A Generic Framework for Tracking Using Particle Filter With Dynamic Shape Prior

Yogesh Rathi, Namrata Vaswani, and Allen Tannenbaum

***Abstract*—Tracking deforming objects involves estimating the global motion of the object and its local deformations as functions of time. Tracking algorithms using Kalman filters or particle filters (PFs) have been proposed for tracking such objects, but these have limitations due to the lack of dynamic shape information. In this paper, we propose a novel method based on employing a locally linear embedding in order to incorporate dynamic shape information into the particle filtering framework for tracking highly deformable objects in the presence of noise and clutter. The PF also models image statistics such as mean and variance of the given data which can be useful in obtaining proper separation of object and background.**

***Index Terms*—Dynamic shape prior, geometric active contours, particle filters (PFs), tracking, unscented Kalman filter.**

I. Introduction

In recent years, there has been substantial research in the field of active vision, more specifically, in the segmentation and tracking of deforming objects (see [1]–[4] and the references therein). Many methods based on geometric information such as edges (see, e.g., [1]) track the global (rigid or affine) motion of the object, whereas other techniques which utilize the statistical properties of the image (e.g, mean and variance of the intensities) [5]–[7] can track local shape deformations of the moving objects. The latter methods track by minimizing an image-based energy functional at each frame and do not incorporate motion dynamics of the moving object into their tracking framework. The former methods can only handle affine motion of the moving object and cannot track local shape deformations. This paper extends the work done in [8] and proposes a geometric observer for infinite-dimensional space of curves within the particle filtering framework. It also incorporates dynamic shape priors and image statistics for tracking highly deformable objects in the presence of noise and clutter.

Manuscript received November 23, 2005; revised December 18, 2006. This work was supported in part by the National Science Foundation; in part by the National Institutes of Health (NAC P41 RR-13218 through Brigham and Women's Hospital); in part by AFOSR; in part by ARO; in part by MURI; in part by MRI-HEL; and in part by the Technion—Israel Institute of Technology. This work was done under the auspices of the National Alliance for Medical Image Computing (NAMIC), funded by the National Institutes of Health through the NIH Roadmap for Medical Research, Grant U54 EB005149. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Vicent Caselles.

Y. Rathi and A. Tannenbaum are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: yogesh.rathi@bme.gatech.edu; tannenba@ece.gatech.edu).

N. Vaswani is with the Department of Electrical and Computer Engineering, Iowa State University, Ames, IA 50011 USA (e-mail: namrata@iastate.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2007.894244

A. Past Work

In order to appreciate this methodology, we briefly review some previous related work. The possible parameterizations of planar shapes described as closed contours are of course very important. Various finite-dimensional parameterizations of continuous curves have been proposed, perhaps most prominently the B-spline representation used for a "snake model" as in [2]. Isard and Blake (see [1] and the references therein) use the B-spline representation for contours of objects and propose the CONDENSATION algorithm [1] which treats the affine group parameters as the state vector, learns a prior dynamical model for them, and employs a particle filter (PF)[9] to estimate them from the (possibly) noisy observations. Since this approach only tracks affine parameters, it cannot handle local deformations of the deforming object.

Another approach for representing contours is via the level set method [10], [11] where the contour is represented as the zero level set of a higher dimensional function, e.g., the signed distance function [10], [11]. For segmenting an object, an initial guess of the contour (represented using the level set function) is deformed until it minimizes an image-based energy functional. Tracking is then performed by minimizing this energy functional at each iteration. This, however, does not take into account the motion dynamics of the moving object. Some previous work on tracking using level set methods is given in [6] and [7].

Shape information is quite useful when tracking in clutter, especially if the object to be tracked gets occluded. Hence, a number of methods have been proposed [5], [6] which incorporate a static shape prior into the tracking framework. The approach of these works is based on the idea that the object being tracked does not undergo a deformation (modulo a rigid transformation). Another method for obtaining a shape prior is via principal component analysis (PCA) [12]. In this case, it is assumed that the shape can undergo small variations which can be captured by doing linear PCA. The dynamics of the variations in shape can be learned using an autoregressive model which can then be utilized within the CONDENSATION framework for tracking small deformations in shape [4], [6]. However, linear PCA is quite inadequate in representing the shape variations if the object being tracked undergoes large deformations (as will be explained in detail subsequently).

In [13], the authors propose a method for segmentation with multiple shape priors. A joint energy which depends on the image and a labelling function is minimized which leads to automatic selection of shape priors appropriate for the object.

However, this method cannot be used to provide dynamic shape prior for tracking deforming objects since it would require prior knowledge about all possible shapes of the object. In [5], the authors minimize a shape-based energy (the Mahalanobis distance) in the kernel space to obtain a shape prior. In this case, a set of training shapes is mapped to a high-dimensional space using a nonlinear function and the Mahalanobis distance between the given shape and the training set is minimized to obtain a shape prior. This method is related to kernel PCA [5], [14] and is shown to perform better than linear PCA in a number of cases. Kernel PCA can also be used to provide dynamic shape prior in any tracking framework.

Other related approaches to tracking are given in [2], [3], [15], and [16]. In [15], the authors propose a generic local observer to incorporate prior information about the system dynamics for tracking deformable objects. They impose a constant velocity prior on the group action and a zero velocity prior on the contour. The observed value of the group action and the contour is obtained by a joint minimization of the energy. This is linearly combined with the value predicted by the system dynamics using an observer.

The authors in [8] employ a particle filtering algorithm for geometric active contours to track highly deformable objects. The tracker, however, fails to maintain the shape of the object being tracked in case of occlusion. The present work extends the method proposed in [8] by incorporating dynamic shape priors into the particle filtering framework based on the use of a locally linear embedding (LLE). We further generalize the algorithm by including the image statistics (mean and variance of the object and background) in the state vector and formulate a geometric active contour method which drives the contour not only based on the image information, but also based on the statistics predicted by the PF. This makes the algorithm more robust to leaks and occlusions.

LLE is widely used for dimensionality reduction and for pattern classification; see [17]–[19]. It attempts to discover the nonlinear structure in high-dimensional data by exploiting the local symmetries of linear reconstructions. To the best of our knowledge, this is the first time LLE has been used for shape analysis and tracking. Of course, the literature reviewed above is by no means exhaustive.

The rest of the paper is organized as follows. Section II briefly explains the basic theory underlying PFs and curve evolution. Section III gives the motivation and describes the concepts of LLE and shape similarity measures. Section IV develops the state space model in detail, and Section V describes the experiments conducted to test the proposed method. Some conclusions and further research directions are discussed in Section VI.

II. Preliminaries

This section briefly describes some of the basic ideas for PFs and curve evolution using level sets which we will need in the present work.

A. Particle Filtering With Importance Sampling

Let $S_t \in \mathbf{R}^n$ be a state vector evolving according to the following difference equation: $S_{t+1} = f_t(S_t, n_t)$ where n_t is i.i.d. random noise with known probability distribution function (pdf). At discrete times, observations $Y_t \in \mathbf{R}^p$ become available. These measurements are related to the state vector via the observation equation: $Y_t = h_t(S_t, \nu_t)$ where ν_t is measurement noise with known pdf. It is assumed that the initial state distribution $p(S_0)$, the state transition function denoted by f_t and the observation likelihood given the state, denoted by $p(Y_t|S_t)$, are known. The PF [9], [20] is a sequential Monte Carlo method which produces at each time t, a cloud of N particles, $\{S_t^{(i)}\}_{i=1}^N$, whose empirical measure closely "follows" $p(S_t|Y_{1:t})$, the posterior distribution of the state given past observations. The first application of PF to tracking in computer vision was the CONDENSATION algorithm [21].

The algorithm starts with sampling N times from the initial state distribution $p(S_0)$ in order to approximate it by $p_0^N(S) = (1/N)\sum_{i=1}^N \delta(S_0 - S_0^{(i)})$, and then *implements the Bayes' recursion* at each time step. Assuming that one can sample from the posterior distribution $p(S_t|Y_{1:t})$, an empirical estimate of this distribution is given by: $p_t^N(S_t|Y_{1:t}) = \sum_{i=1}^N \omega_t^{(i)} \delta(S_t - S_t^{(i)})$, where $\omega_t^{(i)}$ is the weight associated with the ith particle. In general, however, it is usually difficult, if not impossible, to sample efficiently from the posterior $p(S_t|Y_{1:t})$. One solution consists of using the *importance sampling* method given in [22] and described briefly here.

Suppose $p(x) \propto q(x)$ is a probability density from which it may be difficult to draw samples, and suppose that $q(x)$ is a density from which it is easy to sample, and has a heavier tail than $p(x)$ (i.e., there exists a bounded region R such that for all points outside R, $q(x) > p(x)$). $q(x)$ is known as the *proposal density* or the *importance density*. Let $x^i \sim q(x), i = 1, \ldots, N$ be samples generated from $q(\cdot)$. Then, an approximation to $p(\cdot)$ is given by $p(x) \approx \sum_{i=1}^N \tilde{\omega}^{(i)} \delta(x - x^i)$, where $\omega^{(i)} \propto p(x^i)/q(x^i)$ and $\tilde{\omega}^{(i)} = \omega^{(i)} / \sum_j \omega^{(j)}$ is the normalized weight of the ith particle. So, if the samples, $S_t^{(i)}$, were drawn from an importance density, $q(S_t|S_{1:t-1}, Y_{1:t})$, and weighted by

$$\omega_t^{(i)} \propto \frac{p\left(S_t^{(i)}|Y_{1:t}\right)}{q\left(S_t^{(i)}|S_{1:t-1}^{(i)}, Y_{1:t}\right)}$$

then $\sum_{i=1}^N \tilde{\omega}_t^{(i)} \delta(S_t - S_t^{(i)})$ approximates $p(S_t|Y_{1:t})$. The choice of the importance density is a critical design issue for implementing a successful PF. As described in [22], the proposal distribution $q(\cdot)$ should be such that particles generated by it, lie in the regions of high observation likelihood. One way of doing this is to use a proposal density which depends on the current observation. This idea has been used in many past works such as the unscented PF [23] where the proposal density is a Gaussian density with a mean that depends on the current observation.

