technical report EE–96–08-02

Volumetric Deformable Models: Active Blobs

Ross T. Whitaker
Department of Electrical Engineering
University of Tennessee Knoxville, TN 37996-2100

Appears in: Visualization In Biomedical Computing, October 4–7, 1994, Rochester, Minesota (Editor Richard A. Robb.) pp. 122–134

Imaging Robotics and Intelligent Systems Laboratory
Departmentment of Electrical Engineering
University of Tennessee

Abstract

Current modeling and segmentation techniques are not adequate for visualizing certain kinds of 3D medical data. This paper introduces volumetric deformable models, or active blobs, as a means of visualizing and segmenting 3D data. It presents an implicit framework that provides a means of generalizing deformable models to higher dimensions. It presents a new second-order smoothing function that offers some advantages over previous methods for smoothing surfaces. It describes some of the numerical techniques that are necessary for solving the resulting evolution equations. Finally it shows some examples of how active blobs are useful for the segmentation and visualization of 3D ultrasound and MRI data. Active blobs have a number of advantages over other types of modeling techniques. Because of these advantages, active blobs are able to produce models directly from the 3D data, rather than presegmented datasets, edge maps, or sets of landmarks.

1 Introduction

1.1 Motivation

The problem of volume visualization is inevitably linked to segmentation. Conventional visualization techniques fall into two categories. The first category of techniques consists of the so-called "direct" visualization techniques that apply lighting and shading models directly to the volume data [1]. Direct volume visualization techniques are inadequate on two accounts. These techniques rely on local (defined according to some scale) image structure to create the appropriate lighting and shading models. Therefore they typically produce poor results when applied to data that is either noisy or undersampled. Also direct visualization techniques by themselves do not address the issue of relevant anatomy. When attempting to visualize anatomy that is occluded or enclosed in other structures, direct visualization techniques must be combined with other segmentation techniques that incorporate regions of interest [2].

The second category includes model-based visualization techniques. These techniques apply conventional rendering methods to geometric models that are somehow constructed from the volume data. The challenge for model-based techniques is the construction of a geometric model that accurately characterizes the relevant anatomy in the 3D data. Given a segmentation of the data, the construction of a geometric model is relatively straightforward; the difficulty is the segmentation. When visualizing 3D medical data, segmentations are often constructed by hand using contours that are drawn one "slice" at a time. This is a labor-intensive solution that is too slow and expensive to be routine.

This work presents a new type of deformable model which is capable of segmenting volume data given an initial estimate (which can be generated automatically from the data in some cases) and the appropriate tuning of several parameters which control the smoothness and the locality of the solutions. Unlike much of the previous work on 3D deformable models, this technique does not require sets of edges or hand-drawn contours, rather it works directly on the 3D data.

1.2 Deformable models

The fitting of models to data is a fundamental problem in computer vision. Work in computer vision and elsewhere has shown the benefits of using models to improve image segmentation. *Deformable* models are models whose shapes depend on a set of parameters. These parameters are tuned (usually through some minimization process) to change the shape of the model so that it matches some properties derived from the image data. Deformable models typically combine the external data with some internal forces or constraints. These constraints might be (as in this work) enforced in a graded or "fuzzy" fashion. Deformable models find a compromise between the constraints and the shapes indicated by the input data. In this way deformable models resemble the Bayesian approach to image analysis, and the internal constraints of these models are like the prior distributions that drive a MAP estimation [3]. Deformable models are distinguished according to three different properties. These properties are the parameterization, the internal constraints, and the mechanism for reconciling the model and data.

The first distinguishing property of active blobs is the parameterization. A geometric model is essentially a set of numbers that gives rise to a shape. The parameterization is the relationship between those numbers and the shapes which they generate. Perturbations in the parameters generate deformations in the shape, so the relationship between the parameters and the shape of the model plays an important role in the deformation of models. For example, superquadrics [4] generate a range of shapes, from ellipsoids to cube-like objects, with a small number of parameters. These parameters can be tuned to create a shape that resembles some aspect of an image, hereafter referred to as the input data. A variety of other parameterizations for deformable models have been explored, including meshes [5, 6], discrete particle systems [7], and cylinders [8].

Active blobs incorporate an implicit, rather than explicit, representation of shape. Models are characterized by level sets of dense scalar fields. Parameterizations of such dense scalar fields are easy to manipulate. When these scalar fields are sampled on discrete, regular, rectilinear grids, the implicit models can be represented by greyscale digital images.

The second property of deformable models is the internal constraints that different models enforce. The fitting of a deformable model to the input data incorporates some compromise between the input data and the internal constraints of the model. The constraints are often linked to the parameterization. For example when matching 3D models to brain images, one might use a *prototypical* brain model which can undergo a fixed set of distortions such as rotations, translations, and scaling. This is a somewhat "rigid" or "hard" model because its deformations are highly constrained. Such approaches can work in some circumstances but they make some very strong assumptions about the input data (e.g. that they contain a rotated, scaled, etc. version of the prototypical brain). Grenander et al. [9] have developed models which "deduce" the appropriate deformations through a set of examples (a training set). These models undergo deformations that are consistent with the statistical variations in the training data. Cootes et al. [10] have applied a similar approach to 3D medical data. Such approaches are effective for segmentation problems that can be characterized by training sets but are inappropriate in situations where less is known about the input data a priori. In the absence of specific training data, models must incorporate a more general set of constraints that applies to a wider range of circumstances.

A number of the more general modeling approaches introduce internal constraints implicitly via the parameterization. For instance when using polynomial representations, the degree of the polynomial one uses determines the kinds of shapes the model can take. Polynomials of second and third degree can represent only very simple shapes; higher-degree polynomials allow more complexity and thereby enable the model to represent "finer scale" structures. The decision to limit the model to a particular degree of polynomial is a kind of constraint. The same decision applies to Fourier representations (or spherical harmonics in 3D) [11]; the truncation of the frequency representation introduces a smoothness constraint in the model. Constraints of this type are typically sensitive to extrinsic decisions about the construction of the parameterization.

