AUGMENTED PARTICLE FILTERING FOR EFFICIENT VISUAL TRACKING

Chunhua Shen, Michael J. Brooks, Anton van den Hengel

[1]School of Computer Science, University of Adelaide, Australia [2] CSSIP, Australia
email: chhshen@cs.adelaide.edu.au

ABSTRACT

Visual tracking is one of the key tasks in computer vision. The particle filter algorithm has been extensively used to tackle this problem due to its flexibility. However the conventional particle filter uses system transition as the proposal distribution, frequently resulting in poor priors for the filtering step. The main reason is that it is difficult, if not impossible, to accurately model the target's motion. Such a proposal distribution does not take into account the current observations. It is not a trivial task to devise a satisfactory proposal distribution for the particle filter. In this paper we advance a general augmented particle filtering framework for designing the optimal proposal distribution. The essential idea is to augment a second filter's estimate into the proposal distribution design. We then show that several existing improved particle filters can be rationalised within this general framework. Based on this framework we further propose variant algorithms for robust and efficient visual tracking. Experiments indicate that the augmented particle filters are more efficient and robust than the conventional particle filter.

1. INTRODUCTION

Robust visual tracking is one of the core challenges in computer vision with many potential applications. Unfortunately development of a robust and efficient tracking framework remains unsolved. The particle filter (CONDENSATION [1] or sequential Monte Carlo (SMC)) has been extensively studied and used for visual tracking due to its flexibility. Compared with the Kalman filter, it relaxes the linearity and Gaussianity requirements.

From the Bayesian filtering equation, it can be seen clearly that two main components are involved in particle filter based tracking algorithms. One component is the likelihood formulation, *i.e.* the object representation and matching. All approaches can be classified into bottom-up [1], top-down [2] or a combination of both [3, 4]. This issue has important effects on the tracker's robustness. The other component is the importance sampling proposal distribution, which determines the particle filter's efficiency. A typical disadvantage of Monte Carlo sampling based methods is that they are often computationally demanding. For real-time applications such as visual tracking, it is critically important to improve the sampling efficiency so that fewer particles are needed to represent the filtering distribution, thereby reducing the computational burden. For the conventional particle filter, the system transition is used as the proposal distribution for simplicity. However it is not the "optimal" proposal sampling distribution. It has been shown that the "optimal" importance sampling proposal should utilise the most recent observations [5]. While the simple system transition proposal does not take into account the current observations, a more promising proposal distribution should be devised for better sampling efficiency. It is not a trivial task to devise a satisfactory proposal distribution for the particle filter. We shall see that both of these two issues can be accommodated in our more general augmented particle filtering framework.

Related work. Much effort has been expended to improve the performance of the particle filter. Several strategies have been advanced to relocate the particles to compensate for a poorly modelled system transition prior. The works in [6–9] fall into this category; all use hill-climbing (gradient ascent) algorithms to move the particles towards the dominant modes in the likelihood or posterior domain. Thus fewer particles are required to represent the distribution well. We shall refer these algorithms as *mode-climbing* approaches, in contrast to a novel *mode-hopping* approach presented here.

In order to take advantage of the most recent observations, the Kalman filter has been integrated into the particle filter, resulting in the so-called Kalman particle filter. Its variants include the Extended Kalman particle filter and the unscented particle filter. In a Kalman particle filter, the Kalman filter generates Gaussian distributions for each particle from which one performs the sampling. It has been introduced into the visual tracking field and better performances are observed [10, 11]. Nevertheless the price of this improvement is much extra computation. The Kalman particle filter requires running N Kalman filters in parallel for each frame, where N is the number of particles.

