OBJECT TRACKING IN VIDEO USING PARTICLE FILTERING

Krishna V. Tangirala and Kamesh R. Namuduri

Department of Electrical and Computer Engineering
Wichita State University, Wichita, KS 67260, USA.
Email: {kvtangirala, kamesh.namuduri}@wichita.edu

ABSTRACT

Object tracking in video is an important problem which has many applications like video surveillance, target tracking using video sensors, etc. This paper presents an approach to object tracking in a video sequence using particle filtering. The motion edge is modelled using a six parameter model. Spatio-temporal filtering techniques are used to determine the velocity of the moving edge. Particle filtering is used to propagate the multi dimensional posterior density over time. Experimental results have been presented to show the effectiveness of the proposed method.

1. INTRODUCTION

To track an object in a video sequence, the object or its features have to be identified first. An object is defined by the features it exhibits. To detect the motion of an object, the movement of the features must be tracked. After identifying the motion of the features, a prediction model must be formed such that the position of the features can be predicted in the next frame. In this paper, the edges are considered for tracking since edges are most prominent features of an object. This paper is organized as follows. Related work is presented in section 2, followed by motion model in section 3 and the proposed tracking algorithm is presented in section 4. Experimental results are presented in section 5 and conclusions are presented in section 6.

2. RELATED WORK

Though significant work has been done in this area by the research community, the problem is still considered as challenging. The two main approaches for object tracking in video are tracking in transform domain and tracking in spatial domain. Doherty et al [1], [2] proposed a wavelet domain based tracking scheme. Spatial domain based object tracking in a video sequence involves high dimensional nonlinear modelling. In [3], Zhou et al proposed a particle filtering algorithm for tracking of appearances in images. In [4], Yilmaz et al presented a Bayesian framework based technique for tracking objects in a video sequence using level-sets. Sommer et al [5] argue that spatial domain motion analysis using spatio-temporal derivatives is more appropriate for motion analysis than the spectral domain.

3. MOTION MODEL

The motion of an edge is parameterized by $(\theta,\ \mathbf{u}_f,\ \mathbf{u}_b,\ d)$, where, θ is the angle made by the edge, $\mathbf{u}_f$ is the foreground velocity, $\mathbf{u}_b$ is the background velocity and d is the perpendicular distance of the edge from the center of the block. This model is adopted from Black and Fleet's paper [6]. The moving edge (or the motion-edge) can be completely characterized by these parameters. It is assumed that the object, and hence the edge moves with the foreground velocity. The foreground and background velocities help in modelling the occlusion and disocclusion effects due to the movement of the object. Tracking of motion edge can now be re-defined as prediction of the motion-edge parameters for the next frame using the information available from the previous frames. In order to predict and track the parameters, a motion model must be defined, followed by a prediction algorithm, which are defined in the subsequent sections.

It is known that the displaced frame difference between two consecutive frames in a video sequence follows an i.i.d Gaussian random field distribution [6],[7]. That is,

$$I(\mathbf{x}',t) \ = \ I(\mathbf{x},t-1) + v_n(\mathbf{x},t) \tag{1}$$

where, $v_n \ \sim \ \mathcal{N}(0,\sigma_n^2)$ and, $\mathbf{x}'$ is the new pixel location to which $\mathbf{x}$ moved which is given by,

$$\mathbf{x}' \ = \ \begin{cases} \mathbf{x} \ + \ \mathbf{u}_f & \text{if } (\mathbf{x}-\mathbf{x}_c.\mathbf{n}) \ > \ d \\ \mathbf{x} \ + \ \mathbf{u}_b & \text{if } (\mathbf{x}-\mathbf{x}_c.\mathbf{n}) \ < \ d+w \end{cases} \tag{2}$$

where $w \ = \ \max((u_f - u_b), 0)$, and $u_f \ = \ \mathbf{u_f}.\mathbf{n}, u_b \ = \ \mathbf{u_b}.\mathbf{n}$ and $\mathbf{n} \ = \ (\cos(\theta), \sin(\theta))$.