The active blob models presented here incorporate constraints that depend only on the intrinsic shape of the model. Thus the behavior of the model is less dependent on a particular parameterization. This is desirable because the parameterization of the model is often a matter of mathematical necessity or convenience. The internal constraints associated with active blobs are intended to be general, and thus the models are *soft* and able to take on a variety of different shapes while maintaining an internal smoothness constraint that is enforced by a second-order flow.

A third distinguishing property of deformable models is the means by which models and data are reconciled. Often the deformation of models is cast in an energy minimization framework. The problem of fitting the model to the input data is posed as a minimization of an energy functional that combines the internal constraints and the input data. Some algorithms seek a global minimization (or maximization) while other techniques seek out local minima in the vicinity of a set of initial conditions. The volumetric models described here provide a natural scale space which allows the energy minimization problem to be solved in a multiscale fashion. This multiscale approach resembles minimization by graduated nonconvexity. Instead of guaranteeing a minimum solution, it allows one to control the search space based on spatial proximity.

Malladi et al. have presented related work, which involves the embedding of evolving curves for segmentation. In [12], they propose a "seed" which to grows (or shrinks) to fill an area of interest. This "flooding" is constrained by a curvature term as well as the presence of edges in the image. The active blob models do not grow or shrink on their own, but are pulled in the direction indicated by the input data.

2 Active surfaces

Kass et al. [13] propose active contours or "snakes", which is a somewhat general paradigm for deformable models. Snakes are parameterized curves with a set of internal constraints that enforce rigidity and smoothness. This paradigm and its many variations have proven useful for certain kinds of 2D segmentation problems. Rather than review the 2D results, I describe the natural generalization of this approach to 3D and explain some of the problems presented by this generalization.

A surface is a two-parameter object in a three-dimensional space, i.e., a surface $\vec{S}$ is

$$\vec{S} : \underset{r}{U} \times \underset{s}{U} \mapsto \underset{x,y,z}{V}, \tag{2.1}$$

where $V \subset \mathbb{R}^3$, and $U \subset \mathbb{R}$. The 3D energy terms are similar to the 2D energy terms, except that for active surfaces I consider only the membrane term, which produces a second-order flow, and I exclude the bending energy (rigidity), which produces a fourth-order flow [13]. The internal constraints of an active surface incorporate both of the free parameters, r and s:

$$\mathrm{d}E_{\mathrm{mem}} = \vec{S}_{ss} + \vec{S}_{rr} \tag{2.2}$$

$$\mathrm{d}E_{\mathrm{image}} \quad = \quad -\nabla f(\vec{S}), \tag{2.3}$$

where $\mathrm{d}E$ indicates that these terms are derived from the first variation of the corresponding energy terms. The membrane energy E_{mem} causes the surface to contract, and E_{image} is the influence of the input data on the model. The function $f : \mathbb{R}^3 \mapsto \mathbb{R}$ is a scalar field defined over the range of the model which indicates the presence of interesting features. Note that E_{image} could take other forms, such as a set of projections, or a set of range maps, but such possibilities are beyond the scope of this paper. A gradient descent algorithm yields an evolution equation for the surface:

$$\frac{\partial \vec{S}}{\partial t} = \alpha \mathrm{d}E_{\mathrm{mem}} + \beta \mathrm{d}E_{\mathrm{image}}, \tag{2.4}$$

where α and β are free parameters that control the relative influence of the internal constraints and the input data.

Equations 2.2–2.4 describe the evolution of a model in terms of its parameterization. This formulation has some substantial limitations. As stated earlier, the precise behavior of the model depends on the particular parameterization. Often surface parameterizations are limited by topology. For instance the same parameterization that represents a closed curve in R^2 cannot, generally, represent two closed curves in a continuous fashion. In three dimensions a parameterization of a sphere is generally not compatible with a torus. For active surfaces such parameterizations would preclude the possibility of a smooth evolution from a sphere to a torus. As models evolve and undergo large deviations from their original shapes, surface parameterizations often introduce de facto constraints. If these constraints are not desirable, then the model must be reparameterized according to set of heuristics or according to some cost function [14]. For instance the expansion of polygonal models creates a kind of "coarseness" which prevents the model from capturing smaller structures; thus the evolution of polygonal models requires the creation and deletion of polygons [5].

The surface evolution described by Eqs. 2.2 and 2.4 is intended to motivate the development of a new kind of deformable model. The particular energy terms $\mathrm{d}E_{\mathrm{mem}}$ and $\mathrm{d}E_{\mathrm{image}}$ are not essential to the work discussed here. The membrane energy, for instance, will be replaced in Sect. 4 by a more appropriate second-order flow.

3 Embedding active surfaces

Manifolds such as curves and surfaces have certain shape properties that are independent of any particular parameterization. There is an *intrinsic geometry* to such objects which depends only on their shape in the domain. In order to avoid some of the drawbacks of parameterized models, I forego the parameterization and express surfaces implicitly by embedding them as level sets of scalar functions. This strategy removes the parameterization from the model, leaving only the intrinsic geometry. The strategy for embedding active contours consists of four steps.

1. Express the equations of motion for a deformable model in terms of some unspecified parameterization (as was done in Sect. 2).

2. Describe the parameterization in terms of the *intrinsic* local geometry of the model.

3. Assume the model is the level set of a function F.

4. Express the geometry of the level set in terms of the differential structure of F, and create an evolution equation for F.

In order apply this strategy to active surfaces, represent a surface as a level set of some 3D scalar function, $F : \mathbb{R}^3 \times \mathbb{R}^+ \mapsto \mathbb{R}$, which evolves over time. The evolution equation of the individual level surfaces specifies a corresponding evolution equation for the scalar function $F(x, y, z, t)$, which is a dense set of volume data. Using the procedure described in Appendix A, obtain an evolution equation that is the embedding of Eq. 2.4:

$$\frac{\partial F}{\partial t} = \alpha \left(F_{xx} + F_{yy} + F_{zz} - F_{ww}\right) - \beta \nabla F \cdot \nabla f, \tag{3.1}$$

where w indicates derivatives in the normal direction, i.e., $F_w = |\nabla F|$.

Equation 3.1 is invariant under certain kinds of geometric transformations. First, it is invariant to orthogonal group transformations on x, y, z. Thus the position and orientation of the model in space has no impact (to within the error introduced by the parameterization of F) on the behavior of the model. Equation 3.1 is also invariant to monotonic transformations on F; that is, Eq. 3.1 acts *only on level sets* and treats each level set of F as an individual surface evolving under its own set of constraints and forces.