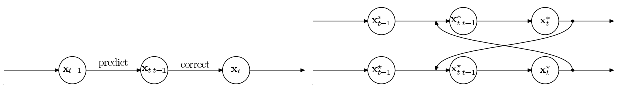
A related work is the Co-inference approach in [3]. Here, Wu *et al.* formulate the multiple cue integration as a graphical model, which is then factorised to achieve feasible inference. The core idea is straightforward: the sampling of particles for one cue from other cue's filtering distributions. This process is iterated until convergence. In this way, different cues support each other and enhance the performance. One problem in [3] is that, each cue is supposed to be reliable. In fact sampling from an unreliable cue's filtering distribution offers no benefit and might even be detrimental to the performance. In the following section we explain the role of the Co-inference strategy in our framework and discuss it in more detail.

Our approach. In this paper we first propose a general augmented particle filtering framework for designing an optimal proposal distribution. The essential idea is to augment a second filter's estimate into the proposal distribution design. We shall see that several existing improved particle filters can be unified into our general framework. Based on this framework we further propose novel variant algorithms for robust and efficient visual tracking.

2. THE AUGMENTED PARTICLE FILTERING FRAMEWORK

Bayesian filtering. Denote the state vector as $\mathbf{x}$ and observations

0-7803-9134-9/05/$20.00 ©2005 IEEE



(a) (b)

Figure 1: Diagrams of: (a) the particle filter and (b) the augmented particle filter. The *predict* step is also known as *time update* while *correct* means *measurement update*.

$\mathbf{z}$, the tracking problem is a Bayesian filtering inference problem:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \propto p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1}), \tag{1}$$

where the prior is the previous posterior propagated across the temporal axis,

$$p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})d\mathbf{x}_{t-1}. \tag{2}$$

The filtering process can be represented by graphical models in Figure 1(a). By propagating the prior $p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})$ through the system dynamics model $p(\mathbf{x}_t|\mathbf{x}_{t-1})$, we obtain the output $\mathbf{x}_{t|t-1}$. In Kalman filters, this step is named *time update*. The predicted state is then corrected by the likelihood $p(\mathbf{z}_t|\mathbf{x}_t)$. This is the *measurement update* step.

The filtering can be realised by Kalman filters when the linearity and Gaussianity conditions are satisfied. Otherwise we have to approximate the integral in Equation (2) with numerical sampling techniques. A typical sampling technique is importance sampling. At each frame t, the posterior is approximated by a set of discrete particles $\{\mathbf{x}_t^{(n)}, w_t^{(n)}\}_{n=1}^N$, where $\sum_{n=1}^N w_t^{(n)} = 1$ holds. If we sample the particles from proposal density $q(\cdot)$, *i.e.* $\mathbf{x}_t^{(n)} \sim q(\mathbf{x}_t|\mathbf{x}_{t-1}^{(n)}, \mathbf{z}_{1:t})(n = 1 \ldots N)$, then each particle's weight is set to

$$w_t^{(n)} \propto \frac{p(\mathbf{z}_t|\mathbf{x}_t^{(n)}p(\mathbf{x}_t^{(n)}|\mathbf{x}_{t-1}^{(n)})}{q(\mathbf{x}_t|\mathbf{x}_{t-1}^{(n)}, \mathbf{z}_{1:t})}. \tag{3}$$

The deficiency of the standard importance sampling is well known. Doucet *et al.* show that the "optimal" importance sampling proposal is $p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(n)}, \mathbf{z}_t)$ [5]. Therefore the most recent observations should be utilised.

Augmented particle filtering. Motivated by the success of improved particle filters such as Kalman particle filters [10,11] and Co-inference [3], we propose a general augmented particle filtering framework in which a two-stage sampling approach is adopted. We illustrate the framework with two filters being augmented. Apparently it is straightforward to extended this framework to the case of more than two filters. The key idea is that we augment a particle filter (or a Kalman filter) to learn a sampling distribution automatically at the first stage. At the second stage, the estimate produced by the first-stage filtering is fed to the second particle filter for constructing an optimal proposal. This process can be iterated to yield better results.[1]

Figure 1(b) depicts this strategy. There are two filters running iteratively. The first filter, which could be a Kalman filter[2] or a particle filter, is used to obtain a rough estimate of the state vector depending on the state estimate of the last step $\mathbf{x}_{t-1}$ and the current observation $\mathbf{z}_t$. Specifically, according to the type of the first-stage filter adopted, we study the algorithm in the following two cases.

