A Cascading Framework of Contour Motion and Deformation Estimation for Non-Rigid Object Tracking

Jie Shao, Fatih Porikli, Rama Chellappa

TR2006-044 October 2006

Abstract

This paper mainly focuses on application for non-rigid contour tracking in heavily cluttered background scenes. Based on the properties of non-rigid contour movements, a cascading framework for estimating contour motion and deformation is proposed. We solve the non-rigid contour tracking problem by decomposing it into three sub problems: motion estimation, deformation estimation, and shape regulations. First, we employ a particle filter to estimate the global motion parameters of the affine transform between successive frames. Then we generate a deformation probabilistic map to deform the contour. To improve the robustness, multiple cues are used for deformation probability estimation. Finally, we use a shape prior model to constrain the deformed contour. This enables us to retrieve the occluded parts of the contours and accurately track them while allowing shape changes specific to the given object types. Our experiments show that the proposed algorithm significantly improves the tracker performance.

International Conference on Image Processing (ICIP 2006)

A Cascading Framework of Contour Motion and Deformation Estimation for Non-Rigid Object Tracking

Jie Shao[1], Fatih Porikli[2] and Rama Chellappa[1]
[1]Center for Automation Research and Department of ECE
University of Maryland, College Park, MD 20742
{shaojie,rama}@cfar.umd.edu
[2] Mitsubishi Electric Research Laboratories, Cambridge, MA 02139
fatih@merl.com

Abstract

This paper mainly focuses on applications for non-rigid contour tracking in heavily cluttered background scenes. Based on the properties of non-rigid contour movements, a cascading framework for estimating contour motion and deformation is proposed. We solve the non-rigid contour tracking problem by decomposing it into three sub problems: motion estimation, deformation estimation, and shape regulation. First, we employ a particle filter to estimate the global motion parameters of the affine transform between successive frames. Then we generate a deformation probabilistic map to deform the contour. To improve the robustness, multiple cues are used for deformation probability estimation. Finally, we use a shape prior model to constrain the deformed contour. This enables us to retrieve the occluded parts of the contours and accurately track them while allowing shape changes specific to the given object types. Our experiments show that the proposed algorithm significantly improves the tracker performance.

Index Terms

Motion, deformation, particle filter, shape subspace, non-rigid tracking.

Partially supported by HSARPA through a subcontract from Honeywell corporation and a summer internship at MERL.

I. Introduction

Visual tracking is an essential component of many applications from intelligent robotics to video surveillance. Basically, there are three groups of tracking methods: correspondence-based, transformation-based, and contour-based. The first group of methods is based on establishing correspondences between feature points. The second group performs tracking by estimating object motion, in which the objects are usually assumed to be made of planar shapes such as ellipses and rectangles. The last group achieves tracking by finding the object contour in successive frames. It applies to cases when not only the location but also the deformation of a target are desired during tracking. Some exemplary applications include surveillance tracking for recognition purpose and echocardiography tracking for computer aided diagnosis (CAD). The tracking approach proposed in this paper belongs to the group of contour-based tracking methods.

A. Related Work

To appreciate the methodology, we briefly introduce some related work in contour tracking. A number of contour based tracking methods have been proposed in literature. As a milestone in contour-based tracking research, CONDENSATION, a parameterized B-spline contour tracking algorithm, was proposed by Isard and Blake [10]. It uses a particle filter as the basic framework to track the global motion and the deformation. The algorithm yields robust results when applied to rigid objects. However, it has no explicit criterion for extracting the exact boundary of a non-rigid object in its observation model during tracking. In [16], Li *et al.* presented a particle filter for non-rigid object contour tracking. But the algorithm lacks of an appropriate model for discriminating a real boundary from all the detected edge points. Snakes [14, 15], also known as dynamic contours, is another common approach that evolves an object boundary such that a weighted sum of external and internal energy terms is minimized. However, the methods are restricted to a relatively small range of scenarios due to the fact that they rely on intensities inside objects to be fairly uniform. Besides, their computational complexity makes them less suitable for real-time applications. The level set approach is also a powerful method that deals with topological changes of the moving level set function by using partial differential equations (PDE) that describe the object motion, boundary and region-based information [19, 25, 26, 30]. But they also prefer uniform intensity distributions inside objects.

Some other approaches closely related to non-rigid contour tracking include [11, 24, 29]. The concepts of motion and deformation were defined in [29]. ***Motion*** is parameterized by a finite dimensional group action, and ***deformation*** is the total deformation of the object contour (infinite dimensional group) modulo the finite dimensional motion group. By incorporating the prior information of the system dynamics in the deformation framework, [11] proposed a nonlinear dynamical model for tracking a slowly deforming and moving contour, with the contour represented implicitly as the infinite-dimensional locus of zeros of a given function. The algorithm suffers from the expensive computation due to the joint minimization for both the group action and the deformation. The work in [24] extended the ideas in [11]. It uses a particle filter to estimate the conditional probability distribution of the motion and the contour, which formalizes the incorporation of a prior system model along with an observation model. However, the algorithm also has limitations. It only counts on appearance cue in the observation model and lacks of ability to handle the moving background cases.

Our approach shares some similarities with [24]. We claim that tracking non-rigid objects can be accomplished by estimating both translational (finite dimensional, ***Motion***) and non-translational movements (infinite dimensional, ***deformation***) of objects. However, instead

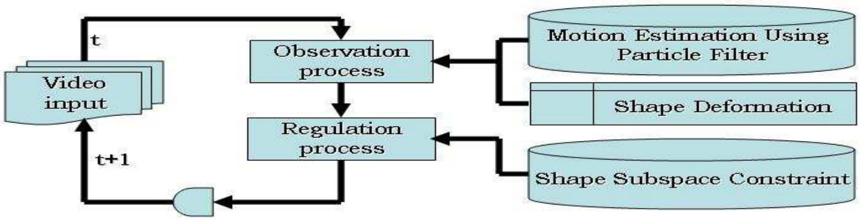


Fig. 1. *Illustration of the proposed tracking system.*

of estimating both motion and deformation in one step, we use a cascading framework. We propose to first estimate the motion, then the deformation. The estimation of deformation fulfill the operation of discriminating the real boundary from all the edge points, the majority of which may be from the background. Robustness can be improved by constraining the deformation by using a prior shape model. Therefore, we decompose the task of tracking non-rigid object contour into three components:

- 2D **Motion estimation** It estimates the object-wise spatial rigid-body motion, including translation and rotation parameters. Since the motion parameters are finite dimensional, we use a particle filter to estimate them.
- 2D **Shape deformation** It captures the pose changes of non-rigid objects. Each pixel on the boundary may have different but correlated deformations. We construct a deformation probability map based on statistically analyzing different cues in each frame. Deformation is determined according to boundary pixels with higher deformation probabilities.
- **Shape regulation** It uses a trained shape subspace to restrict shape deformations. Regulation also reconstructs the occluded parts of the contours. Our method adaptively integrates the off-line trained prior shape model with the object in the current video sequence.

Figure 1 presents a schematic illustration of the proposed system. The rest of the paper is organized as follows. In section 2, we introduce some preliminary concepts of the active contour tracking algorithm. In the next three sections, we describe the three cascading stages in the non-rigid contour tracking algorithm, which are motion estimation, deformation estimation and regulation. We present experimental results in section 6, followed by conclusions and discussions in section 7.

II. Preliminaries

A. B-spline Parametric Curves

In our contour tracking system, the tracking target is represented by the parametric B-spline curve. The visual 2D curves outlining the objects are represented in terms of parametric B-spline curves $\mathbf{r}(s) = [x(s), y(s)]^T$ [8]. The coordinates $[x(s),\ y(s)]$ are both spline functions of the curve parameter s. Furthermore, we use a set of control points $\mathbf{Q} = \{q_1, q_2, \ldots, q_L\}$ to represent the B-spline curve, where each control point is defined as $q_l = (q_l^x, q_l^y)^T$ and L is the number of control points. One important reason for using the control point representation is because a set of $\mathbf{Q}$ can uniquely determine one B-spline curve. If we define the dynamic model that describes the contour motion as an affine transform, it is sufficient to apply the transform to the control points. Once the control points are transformed, the B-spline curve is transformed in the same manner. The property not only significantly improves the computational efficiency, but also helps to discretify the infinite deformation parameters into finite deformation parameters, i.e., the deformation of the curve can be approximated by the deformations on the control points.

B. Particle Filter

The objective of motion tracking is to recursively estimate the state of the dynamic model given some noisy visual observations, which allows us to formulate a Bayesian model:

$$p(\theta_t|Y_{1:t}) \propto p(Y_t|\theta_t) \int p(\theta_t|\theta_{t-1}) p(\theta_{t-1}|Y_{1:t-1})\, d\theta_{t-1}, \tag{1}$$

where θ denotes the state vector, Y the observation, $p(Y_t|\theta_t)$ the likelihood function at time instant t. Observation Y is referred as images, or, visual cues in images. All inferences of the unknown state vector are based on the posterior probability in (1). The basic criterion is to find the vector with the maximum posterior probability. Many techniques can be used to achieve the goal, such as the Kalman filter [13] and the particle filter [7]. The former filter can be used when the data are modelled by a linear Gaussian model. The latter one, also known as the sequential Monte Carlo algorithm, is a set of simulation-based methods proposed to handle more complex data of non-Gaussianity, high dimensionality and nonlinearity. Many contour tracking algorithms use the particle filter algorithm because it is flexible, easy to implement, parallelizable and applicable to general settings. We also use the particle filter to estimate the state vector of the dynamic model.