It is assumed that the motion parameters follow a first order Markov process. Hence, the parameters at time t are dependent only on parameters at time $t-1$. The motion model, from [6] can be described by the following set of

equations.

$$\begin{aligned}
\mathbf{u}_{f,t} &= \mathbf{u}_{f,t-1} + \mathbf{v}_{u,f}, \quad \mathbf{v}_{u,f} \sim \mathcal{N}(0, \sigma_u^2 \mathbf{I}_2) \\
\mathbf{u}_{b,t} &= \mathbf{u}_{b,t-1} + \mathbf{v}_{u,b}, \quad \mathbf{v}_{u,b} \sim \mathcal{N}(0, \sigma_u^2 \mathbf{I}_2) \\
d_t &= d_{t-1} + \mathbf{n}_{t-1}.\mathbf{u}_{f,t-1} + v_d, \quad v_d \sim \mathcal{N}(0, \sigma_d^2) \\
\theta_t &= [\theta_{t-1} + v_\theta] \, mod \, 2\pi, \quad v_\theta \sim \mathcal{N}(0, \sigma_\theta^2) \quad (3)
\end{aligned}$$

The edge is assumed to move with a constant velocity. The orientation of the edge is always in the range of $[-\pi, \pi]$. Hence, the integer multiple of 2π is removed from the updated value of θ before propagating it to the next time step. The edge is assumed to move with the foreground velocity. Hence, the distance of the edge from the center of the block for the next time step is obtained by moving the edge with the normal component of the foreground velocity. Gaussian noise is added to account for modelling errors implicit in this model. For the velocity modelling, a zero mean Gaussian noise with covariance matrix $\sigma_u^2 \mathbf{I}_2$ are used. For the orientation θ and the distance d of the edge, zero mean Gaussian noise with variance of σ_θ^2 and σ_d^2 are used [6]. The state parameters are given by $\mathbf{s} = [\theta, \mathbf{u}_f, \mathbf{u}_b, d]$. The state update equation $p(\mathbf{s}_t \mid \mathbf{s}_{t-1})$ is given by (3). The observation equation is given by (1).

4. OBJECT TRACKING USING PARTICLE FILTERING

The motion-edge parameters or the state parameters and the evolution of the state parameters with time are described in the previous sections. Prediction of the motion parameters involves estimation of the model parameters at time-step t from the data available until the time-step $t-1$. This problem can be identified as a state-space estimation problem which includes a set of state evolution equations, as given in (3) and an observation equation as given in (1). Since this problem involves non-linear models, the particle filtering technique is used for prediction of the state parameters.

Particle filtering is a sequential Monte Carlo technique that recursively computes the posterior probability density function using the concept of importance sampling. When the prior importance function is taken as the importance sampling function, the update step in particle filtering is given by

$$p(\mathbf{s}_t^m \mid \mathbf{z}_t) \propto p(\mathbf{s}_{t-1}^m \mid \mathbf{z}_{t-1}) \, p(\mathbf{z}_t \mid \mathbf{s}_t^{(m)}) \quad (4)$$

The derivation of this equation is given in [8]. The only term to be defined in (4) is the likelihood, given by $p(\mathbf{z}_t|\mathbf{s}_t^{(m)})$. We use the same likelihood function as defined in [6], which is given by

$$p(\mathbf{z}_t \mid \mathbf{s}_t^{(m)}) = \left(\exp \left[\frac{-1}{2\sigma_n^2} \sum_{\mathbf{x} \in \mathcal{R}} E(\mathbf{x}, t; \mathbf{s}_t^{(m)})^2 \right] \right)^{\frac{1}{T}} \quad (5)$$

where, $E(\mathbf{x}, t; \mathbf{s}_t^{(m)}) = I(\mathbf{x}', t) - I(\mathbf{x}, t-1)$, T is the number of pixels sampled and $\mathbf{x}'$ is given by (2).

