Medical imaging continues to permeate the practice of medicine, but automated yet accurate segmentation and labeling of anatomical structures continues to be a major obstacle to computerized medical image analysis. Deformable models, with their roots in estimation theory, optimization, and physics-based dynamical systems, represent a powerful approach to the general problem of medical image segmentation. This chapter presents an introduction to deformable models, beginning with the classical Active Contour Models (ACMs), or snakes, and focusing on explicit, physics-based methods. Snakes are useful for segmenting amorphous shapes when little or no prior knowledge about shape and motion is available. Many extensions of snakes incorporate such additional knowledge. An example presented in this chapter is the use of optical flow forces to incorporate knowledge of shape dynamics and guide the snake deformations to track the leading edge of an injected contrast agent in an echocardiographic image sequence. Active Shape Models (ASMs), or smart snakes, is a powerful method for incorporating statistical models of shape variability in the segmentation process. ASMs and ACMs offer different advantages, and, as such, a method combining both is presented. Statistical knowledge about shape dynamics is useful for segmenting and tracking objects with distinctive motion patterns (such as a beating heart). An extension of the ASM to model knowledge of spatiotemporal constraints is presented.

1. ENERGY-MINIMIZING DEFORMABLE MODELS

1.1. Introduction

The classical deformable model, the Active Contour Model or snakes [1, 2], gained wide acceptance as a segmentation tool due to its robustness to image
noise and boundary gaps. Ideas related to snakes date back to the early 1970s [3, 4]. In short, an ACM is an energy-minimizing contour with smoothness constraints, deformed according to image data. They integrate boundary elements into a single, inherently connected, smooth, mathematically well-defined structure, which can be implemented on the continuum achieving sub-pixel accuracy. ACMs were originally designed to be semiautomatic tools supporting intuitive interaction mechanisms for guiding the segmentation process. In active contour models, a contour is initialized on the image and left to deform in a way that, first, moves it toward features of interest in the image and, second, maintains a certain degree of smoothness in the contour. Consequently, an energy term is associated with the contour and is designed to be inversely proportional to both the contour’s smoothness and its fit to the image data, in order to segment the desired image features. Deformation of the contour in the image will change its energy; thus, one can imagine an energy surface on top of which the contour moves (in a way that resembles the slithering of a snake, and hence the name) while seeking valleys of low energy [5].

1.2. Classical Snakes

A 2D snake in a continuous spatial domain is represented as a 2D parametric contour $\mathbf{v}(s) = (x(s), y(s))$, where $s \in [0, 1]$. In order to fit the snake model to the image data we associate energy terms with the snake and aim to deform the snake in a way that minimizes its total energy. The energy of the snake, $\xi$, depends on both the shape of the contour and the image data $I(x, y)$ reflected via the internal and external energy terms, $\alpha(\mathbf{v}(s))$ and $\beta(\mathbf{v}(s))$, respectively. The total snake energy is written as

$$\xi(\mathbf{v}(s)) = \alpha(\mathbf{v}(s)) + \beta(\mathbf{v}(s)).$$

The internal energy term is given by

$$\alpha(\mathbf{v}(s)) = \int_0^1 w_1(s) \left| \frac{\partial \mathbf{v}(s)}{\partial s} \right|^2 + w_2(s) \left| \frac{\partial^2 \mathbf{v}(s)}{\partial s^2} \right|^2 ds,$$

whereas the external energy term is given as

$$\beta(\mathbf{v}(s)) = \int_0^1 w_3(s) P(\mathbf{v}(s)) ds.$$

Weighting functions $w_1$ and $w_2$ control the tension and flexibility of the contour, respectively, and $w_3$ controls the influence of the image data. $w_i$ can depend on $s$ but are typically set to different constants. For the contour to be attracted to
image features, function $P(v(s))$ is designed such that it has minima where the features have maxima. For example, for the contour to be attracted to high-intensity changes (high gradients), we can choose

$$P(v(s)) = P(x(s), y(s)) = -\|\nabla [G_\sigma * I(x, y)]\|,$$  \hspace{1cm} (4)

where $G_\sigma * I$ denotes the image convolved with a smoothing (e.g., Gaussian) filter with a parameter $\sigma$ controlling the extent of the smoothing (e.g., variance of Gaussian).

The contour $v$ that minimizes energy $\xi$ must, according to the calculus of variations [6], satisfy the vector-valued partial differential (Euler-Lagrange) equation:

$$-\frac{\partial}{\partial s} \left( w_1 \frac{\partial v}{\partial s} \right) + \frac{\partial^2}{\partial s^2} \left( w_2 \frac{\partial^2 v}{\partial s^2} \right) + w_3 \nabla P(v(s)) = 0.$$  \hspace{1cm} (5)

1.3. Dynamic Deformable Models

In order to attack the problem of tracking non-rigid time-varying objects, deformable models were extended to dynamic deformable models. These describe the shape changes in a single model that evolves through time to reach a state of equilibrium where internal forces representing constraints on shape smoothness balance the external image forces and the contour comes to rest [7]. In this case the time-varying (dynamic) contour is written as $v(s, t) = (x(s, t), y(s, t))$, where $s \in [0, 1]$ and the corresponding constraint equation becomes

$$\mu(s) \frac{\partial^2 v}{\partial t^2} + \gamma(s) \frac{\partial v}{\partial t} - \frac{\partial}{\partial s} \left( w_1 \frac{\partial v}{\partial s} \right) + \frac{\partial^2}{\partial s^2} \left( w_2 \frac{\partial^2 v}{\partial s^2} \right) + w_3 \nabla P(v(s, t)) = 0,$$  \hspace{1cm} (6)

where $\mu(s)$ and $\gamma(s)$ are the mass and damping densities, respectively.

1.4. Snakes Drawbacks

Snakes do not, however, solve the entire problem of finding contours in images. This is mainly because they lack high-level control and depend on other mechanisms, such as interaction with the user or some higher-level image understanding process, and information from image data adjacent in time or space [8]. Snakes are sensitive to initialization and parameter selection (e.g., $w_i$ in (2) and (3), or $\sigma$ in (4)) and therefore rely on human guidance to specify approximate shape and starting position for the snake somewhere near the desired contour as well as suitable values of the snake’s parameters (see Figures 1 and 2). Without such guidance snakes can leak or latch onto erroneous edges. In addition to user interaction, other approaches have been proven useful to deal with the snakes’ limited capture range problem and difficulties in progressing into boundary concavities. These include the incorporation of inflation forces [9], multi-resolution
search [10], an initial step to optimize for rigid parameters, distance transform-based potential functions, and Gradient Vector Flow Fields (see Section 1.6.5). In the next chapter, deformable organisms are introduced as an attempt to provide a simple model of the human expert’s cognitive abilities in order to guide the model’s deformations.

Figure 1. Sample frames (progressing left to right, top to bottom) showing incorrect progress of a deformable model (snake) for segmenting the corpus callosum (CC) in a midsagittal brain magnetic resonance image (MRI), due to the wrong choice of parameters. See attached CD for color version.

Figure 2. Sample frames (progressing left to right, top to bottom) showing incorrect progress of a snake segmenting the CC in an MRI image. Leaking of the snake occurs because of the weak edge strength (lower left) and incorrect parameter setting. See attached CD for color version.
1.5. Discretization and Numerical Simulation

For a polygonal snake contour the discrete version of (6) can be written as

$$\mu_i \ddot{v}_i (t) + \gamma_i \dot{v}_i (t) - w_1 v_i'' (t) + w_2 v_i''' (t) + w_3 \nabla P (v_i (t)) = 0,$$

(7)

where \(\{v_i (t) = (x_i (t), y_i (t))\}_{i=1,2,\ldots,N}\) are the nodes of the snake polygon, \(t\) is used as the discrete time variable, and \(i\) is the snake node index. \(\dot{v}\) and \(\ddot{v}\) are the first and second derivates of \(v\) with respect to \(t\). \(v''\) and \(v'''\) are the second and fourth derivates of \(v\) with respect to \(i\). Setting the mass density to zero\(^1\) \((\mu_i = \mu = 0)\) and the damping density to a constant \((\gamma_i = \gamma)\), we rewrite Eq. (7) for simulating the deformations of the discrete snake as

$$\gamma \dot{v}_i - w_1 F_i^{\text{tensile}} (t) + w_2 F_i^{\text{flexural}} (t) = w_3 F_i^{\text{external}} (t).$$

(8)

\(F_i^{\text{tensile}} (t)\) is a tensile force (resisting stretching) acting on node \(i\) at time \(t\) and is given by

$$F_i^{\text{tensile}} (t) = 2v_i (t) - v_{i-1} (t) - v_{i+1} (t).$$

(9)

\(F_i^{\text{flexural}} (t)\) is a flexural force (resisting bending) and is given by

$$F_i^{\text{flexural}} (t) = 2F_i^{\text{tensile}} (t) - F_{i-1}^{\text{tensile}} (t) - F_{i+1}^{\text{tensile}} (t).$$

(10)

\(F_i^{\text{external}} (t)\) is an external (image-derived) force. It is derived in a way that causes the snake node to move toward regions of higher intensity gradient in the image and is given by

$$F_i^{\text{external}} (t) = \nabla P (x_i (t), y_i (t)),\$$

(11)

where \(P (x_i (t), y_i (t))\) is given in (4).

The equation used for updating the position of any snake node \(i\) can be obtained from (8) by using a finite-difference derivative approximation \(\dot{v}_i = (v_i (t+\Delta t) - v_i (t))/\Delta t\), where \(\Delta t\) is a finite time step, yielding

$$v_i (t+\Delta t) = v_i (t) - \frac{\Delta t}{\gamma} (w_1 F_i^{\text{tensile}} (t) - w_2 F_i^{\text{flexural}} (t) + w_3 F_i^{\text{external}} (t)).$$

(12)

1.6. Extensions

1.6.1. Inflation Force

In addition to the above forces, an inflation force, \(F_i^{\text{inflation}} (t)\), can be utilized to allow for initializing the snake farther away from the target boundary.

\(^1\) In static shape recovery problems not involving time-varying data, the mass density is often set to zero, resulting in simplified equations of motion and a snake that comes to rest as soon as the internal forces balance the external forces [11].
\( F_{\text{inflation}}^i (t) \) is given by

\[
F_{\text{inflation}}^i (t) = F (I_s (x_i, y_i)) \mathbf{n}_i (t),
\]

where \( I_s \) is a smoothed version of \( I \), \( \mathbf{n}_i (t) \) is the unit vector in the direction normal to the contour at node \( i \) and the binary function

\[
F (I (x, y)) = \begin{cases} 
+1 & \text{if } I (x, y) \geq T \\
-1 & \text{otherwise}
\end{cases}
\]

links the inflation force to the image data, where \( T \) is an image intensity threshold. More elaborate rules can be used (e.g., using region-based image intensity statistics [12]). Consequently, Eq. (8) becomes

\[
\gamma \dot{v}_i - w_1 F_{\text{tensile}}^i (t) + w_2 F_{\text{flexural}}^i (t) = w_3 F_{\text{external}}^i (t) + q F_{\text{inflation}}^i (t),
\]

where \( q \) is a scalar weighting the inflation force. Subsequently, (12) becomes

\[
v_i (t + \Delta t) = v_i (t) - \frac{\Delta t}{\gamma} (w_1 F_{\text{tensile}}^i (t) - w_2 F_{\text{flexural}}^i (t))
+ w_3 F_{\text{external}}^i (t) + q F_{\text{inflation}}^i (t)).
\]

1.6.2. Adaptive Inflation

In order to dampen the inflation force when the snake nodes reach the target boundary, a node-specific inflation weight \( q_i \) can be used, where \( i \) is the snake node number. Having only a single value for the inflation force for all the nodes is insufficient, since this causes the nodes that reach the target boundary earlier than others to pass over it and cause the snake to leak. If a node reaches the target boundary the inflation direction is reversed (inflation becomes deflation and vice versa), and if a certain number of inflation reversals occur within a limited number of past iterations, then the inflation force is dampened (\( q_i \) is replaced by \( \alpha q_i, \alpha < 1 \)) for that particular node only. This updated Eq. (16) becomes

\[
v_i (t + \Delta t) = v_i (t) - \frac{\Delta t}{\gamma} (w_1 F_{\text{tensile}}^i (t) - w_2 F_{\text{flexural}}^i (t))
+ w_3 F_{\text{external}}^i (t) + q_i F_{\text{inflation}}^i (t)).
\]

1.6.3. Adaptive Subdivision Scheme

Different subdivision schemes can be applied, where the number of snake nodes is adapted based on the distance between nodes and the curvature along the snake. A polygonal snake can start with only a few nodes and then increase the number of nodes as it deforms to accommodate for a longer, concavity seeking snake. Figure 3 presents one possible implementation, while Figure 4 shows a snake segmentation example that utilizes adaptive inflation and subdivision.
Figure 3. Adaptive resampling of a polygonal snake. (Left) Adding a node if the distance between consecutive nodes is large or curvature is high. (Right) Removing a node if the distance between nodes is small or the curvature is small (appropriate distance and curvature thresholds must be chosen). See attached CD for color version.