Case 1: *The first-stage filter is a Kalman filter.* After each particle $\mathbf{x}_{t-1}^{(n)}$ propagates through the system dynamics equation, the sample mean $\widetilde{\mathbf{x}}_{t|t-1}$ and covariance $\widetilde{\mathbf{P}}_{t|t-1}$ are readily available. Then, in the filtering step for the Kalman particle filter algorithm [11, 12], a measurement update with a Kalman filter is applied to each particle. Thus N Kalman filters need to run. Clearly it is computationally expensive, although such an approach might better maintain multiple modes. For the *Maximum A Posterior* (MAP) estimation, the purpose is to find the maximum in the posterior space. Consequently we only use the sample mean $\widetilde{\mathbf{x}}_{t|t-1}$ and associated covariance $\widetilde{\mathbf{P}}_{t|t-1}$ to perform the Kalman update. This way we maintain only the main mode but significantly reduce the computational demand. The Kalman measurement update process writes:

$$\widetilde{\mathbf{x}}_t = \widetilde{\mathbf{x}}_{t|t-1} + \mathbf{K}_t(\mathbf{z}_t - \mathbf{H}\widetilde{\mathbf{x}}_{t|t-1}) \tag{4}$$

$$\widetilde{\mathbf{P}}_t = \widetilde{\mathbf{P}}_{t|t-1} - \mathbf{K}_t\mathbf{H}\widetilde{\mathbf{P}}_{t|t-1} \tag{5}$$

where $\mathbf{H}$ is the measurement matrix (for the extended Kalman filter, $\mathbf{H}$ is a linearised Jacobi measurement matrix) and the Kalman gain $\mathbf{K}_t$ is $\mathbf{K}_t = \widetilde{\mathbf{P}}_{t|t-1}\mathbf{H}^\top(\mathbf{H}\widetilde{\mathbf{P}}_{t|t-1}\mathbf{H}^\top + \mathbf{R}_t)^{-1}$. $\mathbf{R}_t$ is the covariance matrix of the observation noise at time t. Then we obtain the filtered estimate $\widetilde{\mathbf{x}}_t$ and its covariance $\widetilde{\mathbf{P}}_t$ which are more accurate than their predicted version $\widetilde{\mathbf{x}}_{t|t-1}$ and $\widetilde{\mathbf{P}}_{t|t-1}$. For the unscented Kalman filter, Equations (4) and (5) are slightly different. See [12] for details.

At the second step, we need to take full advantage of the sufficient Gaussian statistics produced at the first step. We construct a Gaussian proposal $\mathcal{N}(\mathbf{x}_t; \widetilde{\mathbf{x}}_t, \widetilde{\mathbf{P}}_t)$ from which we sample N particles $\mathbf{x}_t^{(n)}$ for the second particle filter. The second particle filter yields the second-stage posterior estimate $\mathbf{x}_t$ with a weighting calculation, as the standard particle filter does. Note that since the one-to-one mapping relationship between a particle at time $t-1$, $\mathbf{x}_{t-1}^{(n)}$ and its corresponding offspring $\mathbf{x}_t^{(n)}$ at time t has been broken, the weighting equation (3) needs to be modified.

We have drawn N samples $\{\mathbf{x}_t^{(n)}, w_t^{(n)}\}_{n=1}^N$ from the proposal $\mathcal{N}(\mathbf{x}_t; \widetilde{\mathbf{x}}_t, \widetilde{\mathbf{P}}_t)$, and from Equation (1), each particle's weight is

$$w_t^{(n)} = \frac{p(\mathbf{z}_t|\mathbf{x}_t^{(n)})p(\mathbf{x}_t^{(n)}|\mathbf{z}_{1:t-1})}{\mathcal{N}(\mathbf{x}_t^{(n)}; \widetilde{\mathbf{x}}_t, \widetilde{\mathbf{P}}_t)}. \tag{6}$$

