CELL TRACKING USING PARTICLE FILTERS AND LEVEL SETS

Bharath Vishwanath[1] *and Chandra Sekhar Seelamantula*[2]

[1]Department of Electronics and Communication, National Institute of Technology, Surathkal, India
[2]Department of Electrical Engineering, Indian Institute of Science, Bangalore, India

brthv93@gmail.com, chandra.sekhar@ieee.org

ABSTRACT

We propose an algorithm to track moving cells and microbes in a video. A major challenge in tracking living cells is that their movement is often nonlinear which causes problems in case of approaches using the generic particle filter (GPF) framework. In order to overcome this problem, we propose the use of an auxiliary particle filtering (APF) algorithm with dynamic variance adaptation of the posterior distribution to account for nonlinear movements. The object of interest in each frame is segmented using level sets. The proposed tracking algorithm is tested on real data and the tracking performance is compared with that of GPF and APF without dynamic variance adaptation. Experimental results show that the proposed algorithm tracks more accurately compared to GPF and APF without variance adaption, with lesser number of particles, thereby reducing the running time.

Index Terms— Particle filter, level sets, dynamic proposal variance, Markov-Chain Monte-Carlo.

1. INTRODUCTION

Tracking of biological cells and microbes have found applications in several areas such as cell signalling and regulation, observation of microbes, study of cell motility [1], etc. Major challenges in cell tracking are the deforming cell shape and nonlinear movements. While the problem of cell shape deformation has been addressed in algorithms that employ generic particle filters by using level sets [2]-[4], the nonlinear cell movement still remains a challenge. We address the problem of tracking cells which exhibit nonlinear movements.

One of the important reasons for the limited performance in practice is that the posterior distribution does not give a good estimate of the next state. To overcome the problem, we have to refine the posterior distribution from the knowledge of the observation of the next state [5]. Considering this factor, auxiliary particle filter (APF) avoids particles from being moved to low likelihood regions. Thus, the proposed algorithm works well for uneven movements of cells and microbes. The user has to initialize a rectangular contour in the first frame. Thereafter, particle filter predicts the initial rectangular contour required to segment the image in each frame. Images are segmented using distance regularized level sets [6]. Variance of the particles is a key parameter that we have relied on in the paper. Variance of particles is varied dynamically based on the kinematics of the object and the error in the prediction of the next state. Consequently, the system gives a good prediction of the next state with lesser number of particles as compared to generic particle filters (GPF) and APF. Since the proposed method can track sudden movement, one can drop few frames from processing, thus bringing down the overall computation time of the system. In [4], particle filters (PF) estimate the affine parameters of the contour of the object. In contrast, particles are spread over the entire object in our approach. The dimensions of the object is also considered as a parameter to decide the variance of the distribution. Thus, the particle spreads according to the deformations. Hence, the present framework accounts well for deformations, without any affine transformation of the contour.

The authors would like to thank Haricharan Aragonda and Sunder Ram Krishnan for proofreading the paper.

2. PARTICLE FILTER

PF is defined by a state model and an observation model. In the present context, the state model characterizes the kinematics of the object. A sequence of images is taken to be the observation. The PF estimates the state s_{t+1} of a system at time $t+1$ based on the observations made at time t. The filter propagates the observations at time t as hypothesis at time $t+1$ using a state transition function.

$$s_{t+1} \approx f(s_{t+1}/s_t), \tag{1}$$

where s_t represents the state at time t and $f(s_{t+1}/s_t)$ is the state transition function which models the transition of the system from time t to time $t+1$. Based on this hypothesis, partial observations are made according to the observation model. The system is updated using Bayes rule as

$$p(s_{t+1}/y_{t+1}) \propto p(y_{t+1}/s_{t+1})p(s_{t+1}/y_t), \tag{2}$$

where y_t is the observation made at time t and $p(s_t/y_t)$ is the likelihood function at time t. The proportionality constant

978-1-4799-2827-9/13/$31.00 ©2013 IEEE

is the normalization constant of the probability density function (pdf). These two processes are carried out recursively. Monte-Carlo representation estimates the likelihood function. The samples of the likelihood function are referred to as *particles* and their associated likelihood as *weights*. The sequence of states is assumed to be a Markov process. Thus, the PF is popularly referred to as Markov-chain Monte-Carlo filter. A frequent problem encountered in PF is the degeneracy phenomenon, wherein all except few particles have negligible weights. The effect of degeneracy can be reduced by resampling [7]. Resampling eliminates the particles that have negligible weights and duplicates the particles with large weights, thereby generating a new set of particles. Many variants of PF have been discussed in [7] and [8]. The basic framework of the APF suggested in [7] and [9] is amalgamated with improvements suggested to GPF in [10]. This idea is further extended in APF, as discussed in Section 4.