In the particle filter algorithm, the *prediction step* samples new particles based on the state transition probability $p(\theta_t|\theta_{t-1})$, and the previous posterior distribution $p(\theta_{t-1}|Y_{1:t-1})$, while the *update step* is controlled by particle weights characterized by the likelihood function $p(Y_t|\theta_t)$:

$$\omega_t^{(j)} \propto p(Y_t|\theta_t^{(j)}), \tag{2}$$

The algorithm approximates the current posterior distribution $p(\theta_t|Y_{1:t})$ by a set of weighted particles $S_t = \{\theta_t^{(j)}, \omega_t^{(j)}\}_{j=1}^J$ with J representing the number of particles. To avoid the potential of the particles collapsing into a few particles with high weights, sequential importance sampling (SIS) [2, 9] draws particles from a proposal distribution $g(\theta_t^{(j)}|\theta_{t-1}^{(j)}, Y_{1:t})$ and eliminates particles with lower weights. The weights are assigned as:

$$\omega_t^{(j)} \propto \frac{p(Y_t|\theta_t^{(j)})p(\theta_t^j|\theta_{t-1}^{(j)})}{g(\theta_t^{(j)}|\theta_{t-1}^{(j)}, Y_{1:t})}. \tag{3}$$

The selection of proposal distribution depends on the properties of different applications.

III. Motion Estimation

A. Dynamic Motion Model

We use the parameters of 2D affine transform to represent the finite dimensional motion of the object, which allows rotation, translation and is independent of scales. The contour transform is given in terms of the homogeneous coordinates of the control points

$$\begin{bmatrix} q_{l,t-1} \\ 1 \end{bmatrix} = T \cdot \begin{bmatrix} q_{l,t} \\ 1 \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} & T_{13} \\ T_{21} & T_{22} & T_{23} \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} q_{l,t} \\ 1 \end{bmatrix}, \tag{4}$$

where $T \in \mathbb{R}^{3\times 3}$ represents an affine transform matrix with independent scale factors along both x and y axes. Accordingly, the state vector of the dynamic model is defined as

$$\theta_t = (T_{11} \;\; T_{12} \;\; T_{21} \;\; T_{22} \;\; T_{13} \;\; T_{23})^T. \tag{5}$$

B. Estimate State Vector by Particle Filter

We use the particle filter to obtain the MAP (maximum a posterior) estimator of the state vector θ.

B.1 Prediction Step

Rather than using a proposal distribution in prediction, we predict the configuration of particles based on the following state transition model:

$$\theta_t = \hat{\theta}_{t-1} + \nu_t + U_t \tag{6}$$

with $\hat{\theta}_{t-1}$ is the previous state estimate, ν_t as the predicted adaptive velocity in the motion vector and U_t as the driving noise, assumed to be zero-mean Gaussian noise. The computation of ν_t entails the incorporation of the previous particle configuration in the prediction. Therefore, the diversity of particles is not compromised.

The prediction of ν_t is based on the assumption of brightness invariance [32], which means that there exists a θ_t such that the warping patch is similar to the previous image patch. We denote $Z(\mathbf{Q}_t)$ as the intensities (colors) of the control point set $\mathbf{Q}_t$ (for robustness purpose, we use the corresponding intensities of Gaussian smoothed image frames), there exists θ_t that satisfies

$$Z_{t-1}(T(\theta_t) \cdot \mathbf{Q}_t) = Z_{t-1}(\hat{\mathbf{Q}}_{t-1})^1. \tag{7}$$

where $\hat{\mathbf{Q}}_{t-1}$ denote the control point set estimated at $t-1$. We simplify (7) by $\mathcal{T}\{Z_t, \theta_t\} = \hat{Z}_{t-1}$, where $\hat{Z}_{t-1}$ is the corresponding intensities of $\hat{\mathbf{Q}}_{t-1}$. Approximating $\mathcal{T}\{Z_t; \theta_t\}$ via a first-order Taylor series expansion around $\hat{\theta}_{t-1}$ yields:

$$\mathcal{T}\{Z_t; \theta_t\} \simeq \mathcal{T}\{Z_t; \hat{\theta}_{t-1}\} + \mathcal{C}_t(\theta_t - \hat{\theta}_{t-1}) = \mathcal{T}\{Z_t; \hat{\theta}_{t-1}\} + \mathcal{C}_t \nu_t, \tag{8}$$

where $\mathcal{C}_t = \partial \mathcal{T} / \partial \theta$ is the Jacobian matrix. Substituting $\hat{Z}_{t-1}$ into (8), we obtain:

$$\hat{Z}_{t-1} \simeq \mathcal{T}\{Z_t; \hat{\theta}_{t-1}\} + \mathcal{C}_t \nu_t \tag{9}$$

$$\nu_t \simeq -\mathcal{B}_t(\mathcal{T}\{Z_t; \hat{\theta}_{t-1}\} - \hat{Z}_{t-1}) \tag{10}$$

where $\mathcal{B}_t$ is the pseudo-inverse of $\mathcal{C}_t$. Using the differences in motion vectors and the observation matrix as inputs, we obtain a least square (LS) solution to $\mathcal{B}_t$ as:

$$\Theta^{\delta}_{t-1} = [\theta^{(1)}_{t-1} - \hat{\theta}_{t-1}, \ldots, \theta^{(J)}_{t-1} - \hat{\theta}_{t-1}] \tag{11}$$

$$\mathcal{Z}^{\delta}_{t-1} = [Z^{(1)}_{t-1} - \hat{Z}_{t-1}, \ldots, Z^{(J)}_{t-1} - \hat{Z}_{t-1}] \tag{12}$$

$$\mathcal{B}_t = (\Theta^{\delta}_{t-1} \mathcal{Z}^{\delta T}_{t-1})(\mathcal{Z}^{\delta}_{t-1} \mathcal{Z}^{\delta T}_{t-1})^{-1} \tag{13}$$

[1] More strictly, the affine transform on the LHS of (7) should be formulated as $T(\theta_t) \cdot \begin{bmatrix} \mathbf{Q}_t & \mathbf{1} \end{bmatrix}^T$.

where Θ^{δ}_{t-1} is the set of differences between all particle samples of $\{\theta^{(j)}_{t-1}\}^{J}_{j=1}$ and the optimal estimate $\hat{\theta}_{t-1}$, $Z^{(j)}_{t-1}$ is the j^{th} patch sample with state vector sample $\theta^{(j)}_{t-1}$, and $\mathcal{Z}^{\delta}_{t-1}$ is the set of all intensity differences between samples of $\{Z^{(j)}_{t-1}\}^{J}_{j=1}$ and $\hat{Z}_{t-1}$. Obviously, the particle configuration at $t-1$ is incorporated here for prediction.

B.2 Updating Step

The weight is updating based on the likelihood function $p(Y_t|\theta_t)$. We follow the definition of observation model in [10]. We search along the normal line $\mathbf{n}_l$ on each control point q_l, which is determined by corresponding θ, and detect feature points $\{z^{(l)}_j\}^{N_l}_{j=1}$, where N_l is the number of features detected along the normal. The existence of multiple feature points is due to background clutter. Assuming that $\{z^{(l)}_j\}$ can be modeled as a spatial Poisson distribution along the normal lines and the true control point is a Gaussian distribution, the 1-D measurement density along $\mathbf{n}_l$ can be determined by the distances between the feature points to the corresponding control point, formulated as

$$p_l(z|q_l) \propto 1 + \frac{1}{\sqrt{2\pi}\sigma\psi\lambda}\sum_{j=1}^{N_l}\exp{(-\frac{(z^{(l)}_j - q_l)^2}{2\sigma^2})}, \tag{14}$$

where ψ is the probability of non-detection, λ is the density of clutter in the Poisson distribution, σ is the standard deviation of the Gaussian distribution. Figure 2 is an intuitive illustration of evaluating the likelihood function for one control point. With the assumption that feature outputs on distinct control points are statistically independent, the overall likelihood becomes:

$$p(Y|\theta) = \prod_{l=1}^{L} p_l(z|q_l). \tag{15}$$

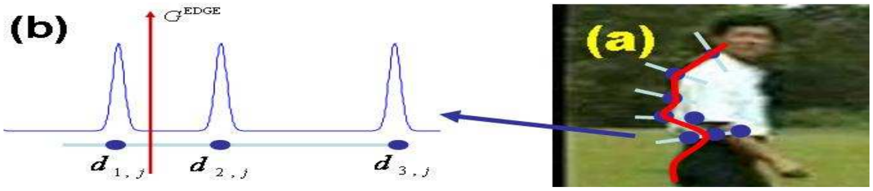


Fig. 2. *(a) The red line is the contour determined by the control points. Light lines are the normal lines on the control points. The solid dots are feature points detected by the normal lines; (b) 1-Dimensional measurement density along the line normal on one control point q_j. $d_{1,j}$, $d_{2,j}$ and $d_{3,j}$ are three distances between the feature points to the corresponding control point. The vertical axis G^{EDGE} represents the gradient magnitude along the normal line $\mathbf{n}_j$.*

IV. Deformation Estimation

After the MAP estimator $\hat{\theta}$ is obtained and the transformed control points $\hat{\mathbf{Q}}$ is acquired based on $\hat{\theta}$, the transformed curve $\tilde{\mathbf{r}}$ is determined. The next thing is to enforce local deformation, i.e., find the real boundary points. [10] used a simple strategy that the exact contour is determined by selecting feature points with maximum gradient magnitudes detected on corresponding normal lines. In other words, the contour estimation only counts on the gradient magnitudes. Unfortunately, this strategy does not always work, especially when the background is heavily cluttered or the object undergoes shape deformations between frames.

