf{I}) \cdot \mathbf{v}^{*[t]} = \gamma^{[t]} \mathbf{v}^{*[t-1]} + \mathbf{1} \Big|_{\gamma^{[t]}} \cdot F_{\mathbf{v}^{*[t-1]}}^* \quad (16)$$

where \mathbf{v} stands for either $\mathbf{v}^{(1)}$, $\mathbf{v}^{(2)}$ or $\mathbf{v}^{(3)}$ and

$$F_{\mathbf{v}^{*[t-1]}}^* = \begin{cases} \Leftrightarrow \frac{\partial P}{\partial v^{(1)}} \Big|_{\mathbf{v}^{*[t-1]}} & \text{if } \mathbf{v} = \mathbf{v}^{(1)} \\ \Leftrightarrow \frac{\partial P}{\partial v^{(2)}} \Big|_{\mathbf{v}^{*[t-1]}} & \text{if } \mathbf{v} = \mathbf{v}^{(2)} \\ \Leftrightarrow \frac{\partial P}{\partial v^{(3)}} \Big|_{\mathbf{v}^{*[t-1]}} & \text{if } \mathbf{v} = \mathbf{v}^{(3)} \end{cases}$$

- | | | |
|-----------|---|---|
| 1. | $\mathcal{N} = \{\text{all vertices}\} \setminus \{\text{anchor-points}\}$
$\mathcal{A} = \{\text{active vertices}\} = \emptyset$
$\mathcal{P} = \{\text{passive vertices}\} = \mathcal{N}$ | force field turned on
force field turned off |
| 2. | Incorporating the boundary conditions
defined by the anchor-elements | |
| 3. | Computing a solution of the Euler differential
equation for the set $\mathcal{N} = \mathcal{A} \cup \mathcal{P}$ | |
| 4. | Updating $\mathcal{A} = \mathcal{A} \cup \mathcal{M}$
and $\mathcal{P} = \mathcal{P} \setminus \mathcal{M}$ | \Leftrightarrow force front propagation
where \mathcal{M} is the boundary of \mathcal{P}
i.e. the vertices immediately adjacent to \mathcal{A} |
| 5. | Repeat 3. & 4. until $\mathcal{A} = \mathcal{N}$ ($\mathcal{P} = \emptyset$) | |

Table 1: The five basic steps of the Velcro optimization procedure.

The superscript $^{[t]}$ denotes the iteration step and $\mathbf{1}$ is an indicator function for the vector $\gamma^{[t]}$

$$\mathbf{1}_{\gamma^{[t]}} \in \mathbb{R}^{(N-M) \times (N-M)} \quad \text{with} \quad \mathbf{1}_{ij} = \begin{cases} 0 & \text{for } i \neq j \\ 1 & \text{if } \gamma_i^{[t]} > 0 \\ 0 & \text{otherwise} \end{cases} \quad (17)$$

which has the form of a diagonal matrix.

The viscosity $\gamma(\boldsymbol{\omega}, t)$ is initialized as $\gamma(\boldsymbol{\omega}, 0) \equiv 0$, $\boldsymbol{\omega} \in \Omega$ and recomputed each time the force boundaries move. For $t = 0$ the Equation (16) will directly yield the discrete version of the differential Equation (14) with boundary conditions. Hence the computation of the initial estimate $\vec{\mathbf{v}}^{*[0]}$ as explained in Section 3.5 is already the first optimization step and fits therefore perfectly into the proposed ‘‘Velcro’’ framework.

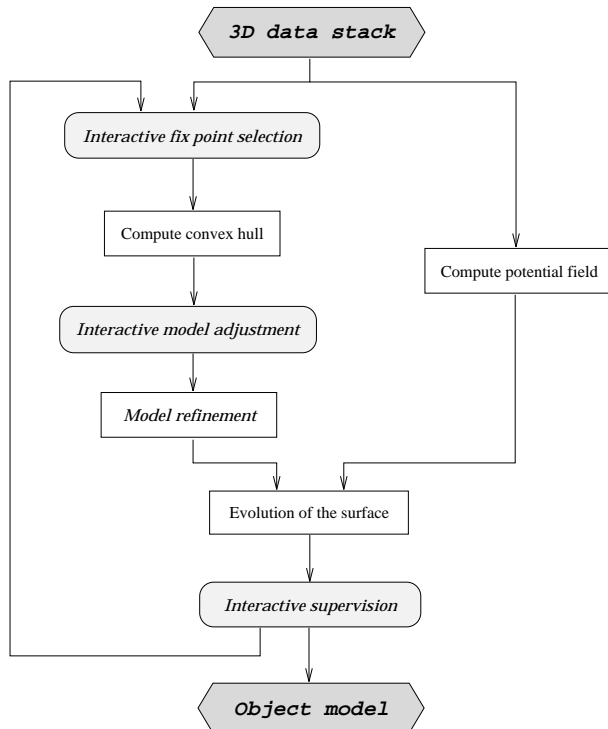


Figure 8: Flow chart illustrating the different parts of the segmentation process. The two interactive steps are light shaded while the pure computation modules have a plain frame. The two hexagons at the top and the bottom denote the input and output respectively.