From the particle filtering equation in (4), it can be observed that the posterior at time instant t is obtained from posterior at time instant $t-1$ by multiplying with the likelihood function. At the first time instant, i.e., at $t = 1$, the previous posterior is not available. Hence, the previous posterior for the first time step, or the initialization prior is assumed to be a six-dimensional Gaussian density function given by,

$$p(\mathbf{s}_{t-1}^m \mid \mathbf{z}_{0:t-1}) \sim \mathcal{N}(\mu, \mathbf{R}) \quad (6)$$

where, $\mu = [\theta, u_{f_h}, u_{f_v}, u_{b_h}, u_{b_v}, d]^T$ and $\mathbf{R} = [\sigma_\theta^2, \sigma_{u_{f_h}}^2, \sigma_{u_{f_v}}^2, \sigma_{u_{b_h}}^2, \sigma_{u_{b_v}}^2, \sigma_d^2,]^T \mathbf{I}_6$. To form the initialization prior, the mean values of the state parameters are required. These mean values are computed using low-level detectors which are described in the next sub-section.

4.1. Low-level detectors

The low-level detectors provide an estimate of the state parameters. They are obtained directly from the individual frames of the video sequence. The following sections describe the methods we used to calculate the state parameters.

4.1.1. Calculation of d and θ

The state parameters are calculated over a neighborhood of pixels in a given frame. In our computations, a square neighborhood of length seventeen pixels is selected. The parameters d and θ are calculated by fitting a straight-line to the motion-edge formed in a neighborhood. It is assumed that the spatial edge captured within a neighborhood is approximately a straight line. The motion-edge can be observed from the frame difference of two consecutive frames. The difference image formed by subtracting two blocks gives the edge formed due to motion or the motion-edge formed by the moving edge.

In order to approximate the motion edge as a straight line, the coordinates that form the motion edge in the selected neighborhood are collected. Line-fitting is implemented by setting a threshold to the frame difference and selecting only those coordinates whose difference in frame intensities is greater than the threshold. A simple two-parameter line-fitting algorithm is used to fit a straight line into the selected pixels. The straight line is represented as $y(x_i; a, b) = a + bx_i$ and the line fitting technique described in [9] was implemented. The model parameters d and θ are given by

$$\begin{aligned}
d &= \frac{bx_c - y_c + a}{\sqrt{b^2 + 1}} \\
\theta &= \arctan b \quad (7)
\end{aligned}$$

4.1.2. Estimation of $\mathbf{u}_f$ and $\mathbf{u}_b$

A moving edge forms a plane in the 3-D space-time cube. The velocity of the edge is determined by using spatio-temporal filters. Four spatio-tempoal filters based on gradient operations [10], [11] are applied to the motion plane to detect its orientation. Each of the filters is a square matrix of size 5x5 with zeros along the orientation of interest. Four orientations, namely, $0°$, $45°$, $90°$ and $135°$ are used, and the two halves are filled with $+1$'s and -1's. One of the filters is given below.

$$F_0 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 \end{bmatrix}$$

Pixels of interest are collected over three frames to form a cube. From the cube hence formed, the motion plane is extracted. Let the result obtained by convolution of the motion plane by the four filters $F_0,\ F_{90},\ F_{45},\ F_{135}$ be $E_0,\ E_{90},\ E_{45},\ E_{135}$ respectively. The ratio of E_0 and E_{90} gives the ratio of the gradient along displacement axis and the time axis. This gradient represents the velocity of the edge. The direction of velocity is obtained by comparing the magnitudes of E_{45} and E_{135}. If E_{45} is greater than E_{135}, the object is moving away from the left-bottom edge of the frame towards the right-bottom edge of the frame. If E_{135} is greater than E_{45}, the object is moving towards the left-bottom edge of the frame from the right-bottom edge of the frame. By selecting a neighborhood that contains the object of interest, the foreground velocity $\mathbf{u}_f$ is calculated, and by choosing a neighborhood away from the object, the background velocity is computed.

