SHAPE-REGULATED PARTICLE FILTERING FOR TRACKING NON-RIGID OBJECTS

Jie Shao, Rama Chellappa and Fatih Porikli[†]

Center for Automation Research and Department of ECE
University of Maryland
College Park, MD 20742
{shaojie,rama}@cfar.umd.edu

[†] Technology Lab
Mitsubishi Electric Research Lab
Cambridge, MA 02139
fatih@merl.com

ABSTRACT

This paper presents an active contour based algorithm for tracking non-rigid objects in heavily cluttered scenes. We decompose the non-rigid contour tracking problem into three subproblems: 2D motion estimation, deformation detection, and shape regulation. First, we employ a particle filter to estimate the affine transform parameters between successive frames. Second, by using a dynamic object model, we generate a probabilistic map of deformation to reshape its contour. Finally, we project the updated model onto a trained shape subspace to constrain deformations to be within possible object appearances. Our experiments show that the proposed algorithm significantly improves the performance of the tracker.

Index Terms— Contour tracking, particle filter, deformation, regulation

1. INTRODUCTION

Visual tracking is an essential component in intelligent robotics, video surveillance and medical imaging. Contour-based tracking approaches attract lots of interest due to their ability to accurately delineate contours or boundaries of objects. A parameterized B-spline contour tracking algorithm, CONDENSATION, was proposed by Isard and Blake [3], which used the particle filter as the basic framework. The CONDENSATION algorithm yields robust results when it is 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. Li *et al.* [5] discussed how to apply the particle filter to non-rigid object contour tracking. But the algorithm does not use an appropriate model for discriminating real boundary from all the edge points. Active 'Snakes' is another common approach that evolves an object boundary to minimize a weighted sum of external and internal energy terms [4]. However, snake based methods are restricted to a relatively small range of scenarios due to the fact that they rely on intensities inside objects to be relatively uniform and differenet from the background. Besides, their computational complexity makes them less suited for real-time applications. The level set approach is also a powerful method that deals with topological changes of the moving front using partial differential equations (PDE) that describe the object motion, boundary and region-based information [6, 8, 9, 11]. But they also suffer from expensive computation.

Compared with existing algorithms, our work emphasizes non-rigid object contour tracking with many important features, including accuracy and robustness. We claim that although rigid and non-rigid objects both involve translations of pixels inside an object boundary, tracking non-rigid objects cannot be accomplished by a simple rigid object motion model without compensating for local deformations. A model that captures both translational and non-translational motions of objects is needed. In addition, the model should include a discriminating measure to extract the real boundary from all the edge points, the majority of which may be from the background. It is also necessary to constrain the deformation of the shape model for ensuring robustness.

Based on the above observations, we decompose the non-rigid object tracking problem into three components: (1) **2D motion estimation** that yields object-wise spatial rigid-body motion, including translation and rotation parameters. We assume that the 2D shape of an object does not drastically change between two successive frames. Thus, we use a particle filter based on an affine model to estimate the motion transition; (2) **2D shape deformation** that refines the pose changes of non-rigid objects. Each pixel on the boundary may have different but correlated deformations. We construct a posterior deformation probability map based on statistical analysis of the corresponding frame; and (3) **Shape regulation** that uses a trained shape subspace to restrict shape deformations. Regulation also reconstructs the occluded parts of the contours.

The rest of the paper is organized as follows. In section 2, we introduce the active contour tracking algorithm. In the next three sections, we address the non-rigid object shape tracking algorithm, focusing on the three components listed above. We present some experimental results in section 6, followed by conclusions and discussions in section 7.

2. OBJECT MODEL AND PARTICLE FILTER

A contour tracking algorithm requires an object representation and a tracking strategy. We represent the 2D curves outlining the objects in terms of parametric B-spline curves. The advantage of using B-spline is that when we define the dynamic model of a curve as an affine transformation, it is sufficient to apply the transform to the control points on the curve, which improves computational efficiency.