4 Segmenting medical data

In this section we outline the process of segmenting 3-D medical data. The flowchart in Figure 8 depicts the different steps of the segmentation.

The governing potential computed by convolving the data with 3-D Gaussian derivative filters provides the necessary information about the surface normals at the selected anchor points. The user then provides a small number of anchor-points, which are used to generate the initial approximation of the surface. At the same time they serve as seed points for the force front propagation of the surface evolution: they provide positional and normal information about the final surface, which is fixed and will not be changed during the evolution procedure. It is therefore vital, that they be carefully selected.

The selection of such points is nontrivial in a 3-D data set. Given the fact that the infor-

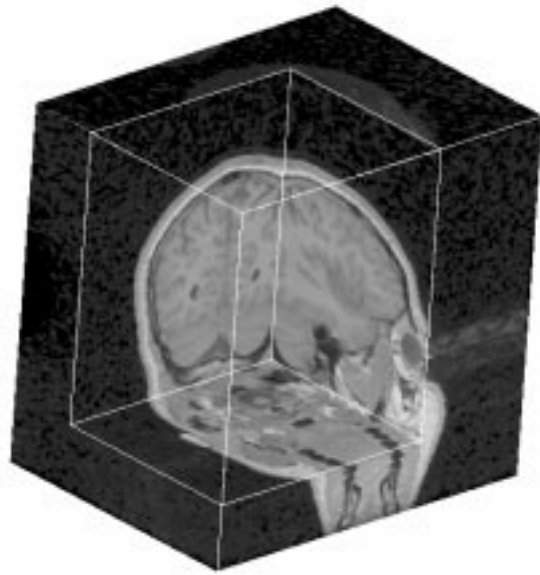


Figure 9: 3-D MR-data of a human brain. The data acquired by the MRI-technique is hard to visualize since it yields an opaque volume of data. The data is a cuboid containing $256 \times 256 \times 123 = 8'060'928$ voxels (volume elements)!

mation which can support the point selection procedure is provided in the form of the original gray-value data or the gradient force field, this selection should happen on 2-D image slices using the MPR tool presented in the Subsection 3.6.

We demonstrate the method on a real MRI acquisition of a human brain, providing dense 3-D data. Hence the “image” which is to be processed is actually a bounded volume of data values. Figure 9 depicts the 3-D data volume where a cuboid has been excavated from the original data.

Our goal is to segment the left lateral ventricle. In order to decrease the computational burden, the data is clipped such that the resulting volume contains the left ventricle only. Figure 10 shows a cut-out of the original data set. The ventricle appears as a gray crescent above the left eye.

We segment the ventricle by using a Velcro Surface. After the anchor points are selected, the convex hull of the point cloud is computed. The Delaunay-tetrahedralization [18] of the

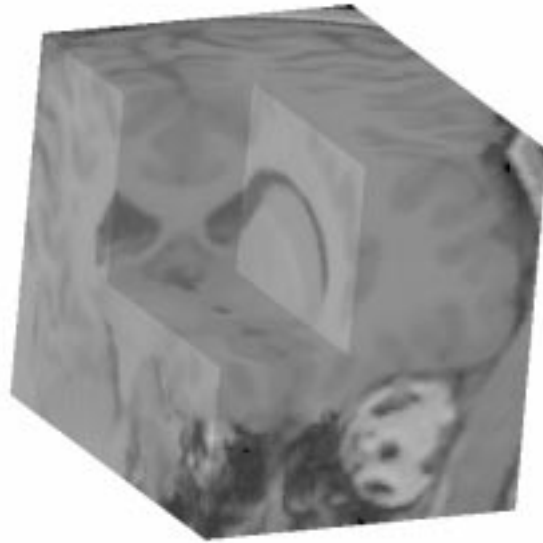


Figure 10: Clipped data set. The cuboid contains now mainly the left part of the patient's brain. The patient is facing towards us. On the front plane a part of his left eye is visible. The gray crescent above the eye is the left lateral ventricle which is clearly visible.

