CHAPTER 3
Image Segmentation Using Deformable Models

Chenyang Xu
The Johns Hopkins University

Dzung L. Pham
National Institute of Aging

Jerry L. Prince
The Johns Hopkins University

Contents

3.1 Introduction 131
3.2 Parametric deformable models 133
3.2.1 Energy minimizing formulation 134
3.2.2 Dynamic force formulation 136
3.2.3 External forces 138
3.2.4 Numerical implementation 144
3.2.5 Discussion 145
3.3 Geometric deformable models 146
3.3.1 Curve evolution theory 146
3.3.2 Level set method 147
3.3.3 Speed functions 150
3.3.4 Relationship to parametric deformable models 152
3.3.5 Numerical implementation 153
3.3.6 Discussion 154
3.4 Extensions of deformable models 154
3.4.1 Deformable Fourier models 155
3.4.2 Deformable models using modal analysis 157
3.4.3 Deformable superquadrics 159
3.4.4 Active shape models 161
3.4.5 Other models 167
3.5 Conclusion and future directions 167

3.6 Further reading 168

3.7 Acknowledgments 168

3.8 References 168

3.1 Introduction

In the past four decades, computerized image segmentation has played an increasingly important role in medical imaging. Segmented images are now used routinely in a multitude of different applications, such as the quantification of tissue volumes [1], diagnosis [2], localization of pathology [3], study of anatomical structure [4, 5], treatment planning [6], partial volume correction of functional imaging data [7], and computer-integrated surgery [8, 9]. Image segmentation remains a difficult task, however, due to both the tremendous variability of object shapes and the variation in image quality (see Fig. 3.1). In particular, medical images are often corrupted by noise and sampling artifacts, which can cause considerable difficulties when applying classical segmentation techniques such as edge detection and thresholding. As a result, these techniques either fail completely or require some kind of postprocessing step to remove invalid object boundaries in the segmentation results.

To address these difficulties, *deformable models* have been extensively studied and widely used in medical image segmentation, with promising results. Deformable models are curves or surfaces defined within an image domain that can move under the influence of *internal forces*, which are defined within the curve or surface itself, and *external forces*, which are computed from the image data. The internal forces are designed to keep the model smooth during deformation. The external forces are defined to move the model toward an object boundary or other desired features within an image. By constraining extracted boundaries to be smooth and incorporating other prior information about the object shape, deformable models offer robustness to both image noise and boundary gaps and allow integrating boundary elements into a coherent and consistent mathematical description. Such a boundary description can then be readily used by subsequent applications. Moreover, since deformable models are implemented on the continuum, the resulting boundary representation can achieve subpixel accuracy, a highly desirable property for medical imaging applications. Figure 3.2 shows two examples of using deformable models to extract object boundaries from medical images. The result is a parametric curve in Fig. 3.2(a) and a parametric surface in Fig. 3.2(b).

Although the term *deformable models* first appeared in the work by Terzopoulos and his collaborators in the late eighties [12–15], the idea of deforming a template for extracting image features dates back much farther, to the work of Fischler and Elschlager's spring-loaded templates [16] and Widrow's rubber mask technique [17]. Similar ideas have also been used in the work by Blake and Zisserman [18], Grenander et al. [19], and Miller et al. [20]. The popularity of deformable models is largely due to the seminal paper "Snakes: Active Contours" by Kass, Witkin, and Terzopoulos [13]. Since its publication, deformable models have grown to be one of the most active and successful research areas in image segmentation. Various names, such as snakes, active contours or surfaces, balloons, and deformable contours or surfaces, have been used in the literature to refer to

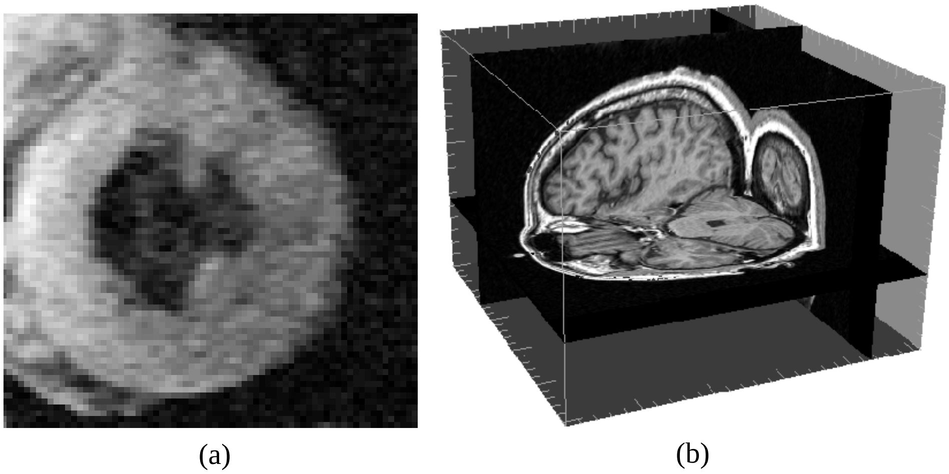


Figure 3.1: Variability of object shapes and image quality. (a) A 2D MR image of the heart left ventricle and (b) a 3D MR image of the brain.

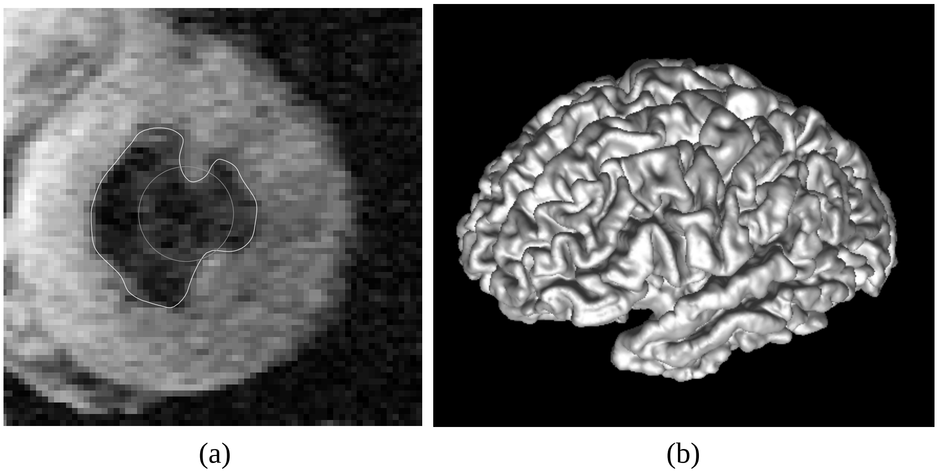


Figure 3.2: Examples of using deformable models to extract object boundaries from medical images. (a) An example of using a deformable contour to extract the inner wall of the left ventricle of a human heart from a 2D MR image. The circular initial deformable contour is plotted in gray and the final converged result is plotted in white [10]. (b) An example of using a deformable surface to reconstruct the brain cortical surface from a 3D MR image [11].

deformable models.

There are basically two types of deformable models: *parametric deformable models* (cf. [13, 21–23]) and *geometric deformable models* (cf. [24–27]). Parametric deformable models represent curves and surfaces explicitly in their parametric forms during deformation. This representation allows direct interaction with the model and can lead to a compact representation for fast real-time implementation. Adaptation of the model topology, however, such as splitting or merging parts during the deformation, can be difficult using parametric models. Geometric deformable models, on the other hand, can handle topological changes naturally. These models, based on the theory of curve evolution [28–31] and the level set method [32, 33], represent curves and surfaces implicitly as a level set of a higher-dimensional scalar function. Their parameterizations are computed only after complete deformation, thereby allowing topological adaptivity to be easily accommodated. Despite this fundamental difference, the underlying principles of both methods are very similar.

This chapter is organized as follows. We first introduce parametric deformable models in Section 3.2, and then describe geometric deformable models in Section 3.3. An explicit mathematical relationship between parametric deformable models and geometric deformable models is presented in Section 3.3.4. In Section 3.4, we provide an overview of several extensions to these deformable models. Finally, in Section 3.5, we conclude the chapter and point out future research directions. We focus on describing the fundamentals of deformable models and their application to image segmentation. Treatment of related work using deformable models in other applications such as image registration and motion estimation is beyond the scope of this chapter. We refer readers interested in these other applications to Section 3.6, where suggestions for further reading are given.

We note that although this chapter primarily deals with 2D deformable models (i.e., deformable contours), the principles discussed here apply to 3D deformable models (i.e., deformable surfaces) as well (cf. [23, 34]).

3.2 Parametric deformable models

In this section, we first describe two different types of formulations for parametric deformable models: an energy minimizing formulation and a dynamic force formulation. Although these two formulations lead to similar results, the first formulation has the advantage that its solution satisfies a minimum principle whereas the second formulation has the flexibility of allowing the use of more general types of external forces. We then present several commonly used external forces that can effectively attract deformable models toward the desired image features. A numerical implementation of 2D deformable models or *deformable contours* is described at the end of this section. Since the implementation of 3D deformable models or *deformable surfaces* is more sophisticated than those of deformable contours, we provide several references in Section 3.2.4 for additional reading rather than presenting an actual implementation.

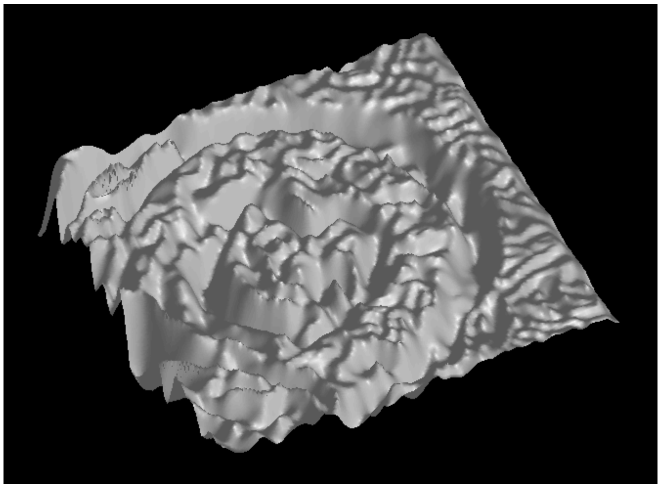


Figure 3.3: A potential energy function derived from Fig. 3.1(a).

3.2.1 Energy minimizing formulation

The basic premise of the energy minimizing formulation of deformable contours is to find a parameterized curve that minimizes the weighted sum of *internal energy* and *potential energy*. The internal energy specifies the tension or the smoothness of the contour. The potential energy is defined over the image domain and typically possesses local minima at the image intensity edges occurring at object boundaries (see Fig. 3.3). Minimizing the total energy yields *internal forces* and *potential forces*. Internal forces hold the curve together (elasticity forces) and keep it from bending too much (bending forces). External forces attract the curve toward the desired object boundaries. To find the object boundary, parametric curves are initialized within the image domain, and are forced to move toward the potential energy minima under the influence of both these forces.

Mathematically, a deformable contour is a curve $\boldsymbol{X}(s) = (X(s), Y(s))$, $s \in [0, 1]$, which moves through the spatial domain of an image to minimize the following energy functional:

$$\mathcal{E}(\boldsymbol{X}) = \mathcal{S}(\boldsymbol{X}) + \mathcal{P}(\boldsymbol{X})\,. \tag{3.1}$$

The first term is the internal energy functional and is defined to be

$$\mathcal{S}(\boldsymbol{X}) = \frac{1}{2}\int_0^1 \alpha(s)\left|\frac{\partial \boldsymbol{X}}{\partial s}\right|^2 + \beta(s)\left|\frac{\partial^2 \boldsymbol{X}}{\partial s^2}\right|^2 ds\,. \tag{3.2}$$

The first-order derivative discourages stretching and makes the model behave like an elastic string. The second-order derivative discourages bending and makes the model behave like a rigid rod. The weighting parameters $\alpha(s)$ and $\beta(s)$ can be used to control the strength of the model's tension and rigidity, respectively. In practice, $\alpha(s)$ and $\beta(s)$ are often chosen to be constants.

The second term is the potential energy functional and is computed by integrating a potential energy function $P(x, y)$ along the contour $\boldsymbol{X}(s)$:

$$\mathcal{P}(\boldsymbol{X}) = \int_0^1 P(\boldsymbol{X}(s))ds\,. \tag{3.3}$$

The potential energy function $P(x, y)$ is derived from the image data and takes smaller values at object boundaries as well as other features of interest. Given a gray-level image $I(x, y)$ viewed as a function of continuous position variables (x, y), a typical potential energy function designed to lead a deformable contour toward step edges is

$$P(x, y) = -w_e \left|\nabla[G_\sigma(x, y) * I(x, y)]\right|^2\,, \tag{3.4}$$

where w_e is a positive weighting parameter, $G_\sigma(x, y)$ is a two-dimensional Gaussian function with standard deviation σ, ∇ is the gradient operator, and $*$ is the 2D image convolution operator. If the desired image features are lines, then the appropriate potential energy function can be defined as follows:

$$P(x, y) = w_l[G_\sigma(x, y) * I(x, y)]\,, \tag{3.5}$$

where w_l is a weighting parameter. Positive w_l is used to find black lines on a white background, while negative w_l is used to find white lines on a black background. For both edge and line potential energies, increasing σ can broaden its attraction range. However, larger σ can also cause a shift in the boundary location, resulting in a less accurate result (this problem can be addressed by using potential energies calculated with different values of σ; see Section 3.2.3).

Regardless of the selection of the exact potential energy function, the procedure for minimizing the energy functional is the same. The problem of finding a curve $\boldsymbol{X}(s)$ that minimizes the energy functional $\mathcal{E}$ is known as a variational problem [35]. It has been shown that the curve that minimizes $\mathcal{E}$ must satisfy the following Euler-Lagrange equation [13, 22]:

$$\frac{\partial}{\partial s}\left(\alpha\frac{\partial \boldsymbol{X}}{\partial s}\right) - \frac{\partial^2}{\partial s^2}\left(\beta\frac{\partial^2 \boldsymbol{X}}{\partial s^2}\right) - \nabla P(\boldsymbol{X}) = 0\,. \tag{3.6}$$

To gain some insight about the physical behavior of deformable contours, we can view Eq. (3.6) as a force balance equation

$$\boldsymbol{F}_{\text{int}}(\boldsymbol{X}) + \boldsymbol{F}_{\text{pot}}(\boldsymbol{X}) = 0\,, \tag{3.7}$$

where the internal force is given by

$$\boldsymbol{F}_{\mathrm{int}}(\boldsymbol{X}) = \frac{\partial}{\partial s}\left(\alpha \frac{\partial \boldsymbol{X}}{\partial s}\right) - \frac{\partial^2}{\partial s^2}\left(\beta \frac{\partial^2 \boldsymbol{X}}{\partial s^2}\right) \tag{3.8}$$

and the potential force is given by

$$\boldsymbol{F}_{\mathrm{pot}}(\boldsymbol{X}) = -\nabla P(\boldsymbol{X})\,. \tag{3.9}$$

The internal force $\boldsymbol{F}_{\mathrm{int}}$ discourages stretching and bending while the potential force $\boldsymbol{F}_{\mathrm{pot}}$ pulls the contour toward the desired object boundaries. In this chapter, we define the forces, derived from the potential energy function $P(x, y)$ given in either Eq. (3.4) or Eq. (3.5), as *Gaussian potential forces*.