Figure 4. Sample frames (progressing left to right, top to bottom) showing progress of correct snake segmentation of the corpus callosum. The segmentation utilizes adaptive inflation and subdivision and an appropriate choice of weights. See attached CD for color version.

1.6.4. User Interaction

User interaction is typically employed to assist the snake in latching onto the desired structure, though its implementation often varies. A user can click a point on the image through which they believe the snake must pass, and the snake node closest to the mouse-click position is found and its position reset to the location of the user-supplied point. A force vector field can also be designed in the vicinity of
the user point that attracts the snake to that location. Alternatively, a force resulting from a virtual spring connecting the mouse-click position with the closest snake contour node can be applied. See examples of user-assisted snake segmentation using the first method in Figures 5 and 6.

### 1.6.5. Gradient Vector Flow Snakes

Traditional snakes require initialization near their targets and consequently are unable to push into concavities. Gradient vector flow (GVF) snakes use a new external force that enables initialization in areas with little gradient information, thus enabling far coarser snake initialization and segmentation of concave objects [13]. The GVF field, denoted by $F_{\text{gvf}}(x, y) = (u(x, y), v(x, y))$, is obtained by minimizing the energy functional $E_{\text{gvf}}$ with respect to $F_{\text{gvf}}$:

$$E_{\text{gvf}} = \int \int (\mu(u_x^2 + u_y^2 + v_x^2 + v_y^2) + |\nabla f|^2 |F_{\text{gvf}} - \nabla f|^2) \, dx \, dy. \quad (18)$$

This minimization diffuses the image gradient such that it smoothly declines in lower-valued regions and directly approximates the gradient in high-magnitude areas, reflected by the first and second terms of (18), respectively (Figure 7). The relative weighting of these two terms is governed by the scalar $\mu$. $f(x, y) = -E_{\text{ext}}(x, y)$, where $E_{\text{ext}}(x, y)$ is an image edge map (for example, obtained using a Sobel or Canny filter). $u_x, u_y, v_x,$ and $v_y$ are the derivatives of the components of $F_{\text{gvf}}, u(x, y),$ and $v(x, y),$ with respect to $x$ and $y$. $F_{\text{gvf}}$ is then used in place of the external forces, $F_{\text{external}}$, in Eq. (12), to obtain

$$v_i(t + \Delta t) = v_i(t) - \frac{\Delta t}{\gamma}(w_1 F_{i}^{\text{tensile}}(t) - w_2 F_{i}^{\text{flexural}}(t) + w_3 F_{i}^{\text{gvf}}(v_i(t))). \quad (19)$$

### 1.6.6. Color Images

2D color images map a two-dimensional ($n = 2$) spatial space $(x, y)$ into a three-dimensional ($m = 3$) color space with coordinates $(u, v, w)$ (e.g., R, G, B). In order to apply snakes to segmenting structures in color images, without first converting the color image to a single band, external forces must be redefined to make use of the color gradient. Let the channel derivative matrix $D$ be defined as

$$D(x, y) = \begin{bmatrix} \partial u/\partial x & \partial u/\partial y \\ \partial v/\partial x & \partial v/\partial y \\ \partial w/\partial x & \partial w/\partial y \end{bmatrix}. \quad (20)$$

Then, the gradient magnitude and the gradient direction are taken as the square root of the largest eigenvalue, $\sqrt{\lambda_{\text{max}}}$, of the matrix $D^T D$ and its corresponding
Figure 5. Sample frames showing progress of user-assisted snake segmentation of brain cerebellum. The small green circles are locations of mouse-clicks by the user. See attached CD for color version.

Figure 6. Segmentation of an oral lesion example using the green band of a digital color image. (a) Initial snake nodes. (b) Final segmentation result (snake nodes shown as white dots and forced points as white circles). Reprinted with permission from [61]. Copyright ©2000, IEEE.

eigenvectors, respectively. In order to apply snakes to segment structures in color images, the external image force becomes $F_{\text{external}}^i(t) = \nabla P(x_i(t), y_i(t)),$
Figure 7. Gradient Vector Flow Field: (a) original image depicting a corpus callosum in a midsagittal brain MRI, (b) gradient vector field, (c) GVF field. Note how in (c) the field is less sparse than in (b) and smoothly extends outward into homogenous regions.

where now (compare with (4))

\[ P(x_i(t), y_i(t)) = -c\sqrt{\lambda_{\text{max}}(x_i(t), y_i(t))}, \]

and \( \lambda_{\text{max}}(x_i(t), y_i(t)) \) is calculated at the location of snake node \( x_i(t), y_i(t) \) at iteration time \( t \). For more details the reader is referred to [14].

1.6.7. Deformable Spring–Mass Mesh Models

Spring mass meshes can also be used to simulate deformable models that are fitted to object shapes in images. A mesh is made up of nodes (masses or particles) and springs (elastic links or connecting segments). A mass \( m_i \), position \( x_i \), velocity \( v_i \), and acceleration \( a_i \) are associated with each node \( n_i \). Two terminal nodes \( n_i \) and \( n_j \), Hooke’s spring constant \( k_s \), damping constant \( k_d \), and rest length \( r_{ij} \) are associated with each spring \( s_{ij} \). By applying Newton’s second law of motion and simulating the dynamics by time integration, the mesh nodes move, deforming the object’s shape. Newton’s second law of motion for node \( n_i \) states that

\[ a_i = \frac{F_i}{m_i}, \]

where \( F_i \) is the total force acting on \( n_i \):

\[ F_i = F_i^{\text{Hooke}} + F_i^{\text{viscous}} + F_i^{\text{user}} + F_i^{\text{external}}. \]

A spring \( s_{ij} \) will cause

\[ F_i^{\text{Hooke}} = -k_s \left( \|x_i - x_j\| - r_{ij} \right) \frac{x_i - x_j}{\|x_i - x_j\|} - \left( k_d (v_i - v_j)^T \frac{x_i - x_j}{\|x_i - x_j\|} \right) \]

\[ = \frac{u_{ij}}{\|u_{ij}\|} \]

where \( u_{ij} = x_i - x_j \)

(23)

to be exerted at \( n_i \) and \( -F_i^{\text{Hooke}} \) on \( n_j \). Viscous drag at \( n_i \) is given by

\[ F_i^{\text{viscous}} = -k_v v_i. \]

(24)
A single user-applied force $F_{\text{user}}^i$ is implemented as the dynamic force resulting from a spring connecting a mesh node to the (varying) position of the user’s point of application. Image forces can be implemented as

$$F_{\text{external}}^i(t) \propto \nabla ||\nabla [G_\sigma \ast I(x_i)]||,$$  \hspace{1cm} (25)

where $I(x_i)$ is the intensity of a pixel at the location of node $n_i$ (see Eq. (4)). Image forces that attract the model to an image boundary are calculated only for boundary mesh nodes (similarly, image forces that attract medial model nodes to medial features can also be applied). Following the calculation of the node forces we compute the new acceleration, velocity, and position of each node given the old velocity and position values, as follows (explicit Euler solution with time step $\Delta t$):

$$a_i = \frac{F_i}{m_i},$$  
$$v_i = v_{\text{old}}^i + a_i \Delta t,$$  
$$x_i = x_{\text{old}}^i + v_i \Delta t.$$  \hspace{1cm} (26)

### 1.6.8. General Shape Parameters

A vector of shape parameters $u$ can also be used to represent the continuous geometric contour model $v(s)$. These shape parameters are generally associated with some local-support basis functions (such as splines and finite elements) or global-support basis functions (such as Fourier bases) [7]. In this case the discrete form of $\xi(v)$ may be written as

$$E(u) = \frac{1}{2} u^T K u + P(u),$$  \hspace{1cm} (27)

where $K$ is called the stiffness matrix and $P(u)$ is the discrete version of external potential $P(v(s))$. The contour parameters that minimize the energy function can now be obtained by solving the set of algebraic equations

$$Ku = -\nabla P.$$  \hspace{1cm} (28)

The motion equation for the contour (represented by $u$) can be written as

$$M\ddot{u} + N\dot{u} + Ku = -\nabla P,$$  \hspace{1cm} (29)

where $M$ and $N$ are mass and damping matrices, respectively [7].

### 1.6.9. A Probabilistic Formulation

An alternative view of deformable models is a probabilistic formulation. The deformable model can be fitted to the image data by finding the model shape
parameters $\mathbf{u}$ that maximize the posterior probability (maximum a posteriori or MAP solution) expressed using Bayes’ theorem as

$$p(\mathbf{u}|I) = \frac{p(I|\mathbf{u})p(\mathbf{u})}{p(I)},$$

(30)

where $p(\mathbf{u})$ is the prior probability density of the model shape parameters: a mechanism for probabilistic regularization. $p(I|\mathbf{u})$ is the probability of producing an image $I$ given the parameters $\mathbf{u}$: an imaging (sensor) model. $p(\mathbf{u})$ and $p(I|\mathbf{u})$ can be written (in the form of Gibbs distribution) as

$$p(\mathbf{u}) = k_1 \exp(-A(\mathbf{u})),$$

(31)

$$p(I|\mathbf{u}) = k_2 \exp(-P(\mathbf{u})),$$

(32)

where $k_1$ and $k_2$ are normalizing constants and $A(\mathbf{u})$ is the discrete version of the internal energy $\mathbf{\alpha}(\mathbf{v})$ and $P(\mathbf{u})$ is the discrete version of external potential $P(\mathbf{v}(s))$.

### 1.6.10. Deformable Surface Models

For segmenting 3D objects in volumetric medical images, one could resort to slice-by-slice segmentation, where in each slice a 2D deformable model is initialized using the results from a neighboring slice. However, this approach does not inherently integrate 3D information. The following steps are generally performed in 3D deformable models. A geometrical 3D shape representation of the deformable model is decided upon; the model is initialized (including initial parameter setting) in the image space, and the model is left to deform to minimize internal and external energy terms by using internal and external forces, for example. Some example implementations of 3D deformable models include Simplex Meshes [15] and finite-element methods [16]. Another implementation is the 3D spring–mass mesh model, which is similar to the 2D spring mass mesh system but now masses are positioned in 3D space so the position, velocity, and acceleration $(x, v, a)$ become 3-vectors and new positions are calculated as in Eq. (26), where only the dimensionality of the position, force, velocity, and acceleration vectors has changed.