We already have a set of discrete particles $\{\mathbf{x}_{t-1}^{(m)}, \frac{1}{N}\}_{m=1}^N$ to represent the posterior at time $t-1$ after *re-sampling*, therefore Equation (2) can be rewritten as

$$p(\mathbf{x}_t^{(n)}|\mathbf{z}_{1:t-1}) = \frac{1}{N}\sum_{m=1}^N p(\mathbf{x}_t^{(n)}|\mathbf{x}_{t-1}^{(m)}). \tag{7}$$

Combining Equations (6) and (7), we obtain the weight calculation equation,

$$w_t^{(n)} = \frac{1}{N}\frac{p(\mathbf{z}_t|\mathbf{x}_t^{(n)})\sum_{m=1}^N p(\mathbf{x}_t^{(n)}|\mathbf{x}_{t-1}^{(m)})}{\mathcal{N}(\mathbf{x}_t^{(n)}; \widetilde{\mathbf{x}}_t, \widetilde{\mathbf{P}}_t)}, \; n = 1 \ldots N. \tag{8}$$

Note that this two-step iteration does not necessarily run once. Multiple loops might be implemented if necessary. The essential point is that these two filters carry out the estimation in a cooperative way: each one utilises the output of the other one at the previous step (or at the previous iteration).

[1] The second particle can also be a Kalman filter. In visual tracking applications, the system is always nonlinear and non-Gaussian, particle filters give better results. We only consider the case that the second filter is a particle filter in this work.

[2] A plain Kalman filter, extended Kalman filter or unscented Kalman filter *etc.*

Compared with the Kalman particle filter, we only use a single Kalman filter to construct a Gaussian proposal distribution. This idea is similar but not identical to the Kalman particle filter algorithm. Assume that the computational bottleneck is the likelihood calculation and ignore the Kalman filter's matrix operation, our strategy reduces the Kalman particle filter's computation complexity from $\mathcal{O}(2N)$ to $\mathcal{O}(N+1)$. When using this framework for tracking, it is also different from the ICONDENSATION tracker [13], in which a colour tracker is used as an auxiliary tracker to roughly estimate the tracked object's state. Then the estimated state is used to generate a proposal for the main tracker. The limitation is that their method is *ad hoc* and it is rather difficult to estimate the covariance uncertainty.

Case 2: *The first-stage filter is a particle filter.* In theory, the particle filter is more general than the Kalman filter. Its shortcoming is that it is more computationally demanding than the Kalman filter. When the first-stage filter is also a particle filter, this algorithm is quite similar to the Co-inference algorithm [3] when the two different cues are used. Instead of deducing this algorithm with a structured variational approximation on a factorised graphical model, we intuitively invent this algorithm based on Figure 1(b).

The particles $\{\mathbf{x}_{t-1}^{(n)}, \frac{1}{N}\}_{n=1}^{N}$ at time $t-1$ are input to the first-stage particle filter, and after a standard particle filtering, we obtain the weighted particle set $\{\widetilde{\mathbf{x}}_t^{(n)}, \widetilde{w}_t^{(n)}\}_{n=1}^{N}$ which are a rough representation of the posterior. More specifically, for the first loop (if multiple loops are adopted as in [3]), we use the system dynamics transition as the proposal and obtain the discrete representation of the posterior with the Equations (2) and (3). Then the second-stage particle filter samples from this discrete density. After finishing the first loop, each filter samples the other one's estimated posterior. In contrast with the Kalman filter which estimates the posterior up to second-order statistics, one can easily calculate higher order statistics from the discrete particles.

Borrowing the idea from the Gaussian particle filter [14], and also making the sampling process consistent with the Kalman first-stage filter, we fit the discrete particles with a single Gaussian distribution $\mathcal{N}(\mathbf{x}_t; \widetilde{\mathbf{x}}_t, \widetilde{\mathbf{P}}_{t|t-1})$ from which N particles are sampled for the next iteration. The Gaussian is calculated before re-sampling so as to suppress the bias produced in re-sampling.

To summarise we present the complete augmented particle filter algorithm in Figure 2.