In our algorithm, we identify the correct feature points by detecting the deformation along the normal lines on transformed control points $\hat{\mathbf{Q}}$. Two elements are involved. One relates to the assumption that real boundary points of an object are detected along the orthogonal directions of the contour. It implies that the scanning range of normal lines influences the probability of the real boundary being detected. The other is that the gradient magnitudes are not sufficiently robust for exact contour delineation, especially when contaminated by background clutters and object textures. Accordingly, our strategy for deformation estimation contains two new features: (1) set normal lines adaptive; and (2) integrate several statistical cues into a deformation confidence map.

A. Set Normal Lines Adaptive

The scanning range of normal lines is determined by both the searching lengths and centers. Earlier algorithms [10, 16, 17, 28, 30] set the search lengths and centers of normal lines identical and fixed, which may result false detections due to lack of modeling shape variations. We enforce adaptability of the normal lines to reduce the probability of false detections.

A.1 Lengths of normal lines

Intuitively, a short normal line may miss the true boundary pixel while a long one may intersect with edge points from background clutter. To reduce the possibility of a normal line intersecting with background clutter without sacrificing the chance of finding the actual boundary pixel, the lengths of normal lines are altered according to the pose variations of the corresponding contour control points in training sequences. For example, in sequences of walking humans, the relative positions of the head and trunk change slightly from frame to frame; on the other hand, the sides, especially the legs and arms change their relative positions more. Therefore, we should set the normal lines with large pose variations longer than those of small pose variations. The pose variations of pixels can be learned offline (for example, walking pedestrian samples from USF dataset [22], consisting of one thousand 120 × 80 binary images with aligned pedestrian silhouettes.) as:

$$\xi(l) = \mathbf{E}\|\mathrm{q}_l^k - \mathbf{E}(\mathrm{q}_l^k)\|^2 \tag{16}$$

$$u(l) \propto L_{min} \log \frac{\xi(l)}{\min(\xi(l))} \tag{17}$$

L_{min} is a constant representing the minimum length, and k denotes the index of training samples.

A.2 Centers of normal lines

Earlier algorithms set the centers of scanning normal lines as the control points on the estimated contour, which may cause the intertwining of normal lines, because the same point may be selected twice along two normal lines and the output contour may end up looped. Making the line centers adaptive by applying a distance transform (DT) [12] significantly reduces the probability of normal line intersections inside the object. The detailed steps are listed in Table I. The given algorithm only concerns close contours. For open contours, we will force them to be closed by simply linking the first point and the last point. Figure 3 demonstrates the procedure of sketching adaptive normal lines along the estimated contour, with which we are able to search for the real contour pixels with more flexibility.

TABLE I

Algorithm 1: set center of the normal line adaptive

1. Based on the transformed control points set $\hat{\mathbf{Q}}_t$ (the result from global motion estimation), construct a binary image $\mathcal{BI}$, set the region Ω circled by the contour $\tilde{\mathbf{r}}_t$ to 1;
2. Apply DT to $\mathcal{BI}$ to obtain a distance map $\mathcal{DI}$, which is defined as:

$$\mathcal{DI}(\mathbf{x}) = \begin{cases} \min_{\mathbf{y}\in\tilde{\mathbf{r}}} dist(\mathbf{x},\mathbf{y}), & \mathbf{x}\in\Omega; \\ 0, & \text{otherwise.} \end{cases} \quad (18)$$

3. On each control point $\hat{q}_l$, draw a normal line $\mathbf{n}_l$, find maximum distance value satisfying $\mathcal{DI}_l = max_{\mathbf{x}\in\mathbf{n}_l}\mathcal{DI}(\mathbf{x})$. The lengths of the normal line on two sides of the control point are set as follows:

$$u(l)_{in} = min(u(l)/2, \mathcal{DI}_l - d_0) \quad (19)$$

$$u(l)_{out} = max(u(l)/2, u(l) - [\mathcal{DI}_l - d_0]) \quad (20)$$

where d_0 represents a minimum safe distance to avoid contour loops. Here $d_0 = 2$.

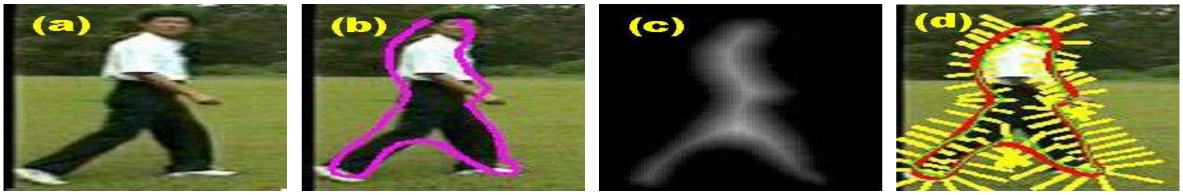


Fig. 3. *An example of how to make a normal line scanning adaptive. (a) cropped original object; (b) estimated contour; (c) distance transformed object; (d) normal lines on control points.*

B. Multi-cue Deformation Probability Map

To extract the real contour, we define a posterior deformation probability map as $P_t(Y|\hat{\mathbf{Q}})$ based on the transformed control point set obtained by global motion estimation with Y represents related visual cues. In the map, a high probability implies that the corresponding pixel is more likely to be on the real contour and a low value implies a lower likelihood. Instead of using edge magnitudes as the only visual cue, we integrate several cues to evaluate the deformation probability.

B.1 Multi-cue Fusion

Fusion with respect to different cues can be interpreted as using multiple measurement sources. Assume that we have M cues, the observation can be represented by $Y = (Y_1, \ldots, Y_M)$. We further assume that the observations are conditionally independent [20]. The deformation probability is therefore factorized as:

$$P_t(Y|\hat{\mathbf{Q}}) = \prod_{i=1}^{M} P_t(Y_i|\hat{\mathbf{Q}}) \tag{21}$$

Scanning for pixels with maximum probability values (ML estimator) on adaptive normal lines, we obtain the refined contour pixels, denoted as $\hat{\mathbf{r}}$. As an example, we show some probability maps from different cues on one processing frame in Figure 4, together with the fusion map $P_t(Y|\hat{\mathbf{Q}})$. Apparently, the fusion result subdues noise from the gradient magnitude cue inside the contour, which is caused by object textures. We introduce the computation of probability maps from different cues in the rest of this section. It is worth nothing that since the scanning range for each pixel is in one dimensional, the two dimensional deformation probability map P can be further simplified to several one dimensional probability vectors associated with each control point. Therefore, the calculation of the deformation probabilities is limited to normal lines. This helps to improve the computational efficiency.

