REAL TIME OBJECT TRACKING

Andrei Ingeaua, Dumitru Ingeaua, Stefan Udristoiu

University of Craiova

Abstract: This article in intended as a short reference for some of the techniques that can be used in object tracking. Object tracking has many application in various fields starting from Mechatronics, military applications, etc. A real interes for us is tracking the object in real time because of it's intensive use in visual servoing. This article will present some methods and algorithms for such an operation starting with HMM (Hidden Markov Model), and continues presenting Viterbi algorithm, computing transitions probalities and finishes with improving transitions probabilities techniques such encoding regions smoothness constraint using JPM and efficient matching by dinamic programming.

1. INTRODUCTION

Reliable object tracking in complex visual environment is of great importance. In addition to its applications in human computer interaction (M. Isard and A. Blake, 1996, 1998) . As pointed in (Yong Rui, Thomas S. Huang, and Sharad Mehrotra , 1999), a unstructured video clip can be organized into key frames, shots and scenes. If we can reliably track an object, it can help detect then if the object is of interest (e.g., bigger size), and can therefore extract more meaningful key frames. Also, object tracking can provide useful information for shot boundary detection.

To start object tracking, usually the trackers need to be initialized by an external module. For example, a human operator can select an object of interest and let the tracking begin. For a more intelligent tracking system, an automatic object detection module can be used to initialize the tracking. Automatic object detection algorithms are usually trained based on a set of images of typical object appearances at different states and viewed from different angles, e.g. the multi-view face detector or hand posture recognition.

Once initiated, the tracking algorithms will conduct tracking based on the high correlations of the object motion, shape or appearance between consecutive video frames. Unfortunately, robust and efficient object tracking is still an open research problem. Two of the major challenges are:

1. Visual measurements for tracking objects are not always reliable. To discriminate objects from background clutter, various image cues have been proposed. Object contour, face template or color distributions are used in (A.J. Colmenzrez et all, 1997 D. Comaniciu et all, 2000, M. Isard et all, 1998, Y. Wu et all, 2000) respectively. In complex environments, none of the above features are robust enough individually. More and more researchers are therefore resorting to multiple visual cues. The major difficulty for multi cue object tracking is, however, how to effectively integrate the cues in a principled way.

2. Tracking objects in nonlinear dynamic systems is not easy. While we know that Kalman filter provides an elegant solution for linear systems, in the real world, the states of the objects are usually nonlinearly related to the measurements through observation models. For example, the contour points of an ellipsoid object are nonlinearly related to the object's position and orientation.

The Hidden Markov Model (HMM) (L.R. Rabiner et all., 1986) provides a potential tool to solve the first difficulty. It can integrate multiple visual cues by expanding the observation vectors and encode the spatial constraints in the state transition probabilities. Optimal contour can be obtained by the efficient

Viterbi algorithm. However, extending the HMM structure from 1D time series to 2D imagery data is challenging. A pseudo-2D-HMM (embedded HMM) has been proposed for character recognition, face recognition, and template matching. A two-level HMM is defined, where super states are used to model the horizontal dimension and embedded states are used to model the vertical dimension. However, this approach requires a large number of parameters to be trained. Instead, we propose a new type of HMM that can probabilistically integrate multiple cues under various spatial constraints, including global shape prior, contour smoothness constraint, and region smoothness constraint. Parametric shape is used to model object contour. Multiple visual cues (e.g., edge and color) are collected along the normal lines of the predicted contour (see Fig. 1(a)). We define the HMM states to be the contour point location on each normal line. This representation allows us to formulate the 2D (in image plane) contour tracking into an easier-to-solve 1D problem.

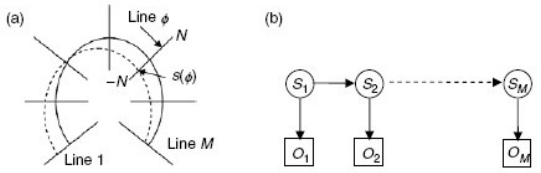

Fig. 1. The new contour model: (a) The contour in 2D image space: The solid curve is the predicted contour. The dashed curve is the true contour. We want to find the s(φ) which is the index of the true contour point on the φth normal line φ∈ [1,M]; (b) the Markovian assumption in our contour model. Note the HMM states are in the spatial domain