### 1.7. Optical Flow Snake Forces

In this section we present a detailed look into an extension of the formulation of Active Contour Models (snakes) that includes an additional contour-deforming force [17, 18]. The new force is derived from the optical flow field calculated between two time-consecutive frames in an image sequence. Addition of the new force assists the snake in tracking desired dynamics while the traditional snake forces guarantee the contour’s smoothness and attraction to edges. The method is
applied to the problem of tracking the leading edge of an injected contrast agent in an echocardiographic image sequence and is shown to improve the tracking performance. A clinical motivation and previous work on echocardiography and video densitometry are initially presented.

1.7.1. Clinical Motivation

The assessment of human right-ventricular (RV) function is of great importance in the study of many diseases afflicting the heart. For example, disturbances in filling and elimination patterns of RV hemodynamics can be interpreted as signs of abnormal RV function. Arrhythmogenic Right-Ventricular Dysplasia (ARVD) is a rare but clinically important disease that afflicts young adults and may play a role in the etiology of sudden death among the young [19]. Impairments in RV function in this group of patients can be described in terms of wall motion abnormalities, or as localized bulging and sacculations. These abnormalities are mainly located at the inflow, outflow, or apical regions. To study these abnormalities we use sequences of contrast echocardiographic images. Automated medical image analysis techniques are needed to extract clinically relevant information from this data.

1.7.2. Medical Imaging Procedure

Thirty patients with biopsy-verified ARVD and 18 healthy volunteers (control group) were investigated by use of contrast echocardiography. The investigations were performed with an Acuson XP128 computer system or a Sequoia system equipped with multi-Hertz transducers. As a contrast agent, 2 ml of Haemaccel<sup>®</sup> (Hoechst) was injected intravenously. Transthoracic apical four-chamber view with focus on the right ventricle was used and continuously recorded during and after injection. The video sequence (Figure 8) of the filling and elimination of the contrast agent was then digitized using a PC with a frame grabber (Matrox, Meteor II), giving about 600 images for each sequence.

1.7.3. Tracking the Contrast Front

In order to characterize the RV flow patterns, we track the front (the leading edge) of the contrast agent during RV filling. Examining the flow of the contrast agent in a typical echocardiographic image sequence (Figure 9) reveals weak ultrasound echoes, echo dropouts, and high levels of noise. Application of simple edge detectors to locate the front of the contrast would therefore result in detecting erroneous edges and gaps.

The strength of active contour models in integrating low-level image information and guaranteeing a smoothly connected contour makes them suitable for this purpose. For tracking the contrast front in a sequence we make use of the motion field obtained by Optical Flow (OF) computation as an additional contour
Figure 8. Sample frames from a digitized image sequence. In frame #1, #59, #124, and #422 the contrast agent has not, reached, just reached, totally filled, and washed out from the RV, respectively. Reprinted with permission from [18]. Copyright ©2000, IEEE.

deoforming force, in order to speed up tracking and influence the snake nodes to match corresponding contrast front regions between frames.

Other authors have investigated similar approaches. In [20] a method for segmenting and tracking cardiac structures in ultrasound image sequences was presented. In integrating the contour’s equation of motion, the method sets the initial velocities of the contour vertices to OF estimates, and sets their positions relative to the final position from the preceding frame. Peterfreund [21] used Kalman filter-based active contours that calculate OF along the contour as system

Figure 9. Successive frames of contrast agent entering the right ventricle of the heart. Reprinted with permission from [17]. Copyright ©2000, IEEE.
measurement to detect and reject measurement that may belong to other objects. Akgul et al. presented an application of tracking 2D contours of tongue surfaces from digital ultrasound image sequences [22]. The method makes use of OF to reduce the computational complexity involved when searching for optimal snake node locations in a dynamic programming setting. This is done by considering only a subset of pixels in a search window. The subset is chosen on the basis of the first OF constraint, namely that the intensity of an object’s point in a dynamic image does not change with time.

1.7.4. Optical Flow

Optical flow [23] is a well-established method for calculating the velocity field \((u(x, y), v(x, y))\) of the apparent 2D motion of pixels in a dynamic image, \(I(x, y, t)\), due to the 3D motion of imaged objects. The OF velocity field is obtained by examining the spatial and temporal changes in intensity values. Classical OF is based on two main constraints. The first states that the brightness of any object point is constant over time. This can be written as

\[
I(x + dx, y + dy, t + dt) = I(x, y, t).
\]

Using Taylor series expansion and neglecting higher-order terms gives the first OF constraint equation:

\[
I_x u + I_y v + I_t = 0,
\]

where \(u = dx/dt\), and \(v = dx/dt\) are the desired velocity field components, \(I_x\) and \(I_y\) are the spatial image derivatives, and \(I_t\) is the temporal image derivative. Equation (33) by itself is insufficient to calculate \((u, v)\); hence a second constraint, the velocity field smoothness constraint, is introduced. The velocity field can now be calculated as that which best satisfies both constraints by minimizing the following square error function:

\[
\xi^2(x, y) = (I_x u + I_y v + I_t)^2 + \lambda (u_x^2 + u_y^2 + v_x^2 + v_y^2),
\]

where \(\lambda\) is a Lagrange multiplier. The following iterative algorithm detailed in [23] is used to find the velocity field (Figure 10):

\[
\begin{align*}
\text{Initialize: } & u(x, y) = v(x, y) = 0 \text{ for all } x, y, \\
\text{do: } & u = \bar{u} - I_x \frac{P}{D}, \quad v = \bar{v} - I_y \frac{P}{D} \text{ while } \xi^2(x, y) > \varepsilon,
\end{align*}
\]

where \(P = I_x \bar{u} + I_y \bar{v} + I_t, D = \lambda^2 + I_x^2 + I_y^2\), and \(\varepsilon\) is a small number.

1.7.5. Optical Flow Snake Forces

In order to track the contrast agent front in an echocardiographic image sequence, we need to accomplish two tasks. The first is to locate the region where
the contrast front has moved from one frame to the next, and the second is to detect this front as a smooth and connected boundary. We use the optical flow to address the first task and snakes to address the second. To combine the two techniques we include an additional force term $F_{\text{flow}}^i(t)$ proportional to the calculated velocity field at the current snake node position $v_i(t) = (x_i(t), y_i(t))$, yielding
\begin{equation}
\dot{v}_i(t) = v_i(t-1) + w_1 F_{\text{tensile}}^i(t) + w_2 F_{\text{flexural}}^i(t) + w_3 F_{\text{external}}^i(t) + w_4 F_{\text{flow}}^i(t),
\end{equation}
where $w_i$ are weighting factors,
\begin{equation}
F_{\text{flow}}^i(t) \propto (u(x_i(t-1), y_i(t-1)), v(x_i(t-1), y_i(t-1))),
\end{equation}
and $u(x_i, y_i), v(x_i, y_i)$ are obtained using algorithm (36) in Section 1.7.4.

1.8. Results

We tracked the leading edge of a contrast agent filling the RV in real ultrasonic image sequences. Images were first smoothed using nonlinear diffusion filtering [24]. The front of the contrast agent was tracked in eight sequences during the RV filling process, five from the ARVD group and three from the control group. In each sequence, the front was tracked, on average, in about eight images. In the example depicted in Figure 11, the snake without OF forces needed 19, 20, 23, 22, 22, and 16 iterations (Figure 11a), whereas the snake with OF forces needed 5, 8, 10, 10, 6 and 10 iterations (Figure 11b). We also show an example of both snakes, with and without OF forces, deforming toward the leading edge of the contrast agent in a single frame (Figure 12). The OF snake (with larger nodes in the figure) progresses faster toward the edge and locates it in only 10 iterations compared to 23 iterations needed for the snake without OF forces. Histograms of the number of iterations needed for the contour to find the edge for all tested frames are also calculated (Figure 13). The mean number of iterations needed with
and without the use of information about the OF was 6.3 and 12.3, respectively. Figure 14 shows the snake contour tracking the leading edge of the contrast front and providing clinically relevant RV hemodynamics measurements.

![Figure 11. Results of tracking a real sequence. Upper frames: without using optical flow forces obtained after 19, 20, 23, 22, 22, and 16 iterations. Lower frames: with optical flow forces obtained after 5, 8, 10, 10, 6, and 10 iterations. Reprinted with permission from [17]. Copyright ©2000, IEEE.](image)

**Figure 11.** Results of tracking a real sequence. Upper frames: without using optical flow forces obtained after 19, 20, 23, 22, 22, and 16 iterations. Lower frames: with optical flow forces obtained after 5, 8, 10, 10, 6, and 10 iterations. Reprinted with permission from [17]. Copyright ©2000, IEEE.

### 2. SMART SNAKES: INCORPORATING KNOWLEDGE ABOUT SHAPE

#### 2.1. Introduction

Active shape models (ASMs) or smart snakes are deformable shape modeling techniques for segmentation of objects in images. ASMs ensure that the deformed shape is consistent with a statistical model of shape variability calculated before-

![Figure 12. The snake with optical flow forces (large nodes) progresses faster toward the contrast front compared to the snake without optical flow forces. The snake nodes are shown after 1 (left-most), 2, 6, 10, 15, and 23 (right-most) iterations. Reprinted with permission from [17]. Copyright ©2000, IEEE.](image)

**Figure 12.** The snake with optical flow forces (large nodes) progresses faster toward the contrast front compared to the snake without optical flow forces. The snake nodes are shown after 1 (left-most), 2, 6, 10, 15, and 23 (right-most) iterations. Reprinted with permission from [17]. Copyright ©2000, IEEE.
hand in a training stage. In an ASM, an initial shape model estimate is deformed to fit the image data. The deformations are chosen such that they minimize a certain cost function and are consistent with the a priori knowledge about the shape of the target object to be segmented. The cost function is inversely proportional to the fit of the model to certain image features. Although incorporating a priori shape knowledge into an ASM generally improves the segmentation results, the technique requires training, becomes less general, and is not suitable on its own for detecting shapes that are dissimilar to those in the training set, which is problematic in cases where unpredictable changes in shape due to pathology may be observed in new cases. In the following sections we describe the training and application stages of an ASM. Further details are provided in Section 4 that describe the extension of 2D ASM to include temporal variations.

2.1.1. Modeling the a priori Knowledge of Shape Variation

A representative training set of images depicting the shape variations we wish to model is collected. A training set of shapes described by corresponding landmarks (distinctive coordinates) is extracted from the training set of images. Given $N$ training example shapes, each represented by $L$ landmarks, each landmark being an $(x, y)$ coordinate, the next step is to obtain a set of aligned shapes, viewed as a collection of $N$ points in a $2L$-dimensional space. The alignment is done by changing the pose (scaling, rotation, and translation) of the shapes such that the sum of distances between corresponding landmarks is minimized. Principal Component Analysis (PCA) is then applied in order to capture the main modes of shape variations of the aligned training set of shapes. Using the result from
Figure 14. The result of tracking the leading edge in one of the sequences from the ARVD group. Each contour represents the contrast agent front at different times, indicated beside each contour. The contrast front enters the RV ($t = 0$) until the RV is totally filled ($t = 2960$ ms). Note how the contrast front in the initial phase of filling ($t = 0$ to 480 ms) moves faster than the final phase ($t = 480$ to 2960 ms). This is indicative of the inhomogeneous operation of the RV, identified via contrast front tracking. Reprinted with permission from [18]. Copyright ©2000, IEEE.
PCA, we can approximate each shape by the sum of a mean shape and a linear combination of, say \( t \), principal components, as follows:

\[
\mathbf{x} = \bar{\mathbf{x}} + \mathbf{Pb}, \tag{39}
\]

where

- \( \mathbf{x} \) is the vector of landmark coordinates,
- \( \bar{\mathbf{x}} \) is the mean shape,
- \( \mathbf{P} \) is the matrix of principal components,
- \( \mathbf{b} \) is a vector of weighting parameters, also called shape parameters, and
- \( \mathbf{x} \) and \( \bar{\mathbf{x}} \) are each of length \( 2L \). \( \mathbf{P} \) is \( 2L \times t \) and \( \mathbf{b} \) is a vector of length \( t \).