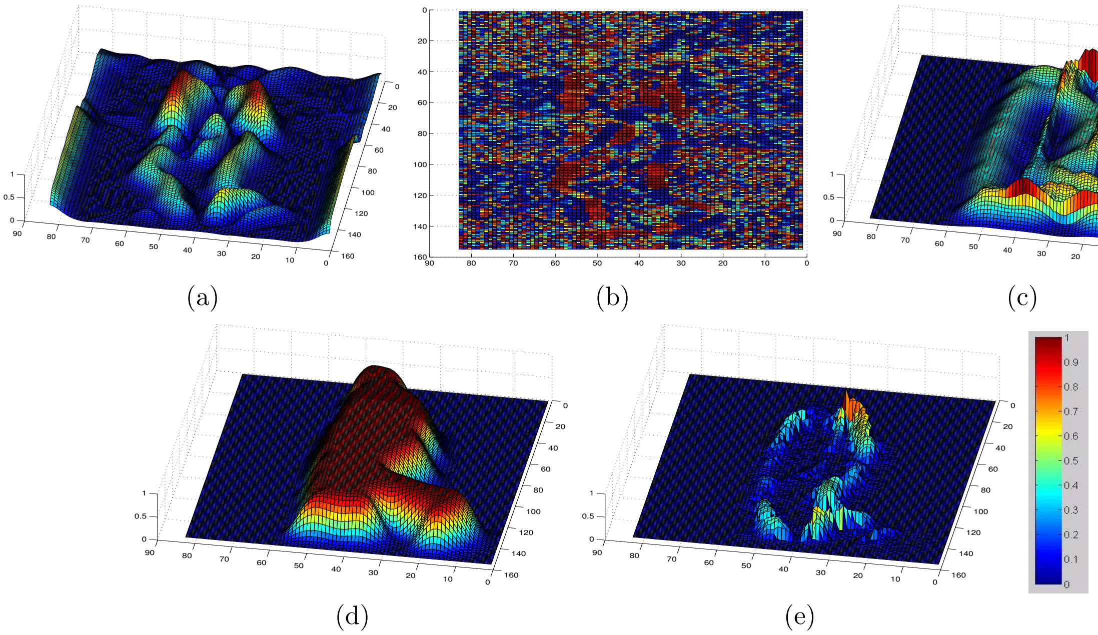
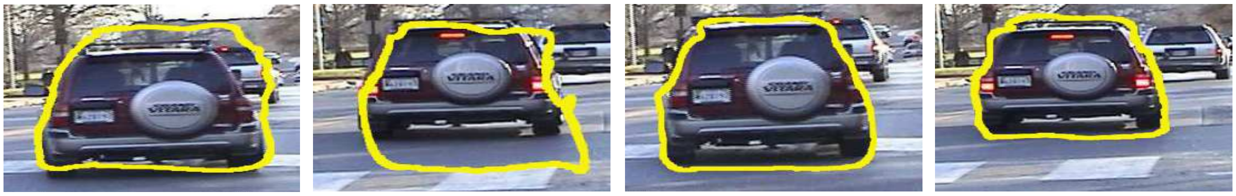


Fig. 4. *An illustration of fusion from different visual sources, including the probability maps of (a)gradient Magnitude; (b) gradient Orientation; (c) shape Template; (d) foreground; (e) fusion.*

B.2 Gradient Magnitude Cues

As shown in Figure 4.(a), gradient magnitude is an important feature for representing an object boundary. However, when the object itself is not homogenous in color or intensity, many edges are generated inside the object. We want to minimize the effect of inside edges. Anisotropic diffusion is one possible approach to make the entire image to be more uniform in color or texture, while still preserving the object boundaries [21]. Therefore it is highly probable that points with high gradient magnitudes after diffusion belong to the boundary of the target.

After the diffused feature map $\mathcal{EI}$ is extracted from the original image, a motion mask $\triangle\mathcal{I}$ indicating the possible area where motion could occur is applied to $\mathcal{EI}$ to further suppress the background clutter. The masked map is then filtered by $\mathbf{G}$, a smoothing Gaussian filter.



(1) Using the regular edge map *(2) Using the feature maps $\mathcal{FI}$*

Fig. 5. *Performance comparisons on a cluttered scene. A stabilization step is applied to obtain the second set of results to obtain $\mathcal{FI}$ because the sequence was acquired by a moving camera.*

Eventually, we map the filtered map to a magnitude probability map P by:

$$P_t(Y_m|\hat{\mathbf{Q}}) = \alpha_m(\mathcal{EI} \bullet \triangle\mathcal{I}) * \mathbf{G} \tag{22}$$

where α_m is a normalizing coefficient, $*$ denotes convolution. Figure 5 demonstrates an example of performance improvement when we utilize the diffusion edge map convolved with the motion mask. Since the background and the tracker object are both moving, a background stabilizing step [31] is applied to estimate the motion mask $\triangle\mathcal{I}$. The stabilization is based on the idea that the background movement can be modelled as a planar affine transform.

B.3 Gradient Orientation Cues

An gradient orientation map $\mathcal{OI}$ provides the orientations of edges. Gradient orientation is a useful feature to discriminate the real object boundaries from all the detected edges, especially when background clutter is present. If we denote the normal orientation of the true boundary as ϕ, it is expected that the local normal orientation should present a Guassian distribution with mean equal to ϕ [5]. While the normal orientation distributions of pixels not on the boundary tend to be a uniform distribution between $[0, 2\pi)$. Figure 4.(b) visually illustrates the different distributions presented by the pixels on the contour and those not on the contour. This leads to the definition of the orientation probability map as:

$$P_t(Y_o|\hat{\mathbf{Q}}) \propto \exp(-\frac{(\mathcal{OI}_t(\mathbf{x}) - \hat{\phi}_{t-1}(l))^2}{\sigma_o^2}) \quad \forall \mathbf{x} \in \mathcal{R}(\mathbf{n}_l), \tag{23}$$

where $\mathcal{R}(\mathbf{n}_l)$ defines a proximity region to $\mathbf{n}_l$:

$$\mathcal{R}(\mathbf{n}_l) \triangleq \{\mathbf{y} \in \mathcal{R}(\mathbf{n}_l) : dist(\mathbf{y}, \mathbf{n}_l) \leq dist(\mathbf{y}, \mathbf{n}_k), k \neq l\} \tag{24}$$

B.4 Shape Template Cues

The shape of tracked object always has certain pattern. Therefore, the shape template could be used as one cue, indicating the probability that each image pixel belongs to the real object contour. Our shape template is different from the static shape energy proposed by Cremers *et al.* [6], which is pretrained and kept unchanged during tracking. We use an online model that incorporates a dynamic part that varies according to the observations (transformed contour by global motion estimation). Denoted Y_s as the shape template observation, it contains the shape prior model $\mathcal{A}_{\mathbf{S}}$ and the dynamic template $\mathcal{A}_{\hat{\mathbf{Q}}}$. The former is a static template created from the training data, and the latter follows a Gaussian distribution, the mean of which is set to the transformed contour $\hat{\mathbf{Q}}$. The probability is given by:

$$P_t(Y_s|\hat{\mathbf{Q}}) = a_t\mathcal{A}_{\mathbf{S}} + (1 - a_t)\mathcal{A}_{\hat{\mathbf{Q}}} \tag{25}$$

where $0 < a_t < 1$ is the weight that controls the integration of $\mathcal{A}_{\mathbf{S}}$ and $\mathcal{A}_{\hat{\mathbf{Q}}}$. The construction of $\mathcal{A}_{\mathbf{S}}$ is straightforward based on how frequently it appears as 1 in the training data. Figure 4.(c) is one example of the probability map from the shape template cue. We show more examples of probability templates in Figure 6.

B.5 Foreground Cues

To suppress the contamination from the background clutter, we use a foreground probability map $P_t(Y_f|\hat{\mathbf{Q}})$ that estimates the likelihood of a pixel belonging to the tracked object

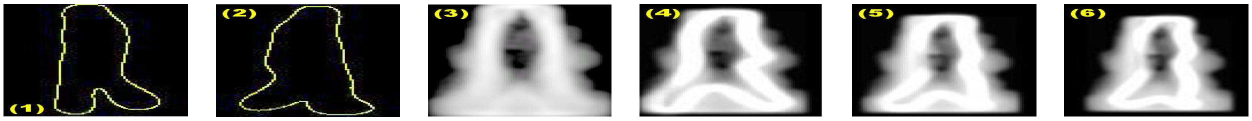


Fig. 6. *Illustrations of adaptive probability shape templates for pedestrian. (1),(2): shape training samples; (3): shape prior model; (4),(5),(6): examples of probability shape templates in different frames.*

or not. This map is calculated by comparing the current frame to a set of background models representing the static parts of the scene. The pixel-wise background models are adapted directly by the previous values for static camera setups. For moving cameras, these models can be fit after consecutive frames are aligned on a mosaic by global motion estimation. We define the background as layers of multivariate Gaussian functions $\{(\mu_t^i, \mathbf{\Sigma}_t^i, \kappa_t^i, \upsilon_t^i)\}_{i=1..\mathcal{K}}$ where μ_t^i is the posterior mean, $\mathbf{\Sigma}_t^i$ is the marginal posterior covariance, υ_t^i is the degrees of freedom, κ_t^i is the number of prior measurements of the i^{th} layer, and $\mathcal{K}$ is the number of layers in 3D color space. At each frame, we update the layer parameters using an online Bayesian estimation method as described in [23]. We order the layers according to confidence scores. Our confidence measure is inversely proportional to the determinant of covariance. Then, we select the layers having confidence value greater than a layer threshold. We measure the Mahalanobis distance of observed color $I(\mathbf{x})$ from the layers

$$d_i(\mathbf{x}) = (I(\mathbf{x}) - \mu_{t-1}^i)^T (\mathbf{\Sigma}_{t-1}^i)^{-1} (I(\mathbf{x}) - \mu_{t-1}^i) \tag{26}$$

and, update the parameters of the confident layers. Pixels that are outside of 99% confidence interval of all confident layers of the background are considered as foreground pixels. After the update, the foreground probability map at a pixel is determined as

$$P_t(Y_f|\hat{\mathbf{Q}}) = \alpha \exp(-\min_{i=1}^{\mathcal{K}} d_i(\mathbf{x})) \tag{27}$$

where α is a normalizing constant. Figure 4.(d) is one example of the probability map based on the foreground cue.