2. CONTUR TRACKING USING HMM

For tracking non-rigid objects, active contour models have been proved to be powerful tools (M. Kasset all 19880, D. Terzopoulos et all, 1992). In traditional active contour methods, the optimization procedure is not very efficient due to the recursive contour refinement procedure [A.A. et all., 1990, J. Denzler et all., 1999). Considering the aperture effect, where only the deformations along the normal lines of the contour can be detected, we can restrict the contour searching to a set of normal lines only (see Fig. 1(a)). In this way, we convert the 2D searching problem into a simpler 1D problem. To define the 1D contour model, let $\varphi = 1, ..., M,$ be the index of the normal lines and $\lambda = -N, ..., N,$ be the index of pixels along a normal line and $\rho_\phi(\lambda)$ denote the image intensity or color at pixel λ on line φ:

$$\rho_\phi(\lambda) = I(x_{\lambda_\varphi}, y_{\lambda_\varphi}) \quad (1)$$

where $(x_{\lambda_\varphi}, y_{\lambda_\varphi})$ is the corresponding image coordinate of the pixel λ on the φ-th normal line. $I(x_{\lambda_\varphi}, y_{\lambda_\varphi})$ is the image intensity or color at $(x_{\lambda_\varphi}, y_{\lambda_\varphi})$.

Each normal line has $2N + 1$ pixels, which are indexed from $-N$ to N. The center of each normal line is placed on the predicted contour position and indexed as 0. If the object had moved exactly as predicted, the detected contour points on all normal lines would have been at the center, i.e., $s(\varphi) = 0, \forall \varphi \in [1, M]$. In reality, however, the object can change its motion and we need to find the true contour point $s(\varphi)$ based on the pixel intensities and various spatial constraints. Note that instead of representing the contour by a 2D image coordinate, we can now represent the contour by a 1D function s(φ), φ = 1, ...,M.

To detect the contour points accurately, different cues (e.g., edge and color) and prior constraints (e.g., contour smoothness constraint) can be integrated by an HMM. The hidden states of the HMM are the true contour points on all the normal lines, denoted as $s = \{s_1, ..., s_\varphi, ...s_M\}$. The observations of the HMM, $O = \{O_1, ..., O_\varphi, ...O_M\}$, are collected along all the normal lines. An HMM is specified by the observation model $P(O_\varphi \mid s_\varphi)$ and the transition probability $p(s_\varphi \mid s_{\varphi-1})$. Given current state s_φ, the current observation O_φ, is independent of the previous state $s_{\varphi-1}$ and the previous observation $O_{\varphi-1}$. Because of the Markovian property, we have $p(s_\varphi|s_1, s_2, ..., s_{\varphi-1}) = p(s_\varphi|s_{\varphi-1})$, which is illustrated in Fig.1(b).

2.1 Observation Likelihood of Multiple Cues

In the HMM, the observation on line φ (represented as O_φ) can include multiple cues. We describe the observation model based on color (i.e., $\rho_\varphi(\lambda)$, $\lambda \in [-N,N]$) and edge detection (i.e., $\mathbf{z}_\varphi$) along the line in this section.

First, the observation likelihood based on the edge detection ($\mathbf{z}_\varphi$) can be derived similar to (M. Isard et all.,1998). Because of noise and image clutter, there can be multiple edges along each normal line. Let J be the number of detected edges, we have

$$\mathbf{z}_\varphi = (z_1, z_2, ..., z_J).$$

Of the J edges, at most one is the true contour. We can therefore define J + 1 hypotheses:

$$H_0 = \{e_j = F : j = 1, ..., J\}$$
$$H_j = \{e_j = T, e_k = F : k = 1, ..., J, k \neq j\} \quad (2)$$

where e_j = T means that the j^{th} edge is the true contour, and e_j = F otherwise. Hypothesis H_0 therefore means the true contour is not detected by the edge detection. With the assumption that the clutter is a Poisson process along the line with spatial density γ and the true target measurement is normally distributed with standard deviation σ_z, we can obtain the edge likelihood model as follows:

$$p(\mathbf{z}_\varphi|s_\varphi = \lambda_\varphi) \propto 1 + \frac{1}{\sqrt{2\pi}\sigma_z q\lambda}\sum_{m=1}^{J}\exp(-\frac{(z_m - \lambda_\varphi)^2}{2\sigma_z^2}) \quad (3)$$

where q is the prior probability of hypothesis H_0. A typical edge-based observation along one normal line is shown in Fig. 2. Multiple peaks appear due to clutter.