Another important property of Eq. 3.1 is that it is (at any particular instant in time) equivalent to an active surface with an explicit parameterization r, s, in which r and s are given in units of arc length and have perpendicular tangents. That is,

$$|S_r| = |S_s| = 1 \quad \text{and} \quad S_r \cdot S_s = 0, \tag{3.2}$$

which I will call an orthonormal parameterization. Thus the evolution of the implicit surfaces in Eq. 3.1 is equivalent to a parametric model if one performs a *reparameterization locally at each time step.* [1]

4 The smoothing of surfaces

The first term of Eq. 3.1 (the α term) is the *mean curvature* of the level set at every point on the level sets of F. This term was developed from a straightforward generalization of a similar energy term for contours in 2D [13]. In the absence of external forces (either $\beta = 0$ or $\nabla f = 0$), Eq. 3.1 describes a mean curvature flow on the level sets of F. The evolution of surfaces under mean curvature flow has been studied quite extensively [15]. Although this type of evolution can produce a kind of "smoothing" on surfaces in some cases, in many other cases it is not a smoothing at all. For instance mean curvature flow can break relatively smooth objects into a number of small, high curvature pieces.

As an alternative to mean curvature flow, I propose an evolution term which consists of the weighted sum of principle curvatures at every point in the surface. Let ω_1 and ω_2 be the principle curvatures of a surface $\vec{S}$ with an orthonormal parameterization and let Ω be the matrix of second-order derivatives of the surface with respect to any two orthogonal directions in the tangent plane, i.e.

$$\Omega = \begin{bmatrix} \vec{S}_{ss} & \vec{S}_{sr} \\ \vec{S}_{rs} & \vec{S}_{rr} \end{bmatrix} = R \begin{bmatrix} \omega_1 & 0 \\ 0 & \omega_2 \end{bmatrix}, \tag{4.1}$$

where R is a rotation matrix that indicates the directions associated with the principle curvatures, ω_1 and ω_2. The mean curvature of a surface is one half the trace of Ω: $(\omega_1 + \omega_2)/2$.

For a curvature flow which smoothes surfaces, compute a weighted sum of the principle curvatures ω_1 and ω_2 which implements an averaging *along the direction of least curvature*. This is the *weighted curvature*, κ_w:

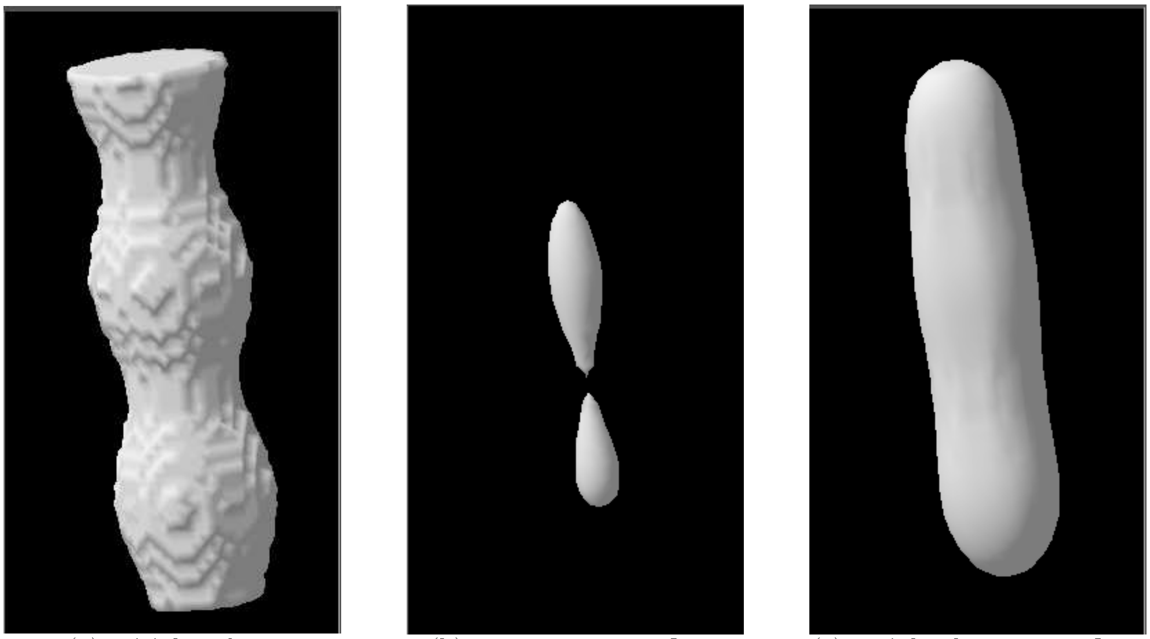
$$\kappa_w = \frac{\omega_1\omega_2^2 + \omega_2\omega_1^2}{\omega_1^2 + \omega_2^2} = \frac{(\omega_1 + \omega_2)\omega_1\omega_2}{\omega_1^2 + \omega_2^2} = \frac{\mathrm{Tr}[\Omega]\,\mathrm{Det}[\Omega]}{||\Omega||^2},$$

where $||\cdot||$ indicates the Euclidean norm. The weighted curvature κ_w is invariant to rotations on Ω, and it can be calculated explicitly from the embedding of F (see Appendix B) without any explicit calculation of the directions or curvatures in the tangent plane of the level set. Figure 4.1 shows an initial surface which breaks into pieces under mean curvature flow but forms a cylindrical shape (and eventually a sphere) under weighted curvature flow.