Thus, the model parameters are computed from images which are the observations. The next sub-section describes the application of the particle filtering algorithm to the tracking problem.

4.2. Tracking

Using the low-level detectors described in previous section, the model parameters (used as mean in (6)) are estimated. The initialization prior is formed by using (6). The likelihood of particles is calculated using (5). Now, the particle filtering equation (4) can be applied directly to obtain the posterior for the next time step. The posterior is marginalized to obtain the mean values of the parameters of interest. The mean values of the parameters of interest are plotted over the next frame using the new region as described by (2). The edge is plotted in the updated neighborhood using the predicted values of d and θ. The fitted straight line is given by $y = bx + a$. The posterior density function gives the slope of the edge, θ and the perpendicular distance of the edge from the center of the block, given by d. From the two state-parameters, the edge is constructed by forming a straight line in the updated neighborhood. The equation of a straight line, in terms of the slope of its perpendicular from the origin α, and the perpendicular distance of the line from the origin, p is given by

$$\frac{x}{p \sec \alpha} + \frac{y}{p \csc \alpha} = 1 \tag{8}$$

With respect to the predicted parameters, p is equal to d, and α is equal to $\frac{\pi}{2} + \theta$. The edge given by the following straight line equation is fitted into the updated neighborhood.

$$y = \frac{p}{\sin \alpha} - x \cot \alpha \tag{9}$$

Thus, the edge of the object, and hence the object is tracked in a video sequence using particle filtering.

5. EXPERIMENTAL RESULTS

The algorithm is simulated in Matlab using the standard flower garden sequence. A square region of 17 pixels wide is chosen as the region of interest or the neighborhood for simulations. In the first set of experiments, a vertical and inclined edge in the flower garden sequence are tracked. The predicted edge is shown over the frame for which prediction is made. In order to distinguish the true and predicted edges, the neighborhood selected is filled with gray color. Tracking of vertical and inclined edges is shown in Figure 1. For each of the experiments, the predicted and true value of the parameter d is plotted for all the frames. It is to be noted that d is defined with respect to the local origin of the square neighborhood. In order to study the performance of the proposed tracking algorithm, the parameter d (for both true and predicted) is added to the coordinates of the center of the neighborhood (true and updated respectively) with respect to the frame. The absolute difference between the true and predicted value gives the error in prediction. The true value of parameter d is obtained from the ground truth information of the video sequence.

6. CONCLUSIONS AND FUTURE WORK

From the experiments it can be concluded that the proposed algorithm can be used for tracking objects in video sequences. Spatio-temporal filtering techniques have been used for estimating the velocity of the moving edge. Particle filter provides en elegant solution to the state-space estimation problem in a highly dimensional non-linear system. Due to the degeneracy of particles, particle filter fails to converge. To overcome these problems, resampling [12], [13] and sample impoverishment [14], [15] techniques have to be used. Also, the current implementation requires user interaction

and relies on some knowledge of the video sequence such as the initial orientation of the edge to pick up the correct motion cube, etc. More pre-processing steps can be included to automate the complete process.

7. REFERENCES

[1] Y. Wang, J. F. Doherty, and R. E. Van Dyck, "Moving object tracking in video," *Artificial Intelligence and Pattern Recognition*, p. 95, 2000.

[2] Y. Wang, R. E. Van Dyck, and J. F. Doherty, "Tracking moving objects in video sequences," *Conference on Information Sciences and Systems*, March 2000.

[3] S. Zhou, R. Chellappa, and B. Moghaddam, "Appearance tracking using adaptive models in a particle filter," *Proc. of 6th Asian Conference on Computer Vision (ACCV)*, January 2004.

[4] A. Yilmaz, X. Li, and M. Shah, "Object contour tracking using level sets," *Proc. of 6th Asian Conference on Computer Vision (ACCV)*, January 2004.

[5] W. Yu, G. Sommer, and K. Daniilids, "Multiple motion analysis: in spatial or in spectral domain?," *Computer Vision and Image Understanding*, vol. 90, no. 2, pp. 129–152, 2003.