convex hull defined by the 22 user-supplied vertices is displayed in Figure 11a). This presents a very first rough model which has to be refined further by removing unnecessary tetrahedrons from the convex model. This interactive sculpturing process yields the final polyhedron in Figure 11b). The faces of this model have been recursively subdivided yielding a model with 1282 vertices and 2560 facets. Subtracting the 22 seed points, the stiffness matrix has 1'587'600 entries in total. Using a special data representation for sparse matrices, we have to store only the non-zero entries. Since the complete discrete difference operator has a support of 19 vertices on a regular hexagonal mesh, the amount of matrix entries reduces to 23'940. This allows to

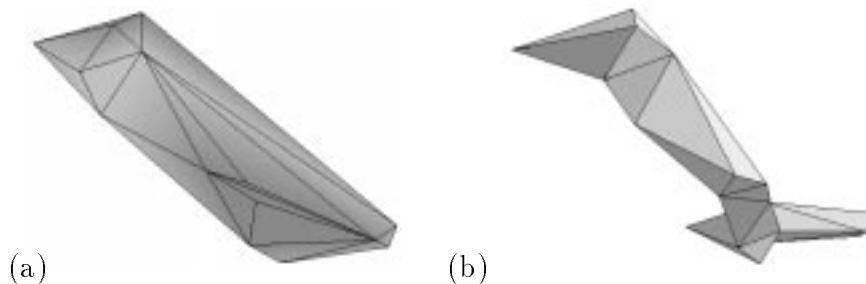


Figure 11: (a) Convex hull model defined by 22 user specified vertices. (b) Refined model after the sculpturing procedure.

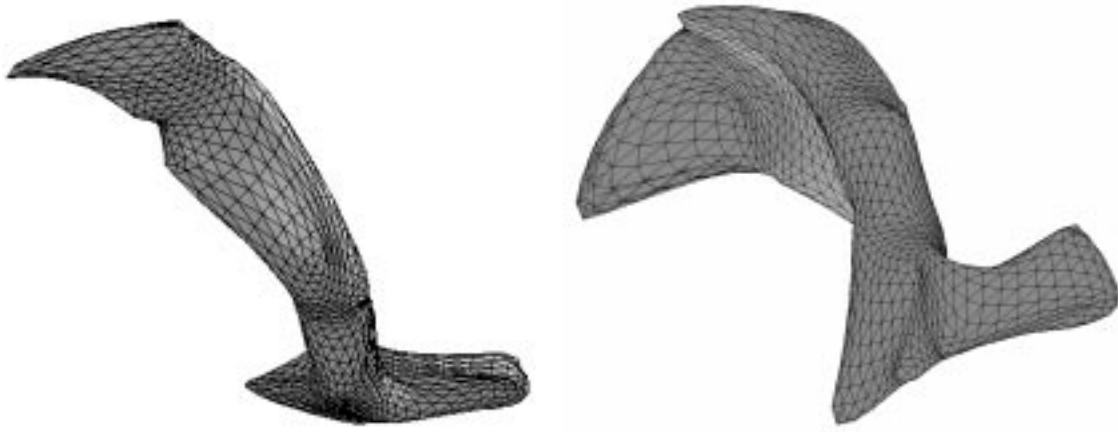


Figure 12: Final result from two different points of view. The faces of the model received after the sculpturing process have been subdivided yielding a model with 1282 vertices and 2560 facets. This refined model was then deformed in accordance to the Velcro optimization procedure and the image data shown in Figure 10. The image shows the final deformed model which outlines the left lateral ventricle. The computation was done for $\tau = 0$.

compute approximately 1 iteration per second on a Sun Sparc workstation.

The initial force-free solution is then perturbed by turning on the external image forces progressively for all the surface nodes, starting from the anchor-points. Figure 12 shows the result of this surface evolution. Due to the chosen refinement level and the value of the parameter $\tau = 0$, the computed model is smooth. The user can govern the surface details he wants to retain during the segmentation on some parts of the surface by selecting anchor-points at those locations.

The resulting segmentation can be supervised by the user. If he is dissatisfied with it—which normally means, that some surface details had been smoothed out—he can add further guiding points, attracting the surface to features not yet considered. The whole process can be then reiterated until a satisfactory segmentation has been obtained.

5 Conclusion and Future Work

We have presented a method that allows initializing and optimizing a 3-D surface model given only a very small number of 3-D seed points and corresponding surface normals. This is valuable in semiautomated applications—such as medical ones—where seed points can be supplied manually by the user with reasonable ease. The user can influence the level of detail in the final representation by choosing the most appropriate force field, adjusting the tension parameter, and by providing more or less seed points for any given part of the surface.