In this paper, the state process is assumed to be Markov, and the observations are conditionally independent given the current state, i.e., $p(Y_t|S_{0:t}) = p(Y_t|S_t)$. This gives the following recursion for the weights [20]:

$$\omega_t^{(i)} \propto \omega_{t-1}^{(i)} \frac{p\left(Y_t|S_t^{(i)}\right) p\left(S_t^{(i)}|S_{t-1}^{(i)}\right)}{q\left(S_t^{(i)}|S_{1:t-1}^{(i)}, Y_{1:t}\right)}. \quad (1)$$

Using this relation and a set of samples $S_t^{(i)}$ sampled from the distribution $q(\cdot)$, the empirical distribution for the posterior is given by $p_t^N(S_t|Y_{1:t}) = \sum_{i=1}^N \tilde{\omega}_t^{(i)} \delta(S_t - S_t^{(i)})$, where $\tilde{\omega}_t^{(i)}$ are normalized weights. Next, resampling is performed so that particles with low weights are eliminated.

B. Curve Evolution

There is a large literature concerning the problem of separating an object from its background; see, e.g., [6], [7], [24], and [25]. Level sets have been used quite successfully for this task. Many region-based active contour models have been inspired by the region competition model of Zhu and Yuille [24]. In this approach, starting from an initial estimate, the curve deforms under the influence of various forces until it fits the object boundaries. The curve evolution equation is obtained as the gradient descent iteration to minimize an "energy" E_{image}. In general, E_{image} may depend on a combination of image-based features and external constraints (smoothness, shape, etc.) (see [26], [27], and the references therein). In [28], the authors have proposed a variational framework for segmenting an object using the first two moments (mean and variance) of image intensities. The main idea behind this formulation is as follows. Let $I(x)$ denote the image, $\Phi : \mathcal{R}^2 \rightarrow \mathcal{R}$ denote the signed distance function that embeds the contour C such that $C = \{\Phi(x) | \Phi(x) = 0\}$ and $H(\Phi)$ denote the Heavy-side function [27]

$$H(\Phi) = \begin{cases} 1, & \Phi \geq 0 \\ 0, & \text{else.} \end{cases}$$

Let us denote by p_{u,σ_u} and p_{v,σ_v}, the probability density functions of the image intensities with mean u, v and standard deviation σ_u, σ_v respectively. Assuming a Gaussian distribution for the intensity inside and outside the object, we would like to maximize the following energy [24], [28]:

$$\begin{aligned} E_{\text{image}} = &- \int_\Omega \left(\log p_{u,\sigma_u}(I(x)) H(\Phi)\right) \\ &+ \left(\log p_{v,\sigma_v}(I(x)) H(-\Phi)\right) dx \\ p_{i,\sigma_i} = &\frac{1}{\sqrt{2\pi}\sigma_i} e^{\frac{-(I(x)-i)^2}{2\sigma_i^2}} \end{aligned} \tag{2}$$

where $i = \{u, v\}$. Alternatively, one could minimize the energy functional [28]

$$\begin{aligned} E_{\text{image}} = &\int_\Omega \left(\log \sigma_v^2 + \frac{(I(x)-v)^2}{\sigma_v^2}\right)(1 - H(\Phi))\, dx \\ &+ \int_\Omega \left(\log \sigma_u^2 + \frac{(I(x)-u)^2}{\sigma_u^2}\right) H(\Phi) dx \\ &+ \nu \int_\Omega \|\nabla H(\Phi)\|\, dx \end{aligned} \tag{3}$$

where we have added a regularizing term that penalizes the length of the contour. The Euler–Lagrange for the above functional is given by the following PDE:

$$\begin{aligned} \frac{\partial \Phi}{\partial \tau} = \delta_\epsilon(\Phi) \Bigg(&\nu \text{div} \frac{\nabla \Phi}{\|\nabla \Phi\|} + \log \frac{\sigma_v^2}{\sigma_u^2} \\ &- \frac{(I(x)-u)^2}{\sigma_u^2} + \frac{(I(x)-v)^2}{\sigma_v^2} \Bigg) \end{aligned} \tag{4}$$

where $\delta_\epsilon(\Phi) = dH/d\Phi$ is the Dirac delta function, ν is a user defined parameter that controls the smoothness of the curve and τ is an artificial time marching parameter.[1] Details on how to evolve a curve using level set methods can be found in [10] and [11].

A different method to minimize the energy (2) using shape gradient is given in [25]. In particular, the regularizing term $\delta_\epsilon(\Phi)$ does not appear in the formulation proposed in [25], resulting in a slightly different curve evolution equation. Note that one could use any type of curve evolution equation (edge or region based) in the algorithm being proposed. We have made this particular choice because it is simple yet powerful in segmenting cluttered images.

III. Motivation for Using LLE

In this section, we compare two shape learning techniques, namely linear PCA and LLE. Of particular interest here is the performance of these methods when the training data contains shapes that have very different geometries. To provide dynamic shape prior for highly deforming objects, any shape learning technique should be able to capture all the different variations in shape that an object undergoes. Linear PCA has been used to provide shape prior in case the variability in shapes is small [12]. However, as will become clear, for large variations in shape, linear PCA is simply inadequate. LLE, on the other hand, provides a good alternative in such scenarios.

In what follows, the shapes (closed curves) are assumed to be embedded in a signed distance function with the zero level set representing the contour. Shape analysis is performed on these embeddings, so that it is easy to incorporate prior shape knowledge in the level-set-based curve evolution equation.

A. Linear PCA

PCA has been widely used in many applications such as dimensionality reduction, shape analysis, data classification etc. In PCA, one computes the linear projections of greatest variance from the top eigenvectors of the data covariance matrix. Its first application to shape analysis in the level set framework was accomplished by embedding a curve C as the zero level set of a signed distance function Φ (see [12]). By doing this, a small set of coefficients can be utilized for a shape prior in various segmentation tasks as shown in [12] and [29]. Linear PCA assumes that the set of permissible shapes form a Gaussian distribution, i.e., all possible shapes can be written as a linear combination of a set of eigen-shapes obtained by doing PCA on the training data set [12], [29].

[1]The artificial time marching parameter τ is different than the time t (which denotes time in the context of the PF) used elsewhere in this paper.

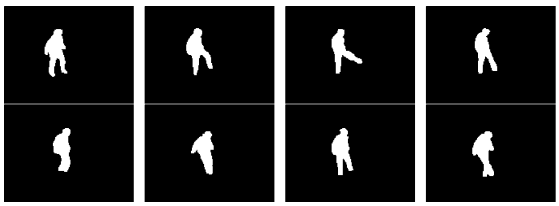

Fig. 1. Few shapes of a man from the training set (note the large deformation in shape).

(a) (b)

Fig. 2. Note that projection in PCA basis does not give a valid shape. (a) Original shape; (b) projection in the PCA basis.

The eigen-shapes can be obtained as follows. Let Φ_i represent the signed distance function corresponding to the surface S_i. All the Φ_is are aligned using a suitable method of registration [30]. The mean surface μ is computed by taking the mean of the signed distance functions, $\mu = (1/n)\sum \Phi_i$. The variance in shape is computed using PCA, i.e., the mean shape μ is subtracted from each Φ_i to create a mean-offset map $\bar{\Phi}_i$. Each such map $\bar{\Phi}_i$ is placed as a column vector in an $N^d \times n$-dimensional matrix M, where $\Phi_i \in \mathbf{R}^{N^d}$. Using singular value decomposition (SVD), the covariance matrix $(1/n)MM^T$ is decomposed as $U\Sigma U^T = (1/n)MM^T$, where U is a matrix whose column vectors represent the set of orthogonal modes of shape variation (eigenshapes) and Σ is a diagonal matrix of corresponding singular values. An estimate of a new shape Φ of the same class of object can be obtained from m principal components using an m-dimensional vector of coefficients, $\alpha = U_m^T(\Phi - \mu)$, where U_m is a matrix consisting of the first m columns of U. Given the coefficients α, an estimate of the shape Φ, namely $\tilde{\Phi}$, can be obtained as: $\tilde{\Phi} = U_m\alpha + \mu$. For more details please refer to [12]. Thus, PCA assumes that the set of training shapes lie on a linear manifold. For small shape deformations, PCA has been used to provide dynamic shape prior [31] for a Bayesian tracker.

We now consider the question of the suitability of PCA for providing reasonable candidate shape priors for highly deforming objects. In particular, we would like to ascertain the ability of PCA to represent previously unseen shapes, given a training set of shapes with large deformation. To do this we consider several shapes of a person (taken as an example of a highly deforming object) as indicated in Fig. 1. Notice the large variation in shape due to movement of the limbs. PCA was performed on 75 such shapes (embedded in a signed distance function). A previously unseen shape was projected into the PCA basis and the coefficients α computed. Given α, the pre-image of the projection $\tilde{\Phi}$ was computed as described before. Fig. 2 shows the original and the pre-image of the projection. As seen, the original and the projected shapes are very different. Thus, in many cases, linear PCA cannot be used to obtain a suitable shape prior if the training set lies on a nonlinear manifold.

B. LLE for Shape Analysis

In [18], the authors proposed an unsupervised LLE algorithm that computes low-dimensional, neighborhood preserving embeddings of high-dimensional data. LLE attempts to discover nonlinear structure in high-dimensional data by exploiting the local symmetries of linear combinations. It has been used in many pattern recognition problems for classification. In this work, we use it in the particle filtering framework for providing dynamic shape prior.