Remarks. Compared with *mode-climbing* approaches (*e.g.* [6–9]) in which a gradient-ascent optimisation method is usually applied to move the particles towards the modes (peaks) in the likelihood space, our filtering-based optimisation method can be regarded as a *mode-hopping* process. Due to the nature of the gradient-ascent based algorithms, the optimisation process is slow and one has to preset the move step. The augmented particles filter finds a Gaussian proposal distribution quickly through one step filtering which is usually close to the main mode.[3] Another requirement of the gradient-ascent optimisation is that the likelihood surface must be smooth, *i.e.* differentiable. If Hessian information is utilised, it is further required that the likelihood function's second derivatives exist everywhere. In visual tracking, many similarity measurement criteria do not satisfy the smooth conditions. This means one has to carefully devise the likelihood function. The augmented particle filter relaxes these restrictions. The difference between these two methods is depicted in Figure 3.

- **Initialisation**:
  Set $t = 1$. Sample N particles $\{\mathbf{x}_{t-1}^{(n)}, w_{t-1}^{(n)}\}_{n=1}^{N}$ from the prior $p(\mathbf{x}_0)$.
- **Re-sampling**:
  Re-sample to obtain N replacement particles $\{\mathbf{x}_t^{(n)}, \frac{1}{N}\}_{n=1}^{N}$, according to the weights $w_t^{(n)}$.
- **First-stage filtering**:
  Given $\{\mathbf{x}_{t-1}^{(n)}, w_{t-1}^{(n)}\}_{n=1}^{N}$ (or the output particles of the previous loop) and the current measurement $\mathbf{z}_t$, update the particles either with a Kalman filter using Equations (4) and (5) or with a particle filter using Equations (2) and (3). Compute the mean $\widetilde{\mathbf{x}}_t$ and covariance $\widetilde{\mathbf{P}}_t$.
- **Second-stage filtering**:
  1. Sample N particles $\{\mathbf{x}_t^{(n)}, w_t^{(n)}\}_{n=1}^{N}$ from the proposal $q(\mathbf{x}_t|\mathbf{x}_{t-1}^{(n)}, \mathbf{z}_{1:t}) = \mathcal{N}(\mathbf{x}_t; \widetilde{\mathbf{x}}_t, \widetilde{\mathbf{P}}_t)$.
  2. Calculate the weight for each particle according to Equation (8). Then normalise the weights.
- (Optional) If necessary, go to the **Re-sampling** step to perform another iteration. Note that this time the input is the output particles of the second-stage particles filtering, instead of $\{\mathbf{x}_{t-1}^{(n)}, w_{t-1}^{(n)}\}_{n=1}^{N}$.
- Set $t = t + 1$, go to the **Re-sampling** step to process the next frame.

Figure 2: The augmented particle filtering algorithm.

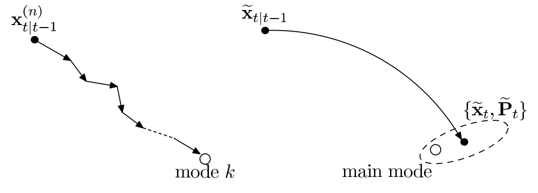

$\mathbf{x}_{t|t-1}^{(n)}$
mode k
(a)
$\widetilde{\mathbf{x}}_{t|t-1}$
$\{\widetilde{\mathbf{x}}_t, \widetilde{\mathbf{P}}_t\}$
main mode
(b)

Figure 3: (a) The gradient-ascent based mode-climbing optimisation needs many steps to drive the particles to the modes, while (b) the filtering based mode-hopping method finds a Gaussian distribution nearby the main mode through single one step filtering.