V. Regulation on Shape Deformation

Based on the estimated global motion and local deformation, we could track the contour from frame to frame. However, what if the deformation estimate is severely corrupted by noise? For example, the exact contour points may not always coincide with maximum probability pixels, but may be present at weaker secondary pixels. In such cases, shape regulation is important to serve as a constraint to recover from errors. The intuition is that learning the prior shape knowledge of the object from the training set could help for delineation. Ideally, the training samples should cover all deformation variations. If an object in one frame exhibits a particular type of deformation not present in the training set, the system searches for the deformation in the subspace that is closest to the target, i.e. the system projects any deformation onto the subspace. The regulation is therefore achieved.

A. Generic Shape Model

There are several approaches for subspace construction. One of them was introduced by Cootes *et al.*, the active shape model (ASM) [3] by using the points distribution model (PDM) was described in [4] for obtaining the shape subspace. Our training method is based on the idea of PDM. A shape model is defined in terms of x and y coordinates of every "landmark" point lying on the outline of the target. The number of "landmark" points is fixed at equal intervals along the contours. The *control points* of B-splines are regarded as these "landmark" points. Table II gives the steps for training a prior model from a set of N samples, each represented by a set of columnized L control points $\mathbf{Q_s}^i = \{q_s^{(i,j)} | 1 \leq i \leq N, 1 \leq j \leq L\}$.

B. Adaptive Shape Model

The constructed shape model is a generic model that can apply to all cases as long as a target belongs to the corresponding object category. However, what we really need is a

TABLE II

ALGORITHM 2: CONSTRUCT PRIOR SHAPE MODEL

(a) Align the set of examples into a common frame of reference, $\mathtt{x}^i$=aligned($\mathbf{Q_s}^i$) [4];
(b) Calculate the mean of the aligned examples $\bar{\mathtt{x}}$, and the deviations $\delta\mathtt{x}^i = \mathtt{x}^i - \bar{\mathtt{x}}$;
(c) Calculate the eigensystem of the covariance matrix of the deviations, $\mathbf{C} = \frac{1}{L}\sum_{i=1}^{M}(\delta\mathtt{x}^i)(\delta\mathtt{x}^i)^T$.
(d) The first $\mathtt{t}$ principal eigenvectors of the eigensystem are used to generate $\mathtt{x} = \bar{\mathtt{x}} + \mathbf{Pb}$, where $\mathbf{b}$ is a $\mathtt{t}$-element vector of shape variation parameters and $\mathbf{P}$ is a $2L \times \mathtt{t}$ matrix of $\mathtt{t}$ eigenvectors, which composes the estimated shape subspace. We denote the eigenvalue diagonal matrix as $\Lambda_{\mathtt{t}}$, which is a $\mathtt{t} \times \mathtt{t}$ matrix.

deformation model which can more accurately represent the shape variations in the sequence being considered. One solution is to update the existing PCA model with the initial contour of the current sequence (either manually marked or automatically detected) [33]. Denote $\mathtt{x}_0$ as the aligned initial contour, vector $\mathbf{b}_s = \mathbf{P}^T(\mathtt{x}_0 - \bar{\mathtt{x}})$ as the subspace component, the projection residue is obtained as:

$$\mathtt{x}_r = \mathtt{x}_0 - \bar{\mathtt{x}} - \mathbf{Pb}_s. \tag{28}$$

The residue part represents the shape variation not being covered in the prior model, so the generic subspace $(\bar{\mathtt{x}}, \mathbf{P}, \Lambda_{\mathtt{t}})$ is updated corresponding to the residue by the following equations:

$$\begin{aligned}
\bar{\mathtt{x}}^* &= \beta\bar{\mathtt{x}} + (1-\beta)\mathtt{x}_r && (29)\\
\mathtt{e}_r &= \frac{\mathtt{x}_r^T}{\|\mathtt{x}_r\|}(\mathtt{x}_0 - \bar{\mathtt{x}}) && (30)\\
\mathbf{C}^* &= \beta\begin{bmatrix} \Lambda_{\mathtt{t}} & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix} + \beta(1-\beta)\begin{bmatrix} \mathbf{b}_s\mathbf{b}_s^T & \mathtt{e}_r\mathbf{b}_s \\ \mathtt{e}_r\mathbf{b}_s^T & \mathtt{e}_r^2 \end{bmatrix} && (31)
\end{aligned}$$

where β is the update weight. By applying SVD to (31), we obtain $(\mathbf{P}^*, \Lambda^*_{\mathtt{t+1}})$ satisfying $\mathbf{P}^*\Lambda^*_{\mathtt{t+1}}(\mathbf{P}^*)^T = \mathbf{C}^*$. For brevity, we still use $(\bar{\mathtt{x}}, \mathbf{P}, \Lambda_{\mathtt{t}})$ to denote the updated shape subspace in the rest of the paper.

C. Subspace Projection

The projection of deformation can be described as representing the deformed contour by a linear combination of basis in the shape subspace. We first align the deformed contour point vector set $\hat{\mathbf{r}}_t$ to $\mathtt{x}_t$, and then apply

$$\mathtt{x}_{p,t} = \mathbf{P}\mathbf{P}^T(\mathtt{x}_t - \bar{\mathtt{x}}) + \bar{\mathtt{x}}, \tag{32}$$

where $\mathtt{x}_{p,t}$ is a linear combination of subspace basis. It is possible that some control points on $\vec{\mathtt{r}}$ may be occluded, or not detected along the normal lines. Let us denote the index set of detected points as $I_d = \{i_1, i_2, \ldots\}$. We can recover a complete projected contour as follows:

$$\mathtt{x}_{p,t} = \mathbf{P}\mathbf{P}^{\dagger}_{I_d}(\mathtt{x}_{I_d,t} - \bar{\mathtt{x}}_{I_d}) + \bar{\mathtt{x}} \tag{33}$$

$$\mathbf{P}^{\dagger}_{I_d} = (\mathbf{P}^T_{I_d}\mathbf{P}_{I_d})^{-1}\mathbf{P}^T_{I_d} \tag{34}$$

A projection example is demonstrated in Figure 7 with a comparison between tracking results with and without shape regulation. Apparently, using the subspace can preclude the contour from deforming to an irregular shape.

D. Alignment

The usage of alignment is to normalize the contour, because the shape subspace is constructed from normalized training samples. It follows the method for rigid shape matching proposed by Cootes *et al.* [3]. The basic idea is to find a transform matrix (containing the

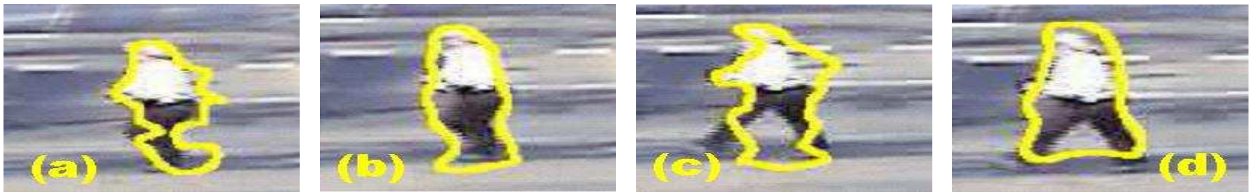


Fig. 7. *Comparisons between tracking results with and without subspace regulation. (a) and (c) are without regulation, (b) and (d) are with regulation.*

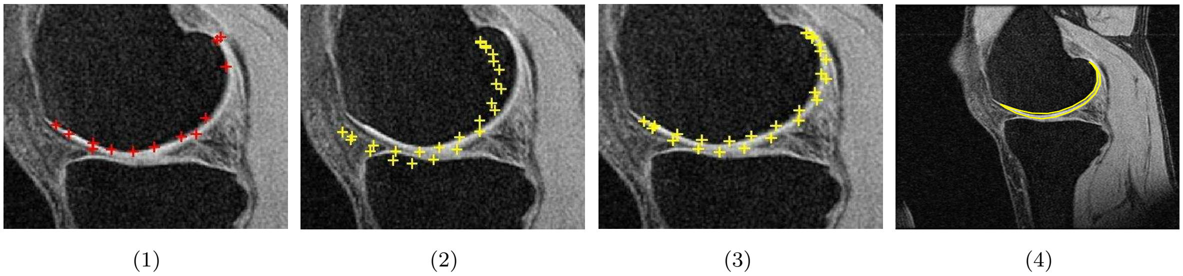


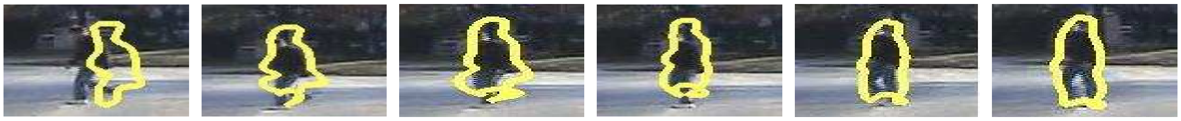
Fig. 8. *One frame from an MRI sequence. The target of interest is the articular cartilage layer. (1) depicts the detected contour pixels. (2) shows the recovered contour pixels using the shape subspace projection. (3) depicts the contour pixels after alignment. (4) the final result.*

rotation, translation and scale coefficients) and match the processing contour to the mean of the shape model. An example is shown in Figure 8. The sequence is collected by magnetic resonance imaging (MRI) of human knees. Our primary interest is in the articular cartilage layer. 8.(1) depicts the detected contour pixels. There are some pixels that are too obscure to be detected. 8.(2) shows the projected contour using the shape subspace without alignment. 8.(3) depicts the contour pixels after alignment, and 8.(4) gives the final result.