The LLE algorithm [18] is based on certain simple geometric principles. Suppose the data consists of n vectors Φ_i of dimension D^2, sampled from some underlying smooth manifold. Provided there is sufficient data, we expect each data point and its neighbors to lie on or close to a locally linear patch of the manifold. We can characterize the local geometry of these patches by a set of coefficients that reconstruct each data point from its neighbors. In the simplest formulation of LLE, one identifies k nearest neighbors of the data point Φ. Reconstruction error is then measured by the cost function: $E(W) = (\Phi - \sum_{j=1}^{k} \alpha_j \Phi_j)^2$. We seek to find the weights $W = \{\alpha_1, \ldots, \alpha_k\}$ so as to minimize the reconstruction error $E(W)$, subject to the constraint that the weights α_j that lie outside a certain neighborhood are zero and $\sum_j \alpha_j = 1$. With these constraints, the weights for points in the neighborhood of Φ can be obtained as [17]

$$E(W) = \sum_{j=1}^{k} (\Phi - \alpha_j \Phi_j)^2 = \sum_{j=1}^{k}\sum_{m=1}^{k} \alpha_j \alpha_m Q_{jm}$$

$$\Rightarrow \alpha_j = \frac{\sum_{m=1}^{k} R_{jm}}{\sum_{p=1}^{k}\sum_{q=1}^{k} R_{pq}}$$

$$\text{where } Q_{jm} = (\Phi - \Phi_j)^T(\Phi - \Phi_m) \text{ and } R = (Q)^{-1}. \quad (5)$$

In applications where dimensionality reduction is the major objective, one proceeds further and computes a low-dimensional vector corresponding to each Φ_i, preserving the neighborhood structure by keeping the weights α_j constant [17]. This work uses LLE only for obtaining the neighborhood structure in the training set and not for dimensionality reduction. Performing shape analysis using low-dimensional vectors is the subject of future study as was done for linear PCA in [29].

In what follows, we assume that a closed curve C_i is represented as the zero level set of a signed distance function Φ_i. Stacking all the columns of Φ_i one below the other, one can obtain a vector of dimension D^2, if Φ_i is of dimension $D \times D$ (in the rest of the paper, we use Φ interchangeably to represent a vector of dimension D^2 or a matrix of dimension $D \times D$. The appropriate dimension can be inferred from the context).

Let us now see how well an unseen shape is represented using LLE. Fig. 3 shows the original and the projected shape obtained from 2 or more of the nearest neighbors. As is evident, LLE performs quite better than PCA. Of course, this leaves open the problem of how many nearest neighbors to choose. One way to proceed is to compute the distance between the original shape and the projected shape using different numbers of nearest neighbors, denoted by k. Fig. 4 shows the plot of distance as a function of the nearest neighbors for two different shapes. As is

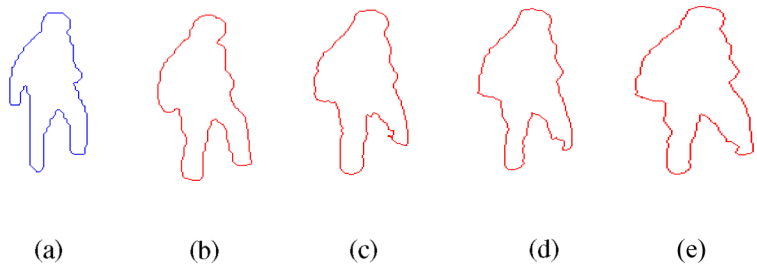

Fig. 3. Shape representation with LLE; k is the number of nearest neighbors used to represent the original shape. (a) Original shape; (b) $k = 2$; (c) $k = 10$; (d)$k = 15$; (e) $k = 20$.

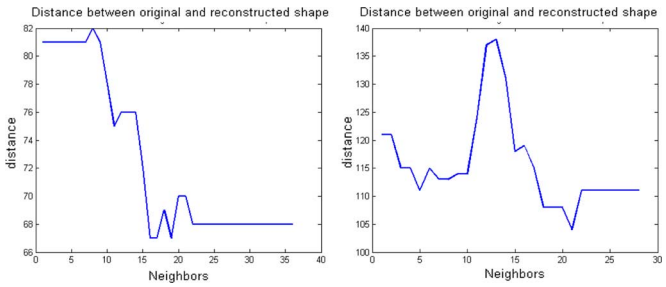

Fig. 4. Graph of the distance between the original shape and the shape obtained using different numbers of nearest neighbors; x axis is the number of nearest neighbors and y axis is the distance between the shapes. Distance was calculated using equation (6). Plots shown are for two different shapes of man from the training data.

clear, the optimal value for k may change for different shapes. Finding the optimal k for each shape can be computationally complex, and will also depend on the training data set. Hence, to reduce the computational complexity, we have used a fixed value for k in all our experiments. This approximation, however, did not seem to effect the overall performance of our tracker.

C. Finding the Nearest Neighbors

In the previous section, we showed how to represent a shape Φ_i by a linear combination of its k neighbors. Here we consider the key issue of how to find the nearest neighbors. One may be tempted to use the Euclidean 2-norm to find distance between shapes, i.e., if $d^2(\Phi_i, \Phi_j)$ is the (squared) distance between Φ_i and Φ_j, then $d^2(\Phi_i, \Phi_j) = \|\Phi_i - \Phi_j\|^2$. However, this norm does not represent the distance between shapes, but only the distance between two vectors. Since we are looking for the nearest neighbors of C_i in the shape space, a similarity measure between shapes is a more appropriate choice. Many measures of similarity have been reported; see , e.g., [5], [6], and [32]. In this paper, we have chosen the following distance measure [33]:

$$d^2(\Phi_i, \Phi_j) = \sum_{p \in Z(\Phi_i)} |\Phi_j(p)| + \sum_{p \in Z(\Phi_j)} |\Phi_i(p)| \qquad (6)$$

where $Z(\Phi_i)$ is the zero level set of Φ_i. The first term in the above equation gives the amount of "work" required to move shape C_i (embedded by Φ_i) to C_j, i.e., the summation can be evaluated by computing the Euclidean distance function of C_j (or, equivalently, absolute value of signed distance function, Φ_j) and summing up values of Φ_j by moving along all points of C_i (or, equivalently, moving along the zero level set $Z(\Phi_i)$ of Φ_i) on a discrete grid. Similarly, the second summation computes the amount of "work" required to move contour C_j to C_i. We chose this particular distance measure because it allows for partial shape matching which is quite useful for occlusion handling. More details about this measure may be found in [33]. The distance measure stated above is not invariant to differences in pose and scale. A rigid registration should be performed between any two shapes before d^2 is computed. There is a large body of literature concerning the problem of rigid registration between two shapes [29], [30]. In this work, we minimize the distance d^2 with respect to the translation and rotation parameters by using gradient descent.

Note that $|\Phi_i|$ is not differentiable at the zero level set; hence, we use a smooth approximation for the absolute value function $|\Phi_i| \approx \sqrt{\Phi_i^2 + \epsilon}$, where ϵ is a small positive constant. Thus, if γ represents one of the rigid parameters (translation, rotation), then the gradient descent equation for each γ is given by

$$\nabla_\gamma d^2(\Phi_i, \Phi_j) = \sum_{p \in Z(\Phi_i)} \frac{\Phi_j \nabla_\gamma \Phi_j}{\sqrt{\Phi_j^2 + \epsilon}} + \sum_{p \in Z(\Phi_j)} \frac{\Phi_i \nabla_\gamma \Phi_i}{\sqrt{\Phi_i^2 + \epsilon}}. \qquad (7)$$

A related registration method, but with a different distance metric is given in [6], [29], and [32]. Details on how to compute $\nabla_\gamma \Phi_i$ can be found in [29]. In the rest of the paper, wherever the distance measure $d^2(\Phi_i, \Phi_j)$ is used, it is assumed that Φ_i, Φ_j have been registered with respect to an appropriate rigid transformation. Our experiments show that, in general, the above gradient descent equation converges quickly requiring only 10–20 iterations. We should note that the development of the remaining algorithm does not depend on a particular choice of the distance measure. Thus, once the distance measure between each Φ_i and the rest of the elements in the training set is known, one can find the nearest neighbors of Φ_i.

IV. State Space Model

This section describes the state space model, the prediction model, and the importance sampling concept used within the particle filtering framework for tracking deformable objects. We will employ the basic theory of particle filtering here as briefly described above, and given in detail in [9] and [23].

Let S_t denote the state vector at time t. The state consists of parameters T that models the rigid (or affine) motion of the object (e.g., $T = [x\ y\ \theta]$ for Euclidean motion), the curve C (embedded as the zero level set of Φ) which models the shape

of the object and the image intensity statistics given by the mean and variance of the object (u, σ_u^2) and background (v, σ_v^2), i.e., $S_t = [T\ \Phi\ u\ v\ \sigma_u^2\ \sigma_v^2]_t$. The observation is the image at time t, i.e., $Y_t = \text{Image}(t)$.

The intensity distribution of the object and the background can provide vital information for any tracking algorithm. For example, in general, the mean intensity of the object being tracked does not change drastically from one frame to the next. Thus, tracking the mean object intensity along with the shape parameter Φ can increase the robustness of the tracker. Hence, we have included the intensity statistics of the object and background in the state vector. Thus, one can obtain a model for the image from the state variables, i.e., T models the location and pose of the object, Φ models the shape of the object, u, σ_u^2 model the intensity distribution inside the object and v, σ_v^2 model the intensity distribution of the background.