In the two-stage filtering process, the observations $\mathbf{z}_{1:t}$ can arise from different cues. This way multiple cues are integrated together seamlessly: different cues support each other during the inference. As we have stated, when the two filters are both particle filters, the framework presented in this paper is identical to the Co-inference algorithm in [3]. However, to adopt a Kalman filter in the first-stage filtering can save significant computational resources. To our knowledge this is the first proposal to use an efficient hybrid Kalman/particle filtering framework to integrate multiple cues in the visual tracking. Although one can also integrate multiple cues in the Kalman particle filter algorithm, our algorithm is more computationally efficient. One problem for a multiple cue integration strategy based on Co-inference is that only when both of the cues are reliable is this cooperative sampling strategy helpful. Otherwise, *e.g.* when one cue is completely invalid, sampling from its posterior will mis-guide the particles into the wrong area of the posterior and damage the inference. A naïve approach has been adopted in [4] to alleviate this problem. Still more elegant approaches are required. Due to the space limit, we shall not discuss the multiple cue integration issue in more detail in this paper.

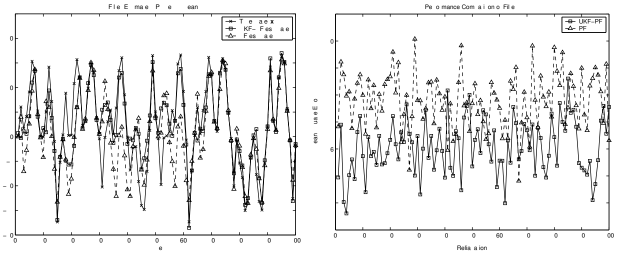
[3] For high-dimensional application, the manifold of the likelihood distribution is extremely complex. Both gradient-ascent optimisation and our augmented particle filter algorithm still might fail if the number of particles is not sufficiently large. The problem of how to tackle the *curse of dimensionality* when sampling in a high-dimensional space remains unsolved.

3. EVALUATION

We examine the performance of the augmented particle filter in two experiments. One is a Kalman/particle filtering approach on a 1D tracking simulation with synthetic data. The other one is a particle/particle filtering approach on a real video tracking application. Note that in this experiment only a single cue is used.

1D simulation. For this simulation, we choose a nonlinear and bimodal model: $x_t = 0.5x_{t-1} + 25\frac{x_{t-1}}{1+x_{t-1}^2} + 8\cos(1.2(t-1)) + u_t$ and $z_t = \frac{x_t^2}{20} + v_t$, where u_t is the system noise following a Gaussian $\mathcal{N}(u_t; 0, 0.6^2)$ and v_t is the measurement noise, also a Gaussian $\mathcal{N}(v_t; 0, 1^2)$. Data are generated with $x_0 = 0.1, t = 1 \ldots 100$.

In this simulation, we implement an augmented particle filter which uses an unscented Kalman filter as the first-stage filter to construct the proposal distribution. The parameters for scaled unscented transformation are $\alpha = 1, \beta = 0$, and $\kappa = 2$. These parameters are optimal for 1D applications [12]. 100 independent simulations are performed to compare the two algorithms. Only 10 particles are used for this simple 1D simulation. In Figure 4(a) we plot the 100 true states and the estimates obtained with the augmented particle filter and the conventional particle filter respectively. The Mean Square Error (MSE) between tryth and the estimates are shown in Figure 4(b) for the 100 simulations. The MSE and the variance of our algorithm is 6.095 and 1.049 respectively which are better than the generic particle filter (MSE mean 7.798 and variance 1.039).



(a)

(b)

Figure 4: (a) Plot of the true state and estimates generated by the conventional particle filter and the augmented particle filter algorithm. (b) Performance comparison of the conventional particle filter and the augmented particle filter. Mean Square Error is plotted for 100 random realisations. For most of the simulations, the augmented particle filter (solid line) outperforms the conventional particle filter (dashed line).

Visual Tracking. We have implemented the probabilistic colour based trackers as in [2]. This time for the augmented particle filter framework, the first-stage filter is a particle filter as well. For both trackers, the state is represented by a 4D vector, *i.e.* the location and scale of the tracked object. The calculation of the likelihood function is based on the colour histogram similarity measurement between the target and the candidate. In order to eliminate the affects of changing illumination, we calculate the histogram in the HSV colour space and only the values of HS are adopted. For 8-bit depth RGB input pictures, the dimension of the HS colour histogram is 16×16. Refer to [2, 4] for the implementation details.