To reduce the clutter, HMM can easily integrate other cues, such as color histogram of the foreground (FG) and background (BG). Let v be the color, $p(v|FG)$ and $p(v|BG)$ represent the color distribution for the FG and BG respectively. If $s\varphi = \lambda\varphi$ is the contour point on line φ, we know that the segment $[-N, s\varphi]$ of line φ is on the FG and the segment $[s\varphi + 1, N]$ is on the BG. Combining the edge likelihood model and the color histogram of the FG/BG, we have the following multicue observation likelihood model:

$$P(O_\varphi|s_\varphi) = p(z_\varphi|s_\varphi) \cdot \prod_{i=-N}^{s_\varphi} P(v = \rho_\varphi(i)|FG) \cdot \prod_{i=s_\varphi+1}^{N} P(v = \rho_\varphi(i)|BG) \quad (4)$$

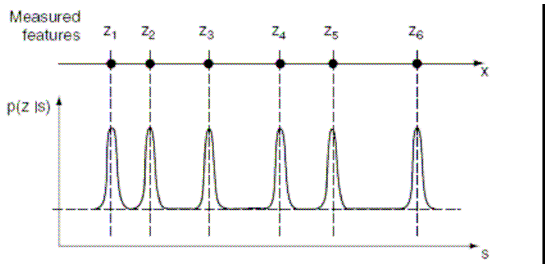

Fig. 2. One-dimensional edge-based observation $p(z_\varphi|s_\varphi)$: z_1, z_2, ..., z_J are the detected edges along one normal line and cause a multiple peak edge-based likelihood model

2.2 Computing Transition Probabilities

In addition to the observation model discussed in the previous subsection, another important component in the HMM is the transition probability. It determines how one state transits to another. In this subsection, we use the standard contour smoothness constraint to derive the transition probability.

The contour smoothness constraint can be encoded in transition probability. To enforce the contour smoothness constraint in the HMM, the constraint needs to be represented in a causal form. In Fig. 1(a), we can see that when the normal lines are dense (30 lines in our experiments), the true contour points on adjacent normal lines tend to have similar amounts of displacement from the predicted contour position (indexed as 0 on each normal line). This constraint is causal and can be captured by transition probabilities $p(s_\varphi|s_{\varphi-1})$ defined as follows:

$$p(s_\varphi|s_{\varphi-1}) = c \cdot e^{-(s_\varphi - s_{\varphi-1})^2/\sigma_s^2} \quad (5)$$

where c is a normalization constant and σ_s regulates the smoothness of the contour. This transition probability penalizes sudden changes of the adjacent contour points, resulting in a smoother contour.

2.3 Best Contour Searching by Viterbi Algorithm

Given the observation sequence $O = \{O\varphi, \varphi \in [1,M]\}$ and the transition probabilities $a_{i,j} = p(s_{\varphi+1} = j|s_\varphi = i)$, the best contour can be found by finding the most likely state sequence $\boldsymbol{s}^*$. This can be efficiently accomplished by the Viterbi algorithm (L.R. Rabiner et all., 1986):

$$\mathbf{s}^* = \arg\max_{\mathbf{s}} P(\mathbf{s}|\mathbf{O}) = \arg\max_{\mathbf{s}} P(\mathbf{s}, \mathbf{O}) \quad (6)$$

Let us define

$$V(\varphi, \lambda) = \max_{s_{\varphi-1}} P(O_\varphi, s_{\varphi-1}, s_\varphi = \lambda) \quad (7)$$

Based on the Markovian assumption, it can be recursively computed as follows:

$$V(\phi, \lambda) = P(O_\phi|s_\phi = \lambda) \cdot \max_j P(s_\phi = \lambda|s_{\phi-1} = j)V(j, \phi - 1) \quad (8)$$

$$j^*(\phi, \lambda) = P(O_\phi|s_\phi = \lambda) \cdot \arg\max_j P(s_\phi = \lambda|s_{\phi-1} = j)V(j, \phi - 1) \quad (9)$$

with the initialization $V(1, \lambda) = \max_{s1} P(O_1|s_1)P(s_1)$, where the initial state probabilities

$P(s_1) = \frac{1}{2N-1}$, $s1 \in [-N,N]$.