To find a solution to Eq. (3.6), the deformable contour is made dynamic by treating $\boldsymbol{X}(s)$ as a function of time t as well as s — i.e., $\boldsymbol{X}(s, t)$. The partial derivative of $\boldsymbol{X}$ with respect to t is then set equal to the left-hand side of Eq. (3.6) as follows:

$$\gamma \frac{\partial \boldsymbol{X}}{\partial t} = \frac{\partial}{\partial s}\left(\alpha \frac{\partial \boldsymbol{X}}{\partial s}\right) - \frac{\partial^2}{\partial s^2}\left(\beta \frac{\partial^2 \boldsymbol{X}}{\partial s^2}\right) - \nabla P(\boldsymbol{X})\,. \tag{3.10}$$

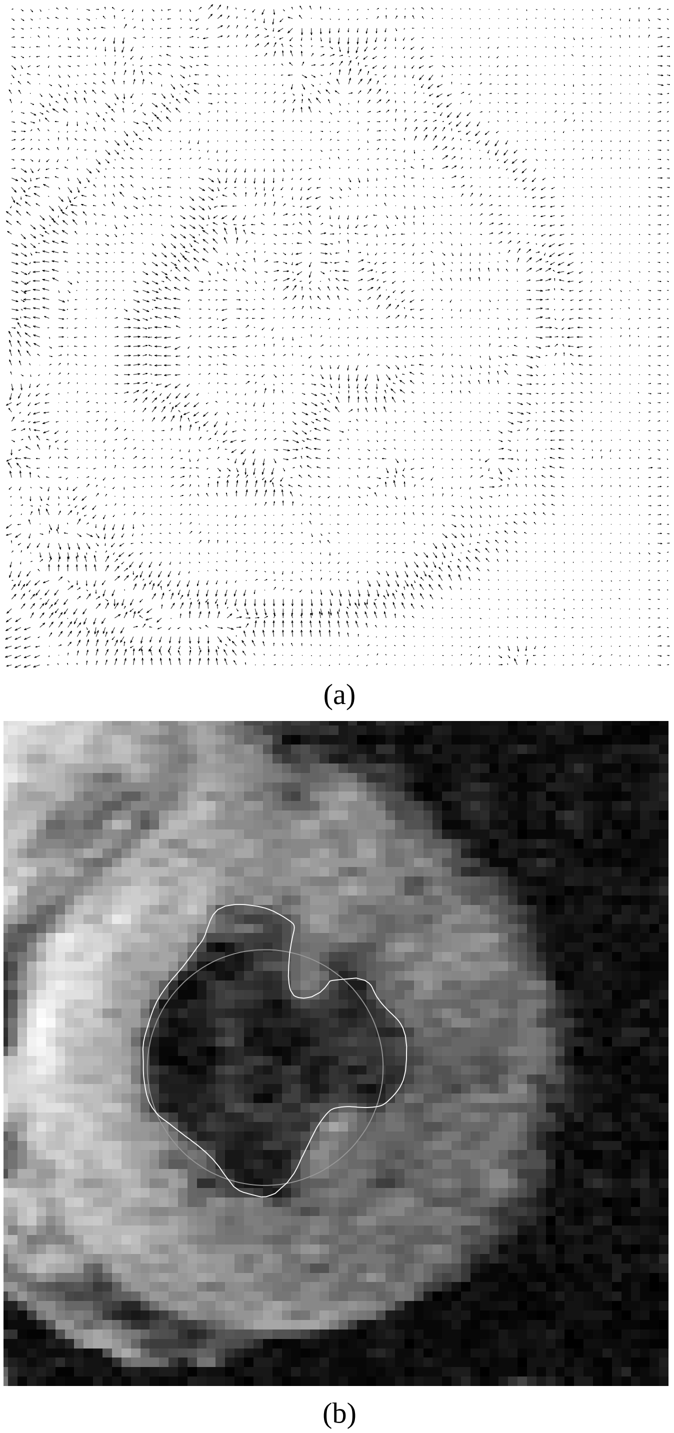
The coefficient γ is introduced to make the units on the left side consistent with the right side. When the solution $\boldsymbol{X}(s, t)$ stabilizes, the left side vanishes and we achieve a solution of Eq. (3.6). We note that this approach of making the time derivative term vanish is equivalent to applying a gradient descent algorithm to find the local minimum of Eq. (3.1) [34]. Thus, the minimization is solved by placing an initial contour on the image domain and allowing it to deform according to Eq. (3.10). Figure 3.4 shows an example of recovering the left ventricle wall using Gaussian potential forces.

3.2.2 Dynamic force formulation

In the previous section, the deformable model was modeled as a static problem, and an artificial variable t was introduced to minimize the energy. It is sometimes more convenient, however, to formulate the deformable model directly from a dynamic problem using a force formulation. Such a formulation permits the use of more general types of external forces that are not potential forces, i.e., forces that cannot be written as the negative gradient of potential energy functions. According to Newton's second law, the dynamics of a contour $\boldsymbol{X}(s, t)$ must satisfy

$$\mu \frac{\partial^2 \boldsymbol{X}}{\partial t^2} = \boldsymbol{F}_{\mathrm{damp}}(\boldsymbol{X}) + \boldsymbol{F}_{\mathrm{int}}(\boldsymbol{X}) + \boldsymbol{F}_{\mathrm{ext}}(\boldsymbol{X})\,, \tag{3.11}$$

where μ is a coefficient that has a mass unit and $\boldsymbol{F}_{\mathrm{damp}}$ is the damping (or viscous) force defined as $-\gamma \partial \boldsymbol{X}/\partial t$, with γ being the damping coefficient. In image segmentation, the mass coefficient μ in front of the inertial term is often set to zero,



(a)

(b)

Figure 3.4: An example of recovering the left ventricle wall using Gaussian potential forces. (a) Gaussian potential forces and (b) the result of applying Gaussian potential forces to a deformable contour, with the circular initial contour shown in gray and the final deformed contour in white.

since the inertial term may cause the contour to pass over the weak edges. The dynamics of the deformable contour without the inertial term becomes

$$\gamma \frac{\partial \boldsymbol{X}}{\partial t} = \boldsymbol{F}_{\text{int}}(\boldsymbol{X}) + \boldsymbol{F}_{\text{ext}}(\boldsymbol{X})\,. \tag{3.12}$$

The internal forces are the same as specified in Eq. (3.8). The external forces can be either potential forces or nonpotential forces. We note, however, nonpotential forces cannot be derived from the variational energy formulation of the previous section. An alternate variational principle does exist (see [36]); however, it is not physically intuitive.

External forces are often expressed as the superposition of several different forces:

$$\boldsymbol{F}_{\text{ext}}(\boldsymbol{X}) = \boldsymbol{F}_1(\boldsymbol{X}) + \boldsymbol{F}_2(\boldsymbol{X}) + \cdots + \boldsymbol{F}_N(\boldsymbol{X})\,,$$

where N is the total number of external forces. This superposition formulation allows the external forces to be broken down into more manageable terms. For example, one might define the external forces to be composed of both Gaussian potential forces and pressure forces, which are described in the next section.

3.2.3 External forces

In this section, we describe several kinds of external forces for deformable models. These external forces are applicable to both deformable contours and deformable surfaces.

Multiscale Gaussian potential force

When using the Gaussian potential force described in Section 3.2.1, σ must be selected to have a small value in order for the deformable model to follow the boundary accurately. As a result, the Gaussian potential force can only attract the model toward the boundary when it is initialized nearby. To remedy this problem, Terzopoulos, Witkin, and Kass [13, 15] proposed using Gaussian potential forces at different scales to broaden its attraction range while maintaining the model's boundary localization accuracy. The basic idea is to first use a large value of σ to create a potential energy function with a broad valley around the boundary. The coarse-scale Gaussian potential force attracts the deformable contour or surface toward the desired boundaries from a long range. When the contour or surface reaches equilibrium, the value of σ is then reduced to allow tracking of the boundary at a finer scale. This scheme effectively extends the attraction range of the Gaussian potential force. A weakness of this approach, however, is that there is no established theorem for how to schedule changes in σ. The ad hoc scheduling schemes that are available may therefore lead to unreliable results.

Pressure force

Cohen [22] proposed to increase the attraction range by using a pressure force together with the Gaussian potential force. The pressure force can either inflate or deflate the model; hence, it removes the requirement to initialize the model near the desired object boundaries. Deformable models that use pressure forces are also known as balloons [22].

The pressure force is defined as

$$\boldsymbol{F}_p(\boldsymbol{X}) = w_p \boldsymbol{N}(\boldsymbol{X}) \,, \tag{3.13}$$

where $\boldsymbol{N}(\boldsymbol{X})$ is the inward unit normal[1] of the model at the point $\boldsymbol{X}$ and w_p is a constant weighting parameter. The sign of w_p determines whether to inflate or deflate the model and is typically chosen by the user. Recently, region information has been used to define w_p with a spatial-varying sign based upon whether the model is inside or outside the desired object (see [37, 38]). The value of w_p determines the strength of the pressure force. It must be carefully selected so that the pressure force is slightly smaller than the Gaussian potential force at significant edges, but large enough to pass through weak or spurious edges. When the model deforms, the pressure force keeps inflating or deflating the model until it is stopped by the Gaussian potential force. An example of using deformable contour with an inflating pressure force is shown in Fig. 3.5. A disadvantage in using pressure forces is that they may cause the deformable model to cross itself and form loops (cf. [39]).

Distance potential force

Another approach for extending attraction range is to define the potential energy function using a distance map as proposed by Cohen and Cohen [40]. The value of the distance map at each pixel is obtained by calculating the distance between the pixel and the closest boundary point, based either on Euclidean distance [41] or Chamfer distance [42]. By defining the potential energy function based on the distance map, one can obtain a potential force field that has a large attraction range.

Given a computed distance map $d(x, y)$, one way of defining a corresponding potential energy, introduced in [40], is as follows:

$$P_d(x, y) = -w_d \exp[-d(x, y)^2] \,. \tag{3.14}$$

The corresponding potential force field is given by $-\nabla P_d(x, y)$.

Gradient vector flow

The distance potential force is based on the principle that the model point should be attracted to the nearest edge points. This principle, however, can cause difficulties when deforming a contour or surface into boundary concavities [43]. A 2D example is shown in Fig. 3.6, where a U-shaped object and a close-up of its distance potential force field within the boundary concavity is depicted. Notice

[1] In the parametric formulation of deformable models, the normal direction is sometimes assumed to be outward. Here we assume an inward direction for consistency with the geometric formulation of deformable models introduced in Section 3.3.

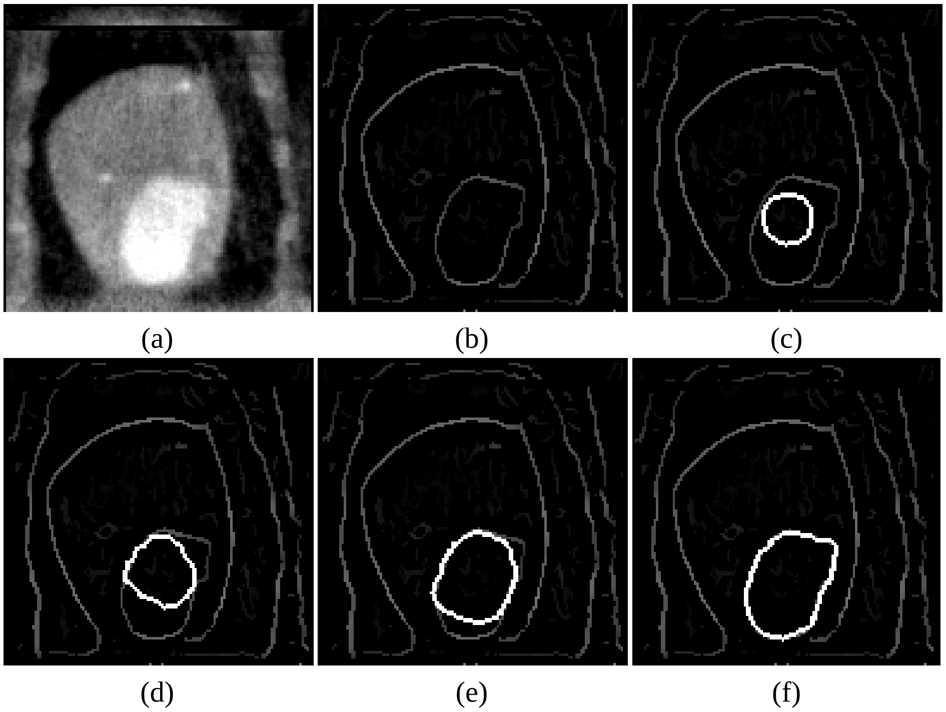


Figure 3.5: An example of pressure forces driven deformable contours. (a) Intensity CT image slice of the left ventricle. (b) Edge detected image. (c) Initial deformable contour. (d)-(f) Deformable contour moving toward the left ventricle boundary, driven by inflating pressure force. Images courtesy of McInerney and Terzopoulos [23], The University of Toronto.

that at the concavity, distance potential forces point horizontally in opposite directions, thus preventing the contour from converging into the boundary concavity. To address this problem, Xu and Prince [10, 43] employed a vector diffusion equation that diffuses the gradient of an edge map in regions distant from the boundary, yielding a different force field called the *gradient vector flow* (GVF) field. The amount of diffusion adapts according to the strength of edges to avoid distorting object boundaries.

A GVF field is defined as the equilibrium solution to the following vector partial differential equation:

$$\frac{\partial \boldsymbol{v}}{\partial t} = g(|\nabla f|)\nabla^2 \boldsymbol{v} - h(|\nabla f|)(\boldsymbol{v} - \nabla f)\,, \tag{3.15}$$

where $\boldsymbol{v}(x, y, 0) = \nabla f$, $\partial \boldsymbol{v}/\partial t$ denotes the partial derivative of $\boldsymbol{v}(x, y, t)$ with respect to t, ∇^2 is the Laplacian operator (applied to each spatial component of $\boldsymbol{v}$ separately), and f is an edge map that has higher value at the desired object

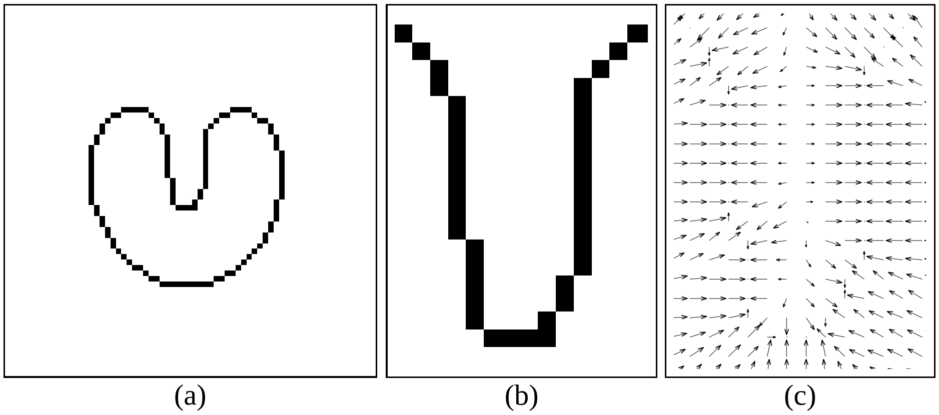


Figure 3.6: An example of distance potential force field. (a) A U-shaped object, a close-up of its (b) boundary concavity, and (c) the distance potential force field within the concavity.

boundary and can be derived using any edge detector. The definition of the GVF field is valid for any dimension. Two examples of $g(r)$ and $h(r)$ are

$$\begin{aligned} g(r) &= \exp\{-(r/\kappa)^2\} \\ h(r) &= 1 - g(r)\,, \end{aligned}$$

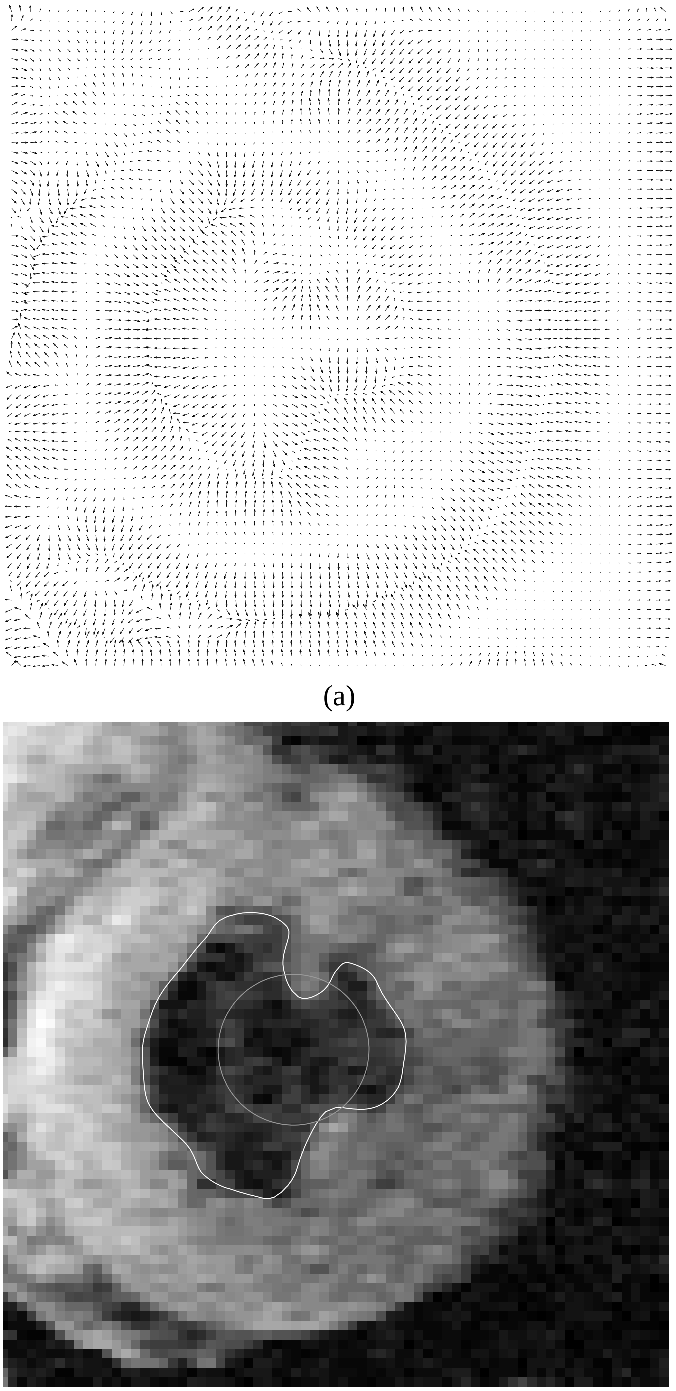
where κ is a scalar and r is a dummy variable, or

$$\begin{aligned} g(r) &= \mu \\ h(r) &= r^2\,, \end{aligned}$$

where μ is a positive scalar. GVF has been shown to have a large attraction range and improved convergence for deforming contours into boundary concavities [10, 43]. An example of using a GVF force field is shown in Fig. 3.7.