3. AUXILIARY PARTICLE FILTERS

APF is used to track an object in a video, which is a time series of images. The coordinates of particles at an instant of time t is represented as $z_t^{(i)}$, i ranging from 1 to N, N being the total number of particles. The corresponding weights of these particles is represented as $w_t^{(i)}$. Let the centroid of the contour given by level sets be (x_c, y_c). Let Δx_c and Δy_c be the displacement of the centroid from time $t-1$ to time t along x and y directions respectively. The energy function defined for an image in [6] to obtain the contour, is considered as the likelihood function to update the weights of particles. The algorithm is as follows:
A rectangular contour initialization is given by the user for the level sets to segment the image. Draw N samples from the image energy $g(y)$ given by the level sets. The value of the image energy evaluated at these pixels are normalized to construct a pdf, that gives the weights $w_{t=0}^{(i)}$.

3.1. State model

Particles are propagated from time *t* to time $t+1$ as

$$z_{t+1}^{(i)} = z_t^{(i)} + v_t + m_t^{(i)}, \tag{3}$$

where v_t is the displacement of the centroid of the contour from the state at $t-1$ to state at t, given by $[\Delta x_c, \Delta y_c]$. $m_t^{(i)}$ are the samples of a two-dimensional Gaussian distribution with zero mean and σ_{1x}^2 and σ_{1y}^2 variances along x and y directions respectively.

3.2. Intermediate Weights

Observe the state s_{t+1}, and compute the intermediate weights $\hat{w}(z_{t+1}^{(i)})$ as

$$\hat{w}(z_{t+1}^{(i)}) \propto g(y_{t+1}/z_{t+1}^{(i)}) w_t^{(i)}, \tag{4}$$

where $g(y_{t+1}/z_{t+1}^{(i)})$ is the image energy evaluated at the pixel coordinates predicted by the particles. The weights are normalized appropriately.

3.3. Resampling

Having observed the state at $t+1$, retrace back to time t and resample $z_t^{(i)}$ using the intermediate weights to obtain $\hat{z}_t^{(i)}$. This is a crucial step, since it selects proper parent particles.

3.4. Updating weights

Propagate the resampled parent particles as

$$\hat{z}_{t+1}^{(i)} = \hat{z}_t^{(i)} + v_t + n_t^{(i)}. \tag{5}$$

$n_t^{(i)}$ are the samples of a two-dimensional Gaussian distribution with zero mean and σ_{2x}^2 and σ_{2y}^2 variances along x and y directions respectively. This perturbs the particle and thus spreads the duplicated particles. Evaluate the likelihood $\tilde{w}(\hat{z}_{t+1}^{(i)})$ at the new pixel coordinates using $g(y_{t+1}/\hat{z}_{t+1}^{(i)})$. Update the weights as

$$w_{t+1}^{(i)} = \frac{\tilde{w}(\hat{z}_{t+1}^{(i)})}{\hat{w}(\hat{z}_{t+1}^{(i)})}. \tag{6}$$

Normalize these weights so as to construct a pdf. The state model is updated and the algorithm is repeated recursively.

4. DYNAMIC PROPOSAL VARIANCE

In order to address the degeneracy problem discussed in Section 2, we need to continuously adapt the variance of the particles. This is referred to as dynamic proposal variance and was put forward in [10]. This idea is made use in the present work to determine the variance of the proposal distribution as,

$$\sigma_{1x}^2 = \sqrt{2} c \Delta x_c \text{ and } \sigma_{1y}^2 = \sqrt{2} c \Delta y_c. \tag{7}$$

The value of c as suggested in [10] ranges from 0.1-0.2. We found that this suggested value worked well for our application as well. This gives a better spread of particles when the object is in motion and reduces the chances of tracking failure. On similar lines, the variance of the Gaussian random variable that perturbs the resampled particles is varied dynamically based on the amount of degeneracy, which is a measure of the spread of the resampled particles in (3.3). Variance adaptation acts as a feedback for the particles based on their accuracy of prediction. Degeneracy is quantified by considering the second-order statistics of the intermediate weights. We make use of the parameter $\hat{N}_{eff}$ as defined in [7] to quantify the amount of degeneracy.