[6] M. J. Black and D. J. Fleet, "Probabilisic detection and tracking of motion boundaries," *International Journal of Computer Vision*, vol. 38, no. 3, pp. 231–245, 2000.

[7] Y. Wang, Y. Zhang, and J. Ostermann, *Video Processing and Communications*, Signal Processing Series. Prentice Hall, 2001.

[8] S. Kambhampati, K. Tangirala, K. R. Namuduri, and S. K. Jayaweera, "Particle filtering for target tracking," *Wireless Personal and Multimedia Communications*, October 2004.

[9] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recepies in C*, Cambridge University Press, New York, USA, second edition, 1998.

[10] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Prentice Hall, New Jersey, USA, second edition, 1992.

[11] A. K. Jain, *Fundamentals Of Digital Image Processing*, Prentice Hall, New Jersey, USA, 1989.

[12] J. Zhang Y. Huang T. Ghirmai Mónica F. Bugallo P. M. Djurić, J. H. Kotecha and J. Míguez, "Particle filtering," *IEEE Signal Processing Magazine*, vol. 20, no. 5, September 2003.

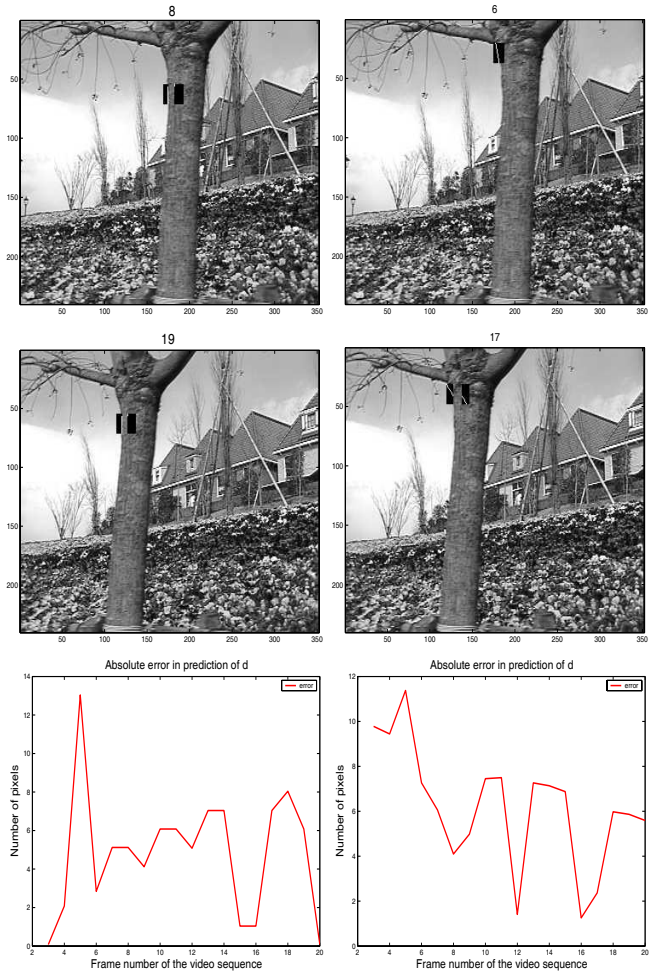

Fig. 1. The first column in this figure shows the tracking of a vertical edge and the second column shows the tracking of an inclined edge. The two graphs in the third row show the performance of the proposed tracking algorithm in the prediction of parameter d.

[13] J. S. Liu and R. Chen, "Sequential monte carlo methods for dynamical systems," *Journal of the American Statistical Association*, vol. 93, pp. 1033–1044, 1998.

[14] S. Arulampam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking," *IEEE* Trans. Sig. Proc., vol. 50, pp. 174–188, Feb 2002.

[15] A. Doucet, S. J. Godsill, and C. Andrieu, *On Sequential Monte Carlo sampling methods for Bayesian filtering*, vol. 3, Statist. Comput. Commun., July 2002.