Equation (39) can be used to generate new shapes by choosing different values for \( \mathbf{b} \). Constraints on these weighting parameters are used to ensure that only allowable shapes are produced. The allowable shapes are those shapes belonging to a region called the Allowable Shape Domain (ASD), which is inferred from the region that the aligned training set occupies. Aside from generating new similar shapes, this model is used to examine the plausibility of other shapes by checking whether or not they lie in the ASD. The above model of the distribution of points (or variation of shapes) is referred to as a Point Distribution Model (PDM).

### 2.1.2. Modeling the Gray-Level Appearance

During image search or the segmentation procedure, there is a need to find the desired movement of the landmarks of a current shape estimate to new and better locations. The gray-level information (image data) is essential to finding such suggested movements. This implies that we need to model the gray level information and make such a model available during image search. This may be done by examining the intensity profiles at each landmark and normal to the boundary created by the landmark and its neighbors. The intensity profiles are then used to derive a normalized intensity difference (gradient, or derivative) profile giving invariance to the offsets and uniform scaling of the gray levels [25]. With \( L \) landmarks for each shape and \( N \) shapes in the training set along with \( N \) training images, we derive \( N \) profiles for each landmark, one from each image, and calculate the mean profile for each landmark using

\[
\bar{\mathbf{y}}_j = \frac{1}{N} \sum_{i=1}^{N} \mathbf{y}_{ij}, \tag{40}
\]

where \( \mathbf{y}_{ij} \) is the normalized derivative profile for the \( j \)th landmark in the \( i \)th image, and \( \bar{\mathbf{y}}_j \) is the mean normalized derivative profile for the \( j \)th landmark.
2.1.3. Model Deformation and Attraction to Image Features

To locate an instance of an object in a new image, we start with an initial shape estimate and then examine the neighborhood of the landmarks of our estimate aiming at finding better locations for the landmarks. We then change the shape and pose of our estimate to better fit the new locations of the landmarks while producing in the process a new acceptable or allowable shape.

An instance of an object is described as the sum of the mean shape obtained from the training and a weighted sum of the principal components, with the possibility of this shape being translated, rotated, or scaled. The weights of the principal components are called the shape parameters and the values for the rotation angle and the scaling factor, and the translation values in the x and y directions are called the pose parameters.

Starting with an initial shape, we extract a perpendicular intensity profile for each landmark and use it to derive a normalized derivative search profile in the same way as described in Section 2.1.2. Within this search profile, we seek a subprofile that matches the mean or expected profile obtained in the training. This can be done by defining the following square-error function, which decreases as the match improves, and minimizing it with respect to the location of this subprofile, \( d \):

\[
 f(d) = (h(d) - \bar{y}_j)^T(h(d) - \bar{y}_j),
\]

(41)

where \( \bar{y}_j \) is the mean normalized derivative profile for the \( j \)th landmark, and \( h(d) \) is a subprofile along the search profile having a length equal to that of \( \bar{y}_j \) and centered around the point on the search profile that is offset by \( d \) from the landmark.

Finding \( d \) that minimizes the above function for all the landmarks, we arrive at \( L \) new locations for the landmark positions. We may choose to control the suggested movement by, for example, changing a small suggested movement to no movement or large suggested movements to only half what is suggested.

The next step is to update the pose and shape parameters in order to move as closely as possible to the new proposed positions with the restriction that the new produced shape is an allowable shape. A practical, though not optimal, solution is to first find only the pose parameters that move the shape estimate as close as possible to the new proposed positions. Then there will remain residual adjustments that can only be satisfied by deforming the shape and hence changing the shape parameters. Finding the pose parameters is done by aligning the current estimate to the new proposed shape. Alignment details are presented in the section for the spatio-temporal case. The remaining landmark position modifications are in general \( 2L \)-dimensional, whereas the shape variations obtained from the model are only \( t \)-dimensional. A least-squares solution can be used to solve the following equation for the changes in shape parameters \( db \) (with an orthonormal matrix
column of \( \mathbf{P} \) we have \( \mathbf{P}^{T} \mathbf{P} = \mathbf{I} \)

\[
dx = \mathbf{P} \mathbf{d} \mathbf{b} \Rightarrow \mathbf{d} \mathbf{b} = (\mathbf{P}^{T} \mathbf{P})^{-1} \mathbf{P}^{T} dx = \mathbf{P}^{T} dx,
\]

where \( dx \) is a vector containing the remaining landmark position modifications, \( \mathbf{d} \mathbf{b} \) is a vector of changes in the shape parameters, and \( \mathbf{P} \) is the matrix of principal components.

Finally, we limit the shape variations to obtain an acceptable or allowable shape within the ASD by applying the constraints on the shape parameters. With this step we complete a single iteration. In a similar fashion, we obtain new estimates and re-iterate until we assume convergence, defined when the parameter changes are insignificant.

2.1.4. Improvements and Extensions to ASMs

In this section we present several enhancements and additions to the basic ASM method. We start by presenting an automatic landmark generation algorithm followed by a multi-resolution implementation of the ASM. Active Appearance Models (AAMs) are then discussed. The use of genetic algorithms and the application of Kalman filtering for tracking using ASMs are also presented here. We then show how vibrational modes can be added to the original variation modes of PDMs. A method for estimating the shape distribution is briefly discussed, and finally some comments on classification using ASMs are presented.

2.2. Automatic Landmark Generation for PDMs

In order to obtain the (PDM) when the shapes are represented by landmarks, it is required that all the landmarks in all the training set of images are individually specified. This is typically done by hand, clicking a mouse on the locations of the landmarks on an image. This is obviously time-consuming and affected by inter- and intra-operator variability. It is also particularly difficult to place the landmarks appropriately when building a 3D model from volume images. A method for automatically generating PDMs is proposed in [26]. The suggested method comprises two main stages. First, a pairwise corresponder is used to establish an approximate set of landmarks on each of the example boundaries. The pairwise corresponder identifies corresponding points on different examples of the object, by using the curvature of the boundary of the objects as a basis for a dynamic programming matching algorithm. The dynamic programming algorithm attempts to match points of high curvature on the two boundaries so as to minimize the discrepancy in the curvature over the length of the boundary. Then, the landmarks are refined using an iterative nonlinear optimization scheme to generate a more compact PDM. The refinement is done iteratively by varying the pose and the shape of the landmarks associated with a given member of the training set in order to minimize a cost function. One term of the cost function decreases as the
shape becomes more similar to the mean shape, thus compacting the PDM model, while another term increases when the landmarks move away from a given boundary. More recently this line of work has been formalized through the concept of Minimum Description Length.

**Multi-Resolution Image Search** An important issue that affects the image search considerably in the ASM is choice of the length of the search profile. In choosing the length of the search profile we are faced with two contradicting requirements: on one hand, the search profile should be long enough to contain within it the target point (the point that we need the landmark point to move to); on the other, we require the search profile to be short for two reasons: first, to reduce the computations required, and second, if the search profile is long and the target point is close to the current position of the landmark, then it will be more probable to move to a farther away noisy point and miss the target. In [27] a multi-resolution approach is suggested where at first the search looks for faraway points and makes large jumps, and as the search homes in on a target structure, it should restrict the search only to close points.

In order to achieve such multi-resolution search, we generate a pyramid of images. This pyramid of images contains the same image but with different resolutions. At the base of the pyramid, Level 0, we have the original image. On the next level we have a lower-resolution version of the image with half the number of pixels along each dimension, and so on, until we reach the highest level of the pyramid. Low pass filtering (smoothing), for example using Gaussian filtering, should always precede image sub-sampling.

The search will begin at the top level of the pyramid (i.e., image with the lowest resolution). The search is then initiated one level below using the search output of the previous level, and so on, until the lowest level of the pyramid (the original image) is reached. In order to be able to perform such a search in each level, we should be equipped with information about the gray-level profiles in each of these levels. This demands that during the training stage we obtain the mean normalized derivative profile for each landmark at all levels of the pyramid.

The criterion utilized in order to change the level of search within the pyramid is as follows. Move to a lower level when a certain percentage of landmarks do not change considerably, for example, when 95% of the landmarks move only within the central 50% of the search profile. A maximum number of iterations can also be devised to prevent getting stuck at a higher level.

**Active Appearance Models (AAMs)** ASMs are statistical models of the shape variations of an object that are used along with additional information about the gray level values to segment new objects. AAMs are an extension of ASMs that combine information about the shape and intensity of an object into one model describing the appearance [28]. The model parameters are changed to locate new
instances of the object in new images. The changes are found by measuring a residual error between the appearance of the model and the data. The relationship between the changes in the model parameters and the residual error is found in the training stage. More details follow.

A shape model and a gray-level model are produced in similar fashion to ASMs then combined to obtain an appearance model. The shape model is given by

$$\mathbf{x} = \bar{\mathbf{x}} + \mathbf{P}_s \mathbf{b}_s.$$  

(43)

The gray level model is obtained by first warping the examples to the mean shape; then the gray-level values are sampled from the shape-normalized image. In order to reduce the effects of global lighting variation, the sampled values are normalized. The gray level model is written as

$$\mathbf{g} = \bar{\mathbf{g}} + \mathbf{P}_g \mathbf{b}_g.$$  

(44)

The complete appearance parameters are the shape and the gray level parameters, $\mathbf{b}_s$ and $\mathbf{b}_g$, combined with the four pose which perform scaling by $s$, rotation by $\theta$, and translation by $t_x$ and $t_y$.

In the AAM search process, the task is to minimize the difference between a new image and one synthesized by the appearance model. The relationship between the difference and the model parameters is approximated linearly. The coefficients of this linear transformation are obtained in a training stage by varying the model parameters and observing the change in the image intensities.

Active Blobs (ABs) [29] constitute an approach similar to AAMs. This method uses the image difference to drive tracking by learning the relationship between image error and parameter offset in an off-line processing stage. The main difference is that ABs are derived from a single example with low-energy mesh deformations (derived using a Finite-Element Method), whereas AAMs use a training set of examples.

Genetic Algorithms  Genetic algorithms employ mechanisms analogous to those involved in natural selection to conduct a search through a given parameter space for an extreme value of some objective function. This concept can be employed in ASMs, where the parameter space consists of possible ranges of values for the pose and shape parameters of the model. The objective function to be maximized reflects the similarity between the gray levels related to the object in the search stage and those found from training. In [30] the use of GA for model-based image interpretation is presented and can be used to fit the formulation of ASMs. The use of GA for finding the model parameters that fit a model to image data is presented below.

A point in the search space is encoded as a chromosome and a population of chromosomes is maintained. Combining existing points generates new points
in the search space. Solutions are obtained by iteratively generating new generations of chromosomes by selective breeding based on the relative values of the objective function for different members of the population. The set of shape and pose parameters is encoded as a string of genes to form a chromosome. Each gene can take one of several values called alleles. It has been shown that long chromosomes with few alleles are preferable to shorter chromosomes with many alleles. Consequently, chromosomes usually contain parameters encoded as bit strings. Mutation and crossover are applied to chromosomes to produce children. Crossover is when the two parent chromosomes are cut at a certain position and the opposing sections combined to produce two children. Mutation is done by selecting a gene and changing its value. In the case of binary representation, the bit is complemented. In the iterative search process, the fitter an individual, i.e., the higher the value of the objective function obtained from the parameters of the chromosome of the individual, the more this individual will be used to produce children. Moreover, certain regions in the search space will be allocated more trials due to the existence of some evidence that these regions are fitter.