Dynamic distance force

An external force that is similar to distance potential force but does not possess the boundary concavity problem has been proposed [44, 45]. This approach derives an external force by computing a signed distance at each point on the deformable contour or surface. This signed distance is calculated by determining the closest boundary point or other image feature along the model's normal direction. The distance values are recomputed each time the model is deformed. Several criteria can be used to define the desired boundary point to be searched. The most common one is to use image pixels that have a high image intensity gradient magnitude or edge points generated by an edge detector. A threshold is specified for the maximum search distance to avoid confusion with outliers and to reduce the computation time. The resulting force, which we refer to as the *dynamic distance*



(a)

(b)

Figure 3.7: An example of the gradient vector flow driven deformable contours. (a) A gradient vector flow force field and (b) the result of applying gradient vector flow force to a deformable contour, with the circular initial contour shown in gray and the final deformed contour in white.

force, can attract deformable models to the desired image feature from a fairly long range limited only by the threshold.

Given a point $\boldsymbol{X}$ on the contour or surface, its inward unit normal $\boldsymbol{N}(\boldsymbol{X})$, the computed signed distance $D(\boldsymbol{X})$, and a specified distance threshold $D_{\max}$, a typical definition for the dynamic distance force is

$$\boldsymbol{F}_D(\boldsymbol{X}) = w_D \frac{D(\boldsymbol{X})}{D_{\max}} \boldsymbol{N}(\boldsymbol{X})\,. \tag{3.16}$$

The weakness of this method is that a relatively time-consuming 1D search along the normal direction must be performed each time the model deforms. Setting the search distance threshold lower can reduce the run time but has the undesirable side effect of decreasing the attraction range of the dynamic distance force.

Interactive force

In many clinical situations, it is important to allow an operator to interact with the deformable model as it is deforming. This interaction improves the accuracy of the segmentation result when automated external forces fail to deform the model to the desired feature in certain regions. For example, the user may want to pull the model toward significant image features, or would like to constrain the model so that it must pass through a set of *landmark points* identified by an expert. Deformable models allow these kinds of user interactions to be conveniently modeled as additional force terms.

Two kinds of commonly used interactive forces are *spring forces* and *volcano forces*, proposed by Kass et al. [13]. Spring forces are defined to be proportional to the distance between a point $\boldsymbol{X}$ on the model and a user-specified point $\boldsymbol{p}$:

$$\boldsymbol{F}_s = w_s(\boldsymbol{p} - \boldsymbol{X})\,. \tag{3.17}$$

Spring forces act to pull the model toward $\boldsymbol{p}$. The further away the model is from $\boldsymbol{p}$, the stronger the pulling force. The point $\boldsymbol{X}$ is selected by finding the closest point on the model to $\boldsymbol{p}$ using a heuristic search around a local neighborhood of $\boldsymbol{p}$. An example of using spring forces is shown in Fig. 3.8.

Volcano forces are designed to push the model away from a local region around a "volcano" point $\boldsymbol{p}$. For computational efficiency, the force is only computed in a neighborhood $\mathcal{N}(\boldsymbol{p})$ as follows:

$$\boldsymbol{F}_v(\boldsymbol{X}) = \begin{cases} w_v \frac{\boldsymbol{r}}{|\boldsymbol{r}|^3} & \boldsymbol{X} \in \mathcal{N}(\boldsymbol{p}) \\ 0 & \boldsymbol{X} \notin \mathcal{N}(\boldsymbol{p}) \end{cases}\,, \tag{3.18}$$

where $\boldsymbol{r} = \boldsymbol{X} - \boldsymbol{p}$. Note that the magnitude of the forces is limited near $\boldsymbol{r} = 0$ to avoid numerical instability. Another possible definition for volcano forces is

$$\boldsymbol{F}_v(\boldsymbol{X}) = \begin{cases} w_v \exp(-\frac{|\boldsymbol{r}|^2}{\sigma_v^2}) \frac{\boldsymbol{r}}{|\boldsymbol{r}|} & \boldsymbol{X} \in \mathcal{N}(\boldsymbol{p}) \\ 0 & \boldsymbol{X} \notin \mathcal{N}(\boldsymbol{p}) \end{cases}\,, \tag{3.19}$$

where σ_v is used to adjust the strength distribution of the volcano force.

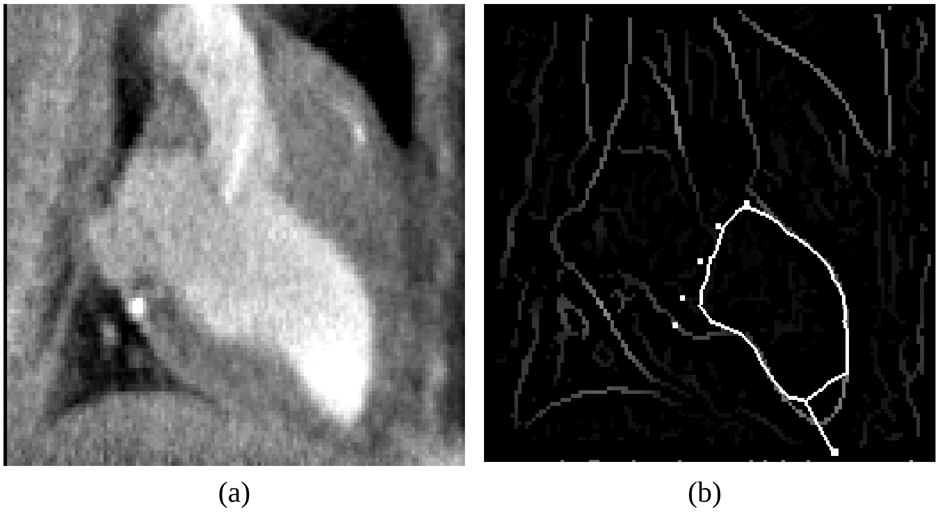


Figure 3.8: Example of interactive forces. (a) A CT image slice of a canine left ventricle. (b) A deformable contour moves toward high gradients in the edge detected image, influenced by landmark points near the center of the image and a spring force that pulls the contour toward an edge at the bottom right. Image courtesy of McInerney and Terzopoulos [23], The University of Toronto.

3.2.4 Numerical implementation

Various numerical implementations of deformable models have been reported in the literature. For examples, the finite difference method [13], dynamic programming [21], and greedy algorithm [46] have been used to implement deformable contours, while finite difference methods [15] and finite element methods [23, 34, 47] have been used to implement deformable surfaces. The finite difference method requires only local operations and is efficient to compute. The finite element method, on the other hand, is more costly to compute but has the advantage of being well adapted to the irregular mesh representations of deformable surfaces. In this section, we present the finite difference method implementation for deformable contours as described in [13].

Since the numerical scheme proposed by [13] does not require external forces to be potential forces, it can be used to implement deformable contours using either potential forces or nonpotential forces. By approximating the derivatives in Eq. (3.12) with finite differences, and converting to the vector notation

$\boldsymbol{X}_i^n = (X_i^n, Y_i^n) = (X(ih, n\Delta t), Y(ih, n\Delta t))$, we can rewrite Eq. (3.12) as

$$\begin{aligned}\gamma\frac{\boldsymbol{X}_i^n - \boldsymbol{X}_i^{n-1}}{\Delta t} &= \frac{1}{h^2}[\alpha_{i+1}(\boldsymbol{X}_{i+1}^n - \boldsymbol{X}_i^n) - \alpha_i(\boldsymbol{X}_i^n - \boldsymbol{X}_{i-1}^n)] \\ &\quad -\frac{1}{h^4}[\beta_{i-1}(\boldsymbol{X}_{i-2}^n - 2\boldsymbol{X}_{i-1}^n + \boldsymbol{X}_i^n) \\ &\quad -2\beta_i(\boldsymbol{X}_{i-1}^n - 2\boldsymbol{X}_i^n + \boldsymbol{X}_{i+1}^n) \\ &\quad +\beta_{i+1}(\boldsymbol{X}_i^n - 2\boldsymbol{X}_{i+1}^n + \boldsymbol{X}_{i+2}^n)] + \boldsymbol{F}_{\text{ext}}(\boldsymbol{X}_i^{n-1}) \quad (3.20)\end{aligned}$$

where γ is the damping coefficient, $\alpha_i = \alpha(ih)$, $\beta_i = \beta(ih)$, h the step size in space, and Δt the step size in time. In general, the external force $\boldsymbol{F}_{\text{ext}}$ is stored as a discrete vector field, i.e., a finite set of vectors defined on an image grid. The value of $\boldsymbol{F}_{\text{ext}}$ at any location $\boldsymbol{X}_i$ can be obtained through a bilinear interpolation of the external force values at the grid points near $\boldsymbol{X}_i$.

Equation (3.20) can be written in a compact matrix form as

$$\frac{\boldsymbol{X}^n - \boldsymbol{X}^{n-1}}{\tau} = \boldsymbol{A}\boldsymbol{X}^n + \boldsymbol{F}_{\text{ext}}(\boldsymbol{X}^{n-1}), \quad (3.21)$$

where $\tau = \Delta t/\gamma$, $\boldsymbol{X}^n$, $\boldsymbol{X}^{n-1}$, and $\boldsymbol{F}_{\text{ext}}(\boldsymbol{X}^{n-1})$ are $m \times 2$ matrices, and $\boldsymbol{A}$ is an $m \times m$ pentadiagonal banded matrix with m being the number of sample points. Equation (3.21) can then be solved iteratively by matrix inversion using the following equation:

$$\boldsymbol{X}^n = (\boldsymbol{I} - \tau\boldsymbol{A})^{-1}[\boldsymbol{X}^{n-1} + \tau\boldsymbol{F}_{\text{ext}}(\boldsymbol{X}^{n-1})]. \quad (3.22)$$

The inverse of the matrix $\boldsymbol{I} - \tau\boldsymbol{A}$ can be calculated efficiently by LU decomposition[2]. The decomposition needs only to be performed once for deformation processes that do not alter the elasticity or rigidity parameters.

3.2.5 Discussion

So far, we have formulated the deformable model as a continuous curve or surface. In practice, however, it is sometimes more straightforward to design the deformable models from a discrete point of view. Example of work in this area includes [48–53].

Parametric deformable models have been applied successfully in a wide range of applications; however, they have two main limitations. First, in situations where the initial model and the desired object boundary differ greatly in size and shape, the model must be reparameterized dynamically to faithfully recover the object boundary. Methods for reparameterization in 2D are usually straightforward and require moderate computational overhead. Reparameterization in 3D, however, requires complicated and computationally expensive methods. The second limitation

[2] LU decomposition stands for Lower and Upper triangular decomposition, a well-known technique in linear algebra.

with the parametric approach is that it has difficulty dealing with topological adaptation such as splitting or merging model parts, a useful property for recovering either multiple objects or an object with unknown topology. This difficulty is caused by the fact that a new parameterization must be constructed whenever the topology change occurs, which requires sophisticated schemes [54, 55].

3.3 Geometric deformable models

Geometric deformable models, proposed independently by Caselles et al. [24] and Malladi et al. [25], provide an elegant solution to address the primary limitations of parametric deformable models. These models are based on curve evolution theory [28–31] and the level set method [32, 33]. In particular, curves and surfaces are evolved using only geometric measures, resulting in an evolution that is independent of the parameterization. As in parametric deformable models, the evolution is coupled with the image data to recover object boundaries. Since the evolution is independent of the parameterization, the evolving curves and surfaces can be represented implicitly as a level set of a higher-dimensional function. As a result, topology changes can be handled automatically.

In this section, we first review the fundamental concepts in curve evolution theory and the level set method. We next present three types of geometric deformable models, the difference being in the design of speed functions. We then show a mathematical relationship between a particular class of parametric and geometric models. Next, we describe a numerical implementation of geometric deformable models proposed by Osher and Sethian [32] in Section 3.3.5. Finally, at the end of this section we compare geometric deformable models with parametric deformable models. We note that although the geometric deformable models are presented in 2D, their formulation can be directly extended to 3D. A thorough treatment on evolving curves and surfaces using the level set representation can be found in [33].

3.3.1 Curve evolution theory

The purpose of curve evolution theory is to study the deformation of curves using only geometric measures such as the unit normal and curvature as opposed to the quantities that depend on parameters such as the derivatives of an arbitrary parameterized curve. Let us consider a moving curve $\boldsymbol{X}(s,t) = [X(s,t), Y(s,t)]$, where s is any parameterization and t is the time, and denote its inward unit normal as $\boldsymbol{N}$ and its curvature as κ, respectively. The evolution of the curve along its normal direction can be characterized by the following partial differential equation:

$$\frac{\partial \boldsymbol{X}}{\partial t} = V(\kappa)\boldsymbol{N}\,, \tag{3.23}$$

where $V(\kappa)$ is called *speed function,* since it determines the speed of the curve evolution. We note that a curve moving in some arbitrary direction can always be reparameterized to have the same form as Eq. (3.23) [56]. The intuition behind this fact is that the tangent deformation affects only the curve's parameterization, not its shape and geometry.

The most extensively studied curve deformations in curve evolution theory are *curvature deformation* and *constant deformation*. Curvature deformation is given by the so-called *geometric heat equation*

$$\frac{\partial \boldsymbol{X}}{\partial t} = \alpha \kappa \boldsymbol{N} \,,$$

where α is a positive constant. This equation will smooth a curve, eventually shrinking it to a circular point [57]. The use of the curvature deformation has an effect similar to the use of the elastic internal force in parametric deformable models.

Constant deformation is given by

$$\frac{\partial \boldsymbol{X}}{\partial t} = V_0 \boldsymbol{N} \,,$$

where V_0 is a coefficient determining the speed and direction of deformation. Constant deformation plays the same role as the pressure force in parametric deformable models. The properties of curvature deformation and constant deformation are complementary to each other. Curvature deformation removes singularities by smoothing the curve, while constant deformation can create singularities from an initially smooth curve.