Given the description above, we would like to recursively obtain the probability $p(S_t|Y_{1:t})$ as described above in Section II-A. In general, it is quite difficult to sample from the prior distribution $p(S_{t-1}|Y_{1:t-1})$, especially in this particular case, where the state space consists of the curve C which has to be sampled from an infinite-dimensional space of curves. Hence, we perform *importance sampling* as described below. The proposed algorithm is analogous to the unscented particle filter (UPF) [23], wherein an unscented Kalman filter (UKF) is used to obtain the importance distribution. In fact, we propose a novel method which is conceptually similar to the UKF to obtain the importance density function for each particle i. For a better comprehension of the proposed method, we briefly describe the main concepts underlying the UPF as follows [23].

1) At time $t = 0$, initialize the particles (state) by sampling from the prior distribution.
2) *Importance Sampling*: For $t = 1, 2, \ldots$, update the particles by sampling from the importance density function obtained using UKF.
3) Resample so that particles with low weights are eliminated.

The UKF in the importance sampling step above can be briefly described as follows.

1) Deterministically obtain sigma points (particles) using the mean and covariance of the samples from the prior distribution.
2) Time update: Propagate each of the sigma points into the future using the function f_t.
3) Measurement update: Incorporate the new observation to compute the updated mean and covariance

$$S_t = \{\text{prediction of } S_t\} + K\left(\{\text{observation of } Y_t\} - \{\text{prediction of } Y_t\}\right)$$

where K is the Kalman gain which determines how much to trust the system model versus the observation.

In the UPF framework, UKF is used to obtain the proposal density function $q(S_t^{(i)}|S_{t-1}^{(i)}, Y_t) = N(\bar{S}_t^{(i)}, \bar{P}_t^{(i)})$ for each particle i of the PF (see [23] for details). $S_t^{(i)}$ is then obtained by sampling from $q(\cdot)$. We now describe the *time update* step and the *measurement update* step in the proposed framework to obtain the importance density function.

A. Time Update

The state $\hat{S}_t$ at time t is given by: $\hat{S}_t = f_t(S_{t-1}, i_t, n_t)$ where n_t is random noise vector, i_t is any user defined input data (in our case, it is the set of training shapes) and f_t is possibly a nonlinear function [23]. The problem of tracking deforming objects can be separated into two parts [3]:

1) tracking the global rigid motion of the object;
2) tracking local deformations in the shape of the object, which can be defined as any departure from rigidity.

Accordingly, we assume that the parameters that represent rigid motion T_t and the parameters that represent the shape Φ_t are independent. Thus, it is assumed that the shape of an object does not depend on its location in the image, but only on its previous shape and the location of an object in space does not depend on the previous shape. Hence, the prediction step consists of predicting the spatial position of the object using the following:

$$\hat{T}_t = T_{t-1} + n_t^{(T)} \tag{8}$$

where $n_t^{(T)}$ is random Gaussian noise vector with variance σ_T^2. The prediction for shape Φ_t is obtained as follows:

$$\hat{\Phi}_t = p_1\Phi_{t-1}^{(N_1)} + p_2\Phi_{t-1}^{(N_2)} + \ldots + p_k\Phi_{t-1}^{(N_k)} \tag{9}$$

where $p_1, p_2, \ldots p_k$ are user-defined weights such that $\sum_i p_i = 1$ and $\Phi_{t-1}^{(N_i)}$, $i = 1, \ldots, k$ are the k nearest neighbors of Φ_{t-1} obtained as described in Section III-C. Note that we have used the data from the training set to obtain $\hat{\Phi}_t$. This is the reason f_t is a function of i_t (user data) as defined above.

A more generalized formulation of the prediction step above can be obtained by sampling the weights p_i from a known distribution to obtain a set of possible shapes and then choosing the predicted shape from this set, based on certain criteria. A different approach to choosing these weights could be based on their distances from the shape Φ_{t-1}, i.e., the nearest neighbor gets higher weight than the next nearest, and so on. Thus

$$\mathbf{p} = [p_1, \ldots, p_k] = [a_1, \ldots, a_k] + n_a \tag{10}$$

where $a_1 \geq a_2, \ldots, \geq a_k$ and $n_a \sim N(0, \Sigma_a)$.

We believe that one of the main contributions of this paper is the formulation of a scheme that allows one to dynamically predict the shape of the object without learning the sequence in which they occur. Thus, the only knowledge required in this prediction step is a training set of shapes. In particular, one does not need to sample from an infinite-dimensional space of shapes (curves), but only from a set containing the linear combination of k nearest neighbors of Φ_{t-1}. This not only reduces the search space dramatically, but also allows to sample from a finite set of possible shapes.

The system model for the image intensities is assumed to be Gaussian with a certain variance. Furthermore, it is assumed that the image statistics are independent. Thus, the prediction

for object statistics u, σ_u^2 and background statistics v, σ_v^2 can be obtained as follows:[2]

$$\begin{aligned} \hat{u}_t &= u_{t-1} + n_t^{(u)}, \quad \hat{v}_t = v_{t-1} + n_t^{(v)}, \\ \hat{\sigma}_{u,t}^2 &= \sigma_{u,t-1}^2 + n_t^{(su)}, \quad \hat{\sigma}_{v,t}^2 = \sigma_{v,t-1}^2 + n_t^{(sv)} \end{aligned} \tag{11}$$

where $n_t^{(u)}, n_t^{(v)}, n_t^{(su)}, n_t^{(sv)} \sim N(0,1)$.

B. Measurement Update

We now proceed to describe how to obtain samples from the proposal distribution based on the current observation. In particular, we perform importance sampling for each element of the state vector. The entire algorithm can be divided into two steps.

1) Sample from the proposal distribution for the rigid parameters T based on the current image Y_t.
2) Obtain samples for the shape parameter Φ and the image statistics parameters $u, v, \sigma_u^2, \sigma_v^2$.

The first step can be implemented in the following manner: At time t, for each particle i, generate samples as described in the prediction step in (8). Using the image at time $t(Y_t)$, a rigid transformation is applied to each $\hat{\Phi}_t^{(i)}$ (in particular $C_t^{(i)}$) by doing L_r iterations of gradient descent on the image energy E_{image} with respect to the rigid transformation parameters T, i.e., we generate

$$\tilde{\Phi}_t^{(i)} = f_R^{L_r}\left(\hat{\Phi}_t^{(i)}, Y_t\right) \tag{12}$$

where $f_R^{L_r}(\Phi, Y)$ is given by

$$\begin{aligned} &r^j = r^{j-1} - \alpha^j \nabla_r E_{\text{image}}(r^{j-1}, \Phi, Y), \;\; j = 1,2,3,\ldots,L_r \\ &r^0 = r, \;\; T = r^{L_r} \quad \text{and} \quad f_R^{L_r}(\Phi, Y) = T\Phi. \end{aligned} \tag{13}$$

The above gradient descent algorithm can be implemented as given in [29] for each of the translation, scale and rotation parameters.

In the second step, importance sampling is performed for the rest of the state vector by doing a few (L_d) iterations of gradient descent ("curve evolution") on the energy, E, i.e.,

$$\Phi_t^{(i)} = f_{CE}^{L_d}\left(\tilde{\Phi}_t^{(i)}, Y_t\right) \tag{14}$$

where $f_{CE}^{L_d}(\mu, Y)$ is given by

$$\begin{aligned} &\mu^j = \mu^{j-1} - \alpha^j \nabla_\mu E(\mu^{j-1}, Y), \quad j = 1,2,3,\ldots,L_d \\ &\mu^0 = \mu \quad \text{and} \quad f_{CE}^{L_d}(\mu, Y) = \mu^{L_d} \end{aligned} \tag{15}$$

[2]We would like to thank the anonymous reviewer for pointing out that on some special class of images the above model cannot be directly used. Specifically, for images with variances very close to zero, the prediction for $\sigma_{u,t}^2, \sigma_{v,t}^2$ may be negative. For such cases, one can either remove the variances from the state vector or the prediction step can be modified to $\hat{\sigma}_{u,t}^2 = \max(\sigma_{u,t-1}^2 + n_t^{(su)}, \epsilon_u)$, where ϵ_u is a small positive constant.

where

$$E = E_{\text{image}} + \beta_s E_{\text{shape}} + \beta_{pf} E_{pf}.$$

The energy E_{image} is as defined in (3) and E_{shape} is defined by [6]

$$E_{\text{shape}}(\Phi) = \int_\Omega \bar{\Phi}(x)dx \tag{16}$$

where $\bar{\Phi}(x)$ is the contour obtained from a linear combination of the nearest neighbors of Φ, with weights obtained from (5). Note that $\bar{\Phi}$ of course depends on Φ. We, however, assume a piecewise constant model for E_{shape}. Thus, at each iteration of gradient descent, $\bar{\Phi}$ is obtained by finding the nearest neighbors of Φ and taking a linear combination of these.

This idea has been used in a different context by Chan and Vese [27] and Yezzi [34] to segment images based on mean intensities. In the Chan–Vese model, one computes the mean intensities inside and outside the object after every iteration (even though the mean intensities depend on the evolving contour). Similarly, we compute the shape $\bar{\Phi}$ after every iteration. The corresponding curve evolution equation is given by

$$\frac{\partial \Phi}{\partial \tau} = \bar{\Phi}(x)\|\nabla \Phi\|. \tag{17}$$

This PDE tries to drive the current contour embedded by Φ towards that embedded by $\bar{\Phi}$. This ensures that the current contour (embedded in Φ) is close to the set of learnt shapes. This is another instance where LLE is being used in the proposed algorithm. An alternative to obtaining the evolution equation for E_{shape} without the piece-wise constant assumption can be found in [25], where the authors use the shape gradient for computing the appropriate flow. Of course, the resulting flow is more accurate but quite complicated. The piece-wise constant assumption did not seem to affect the performance of the algorithm and worked well in practice keeping the overall gradient descent quite simple to compute.