5 Numerical issues

The previous section describes the continuous mathematics for embedding active surfaces, but it does not address the practical problems of solving these equations in the discrete domain. In general the scalar function F is not a continuous mapping from the image space, $V \subset \mathbb{R}^3$, to the real numbers, but rather it

[1] Equation 3.2 does not specify r and s uniquely, but the evolution equation, Eq. 3.1, is invariant under transformations that describe the set of parameterizations defined by Eq. 3.2



(a) Initial surface (b) Mean curvature flow (c) Weighted curvature flow

Figure 4.1: Surface renderings show (a) a dumbbell shaped object, which breaks (b) into pieces under mean curvature flow (after $t = 12.5$ pixel-squared units), but which remains a single, smooth object (c) under weighted curvature flow after the same amount of time.

has some discrete representation in a digital computer. There are many options for representing $F : V \mapsto \mathbb{R}$; for this work I represent F as a discretely-sampled image that is sampled on a finite, Cartesian grid. That is, F is a collection of *voxels* in the form of a 3D digital image. The strategy of embedding active contours does not depend on this particular representation. I have chosen this representation because it provides a mechanism for performing numerical differentiation at multiple scales [16, 17]. There are several important numerical issues regarding this discrete representation of F.

5.1 Discrete derivatives

Derivatives of F are measured using finite differences with the "tightest fitting" kernels of a particular order [17, 18]. All derivatives are measured in the directions x, y and z that are associated with the discrete grid. The derivatives in local Gauge coordinates are computed from these discrete Cartesian derivatives. Directional derivatives are calculated using a dot product with a unit vector in the desired direction.

5.2 Stability

The evolution equations for the embedded contours have the form of a Hamilton-Jacobi equation, i.e.,

$$\frac{\partial F(\vec{x}, t)}{\partial t} = G(\vec{x}, t)|\nabla F(\vec{x}, t)|, \tag{5.1}$$

where $\vec{x} = x, y, z$. Discrete solutions using finite forward differences in time are generally not stable. When implemented on a discrete grid using finite forward differences in time, i.e.,

$$F(\vec{x}, t + \Delta t) = F(\vec{x}, t) + \frac{\partial F(\vec{x}, t)}{\partial t}\Delta t, \tag{5.2}$$

such equations can "overshoot" [19, 20] near sharp edges causing ringing and instability.

Stable to solutions of Eq. 5.1 can be computed by using a *viscosity* approximation. There are several approaches to computing viscosity approximations. Osher and Sethian [20], for instance, propose an "up wind" scheme which incorporates piecewise continuous approximations to F and utilizes "one-sided" (or up-wind) derivatives in approximations to $|\nabla F|$. Instead of an up-wind scheme I use a second-order ("diffusion") term in order to stable solutions to Eq. 5.1, i.e.,

$$F(\vec{x}, t + \Delta t) = F(\vec{x}, t) + \left[G(x, y, t)|\nabla F(\vec{x}, t)| + \frac{1}{2}|G(x, y, t)| F_{ww}(\vec{x}, t) \right] \Delta t. \tag{5.3}$$

For functions of one dimension, this formulation is identical to the first-order up-wind scheme proposed by [20] except at extrema. However the formulation I use here generalizes to higher dimensions in a rotationally invariant manner, whereas the up-wind approaches typically assume that the velocity of a moving wave front is in one of the cardinal directions (i.e. aligned with the grid on which F is sampled). Solutions to 5.3 require some specification of the boundary conditions on the image space V. For the results in this paper I use $F(\vec{x})_{NN} = 0 \ \forall \vec{x} \in \partial V$, where ∂V is the boundary of the image space and N is the direction normal to the boundary. These boundary conditions enable level sets to move across the boundaries unimpeded.

This second-order viscosity scheme requires time steps Δt that are inversely proportional to the velocity of the fastest moving wave front. That is,

$$\Delta t \leq \frac{1}{\sup_{x \in V} \{|G(\vec{x}, t)|\}}, \tag{5.4}$$

which is required in order to maintain the stability of the viscosity term, F_{ww}. The mean and weighted curvature flows are stable when using forward differences in time and centralized differences in space. Therefore, the $|G(\vec{x}, t)|$ term used for the viscosity solution includes only the influence of the image energy dE_{image}.

5.3 Scale continuation

Two of the disadvantages of working with volumetric models are the relatively large amount of computation that is necessary for a single iteration and the limited speed of moving wavefronts, which is imposed by the stability requirement. Parameterized models typically provide some relatively small set of control points or parameters, but embedded models require calculations on dense sets in the range of the model. This computational burden is partially offset by the natural scale space that exists for greyscale images [21]. Solutions can be sought on some coarse grid, where greater wavefront speeds are permitted, and then extended to progressively finer grids in order to fill in the details that are lost at larger scales (coarse grids). This strategy is depicted in Fig. 5.1.

Besides providing some computational advantages, this coarse-to-fine scale strategy allows image features to attract models from a distance. Normally the deformation of a model under gradient descent results in minima that are local (in the space of deformations). This means that models can get "hung up" on smaller, less interesting features because they are "closer" to the initial conditions even though these features may be less important than other nearby image structures. Alternatively a global minimization may also be unsatisfactory, because one may want results that are sensitive to the initial conditions, i.e., the placement of the initial model can serve as a means of specifying interesting structure, as will be shown in Sect. 6.2). One of the findings of this research is that a multiscale approach provides a means of specifying the *locality* of solution. If the evolution process begins at a very coarse scale, the results are virtually independent of the initial conditions. If the evolution process begins at a small scale, the results are highly sensitive to the placement of the initial model.

6 Results

6.1 Visualization of 3D ultrasound data

Three-dimensional ultrasound data presents a unique challenge for volume visualization because of the high level of noise as well as specularities and echo dropout. The following results are from a 3D dataset that was scanned using a custom rotating phased array transducer which is designed to sweep out a volume. Before

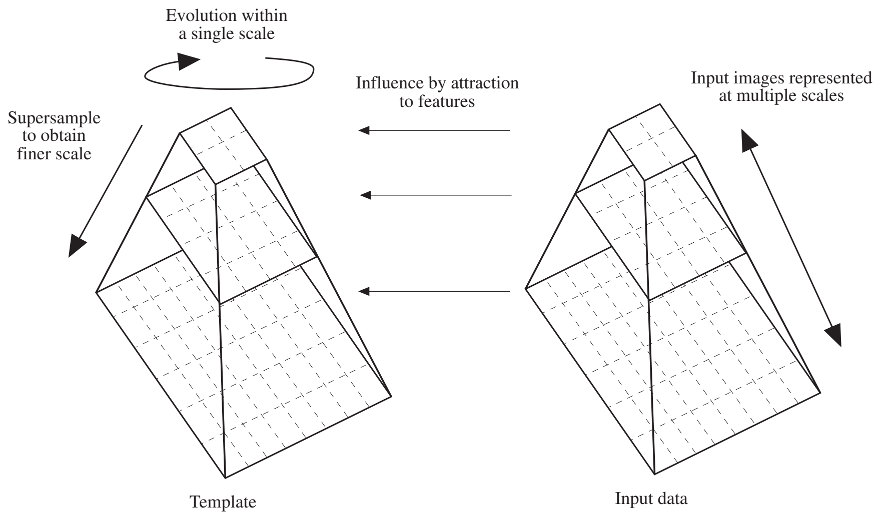


Figure 5.1: A multiscale approach to solving the evolution equation.