$$\hat{N}_{eff} = \frac{1}{\sum_{k=1}^{N} (\hat{w}_t^{(i)})^2}. \tag{8}$$

We define a parameter β as

$$\beta = \frac{\hat{N}_{eff}}{N}. \tag{9}$$

Value of β greater than a certain threshold signifies that a sufficient number of particles estimated the next state appropriately. Since this reflects good prediction, the variance of the prediction noise is taken to be of small value. However, if β falls below a threshold, it signifies that many particles collapsed over few pixels on resampling, indicating a possible error in prediction. Hence, a larger value of variance has to be taken to perturb the particles. The variances σ_{2x}^2 and σ_{2y}^2 are taken as

$$\sigma_{2x}^2 = k_2 x_d \text{ and } \sigma_{2y}^2 = k_2 y_d, \tag{10}$$

where x_d and y_d are the horizontal and vertical dimensions of the rectangle that fits the object contour. Constraining the variance to be a function of object dimensions is advantageous as the tracking process then adapts to object deformations over time. The following values of k_2 were empirically determined after optimizing the value for best tracking performance over a set of 18 videos.

$$k_2 = \begin{cases} 0.05 & \text{if } \beta > 0.75, \\ 0.1 & \text{if } 0.5 <= \beta <= 0.75, \\ 0.15 & \text{if } \beta < 0.5. \end{cases} \tag{11}$$

A large deviation from the suggested values led the PF to diverge or collapse and lose track of the object.

5. EXPERIMENTAL RESULTS

In this section, we compare the tracking performance of GPF, APF, and the proposed method. The algorithms are evaluated in terms of average execution time per frame and the number of particles needed to track accurately. All the videos have a frame rate of 20 fps. The results pertain to a MATLAB implementation of the code on a 2.5GHz Intel Core i5 processor. Tracking results on various videos are illustrated using a series of images in Fig. 1. The tracking problem was considered by processing alternate frames during which, GPF deviated at certain points due to nonlinear motions. On the other hand, APF worked well for these conditions, and could track with lesser particles. With the inclusion of variance adaptation, the particle requirement dropped further to the values given in Table 1. Consequently, there was a reduction in the computation time which is given in Table 2.

The image sequence of Colpidium in row 1 of Fig. 1 reflects that the present framework can work well with rotations, although we have not considered an affine parameter for the same. In row 2, we show the image sequence of a tracked Phacus. Phacus tends to contract or elongate during its locomotion. These deformations can be clearly perceived in the figure. Robustness to the sudden movement can be clearly observed in the plots of the centroid of the contour in Fig. 2. At the points of abrupt motions, GPF and APF gave a poor estimate of the centroid. However, our approach proved robust for the same. The plot labeled 'ground truth' was obtained by manually initializing the contour for level sets in each frame. Experimental results are available at *www.brthvnitk.wix.com/celltracking.*

Object Tracked	Resolution	GPF	APF	Our Approach
Euglena	85×65	75	60	40
Phacus	85×65	80	70	50
DNA	320×240	90	70	50
Granulocyte	320×240	75	50	40
Leukocyte	320×240	90	60	50
DNA	320×240	110	80	60

Table 1. Number of particles for different approaches

Object Tracked	GPF	APF	Our Approach
Euglena	0.427	0.33	0.254
Phacus	0.419	0.325	0.283
DNA	3.821	3.365	2.897
Granulocyte	3.762	3.218	2.587
Leukocyte	3.189	2.767	2.171
DNA	3.373	2.869	2.262

Table 2. Computation time

6. CONCLUSIONS

We proposed an APF-based algorithm for tracking deforming objects, and demonstrated its robustness to nonlinear cell movements. The tracking performance of the proposed algorithm was compared with that of GPF and APF. Apart from the improved ability to track nonlinear movements of cells, the proposed algorithm can operate accurately using lesser number of particles and at lower frames rates making it computationally faster than the GPF and basic APF framework.

As a part of future work, level sets could be replaced with segmentation algorithms that incorporate shape priors. This is of interest in cases when there are other objects in the close vicinity of the object of interest.

7. REFERENCES

[1] E. Meijering etal, "Tracking in cell and developmental biology," *Semin. Cell and Dev. Biol.*, vol. 20, no. 8, pp. 894–902, Oct. 2009.