VI. Implementation and Experiments

A. Algorithmic Implementation

A summary of the complete contour tracking algorithm is given in Table III. We want to further discuss the initialization. The easiest way to acquire the initial contour to start tracking is by manually sketching it around the object of interest in the first frame. The pixels along the contour are sorted in the clockwise order. The control points are then selected based on the uniform arc length rule which becomes the initial contour. We can also apply automatic shape detection methods, such as the direct use of probability shape template to detect pedestrians [18]. A nice property of our tracking algorithm is that it has a high tolerance to localization errors in the initial contours. In our experiments, we observed that a very approximate but reasonable initial guess can evolve to capture the real boundary



frame 1 frame 2 frame 3 frame 4 frame 5 frame 6

Fig. 9. *An example of starting tracking using a very rough initial contour. After tracking for six frames, we find that the contour has been attached to the object fairly precisely.*

of the object of interest after tracking for three to six frames, as one example demonstrated in Figure 9.

B. Experimental Results

We have applied the cascading contour tracker to different sets of outdoor surveillance video sequences, containing moving people and vehicles. All the objects of interests are assumed to be objects moving in non-rigid forms. In most cases, the backgrounds contain clutter that undermines the tracker's performance. Among them, four sets of sequences were captured by moving cameras, which means that when computing the probability maps, the foreground cue is not involved. The other two sets of sequences were captured using static

TABLE III

Algorithm 3: Active Contour Tracking

Step1. Initialization: Draw a set of particles from the prior $p(\theta_0)$ to obtain $\{\theta_0^{(j)}, \omega_0^{(j)}\}, j = 1, \ldots, J$, where J is the number of the particles. Get the initial control point set $\{\mathbf{Q}_0^{(j)}\}$ from θ_0. Set $t = 1$.

Step2. Global motion estimation:

Step2.1 Prediction: Estimate the state vector shift ν_t, draw particles $\{\theta_t^{(j)}\}$ and accordingly the control point set samples $\{\mathbf{Q}_t^{(j)}\}, j = 1, \ldots, J$.

Step2.2 Update: Calculate the likelihood function $\mathcal{L}(Y_t|\theta_t^{(j)})$ and the posterior $\pi_t^{(j)} = p(\theta_t^{(j)}|Y_{1:t})$ for each sample, then normalize $\{\pi_t^{(j)}\}$ and update $\{\theta_t^{(j)}, \omega_t^{(j)}\}, j = 1, \ldots, J$. Find the MAP estimator of the global motion $\hat{\theta}_t = \theta_t^{\arg\max_{j\in\{1,\ldots,J\}} \pi_t^{(j)}}$ and the corresponding $\hat{\mathbf{Q}}_t$.

Step3. Local Deformation Estimation: Based on the estimated $\hat{\mathbf{Q}}_t$, generate the deformation probability map P_t. The deformed contour $\hat{\mathbf{r}}_t$ can be determined by scanning along the adaptive normal lines $\mathbf{n}_{l,t}$ for pixels with maximum deformation probabilities.

Step4. Regulation: Project $\hat{\mathbf{r}}_t$ onto the shape subspace to acquire the final estimated contour.

Step5. $t \rightarrow t + 1$, go to step.2.

cameras, so the foreground cue is used for generating the deformation probability map. The algorithm speed implemented by C++ code are 6-10 fps (frames per second) on a 1.5GHz PC running Windows XP. In our experiments, most comparisons are made between our method and the traditional method described in [10].

B.1 Experiment on Stationary Camera Data

Figure 10 shows a sequence (the frame size is 437×90×3) acquired by a stationary camera. It contains a pedestrian walking through the scene. The traditional active contour tracker works not well due to two possible reasons: (1) the normal lines on some of the control points do not detect the edge points; and (2) in some frames the contour gets intertwined. Our proposed method takes advantage of the adaptive normal-line strategy to avoid intertwined contour, and the subspace projection to recover missing edge points. The proposed method achieves satisfactory result throughout 159 frames where the object is presented.

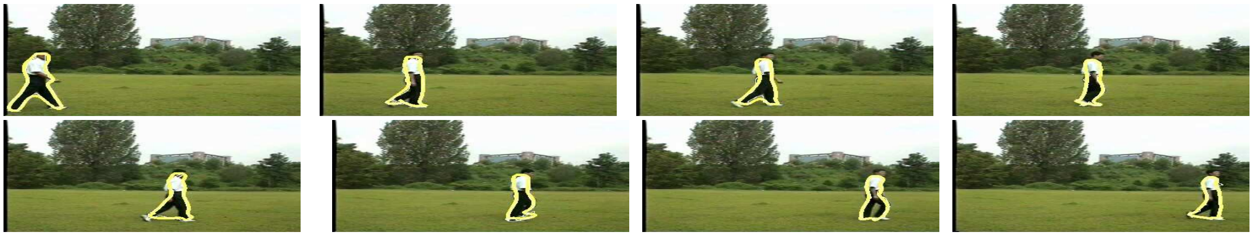


Fig. 10. *Pedestrian tracking performance with a stationary camera. The thick yellow lines represent the final tracked contours.*

B.2 Experiments on Moving Camera Data

Figures 11, 12, 13 and 14 demonstrate some typical results for moving camera sequences.

Figure 11 shows a sequence (the frame size is 543×814×3) capturing a moving SUV by a camera from another vehicle following it. Although the rear view of the vehicle is a rigid object, the surrounding disturbances and small view changes throughout the video lead to failures when the traditional rigid-object trackers are used. Two major distractions are from plain road marks and the sudden appearance of another vehicle in front of our target. Still, we obtain good results due to the usage of a deformation map based on several statistical cues.

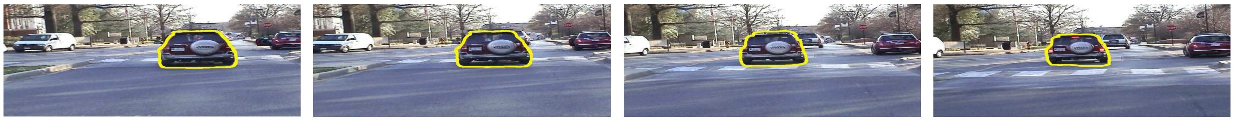


Fig. 11. *The results of tracking a vehicle from the rear view. The comparison result is given in Figure 5.*

Figure 12 and 13 illustrate two challenging sequences with walking pedestrians crossing the road. The frame size in both sequences is 541×818×3. Tracking difficulties arise due to the following facts: 1) the camera is moving forward very fast and, therefore, the global motion of the pedestrian not only includes translation and rotation, but zoom as well; 2) the background is full of road marks and shadows. The traditional contour trackers get distracted by these background clutters; 3) the pedestrian in Figure 12 wears a white shirt and a black pant. The strong contrast between two parts usually leads to the tracking result shrinking to either the upper part or the lower part of the body; and (4) the object has very similar color with respect to the background (woods) in Figure 13. The results of our tracker are satisfactory. We notice that the poses of the pedestrians vary significantly from frame to frame in both sequences. However, the tracking remains robust.