For this video, the motion of the basketball is hard to model. 50 particles are used for the first-stage particle filter and 150 particles for the second-stage main particle filter. With 50 particles, the first-stage filter estimates the location of the basketball and provides a promising proposal distribution. Note that although only one iteration is adopted in the augmentation, encouraging improvements are observed over the generic particle filter.

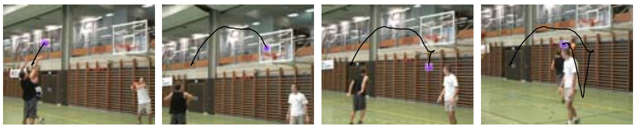


Figure 5: Basketball tracking. The frame numbers are $12, 36, 48, 72$ respectively out of a total of 78 frames. Our tracker tracks the whole sequences accurately. The black line indicates the basketball's moving trajectory. See the video at $www.cs.adelaide.edu.au/\sim vision/demo/$.

4. CONCLUSION

We have presented a more general augmented particle filter framework for visual tracking, which can flexibly integrate different filters and perform inference in a cooperative way. It is also easy to implement a multiple cue tracker similar to the Co-inference algorithm [3]. Experiments on synthetic data and real video sequences reveal its efficiency. Notice that the weight calculation in this framework involves a computationally expensive $\mathcal{O}(N^2)$ prior computation. We will modify the framework to avoid this calculation, making it more efficient with the importance sampling trick as in [9]. Future work will also test multiple cue tracking with more real videos in our framework.

References

[1] M. Isard and A. Blake. CONDENSATION – Conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.

[2] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. In *7th European Conference on Computer Vision*, volume 2350 of *Lecture Notes in Computer Science*, pages 661–675, Copenhagen, Denmark, 2002. Springer.

[3] Y. Wu and T. Huang. A co-inference approach to robust visual tracking. In *IEEE International Conference on Computer Vision*, volume 2, pages 26–33, Vancouver, Canada, July 2001.

[4] C. Shen, A. van den Hengel, and A. Dick. Probabilistic multiple cue integration for particle filter based tracking. In *International Conference on Digital Image Computing: Techniques and Applications*, volume 1, pages 399–408, Sydney, 2003.

[5] A. Doucet, S. Godsill, and C. Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10(3):197–208, 2000.

[6] T.-J. Cham and J. M. Rehg. A multiple hypothesis approach to figure tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 239–245, Fort Collins, Colorado, 1999.

[7] C. Chang and R. Ansari. Kernel particle filter: Iterative sampling for efficient visual tracking. In *IEEE International Conference on Image Processing*, Barcelona, Spain, 2003.

[8] C. Shan, Y. Wei, T. Tan, and F. Ojardias. Real time hand tracking by combining particle filtering and mean shift. In *6th International Conference on Automatic Face and Gesture Recognition*, Seoul, Korea, May 2004.

[9] M. Bray, E. Koller-Meier, and L. V. Gool. Smart particle filtering for 3d hand tracking. In *6th IEEE International Conference on Automatic Face and Gesture Recognition*, Seoul, Korea, May 2004.

[10] Y. Rui and Y. Chen. Better proposal distributions: Object tracking using unscented particle filter. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 786–793, Kauai, Hawaii, December 2001.

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

[12] R. van der Merwe, A. Doucet, N. de Freitas, and E. Wan. The unscented particle filter. Technical Report CUED/F-INFENG/TR 380, Cambridge University Department of Engineering, UK, May 2000. available at http://cslu.cse.ogi.edu/nsel/research/ukf.html.

[13] M. Isard and A. Blake. ICONDENSATION: Unifying low-level and high-level tracking in a stochastic framework. In *5th European Conference on Computer Vision*, volume 1, pages 893–908, 1998.

[14] J.H. Kotecha and P.M. Djurić. Gaussian particle filtering. *IEEE Transactions on Signal Processing*, 51(10):2592–2601, October 2003.