The term $j^*(\varphi, \lambda)$ records the "best previous state" from state λ at line φ. We therefore obtain at the end of the sequence $\max_{\mathbf{s}} P(\mathbf{O}, \mathbf{s}) = \max_\lambda V(M,\lambda)$. The optimal state sequence $\mathbf{s}^*$ can be obtained by back tracking j^*, starting from $s_M^* = \arg\max_\lambda V(M,\lambda)$, with $s_{\varphi-1}^* = j^*(s_\varphi^*, \varphi)$. The computational cost of the Viterbi algorithm is $O(M \cdot (2N+1))$. Unlike traditional active contour model (A.A. Amini et all., 1990, J. Denzler et all, 1999), this method can give us the optimal contour without recursively searching the 2D image plane. The best state sequence $\mathbf{s}^* = \{ s_1^*$,

..., s_M^* } will be used in the Unscented Kalman Filter (UKF) to calculate the innovations and estimate the contour parameter.

3 IMPROVING TRANSITION PROBABILITIES

The transition probability is one of the most important components in an HMM. It encodes the spatial constraints between the neighboring contour points. In Sect. 2.2, we derive a simplified way of computing transition probabilities based on the contour smoothness constraint. Even though simple, it only considers the contour points themselves and ignores all the other pixels on the normal lines, which can be dangerous especially when the clutter also has smooth contour and is close to the tracked objects (e.g., the dark rectangle in Fig. 3(a)). To estimate the contour transition robustly, we should consider all detected edges jointly similar to the Joint probability data association (JPDAF) in (Y. Bar-Shalom et all., 1988). We introduce joint probabilistic matching (JPM) into the HMM for calculating more accurate transition probabilities. With contexture information, the new transition probabilities are more robust. An efficient optimization algorithm based on dynamic programming is developed to calculate the JPM term in real time.

3.1 Encoding Region Smoothness Constraint Using JPM

Since the true contour can be any pixel on the normal line, we have to estimate the transition between the pixels on the neighboring normal lines. Let s_φ and $s_{\varphi+1}$ be the contour points on line φ and line $\varphi + 1$, respectively. These two contour points segment the two lines into foreground (FG) segments and background (BG) segments. If the object is opaque, the edges on FG cannot be matched to BG. To exploit this constraint, we need to track the transitions of all the pixels on FG/BG together. That is, it is not a matching of contour points only, but rather a matching of the whole neighboring normal lines. The transition probabilities based on this new matching paradigm enforce not only the contour smoothness but also region smoothness constraint and are therefore more accurate and robust to clutter.

Let $E^F(i, j)$ and $E^B(i, j)$ be the matching errors of the neighboring foreground segments (i.e., segment [−N, i] on line φ and [−N, j] on line φ + 1) and background segments (i.e., segment [*i*+1,*N*] on line φ and [j +1,N] on line φ + 1), respectively. Let δ(i) = j specify that pixel i on line φ should be matched to pixel j on line φ + 1 and δ() is monotonic (i.e., δ(i − 1) <= δ(i)). Then, the matching cost is defined as:

$$E^F(i, j) = \min_\delta \sum_{k=-N}^{i} \|\rho_\varphi(k) - \rho_{\varphi+1}(\delta(k))\|_2, \quad \delta(k) \in [-N, j] \tag{10}$$

$$EB(i, j) = \min_\delta \sum_{k=i+1}^{N} \|\rho_\varphi(k) - \rho_{\varphi+1}(\delta(k))\|_2, \quad \delta(k) \in [j + 1, N] \tag{11}$$

A more accurate transition probability can then be estimated based on the matching cost (compare with Eq. (5)):

$$\log(p(s_2|s_1)) = E^F(s_1, s_2) + E^B(s_1, s_2) + (s_2 - s_1)^2 / \sigma_s^2 \tag{12}$$

The importance of the new matching cost can be illustrated by a synthesized image in Fig. 3. There are two regions where the grey region is the object to track and the darker rectangle is a background object. There are two adjacent normal lines shown in the figure, i.e., line 1 and line 2. Points '*a*' and '*b*' are detected edge points on line 1. Similarly, points '*c*' and '*d*' are detected edge points on line 2. Our goal is to find the true contour points on these two normal lines. The pixel intensities along these two lines are shown in Fig. 3(b). They are similar to each other except for some distortions. Based on the contour smoothness constraint only, the contour from '*a*' to '*c*' and the contour from '*b*' to '*c*' have almost the same transition probabilities because

$$|a - c| \approx |b - c|.$$

However, if we consider all the pixels on the normal lines together, we can see that '*ac*' is not a good contour candidate because '*b*' and '*d*' are now on foreground and background respectively and they have no matching on the neighboring lines. The contour candidates '*ad*' and '*bc*' are better because they segment the two normal lines into matching FG/BG (The best choice between these two should be further decided based on Viterbi algorithm in 2.3.)