**Tracking Using ASMs and Kalman Filtering** A simple implementation of tracking was mentioned in [31], where the investigators simply set the initial approximation of model parameters in a certain frame to the solution from the previous frame in an image sequence. Here we describe the use of Kalman filtering for tracking using an ASM as described by [32].

The state space vector is composed of the shape parameters, the origin of the shape, the velocity of the origin, and alignment terms. The shape parameters are modeled as a simple discrete stochastic process, where the current shape parameter is equal to the previous parameter plus an additive random noise term. The shape parameters are assumed to vary independently, and the variance of the noise is set to be proportional to the variance of the shape parameter (the corresponding eigenvalue obtained in the PCA). The origin is assumed to undergo a uniform 2D motion with a randomly varying force represented by an additive random noise. The alignment parameters (two parameters incorporating rotation and scaling) are assumed constant with additive random noise.

The observations (observed features) are obtained at each new frame and represented by the points of maximum contrast. Measurements are made by searching along the normal to the estimated contour at the boundary points. The measurement model is formulated according to the fact that the curve points are related to the state space parameters as a rotated, scaled, and translated version of the mean shape plus the principal components weighted by the shape parameters. This measurement model is nonlinear; hence the extended Kalman filter can be used. Another approach that simplifies the problem is suggested. The change in origin is first estimated with the assumptions that the shape and alignment parameters are fixed. Then the effect of the origin shift is removed and the alignment parameters are estimated. Finally, the effect of change in alignment is removed.
and the shape parameters updated. Following the above assumptions, the standard
discrete Kalman filter is used to update the state estimates and covariance.

**Augmenting the Statistical Variations with Vibrational Modes** In [33] a method
is presented for combining two approaches to modeling flexible objects. Modal
analysis using Finite-Element Methods (FEMs) generates a set of vibrational
modes for a single shape. Point Distribution Models (PDMs) generate a statistical
model of shape variation from a set of training example shapes. The proposed
method combines the two and generates vibrational modes when few example
shapes are available and changes gradually to using more statistical modes of
variation when a large set is presented.

Finite-Element Methods take a single shape and treat it as if it were made
of a flexible material. The techniques of Modal Analysis give a set of linear
deformations of the shape equivalent to the modes of vibration of the original
shape. Although a flexible model of a shape with these vibrational modes can be
used in image search, the resulting shapes may not represent the real variations
that occur in a class of shapes.

Similar to previous discussions, a shape here is represented by a set of land-
marks (coordinate points); the landmarks are considered nodes with certain masses
that are mutually interconnected by springs with constant stiffness and rest lengths
equal to the distance between the points.

In an FEM, new shapes can be generated using

\[ \mathbf{x} = \hat{\mathbf{x}} + \Phi \mathbf{u}, \]  

(45)

where \( \hat{\mathbf{x}} \) is a vector of the original points of the shape include this here, \( \Phi \) is a matrix
of eigenvectors representing the vibrational modes, and \( \mathbf{u} \) is a set of parameters.

Matrix \( \Phi \) can be found by solving the following generalized eigensystem:

\[ \mathbf{K} \Phi = \mathbf{M} \Phi \Omega^2, \]  

(46)

where \( \mathbf{K} \) is the stiffness matrix and \( \mathbf{M} \) is the mass matrix.

Matrix \( \Omega^2 = diag \left( \omega_1^2, \omega_2^2, \ldots, \omega_{2L}^2 \right) \) is a diagonal matrix of eigen-
values associated with the eigenvectors of \( \Phi \), and \( \omega_i^2 \) is the frequency of the \( i \)th
vibrational mode.

For a shape of \( L \) landmarks \( \Phi, \mathbf{M}, \) and \( \mathbf{K} \) are \( 2L \times 2L \) matrices. Vectors \( \hat{\mathbf{x}} \)
and \( \mathbf{x} \) are both of length \( 2L \). Matrix \( \mathbf{M} \) is set to the identity matrix, and \( \mathbf{K} \) can be
calculated as

\[ \mathbf{K} = \begin{pmatrix} \mathbf{K}_{xx} & \mathbf{K}_{yx} \\ \mathbf{K}_{xy} & \mathbf{K}_{yy} \end{pmatrix}, \]  

(47)
where \( \mathbf{K}_{xx}, \mathbf{K}_{yx} = \mathbf{K}_{xy}^T \), and \( \mathbf{K}_{yy} \) are \( L \times L \) with off-diagonal elements given by
\[
\begin{align*}
    k_{xxij} &= d^2_{xij}/d^2_{ij} \\
    k_{yyij} &= d^2_{yij}/d^2_{ij} \\
    k_{xyij} &= d_{xij}d_{yij}/d^2_{ij}
\end{align*}
\]  
and diagonal elements \( k_{yyii} = k_{xxii} = 1.0 \) and \( k_{xyii} = 0 \), where
\[
\begin{align*}
    d_{xij} &= x_i - x_j \\
    d_{yij} &= y_i - y_j \\
    d^2_{ij} &= d^2_{xij} + d^2_{yij}
\end{align*}
\]  
(x, y are from the vector x). (49)

One suggestion for combining the FEM and PDM is by using an equation of the form
\[ \mathbf{x} = \mathbf{\hat{x}} + \mathbf{\Phi u} + \mathbf{Pb}, \]
where \( \mathbf{\hat{x}} \) is a mean shape, \( \mathbf{P} \) is a matrix representing the statistical variations, and \( \mathbf{b} \) is a vector of parameters. Since the statistical modes of variation and the vibrational modes are not independent, this approach is unsatisfactory due to the redundancy in the model.

An alternative approach for combining the models is as follows. If only one shape example exists, then we are left with only the FEM model. Given two shape examples, we calculate the vibrational modes for each shape, and use them to generate more shapes by randomly selecting the parameters \( \mathbf{u} \). We then train a PDM on this new set of examples. If more than two shapes exist, then we need to reduce the effect of the vibrational modes and increase the effect of the statistical modes of variation because, as mentioned earlier, the vibrational modes may not represent real shape variations of a certain class of objects.

This idea can be realized by first choosing the distribution of \( \mathbf{u} \) to have a zero mean and a diagonal covariance matrix \( \mathbf{S}_u = \alpha \mathbf{A} \), with \( (\mathbf{S}_u)_{ii} = \alpha \omega_i^{-2} \) (this gives more variations in the low-frequency vibrational modes, and less variation in the high-frequency vibrational modes), and then calculating the eigenvectors and eigenvalues of the “combined-model” covariance matrix of the generated example shapes. Now, the derivation of this covariance matrix is presented.

Starting with \( m \) original example shapes: \( \mathbf{x}_i \) for \( 1 \leq i \leq m \), with a mean
\[ \mathbf{\bar{x}} = \frac{1}{m} \sum_{i=1}^{m} \mathbf{x}_i. \]  
(51)

The PDM is obtained by finding the eigenvectors of the covariance matrix:
\[ \mathbf{S} = \frac{1}{m-1} \sum_{i=1}^{m} (\mathbf{x}_i - \mathbf{\bar{x}})^T(\mathbf{x}_i - \mathbf{\bar{x}}). \]  
(52)
One example shape \( x_i \) can be used to produce other shapes from its vibrational modes as

\[
x_{gi} = x_i + \Phi_i u,
\]

where \( x_{gi} \) is a shape generated from the vibrational modes of \( x_i \).

The covariance matrix of \( x_{gi} \) is given by

\[
S_{gi} = S + \text{cov}(\Phi_i u) = S + \Phi_i S_u \Phi_i^T.
\]

Using (\( S_u = \alpha \Lambda \)), we obtain

\[
S_{gi} = S + \alpha \Phi_i \Lambda \Phi_i^T.
\]

The “combined-model” covariance, \( S_c \), of the shapes \( (x_{gi}, 1 \leq i \leq m) \) generated from the original shapes \( (x_i, 1 \leq i \leq m) \) can be estimated by averaging the individual covariances as

\[
S_c = \frac{1}{m} \sum_{i=1}^{m} S_{gi} = \frac{1}{m} \sum_{i=1}^{m} S + \alpha \Phi_i \Lambda \Phi_i^T.
\]

This gives

\[
S_c = S + \alpha \left( \frac{1}{m} \sum_{i=1}^{m} \Phi_i \Lambda \Phi_i^T \right).
\]

Now we find the eigenvectors and eigenvalues of \( S_c \) and proceed as with the normal PDM.

If we have only one example, then from (51) we get \( \bar{x} = x_1 \), and from (52) we get \( S = 0 \), and thus we have only the effects of the FEM model in (55). If the number of samples is very large, then we wish to ignore the effects of the vibrational modes. This can be done by setting \( \alpha \) to zero, and then \( S_c = S \), and thus we obtain the PDM for the original example shapes. In the intermediate cases, we wish to have a large value for \( \alpha \) when there are few original examples and smaller values as more examples become available. This may be achieved by setting \( \alpha \propto 1/m \).

**Extension to 3D Data** The mathematical foundation formulation of PDMs can be directly extended to 3D. What remains is the specific implementation details of the representation of 3D surfaces and the search algorithm used. In [30] the surfaces are represented by a set of 3D landmarks. A vector of length \( 3L \) is used to represent a surface with \( L \) landmarks and can be written as

\[
x = \begin{bmatrix} x_1, y_1, z_1 & x_2, y_2, z_2 & \ldots & x_L, y_L, z_L \end{bmatrix}^T.
\]

In the training stage of the 2D PDM as well as in 3D, we need to provide the coordinates of the landmarks for each shape or surface in the training set. The
problem is solved in 2D, by providing a human expert with a utility to “point and click” landmarks on a 2D image. In 3D it is not trivial to build a utility that allows the operator to navigate in a 3D space of intensity values and locate the landmarks. In the currently discussed work, the segmentation is generated by hand in a slice-by-slice fashion and the aligning procedure allowed for rotation only in the slice plane, in addition to scaling and translation. In the search stage, the strongest edge or a representative profile is sought along a search profile normal to the 3D surface.

**Kernel-Based Estimation of the Shape Density Distribution**  In Section 2.1.1 we represented the training set of shapes as a cloud of points in a $2L$-dimensional space. The ASD was defined to be a hyper-ellipsoid of some reduced dimensionality (given by the main mode of variation obtained by PCA). An alternative approach to model the allowable shape domain and utilize it in image search is as follows. As before, we are first presented with a set of example shapes represented by a vector in a multidimensional space that we align to a common coordinate frame. Now we estimate the shape probability density function (PDF) using a certain kernel, for example, convolving the shape points with a multidimensional Gaussian. In the search process, the gradient of the PDF can be used to lead us to a more plausible (or in this case, more probable) shape. The reader is referred to [27] for more practical details.

**Classification Using ASMs**  The idea of using an ASM for classification can be explained by summarizing the work done in [34] on recognizing human faces. The shape model and the local gray level profile model are augmented with a model for the appearance of a shape-free face obtained by deforming each face to have the same shape as the mean face (using thin plate splines as described in [35]). In the training stage a number of training faces of the same and different individuals were outlined. In the application stage, an ASM is used to segment a face, and then the shape, shape-free, and local gray-level parameters are extracted and used as features for classification.