The PF-based energy E_{pf} is defined as

$$E_{pf} = E_u + E_v + E_{u,\text{var}} + E_{v,\text{var}} \tag{18}$$

where[3]

$$\begin{aligned} E_u &= (u - \hat{u}_t)^2, \quad E_v = (v - \hat{v}_t)^2, \\ E_{u,\text{var}} &= \left(\log \sigma_u^2 - \log \hat{\sigma}_{u,t}^2\right)^2 \\ E_{v,\text{var}} &= \left(\log \sigma_v^2 - \log \hat{\sigma}_{v,t}^2\right)^2 \end{aligned} \tag{19}$$

[3]Note that $\hat{u}_t$, $\hat{v}_t$, etc., are obtained from (11) and do not denote partial derivatives.

The corresponding Euler–Lagrange for these energy functionals are given by

$$\begin{aligned}
-\nabla_C E_u &= \frac{\partial C}{\partial \tau} = 2(u-\hat{u}_t)\frac{(I-u)}{A_u}\mathcal{N} \\
-\nabla_C E_v &= \frac{\partial C}{\partial \tau} = 2(v-\hat{v}_t)\frac{(I-v)}{A_v}\mathcal{N} \\
-\nabla_C E_{u,\text{var}} &= \frac{\partial C}{\partial \tau} \\
&= \frac{2}{A_u}\left(\log\sigma_u^2 - \log\hat{\sigma}_{u,t}^2\right)\left(1-\frac{(I-u)^2}{\sigma_u^2}\right)\mathcal{N} \\
-\nabla_C E_{v,\text{var}} &= \frac{\partial C}{\partial \tau} \\
&= \frac{2}{A_v}\left(\log\sigma_v^2 - \log\hat{\sigma}_{v,t}^2\right)\left(1-\frac{(I-v)^2}{\sigma_v^2}\right)\mathcal{N}
\end{aligned} \tag{20}$$

where $\mathcal{N}$ is the unit inward normal and

$$A_u = \int\int_{R_{\text{in}}} dA, \quad A_v = \int\int_{R_{\text{out}}} dA$$

with R_{in}, R_{out} = region inside and outside C, respectively, and

$$\begin{aligned}
u &= \frac{\int\int_{R_{\text{in}}} I dA}{A_u}, \quad v = \frac{\int\int_{R_{\text{out}}} I dA}{A_v}, \\
\sigma_u^2 &= \frac{\int\int_{R_{\text{in}}} I^2 dA}{A_u} - u^2, \quad \sigma_v^2 = \frac{\int\int_{R_{\text{out}}} I^2 dA}{A_v} - v^2.
\end{aligned}$$

Thus, the contour evolution corresponding to E_{pf} tries to drive the contour towards the region which will satisfy the statistics of the image as predicted by the PF in (11). This ensures that the proposal distribution so obtained depends on the system model as well as the current observation which is another point of departure from the method proposed in [8] where the proposal density heavily depended on the current observation. Note that constraining the object intensity moments as described above helps in avoiding leaks (unwanted departures from the actual shape) along with preventing large and sudden changes in estimation of the object intensity distribution. Incorporation of image statistics as described in (18) and (20) into the PF is another contribution of the present work.

Thus, the overall gradient descent equation in (14) moves the contour towards the joint minimizer of the image-based term E_{image}, the shape-based term E_{shape}, and the PF-based term E_{pf}. The parameters β_s, β_{pf} are chosen based on how much one wants to trust the image (observation) information (E_{image}) versus the system model (E_{shape}, E_{pf}).

Equation (15) may be implemented by summing the PDEs (4), (17), and (20) as follows:

$$\frac{\partial \Phi}{\partial \tau} = V_{\text{image}} + (\beta_s \bar{\Phi} + \beta_{pf} V_{pf})\|\nabla\Phi\|, \tag{21}$$

where V_{image} is the speed term from (4), V_{pf} is the speed term from (20) and $\bar{\Phi}$ is the shape term from (17). We perform only L (L_d or L_r) iterations of gradient descent since we do not want to evolve the curve until it reaches a minimizer of the energy, E. Evolving to the local minimizer is not desirable since the minimizer would be independent of all starting contours in its domain of attraction, and would be highly dependent upon the observation, Y_t. Thus, the state at time t would loose its dependence on the state at time $t-1$, and this may cause loss of track in cases where the observation is bad. Thus, the choice of L is quite important.

Choosing L to be too large (taking the curve very close to the minimizer) can move the particles too close to the current observation, i.e., trust in the observation is high. If L is chosen to be too small, it implies very low trust in the observations obtained and as a result the particles will not be moved to the region of high observation likelihood. Hence, the choice of L depends on how much one trusts the system model versus the obtained measurements. Note that, L will of course also depend on the step-size of the gradient descent algorithm, as well as the type of PDE used in the curve evolution equation. Thus, L iterations of gradient descent performs the same function that the term $K(\{\text{observation of } Y_t\} - \{\text{prediction of } Y_t\})$ performs in the UKF framework, i.e.,

$$\begin{aligned}
S_t &= \{\text{prediction of } S_t\} \\
&\quad + K\left(\{\text{observation of } Y_t\} - \{\text{prediction of } Y_t\}\right) \\
&= \{\text{prediction of } S_t\} \\
&\quad + \{\text{L iterations of gradient descent}\}.
\end{aligned} \tag{22}$$

In UKF, the Kalman gain K is analytically determined by the system model under a Gaussian state distribution assumption. Thus, K is small if trust in the obtained measurement is less, whereas K is large otherwise. In the present case, however, L is a fixed model parameter which determines how much influence does the current observation Y_t have on the predicted state.

The above framework is very similar to the use of UKF in the UPF framework [23] with the assumption that the distribution for each particle is highly peaked around $S_t^{(i)}$ (variance very close to zero). Thus, the sample $S_t^{(i)}$ is obtained as

$$S_t^{(i)} \sim q\left(S_t^{(i)} | S_{t-1}^{(i)}, Y_t\right) = N\left(f_{CE}^{L_d}(\cdot), \epsilon\right), \text{ with } \epsilon \to 0. \tag{23}$$

This formulation is quite advantageous (compared to UKF), since it does not require expensive computation of the covariance matrix[4] for Φ (the shape vector), no computation of sigma points (since the covariance is assumed to be zero) is done, and finally one does not need a separate mechanism to compare the actual measurements with the predicted measurements (see the algorithm for UKF in [23]).

[4]In some cases, one may not be able to compute the covariance matrix easily if the dimension is very large, for example, if the shape vector Φ is of dimension 500×500.

To summarize, a sample from the proposal distribution may be obtained as follows:

$$\begin{aligned}\tilde{\Phi}_t^{(i)} &= T_t\hat{\Phi}_t = f_R^{L_r}\left(\hat{\Phi}_t^{(i)}, Y_t\right), \quad \Phi_t^{(i)} = f_{CE}^{L_d}\left(\tilde{\Phi}_t^{(i)}, Y_t\right)\\ &= \text{level set corresponding to curve } C_t^{(i)}\\ u_t^{(i)} &= \text{mean intensity inside curve } C_t^{(i)}\\ v_t^{(i)} &= \text{mean intensity outside curve } C_t^{(i)}\\ \sigma_{u,t}^{2,(i)} &= \text{variance of intensity inside curve } C_t^{(i)}\\ \sigma_{v,t}^{2,(i)} &= \text{variance of intensity outside curve } C_t^{(i)}. \end{aligned} \tag{24}$$

C. Statistics for Textured Objects

The above formulation can only track regions with homogeneous image intensities, since we only include the mean and variance of the object in the state space. However, one can easily extend this formulation to track complex textured objects by incorporating the response of an appropriate filter (e.g., Gabor) into the state space model, and then trying to drive the contour evolution towards the filter response predicted by the PF, analogous to (20) above. One very simple extension can be obtained as follows: To incorporate information about texture, one can obtain a vector valued image $\hat{I}$ from the given gray scale image I by using the gradient information at each point as follows:

$$\hat{I} = \frac{\left(I_x^2 - I_y^2, 2I_xI_y\right)}{I_x^2 + I_y^2} \tag{25}$$

where I_x and I_y are spatial derivatives in the x and y directions respectively. Consequently, one works with this new vector image $\hat{I}$ with u, v being vector means and σ_u^2, σ_v^2 being the corresponding covariance matrices. This idea was first proposed in [34] to segment textured images. To make the mathematical analysis a bit easier, let us assume that we only include the vector means of the object ($u = [u_1\ u_2]$) and background ($v = [v_1\ v_2]$) in the state vector. To separate the object from background using only the mean intensities, one can directly use the energy functional defined in [34] [instead of E_{image} defined earlier in (3)]. Notice that the two components of $\hat{I} = [I_1\ I_2]$ are independent. Hence, the energy functional E_u can now be modified for the vector case as follows:

$$E_u = (u_1 - \hat{u}_{t,1})^2 + (u_2 - \hat{u}_{t,2})^2.$$

The corresponding Euler–Lagrange can be written as

$$-\nabla_C E_u = \frac{\partial C}{\partial \tau} = \sum_{i=1}^{2} 2(u_i - \hat{u}_{i,t})\frac{(\hat{I}_i - u_i)}{A_u}\mathcal{N}.$$

Similar changes can be made to E_v and the corresponding Euler–Lagrange computed. This formulation can now be used to track textured objects. One can also use a structure tensor, as has been proposed in many works, to find the image $\hat{I}$ and segment the resulting image [35]. Thus, the proposed method can easily be extended to track objects with nonhomogeneous intensity distribution. In this work, however, we restricted our experiments to nontextured images.