The comparison between traditional smoothness constraint and JPM based smoothness constraint is shown in Fig. 3(c) and (d). Without joint matching terms, the contour is distracted by the strong edge of the background clutter in Fig. 3(c). In Fig. 3(d), the matching cost has large penalty for the contour to jump to background clutter and then jump back. Hence we obtain the correct contour.

Unlike the uniform statistic region model in, our matching term is more relaxed. The object can have multiple regions (e.g., the side view of human head with face and hair as in the experiments), each of which has continuous boundaries. To illustrate this, another test is shown in Fig. 4 which has different intensity regions in the foreground. We can see the difference between Figs. 3 and 4: the observations on line 2 are not the same. There is no segment '*cd*'. No matter we match '*c*' to '*a*' or '*b*', the other edge will have no matching part. Therefore, the matching cost

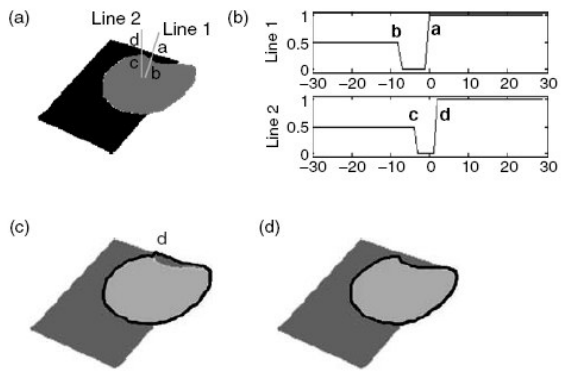

Fig. 3. Illustration of the JPM: (a) Synthesized image with the grey object to track.(b) The observation on normal line 1 and 2. (c) Based on traditional contour smoothnessonly, the detection is distracted by strong continuous edges on the background.(d) With the JPM, the contour is correctly detected

is the same for matching '*a*' to '*c*' or '*b*' to '*c*'. The algorithm favors the result in Fig. 4(b) because it is smoother.

3.2 Efficient Matching by Dynamic Programming

To ensure real-time performance, we propose an efficient algorithm to calculate the JPM. There are $(2N + 1)^2$ possible state transitions between the neighboring normal lines. We propose an efficient dynamic programming algorithm to calculate all $(2N + 1)^2$ matching probabilities with $2 \cdot (2N + 1)^2$ computational cost.

Given observation on lines 1 and 2, the calculation of the matching probabilities can be explained in the following recursive equation:

$$E^F(i, j) = \min(E^F(i-1, j) + d, E^F(i, j-1) + d, E^F(i-1, j-1)) + e(i, j) \quad (13)$$

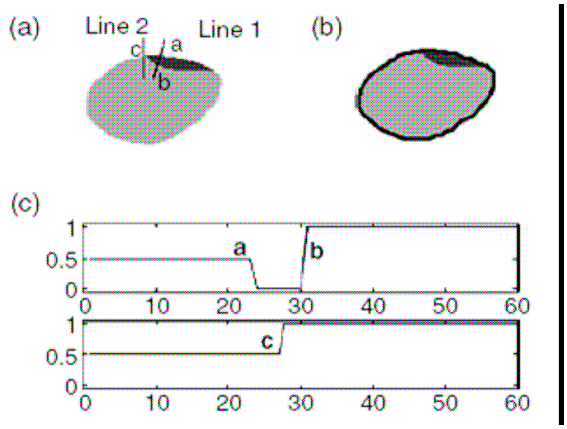

Fig. 4. Foreground object with multiple regions: (a) The synthesized image. (b) Contour tracking with JPM. (c) The observation on line 1 and line 2

where e(., .) is the cost of matching two pixels. $E^F(i,j)$ is the minimal matching cost between segment [−N, i] on line 1 and segment [−N, j] on line 2. We start from $E^F(-N, j) = E^F(i,-N) = 0$, where $i, j \in [-N,N]$ and use the above recursion to obtain the matching cost $E^F(i, j)$ from i = −N to N and j = −N to N. A similar process is used to calculate EB(i, j), but starting from $E^B(i,N) = E^B(N, j) = 0$ and propagate to $E^B(-N,-N)$. With all the matching cost, the state transition probabilities can be computed as in Eq.(12) and contour detection can be accomplished by the Viterbi algorithm.