applying the active blob technique, the data was subsampled by a factor of 2 (making a $128 \times 64 \times 96$ volume) in order to make the results more manageable for storage and visualization. Also before processing, a large obstruction was removed from directly in front of the face of the fetus. This edit was made with a single cutting plane that was placed manually with an image editor.

Figure 6.1 shows an isosurface rendering for both the original data and a deformed model. The initial model is a blurred and thresholded version of the input data. The image on the right shows that the deformable blob appears to contain less noise than the original and yet retains much of the shape of the anatomy. The lips and nose are distorted by the noise in the input data but appear quite clearly in the model.

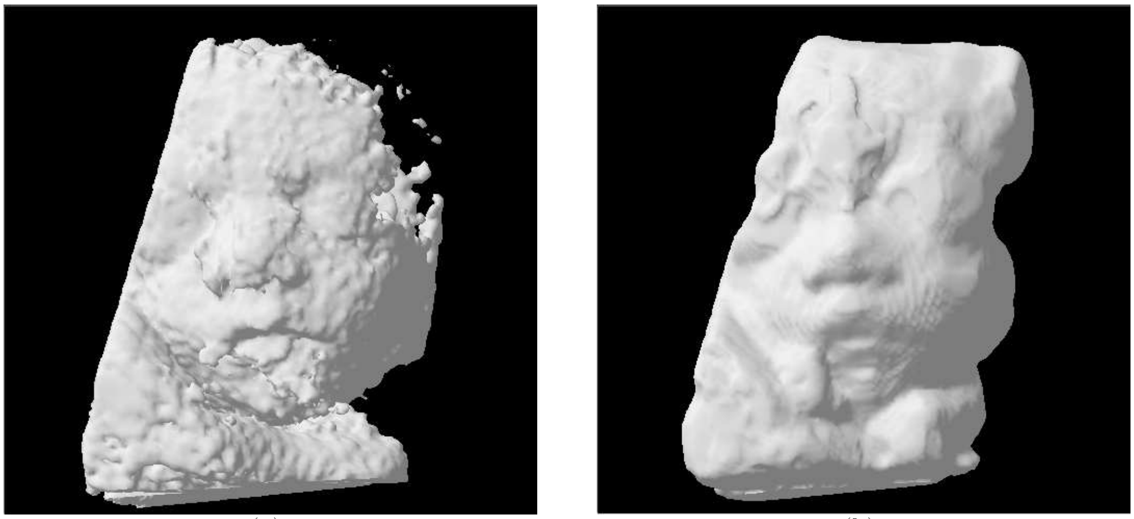
6.2 Interpolation and segmentation of MRI

THe MRI data set of a human head in Fig. 6.2 consists of 22 slices. The sampling in the z (transversal) direction is less dense than the x and y directions. In this example the active blob models are used to construct models of both the head and the tumor. These models are produced with a 3 to 1 supersampling in the z direction and a 2 to 1 subsampling in the x and y directions. This sampling is used because it produces models that resemble more closely the proportions of the actual anatomy.

Figure 6.3(a) shows the result of a linear interpolation of the original slices and a threshold of the resulting data. Notice the artifacts that result from the interpolation as well as the noise around the eyes and nose. Figure 6.3(b) shows the initial models that were used to do the segmentation of the head and tumor. Figures 6.3(c)-(f) show the results of the models. On slices which coincide with the original data, the models are attracted to edges, as indicated by high gradient magnitude in the input data (the z derivative is ignored in this edge calculation). Between slices the model follows the internal constraints described by weighted-curvature flow. The result is an interpolation based on 3D shape in conjunction with a segmentation. The weighted curvature flow provides a smoothing which does not destroy important structures like the nose, eyes and ears.

7 Conclusions

Active blobs are volumetric deformable models that treat each level set of 3D function as an active surface. The volumetric representation has several important practical implications for 3D medical data. First, active blobs are topologically flexible, that is, the models can "split" into pieces to form multiple objects



(a) (b)

Figure 6.1: An isosurface rendering of 3D ultrasound data (a) shows the problems of noise and incomplete data. A rendering of a deformable blob model (b) which is attracted to areas of high gradient magnitude shows more clearly the features of the fetus.

[22]. Second, rather than a single surface model, active blobs can represent any number of surfaces in a 3D image (they must be closed and cannot cross each other). In this way the models can represent uncertainty. The result of this evolution equation is not a single contour, but a *contour density* at each point in the image space. This is important for volume visualization, because the models can be displayed by some method, such as volume rendering, which captures that uncertainty. The evolution of the embedding F is a differential expression that is invariant to orthogonal group transformations (rotations and translations). The shapes formed by the level sets of F are restricted only by the spacing of the pixel grid used to represent F. This grid can be made with finer spacing than the input data, as was shown in Sect. 6.2. Finally, the representation of deformable models in terms of an *image* provides a method for multi-scale analysis. The analysis starts on a coarse grid and then proceeds to progressively finer grids as the energy reaches a minimum. This reduces computation time and controls the relative importance of large and small scale structures in the input image.

8 Acknowledgments

Thanks to the University of North Carolina Department of Computer Science for providing computing resources to assist this work. The MRI dataset in Fig. 6.2 is from The University of North Carolina Hospitals, Department of Radiation Oncology. Tomographic Technologies Incorporated provided the ultrasound data used in Fig. 6.1. This work is supported by Bull SA, ICL Plc, and Siemens AG.