The basic idea of the geometric deformable model is to couple the speed of deformation (using curvature and/or constant deformation) with the image data, so that the evolution of the curve stops at object boundaries. The evolution is implemented using the level set method. Thus, most of the research in geometric deformable models has been focused in the design of speed functions. We review several representative speed functions in Section 3.3.3.

3.3.2 Level set method

We now review the level set method for implementing curve evolution. The level set method is used to account for automatic topology adaptation, and it also provides the basis for a numerical scheme that is used by geometric deformable models. The level set method for evolving curves is due to Osher and Sethian [32, 58, 59].

In the level set method, the curve is represented implicitly as a level set of a 2D scalar function — referred to as the level set function — which is usually defined on the same domain as the image. The level set is defined as the set of points that have the same function value. Figure 3.9 shows an example of embedding a curve as a zero level set. It is worth noting that the level set function is different from the level sets of images, which are sometimes used for image enhancement [60]. The sole purpose of the level set function is to provide an implicit representation of the evolving curve.

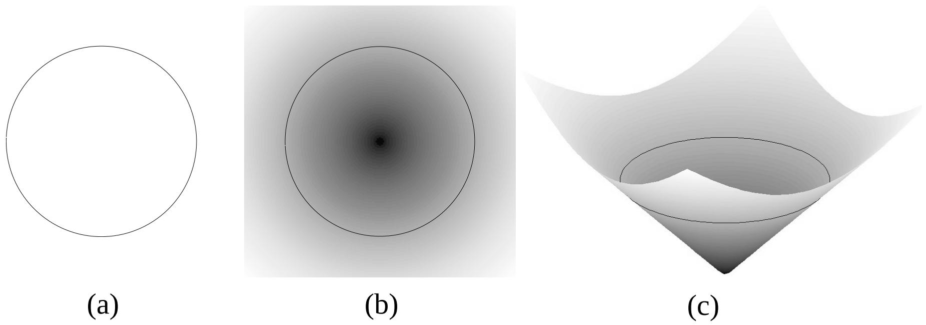


Figure 3.9: An example of embedding a curve as a level set. (a) A single curve. (b) The level set function where the curve is embedded as the zero level set (in black). (c) The height map of the level set function with its zero level set depicted in black.

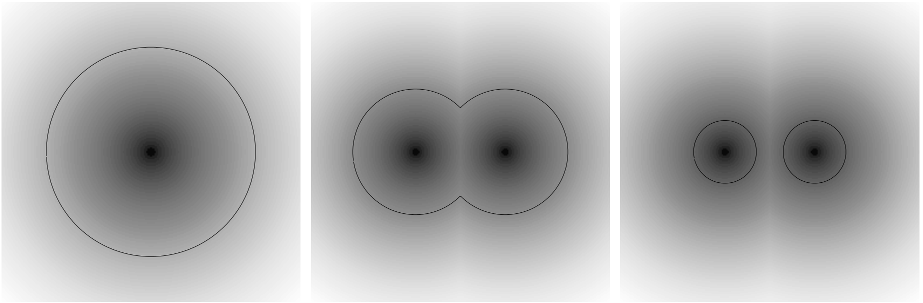


Figure 3.10: From left to right, the zero level set splits into two curves while the level set function still remains a valid function.

Instead of tracking a curve through time, the level set method evolves a curve by updating the level set function at fixed coordinates through time. This perspective is similar to that of an Eulerian formulation of motion as opposed to a Lagrangian formulation, which is analogous to the parametric deformable model. A useful property of this approach is that the level set function remains a valid function while the embedded curve can change its topology. This situation is depicted in Fig 3.10.

We now derive the level set embedding of the curve evolution equation (3.23). Given a level set function $\phi(x, y, t)$ with the contour $\boldsymbol{X}(s, t)$ as its zero level set, we have

$$\phi[\boldsymbol{X}(s,t), t] = 0\,.$$

Differentiating the above equation with respect to t and using the chain rule, we obtain

$$\frac{\partial \phi}{\partial t} + \nabla \phi \cdot \frac{\partial \boldsymbol{X}}{\partial t} = 0\,, \tag{3.24}$$

where $\nabla \phi$ denotes the gradient of ϕ.

We assume that ϕ is negative inside the zero level set and positive outside. Accordingly, the inward unit normal to the level set curve is given by

$$\boldsymbol{N} = -\frac{\nabla \phi}{|\nabla \phi|}\,. \tag{3.25}$$

Using this fact and Eq. (3.23), we can rewrite Eq. (3.24) as

$$\frac{\partial \phi}{\partial t} = V(\kappa)|\nabla \phi|\,, \tag{3.26}$$

where the curvature κ at the zero level set is given by

$$\kappa = \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|} = \frac{\phi_{xx}\phi_y^2 - 2\phi_x\phi_y\phi_{xy} + \phi_{yy}\phi_x^2}{(\phi_x^2 + \phi_y^2)^{3/2}}\,. \tag{3.27}$$

The relationship between Eq. (3.23) and Eq. (3.26) provides the basis for performing curve evolution using the level set method.

Three issues need to be considered in order to implement geometric deformable contours:

1. An initial function $\phi(x, y, t = 0)$ must be constructed such that its zero level set corresponds to the position of the initial contour. A common choice is to set $\phi(x, y, 0) = D(x, y)$, where $D(x, y)$ is the signed distance from each grid point to the zero level set. The computation of the signed distance for an arbitrary initial curve is expensive. Recently, Sethian and Malladi developed a method called the fast marching method, which can construct the signed distance function in $O(N \log N)$, where N is the number of pixels. Certain situations may arise, however, where the distance may be computed much more efficiently. For example, when the zero level set can be described by the exterior boundary of the union of a collection of disks, the signed distance function can be computed in $O(N)$ as

   $$D(\boldsymbol{x}) = \min_{i=1,2,\cdots,M} (|\boldsymbol{x} - \boldsymbol{c}_i| - r_i)\,,$$

   where $\boldsymbol{x} = (x, y)$, M is the number of initial disks, $\boldsymbol{c}_i$ and r_i are the center and radius of each disk.

2. Since the evolution equation (3.26) is derived for the zero level set only, the speed function $V(\kappa)$, in general, is not defined on other level sets. Hence, we need a method to extend the speed function $V(\kappa)$ to all of the level sets.

We note that the expressions for the unit normal and the curvature, however, hold for all level sets. Many approaches for such extensions have been developed (see [33] for a detailed discussion on this topic). However, the level set function that evolves using these extended speed functions can lose its property of being a signed distance function, causing inaccuracy in curvature and normal calculations. As a result, reinitialization of the level set function to a signed distance function is often required for these schemes. Recently, a method that does not suffer from this problem was proposed by Adalsteinsson and Sethian [61]. This method casts the speed extension problem as a boundary value problem, which can then be solved efficiently using the fast marching method.

3. In the application of geometric contours, constant deformation is often used to account for large-scale deformation and narrow boundary indentation and protrusion recovery. Constant deformation, however, can cause the formation of sharp corners from an initial smooth zero level set. Once the corner is developed, it is not clear how to continue the deformation, since the definition of the normal direction becomes ambiguous. A natural way to continue the deformation is to impose the so-called *entropy condition* originally proposed in the area of interface propagation by Sethian [62]. In Section 3.3.5, we describe an entropy satisfying numerical scheme, proposed by Osher and Sethian [32], which implements geometric deformable contours.

3.3.3 Speed functions

In this section, we provide a brief overview of three examples of speed functions used by geometric deformable contours.

The geometric deformable contour formulation, proposed by Caselles et al. [24] and Malladi et al. [25], takes the following form:

$$\frac{\partial \phi}{\partial t} = c(\kappa + V_0)|\nabla \phi| \, , \tag{3.28}$$

where

$$c = \frac{1}{1 + |\nabla(G_\sigma * I)|} \, . \tag{3.29}$$

Positive V_0 shrinks the curve, and negative V_0 expands the curve. The curve evolution is coupled with the image data through a multiplicative stopping term c. This scheme can work well for objects that have good contrast. However, when the object boundary is indistinct or has gaps, the geometric deformable contour may leak out because the multiplicative term only slows down the curve near the boundary rather than completely stopping the curve. Once the curve passes the boundary, it will not be pulled back to recover the correct boundary.

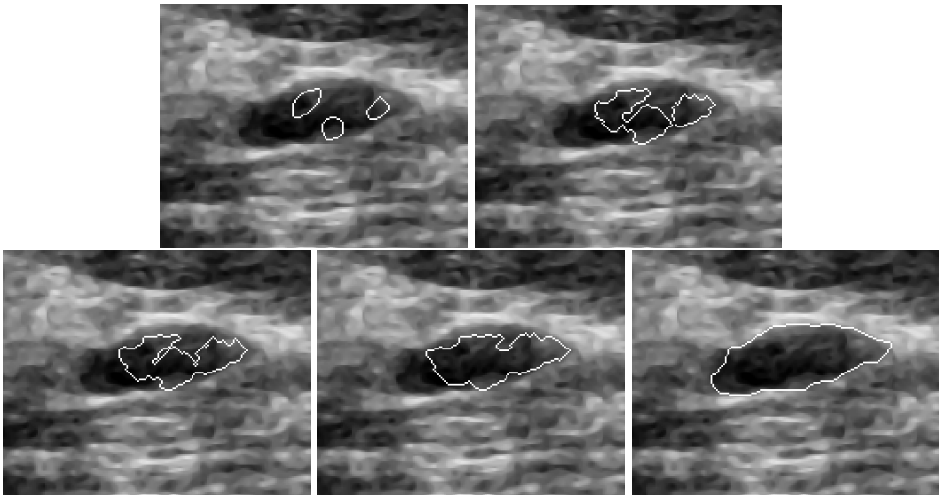


Figure 3.11: Contour extraction of cyst form ultrasound breast image via merging multiple initial level sets. Images courtesy of Yezzi [63], Georgia Institute of Technology.

To remedy the latter problem, Caselles et al. [26,64] and Kichenassamy et al. [63, 65] used an energy minimization formulation to design the speed function. This leads to the following geometric deformable contour formulation:

$$\frac{\partial \phi}{\partial t} = c(\kappa + V_0)|\nabla \phi| + \nabla c \cdot \nabla \phi \, . \tag{3.30}$$

Note that the resulting speed function has an extra stopping term $\nabla c \cdot \nabla \phi$ that can pull back the contour if it passes the boundary. This term behaves in similar fashion to the Gaussian potential force in the parametric formulation. An example of using this type of geometrical deformable contours is shown in Fig. 3.11.

The latter formulation can still generate curves that pass through boundary gaps. Siddiqi et al. [66] partially address this problem by altering the constant speed term through energy minimization, leading to the following geometric deformable contour:

$$\frac{\partial \phi}{\partial t} = \lambda(c\kappa|\nabla \phi| + \nabla c \cdot \nabla \phi) + (c + \frac{1}{2}\boldsymbol{X} \cdot \nabla c)|\nabla \phi| \, . \tag{3.31}$$

In this case, the constant speed term V_0 in Eq. (3.30) is replaced by the second term, and the term $\frac{1}{2}\boldsymbol{X} \cdot \nabla c$ provides additional stopping power that can prevent the geometrical contour from leaking through small boundary gaps. The second term can be used alone as the speed function for shape recovery as well. Figure 3.12 shows an example of this deformable contour model. Although this model is robust to small gaps, large boundary gaps can still cause problems.

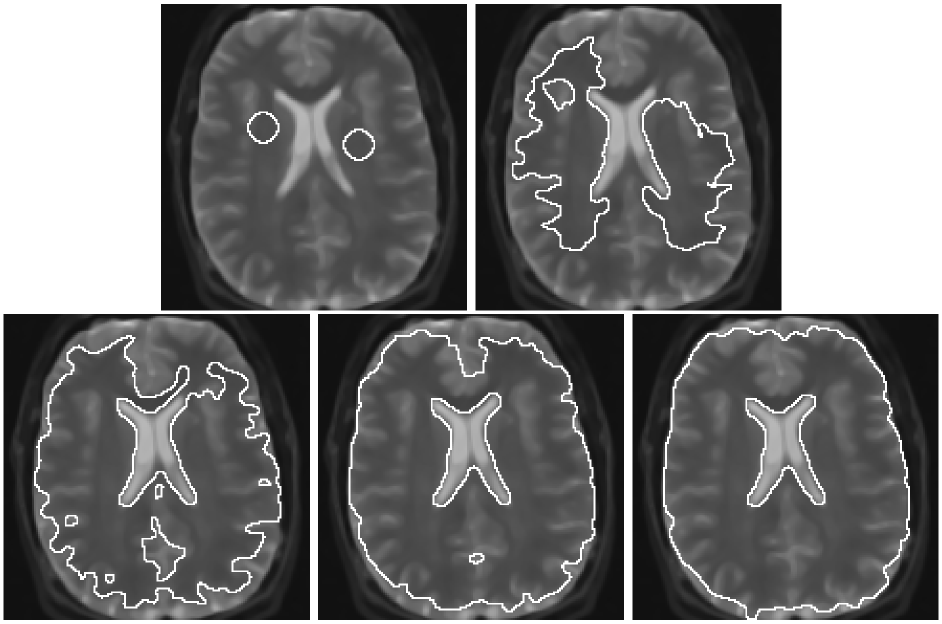


Figure 3.12: Segmentation of the brain using only the second term in (3.31). Left to right and top to bottom: iterations 1, 400, 800, 1200, and 1600. Images courtesy of Siddiqi [66], McGill University.

At this time, there is no geometric deformable contour model possessing the property of convergence to both perceptual boundaries (large boundary gaps) and boundary concavities as there are in parametric deformable contours [43].

3.3.4 Relationship to parametric deformable models

In the previous section, we described three types of geometric deformable contours that behave similarly to the parametric deformable contours but have the advantage of being able to change their topology automatically. The relationship between parametric deformable contours and geometric deformable contours can be formulated more precisely. Through an energy minimization formulation, Caselles et al. [64] showed that the geometric deformable contour in Eq. (3.30) is equivalent to the parametric deformable contour without the rigidity term. This derivation only permits the use of a speed function induced by a potential force, a property shared by almost all the geometric deformable models. In this section, we derive an explicit mathematical relationship between a dynamic force formulation of parametric deformable models and a geometric deformable model formulation, thus permitting the use of speed functions derived from nonpotential forces, i.e., forces that cannot be expressed as the negative gradient of potential energy functions.

For the convenience of derivation, we consider a simplified but more commonly used dynamic force formulation for parametric deformable contours:

$$\gamma \frac{\partial \boldsymbol{X}}{\partial t} = \alpha \frac{\partial^2 \boldsymbol{X}}{\partial s^2} + \boldsymbol{F}_p(\boldsymbol{X}) + \boldsymbol{F}_{\text{ext}}(\boldsymbol{X})\,. \tag{3.32}$$

Note that since the use of a pressure force $\boldsymbol{F}_p = w_p \boldsymbol{N}$ can cause singularities during deformation and requires special numerical implementation, we have separated it from the rest of the external forces. To represent Eq. (3.32) using a level set representation, we need to recast this formulation into the standard curve evolution form defined in Eq. (3.23). The corresponding geometric deformable contour in level set representation can then be obtained by using Eq. (3.26).

Since the contour's tangential motion only affects its parameterization but not its geometry, we modify Eq. (3.32) by considering only the normal components of internal and external forces. Given a parameterized curve $\boldsymbol{X}(s,t)$, where s is the arc-length parameterization of the curve, its inward unit normal $\boldsymbol{N}$ and curvature κ, we can use the fact that $\partial^2 \boldsymbol{X}/\partial s^2 = \kappa \boldsymbol{N}$ to rewrite Eq. (3.32) as follows:

$$\frac{\partial \boldsymbol{X}}{\partial t} = (\epsilon\kappa + V_p + \boldsymbol{V}_{\text{ext}} \cdot \boldsymbol{N})\boldsymbol{N}\,, \tag{3.33}$$

where $\epsilon = \alpha/\gamma$, $V_p = w_p/\gamma$, and $\boldsymbol{V}_{\text{ext}} = \boldsymbol{F}_{\text{ext}}/\gamma$. Here, we have divided through by γ so that both sides have units of velocity. If we let $V(\kappa) = \epsilon\kappa + V_p + \boldsymbol{V}_{\text{ext}} \cdot \boldsymbol{N}$, where $\boldsymbol{N}$ is given by Eq. (3.25), and substitute $V(\kappa)$ into Eq. (3.26), we obtain the following geometric deformable contour evolution equation:

$$\frac{\partial \phi}{\partial t} = V(\kappa)|\nabla\phi| = (\epsilon\kappa + V_p)|\nabla\phi| - \boldsymbol{V}_{\text{ext}} \cdot \nabla\phi\,. \tag{3.34}$$

If we allow both ϵ and V_p to be functions defined on the image domain, then Eq. (3.34) generalizes Eq. (3.31) and can be used to implement almost any parametric deformable model as a geometric deformable model.