4 CONCLUSIONS

Object tracking algorithms provide high-level semantic information about the objects in the videos and their motion trajectories and interactions, which can be very helpful in understanding the videos or classifying/querying the videos in the multimedia database. Initialization is necessary to start the tracking process. It can be done either manually or by an automatic object detection module (e.g. face detection (S.Z. Li, 2001).

In the interactive video/multimedia framework, it is important that the tracking modules can be initialized easily. Some tracking methods require strict and precise initialization. For example, many color based tracking methods (e.g., S.T. Birchfield, 1998, D. Comaniciu et all., 2000]) require a typical object color histogram. In (S.T. Birchfield, 1998) a side view of the human head is used to train a typical color model of both skin color and hair color to track the human head with out-of-plane rotation.

Another approach could be a HMM-UKF (Unscented Kalman Filter) framework that can be initialized by a rough bounding box indicating the object position and then adapt itself to the changing appearance or environments, which allows it to be easily integrated with external face detector or manual initialization.

To future improve the tracking results, it is possible to combined the HMM modeling with particle filters to handle non-Gaussian systems and maintain multiple hypotheses during object tracking.

5. REFERENCES

A.A. Amini, T.E. Weymouth, and R.C. Jain, 1990. Using dynamic programming for solving variational problems in vision. *IEEE Trans. Pattern Anal. Machine Intell.*, 12(9):855–867.

Y. Bar-Shalom and T.E. Fortmann, 1988. *Tracking and Data Association*. Academic Press, Orlando, Florida,.

S.T. Birchfield, 1998. Elliptical head tracking using intensity gradients and color histograms. In *Proc. IEEE Int'l Conf. on Comput. Vis. and Patt. Recog.*, pages 232–237.

A.J. Colmenzrez and T.S. Huang, 1997. Face detection with information-based maximum discrimination. In *Proc. IEEE Int'l Conf. on Comput. Vis. and Patt.Recog.*, pages 782–787,. Locating Information in Video by Browsing and Searching 229

D. Comaniciu, V. Ramesh, and P. Meer, 2000. Real-time tracking of non-rigid objects using mean shift. In *Proc. IEEE Int'l Conf. on Comput. Vis. and Patt. Recog.*, pages II 142–149,.

J. Denzler and H. Niemann, 1999. Active-rays: Polar-transformed active contours for real-time contour tracking. *Real-Time Imaging*, 5(3):203–13.

M. Isard and A. Blake, 1996. Contour tracking by stochastic propagation of conditional density. In *Proc. European Conf. on Computer Vision*, pages I:343–356.

M. Isard and A. Blake. 1998, CONDENSATION - conditional density propagation for visual tracking. *Int. J. Computer Vision*, 29(1):5–28.

M. Isard and A. Blake, 1998. ICONDENSATION: Unifying low-level and high-level tracking in a stochastic framework. In *Proc. European Conf. on Computer Vision*, pages 767–781.

M. Kass, A. Witkin, and D. Terzopoulos. Snakes, 1988: Active contour models. *IJCV*, 1(4):321–331.

S.Z. Li, Q.D. Fu, L. Gu, B. Scholkopf, Y. Cheng, and H.J. Zhang, 2001. Kernel machine based learning for multi-view face detection and pose estimation. In *Proc. IEEE Int'l Conf. on Computer Vision*, pages II: 674–679.

L.R. Rabiner and B.H. Juang, 1986. An introduction to hidden Markov models. *IEEE Trans. Acoust., Speech, Signal Processing*, 3(1):4–15,

Yong Rui, Thomas S. Huang, and Sharad Mehrotra , 1999. Constructing table-of content for videos. *Multimedia Syst.*, 7(5):359–368.

D. Terzopoulos and R. Szeliski, 1992. Tracking with Kalman snakes. In *Active Vision*, pages 3–20. Cambridge, MA: MIT Press.

Y. Wu and T.S. Huang, 2000. Color tracking by transductive learning. In *Proc. IEEE Int'l Conf. on Comput. Vis. and Patt. Recog.*, pages I:133–138.