We use the particle filter algorithm as our tracking strategy. The basic idea is: Let θ denote the state vector, Y the observations, $\mathcal{L}(Y_t|\theta_t)$ the likelihood function at time instant t. The problem of tracking can be formulated as maximizing the posterior:

$$p(\theta_t|Y_{t-1}) \propto \mathcal{L}(Y_t|\theta_t) \int p(\theta_t|\theta_{t-1}) p(\theta_{t-1}|Y_{1:t-1}) d\theta_{t-1} \quad (1)$$

with the first-order Markovian assumption $p(\theta_t|\theta_{1:t-1}) = p(\theta_t|\theta_{t-1})$.

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

3. STAGE I: MOTION TRANSITION

3.1. Dynamic Transition Model

We use a 2D affine transform model to represent the global transform of the object, with the knowledge that the B-spline curve is invariant to the affine transform. The transform of the active contours is given in terms of control points as:

$$\begin{bmatrix} \mathrm{r}_t(s) \\ 1 \end{bmatrix} = \mathcal{T} \cdot \begin{bmatrix} \mathrm{r}_{t-1}(s) \\ 1 \end{bmatrix}, \qquad (2)$$

which is still an affine transform denoted by $\mathcal{T}$. $\mathrm{r} = (x, y)$ refers to the x,y coordinates of the control point, s is the B-spline curve parameter. The parameters in the affine matrix compose the state vector of the particle filter:

$$\theta_t = (\mathcal{T}_{11}\ \mathcal{T}_{12}\ \mathcal{T}_{21}\ \mathcal{T}_{22}\ \mathcal{T}_{13}\ \mathcal{T}_{23})^T. \qquad (3)$$

We update the proposal distribution using the state transition model as $\theta_t = \tilde{\theta}_{t-1} + \nu_t + U_t$ with ν_t as a shift in the motion vector and U_t as the driving noise, assumed to be Gaussian. As a result, the system does not need as many particles as in the original particle filter algorithm without sacrificing the performance. The estimation of ν_t is based on the assumption of brightness invariance [13].

3.2. Observation Model

We follow the traditional strategy to estimate the observation model: At each control point $\mathrm{r}(s_l), (l = 1, \ldots, M)$, we search along the crossing normal lines $\mathbf{n}_l$ for feature points. It is expected that more than one feature $z_j^{(l)}, (j = 1, \ldots, n_l)$ is detected, due to background clutter. Assuming that clutter $z_j^{(l)}$ can be modeled as a spatial Poisson distribution along the normal lines and the true target measurement is a Gaussian, the 1-D measurement density along the line normal to $\mathrm{r}(s_l)$ can be modeled as [3]:

$$P_l(z|r = \mathrm{r}(s_l)) \propto 1 + \frac{1}{\sqrt{2\pi}\sigma q \lambda} \sum_{j=1}^{n_l} \exp\left(-\frac{(z_j^{(l)} - \mathrm{r}(s_l))^2}{2\sigma^2}\right) \quad (4)$$

where n_l is the number of features detected along the normal, q is the probability of invisible target, λ is the spatial density of the Poisson distribution, σ is the standard deviation of the Gaussian distribution. With the assumption that the feature outputs on distinct normal lines are statistically independent, the overall likelihood in Eq. (1) becomes:

$$p(Y|\theta) = p(\mathrm{Z}|\theta) = \prod_{l=1}^{M} p_l(z|r = \mathrm{r}(s_l)) \qquad (5)$$

We select the sample that maximizes the posterior in Eq. (1) as the current estimate. In the original active contour algorithm, the exact contour points of the tracking object are selected as feature points with maximum gradient magnitudes from all the feature points detected on each of the normal lines. Unfortunately, this strategy does not always work, especially when the background is heavily cluttered or the object undergoes shape deformations between frames. Identifying the correct feature points along the normal lines becomes a challenge. Our algorithm uses a distinct shape analysis step to capture the accurate contour of the object.