D. Setting the Importance Weights

As described before, the importance weights are given by [20]

$$\omega_t^{(i)} \propto \omega_{t-1}^{(i)} \frac{p\left(Y_t|S_t^{(i)}\right)p\left(S_t^{(i)}|S_{t-1}^{(i)}\right)}{q\left(S_t^{(i)}|S_{t-1}^{(i)}, Y_t\right)}. \tag{26}$$

The likelihood probability, i.e., the probability of observing the image given the state (object shape, position, and intensity distribution) $p(Y_t|S_t)$ is defined as

$$p(Y_t|S_t) \propto e^{\frac{-E(S_t,Y_t)}{\sigma_{\text{tot}}^2}}. \tag{27}$$

The prior density is given by

$$\begin{aligned}p(S_t|S_{t-1}) = p(T_t|T_{t-1})p(\Phi_t|\Phi_{t-1})p(u_t|u_{t-1})p(v_t|v_{t-1})\\ \times p\left(\sigma_{u,t}^2|\sigma_{u,t-1}^2\right)p\left(\sigma_{v,t}^2|\sigma_{v,t-1}^2\right)\end{aligned}$$

with

$$\begin{aligned}p(T_t|T_{t-1}) &\propto e^{\frac{-|T_t - T_{t-1}|}{\sigma_T^2}}\\ p(\Phi_t|\Phi_{t-1}) &\propto e^{\frac{-d^2(\Phi_t,\Phi_{t-1})}{\sigma_\phi^2}} + e^{\frac{-d^2(\Phi_t,\breve{\Phi}_{t-1})}{\sigma_\phi^2}}\\ p(b_t|b_{t-1}) &\propto e^{\frac{-|b_t - b_{t-1}|}{\sigma_b^2}}, \text{ where, } b = u, v, \sigma_u^2, \sigma_v^2\end{aligned} \tag{28}$$

where d^2 is the (squared) distance measure defined above in (6), and $\breve{\Phi}_{t-1}$ is the maximum *a posteriori* (MAP) estimate of the shape at time $t-1$. We should note that using the MAP shape information available from time $t-1$ is quite essential since it adds more weight to particles which are closer to the previous best estimate than particles that are far away. This is quite useful in case of occlusion wherein particles which look like the previous best shape are given higher probability, despite the occlusion. The model parameter σ_ϕ^2 is chosen based on the size of the variation in the shape of an object from one frame to the next. So one should choose a smaller value for σ_ϕ^2 if the variation in shape is small and larger otherwise.

From (23), and assuming a very small but nonzero variance, one can write $q(S_t^{(i)}|S_{t-1}^{(i)}, Y_t) = c$, where c is a constant. In addition, we assume that c (in particular, the variance) is the same for all particles $i = 1, \ldots, N$. Thus, using (23), (27), and (28), one can compute the importance weights

$$\begin{aligned}\omega_t^{(i)} \propto \omega_{t-1}^{(i)} e^{\frac{-E\left(Y_t, S_t^{(i)}\right)}{\sigma_{\text{tot}}^2}} & p\left(T_t^{(i)}|T_{t-1}^{(i)}\right)p\left(\Phi_t^{(i)}|\Phi_{t-1}^{(i)}\right)\\ & \times p\left(u_t^{(i)}|u_{t-1}^{(i)}\right)p\left(v_t^{(i)}|v_{t-1}^{(i)}\right)\\ & \times p\left(\sigma_{u,t}^{2,(i)}|\sigma_{u,t-1}^{2,(i)}\right)p\left(\sigma_{v,t}^{2,(i)}|\sigma_{v,t-1}^{2,(i)}\right).\end{aligned}$$

E. Improvement Using Importance Sampling

Fig. 5 shows the histogram of the likelihood probability of the particles with and without using the importance density. As can be seen, more particles are moved to the region of high likelihood if the importance distribution $q(\cdot)$ is used.

V. Particle Filtering Algorithm

Based on the description above, the complete particle filtering algorithm can be summarized as follows.

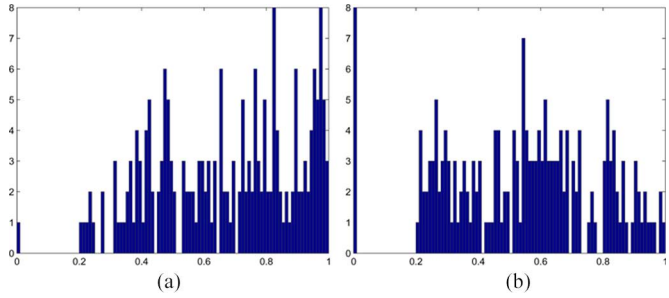

Fig. 5. Likelihood probability distribution (a) with and (b) without using importance density $q(.)$ for frame 2 of octopus sequence (200 particles).

For each particle $i = 1, 2, \ldots, N$ perform the following.

1) **Importance Sampling**:
   a) Obtain model prediction for $\hat{S}_t^{(i)}$ using (8), (9), and (11) as

$$\begin{aligned}\hat{T}_t^{(i)} &= T_{t-1}^{(i)} + n_t^T \\ \hat{\Phi}_t^{(i)} &= p_1 \Phi_{t-1}^{(i),(N_1)} + p_2 \Phi_{t-1}^{(i),(N_2)} \ldots + p_k \Phi_{t-1}^{(i),(N_k)} \\ \hat{u}_t^{(i)} &= u_{t-1}^{(i)} + n_t^{(u)},\ \hat{v}_t^{(i)} = v_{t-1}^{(i)} + n_t^{(v)} \\ \hat{\sigma}_{u,t}^{2,(i)} &= \sigma_{u,t-1}^{2,(i)} + n_t^{(su)},\ \hat{\sigma}_{v,t}^{2,(i)} = \sigma_{v,t-1}^{2,(i)} + n_t^{(sv)}.\end{aligned}$$

   b) Perform L (L_r and L_d) steps of curve evolution on each $\hat{\Phi}_t^{(i)}$

$$\Phi_t^{(i)} = f_{CE}^{L_d}\left(\tilde{\Phi}_t^{(i)}, Y_t\right),\ \tilde{\Phi}_t^{(i)} = T_t \hat{\Phi}_t = f_R^{L_r}\left(\hat{\Phi}_t^{(i)}, Y_t\right).$$

2) **Weighting and Resampling**:
   a) Calculate weights and normalize

$$\omega_t^{(i)} = \frac{e^{\frac{-E\left(Y_t, S_t^{(i)}\right)}{\sigma_{\text{tot}}^2}} p\left(S_t^{(i)} | S_{t-1}^{(i)}\right)}{c}$$

$$\tilde{\omega}_t^{(i)} = \frac{\omega_t^{(i)}}{\sum_{j=1}^N \omega_t^{(j)}}$$

   where $p(S_t^{(i)}|S_{t-1}^{(i)})$ is as defined in (28).
   b) Resample to generate N particles $\{S_t^{(i)}\}_{i=1}^N$ distributed according to $p(S_t|Y_{1:t})$.
3) Go back to the importance sampling step for $t+1$.

The resampling step improves sampling efficiency by eliminating particles with very low weights.

VI. Experiments

The proposed algorithm was tested on four different sequences, and the corresponding results are presented in this section. We certainly do not claim that the method proposed in this paper is the best one for every image sequence on which it was tested, but it did give reasonable results with a small number of particles on all of the image sequences. We should add that to the best of our knowledge this is the first time dynamic shape prior in a level set framework has been used in conjunction with the PF [9] for tracking such deforming objects. Also note that the framework is quite general in that it allows to track multiple objects with different intensity statistics. It can also be used to track color image sequences with the statistics corresponding to each color channel included in the state vector. In this work, however, we restricted our experiments to gray level images.

A. Choice of Model Parameters

Parameters chosen determine the system model for each of the tracking sequence. Hence, the choice of these parameters is indeed important. We have chosen the following set of parameters in our experiments for all the test sequences, i.e., the same set of parameters were used in obtaining all the results.

1) Choosing k, the number of nearest neighbors: k will depend on the number of similar shapes available in the training set [18]. If the training set contains many shapes with high shape similarity measure (6), then k can be higher. In our experiments, the training set was obtained by hand-segmenting images from the test sequence. Choosing $k = 2$ gave acceptable results.
2) Choosing σ_d^2: A classical choice [5] is $\sigma_d^2 = c(1/N)\sum_{i=1}^N \min_{j \neq i} d^2(\Phi_i, \Phi_j)$. For all the test sequences, $c = 1/20$ was used. A good choice for σ_ϕ^2 in (28) was $\sigma_\phi^2 = 10\sigma_d^2$.
3) The algorithm presented in this paper is quite general. However, even if we use a particular case of this general method, the results are encouraging. In particular, we have used the following:
   a) $S_t = [T_t\ \Phi_t\ \sigma_{u,t}^2]$;
   b) from Section IV-D and (a) above, we get $p(S_t|S_{t-1}) = p(\Phi_t|\Phi_{t-1})p(T_t|T_{t-1})p(\sigma_{u,t}^2|\sigma_{u,t-1}^2)$;
   c) accordingly, $E_{pf} = E_{u,u\text{var}}$ was used and only the corresponding gradient descent term (20) was incorporated in the curve evolution equation.
4) σ_T^2 models the motion dynamics of the object being tracked. In all the test sequences, since the spatial motion of the object was not large, we used $\sigma_T^2 = 1000$. Also, only translational motion was assumed, i.e., $T = [x\ y]$.
5) The conditional distribution for $\sigma_{u,t}^2$ was assumed to be $p(\sigma_{u,t}^2|\sigma_{u,t-1}^2) = e^{(-|\sigma_{u,t}^2 - \sigma_{u,t-1}^2|)/10}$ for all test sequences.
6) We assumed $[p_1\ p_2] = [0.6\ 0.4] + n_a$, with the other weights assumed to be zero (see (10)). Our experiments show that the tracking results are not sensitive to this particular choice of the parameters. Of course, p_1, p_2 should be normalized so that $\sum_i p_i = 1$.
7) σ_{tot}^2 depends on the image statistics and will be different for different image sequences. In order avoid choosing a different parameter for each image sequence, one can normalize E and then use a fixed value for σ_{tot}^2. In our experiments we have used $\sigma_{\text{tot}}^2 = 1$.
8) $L_r = 3$ and $L_d = 14$ were used in tracking all the test sequences.