[2] X. Fei and K. Hashimoto, "An object tracking algorithm

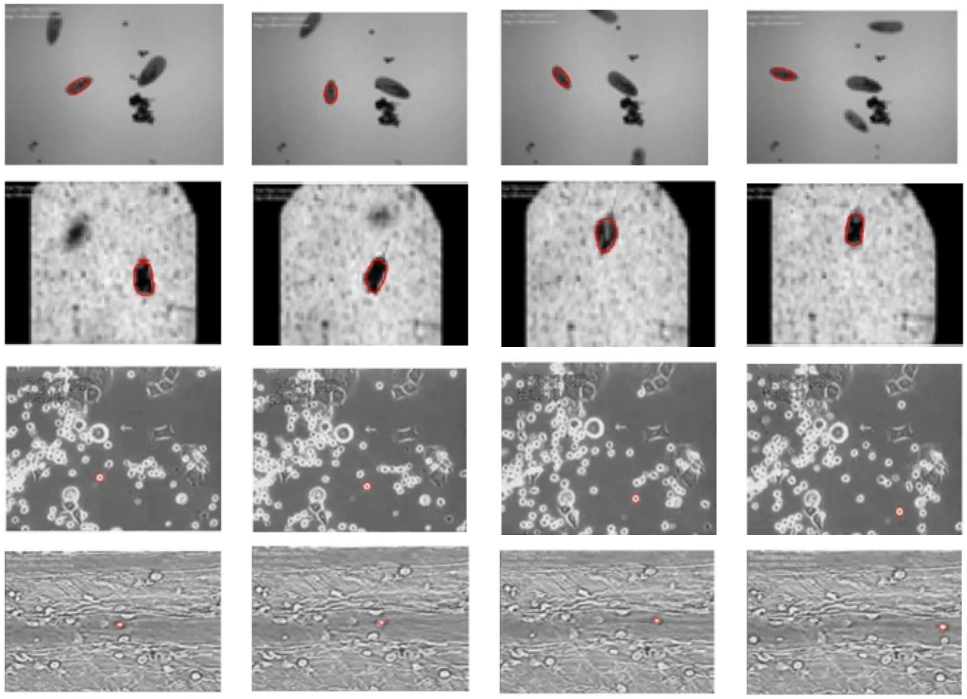

Fig. 1. Row 1: Colpidium, Row 2: Phacus, Row 3: Granulocytes, and Row 4: Leukocytes. (color in electronic version).

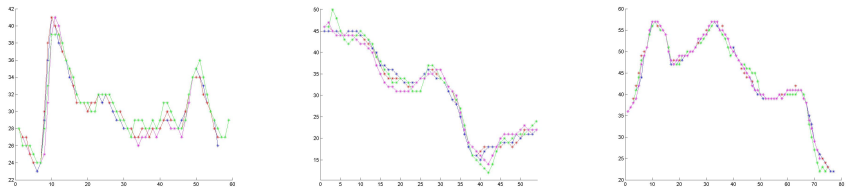

Fig. 2. Plots of y-coordinate of the tracked Phacus, Colpidium and DNA against the frame number. Red: ground truth, Green: GPF, Magenta: APF, and Blue: Our approach. (color in electronic version).

based on particle filtering with region based level sets," in *IEEE/RSJ Int. Conf. on Robotics and Syst.*, Oct. 2010.

[3] Y. Xu, C. Wu , X. Zhang, and Y. Zhao, "Tracking moving vehicles using particle filtering and level set," in *Proc. CVPR*, 2010.

[4] Y. Rathi, N. Vaswani, A. Tannenbaum, and A. Yezzi-Tracking, " Tracking Deforming objects using particle filtering for geometric active contours," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 8, Aug. 2007.

[5] Y. Rui and Y. Chen, "Better proposal distributions: Object tracking using unscented particle filter," in *CVPR*, 2001, pp. 786–793.

[6] C. Li, C. Xu, C. Gui, and M. D. Fox, "Distance regularized level set evolution and its application to image segmentation," *IEEE Trans. Image Process.*, vol. 19, no. 12, Dec. 2010.

[7] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian Bayesian tracking," *IEEE Trans. Signal Process.*, vol. 50, pp. 174–188, Feb. 2002.

[8] S. Godsill, *Sequential Monte Carlo Methods*, University of Cambridge, Sept. 2011.

[9] M. Pitt and N. Shephard, "Filtering via simulation: Auxiliary particle filters," *J. Amer. Statist. Assoc.*, vol. 94, no. 446, pp. 590–599, 1999.

[10] P. Pan and D. Schonfeld, "Dynamic proposal variance and optimal particle allocation in particle filtering for video tracking," *IEEE Trans. on Circuits and Syst. for Video Technol.*, vol. 18, no. 9, Sept. 2008.