Appendix

A The implicit formulation of level surfaces

Represent an active surface $\vec{S}(r,s,t)$ as a level set of a scalar function, $F(x,y,z,t) : \mathbb{R}^3 \times \mathbb{R}^+ \mapsto \mathbb{R}$, i.e.,

$$F\left(\vec{S}(r,s,t),t\right) = k. \tag{A.1}$$

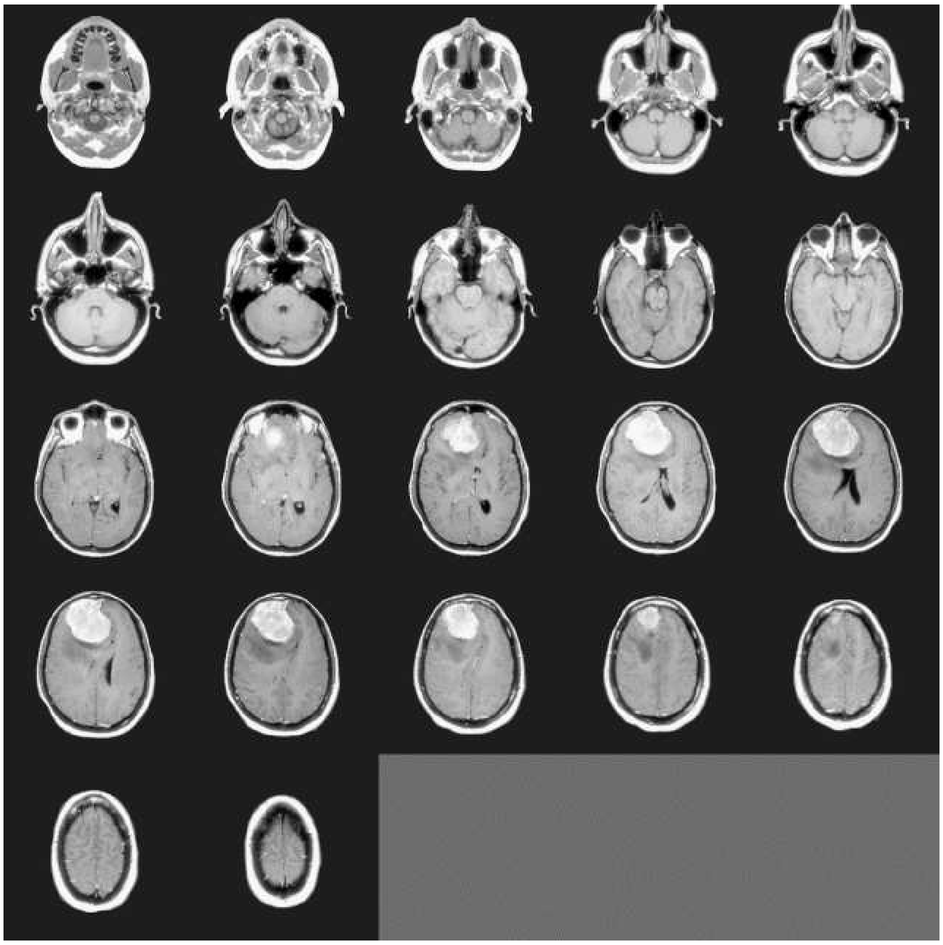


Figure 6.2: The MRI data set contains 22 slices. The voxel resolution in the x, y, and z directions are 0.090278, 0.090278, and 0.65 centimeters respectively.