4. STAGE II: CONTOUR DEFORMATION

4.1. Adaptive Normal Line Scanning

In the deformation step, we set the normal lines adaptive in the following two aspects.

Adaptive line length The lengths of the normal lines, denoted as $u(l)$, influence tracking results: a normal line with a short length may miss the true boundary pixel while a normal line with a longer length may intersect with edge points from background clutter. To avoid false detections, the length of the normal line is altered according to the pose variances of the corresponding contour control points throughout the video sequence. 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 legs and arms change their relative positions more. Therefore, the lengths of the normal lines with large pose variances are set larger than those of small pose variances. The pose variances of pixels can be obtained offline during training:

$$\begin{aligned} \xi(l) &= \mathbf{E}_k \|\mathrm{r}_k(s_l) - \mathbf{E}_k(\mathrm{r}_k(s_l))\|^2 \\ u(l) &\propto L_{min} \log \frac{\xi(l)}{\min(\xi(l))} \end{aligned}$$

where ξ is the pose variance, L_{min} is a constant term for the minimum length, and k denotes the index of training samples.

Adaptive line center The original algorithm sets the centers of the scanning normal lines as control points on the estimated contour, which may cause the normal lines to cross each other or even cross the other side of the contour when the distance between the two sides of the contour is small, which results in contour 'looping'. Making the line centers adaptive by applying a distance transform (DT), reduces the probability of crossing. Denote $\tilde{\mathbf{r}} = \mathrm{r}(s_l), 1 \leq l \leq M$, the estimated contour from stage I. The steps for accomplishing this are listed as follows:

- Based on the estimated contour, construct a binary image $\mathcal{BI}$ by setting the region Ω circled by $\tilde{\mathbf{r}}_t$ to 1 and the rest to 0;
- Apply DT to $\mathcal{BI}$, obtain a distance map $\mathcal{DI}$, with each pixel $\mathbf{x} \in \Omega$ having a value of minimum distance to the region boundary;
- On both sides of each support point $\tilde{\mathrm{r}}(s_l)$, draw a normal line $\mathbf{n}_l$ with fixed initial length, find maximum distance value satisfying $\mathcal{DI}(l)_{max} = max_{\mathbf{x} \in \mathbf{n}_l} \mathcal{DI}_{\mathbf{x}}$. Denoting the side containing the pixel with maximum distance by $\aleph$, and the other side by $\bar{\aleph}$, the lengths of two sides are set as follows:

$$\begin{aligned} u(l)_{\aleph} &= \mathcal{DI}(l)_{max} - d_0 \\ u(l)_{\bar{\aleph}} &= 2u(l) - [\mathcal{DI}(l)_{max} - d_0] \end{aligned}$$

where d_0 represents a minimum safe distance to avoid intertwining contours. We set d_0 to 2 in our experiments.

Fig.1 (A) demonstrates the procedure of sketching adaptive normal lines along the estimated contour, with which we are able to search for the real contour pixels using different relative pose variances of the contour points.

4.2. Statistical Analysis with Multi-cues

Instead of focusing on only the magnitudes of the feature points as measurements to determine whether the feature points belong to the real contour or not, we use a statistical approach to extract the real contour pixels. A deformation probability map P, in which a high probability implies that the pixel is more likely to be on the real contour and a low value implies a lower likelihood, is generated using several cues.