### 3. STATISTICALLY CONSTRAINED SNAKES: COMBINING ACMS AND ASMS

In this section we present a method for constraining the deformations of snakes in a way that is consistent with a training set of images. The described methodology is demonstrated by locating the left-ventricular boundary in echocardiographic images.
3.1. Introduction

Ultrasound echocardiography is a valuable noninvasive and relatively inexpensive tool for clinical diagnosis and analysis of heart functions including ventricular wall motion. An important step toward this analysis is segmentation of endocardial boundaries of the left ventricle (LV) [20, 36–40]. Although segmenting anatomical objects in high-SNR images can be done with simple techniques, problems do arise when the images are corrupted with noise and the object itself is not clearly or completely visible in the image. This is clearly the case in heart images obtained by ultrasonography, which are characterized by weak echoes, echo dropouts, and high levels of speckle noise. These image artifacts often result in detecting erroneous object boundaries or failing to detect true ones. Snakes [2] and its variants [11, 41–43] overcome parts of these limitations by considering the boundary as a single, inherently connected, and smooth structure, and also by supporting intuitive, interactive mechanisms for guiding the segmentation. In our application of locating the human LV boundary in echocardiography, human guidance is often needed to guarantee acceptable results. A potential remedy is to present the snake with a priori information about the typical shape of the LV. Statistical knowledge about shape variation can be obtained using PDMs, which are central to the ASMs segmentation technique [33]. PDMs, which are obtained by performing PCA on landmark coordinates labeled on many example images, have been applied to the analysis of echocardiograms [37]. However, this procedure is problematic since manual labeling of corresponding landmark points is required. In our application it is tedious to obtain a training data set delineated by experts with point correspondence, let alone the fact that defining a sufficient number of landmarks on the LV boundary is a challenging task in and of itself.

The method described in this section is similar to both ASMs, but without the landmark identification and correspondence requirement, and ACMs, but enforced with a priori information about shape variation. We adopt an approach similar to PDMs for capturing the main modes of ventricular shape variation. However, in our method, rather than representing the object boundaries by spatially corresponding landmarks, we employ a frequency-based boundary representation, namely Discrete Cosine Transform (DCT) coefficients. These new shape descriptors eliminate the need for spatial point correspondence. PCA, which is central to ASMs, is applied to this set of shape descriptors. An average object shape is extracted along with a set of significant shape variational modes. Armed with this model of shape variation, we find the boundaries in unknown images by placing an initial ACM and allowing it to deform only according to the examined shape variations. A similar approach using Fourier descriptors and applied to locating the corpus callosum in 2D MRI was reported in [60]. Results of segmenting the human LV in real echocardiographic data using the discussed methodology are also presented.
3.2. Methods

3.2.1. Overview

This section presents a general overview of the method (see Figure 15). We used snakes as the underlying segmentation technique. In order to arm the snake model with a priori information about the typical shape variations of the LV that may be encountered during the segmentation stage, a training set of images is provided. This set is manually delineated by medical experts without the requirement of complete landmark correspondence between different images. The entire set of manually traced contours is then studied to model the typical ventricular shape variations. This is done by first applying a re-parametrization of the contours, which gives a set of DCT coefficients replacing the spatial coordinates. We then apply PCA to find the strongest modes of shape variation. The result is an average ventricular shape, represented by a set of average DCT coefficients, in addition to the principal components, along with the fraction of variation each component explains. To segment a new image of the LV, we initialize a snake and, unlike classical snakes, do not allow it to freely deform according to internal and external energy terms, but instead we constrain its deformations in such a way that the resulting contour is similar to the training set. To attain the constrained deformations we obtain the vector of DCT coefficients for the active contour coordinates, project it onto an allowable snake space defined by the main principal components, and then perform an Inverse DCT (IDCT), which converts the constrained DCT coefficients back to spatial coordinates. This is repeated until convergence, which is reached when the majority of snake nodes do not change their locations significantly. The shape models generated are normalized with respect to the similarity transformation parameters: rotation angle, scaling factor, and two translation parameters.

Figure 15. Flowchart depicting the main steps involved in the use of a statistically constrained snake for image segmentation. Reprinted with permission from [62]. Copyright ©2000, IEEE.
3.2.2. Representing Contours by DCT Coefficients

We use snakes as the underlying segmentation technique. A snake contour is originally represented by a set of \( N \) nodes \( \{ v_i(t) = (x_i(t), y_i(t)) \} \), \( i = 1, 2, \ldots, N \), and is deformed according to internal and external forces. Snake contour re-parametrization is obtained via the use of the one-dimensional Discrete Cosine Transform (DCT) of the snake coordinates. The 1D DCT of sequence \( x_i \) of snake contour coordinates is defined as

\[
X(k) = w(k) \sum_{i=1}^{N} x_i \cos \frac{\pi (2i-1)(k-1)}{2N}, k = 1, \ldots, N, \tag{56}
\]

and the inverse DCT is given as

\[
x_i = \sum_{k=1}^{N} w(k) X(k) \cos \frac{\pi (2i-1)(k-1)}{2N}, i = 1, \ldots, N, \tag{57}
\]

where

\[
w(k) = \begin{cases} 
\frac{1}{\sqrt{N}}, & k = 1, \\
\sqrt{\frac{2}{N}}, & 2 \leq k \leq N,
\end{cases} \tag{58}
\]

and \( X(k) \) are the DCT coefficients. Similar equations are used for the \( y_i \) coordinates and the \( Y(k) \) DCT coefficients. The DCT was favored as the new frequency domain shape parametrization because it produces real coefficients, has excellent energy compaction properties, and the correspondence between the coefficients (when transforming contours with no point correspondence) is readily available. The latter property stems from the fact that corresponding DCT coefficients capture specific spatial contour frequencies.

3.2.3. Principal Component Analysis

In order to identify the main modes of shape variation found in the training contours, we perform PCA on the DCT coefficients representing them. The same number, say \( M \), of DCT coefficients is obtained for the set of \( x \) and \( y \) coordinates that represent each shape in the training set. This is done either by interpolating the spatial coordinates or truncating the DCT coefficients. PCA yields the principal components (PCs) or main variation modes, \( a_j \), and the variance explained by each mode, \( \lambda_j \). The first \( t \) PCs, sufficient to explain most of the variance, are used, i.e., \( j = 1, 2, \ldots, t \). The average of the coefficient vectors, \( \bar{X} \), is also calculated. The same procedure is performed for the \( y \) coordinates.

3.2.4. Constraining Contour Deformation

Subsequent to providing a set of images containing the object of interest, the training set of tracings is obtained (contours represented by coordinate-vectors
of varying length with no point correspondence). DCT coefficients \((X)\) are then obtained followed by PCA. Presented with a new image, a snake contour is first initialized by specifying the starting and endpoints of the contour, and then allowed to deform by applying forces that minimize traditional energy terms. In order to guarantee a snake contour resembling an acceptable shape (similar to those in the training set), we constrain the resulting deformed contour, \(\{v_i(t), i = 1, \ldots, N\}\), by projecting vector \(X\) (consisting of \(M\) DCT coefficients) onto the subspace of principal components (the allowable shape space) according to

\[
X_{\text{proj}} = \bar{X} + Ab,
\]

where \(b\) is a vector of scalars weighting the main variation modes in \(A\) and is calculated as

\[
b = (A^T A)^{-1} A^T (X - \bar{X}),
\]

and \(A = [a_1, a_2, \ldots, a_t]\). Prior to performing the IDCT, we restrict the projected coefficients \((X_{\text{proj}})\) to lie within \(\pm 3\sqrt{\lambda_j}\), since in this application the population typically lies within three standard deviations from the mean. Again, the same procedure is performed for the \(y\) coordinates. The statistical constraints can be applied after each snake deformation step or only when the resulting DCT coefficients of the snake contour are different (using some norm and threshold) from the mean coefficients. Note that the DCT coefficients are obtained and constrained for shapes normalized with respect to similarity transformation parameters utilizing corresponding starting and ending contour points.

3.3. Results

The described methodology was applied for segmenting the LV ventricle in real echocardiographic images. We collected 105 images of the human LV. The ventricular boundaries were manually traced by a medical expert. There was no point correspondence between the frames, with the number of traced points varying between 28 and 312 (Figures 16 and 17). The DCT of the manual tracings was then obtained.

Figure 18 shows an example of the manual tracings and the resulting contour after IDCT of the truncated DCT coefficients. The ratio “energy of truncated contour”/“energy of the original contour” for increasing numbers of DCT coefficients was examined in order to determine how many DCT coefficients to use. This was followed by a PCA of the truncated DCT coefficients. Five variation modes of 56 possible were enough to explain 95% of the total variation, 12 were enough for 99%, and 24 for 99.9%. Figure 19 depicts the first and second shape variation modes found in the training set.

To illustrate applying shape constraints we used test examples. We began with one of the manual tracings, added Gaussian noise, performed DCT, truncated certain DCT coefficients, projected the remaining coefficients on the allowable shape
Figure 16. Sample of the echocardiographic training image set.

Figure 17. Manual tracings of the LV boundary in the training images.
Figure 18. Ultrasound image with manual tracing (continuous) and the contour after IDCT of truncated DCT coefficients (dots). Reprinted with permission from [62]. Copyright ©2000, IEEE.

Figure 19. Mean contour and the first and second variation modes (weighted by ±1 std). Reprinted with permission from [62]. Copyright ©2000, IEEE.
space, and then performed the IDCT. It was visually obvious how the constrained contour resembles a much more plausible boundary of the LV than the noisy one (Figure 20).

![Figure 20](image)

**Figure 20.** (a) Manual tracing. (b) Noisy version of (a). (c) IDCT of truncated DCT coefficients of (b). (d) Projection of (c) on the allowable shape space (note the similarity to (a)). Reprinted with permission from [62]. Copyright ©2000, IEEE.

More importantly, results on real echocardiographic data were obtained by initializing the snake on an image that was not included in the training set (i.e., cross-validation was used) and then allowing it to deform under forces that minimize its energy. This was followed by a DCT–Truncation–Projection–IDCT procedure. The outcome of the snake segmentation alone, due to noise and echo dropouts in the image, often gave unreasonable and unacceptable LV shapes. Conversely, employing constrained deformations resulted in acceptable LV boundaries (Figures 21 and 22).
Figure 21. Snake contours (dashed) and constrained contours (continuous) with increasing number of iterations (left to right, top to bottom). Reprinted with permission from [62]. Copyright ©2000, IEEE.

Figure 22. Progress ((a)–(d)) of a snake overlain on an ultrasound image of the left ventricle (dashed), and the result of DCT–Truncation–Projection–IDCT (continuous). Reprinted with permission from [62]. Copyright ©2000, IEEE.
3.4. Summary

We presented a method for constraining the deformations of an active contour according to training examples and applied it to segmenting the human left ventricle in echocardiographic (ultrasound) images. To capture the typical shape variations of the training set, principal component analysis was performed on frequency-domain shape descriptors in order to avoid the drawbacks associated with labeling corresponding landmarks (only the start and endpoints of the contours correspond). The method utilizes the strength of ACMs in producing smooth and connected boundaries along with the strength of ASMs in producing shapes similar to those in a training set. More plausible LV shapes resulted when employing the new method compared to classical snakes.

4. DEFORMABLE SPATIOTEMPORAL SHAPE MODELS: EXTENDING ACTIVE SHAPE MODELS TO 2D+TIME

In this section we extend 2D ASMs to 2D+time by presenting a method for modeling and segmenting spatiotemporal shapes (ST shapes). The modeling part consists of constructing a statistical model of ST shape parameters. This model describes the principal modes of variation of the ST shape in addition to constraints on the allowed variations. An active approach is used in segmentation where an initial ST shape is deformed to better fit the data, and the optimal proposed deformation is calculated using dynamic programming. Segmentation results on both synthetic and real data are presented.