The capability presented here should also be valuable for fully automated applications. Vision algorithms—such as the so-called “shape from X” methods—often provide high-quality results in some parts of a scene but may be unreliable in others. Our method has the potential to allow the use of the most reliable surface patches as anchors and the propagation of information to other parts of the surface. More specifically, we have previously proposed a surface reconstruction method [19] that recovers 3-D surfaces as triangulated meshes by combining monocular shading cues, stereoscopic cues from arbitrary number of images and geometric constraints into an objective function that can be minimized by deforming the mesh. The current implementation relies on the “traditional” snake optimization and we believe that the convergence properties of the algorithm will be greatly enhanced by replacing it by the method presented in this paper. In practice, this will be done by computing the position of some of the mesh facets on an individual basis wherever this can be done reliably. The vertices of these facets will then be used as anchor-points.

When dealing with larger problems and more complex surfaces than the ones presented here it becomes important to prevent the number of mesh vertices to grow so much that the matrix K becomes unmanageable, even using sparse matrix computation techniques. One way to achieve this goal is to replace our quasi-regular tessellations by irregular ones that can have a few large facets in low-curvature areas and more smaller facets in high-curvature ones [20].

It has been shown that this can be done within the deformable model framework by replac-

ing the finite differences estimators by a finite element representation [21, 22]. Using such a representation, the deformation energy E_D of Equation (6) can still be computed as a quadratic form involving a stiffness matrix K . The computation of K would be different but the rest of the approach would remain. Generalizing this approach to handle fully irregular meshes should therefore be relatively straightforward.

Further development of deformable Velcro surfaces will primarily address the implementational and the user interface level. The effort required for interactive initialization of 3-D medical data sets will be reduced further by feeding the preliminary segmentation back to the initialization tool for visual comparison. Insufficient choice of seed points or erroneous assignment to initial facet vertices will thus become apparent and facilitate refinement. Replacing the manual sculpturing by interactively guided heuristic methods will further simplify the initialization process. The governing objective is to develop the mathematical theory, the implementation, and the user interface of a tool for the segmentation of 3D objects without requiring expert Computer-Vision knowledge for its use.

Appendix

In order derive an alternate formulation of the Expression 5 we rewrite both, the elasticity and the rigidity term. The surface's elasticity term then becomes

$$\begin{aligned} \iint_{\Omega} \tau(\omega) \left[\left| \frac{\partial \vec{v}}{\partial \omega_1} \right|^2 + \left| \frac{\partial \vec{v}}{\partial \omega_2} \right|^2 \right] d\omega \\ = \iint_{\Omega} \tau(\omega) \sum_{i=1}^3 [v_{\omega_1}^{(i)2} + v_{\omega_2}^{(i)2}] d\omega = \iint_{\Omega} \tau(\omega) \sum_{i=1}^3 |\nabla v^{(i)}|^2 d\omega . \end{aligned} \quad (\text{A.1})$$

To rewrite the surface's rigidity term we need the expression of the squared Laplacian of $v(\omega_1, \omega_2)$ which is

$$(\Delta v)^2 = (v_{\omega_1 \omega_1} + v_{\omega_2 \omega_2})^2 = v_{\omega_1 \omega_1}^2 + 2v_{\omega_1 \omega_1} \cdot v_{\omega_2 \omega_2} + v_{\omega_2 \omega_2}^2$$

Furthermore, we define the determinant of the Hessian of a function $v(\omega_1, \omega_2)$ to be

$$H(v) = \begin{vmatrix} \frac{\partial^2 v}{\partial \omega_1^2} & \frac{\partial^2 v}{\partial \omega_1 \partial \omega_2} \\ \frac{\partial^2 v}{\partial \omega_2 \partial \omega_1} & \frac{\partial^2 v}{\partial \omega_2^2} \end{vmatrix} = v_{\omega_1 \omega_1} \cdot v_{\omega_2 \omega_2} \Leftrightarrow (v_{\omega_1 \omega_2})^2$$

Therefore the following equality holds for the surface's bending energy

$$\begin{aligned} \iint_{\Omega} \rho(\omega) \left[\left| \frac{\partial^2 \vec{v}}{\partial \omega_1^2} \right|^2 + 2 \left| \frac{\partial^2 \vec{v}}{\partial \omega_1 \partial \omega_2} \right|^2 + \left| \frac{\partial^2 \vec{v}}{\partial \omega_2^2} \right|^2 \right] d\omega \\ = \iint_{\Omega} \rho(\omega) \sum_{i=1}^3 [v_{\omega_1 \omega_1}^{(i)2} + 2v_{\omega_1 \omega_2}^{(i)2} + v_{\omega_2 \omega_2}^{(i)2}] d\omega \quad (\text{A.2}) \\ = \iint_{\Omega} \rho(\omega) \sum_{i=1}^3 (\Delta v^{(i)})^2 \Leftrightarrow 2H(v^{(i)}) d\omega \end{aligned}$$

Using both Equalities (A.1) and (A.2) the term $E_D(\vec{v})$ can be expressed in a form independent of a particular parameterization.