B. Numerical Issues

The number of particles required for a successful implementation of any particle filtering algorithm grows exponentially with the dimension of the state vector. For example, the CONDENSATION algorithm with a 3-D state vector requires

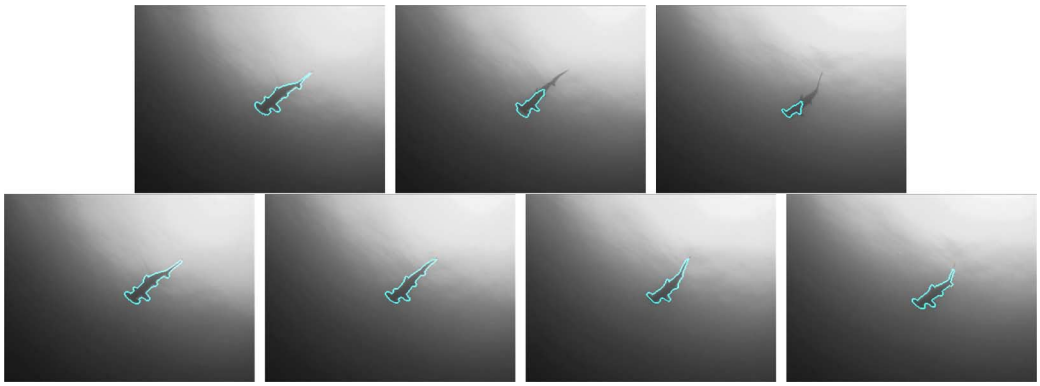

Fig. 6. Shark sequence I. First row: tracking results without shape information. Second row: Results using the proposed algorithm.

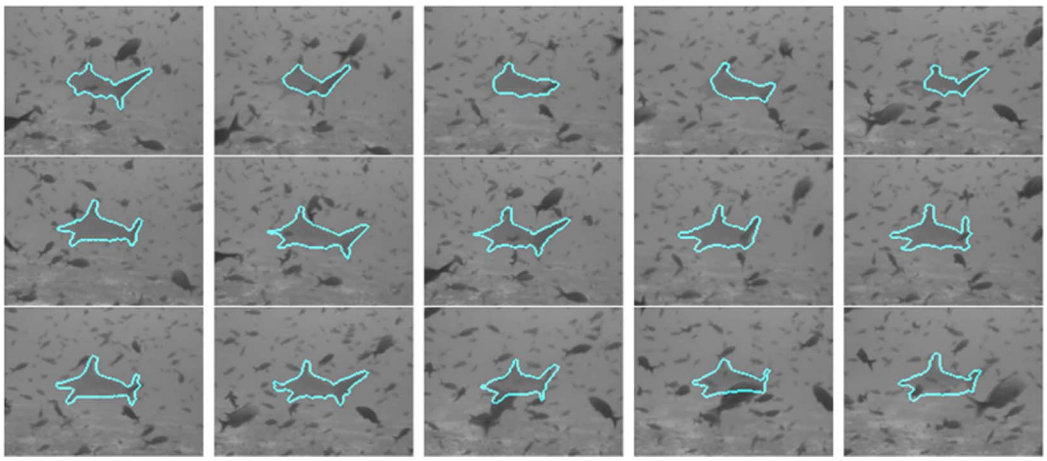

Fig. 7. Shark sequence II. First row shows tracking results with no shape information. Next two rows shows results using the proposed algorithm.

about 2000–5000 particles. In this work, however, the proposed formulation allows for successful tracking with just about 50–100 particles. This is due to the formulation of the importance density function as a gradient descent step described previously in Section IV. This step makes the proposed algorithm practically implementable. Note that the state vector for all the sequences was 4-D in addition to the very high-dimensional representation of the contour (typically 200–300 on a discrete grid). Of course, we do not claim that the method in this work can be implemented in real-time (10–20 Hz), but that it is feasible to implement such a high-dimensional state-based algorithm in reasonable time. Specifically, it took about 30 s to process one frame with un-optimized Matlab code on a 3-GHz Windows machine.

We should note that the most time consuming part of the proposed method is the gradient descent step and the rigid registration step (for providing shape information). Recently, [36] proposed a real-time level-set-based algorithm for tracking. One could use such a technique for speeding up the gradient descent step. We believe that this would significantly improve the speed of the algorithm.

C. Shark Sequence I

In this sequence, there is significant change in the shape of the shark as it moves. The intensity distribution also varies quite a bit from the head to tail. Results of tracking the shark without shape information using the algorithm given in [8] is shown in Fig. 6. Trying to track the sequence with (4) alone is also not encouraging. Note that, due to large shape variations, PCA cannot be used to provide shape information. Fig. 6 shows tracking results using the proposed algorithm with 42 particles. 8% of the images were hand-segmented to obtain the training set.

D. Shark Sequence II

This sequence has very low contrast (object boundaries in some images are barely visible even to human observers) with a shark moving amid a lot of other fish (clutter) which partially occlude it simultaneously in many places. This results in a dramatic change in the image statistics of the shark if a fish from the background occludes the shark. The training set was obtained by hand segmenting 10% of the images from the image sequence. This is because the shape variation is large from one frame to the next. Tracking results without shape prior information but using the intensity moments within the state vector is shown in Fig. 7. As can be seen, even though the algorithm tracks the shark, it is unable to maintain the shape. Results using the proposed algorithm (with shape prior) are shown in Fig. 7. This sequence demonstrates the robustness of the proposed algorithm in the presence of noise and clutter. The number of particles used for this sequence was 56.

E. Octopus Sequence

As seen in Fig. 8, the shape of the octopus undergoes large changes as it moves in a cluttered environment. It gets occluded

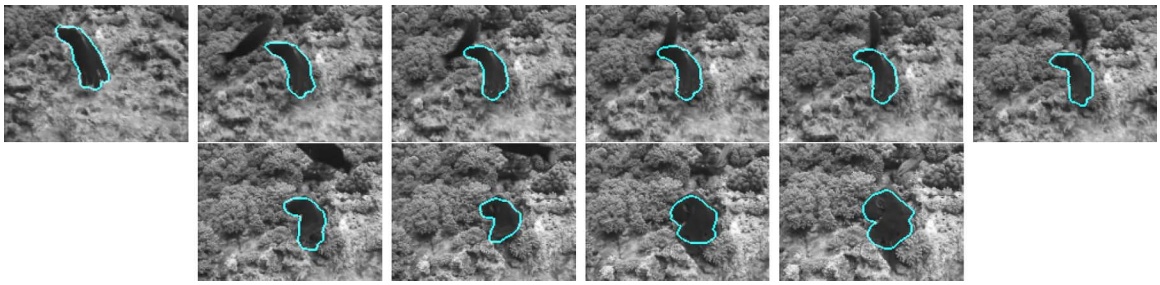

Fig. 8. Octopus sequence: Results using the proposed algorithm. Notice that a fish with the same mean intensity occludes the octopus.

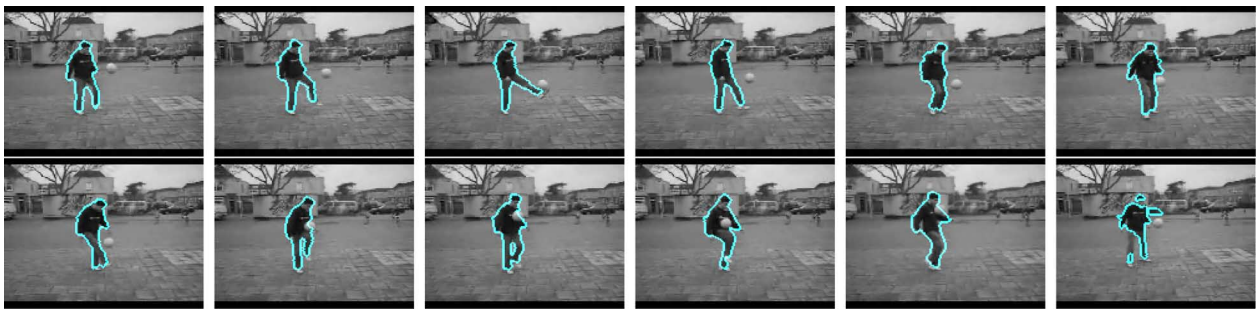

Fig. 9. Results of tracking using the proposed method. Top row shows shape deformation after every 3–4 frames. Last image at the bottom right is the segmentation using equation (4).

for several frames by a fish having the same mean intensity (see the top row). Tracking this sequence using (4) or any other method without shape information may result in the curve leaking to encompass the fish. Fig. 8 shows tracking results using the proposed method. As seen, the shape of the octopus is maintained despite the occlusion. For tracking this sequence we used 40 particles, and the training set included 9% of the possible shapes.

F. Soccer Sequence

This sequence tracks a man playing soccer. There is large deformation in the shape due to movement of the limbs (hands and legs) as the person tosses the ball around. The deformation is also great from one frame to next when the legs occlude each other and separate out. There is clutter in the background which would cause leaks if geometric active contours or the particle filtering algorithm as given in [8] were used to track this sequence. Results of tracking using the proposed method are shown in Fig. 9. Here, we employed 70 particles, and 15% of the possible shapes were included in the training set.