4.1. Introduction

Much work has been done on tracking rigid objects in 2D sequences. In many image analysis applications, however, there is a need for modeling and locating non-rigid time-varying object shapes. One approach for dealing with such objects is the use of deformable models. Deformable models [59] such as snakes [2] and its variants [16, 42–45], have attracted considerable attention and are widely used for segmenting non-rigid objects in 2D and 3D (volume) images. However there are several well-known problems associated with snakes. They were designed as interactive models and therefore rely on a user to overcome initialization sensitivity. They were also designed as general models showing no preference among a set of equally smooth shapes. This generality can cause unacceptable results when snakes are used to segment objects with shape abnormalities arising from occlusion, closely located but irrelevant structures, or noise. Thus, techniques that incorporate a priori knowledge of object shape were introduced [46, 47]. In ASMs [46] the statistical variation of shapes is modeled beforehand in accordance with a training set of known examples. In order to attack the problem of tracking non-rigid time-varying objects, deformable models were extended to dynamic
deformable models [16, 29, 48–51]. These describe the shape changes (over time) in a single model that evolves through time to reach a state of equilibrium where internal forces, representing constraints on shape smoothness, balance the external image forces and the contour comes to rest. Deformable models have been constructed by applying a probabilistic framework and lead to techniques such as “Kalman snakes” [52]. Motion tracking using deformable models has been used for tracking non-rigid structures such as blood cells [48], and much attention has been given to the human heart and tracking of the left ventricle in both 2D and 3D [16, 49, 50, 53]. In addition to tracking rigid objects, previous work has focused on arbitrary non-rigid motion and gave little attention to tracking objects moving in specific motion patterns, without incorporation of statistical prior knowledge in both 2D and time [54].

We present a method for locating ST shapes in image sequences. We extend ASMs [46] to include knowledge of temporal shape variations and present a new ST shape modeling and segmentation technique. The method is well suited to model and segment objects with specific motion patterns, as in echocardiography.

4.2. Method

In order to model a certain class of ST shapes, a representative training set of known shapes is collected. The set should be large enough to include most of the shape variations we need to model. Next, all the ST shapes in the training set are parametrized. A data dimensionality reduction stage is then performed by capturing only the main modes of ST shape variations. In addition to constructing the ST shape model, the training stage also includes the modeling of gray-level information. The task is then to locate an ST shape given a new unknown image sequence. An average ST shape is first initialized, “optimal” deformations are then proposed, and the deformations are constrained to agree with the training data. The proposed changes minimize a cost function that takes into account both the temporal shape smoothness constraints and the gray-level appearance constraints. The search for the optimum proposed change is done using dynamic programming. The following sections present the various steps involved in detail.

4.2.1. Statistical ST Shape Variation

The Training Set  We collect \( N \) training frame sequences each with \( F \) frames. The training set, \( \Phi_V = [V_1, V_2, \ldots, V_N] \), displays similar objects and similar object motion patterns. \( \Phi_V(i) = V_i = [f_{i1}, f_{i2}, \ldots, f_{iF}] \) is the \( i \text{th} \) frame sequence containing \( F \) frames, and \( V_j(i) \equiv \Phi_V(i, j) = f_{ij} \) is the \( j \text{th} \) frame of the \( i \text{th} \) frame sequence containing the intensity value \( f_{ij}(r, c) \equiv \Phi_V(i, j, r, c) \) at the \( r \text{th} \) row and \( c \text{th} \) column of the frame.
The ST Shape Parameters  We introduce $S_i$ to denote the parameter vector representing the $i$th ST shape. Parametrization is done using landmarks (other shape parametrization methods may be utilized, e.g., Fourier descriptors [55] or B-Splines [51]). Landmarks are labeled either manually, as when a cardiologist labels the heart chamber boundaries [30, 46], or (semi-)automatically [26]. Each landmark point is represented by its coordinate. Using $L$ landmarks per frame and $F$ frames per sequence, we can write the training set of ST shapes as $\Phi_S = [S_1, S_2, \ldots, S_N]$, where $\Phi_S(i) = S_i = [r_{i1}, r_{i2}, \ldots, r_{iP}]$ is the $i$th ST shape containing $F$ shapes and $S_i(j) \equiv \Phi_S(i, j) = r_{ij}$ is the $j$th shape of the $i$th ST shape. $r_{ij}$ can be written as $r_{ij} = [x_{ij1}, y_{ij1}, x_{ij2}, y_{ij2}, \ldots, x_{ijL}, y_{ijL}]$, where $x_{ijk} = r_{ij}(k, 1) \equiv \Phi_S(i, j, k, 1)$ and $y_{ijk} = r_{ij}(k, 2) \equiv \Phi_S(i, j, k, 2)$ are the coordinates of the $k$th landmark of the shape $r_{ij}$.

ST Shape Alignment  Next, the ST shapes are aligned in order to allow comparing equivalent points from different ST shapes. This is done by rotating, scaling, and translating the shape in each frame of the ST shape by an amount that is fixed within one ST shape. A weighted least-squares approach is used for aligning two sequences and an iterative algorithm is used to align all the ST shapes. Given two ST shapes,

$$S_1 = [x_{111}, y_{111}, \ldots, x_{11L}, y_{11L}, x_{121}, y_{121}, \ldots, x_{12L}, y_{12L}, \ldots, x_{1F1}, y_{1F1}, \ldots, x_{1FL}, y_{1FL}]$$

and

$$S_2 = [x_{211}, y_{211}, \ldots, x_{21L}, y_{21L}, x_{221}, y_{221}, \ldots, x_{22L}, y_{22L}, \ldots, x_{2F1}, y_{2F1}, \ldots, x_{2FL}, y_{2FL}],$$

we need to find rotation angle $\theta$, scaling factor $s$, and the value of translation $(t_x, t_y)$ that will align $S_2$ to $S_1$. To align $S_2$ to $S_1$, $S_2$ is mapped to

$$\hat{S}_2 = [\hat{x}_{211}, \hat{y}_{211}, \ldots, \hat{x}_{21L}, \hat{y}_{21L}, \hat{x}_{221}, \hat{y}_{221}, \ldots, \hat{x}_{22L}, \hat{y}_{22L}, \ldots, \hat{x}_{2F1}, \hat{y}_{2F1}, \ldots, \hat{x}_{2FL}, \hat{y}_{2FL}],$$

using $\hat{S}_2 = M (s, \theta) [S_2] + t$, where $M (s, \theta) [S_2]$ is a rotated then scaled version of each coordinate of $S_2$ (by $\theta$ and $s$, respectively), and $t = [t_x, t_y, t_x, t_y, \ldots, t_x, t_y]^T$ is a translation vector of length $2FL$. The weighted distance between $S_1$ and $\hat{S}_2$ in the $2FL$-dimensional space is given by

$$d^2_{12} = (\hat{S}_2 - S_1)^T W^T W (\hat{S}_2 - S_1),$$

where

$$W = \text{diag}(w_{11x}, w_{11y}, \ldots, w_{1Lx}, w_{1Ly}, w_{21x}, w_{21y}, \ldots, w_{2Lx}, w_{2Ly}, \ldots, w_{FLx}, w_{FLy}).$$
The elements of $W$ reflect our trust in each coordinate and are chosen to be proportional to the “stability” of the different landmarks over the training set [46].

To rotate, scale, and translate a single coordinate, $(x_{2kl}, y_{2kl})$, we use

$$
\begin{bmatrix}
    a_x & -a_y \\
    a_y & a_x
\end{bmatrix}
\begin{bmatrix}
    x_{2kl} \\
    y_{2kl}
\end{bmatrix}
+ \begin{bmatrix}
    t_x \\
    t_y
\end{bmatrix},
$$

where $a_x = s \cos(\theta)$ and $a_y = s \sin(\theta)$.

$\hat{S}_2$ can now be rewritten as $\hat{S}_2 = Az$, where $z = [a_x, a_y, t_x, t_y]^T$ and

$$
A^T = \begin{bmatrix}
    x_{211} & y_{211} & \ldots & x_{21L} & y_{21L} & x_{221} & y_{221} & \ldots & x_{22L} & y_{22L} & \ldots
    & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots
    & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots
    & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots
    & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots
    & 0 & 1 & \ldots & 0 & 1 & \ldots & 0 & 1 & \ldots & 0 & 1
\end{bmatrix}.
$$

Distance $d_{12}^2$ can now be rewritten as $d_{12}^2 = (Az - S_1)^T W^T W (Az - S_1)$, and we can solve for $z$ (least-squares solution) that minimizes $d_{12}^2$ according to $z = ((W^T W)^{-1}) (W^T W S_1) = (A^T W^T W A)^{-1} A^T W^T W S_1$. Once $z = [a_x, a_y, t_x, t_y]^T$ is calculated, $s \theta$ can be found using $\theta = \arctan\left(\frac{a_y}{a_x}\right)$ and $s = a_x / \cos(\arctan\left(\frac{a_y}{a_x}\right))$. We note that when the observed motion patterns in the training sequences span different time intervals, temporal resampling or aligning that incorporates temporal scaling is performed.

**Main ST Shape Variation Modes** The $N$ aligned ST shapes, each of length $2FL$ and represented by $\{S_1, S_2, \ldots, S_N\}$, map to a “cloud” of $N$ points in a $2FL$-dimensional space. It is assumed that these $N$ points are contained within a hyper-ellipsoid of this $2FL$-dimensional space. We call this region the Allowable ST Shape Domain (ASTSD). We then apply PCA to the aligned training set of ST shapes in order to find the main modes of ST shape variation. The resulting principal components (PCs) are the eigenvectors $p_k$ ($1 \leq k \leq 2FL$) of the covariance matrix of the observations, $C_S$, found from $C_S p_k = \lambda_k p_k$. $\lambda_k$ is the $k$th eigenvalue of $C_S$ ($\lambda_k \geq \lambda_{k+1}$) and is equal to the variance along the $k$th PC.

The mean ST shape is calculated as $\bar{S} = \frac{1}{N} \sum_{i=1}^{N} S_i$. The PCs are normalized to unit length and are mutually orthogonal.

**Model Representation** We now express each ST shape, $S_i$, as the sum of the mean ST shape, $\bar{S}$, and a linear combination of the principal modes of variation, $Pb_i$. This gives $S_i = \bar{S} + Pb_i$, where $b_i = [b_{i,1}, b_{i,2}, \ldots, b_{i,2FL}]^T$ and $P = [p_1, p_2, \ldots, p_{2FL}]$. We constrain $b_l$ to $b_{l,\text{min}} \leq b_l \leq b_{l,\text{max}}$ with $b_{l,\text{min}} = -b_{l,\text{max}}$ and $1 \leq l \leq 2FL$. $b_{l,\text{max}}$ is chosen to be proportional to $\sqrt{\lambda_l}$ ($-3\sqrt{\lambda_l} \leq b_l \leq 3\sqrt{\lambda_l}$ is typically used). In practice, only the first $t$ (out of $2FL$) PCs explaining a sufficiently high percentage of the total variance of the original data are used, and the fundamental equation becomes

$$S = \bar{S} + P_t b,$$

(61)
where $b = [b_1, b_2, ..., b_t]^T$, $P_t = [p_1, p_2, ..., p_t]$, and the constraints on $b$ become $b_{l_{min}} \leq b_l \leq b_{l_{max}}$, where $1 \leq l \leq t$.

### 4.2.2. Gray-Level Training

The information contained in the ST shape model alone is typically not enough for spatiotemporal segmentation. Therefore, additional representative information about the intensities or gray levels relating to the object is also desired and collected in the gray-level training stage. In the search stage, new estimates of the ST shape are sought that will better match the gray-level prior knowledge. Different gray-level representative information can be used, e.g., gathering the intensity values in the entire patch contained within the object [28] or parametrizing the profiles or patches around the landmark. In this implementation we follow [46] and use a mean normalized derivative (difference) profile, passing through each landmark and perpendicular to the boundary created by the neighboring ones. For the $k$th landmark this profile is given by

$$
\bar{y}_k = \frac{1}{FN} \sum_{j=1}^{F} \sum_{i=1}^{N} y_{ijk},
$$

(62)

where $y_{ijk}$ is the representative profile for the $k$th landmark in the $j$th shape of the $i$th ST shape. Using gray-level information, temporal and shape constraints, the model is guided to a better estimate of the dynamic object hidden in the new frame sequence.