3.3.5 Numerical implementation

In this section, we provide a numerical implementation that is adapted from [33] for Eq. (3.34), in which ϵ and V_p are allowed to be functions. The spatial derivatives are implemented using a special numerical scheme that can handle the formation of sharp corners during deformation. The numerical implementation is given as follows:

$$\begin{aligned} \phi_{ij}^{n+1} \quad = \quad & \phi_{ij}^{n} + \Delta t\{\epsilon\kappa_{ij}^{n}[(D_{ij}^{0x})^2 + (D_{ij}^{0y})^2]^{1/2} \\ & + \max(V_{\mathrm{p}\,ij}, 0)\nabla^{+} + \min(V_{\mathrm{p}\,ij}, 0)\nabla^{-} \\ & - [\max(u_{ij}^{n}, 0)D_{ij}^{-x} + \min(u_{ij}^{n}, 0)D_{ij}^{+x} \\ & + \max(v_{ij}^{n}, 0)D_{ij}^{-y} + \min(v_{ij}^{n}, 0)D_{ij}^{+y}]\}\,, \end{aligned} \tag{3.35}$$

where $\boldsymbol{V}_{\text{ext}} = (u, v)$, and $\kappa_{i,j}^n$ is the central difference approximation to the curvature expression given in Eq. (3.27). The first-order numerical derivatives and the gradient of the level set function ϕ are given by

$$\begin{aligned}
D_{ij}^{-x} &= \frac{\phi_{ij}^n - \phi_{i-1,j}^n}{\Delta x}, \quad D_{ij}^{+x} = \frac{\phi_{i+1,j}^n - \phi_{i,j}^n}{\Delta x}, \\
D_{ij}^{-y} &= \frac{\phi_{ij}^n - \phi_{i,j-1}^n}{\Delta y}, \quad D_{ij}^{+y} = \frac{\phi_{i,j+1}^n - \phi_{i,j}^n}{\Delta x}, \\
D_{ij}^{0x} &= \frac{\phi_{i+1,j}^n - \phi_{i-1,j}^n}{2\Delta x}, \quad D_{ij}^{0y} = \frac{\phi_{i,j+1}^n - \phi_{i,j-1}^n}{2\Delta y}, \\
\nabla^+ &= [\max(D_{ij}^{-x}, 0)^2 + \min(D_{ij}^{+x}, 0)^2 + \max(D_{ij}^{-y}, 0)^2 + \\
&\quad \min(D_{ij}^{+y}, 0)^2]^{1/2}, \\
\nabla^- &= [\max(D_{ij}^{+x}, 0)^2 + \min(D_{ij}^{-x}, 0)^2 + \max(D_{ij}^{+y}, 0)^2 + \\
&\quad \min(D_{ij}^{-y}, 0)^2]^{1/2}.
\end{aligned}$$

A detailed description of the principle behind this numerical method is described in [33]. We note that more efficient implementations of geometric deformable models have been developed, including the particularly noteworthy narrow-band level set method described in [25, 67].

3.3.6 Discussion

Although topological adaptation can be useful in many application, it can sometimes lead to undesirable results. Geometric deformable models may generate shapes that have inconsistent topology with respect to the actual object, when applied to noisy images with significant boundary gaps. In these situations, the significance of ensuring a correct topology is often a necessary condition for many subsequent applications. For example, in the brain functional study using fMRI or PET data, it is necessary to unfold the extracted cortical surface and create a flat or spherical map so that a user can visualize the functional activation in deep buried cortical regions (see [68, 69]). Parametric deformable models are better suited to these applications because of their strict control on topology.

3.4 Extensions of deformable models

Numerous extensions have been proposed to the deformable models described in the previous sections, particularly to extend the parametric deformable models. These extensions address two major areas for improving standard deformable models. The first area is the incorporation of additional prior knowledge into the models. Use of prior knowledge in a deformable model can lead to more robust and accurate results. This is especially true in applications where a particular structure that requires delineation has similar shape across a large number of subjects. Incorporation of prior knowledge requires a training step that involves manual interaction to accumulate information on the variability of the object shape being delineated. This information is then used to constrain the actual deformation of the contour or surface to extract shapes consistent with the training data.

The second area that has been addressed by various extensions of deformable models is in modeling global shape properties. Traditional parametric and geometric deformable models are local models — contours or surfaces are assumed to be locally smooth. Global properties such as orientation and size are not explicitly modeled. Modeling of global properties can provide greater robustness to initialization. Furthermore, global properties are important in object recognition and image interpretation applications because they can be characterized using only a few parameters. Note that although prior knowledge and global shape properties are distinct concepts, they are often used in conjunction with one another. Global properties tend to be much more stable than local properties. Therefore, if information about the global properties is known *a priori*, it can be used to greatly improve the performance of the deformable model.

In this section, we review several extensions of deformable models that use prior knowledge and/or global shape properties. We focus on revealing the fundamental principles of each extension and refer the reader to the cited literature for a full treatment of the topic.

3.4.1 Deformable Fourier models

In standard deformable models, a direct parameterization is typically utilized for representing curves and surfaces. Staib and Duncan [70] have proposed using a Fourier representation for parameterizing deformable contours and surfaces. A Fourier representation for a closed contour is expressed as

$$\boldsymbol{X}(s) = \begin{bmatrix} X(s) \\ Y(s) \end{bmatrix} = \begin{bmatrix} a_0 \\ c_0 \end{bmatrix} + \sum_{k=1}^{\infty} \begin{bmatrix} a_k & b_k \\ c_k & d_k \end{bmatrix} \begin{bmatrix} \cos 2\pi ks \\ \sin 2\pi ks \end{bmatrix}, \tag{3.36}$$

where $a_0, c_0, a_1, b_1, c_1, d_1, \ldots$ are Fourier coefficients. The Fourier coefficients of $X(s)$ are computed by

$$\begin{aligned} a_0 &= \frac{1}{2\pi} \int_0^1 X(s) ds \\ a_k &= \frac{1}{\pi} \int_0^1 X(s) \cos 2\pi ks \, ds \\ b_k &= \frac{1}{\pi} \int_0^1 X(s) \sin 2\pi ks \, ds \, , \end{aligned}$$

and the coefficients of $Y(s)$ are computed in analogous fashion. Open contours can also be parameterized using a straightforward modification of Eq. (3.36), as described in [70].

The advantages of the Fourier representation are that a compact representation of smooth shapes can be obtained by truncating the series and that a geometric description of the shape can be derived to characterize global shape properties. From Eq. (3.36), the coefficients a_0 and c_0 define the translation of the contour. Each subsequent term in the series expansion follows the parametric form of an ellipse. It is possible to map the coefficients to a parameter set that describes the object shape in terms of standard properties of ellipses [70]. Furthermore, like the Fourier coefficients, these parameters follow a scale ordering, where low index parameters describe global properties and higher indexed parameters describe more local deformations.

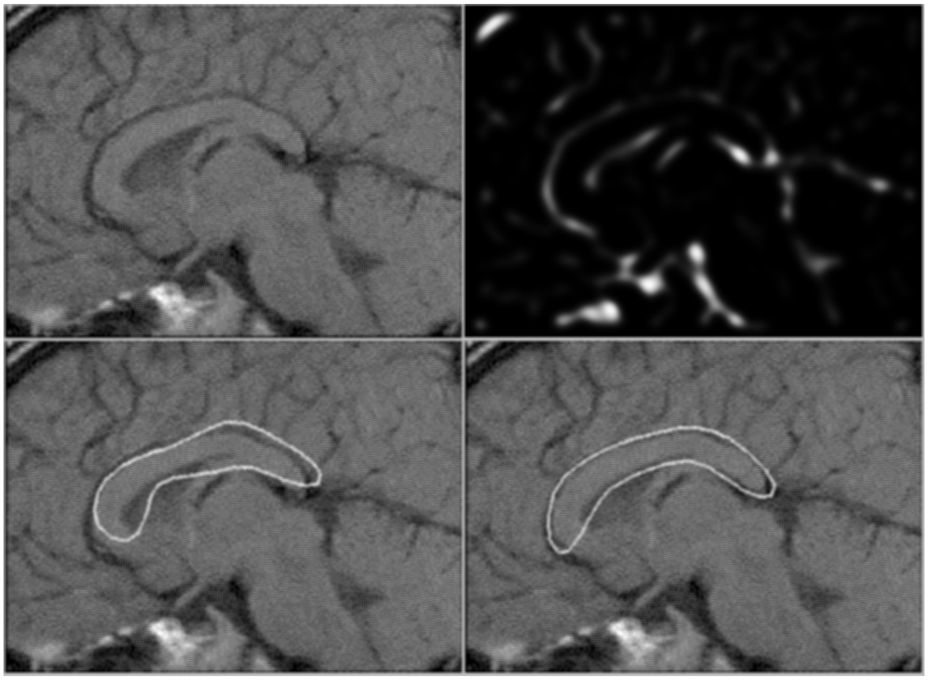


Figure 3.13: Segmenting the corpus callosum from an MR midbrain sagittal image using a deformable Fourier model. Top left: MR image (146×106). Top right: positive magnitude of the Laplacian of the Gaussian ($\sigma = 2.2$). Bottom left: initial contour (six harmonics). Bottom right: final contour on the corpus callosum of the brain. Images courtesy of Staib and Duncan [70], Yale University.

Staib and Duncan apply a Bayesian approach to incorporating prior information into their model. A prior probability function is defined by first manually or semi-automatically delineating structures of the same class as the structure to be extracted. Next, these structures are parameterized using the Fourier coefficients, or using the converted parameter set based on ellipses. Mean and variance statistics are finally computed for each of the parameters.

Assuming independence between the parameters, the multivariate Gaussian prior probability function is given by

$$Pr(\boldsymbol{p}) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma_i^2}} e^{-\frac{(p_i-\mu_i)^2}{2\sigma_i^2}} , \tag{3.37}$$

where $\boldsymbol{p} = (p_1, p_2, \cdots, p_N)$ is the parameter vector derived by truncating the Fourier coefficients[3], μ_i is the mean of the ith parameter in the training data, and σ_i^2 is the variance. A posterior probability function is defined that balances the prior probability model and a data model, which measures the discrepancy between boundary features in the image and the deformable contour. In [70], a gradient ascent method was used to maximize the posterior probability function. More recently, a genetic algorithm was proposed in [71]. Figure 3.13 shows an example of using the deformable Fourier model to recover the corpus callosum of the human brain.

3.4.2 Deformable models using modal analysis

Another way to restrict the mostly unstructured motion associated with the standard deformable model is to use modal analysis (Pentland and Horowitz [72], Nastar and Ayache [53]). This approach is similar to the deformable Fourier model except that both the basis functions and the nominal values of their coefficients are derived from a template object shape.

Deformable models based on modal analysis use the theory of finite elements [73]. An object is assumed to be represented by a finite set of *elements* whose positions are defined by the positions of n *nodes*, which are points in d-dimensional space. The node positions can be stacked into a vector $\boldsymbol{X}$, which has length nd, and element interpolation characterizes the complete object shape on the continuum. If the object moves or deforms, its new position is given by $\boldsymbol{X} + \boldsymbol{U}$, where $\boldsymbol{U}$ is a vector of length nd representing the collection of nodal displacements.

The equation governing the object's motion can be written as a collection of ordinary differential equations constraining the nodal displacements. This is compactly written as

$$\boldsymbol{M}\frac{d^2\boldsymbol{U}}{dt^2} + \boldsymbol{C}\frac{d\boldsymbol{U}}{dt} + \boldsymbol{K}\boldsymbol{U} = \boldsymbol{f} ,$$

where $\boldsymbol{M}$, $\boldsymbol{C}$, and $\boldsymbol{K}$ are the mass, damping, and stiffness matrices of the system and $\boldsymbol{f}$ is an nd-dimensional vector of external forces acting on the nodes. Both $\boldsymbol{U}$ and $\boldsymbol{f}$ are assumed to be functions of time. Derivation of $\boldsymbol{M}$, $\boldsymbol{C}$, and $\boldsymbol{K}$ are described in the literature (cf. Pentland and Horowitz [72], Terzopoulos and Metaxas [47]).

Solution of the generalized eigenvalue problem

$$\boldsymbol{K}\boldsymbol{\phi}_i = \omega_i^2 \boldsymbol{M}\boldsymbol{\phi}_i$$

[3] Two modified definitions for the parameter vector were also proposed in [70].

yields the *modes* $\boldsymbol{\phi}_i$ and eigenvalues ω_i, $i = 1, \dots, nd$. The nodal displacements can be written as

$$\boldsymbol{U} = \Phi\tilde{\boldsymbol{U}} \,,$$

where Φ is the (orthogonal) matrix whose columns comprise the modes and $\tilde{\boldsymbol{U}}$ is a vector of *motion coefficients*. The governing equation can then be written as

$$\frac{d^2\tilde{\boldsymbol{U}}}{dt^2} + \Phi^{\mathrm{T}}\boldsymbol{C}\Phi\frac{d\tilde{\boldsymbol{U}}}{dt} + \Omega^2\tilde{\boldsymbol{U}} = \Phi^{\mathrm{T}}\boldsymbol{f} \,, \tag{3.38}$$

where Ω^2 is a diagonal matrix having the eigenvalues corresponding to the modes on its diagonal. It is customary to assume the Rayleigh condition, which implies that $\Phi^{\mathrm{T}}\boldsymbol{C}\Phi$ is also a diagonal matrix. This decouples the equations in (3.38), leaving nd equations to solve for the nd motion coefficients.

Shape variations are constrained and computation times are reduced by approximating the nodal positions using only the pd lower-order modes (those corresponding to the larger eigenvalues in Ω^2). This is conceptually equivalent to keeping the lowest-order Fourier coefficients, but this approximation does not necessarily smooth the shape since sharp bends are still allowed if the lowest-order modes possess such bends. In this case, the nodal positions become

$$\boldsymbol{U} \approx \Phi_p\hat{\boldsymbol{U}} \,, \tag{3.39}$$

where Φ_p is the matrix consisting of the first pd columns of Φ, and $\hat{\boldsymbol{U}}$ is the vector comprising the first pd motion coefficients from $\tilde{\boldsymbol{U}}$. The governing equations become

$$\frac{d^2\hat{\boldsymbol{U}}}{dt^2} + \boldsymbol{C}_p\frac{d\hat{\boldsymbol{U}}}{dt} + \Omega_p^2\hat{\boldsymbol{U}} = \hat{\boldsymbol{f}} \,, \tag{3.40}$$

where $\boldsymbol{C}_p = \Phi_p^{\mathrm{T}}\boldsymbol{C}\Phi$, Ω_p^2 is the diagonal matrix comprising the first pd eigenvalues, and

$$\hat{\boldsymbol{f}} = \Phi_p^{\mathrm{T}}\boldsymbol{f} \,. \tag{3.41}$$

To implement a deformable model using reduced modal analysis, we assume that external forces $\boldsymbol{f}$ have been specified exactly as in standard deformable models (see Section 3.2.3). We also assume that the initial displacements (from the template object) are zero and the initial velocities are zero. Approximate integration can then be accomplished using the explicit Euler scheme:

$$\begin{aligned}
\frac{d^2}{dt^2}\hat{\boldsymbol{U}}(t) &= \hat{\boldsymbol{f}}(t) - \boldsymbol{C}_p\frac{d}{dt}\hat{\boldsymbol{U}}(t) - \Omega_p^2\hat{\boldsymbol{U}}(t) \\
\frac{d}{dt}\hat{\boldsymbol{U}}(t+\Delta t) &= \frac{d}{dt}\hat{\boldsymbol{U}}(t) + \Delta t\frac{d^2}{dt^2}\hat{\boldsymbol{U}}(t) \\
\hat{\boldsymbol{U}}(t+\Delta t) &= \hat{\boldsymbol{U}}(t) + \Delta t\frac{d}{dt}\hat{\boldsymbol{U}}(t+\Delta t) \,,
\end{aligned}$$

where Δt is a time step that must be chosen small enough for convergence and good accuracy. The nodal displacements are given by Eq. (3.39) and the nodal positions are given by $\boldsymbol{X} + \boldsymbol{U}$. Using this information, the vector of external forces can be recomputed from the image data after each time step. Solution of the explicit Euler scheme equations is particularly easy because the equations are decoupled, and it is very fast because only the pd retained modes need be computed.