References

- [1] I. Carlbom, D. Terzopoulos, and K. Harris. Computer-Assisted Registration, Segmentation, and 3D Reconstruction from Images of Neuronal Tissue Sections. *IEEE Transactions on Medical Imaging*, 13(2):351–362, 1994.
- [2] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically deformable models. *Computer Graphics*, 21(4):205–214, 1987.
- [3] W. Neuenschwander, P. Fua, G. Székely, and O. Kübler. Making Snakes Converge from Minimal Initialization. In *International Conference on Pattern Recognition*, pages 613–615, October 1994.
- [4] A. Pentland. Automatic Extraction of Deformable Part Models. *International Journal of Computer Vision*, 4(2):107–126, March 1990.

-
- [5] D. Terzopoulos and D. Metaxas. Dynamic 3D Models with Local and Global Deformations: Deformable Superquadrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7):703–714, 1991.
- [6] D. G. Lowe. Fitting Parameterized Three-Dimensional Models to Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(441-450), 1991.
- [7] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active Contour Models. *International Journal of Computer Vision*, 1(4):321–331, 1988.
- [8] I. Cohen, L. D. Cohen, and N. Ayache. Using Deformable Surfaces to Segment 3-D Images and Infer Differential Structures. *Computer Vision, Graphics, and Image Processing: Image Understanding*, 56(2):242–263, September 1992.
- [9] R. Malladi, J.A. Sethian, and B.C. Vemuri. Evolutionary Fronts for Topology-Independent Shape Modeling and Recovery. In *Proceedings of the Forth European Conference on Computer Vision, Stockholm, Sweden*, volume 1, pages 3–13, 1994.
- [10] H. Tek and B.B. Kimia. Shock-Based Reaction-Diffusion Bubbles for Image Segmentation. Technical Report LEMS-138, LEMS, Division of Engineering, Brown University, August 1994.
- [11] Y. Chen and G. Medioni. Surface Descriptions of Complex Objects from Multiple Range Images. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 437–442, Seattle, WA, June 1994.
- [12] P. Fua and Y.G. Leclerc. Object-Centered Surface Reconstruction: Combining Multi-Image Stereo and Shading. *International Journal of Computer Vision*, 1994. Accepted for publication, available as Tech Note 535, Artificial Intelligence Center, SRI International.
- [13] D. Terzopoulos. On matching deformable models to images. *Topical Meeting on Machine Vision Tech. Digest Series*, 12:160–167, 1987.

-
- [14] R. Courant and D. Hilbert. *Methods of mathematical physics*, volume 1. Wiley: New York, 1989.
- [15] C. Grossman and H.-G. Roos. *Numerik partieller Differentialgleichungen*. B.G. Teubner, 1992.
- [16] C. Pommerell and W. Fichtner. PILS: An iterative linear solver package for ill-conditioned systems. In *Supercomputing '91*, pages 588–599, November 1991.
- [17] K.S. Kunder and A. Sangiovanni-Vincentelli. Sparse User's Guide, A Sparse Linear Equation Solver. Technical report, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, April 1988.
- [18] J.-D. Boissonnat. Geometric structures for three-dimensional shape representation. *ACM Transactions on Graphics*, 3:266–286, October 1984.
- [19] P. Fua and Y.G. Leclerc. Using 3-Dimensional Meshes To Combine Image-Based and Geometry-Based Constraints. In *European Conference on Computer Vision*, pages 281–291, Stockholm, Sweden, May 1994.
- [20] W.C. Huang and D.B. Goldgof. Adaptive-Size Meshes for Rigid and Nonrigid Shape Analysis and Synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):611–616, June 1993.
- [21] T. McInerney and D. Terzopoulos. A finite element model for 3D shape reconstruction and nonrigid motion tracking. In *Proceedings of the Fourth International Conference on Computer Vision, Berlin, Germany*, pages 518–523, May 1993.
- [22] E. Koh, D. Metaxas, and N. Badler. Hierarchical Shape Representation Using Locally Adaptive Finite Elements. In *European Conference on Computer Vision*, Stockholm, Sweden, May 1994.