### 4.2.3. ST Shape Segmentation Algorithm

Given a new frame sequence, the task is to locate the object in all the frames or equivalently locate the ST shape. An initial estimate of the ST shape parameters is chosen at first, and then changes to the parameters are proposed. The pose of the current estimate is then changed and suitable weights for the modes of variation chosen in order to fit the model to the proposed changes. This is done with the restriction that the changes can only be made in accordance with the model (with reduced dimensionality) and the training set. New changes are then proposed, and so on. Here we present a detailed discussion of these steps.

**Initial estimate.** The search starts by guessing an initial ST shape:

$$
\hat{S}^{(0)} = M \left( s^{(0)}, \theta^{(0)} \right) \left[ \bar{S} + P_t b^{(0)} \right] + t^{(0)},
$$

(63)

where $t = [t_x, t_y, t_x, t_y, ..., t_x, t_y]^T$ is of length $2FL$. $M \left( s, \theta \right) [S] + t$ scales, rotates, and translates $S$ by $s$, $\theta$, and $t$, respectively. Both $\bar{S}$ and $P_t$ are obtained from the training stage. A typical initialization would set $b^{(0)}$ to zero and set $s^{(0)}$, $\theta^{(0)}$, and $t^{(0)}$ to values that place the initial sequence in the vicinity of the target.
Proposing a new sequence. For each landmark, say the $k$th landmark in the $j$th frame, we define a search profile $h_{jk} = [h_{jkl1}, h_{jkl2}, \ldots, h_{jklH}]$ that is differentiated and normalized as done with the training profiles. This gives $H^F$ possibilities for the proposed positions of the $k$th landmarks in the $F$ frames (see Figure 23).

Since locating the new positions (1 out of $H^F$ possible) is computationally demanding, we formulate the problem as a multistage decision process and use dynamic programming \cite{41} to find the optimum proposed landmark positions by minimizing a cost function. The cost function comprises two terms: one due to large temporal landmark position changes, and another reflecting the mismatch between the gray-level values surrounding the current landmarks and those expected values found in the gray-level training stage. In the following paragraphs, we detail our implementation of dynamic programming.

We calculate a gray-level mismatch value $M_k(j, l)$ for each point along each search profile in all the frames according to

$$M_k(j, l) = (h_{jk}(l) - \bar{y}_k)^T W^T W (h_{jk}(l) - \bar{y}_k),$$

where $1 \leq k \leq L$, $1 \leq j \leq F$, $1 \leq l \leq H$, $h_{jk}(l)$ is a subprofile of length $G - 1$ anchored at the $l$th location of the search profile $h_{jk}$, and $W$ is a diagonal weighting matrix ($W = I$ was used). Additionally, we calculate a temporal discontinuity value, $d_{jk}(l_j, l_{j-1})$, corresponding to moving the $k$th landmark in frame $j - 1$ to location $l_{j-1}$, and the $k$th landmark in frame $j$ to location $l_j$, each along its respective search profile, according to

$$d_{jk}^2(l_j, l_{j-1}) = (c_{jnx}(l_j) - c_{j-1nx}(l_{j-1}))^2 + (c_{jny}(l_j) - c_{j-1ny}(l_{j-1}))^2,$$

where $c_{jnx} = [x_{jk1}, x_{jk2}, \ldots, x_{jkH}]$ and $c_{jny} = [y_{jk1}, y_{jk2}, \ldots, y_{jkH}]$ are the search profile coordinates of the $k$th landmark in the $j$th frame. We compare the accumulated costs of moving the $k$th landmark to the $l$th position in the $j$th frame, $2 \leq j \leq F$, from any of the $H$ positions in frame $j - 1$ and assign the least value to $A_k(j, l)$, i.e.,

$$A_k(j, l) = \min \{t_{jkl1}, t_{jkl2}, \ldots, t_{jklH}\}$$

$$t_{jklm} = w_d d_{jk}(l, m) + w_m M_k(j, l) + A_k(j - 1, m),$$

where $w_d$ and $w_m$ satisfy $w_d + w_m = 1$, control the relative importance of temporal discontinuity and gray-level mismatch. We also assign an index or a pointer, $P_k(j, l)$, to the location of the best landmark in the previous frames. Applying the same procedure to the $k$th landmark in all the $F$ frames yields $F \times H$ accumulated values and $F \times H$ pointers (no temporal discontinuity cost is associated with the first frame).

To find the proposed positions of the $k$th landmark in all the frames we find the location, call it $m_F$, of the minimum accumulated cost along the search profile of
Figure 23. Proposing a new ST shape. (a) An illustration of an ST shape overlain on an image sequence. The search profiles of one landmark in two frames are shown in white. Examples of proposed landmark positions are shown as black squares. (b) The different choices of the new positions of landmark $i$ in all frames. Reprinted with permission from [63]. Copyright ©2004, Elsevier.
the landmark in the last frame, frame $F$. Then we use $m_F$ to find the proposed landmark position in the second-to-last frame, frame $F-1$, as $m_{F-1} = P_k(F, m_F)$. Its coordinates will be $(c_{F-1kx}(m_{F-1}), c_{F-1ky}(m_{F-1}))$. In general the proposed coordinates of the $k$th landmark of the $j$th frame will be

$$(x, y) : (c_{jkx}(m_j), c_{jky}(m_j)), \quad (68)$$

$$m_j = P_k(j + 1, m_{j+1}). \quad (69)$$

Tracking back to the first frame, we acquire the coordinates of the proposed positions of the $k$th landmark in all frames. Similarly, we obtain the proposed positions for all the landmarks ($1 \leq k \leq L$), which define the ST shape changes $d\hat{S}^{(0)}_{\text{proposed}}$.

**Limiting the proposed sequence.** Since the proposed ST shape ($\hat{S}^{(0)} + d\hat{S}^{(0)}_{\text{proposed}}$) will generally not conform to our model of reduced dimensionality and will not lie in the ASTSD, it cannot be accepted as an ST shape estimate. Therefore, we need to find an acceptable ST shape that is closest to the proposed one. This is done by first finding the pose parameters ($s^{(1)}, \theta^{(1)}$, and $t^{(1)}$) that will align $S$ to $\hat{S}^{(0)} + d\hat{S}^{(0)}_{\text{proposed}}$ by mapping $S$ to $M \left(s^{(1)}, \theta^{(1)}\right) \left[S\right] + t^{(1)}$, and then finding the extra ST shape modifications $dS^{(1)}$ that, when combined with the pose parameters, will map exactly to $\hat{S}^{(0)} + d\hat{S}^{(0)}_{\text{proposed}}$. The latter is done by solving the following equation for $dS^{(1)}$:

$$M \left(s^{(1)}, \theta^{(1)}\right) \left[S + dS^{(1)}\right] + t^{(1)} = \hat{S}^{(0)} + d\hat{S}^{(0)}_{\text{proposed}} \Rightarrow \quad \ (70)$$

$$dS^{(1)} = M \left(s^{(1)}, \theta^{(1)}\right)^{-1} \left[\hat{S}^{(0)} + d\hat{S}^{(0)}_{\text{proposed}} - t^{(1)}\right] - S, \quad \ (71)$$

where $M \left(s^{(1)}, \theta^{(1)}\right)^{-1} = M \left((s^{(1)})^{-1}, -\theta^{(1)}\right)$. In order to find the new shape parameters, $b^{(1)}$, we need to solve $dS^{(1)} = P_t b^{(1)}$, which, in general, has no solution since $dS^{(1)}$ lies in a $2FL$-dimensional space, whereas $P_t$ spans only a $t$-dimensional space. The best least-squares solution is obtained as

$$b^{(1)} = P_t^T dS^{(1)}. \quad (72)$$

Finally, using the constraints discussed earlier, $b_{l_{\text{min}}} \leq b_l \leq b_{l_{\text{max}}}$, where $1 \leq l \leq t$, we limit these ST shape variations and obtain an acceptable or allowable shape within the ASTSD. By updating $b^{(0)}$ to $b^{(1)}$, we have the new values for all the parameters $s^{(1)}, \theta^{(1)}$, $b^{(1)}$, and $t^{(1)}$.

**Updating the estimate and reiterating.** Similarly, new ST shape estimates can be obtained:

$$\hat{S}^{(i)} = M \left(s^{(i)}, \theta^{(i)}\right) \left[S + P_t b^{(i)}\right] + t^{(i)} \rightarrow \quad \hat{S}^{(i+1)} = M \left(s^{(i+1)}, \theta^{(i+1)}\right) \left[S + P_t b^{(i+1)}\right] + t^{(i+1)} \quad \ (73)$$
for $i = 1, 2, 3, \ldots$. Checking for convergence can be done by examining the changes, i.e., if the new estimate is not much different (according to some predefined threshold), then the search is completed; otherwise we reiterate.

4.3. Results

We present results of locating the spatiotemporal shape of the left ventricle in real echocardiographic image sequences. The training data set consisted of 6 frame sequences, each sequence including 21 frames, and each frame of size $255 \times 254$ pixels (i.e., the size of $\Phi_V = 6 \times 21 \times 255 \times 254$). The number of $(x, y)$ landmark coordinates in each frame was 25 (size of $\Phi_S = 6 \times 21 \times 25 \times 2$). Three ST shape parameters were used to explain 94.2% of the total ST shape variations. The gray-level search was conducted on a profile of length 60 pixels, and the training profile was of length 26 pixels. Figure 24 illustrates how statistical spatiotemporal prior knowledge is used to constrain the proposed segmentation and produce the final left-ventricular segmentation. Figure 25 shows additional segmentation results.

We applied this method to segmenting astrocyte cells in a 3D fluorescence image, where the spatial $z$-axis replaces time. The training data set consisted of 8 volumes (out of 9, leave-one-out validation), and each included 11 image slices, each image being of size $128 \times 128$ pixels (i.e., the size of $\Phi_V = 8 \times 11 \times 128 \times 128$). The number of $(x, y)$ landmark coordinates in each slice was 40 (size of $\Phi_S = 8 \times 11 \times 40 \times 2$). Seven shape parameters were used to explain 99.5% of the total shape variations. The gray-level search was conducted on a profile of length 40 pixels, and the training profile was of length 12 pixels. Figure 26 illustrates example segmentation results.

4.4. Summary

Motivated by the fact that many image analysis applications require robust methods for representing, locating, and analyzing non-rigid time-varying shapes, we presented an extension of a 2D ASM to 2D+time. This method models the gray-level information and the spatiotemporal variations of a time-varying object in a training set. The model was then used for locating similar moving objects in a new image sequence. The segmentation technique was based on deforming a spatiotemporal shape to better fit the image sequence data only in ways consistent with the training set. The proposed deformations were calculated by minimizing an energy function using dynamic programming. The energy function included terms reflecting temporal smoothness and gray-level information constraints.
Figure 24. Left-ventricular segmentation result from two echocardiographic image sequences. Ultrasound frames are shown with the ST shape overlain (a,c) before and (b,d) after projection onto the ASTSD (frames progress from left to right, top to bottom). Reprinted with permission from [63]. Copyright ©2004, Elsevier.
Figure 25. Additional left-ventricular segmentation results from an echocardiographic image sequence (frames progress from left to right, top to bottom). Reprinted with permission from [63]. Copyright ©2004, Elsevier.
Figure 26. Segmenting a 3D astrocyte cell (spatial $z$-axis replaces time): (a) initial shape model and (b) segmentation result overlain in white on a fluorescence 3D image. Reprinted with permission from [63]. Copyright ©2004, Elsevier.

5. REFERENCES