3.4.3 Deformable superquadrics

Another extension of deformable models that has been used for incorporating local and global shape features is the *deformable superquadric*, proposed by Terzopoulos and Metaxas [47]. This is essentially a hybrid technique where a superquadric surface, which can be defined with a relatively small number of parameters, is allowed to deform locally for reconstructing the shape of an object. Although the fitting of global and local deformations is performed simultaneously, the global deformation is forced to account for as much of the object shape as possible. The estimated superquadric therefore captures the global shape characteristics and can readily be used in object recognition applications, while the local deformations capture the details of the object shape.

Terzopoulos and Metaxas consider models that are closed surfaces, denoted by $\boldsymbol{x}(\boldsymbol{u})$, where the parametric coordinates $\boldsymbol{u} = [u, v]$. This surface can be expressed as

$$\boldsymbol{x}(\boldsymbol{u}) = \boldsymbol{c} + \boldsymbol{R}\boldsymbol{p}(\boldsymbol{u}) \,, \tag{3.42}$$

where $\boldsymbol{c}$ is a translation vector, and $\boldsymbol{R}$ is a rotation matrix. The vector function $\boldsymbol{p}(\boldsymbol{u})$ denotes the model shape irrespective of pose and can further be expressed as

$$\boldsymbol{p}(\boldsymbol{u}) = \boldsymbol{s}(\boldsymbol{u}) + \boldsymbol{d}(\boldsymbol{u}) \,, \tag{3.43}$$

where $\boldsymbol{s}(\boldsymbol{u})$ is a reference shape consisting of the low parameter global shape model, and $\boldsymbol{d}(\boldsymbol{u})$ is a displacement function consisting of the local deformations.

The reference shapes in this case are superquadrics, which are an extension of standard quadric surfaces. These surfaces have been used in a variety of applications for computer graphics and computer vision, because of their ability to accommodate a large number of shapes with relatively few parameters. The kind of superquadric of interest here is the superellipsoid, which can be expressed implicitly as [74]

$$\left(\left(\left(\frac{x}{a_1}\right)^{2/\epsilon_2} + \left(\frac{y}{a_2}\right)^{2/\epsilon_2} \right)^{\epsilon_2/\epsilon_1} + \left(\frac{z}{a_3}\right)^{2/\epsilon_1} \right)^{\epsilon_1/2} = 1 \,, \tag{3.44}$$

where $0 \leq a_1, a_2, a_3 \leq 1$ are aspect ratio parameters, and $\epsilon_1, \epsilon_2 \geq 0$ control the "squareness" of the shape. Using a spherical coordinate reference frame, Terzopoulos and Metaxas employ the following expression for the superellipsoid:

$$\boldsymbol{s}(\boldsymbol{u}) = a \begin{bmatrix} a_1 C_u^{\epsilon_1} C_v^{\epsilon_2} \\ a_2 C_u^{\epsilon_1} S_v^{\epsilon_2} \\ a_3 S_u^{\epsilon_1} \end{bmatrix} , \tag{3.45}$$

where $-\pi/2 \leq u \leq \pi/2$, $-\pi \leq v \leq \pi$, $S_\eta^\epsilon = \operatorname{sgn}(\sin\eta)|\sin\eta|^\epsilon$, and $C_\eta^\epsilon = \operatorname{sgn}(\cos\eta)|\cos\eta|^\epsilon$. The parameter $a \geq 0$ controls the scale of the shape. Thus, the reference shape $\boldsymbol{s}(\boldsymbol{u})$ is characterized by a total of six parameters, which can be collected into a single vector $\boldsymbol{q}_s$:

$$\boldsymbol{q}_s = [a, a_1, a_2, a_3, \epsilon_1, \epsilon_2]^T . \tag{3.46}$$

The displacement function $\boldsymbol{d}(\boldsymbol{u})$ is decomposed into a linear combination of finite element basis functions and can be written as

$$\boldsymbol{d}(\boldsymbol{u}) = \boldsymbol{B}(\boldsymbol{u})\boldsymbol{q}_d , \tag{3.47}$$

where $\boldsymbol{B}$ is a matrix of the basis functions and $\boldsymbol{q}_d$ is a vector of the local deformation parameters [47].

We denote the vector of all the parameters required by the deformable superquadric to reconstruct a shape as $\boldsymbol{q}$, which consists of $\boldsymbol{q}_s$, $\boldsymbol{q}_d$, as well as the rotation and translation parameters of the model from Eq. (3.42). Terzopoulos and Metaxas use a physics-based model based on the traditional parametric deformable model to introduce a time variable and model the deformation process (see Section 3.2.2). Given some initialization for $\boldsymbol{q}$, a simplified dynamic force equation can be written as

$$\boldsymbol{C}\frac{\partial \boldsymbol{q}}{\partial t} + \boldsymbol{K}\boldsymbol{q} = \boldsymbol{f}_q , \tag{3.48}$$

where the first term represents damping forces controlled by the damping matrix $\boldsymbol{C}$, the second term represents internal forces of the model controlled by the stiffness matrix $\boldsymbol{K}$, and $\boldsymbol{f}_q$ are the external forces. As with the parametric deformable model, the model deforms according to Eq. (3.48) until these forces reach equilibrium.

An important aspect in such a hybrid model is that the global reference shape should account for as much of the shape to be reconstructed as possible. This is accomplished in Eq. (3.48) by appropriately defining the stiffness matrix $\boldsymbol{K}$. In particular, all entries of $\boldsymbol{K}$ that do not correspond to local deformations are set to zero. This amounts to imposing no penalty on the evolution of the rotation, translation, and superquadric parameters. On the other hand, entries corresponding to the local deformation parameters are selected such that their evolution is restricted with respect to their magnitude and first derivative.

As with traditional parametric deformable models, the deformable superquadric is also well suited to motion estimation tasks, as is described in [75]. For this reason, a popular application of models based on superquadrics has been in cardiac imaging [74], where the simple shape of the heart can be readily modeled by a superellipsoid. The deformable superquadric model has been extended by Vemuri and Radisavljevic [76], who employed a wavelet parameterization of the local deformation process. The multiresolution nature of the wavelet decomposition allows for a smooth transition between the global superquadric and the local descriptors. They also present a method utilizing training data for obtaining a prior model of the global parameters. The deformable superquadric model has also been adapted to multilevel shape representation [77].

3.4.4 Active shape models

Active shape models (ASMs) proposed by Cootes et al. [78, 79] use a different approach to incorporate prior shape information. Their prior models are not based on the parameterization, but are instead based on a set of points defined at various features in the image. In the following, we summarize how the prior model is constructed and used to enhance the performance of a deformable model and how the ASM paradigm can be extended to incorporate prior information on the image intensity rather than on the shape alone.

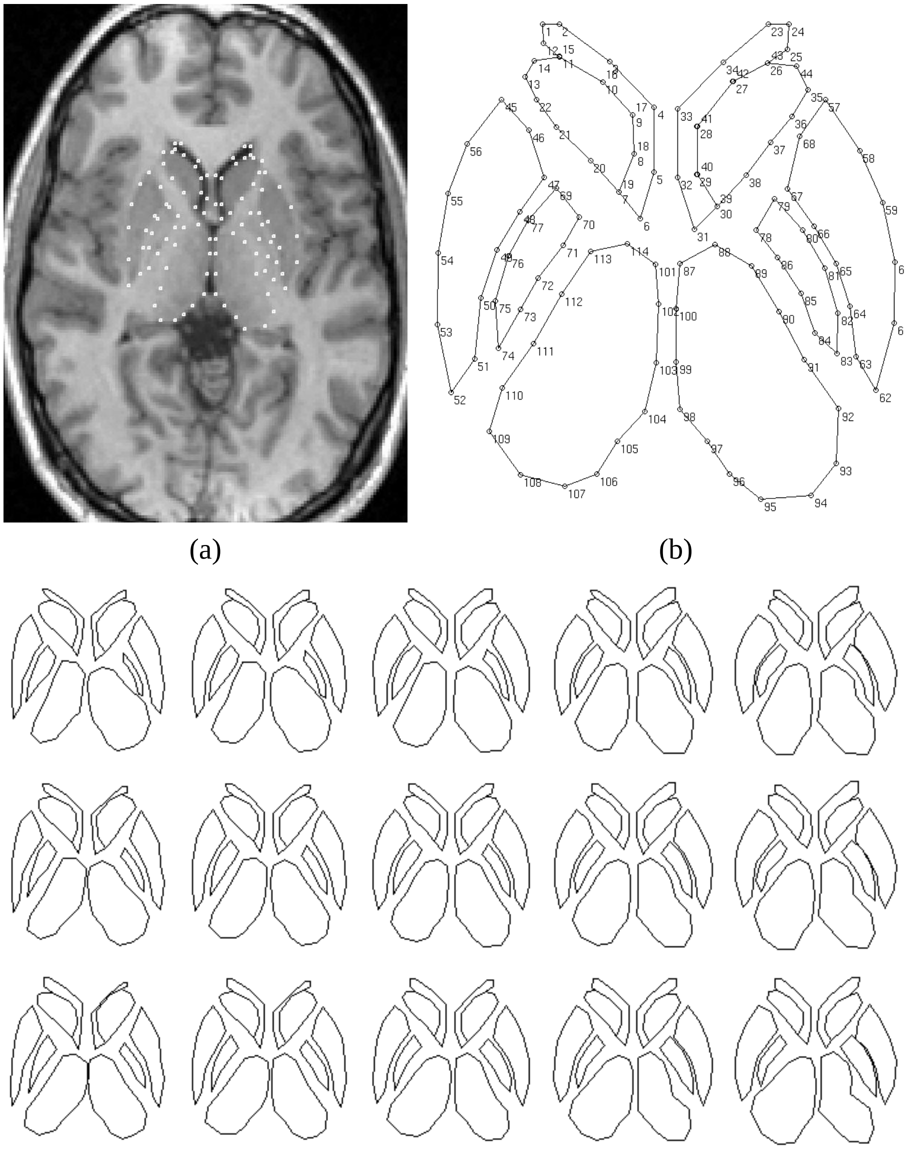
Construction of the ASM prior model

The ASM prior model is constructed by first establishing a set of labeled point features, or landmarks, within the class of images to be processed [see Figs. 3.14(a) and (b)]. These points are manually selected on each of the images in the training set[4]. Once selected, the set of points for each image is aligned to one another with respect to translation, rotation, and scaling. This is accomplished using an iterative algorithm based on the Procrustes method [80]. This linear alignment allows studying the object shape in a common coordinate frame, which we will refer to as the model space of the ASM. After the alignment, there is typically still a substantial amount of variability in the coordinates of each point. To compactly describe this variability as a prior model, Cootes and Taylor developed the *Point Distribution Model* (PDM), which we now describe.

Given N aligned shapes $\boldsymbol{Y}_1, \boldsymbol{Y}_2, \cdots, \boldsymbol{Y}_N$ in the model space, where $\boldsymbol{Y}_i = (x_{i0}, y_{i0}, \cdots, x_{in-1}, y_{in-1})^T$ is a $2n$-dimensional vector describing the coordinates of the n points from the ith shape, the mean shape, $\bar{\boldsymbol{Y}}$, is defined to be

$$\bar{\boldsymbol{Y}} = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{Y}_i \,. \tag{3.49}$$

[4] See the remarks at the end of this section for recent work on automated landmark labeling.



(a)

(b)

(c)

Figure 3.14: An example of constructing Point Distribution Models. (a) An MR brain image, transaxial slice, with 114 landmark points of deep neuroanatomical structures superimposed. (b) A 114-point shape model of 10 brain structures. (c) Effect of simultaneously varying the model's parameters corresponding to the first two largest eigenvalues (on a bi-dimensional grid). Images courtesy of Duta and Sonka [81], The University of Iowa.

A covariance matrix, $\boldsymbol{S}$, is computed by

$$\boldsymbol{S} = \frac{1}{N-1}\sum_{i=1}^{N}(\boldsymbol{Y}_i - \bar{\boldsymbol{Y}})(\boldsymbol{Y}_i - \bar{\boldsymbol{Y}})^T \,. \tag{3.50}$$

The eigenvectors corresponding to the largest eigenvalues of the covariance matrix describe the most significant modes of variation. Because almost all of the variability in the model can be described using these eigenvectors, only m such eigenvectors are selected to characterize the entire variability of the training set. Note that in general, m is significantly smaller than the number of points in the model.

Using a *principal component analysis* (PCA), any shape $\boldsymbol{Y}$ in the training set can be approximated by

$$\boldsymbol{Y} \approx \bar{\boldsymbol{Y}} + \boldsymbol{P}\boldsymbol{b} \,, \tag{3.51}$$

where $\boldsymbol{P} = (\boldsymbol{p}_1\boldsymbol{p}_2 \cdots \boldsymbol{p}_m)$ is the matrix of the first m eigenvectors, and $\boldsymbol{b} = (b_1 b_2 \cdots b_m)^T$ is a vector of weights, referred to as the *shape parameters*. The change of shape can be made by varying $\boldsymbol{b}$ accordingly. Limits on the values of $\boldsymbol{b}$ are imposed to constrain the actual amount of deviation from the mean shape. Figure 3.14(c) shows a collection of shapes generated for several subcortical structures from similar transaxial MR brain images by using the two most significant eigenvectors.

Model fitting procedure

The key idea of ASMs is to constrain the behavior of deformable models using the PDM obtained as described in the previous section (cf. [79, 81, 82]). At each iteration, a standard deformation of the parametric deformable model is approximated by adjusting both the pose (translation, rotation, and scale) parameters and the shape parameters of the model instance. Thus, only deformations that produce shapes similar to those in the training set are allowed. The iteration stops when changes in both the pose and shape parameters are insignificant. Figure 3.15 shows an example of using active shape models to extract the heart wall from an ultrasound image.