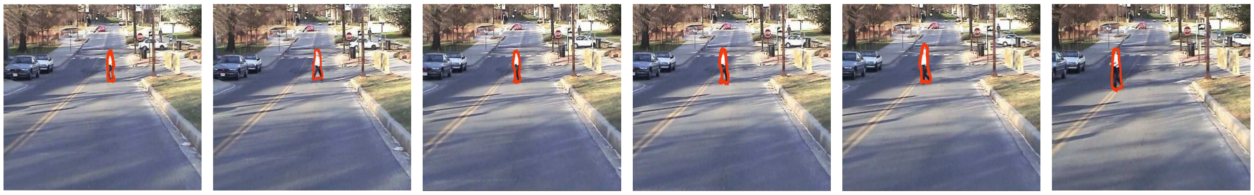
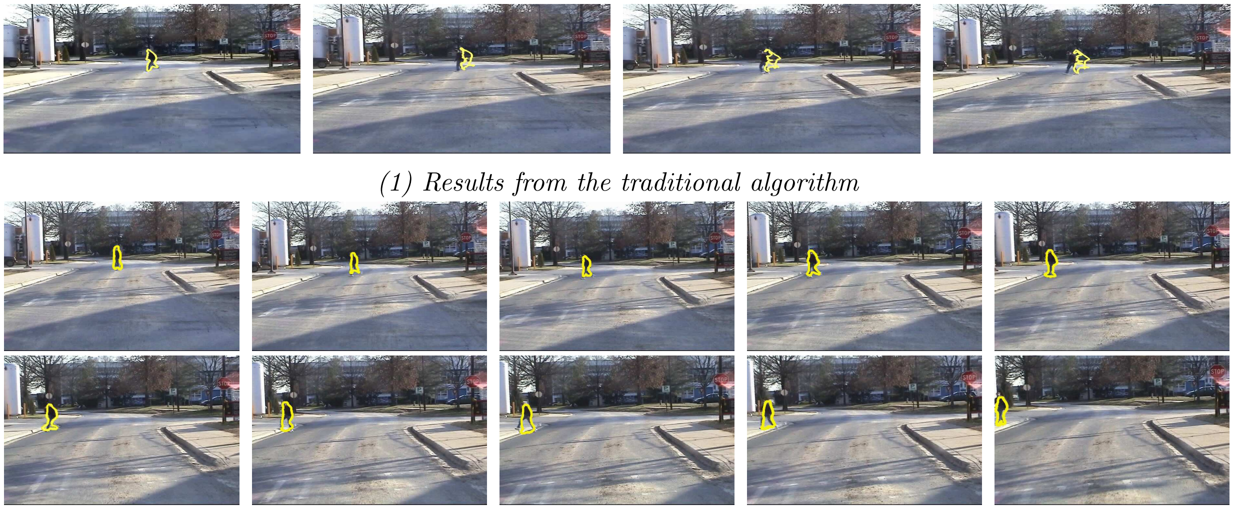


Fig. 12. *A sequence with both background and object moving. The challenges of processing this sequence are due to: 1) camera motion; 2) background clutter; 3) the pants and the shirt that the pedestrian is wearing are of very different colors.*

Figure 14 gives an example with a sequence containing a moving truck (the frame size



(1) Results from the traditional algorithm

(2) Results from the proposed algorithm

Fig. 13. *A sequence with both background and object moving. The background is heavily cluttered, and the intensity of the tracking pedestrian is very similar to the background color.*

is 480×720×3). Although a truck can not be taken as a non-rigid object, we still observe 2D shape deformation on the truck due to its 3D rotation. This sequence demonstrates the advantage of using contour-based tracking. The truck in the sequence changes views from the back view to the side view, which means that some of corresponding points on the object may disappear in the scene. Such a view change will strongly undermine the result of a regular 2D appearance-based tracker unless a 3D appearance model applies. As mentioned in section I, contour-based tracking can always yield robust results without requiring correspondences and 3D reconstruction. The results are promising even with the presence of many challenges, such as obscure boundaries, low color contrast, nonstationary camera and background clutter.

B.3 Experiments on Occlusion Data

We exploit the application of shape regulation to recover occluded contours, the results of which are shown in Figure 15. This sequence (the frame size is 576×768×3) contains walking humans acquired by a stationary camera. In the last several frames, the pedestrian

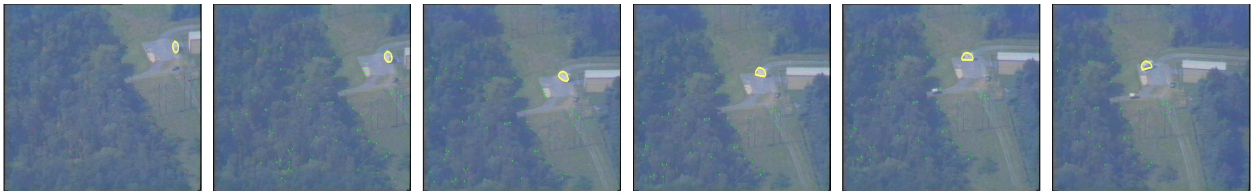


Fig. 14. *An airborne sequence with a white truck moving on the ground. The truck shows a back view in frames (1) and (2), then shows the back-right view in frame (3), a side view in (4) and (5) and a front-right view in frame (6). This experiment demonstrates that the contour-based tracker can give satisfactory results on sequences containing 3D object rotations.*

is partially occluded by trees. With subspace projection, we see that the occluded parts have been reconstructed. In this sequence, the clutter is heavy due to the presence of parked cars and trees in the scene.

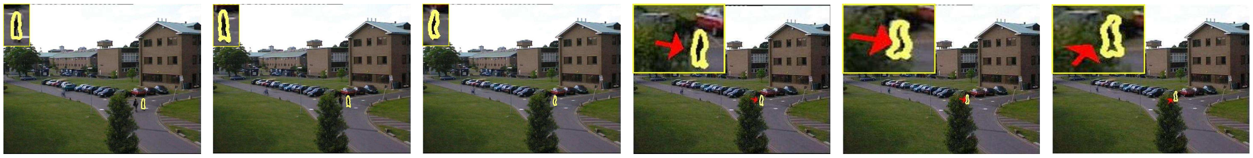


Fig. 15. *An example of an occluded contour being recovered by shape subspace reprojection. In the last three frames, the pedestrian is partially occluded by surrounding trees. Our tracking result recovers the partially occluded contour. Red arrows indicate the occluded parts. We may also note that parked cars contribute to background clutter.*

B.4 Experiments on medical sequence

The MRI sequence in our experiment consists of 2D image slices that form a 3D image cube for subsequent 3D visualization, in which we are particularly interested in the articular cartilage layer (a very thin, white, crescent-like layer). The difficulties of tracking these layers are due to the following facts: the surrounding tissues often have similar intensity values, which leads to some boundary points becoming nearly undetectable; the sequence is of low resolution due to the preprocessing step that enlarges the original image. The traditional active contour method is not effective in this case, because it lacks means to handle edge point occlusion. The "snake" method also has difficulty in finding the correct boundary due to the similar intensity values between the cartilage layer and the tissues surrounding it. Our

TABLE IV

Comparison of tracking results in heavy-cluttered background sequences and non-rigid object sequences. MSSD is the Mean Sum of Squared Distance, VAR is the variance of the distances, ACT is the traditional active contour tracking algorithm.

	ACT		Cascading Tracker	
	MSSD	VAR	MSSD	VAR
heavy-cluttered background	14.7584	74.4052	7.9129	12.5602
non-rigid object	9.9741	24.6597	6.0571	8.5222

method works well in this scenario. Figure 16 demonstrates the tracking results of applying the proposed algorithm to the MRI sequence, in which the frame size is 584×584. As a comparison, we also provide a set of tracking results using the traditional contour tracker.

B.5 Performance Evaluation

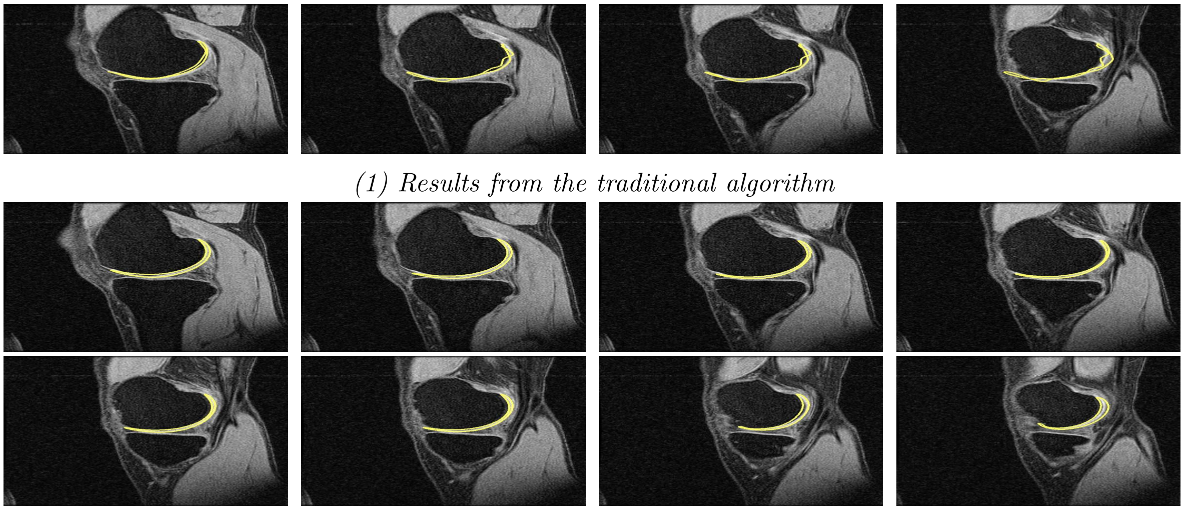
We use the Mean Sum of Squared Distance (MSSD) [1] measure to evaluate the tracking performance of different algorithms. For a sequence with K frames, where contour $\mathbf{r}_k$ in each frame has L control points, $\{(x_{k,1}, y_{k,1}), \ldots, (x_{k,m}, y_{k,m})\}$, we define:

$$MSSD = \frac{1}{K} \sum_{k=1}^{K} \frac{1}{L} \sum_{j=1}^{L} (x_{k,j} - x_{k,j}^0)^2 + (y_{k,j} - y_{k,j}^0)^2 \tag{35}$$

where $[x^0, y^0]$ represents the corresponding ground truth. We compare the proposed algorithm with the active contour tracking method for two cases: sequences with heavily-cluttered backgrounds and sequences with strong non-rigid movements. Table IV shows the values of MSSD and variance of squared distance in these two cases respectively. From all the experimental results and the comparison table, we find that the cascading tracker performs well in most of the cases.