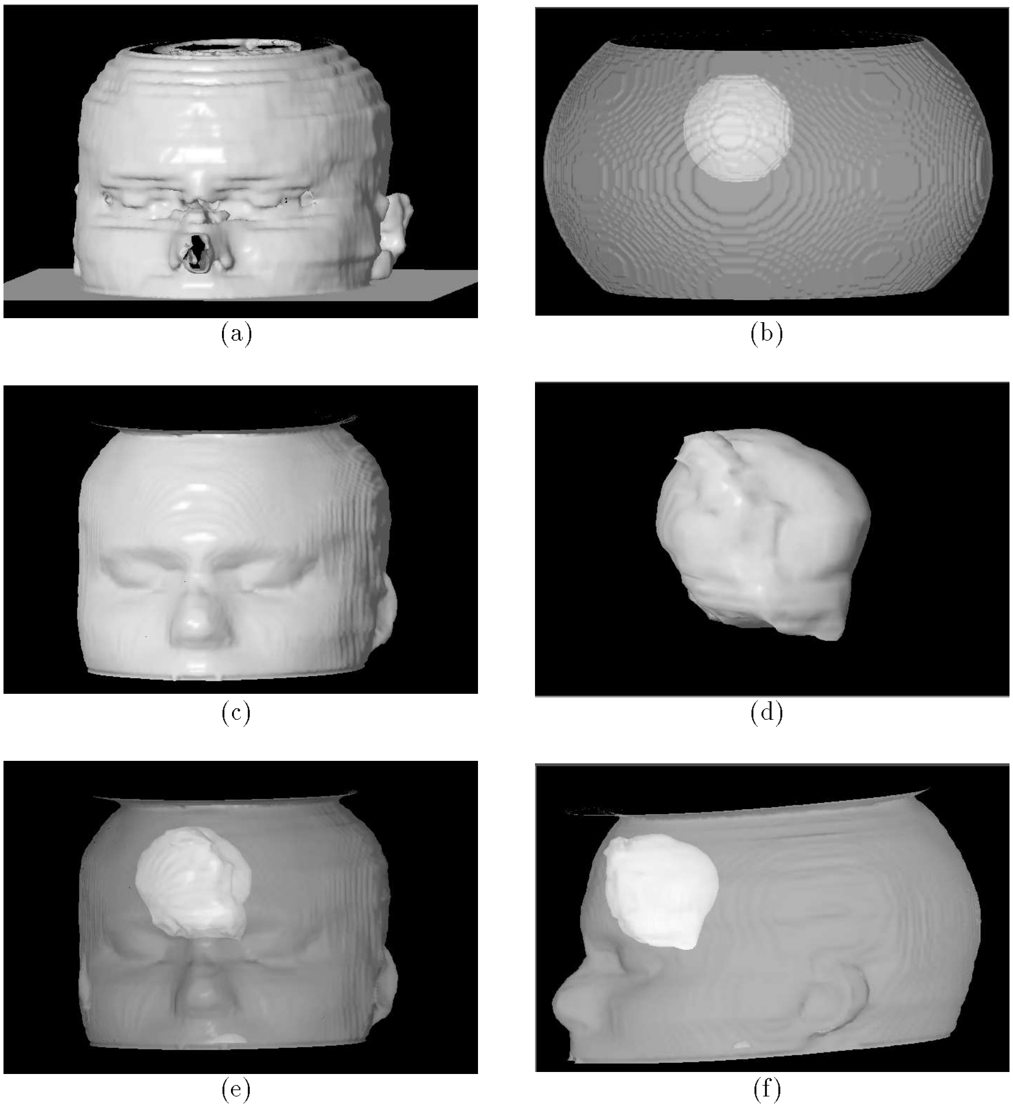


Figure 6.3: From the MRI data of Fig. 6.2: (a) an isosurface rendering of a 3 to 1 linear interpolation of the original data, (b) the ellipsoidal models that were used as the initial conditions for segmenting the head and tumor, (c) the result of the head model after 40 iterations performed at each of 3 different scales, (d) the resulting tumor model which was produced with the same parameters (α and β) as the head model but with a different initial model. Transparent renderings (e)–(f) of both the head and tumor models show the relative positions of the two.

The surface $\vec{S}$ remains a level set of F over time, so the time derivative is zero:

$$0 = \frac{\partial F(\vec{S},t)}{\partial t} + \nabla F(\vec{S},t) \cdot \frac{\partial \vec{S}}{\partial t}, \tag{A.2}$$

where

$$\nabla F(x,y,z) \stackrel{\text{def}}{=} \frac{\partial F}{\partial x}, \frac{\partial F}{\partial y} \frac{\partial F}{\partial z}. \tag{A.3}$$

Thus,

$$\frac{\partial F}{\partial t} = -\nabla F \cdot \frac{\partial \vec{S}}{\partial t} = -|\nabla F| \frac{\partial \vec{S}}{\partial t} \cdot \vec{N}, \tag{A.4}$$

where $\vec{N}$ is the normal to the level surface.

The evolution of $\vec{S}(r,s)$ (Eq. 2.4) is expressed in terms of derivatives of F under the assumption of a orthonormal parameterization, $|\vec{S}_r| = |\vec{S}_s| = 1$ and $\vec{S}_r \cdot \vec{S}_s = 0$:

$$\vec{S}_{rr} \cdot \vec{N} = -\frac{F_{uu}}{F_w}, \; \vec{S}_{ss} \cdot N = -\frac{F_{vv}}{F_w}, \quad \text{and} \tag{A.5}$$

$$\nabla f \cdot N = \nabla f \cdot \frac{\nabla F}{|\nabla F|}, \tag{A.6}$$

where the subscripts in u and v represent derivatives with respect the directions S_r and S_s respectively.

Multiplying by $-|\nabla F|$ and adding the energy terms gives

$$\frac{\partial F}{\partial t} = \alpha \left(F_{uu} + F_{vv} \right) - \beta \left(\nabla F \cdot \nabla f \right). \tag{A.7}$$

Note that $F_{uu} + F_{vv}$ is twice the mean curvature of the level set which is invariant to rotations (within the tangent plane) of the u-v coordinate system.

B Numerical scheme for the second-order geometry of level sets

The weighted curvature term described in Section 4 requires the calculation of three separate invariant expressions of the second-order structure of the level sets of F. In this section I describe a method for calculating the invariant second-order properties of the level sets of 3D without any explicit representation of u and v, i.e., these invariants can be computed directly from the first- and second-order differential structure of F.

The second-order structure of F is given by the Hessian:

$$\mathrm{D}^2 F = H = \begin{bmatrix} F_{xx} & F_{xy} & F_{xz} \\ F_{yx} & F_{yy} & F_{yz} \\ F_{zx} & F_{zy} & F_{zz} \end{bmatrix} = R \begin{bmatrix} F_{uu} & F_{uv} & F_{uw} \\ F_{vu} & F_{vv} & F_{vw} \\ F_{wu} & F_{wv} & F_{ww} \end{bmatrix} = RW, \tag{B.1}$$

where R is a 3×3 rotation matrix that aligns the w direction with the gradient of F. The directions u and v lie in the tangent plane to the level set of F. The trace, determinant, and Euclidean norm of $\mathrm{D}^2 F$ are invariant to rotations. Thus,

$$\mathrm{Tr}[H] = \mathrm{Tr}[W], \quad \mathrm{Det}[H] = \mathrm{Det}[W], \quad \text{and} \quad ||H|| = ||W||. \tag{B.2}$$

This relationship is important because H can be computed by direct differentiation of F with respect to any Cartesian basis, while W assumes an explicit representation of the tangent planes to the level sets of F. Only the term F_{ww} can be computed directly from F, i.e.,

$$F_{ww} = (1/|\nabla F|)^2 \nabla F \; H \; \nabla F. \tag{B.3}$$

The second-order properties of the level sets of F are described by the upper left 2×2 submatrix of W,

$$W_{(2)} = \begin{bmatrix} F_{uu} & F_{uv} \\ F_{vu} & F_{vv} \end{bmatrix}, \tag{B.4}$$

which characterizes the second-order information of F restricted to the tangent plane of the level surface. The trace of this matrix is twice the mean curvature of the surface (times the gradient magnitude, which is necessary for the evolution equation) which is given by

$$\begin{aligned}\mathrm{Tr}[W_{(2)}] &= F_{uu} + F_{vv} = F_{uu} + F_{vv} + F_{ww} - F_{ww} \\ &= \mathrm{Tr}[H] - F_{ww} = F_{xx} + F_{yy} + F_{zz} - F_{ww}. \end{aligned} \tag{B.5}$$

The norm of $W_{(2)}$ is the deviation from flatness of the level surface (to within a factor of $|\nabla F|^2$):

$$\|W_{(2)}\|^2 = F_{uu}^2 + 2F_{uv}^2 + F_{vv}^2 = \|W\|^2 - 2\left(F_{uw}^2 + F_{vw}^2\right) - F_{ww}^2 \tag{B.6}$$

$$= \|H\|^2 - 2\|\nabla F_w\|^2 + F_{ww}^2 = \|H\|^2 - 2\|H\nabla F\|^2 + F_{ww}^2, \tag{B.7}$$

which is computed directly from first- and second-order derivatives of F.