Let us denote the position of the model instance at the beginning of a deformation step as $\boldsymbol{X} = (X_0, Y_0, \cdots, X_{n-1}, Y_{n-1})^T$, and the required deformation computed from both internal and external forces as a displacement vector $d\boldsymbol{X} = (dX_0, dY_0, \cdots, dX_{n-1}, dY_{n-1})^T$. Then the position of the model instance, $\boldsymbol{X}$, can be compactly represented by its pose and shape parameters, i.e.,

$$\boldsymbol{Y} = \bar{\boldsymbol{Y}} + \boldsymbol{P}\boldsymbol{b}, \quad \text{and} \tag{3.52}$$

$$\boldsymbol{X} = M(s,\theta)[\boldsymbol{Y}] + \boldsymbol{X}_c \,, \tag{3.53}$$

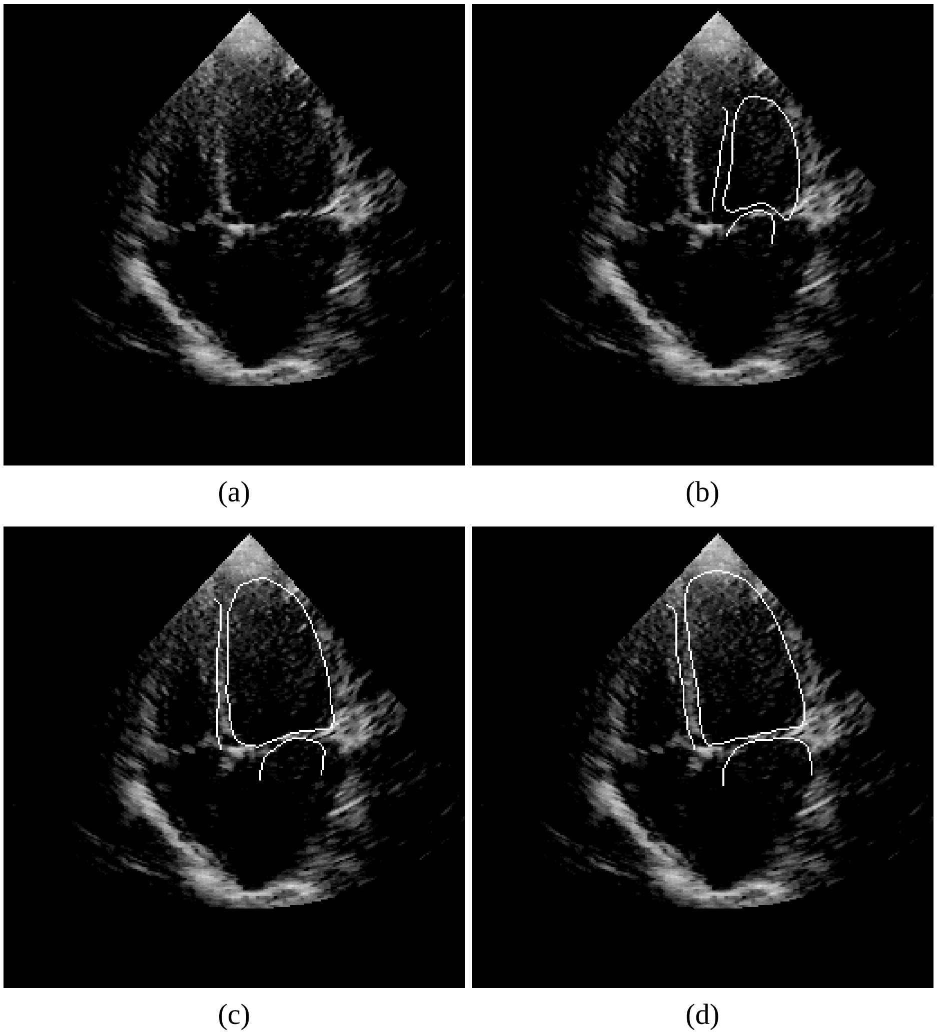


Figure 3.15: An example of Active Shape Models. (a) An echocardiogram image. (b) The initial position of the heart chamber boundary model. The location of the model after (c) 80 and (d) 200 iterations. Images courtesy of Cootes et al. [79], The University of Manchester.

where s is the scaling factor, θ is the rotation angle, $M(s, \theta)[\boldsymbol{Y}]$ is a linear transformation that performs scaling and rotation on $\boldsymbol{Y}$, and $\boldsymbol{X}_c = (X_c, Y_c)$ is the center of the model instance.

First, a global fit is performed by adjusting the pose parameters so that the generated model instance aligns best with the expected model instance $\boldsymbol{X} + d\boldsymbol{X}$. The proper pose parameter adjustments, ds, $d\theta$, and $d\boldsymbol{X}_c$, can be estimated efficiently using a standard least-squares approach (see [79] for details).

After adjusting the pose parameters, the remaining difference between the generated and expected model instances can be explained by varying the shape parameters. To calculate the adjustment to the shape parameters, first we need to find the corresponding residue, $d\boldsymbol{Y}$, in the model space, which is required to satisfy the following constraint:

$$\boldsymbol{X} + d\boldsymbol{X} = M(s(1+ds), \theta + d\theta)[\boldsymbol{Y} + d\boldsymbol{Y}] + \boldsymbol{X}_c + d\boldsymbol{X}_c \, . \tag{3.54}$$

Solving the above equation for $d\boldsymbol{Y}$ yields

$$d\boldsymbol{Y} = M(s^{-1}(1+ds)^{-1}, \theta - d\theta)[\boldsymbol{Z}] - \boldsymbol{X} \, , \tag{3.55}$$

where $\boldsymbol{Z} = M(s,\theta)[\boldsymbol{Y}] + d\boldsymbol{X} - d\boldsymbol{X}_c$. Note that to derive Eq. (3.55), both Eq. (3.52) and $M^{-1}(s,\theta)[\;] = M(s^{-1}, -\theta)[\;]$ are used.

Having solved for $d\boldsymbol{Y}$, we next find the adjustments, $d\boldsymbol{b}$, to the shape parameters such that

$$\boldsymbol{Y} + d\boldsymbol{Y} \approx \bar{\boldsymbol{Y}} + \boldsymbol{P}(\boldsymbol{b} + d\boldsymbol{b}) \, . \tag{3.56}$$

A solution can be obtained using a least-squares approximation [83], yielding

$$d\boldsymbol{b} = \boldsymbol{P}^T d\boldsymbol{Y} \, . \tag{3.57}$$

Note that $\boldsymbol{P}^T\boldsymbol{P} = \boldsymbol{I}$.

To summarize, an iteration step of the ASM consists of first finding a displacement of the model instance in the image space, then calculating the corresponding adjustments to both the pose and shape parameters, and updating the parameters accordingly. Note that in practice, weighted adjustments are usually used to update both the pose and shape parameters [79]. When the shape parameters are updated, their values are limited within a specified range so that the shape of the model instance remains similar to the shapes of the training examples.

Active appearance models

A limitation of the ASM is that its prior model does not consider gray-level variation of the object instance across images. To overcome this difficulty, Edwards, Cootes, and Taylor [84–86] proposed an extension to the ASM, called *active appearance models* (AAM). In AAM, a new prior model is constructed using both shape and grey-level information. Because the objects represented by AAMs are more specific than those represented by ASMs, in many applications, AAMs can lead to more robust results than ASMs.

We will now describe how AAMs are constructed. First, the shape difference of each object instance is compensated by warping the instance image in such a way that the warped instance shape matches the mean shape obtained through the PDM procedure of the ASM. The warping step is implemented using a triangulation algorithm (see [87]). The resulting shape-normalized images can then be used to analyze grey-level variations seen from various example images.

Next, a PCA is applied to the shape-normalized images, yielding a linear model that characterizes the grey-level variation, i.e.,

$$\boldsymbol{g} = \bar{\boldsymbol{g}} + \boldsymbol{P}_g \boldsymbol{b}_g \,, \tag{3.58}$$

where $\bar{\boldsymbol{g}}$ is the mean normalized gray-level vector, $\boldsymbol{P}_g$ is a matrix consisting of significant modes of gray-level variations, and $\boldsymbol{b}_g$ is the gray-level parameters that weight the contribution from different modes of gray-level variations in $\boldsymbol{P}_g$. As described previously in Eq. (3.52), the instance shape is given by

$$\boldsymbol{Y} = \bar{\boldsymbol{Y}} + \boldsymbol{P}_s \boldsymbol{b}_s \,. \tag{3.59}$$

Here, for consistency with Eq. (3.58), $\boldsymbol{P}_s$ and $\boldsymbol{b}_s$ are used to denote the significant modes of shape variation and the shape parameters, respectively. Thus, given any instance image of the object of interest, its shape and gray-level pattern can be represented compactly using the vectors $\boldsymbol{b}_s$ and $\boldsymbol{b}_g$.

Because the shape and grey-level parameters may be correlated, a further PCA is applied to these combined shape and grey-level vectors $\boldsymbol{b} = [\boldsymbol{W}_s \boldsymbol{b}_s, \boldsymbol{b}_g]^T$, where $\boldsymbol{W}_s$ is a diagonal matrix of weights to compensate the difference in units between the shape and grey-level parameters. The PCA yields another linear model

$$\boldsymbol{b} = \boldsymbol{Q}\boldsymbol{c} = \begin{pmatrix} \boldsymbol{Q}_s \\ \boldsymbol{Q}_g \end{pmatrix} \boldsymbol{c} \,, \tag{3.60}$$

where $\boldsymbol{Q}$ is a set of orthogonal modes, Q_s and Q_g are the corresponding submatrices for the shape and gray-level parameters, respectively, and $\boldsymbol{c}$ is referred to as the *appearance parameters* that regulate the variations of both the shape and gray-level pattern of the model.

The final representation of the shape and graylevels in terms of $\boldsymbol{c}$ is given by

$$\boldsymbol{Y} = \bar{\boldsymbol{Y}} + \boldsymbol{P}_s \boldsymbol{W}_s^{-1} \boldsymbol{Q}_s \boldsymbol{c} \tag{3.61}$$

$$\boldsymbol{g} = \bar{\boldsymbol{g}} + \boldsymbol{P}_g \boldsymbol{Q}_g \boldsymbol{c} \,. \tag{3.62}$$

Despite the fact that the number of the appearance parameters is less than the total number of the parameters in the original gray-level vector, matching the appearance model to an unseen image can be a time-consuming task. In [85], Cootes, Edwards, and Taylor proposed a fast matching algorithm that first learns a linear relationship between matching errors and desired parameter adjustments from training examples, then uses this information to predict the parameter adjustments in the real matching process.

Remarks

In addition to the AAM extension to the ASM, there are many other extensions. Duta and Sonka [81] applied the ASM to segment subcortical structures from MR brain images. They improved the overall reconstruction accuracy of the ASM algorithm by incorporating an outlier-detection algorithm during each deformation step. Wang and Staib [82] incorporated an additional smoothness prior into the PDM models to allow the generation of more flexible shape instances. They reformulated the ASM as a Bayesian problem and solved the problem by maximizing the *a posteriori* probability.

A major limitation of the ASM is the requirement to place landmarks on the training images. This procedure is a laborious task for annotating 2D images and becomes even more demanding for annotating 3D images. This limitation, however, has been partially alleviated by the recent automatic labeling work [88–90].

3.4.5 Other models

Additional extensions have also been proposed to use global shape information or prior shape information. For example, Ip and Shen [91] incorporated prior shape information by using an affine transformation to align a shape template with the deformable model and guide the model's deformation to produce a shape consistent with the template.

The deformable Fourier model, active shape model, and other extensions we discussed so far are all parametric deformable models. Guo and Vemuri [92] have proposed a framework for incorporating global shape prior information into geometric deformable models. Like the deformable superquadric, their *hybrid geometric deformable model* uses a combination of an underlying, low parameter, generator shape that is allowed to evolve. Their model thus retains the advantages of traditional geometric deformable models, such as topological adaptivity.

External forces for deformable models are typically defined from edges in the image. Fritsch et al. [93] have developed a technique called *deformable shape loci*, which uses information on the medial loci or *cores* of the shapes to be extracted (see Section 14.3.11). The incorporation of cores provides greater robustness to image disturbances such as noise and blurring than purely edge-based models. This allows their model to be fairly robust to initialization as well as imaging artifacts. They also employed a probabilistic prior model for important shape features as well as for the spatial relationships between these features.

3.5 Conclusion and future directions

In this chapter, we have described the fundamental formulation of both parametric and geometric deformable models and shown that they can be used in recovering shape boundaries. We have also derived an explicit mathematical relationship between these two formulations that allows one to share the design of external forces and speed functions. This may lead to new, improved deformable models. Finally, we give a brief overview of several important extensions of deformable models that use application-specific prior knowledge and/or global shape properties to obtain more robust and accurate results.

We expect that further improvements in deformable models will be made by the continued research in both external force and speed function design, model representation, model training and learning, and model performance validation. Another challenging research direction is to develop deformable models that have a greater control in topology. For example, models that can both constrain or change topology depending on the requirements of an application would be extremely useful. Promising approaches have been proposed recently, such as the work by McInerney and Terzopoulos [94], who developed a hybrid method that maintains both implicit and explicit representation for a given model to allow more effective control of the topology. Finally, integrating deformable models with existing medical systems, such as surgical simulation, planning, and treatment systems, can further validate the application of deformable models in a clinical setting and may in turn stimulate the development of better deformable models.

3.6 Further reading

Several current texts deal with deformable models. The book by Blake and Yuille [95] contains an excellent collection of papers on the theory and practice of deformable models. Application of deformable models in motion tracking is covered in great depth in two recent books [96] by Metaxas and [97] by Blake and Isard, respectively. The book edited by Singh, Goldgolf, and Terzopoulos [98] consists of a valuable collection of papers on deformable models and their application in medical image analysis. The book by Sethian [33] on level set methods is a comprehensive resource for geometric deformable models. A recent survey paper by McInerney and Terzopoulos [99] provides an excellent source for learning the application of deformable models in medical image analysis.

3.7 Acknowledgments

The authors would like to thank Milan Sonka, Michael Fitzpartrick, and David Hawkes for reading and commenting upon the draft of this chapter. The work was supported in part by an NSF Presidential Faculty Grant (MIP93-50336) and an NIH Grant (R01NS37747).

3.8 References

[1] S. M. Larie and S. S. Abukmeil, “Brain abnormality in schizophrenia: a systematic and quantitative review of volumetric magnetic resonance imaging studies,” *J. Psych.*, vol. 172, pp. 110–120, 1998.

[2] P. Taylor, “Invited review: computer aids for decision-making in diagnostic radiology — a literature review,” *Brit. J. Radiol.*, vol. 68, pp. 945–957, 1995.

[3] A. P. Zijdenbos and B. M. Dawant, “Brain segmentation and white matter lesion detection in MR images,” *Critical Reviews in Biomedical Engineering*, vol. 22, pp. 401–465, 1994.

[4] A. J. Worth, N. Makris, V. S. Caviness, and D. N. Kennedy, “Neuroanatomical segmentation in MRI: technological objectives,” *Int’l J. Patt. Recog. Artificial Intell.*, vol. 11, pp. 1161–1187, 1997.

[5] C. A. Davatzikos and J. L. Prince, “An active contour model for mapping the cortex,” *IEEE Trans. Med. Imag.*, vol. 14, pp. 65–80, 1995.

[6] V. S. Khoo, D. P. Dearnaley, D. J. Finnigan, A. Padhani, S. F. Tanner, and M. O. Leach, “Magnetic resonance imaging (MRI): considerations and applications in radiotheraphy treatment planning,” *Radiother. Oncol.*, vol. 42, pp. 1–15, 1997.

[7] H. W. Muller-Gartner, J. M. Links, J. L. Prince, R. N. Bryan, E. McVeigh, J. P. Leal, C. Davatzikos, and J. J. Frost, “Measurement of radiotracer concentration in brain gray matter using positron emission tomography: MRI-based correction for partial volume effects,” *J. Cereb. Blood Flow Metab.*, vol. 12, pp. 571–583, 1992.

[8] N. Ayache, P. Cinquin, I. Cohen, L. Cohen, F. Leitner, and O. Monga, “Segmentation of complex three-dimensional medical objects: a challenge and a requirement for computer-assisted surgery planning and performance,” in *Computer-Integrated Surgery: Technology and Clinical Applications* (R. H. Taylor, S. Lavallee, G. C. Burdea, and R. Mosges, eds.), pp. 59–74, MIT Press, 1996.

[9] W. E. L. Grimson, G. J. Ettinger, T. Kapur, M. E. Leventon, W. M. Wells, *et al.*, “Utilizing segmented MRI data in image-guided surgery,” *Int’l J. Patt. Recog. Artificial Intell.*, vol. 11, pp. 1367–1397, 1997.

[10] C. Xu and J. L. Prince, “Generalized gradient vector flow external forces for active contours,” *Signal Processing — An International Journal*, vol. 71, no. 2, pp. 131–139, 1998.

[11] C. Xu, D. L. Pham, M. E. Rettmann, D. N. Yu, and J. L. Prince, “Reconstruction of the human cerebral cortex from magnetic resonance images,” *IEEE Trans. Med. Imag.*, vol. 18, pp. 467–480, 1999.

[12] D. Terzopoulos, “On matching deformable models to images.” Technical Report 60, Schlumberger Palo Alto research, 1986. Reprinted in *Topical Meeting on Machine Vision*, Technical Digest Series, Vol. 12, 1987, 160-167.