VII. Discussion

The cascading contour tracker we have presented is motivated by the fact that the non-rigid movement can be decomposed to global motion and local deformation. The algorithm



(1) Results from the traditional algorithm

(2) Results from the proposed algorithm

Fig. 16. *Magnetic resonance imaging scans of human knees. The area of interest is the articular cartilage in each image. The upper row demonstrates the results by applying the traditional algorithm. The middle and bottom rows show the results by applying our proposed method.*

contains three major steps: motion estimation, deformation estimation and shape regulation. The following discussion covers the major aspects of the algorithm.

Multi-step vs. Single-step Compared with most methods using single-step estimation [10, 16, 24] to obtain the non-rigid contour movement with motion and deformation simultaneously, we choose a cascading framework to estimate motion and deformation separately. One-step estimation presents more systematic formulas, but it suffers from high dimensional computation and poor efficiency. Fortunately, the multi-step approach characterizes an efficient solution. The explanation is as follows: we use an affine transform to model the global motion. Therefore, the motion state vector is only six-dimensional, which eases the burden of the particle filter. We interpret shape deformation by the deformations occurred on control points, which approximates the infinite dimensional by finite dimensional. Since the processing is carried out in a successive manner and a rough contour is obtained by motion estimation, the deformation is searched only in a proximate region of the rough contour and does not involve numerical simulations (particle filter). Thus, the entire computation complexity is reduced.

Multi-cue vs. Single-cue Although gradient magnitude is a very strong cue to estimate the boundary pixels, sometimes it is not robust. In our method, estimating deformation counts on more visual cues with the aim to render contour detection more robust. We could further improve the fusion method by associating adaptive fusion weights with different cues [27].

Adaptive vs. Non-adaptive Normal Line We adaptively set the scanning normal lines according to the prior shape pose variations and previous shape estimate. The advantage of adaptability is: (1) it reduces the probability of loop occurrence; and (2) setting appropriate lengths of normal lines increases the possibility of correct detection for real boundary pixels, while reduces the chance for false detection.

Regulation vs. Non-regulation Regulation is important for non-rigid contour tracking. It not only constrains shape deformation, corrects estimating errors, but recovers the occluding contour pixels as well. Therefore, the method is more applicable to scenarios with non-rigid object movements and heavily-cluttered backgrounds.

Our method can successfully track non-rigid objects and get tight contours enclosing the changing shapes of the targets throughout the sequences. We are currently working on extensions of the algorithm to the multi-target tracking problem. Model regulation is also worth further exploiting. We only use shape prior knowledge as constraint in this paper. However, training samples contain not only shape, but appearance and motion information as well. Constructing a prior model based on all information to improve the robustness of tracking will be an interesting problem.

References

[1] Y. Akgul and C. Kambhamettu. A coarse-to-fine deformable contour optimization framework. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 25(2):174–186, Febuary 2003.

[2] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Trans. on Signal Processing*, 50(2):174–188, Febuary 2002.

[3] T. F. Cootes and C. J. Taylor. Active shape models. In D. Hogg and R. Boyle, editors, *3rd British Machine Vision Conference*, pages 266–275. Spinger-verlag, Sept 1992.

[4] T. F. Cootes and C. J. Taylor. Combining point distribution models with shape models based on finite-element analysis. In E. Hancock, editor, *5th British Machine Vision Conference*, pages 419–428, York, England, Sept 1994. BMVA press.

[5] J. M. Coughlan and S. J. Ferreira. Finding deformable shapes using loopy belief propagation. In *European Conf. Computer Vision*, volume 3, pages 453–468. Springer-verlag, 2002.

[6] D. Cremers, T. Kohlberger, and C. Schnörr. Nonlinear shape statistics in mumford-shah based segmentation. In *European Conference on Computer Vision*, Copenhagen, May 28-31 2002. Spinger.

[7] A. Doucet and N. de Freitas. *Sequential Monte Carlo Methods in Practice.* Springer-Verlag, 2001.

[8] J. Foley, A. van Dam, S. Feiner, and J. Hughes. *Computer Graphics Principles and Practice*, chapter 13, pages 563–604. Addison-Wesley, 1990.

[9] N. J. Gordon, D. J. Salmond, and A. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEEE Proceedings on Radar and Signal Processing*, 140:107–113, 1993.

[10] M. Isard and A. Blake. Condensation: conditional density propagation for visual tracking. *Int. Journal Computer Vision*, 29(1):5–28, 1998.

[11] J. Jackson, A. J. Yezzi, and S. Soatto. Tracking deformable moving objects under severe occlusions. In *IEEE Conf. on Decision and Control*, 2004.

[12] A. Jain. *Fundamentals of Digital Image Processing*, chapter 9. Prentice-Hall, 1989.

[13] R. E. Kalman. A new approach to linear filtering and prediction problems. *Trans. of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[14] M. Kass, A. Witkins, and Terzopoulos. Snakes: active contour models. *Int. Journal Computer Vision*, 1(4):321–331, January 1988.

[15] F. Leymarie and M. D. Levine. Tracking deformable objects in the plane using an active contour model. *IEEE Trans. on Pattern Analysis Machine Intelligence*, 15(6):617–634, June 1993.

[16] P. Li, T. Zhang, and A. E. C. Pece. Visual contour tracking based on particle filters. *Image Vision Comput.*, 21(1):111–123, 2003.

[17] D. Metaxas and D. Terzopouilos. Shape and nonrigid motion estimation through physics-based synthesis. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 15(6):580–591, 1993.

[18] H. Nanda and L. Davis. Probabilistic template based pedestrian detection in infrared videos. In *IEEE Intelligent Vehicle Symposium*, Versailles, France, June 18-20, 2002.

[19] K. P. Nikos and D. Rachid. A pde-based level-set approach for detection and tracking of moving objects. In *Int. Conf. on Computer Vision*, page 1139. IEEE computer society, 1998.

[20] P. Pérez, J. Vermaak, and A. Blake. Data fusion for visual tracking with particles. 2004.

[21] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. on Pattern Analysis Machine Intelligence*, 12(7):629–639, jul 1990.

[22] P. J. Phillips, S. Sarkar, I. Robledo, P. Grother, and K. W. Bowyer. The gait identification challenge problem: Data sets and baseline algorithm. In *Intl. Conf. on Pattern Recognition*, 2002.

[23] F. Porikli and O. Tuzel. Bayesian background modeling for foreground detection. In *Proc. of ACM Visual Surveillance and Sensor Network*, 2005.

[24] Y. Rathi, N. Vaswani, A. Tannenbaum, and A. Yezzi. Particle filtering for geometric active contours with application to tracking moving and deforming objects. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 2–9, 2005.

[25] M. Rousson and N. Paragios. Shape priors for level set representations. In *European Conference on Computer Vision*, pages 78–92, 2002.

[26] G. Sapiro. *Geometric partial differential equations and image analysis.* Cambridge University Press, New York, NY, USA, 2001.

[27] M. Spengler and B. Schiele. Towards robust multi-cue integration for visual tracking. *Machine Vision and Application*, 14:50–58, 2003.

[28] G. Xu, E. Segawa, and S. Tsuji. A robust active contour model with insensitive parameters. In *Int. Conference on Computer Vision*, pages 562–566, 1993.

[29] A. Yezzi and S. Soatto. Deformation: Deforming motion, shape average and the joint registration and approximation of structures in images. *International Journal of Computer Vision*, 53(2):153–167, Febuary 2003.

[30] A. Yilmaz, X. Li, and M. Shah. Contour-based object tracking with occlusion handling in video acquired using mobile cameras. *IEEE Trans. on Pattern Analysis Machine Intelligence*, 26(11):1531–1536, Nov 2004.

[31] Q. Zheng and R. Chellappa. A computational vision approach to image registration. *IEEE Trans. Image Processing*, 2:311–326, 1993.

[32] S. Zhou, R. Chellappa, and B. Moghaddam. Visual tracking and recognition using appearance-adaptive models in particle filters. *IEEE Trans. Image Processing*, 13(11):1491–1506, November 2004.

[33] X. S. Zhou, D. Comaniciu, and A. Gupta. An information fusion framework for robust shape tracking. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(1):115–129, 2005.