VII. Conclusions and Limitations

In this paper, we have presented an approach which incorporates dynamic shape prior information and image statistics into a particle filtering algorithm for tracking highly deformable objects in presence of noise and clutter. The shape prior information is obtained using LLE for shapes. No motion or shape dynamics are required to be known for tracking complex sequences, i.e., no learning is required. The only information needed is a set of shapes that can appropriately represent the various deformations of the object being tracked. The proposed algorithm can be further generalized to include the intensity distribution of the object and background as part of the state vector (to track objects with texture) instead of mean and variance.

Nevertheless, the current algorithm has certain limitations. First, it is computationally very expensive, as each particle has to be evolved for many iterations. Second, the training set should contain sufficient number of possible shapes of the object being tracked so that LLE can be used, i.e., each shape should have at least two neighbors which can be used to provide shape prior. Finally, it is quite difficult to know the size of the training set to use to obtain reasonable tracking results.

Future work involves obtaining samples for shapes by traveling along the geodesic path from one shape to the other in the shape space as given in [37]–[39]. This could further reduce the number of elements required in the training set and samples could be generated online during the tracking process.

Acknowledgment

The authors would like to thank S. Dambreville for fruitful discussions on various topics involving tracking.

References

[1] A. Blake and M. Isard, Eds., *Active Contours*. New York: Springer, 1998.
[2] D. Terzopoulos and R. Szeliski, "Tracking with Kalman snakes," in *Active Vision*. Cambridge, MA: MIT Press, 1992, pp. 3–20.
[3] A. Yezzi and S. Soatto, "Deformotion: Deforming motion, shape average and the joint registration and approximation of structures in images," *Int. J. Comput. Vis.*, vol. 53, no. 2, pp. 153–167, 2003.
[4] S. Dambreville, Y. Rathi, and A. Tannenbaum, "Tracking deformable objects with unscented kalman filtering and geometric active contours," presented at the American Control Conf., 2006.
[5] D. Cremers, T. Kohlberger, and C. Schnorr, "Nonlinear shape statistics in mumford-shah based segmentation," in *Proc. 7th ECCV*, 2002, vol. 2351, pp. 93–108.
[6] T. Zhang and D. Freedman, "Tracking objects using density matching and shape priors," in *Proc. 9th IEEE Int. Conf. Computer Vision*, 2003, pp. 1950–1954.

[7] N. Paragios and R. Deriche, "Geodesic active contorus and level sets for the detection and tracking of moving objects," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 3, pp. 266–280, Mar. 2000.
[8] Y. Rathi, N. Vaswani, A. Tannenbaum, and A. Yezzi, "Particle filtering for geometric active contours with application to tracking moving and deforming objects," presented at the IEEE Conf. Computer Vision and Pattern Recognition, 2005.
[9] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/nongaussian bayesian state estimation," *Proc. Inst. Elect. Eng. Radar and Signal Processing*, vol. 140, no. 2, pp. 107–113, 1993.
[10] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*. New York: Springer Verlag, 2003.
[11] J. A. Sethian, *Level Set Methods and Fast Marching Methods*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 1999.
[12] M. Leventon, W. E. L. Grimson, and O. Faugeras, "Statistical shape influence in geodesic active contours," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2000, pp. 1316–1324.
[13] D. Cremers, N. Sochen, and C. Schnorr, "A multiphase dynamic labeling model for variational recognition-driven image segmentation," *Int. J. Comput. Vis.*, 2005.
[14] Y. Rathi, S. Dambreville, and A. Tannenbaum, "Statistical shape analysis using kernel PCA," presented at the SPIE Electronic Imaging, .
[15] J. Jackson, A. Yezzi, and S. Soatto, "Tracking deformable moving objects under severe occlusions," presented at the IEEE Conf. Decision and Control, 2004.
[16] N. Peterfreund, "Robust tracking of position and velocity with Kalman snakes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 6, pp. 564–569, Jun. 1999.
[17] D. Ridder and R. Duin, "Locally linear embedding for classification," Tech. Rep. PH-2002-01, Pattern Recognition Group, Delft Univ. Technol., Delft, The Netherlands, 2002.
[18] S. Roweis and L. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.
[19] J. Costa and A. Hero, "Geodesic entropy graphs for dimension and entropy estimation in manifold learning," *IEEE Tran. Signal Process.*, vol. 52, no. 8, pp. 2210–2221, Aug., 2004.
[20] A. Doucet, N. deFreitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*. New York: Springer, 2001.
[21] M. Isard and A. Blake, "Condensation—Conditional density propagation for visual tracking," *Int. J. Comput. Vis.*, vol. 29, no. 1, pp. 5–28, 1998.
[22] A. Doucet, "On sequential Monte Carlo sampling methods for Bayesian filtering," Tech. Rep. CUED/F-INFENG/TR. 310, Dept. Eng., Cambridge Univ., Cambridge, U.K., 1998.
[23] R. van der Merwe, N. de Freitas, A. Doucet, and E. Wan, "The unscented particle filter," *Adv. Neural Inf. Process. Syst.*, vol. 13, Nov. 2001.
[24] S. Zhu and A. Yuille, "Region competition: Unifying snakes, region growing and bayes/mdl for multiband image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 9, pp. 804–906, Sep. 1996.
[25] G. Aubert, M. Barlaud, O. Faugeras, and S. Jehan-Besson, "Image segmentation using active contours: calculus of variations or shape gradients," Tech. Rep., 2002.
[26] D. Mumford and J. Shah, "Optimal approximation by piecewise smooth functions and associated variational problems," *Commun. Pure Appl. Math.*, vol. 42, pp. 577–685, 1989.
[27] T. Chan and L. Vese, "Active contours without edges," *IEEE Trans. Image Process.*, vol. 10, no. 2, pp. 266–277, Feb. 2001.
[28] R. Deriche and M. Rousson, "A variational framework for active and adaptative segmentation of vector valued images," in *Proc. Motion Workshop*, 2002, p. 56.
[29] A. Tsai, A. Yezzi, W. Wells, C. Tempany, D. Tucker, A. Fan, E. Grimson, and A. Willsky, "A shape-based approach to the sementation of medical imagery using level sets," *IEEE Trans. Med. Imag.*, vol. 22, no. 2, pp. 137–153, Feb. 2003.
[30] T. Cootes, C. Taylor, D. Cooper, and J. Graham, "Active shape models, their training and application," in *Comput. Vis. Image Understand.*, 1995, vol. 61, pp. 38–59.
[31] D. Cremers, "Dynamical statistical shape priors for level set based tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 8, pp. 1262–1273, Aug. 2006.
[32] D. Cremers and S. Soatto, "A pseudo-distance for shape priors in level set segmentation," presented at the IEEE Workshop on Variational, Geometric and Level Set Methods in Computer Vision, 2003.
[33] T. Funkhouser, M. Kazhdan, P. Shilane, P. Min, W. Kiefer, A. Tal, S. Rusinkiewicz, and D. Dobkin, "Modeling by example," presented at the ACM SIGGRAPH, 2004.
[34] A. Yezzi, A Tsai, and A. Willsky, "A statistical approach to curve evolution for image segmentation," Tech. Rep., Massachusetts Inst. Technol., Cambridge, 1999.
[35] Z. Wang and B. C. Vemuri, "An affine invariant tensor dissimilarity measure and its applications to tensor-valued image segmentation," in *Proc. CVPR*, 2004, pp. 228–233.
[36] Y. Shi and W. C. Karl, "Real-time tracking using level sets," presented at the CVPR, 2005.
[37] A. Srivastava, S. Joshi, W. Mio, and X. Liu, "Statistical shape analysis: Clustering, learning and testing," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 4, pp. 590–602, Apr. 2005.
[38] A. Yezzi and A. C. G. Mennucci, Metrics in the space of curves [Online]. Available: http://www.arxiv.org/abs/math.DG/0412454 2004
[39] P. W. Michor and D. Mumford, Riemannian geomtries of space of plane curves [Online]. Available: http://www.front.math.ucdavis.edu/math.DG/0312384

Yogesh Rathi received the B.S. degree in electrical engineering and the M.S. degree in mathematics from the Birla Institute of Technology and Science, Pilani, India, in 1997, and the Ph.D. degree in electrical engineering from the Georgia Institute of Technology, Atlanta, in 2006. His Ph.D. dissertation titled "Filtering for Closed Curves" was on tracking highly deformable objects in the presence of noise and clutter.

He is presently a Research Scientist in the Minerva Research Group, Georgia Institute of Technology. His research interests include image processing, medical imaging, computer vision, control, and machine learning.

Namrata Vaswani received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Delhi, in August 1999, and the Ph.D. degree in electrical and computer engineering from the University of Maryland, College Park, in August 2004. Her Ph.D. thesis was on change detection in stochastic shape dynamical models and applications to activity modeling and abnormal activity detection.

She was a Postdoctoral Fellow in the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, in 2004 and 2005. She is currently an Assistant Professor with the Department of Electrical and Computer Engineering, Iowa State University, Ames. Her research interests are in detection and estimation problems in statistical signal and video processing, computer vision, and biomedical image analysis. In particular, she is interested in particle filtering theory and applications in tracking and change detection, as well as shape analysis and filtering.

Allen Tannenbaum was born in New York City in 1953. He received the Ph.D. degree in mathematics from Harvard University, Cambridge, MA, in 1976.

He has held faculty positions at the Weizmann Institute of Science; McGill University, Montréal, QC, Canada; ETH, Zurich, Switzerland; the Technion—Israel Institute of Technology, Haifa; the Ben-Gurion University of the Negev, Israel; and the University of Minnesota, Minneapolis. He is presently the Julian Hightower Professor of Electrical and Biomedical Engineering at The Georgia Institute of Technology, Atlanta/Emory. He has done research in image processing, medical imaging, computer vision, robust control, systems theory, robotics, semiconductor process control, operator theory, functional analysis, cryptography, algebraic geometry, and invariant theory.