[13] M. Kass, A. Witkin, and D. Terzopoulos, “Snakes: active contour models,” *Int’l J. Comp. Vis.*, vol. 1, no. 4, pp. 321–331, 1987.

[14] D. Terzopoulos and K. Fleischer, “Deformable models,” *The Visual Computer*, vol. 4, pp. 306–331, 1988.

[15] D. Terzopoulos, A. Witkin, and M. Kass, “Constraints on deformable models: recovering 3D shape and nonrigid motion,” *Artificial Intelligence*, vol. 36, no. 1, pp. 91–123, 1988.

[16] M. A. Fischler and R. A. Elschlager, “The representation and matching of pictorial structures,” *IEEE Trans. on Computers*, vol. 22, no. 1, pp. 67–92, 1973.

[17] B. Widrow, “The “rubber-mask” technique,” *Pattern Recognition*, vol. 5, pp. 175–211, 1973.

[18] A. Blake and A. Zisserman, *Visual Reconstruction*. Boston: MIT Press, 1987.

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

[20] M. I. Miller, G. E. Christensen, Y. Amit, and U. Grenander, “Mathematical textbook of deformable neuroanatomies,” *Proc. National Academy of Science*, vol. 90, pp. 11944–11948, 1993.

[21] A. A. Amini, T. E. Weymouth, and R. C. Jain, “Using dynamic programming for solving variational problems in vision,” *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 12, no. 9, pp. 855–867, 1990.

[22] L. D. Cohen, “On active contour models and balloons,” *CVGIP: Imag. Under.*, vol. 53, no. 2, pp. 211–218, 1991.

[23] T. McInerney and D. Terzopoulos, “A dynamic finite element surface model for segmentation and tracking in multidimensional medical images with application to cardiac 4D image analysis,” *Comp. Med. Imag. Graph.*, vol. 19, no. 1, pp. 69–83, 1995.

[24] V. Caselles, F. Catte, T. Coll, and F. Dibos, “A geometric model for active contours,” *Numerische Mathematik*, vol. 66, pp. 1–31, 1993.

[25] R. Malladi, J. A. Sethian, and B. C. Vemuri, “Shape modeling with front propagation: a level set approach,” *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 17, no. 2, pp. 158–175, 1995.

[26] V. Caselles, R. Kimmel, and G. Sapiro, “Geodesic active contours,” in *Proc. 5th Int’l Conf. Comp. Vis.*, pp. 694–699, 1995.

[27] R. T. Whitaker, “Volumetric deformable models: active blobs,” Tech. Rep. ECRC-94-25, European Computer-Industry Research Centre GmbH, 1994.

[28] G. Sapiro and A. Tannenbaum, “Affine invariant scale-space,” *Int’l J. Comp. Vis.*, vol. 11, no. 1, pp. 25–44, 1993.

[29] B. B. Kimia, A. R. Tannenbaum, and S. W. Zucker, “Shapes, shocks, and deformations I: the components of two-dimensional shape and the reaction-diffusion space,” *Int’l J. Comp. Vis.*, vol. 15, pp. 189–224, 1995.

[30] R. Kimmel, A. Amir, and A. M. Bruckstein, “Finding shortest paths on surfaces using level sets propagation,” *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 17, no. 6, pp. 635–640, 1995.

[31] L. Alvarez, F. Guichard, P. L. Lions, and J. M. Morel, “Axioms and fundamental equations of image processing,” *Archive for Rational Mechanics and Analysis*, vol. 123, no. 3, pp. 199–257, 1993.

[32] S. Osher and J. A. Sethian, “Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations,” *J. Computational Physics*, vol. 79, pp. 12–49, 1988.

[33] J. A. Sethian, *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Material Science*. Cambridge, UK: Cambridge University Press, 2nd ed., 1999.

[34] I. Cohen, L. D. Cohen, and N. Ayache, “Using deformable surfaces to segment 3-D images and infer differential structures,” *CVGIP: Imag. Under.*, vol. 56, no. 2, pp. 242–263, 1992.

[35] R. Courant and D. Hilbert, *Methods of Mathematical Physics*, vol. 1. New York: Interscience, 1953.

[36] J. L. Prince and C. Xu, “Nonconservative force models in active geometry,” in *Proc. IEEE Image and Multidimensional Signal Processing Workshop (IMDSP’98)*, pp. 139–142, 1998.

[37] R. Ronfard, “Region-based strategies for active contour models,” *Int’l J. Comp. Vis.*, vol. 13, no. 2, pp. 229–251, 1994.

[38] C. S. Poon and M. Braun, “Image segmentation by a deformable contour model incorporating region analysis,” *Phys. Med. Biol.*, vol. 42, pp. 1833–1841, 1997.

[39] H. Tek and B. B. Kimia, “Volumetric segmentation of medical images by three-dimensional bubbles,” *Comp. Vis. Imag. Under.*, vol. 65, pp. 246–258, 1997.

[40] L. D. Cohen and I. Cohen, “Finite-element methods for active contour models and balloons for 2-D and 3-D images,” *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 15, no. 11, pp. 1131–1147, 1993.

[41] P. E. Danielsson, “Euclidean distance mapping,” *Comp. Graph. Imag. Proc.*, vol. 14, pp. 227–248, 1980.

[42] G. Borgefors, “Distance transformations in arbitrary dimensions,” *Comp. Vis. Graph. Imag. Proc.*, vol. 27, pp. 321–345, 1984.

[43] C. Xu and J. L. Prince, “Snakes, shapes, and gradient vector flow,” *IEEE Trans. Imag. Proc.*, vol. 7, no. 3, pp. 359–369, 1998.

[44] H. Delingette, “Simplex meshes: a general representation for 3D shape reconstruction,” Tech. Rep. TR2214, INRIA, Sophia-Antipolis, France, 1994.

[45] D. MacDonald, D. Avis, and A. C. Evans, “Multiple surface identification and matching in magnetic resonance images,” in *SPIE Proc. Visualization in Biomedical Computing*, vol. 2359, pp. 160–169, 1994.

[46] D. J. Williams and M. Shah, “A fast algorithm for active contours and curvature estimation,” *CVGIP: Imag. Under.*, vol. 55, no. 1, pp. 14–26, 1992.

[47] D. Terzopoulos and D. Metaxas, “Dynamic 3D models with local and global deformations: deformable superquadrics,” *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 13, pp. 703–714, 1991.

[48] A. Gupta, L. von Kurowski, A. Singh, D. Geiger, C.-C. Liang, M.-Y. Chiu, L. P. Adler, M. Haacke, and D. L. Wilson, “Cardiac MR image segmentation using deformable models,” in *Proc. IEEE Conf. Computers in Cardiology*, pp. 747–750, 1993.

[49] H. Delingette, “Adaptive and deformable models based on simplex meshes,” in *Proc. IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, pp. 152–157, 1994.

[50] S. Kumar and D. Goldgof, “Automatic tracking of SPAMM grid and the estimation of deformation parameters from cardiac MR images,” *IEEE Trans. Med. Imag.*, vol. 13, pp. 122–132, 1994.

[51] D. Geiger, A. Gupta, L. A. Costa, and J. Vlontzos, “Dynamic programming for detecting, tracking, and matching deformable contours,” *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 17, pp. 294–402, 1995.

[52] S. Lobregt and M. A. Viergever, “A discrete dynamic contour model,” *IEEE Trans. Med. Imag.*, vol. 14, pp. 12–24, 1995.

[53] C. Nastar and N. Ayache, “Frequency-based nonrigid motion analysis: application to four dimensional medical images,” *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 18, pp. 1067–1079, 1996.

[54] R. Durikovic, K. Kaneda, and H. Yamashita, “Dynamic contour: a texture approach and contour operations,” *The Visual Computer*, vol. 11, pp. 277–289, 1995.

[55] T. McInerney and D. Terzopoulos, “Topologically adaptable snakes,” in *Proc. 5th Int’l Conf. Comp. Vis.*, pp. 840–845, 1995.

[56] B. B. Kimia, *Conservation Laws and a Theory of Shape*. Ph.D. thesis, McGill Centre for Intelligent Machines, McGill University, Montreal, Canada, 1990.

[57] M. A. Grayson, “Shortening embedded curves,” *Annals of Mathematics*, vol. 129, pp. 71–111, 1989.

[58] J. A. Sethian, “Curvature and evolution of fronts,” *Commun. Math. Phys.*, vol. 101, pp. 487–499, 1985.

[59] J. A. Sethian, “A review of recent numerical algorithms for hypersurfaces moving with curvature dependent speed,” *J. Differential Geometry*, vol. 31, pp. 131–161, 1989.

[60] G. Sapiro, “Geometric partial differential equations in image analysis: past, present, and future,” in *Proc. IEEE Int’l Conf. Imag. Proc.*, vol. 3, pp. 1–4, 1995.

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

[62] J. A. Sethian, *An Analysis of Flame Propagation*. Ph.D. thesis, Dept. of Mathematics, University of California, Berkeley, CA, 1982.

[63] A. Yezzi, S. Kichenassamy, A. Kumar, P. Olver, and A. Tennenbaum, “A geometric snake model for segmentation of medical imagery,” *IEEE Trans. Med. Imag.*, vol. 16, pp. 199–209, 1997.

[64] V. Caselles, R. Kimmel, and G. Sapiro, “Geodesic active contours,” *Int’l J. Comp. Vis.*, vol. 22, pp. 61–79, 1997.

[65] S. Kichenassamy, A. Kumar, P. Olver, A. Tennenbaum, and A. Yezzi, “Conformal curvature flows: from phase transitions to active vision,” *Arch. Rational Mech. Anal.*, vol. 134, pp. 275–301, 1996.

[66] K. Siddiqi, Y. B. Lauzière, A. Tannenbaum, and S. W. Zucker, “Area and length minimizing flows for shape segmentation,” *IEEE Trans. Imag. Proc.*, vol. 7, pp. 433–443, 1998.

[67] D. L. Chopp, “Computing minimal surfaces via level set curvature flow,” *J. of Comp. Phys.*, vol. 106, pp. 77–91, 1993.

[68] P. C. Teo, G. Sapiro, and B. A. Wandell, “Creating connected representations of cortical gray matter for functional MRI visualization,” *IEEE Trans. Med. Imag.*, vol. 16, pp. 852–863, 1997.

[69] C. Xu, D. L. Pham, and J. L. Prince, “Finding the brain cortex using fuzzy segmentation, isosurfaces, and deformable surface models,” in *Proc. Information Processing in Medical Imaging (IPMI’97)*, pp. 399–404, 1997.

[70] L. H. Staib and J. S. Duncan, “Boundary finding with parametrically deformable models,” *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 14, no. 11, pp. 1061–1075, 1992.

[71] K. Delibasis, P. E. Undrill, and G. G. Cameron, “Designing Fourier descriptor-based geometric models for object interpretation in medical images using genetic algorithms,” *Comp. Vis. Imag. Under.*, vol. 66, pp. 286–300, 1997.

[72] A. Pentland and B. Horowitz, “Recovery of nonrigid motion and structure,” *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 13, pp. 730–742, 1991.

[73] K. H. Huebner, E. A. Thornton, and T. G. Byrom, *The Finite Element Method for Engineers*. New York: John Wiley & Sons, 3rd ed., 1994.

[74] E. Bardinet, L. D. Cohen, and N. Ayache, “A parametric deformable model to fit unstructured 3D data,” *Comp. Vis. Imag. Under.*, vol. 71, pp. 39–54, 1998.

[75] D. Metaxas and D. Terzopoulos, “Shape and nonrigid motion estimation through physics-based synthesis,” *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 15, pp. 580–591, 1993.

[76] B. C. Vemuri and A. Radisavljevic, “Multiresolution stochastic hybrid shape models with fractal priors,” *ACM Trans. Graph.*, vol. 13, pp. 177–207, 1994.

[77] D. Metaxas, E. Koh, and N. J. Badler, “Multi-level shape representation using global deformations and locally adaptive finite elements,” *Int’l J. Comp. Vis.*, vol. 25, pp. 49–61, 1997.

[78] T. F. Cootes, A. Hill, C. J. Taylor, and J. Haslam, “Use of active shape models for locating structures in medical images,” *Imag. Vis. Computing J.*, vol. 12, no. 6, pp. 355–366, 1994.

[79] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham, “Active shape models – their training and application,” *Comp. Vis. Imag. Under.*, vol. 61, no. 1, pp. 38–59, 1995.

[80] J. C. Gower, “Generalized Procrustes analysis,” *Psychometrika*, vol. 40, pp. 33–51, 1975.

[81] N. Duta and M. Sonka, “Segmentation and interpretation of MR brain images: an improved active shape model,” *IEEE Trans. Med. Imag.*, vol. 17, pp. 1049–1062, 1998.

[82] Y. Wang and L. H. Staib, “Boundary finding with correspondence using statistical shape models,” in *Proc. IEEE Conf. Comp. Vis. Patt. Recog.*, pp. 338–345, 1998.

[83] G. H. Golub and C. F. V. Loan, *Matrix Computations*. Baltimore, MD: The Johns Hopkins University Press, 3rd ed., 1996.

[84] G. J. Edwards, C. J. Taylor, and T. F. Cootes, "Interpreting face images using active appearance models," in *Proc. Int'l Conf. Automatic Face Gesture Recog.*, pp. 300–305, 1998.

[85] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active appearance models," in *Proc. European Conf. Comp. Vis.*, pp. 484–498, 1998.

[86] T. F. Cootes, C. Beeston, G. J. Edwards, and C. J. Taylor, "A unified framework for atlas matching using active appearance models," in *Proc. Information Processing in Medical Imaging (IPMI'99)*, pp. 323–333, 1999.

[87] G. J. Edwards, C. J. Taylor, and T. F. Cootes, "Learning to identify and track faces in image sequences," in *Proc. British Mach. Vis. Conf.*, pp. 317–322, 1997.

[88] A. Hill and C. J. Taylor, "Automatic landmark identification using a new method of non-rigid correspondence," in *Proc. Information Processing in Medical Imaging (IPMI'97)*, pp. 483–488, Springer-Verlag, 1997.

[89] A. C. W. Kotcheff and C. J. Taylor, "Automatic construction of eigenshape models by genetic algorithm," in *Proc. Information Processing in Medical Imaging (IPMI'97)*, pp. 1–14, Springer-Verlag, 1997.

[90] N. Duta, M. Sonka, and A. K. Jain, "Learning shape models from examples using automatic shape clustering and Procrustes analysis," in *Proc. Information Processing in Medical Imaging (IPMI'99)*, pp. 370–375, 1999.

[91] H. H. S. Ip and D. Shen, "An affine-invariant active contour model (AI-snake) for model-based segmentation," *Imag. Vis. Computing J.*, vol. 16, pp. 135–146, 1998.

[92] Y. Guo and B. C. Vemuri, "Hybrid geometric active models for shape recovery in medical images," in *Proc. Information Processing in Medical Imaging (IPMI'99)*, pp. 112–125, 1999.

[93] D. Fritsch, S. Pizer, L. Yu, V. Johnson, and E. Chaney, "Segmentation of medical image objects using deformable shape loci," in *Proc. Information Processing in Medical Imaging (IPMI'97)*, pp. 127–140, 1997.

[94] T. McInerney, *Topologically Adaptable Deformable Models for Medical Image Analysis*. Ph.D. thesis, Department of Computer Science, University of Toronto, 1997.

[95] A. Blake and A. Yuille, eds., *Active Vision*. Series: Artificial intelligence, Cambridge, Massachusetts: The MIT Press, 1992.

[96] D. N. Metaxas, *Physics-Based Deformable Models*. Boston: Kluwer Academic Publishers, 1996.

[97] A. Blake and M. Isard, *Active Contours: The Application of Techniques from Graphics, Vision, Control Theory and Statistics to Visual Tracking of Shapes in Motion*. New York: Springer-Verlag, 1998.

[98] A. Singh, D. Goldgof, and D. Terzopoulos, *Deformable models in medical image analysis*. Los Alamitos, CA: IEEE Computer Society, 1998.

[99] T. McInerney and D. Terzopoulos, "Deformable models in medical image analysis: a survey," *Med. Imag. Anal.*, vol. 1, no. 2, pp. 91–108, 1996.