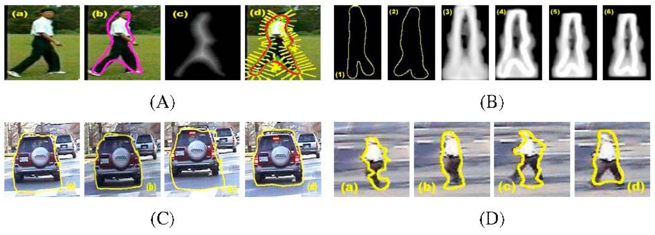

Fig. 1. ***(A)*** *An example of how to make a normal line scanning adaptive is shown. (a) cropped original object; (b) estimated contour; (c) distance transformed object; (d) normal lines on support pixels;* ***(B)*** *Illustrations of adaptive probability shape templates for a pedestrian. (1),(2): shape training samples; (3): shape prior model; (4),(5),(6): examples of probability shape templates in different frames;* ***(C)*** *Performance comparisons in a cluttered scene. (a) and (c): contour tracker using the normal edge map; (b) and (d): contour tracker using the feature maps F defined in Eq.(7). A stabilization step is applied to obtain F because the sequence is from a non-stationary camera;* ***(D)*** *Comparisons between tracking results with and without subspace regulation. (a) and (c) are without regulation, (b) and (d) are with regulation.*

4.2.1. P_s: *Probabilistic Shape Template*

A statistical shape template P_s is constructed using the shape prior model $\mathcal{P}_m$ as a static template created from the training data, and the dynamic template derived from the current estimated contour $\mathcal{P}_{\tilde{\mathbf{r}}}$. P_s assigns a probability to each possible deformation of the shape, and is updated in each frame. The probability is given as:

$$P_{s,t} = a_t \mathcal{P}_m + (1 - a_t)(P_{s,t-1} + \delta P_{s,t}) = a_t \mathcal{P}_m + (1 - a_t)\mathcal{P}_{\tilde{\mathbf{r}},t} \tag{6}$$

where $0 < a_t < 1$ is the integrating weight that controls the relative contributions from $\mathcal{P}_m$ and $\mathcal{P}_{\tilde{\mathbf{r}},t}$, $\delta P_{s,t}$ is the shift of the dynamic template from $t-1$ to t. The probabilistic template is a shape-based cue accounting for variations in deformation, i.e., indicating the probability that each image pixel belongs to the real object contour. In Fig.1 (B), we give some examples of the probability templates.

4.2.2. P_m: *Probability Map of Gradient Magnitude*

Edges are important features characterizing the object boundaries. However, as the object itself is not homogenous in color or intensity, edges are also present due to textures. We want to avoid distractions from such inside edges. We use anisotropic diffusion to make the entire image appear to be more uniform in color or texture, while still preserving the object boundaries [7]. The points with high gradient magnitudes after diffusion are more likely to be from the boundary of the object. After the edge map $\mathcal{EI}$ is extracted from the original image using diffusion, a motion mask $\triangle\mathcal{I}$ indicating the possible area where motion could occur is convolved with $\mathcal{EI}$ to further suppress the background clutter. Finally, the feature map $\mathcal{FI}$ is derived as:

$$\mathcal{FI} = \frac{\mathcal{EI} * \triangle\mathcal{I}}{\max(\triangle\mathcal{I})} \Rightarrow F = |\mathbf{G} * \mathcal{FI}| \Rightarrow P_{m,t} = \frac{F_t}{\max(F_t)} \tag{7}$$

where $\mathbf{G}$ is a smoothing Gaussian filter and F is the magnitude map of the diffused image gradient, mapped into a magnitude probability map P_m. Fig.1 (C) demonstrates an example of performance improvement when we utilize the diffusion edge map convolved with the motion mask. A background stabilizing step [12] is applied before the motion mask is estimated when the background and the tracked object are both moving.

4.2.3. P_o: *Probability Map of Gradient Orientation*

Theoretically, an edge map gives local evidence for the presence or absence of object boundaries at each pixel, while an orientation map $\mathcal{OI}$ provides the orientations of edges. Orientation is one 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 orientation should roughly equal to either ϕ or $\phi \pm \pi$ [2], yet the orientation distribution of pixels off the boundary tends to be a uniform distribution between $[0, 2\pi)$. This leads to the orientation probability map to be:

$$P_{o,t}(\mathbf{x}) \propto \exp(-\frac{(\mathcal{OI}_t(\mathbf{x}) - \tilde{\phi}_t(l))^2}{\sigma_o^2}) \quad \forall \mathbf{x} \in \mathbf{n}_l \tag{8}$$

where we assume $\tilde{\phi}_t(l) = \phi_{t-1}(l)$.