The Gaussian curvature of the level set, which is given by the determinant of $W_{(2)}$ (divided by $|\nabla F|^2$), is a combination of the other two invariants:

$$\mathrm{Det}[W_{(2)}] = \frac{1}{2}\left(\left(\mathrm{Tr}[W_{(2)}]\right)^2 - \|W_{(2)}\|^2\right). \tag{B.8}$$

Finally, the explicit expression of the weighted curvature, which is used instead of the mean flow (the α term in Eq. A.7) to enforce "smoothness" in the implicit models, is

$$\kappa_w = \frac{1}{2}(F_{xx} + F_{yy} + F_{zz} - F_{ww})\left(\frac{(F_{xx} + F_{yy} + F_{zz} - F_{ww})^2}{\|H\| - 2\|H\nabla F\|^2 + F_{ww}^2} - 1\right). \tag{B.9}$$

Explicit values for principle curvatures, ω_1 and ω_2, are not necessary for the evolution equations presented in this paper. However, they can be calculated by solving the quadratic equation that results from values for the mean and Gaussian curvatures. Thus even the principle curvatures can be calculated without an explicit representation of the tangent space to the level surfaces.

Bibliography

[1] R. Drebin, L. Carpenter, and P. Hanrahan, "Volume rendering," *Computer Graphics*, vol. 22, no. 4, pp. 65–74, 1988.

[2] T. Yoo, U. Neumann, H. Fuchs, S. Pizer, T. Cullip, J. Rhoades, and R. Whitaker, "Direct visualization of volume data," *IEEE Computer Graphics and Applications*, vol. 12, no. 4, pp. 63–71, 1992.

[3] V. E. Johnson, "A framework for incorporating structural prior information into the estimation of medical images," in Barrett and Gmitro [23], pp. 307–321.

[4] D. Terzopoulos and D. Metaxas, "Dynamic 3d models with local and global deformations: Deformable superquadrics," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 7, pp. 703–714, 1991.

[5] J. Miller, D. Breen, W. Lorensen, R. O'Bara, and M. Wozny, "Geometrically deformed models: A method for extracting closed geometric models from volume data," in *SIGGRAPH 1991*, pp. 217–226, ACM Press, 1991.

[6] C. Nastar and N. Ayache, "Non-rigid motion analysis in medical images: a physically based approach," in Barrett and Gmitro [23], pp. 17–32.

[7] R. Szeliski, D. Tonnesen, and D. Terzopoulos, "Modeling surfaces of arbitrary topology with dynamic particles," in *Proceedings Fourth International Conference on Computer Vision (ICCV'93)*, (Berlin, Germany), pp. 82–87, IEEE Computer Society Press, 1993.

[8] D. Terzopoulos, A. Witkin, and M. Kass, "Constraints on deformable models: Recovering 3d shape and nonrigid motion," *Artificial Intellegence*, vol. 36, no. 1, pp. 91–123, 1988.

[9] U. Grenander, Y. Chow, and D. Keenan, *Hands: A Pattern Theoretic Study of Biological Shapes.* New York: Springer-Verlag, 1991.

[10] T. Cootes, A. Hill, C. Taylor, and J. Haslam, "The use of active shape models for locating structures in medical images," in Barrett and Gmitro [23], pp. 33–47.

[11] L. Staib and J. Duncan, "Boundary finding with parametrically deformable models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, pp. 1061–1075, 1992.

[12] R. Malladi, J. Sethian, and B. Vemuri, "Evolutionary fronts for topology-independent shape modeling and recovery," in *Third European Conference on Computer Vision* (J.-O. Eklundh, ed.), pp. 3–13, Springer-Verlag, 1994.

[13] M. Kass, A. Witkin, and D. Terzopoulis, "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 1, pp. 321–323, 1987.

[14] C. Brechbühler, G. Gerig, and O. Kübler, "Towards representation of 3d shape: Global surface parametrization," in *Visual Form: Analysis and Recognition* (C. Arcelli, L. P. Cordella, and G. Sanniti di Baya, eds.), (New York and London), pp. 79–88, Plenum Press, 1992.

[15] K. A. Brakke, *The Motion of a Surface by its Mean Curvature.* Princeton, NJ: Pinceton University Press, 1978.

[16] B. M. ter Haar Romeny, L. M. Florack, J. J. Koenderink, and M. A. Viergever, "Scale space: its natural operators and differential invariants," in *Lecture Notes in Computer Science* (A. C. F. Colchester and D. J. Hawkes, eds.), vol. 511, pp. 239–255, Springer-Verlag, 1991.

[17] T. Lindeberg, *Discrete Scale-space Theory and the Scale-space Primal Sketch.* PhD thesis, Royal Institute of Technology, Department of Numerical Analysis and Computing Science, Royal Institute of Technology, S-100 44 Stockholm, Sweden, 1991. TRITA-NA-P9108.

[18] R. T. Whitaker, *Geometry-Limited Diffusion.* PhD thesis, The University of North Carolina, Chapel Hill, North Carolina 27599-3175, 1993.

[19] R. van den Boomgaard, *Mathematical Morphology: Extensions Towards Computer Vision.* PhD thesis, University of Amsterdam, 1992.

[20] S. Osher and J. A. Sethian, "Fronts propogating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations," *Journal of Computational Physics*, vol. 79, pp. 12–49, 1988.

[21] J. J. Koenderink, "The structure of images," *Biol. Cybern.*, vol. 50, pp. 363–370, 1984.

[22] R. T. Whitaker and D. T. Chen, "Embedded active surface for volume visualization," in *SPIE Medical Imaging 1994*, (Newport Beach, California), 1994.

[23] H. H. Barrett and A. F. Gmitro, eds., *Information Processing in Medical Imaging (IPMI'93).* No. 687 in Lecture Notes in Computer Science, Springer-Verlag, 1993.