4.2.4. *Fusion*

All the probability maps generated from multiple cues are not strictly independent, but indirectly coupled by the result they agree upon [10]. Thus, the deformation probability is approximated as:

$$\begin{aligned} P_t(Y|\tilde{\mathbf{r}}) = P_t(s, m, o|\tilde{\mathbf{r}}) &\propto P_t(s|\tilde{\mathbf{r}})P_t(m|\tilde{\mathbf{r}})P_t(o|\tilde{\mathbf{r}}) \\ &\approx P_{s,t} \cdot P_{m,t} \cdot P_{o,t} \end{aligned} \tag{9}$$

Scanning for pixels with maximum probability values on adaptive normal lines, we obtain the refined contour pixels, denoted as $\tilde{\mathbf{r}}$.

5. STAGE III: SHAPE REGULATION

Shape regulation shares equal importance with other processes that decide the performance of the contour tracker, especially when the target is non-rigid and a deformation has been estimated. We use shape subspace to constrain the deformation. There are several approaches for subspace construction. Our algorithm follows the subspace model introduced by Cootes *et al.*, the active shape model (ASM) [1]. The shape subspace is trained from a set of L samples, each of which is represented by a set of columnized M control points $\{\mathrm{r}_i^j; 1 \leq i \leq M, 1 \leq j \leq L\}$, we finally obtain the following subspace representation $\mathrm{x} = \bar{\mathrm{x}} + \mathbf{Pb}$, where x_i is aligned and columnized r_i, $\bar{\mathrm{x}}$ is the mean of the aligned examples, $\mathbf{b}$ is a t-element vector of shape variation parameters and $\mathbf{P}$ is a $(2M \times t)$ matrix of t eigenvectors, which composes the estimated shape subspace Φ_s. The deformed contour is fitted into the subspace. The whole operation can be described as interpreting the deformed contour using a linear combination of shape subspace basis. Another advantage of using the subspace is that it can partially solve the occlusion problems. Denoting the detected contour point vector set as $\{\tilde{\mathbf{r}} : \mathrm{r}_i, 1 \leq i \leq M\}$, [1] we first normalize $\tilde{\mathbf{r}}$ to $\tilde{\mathrm{r}}_n$ and, then apply:

$$\tilde{\mathbf{r}}_{proj} = \mathbf{PP}^T(\tilde{\mathbf{r}}_n - \bar{\mathrm{x}}) + \bar{\mathrm{x}} \tag{10}$$

where $\tilde{\mathbf{r}}_{proj}$ is a linear combination of subspace basis. In surveillance sequences, it is highly possible that some points in $\tilde{\mathbf{r}}$ are occluded, or not detected along the normal lines. Let us denote the

[1]For brevity, here $\tilde{\mathbf{r}}$ represents the columnized contour point vector.

index set of detected points as $I_d = \{i_1, i_2, \ldots\}$. We can recover a complete reprojected contour as follows:

$$\tilde{\mathbf{r}}_{proj} = \mathbf{P}\mathbf{P}^{\dagger}_{I_d}(\tilde{\mathbf{r}}_{n,I_d} - \bar{\mathrm{x}}_{I_d}) + \bar{\mathrm{x}} \quad (11)$$

$$\mathbf{P}^{\dagger}_{I_d} = (\mathbf{P}^T_{I_d}\mathbf{P}_{I_d})^{-1}\mathbf{P}^T_{I_d} \quad (12)$$

We give an example in Fig.1 (D) with a comparison between tracking results with and without shape regulation to demonstrate the performance improvement of the tracker after we apply shape regulation.

6. EXPERIMENTS

We have applied the three-stage contour tracking algorithm to different sets of outdoor surveillance video sequences, containing moving humans and vehicles. The algorithm speed implemented by C++ code are 6-10 fps (frames per second) respectively on a 1.5GHz PC running Windows XP. The tracking results are shown in Fig. 2 and 3. Fig.2 shows two challenging sequences with difficulties due to moving camera and cluttered background. The result of our tracker is satisfactory. Fig.3 gives a good example of how shape regulation helps to recover the occluded contour. From the experimental results, we find that the tracker performs well in most of the cases.

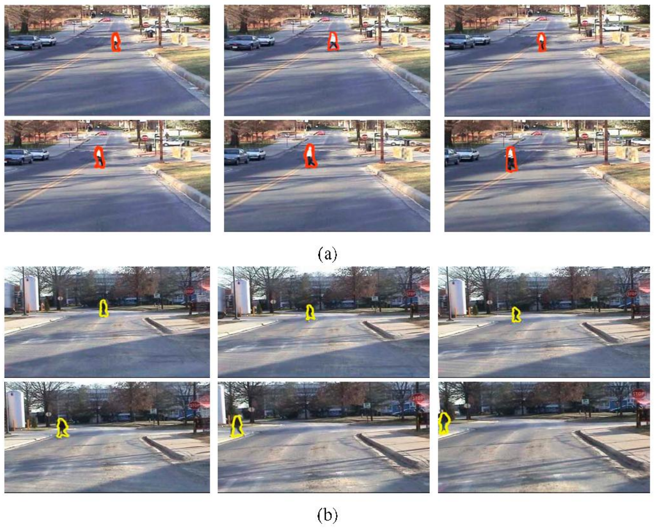

Fig. 2. *Two sequences with both background and object moving.*

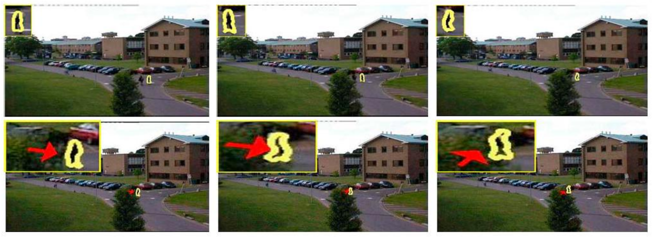

Fig. 3. *An example of occluded contour being recovered by shape subspace reprojection. In the last 3 frames, the pedestrian is partly occluded by surrounding trees. Our tracking result recovers the partially occluded contour. Red arrows indicate the occlusion parts. We may also notice that parking cars contributes to background clutters.*

7. DISCUSSION

The contour tracker we have presented is based on active contour tracking algorithm using the particle filter, but enhanced using three key steps. The three steps are motion transition, shape deformation and shape regulation, respectively. Compared to existing algorithms, our algorithm is more applicable to scenarios with non-rigid object movements and heavily-cluttered backgrounds. The merits of our algorithm are: (1) We model the non-rigid object movement as a concatenation of global object transition, local boundary deformation and constrained shape subspace representation; (2) The scanning normal lines are adaptive, more flexible to shape pose variations and prohibit contour 'looping' as well; (3) We generate a posterior deformation probability map to extract precise outlines of non-rigid objects; and (4) We use a shape subspace to restrict shape deformation and recover occluded boundary pixels which significantly reduces the risks of over-deformation. The method can successfully track non-rigid objects in real time and get tight contours enclosing the changing shapes of the targets throughout the sequences. We are further considering extension of the algorithm to the multi-target tracking problem.

References

[1] 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.

[2] 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.

[3] M. Isard and A. Blake. Contour tracking by stocastic propagation of conditional density. In *European Conference on Computer Vision*, pages 343–356, Cambridge, England, 1996.

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

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

[6] 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.

[7] 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.

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

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

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

[11] 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.

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

[13] 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.