HYBRID PARTICLE FILTER AND MEAN SHIFT TRACKER WITH ADAPTIVE TRANSITION MODEL

Emilio Maggio and Andrea Cavallaro

Multimedia and Vision Laboratory
Queen Mary, University of London – Mile End Road, London, E1 4NS (United Kingdom)
Email: {emilio.maggio,andrea.cavallaro}@elec.qmul.ac.uk

ABSTRACT

We propose a tracking algorithm based on a combination of Particle Filter and Mean Shift, and enhanced with a new adaptive state transition model. Particle Filter is robust to partial and total occlusions, can deal with multi-modal *pdf*s and can recover lost tracks. However, its complexity dramatically increases with the dimensionality of the sampled *pdf*. Mean Shift has a low complexity, but is unable to deal with multi-modal *pdf*s. To overcome these problems, the proposed tracker first produces a smaller number of samples than Particle Filter and then shifts the samples toward a close local maximum using Mean Shift. The transition model predicts the state based on adaptive variances. Experimental results show that the combined tracker outperforms Particle Filter and Mean Shift in terms of accuracy in estimating the target size and position while generating 80% less samples than Particle Filter.

1. INTRODUCTION

Tracking algorithms can be classified into two major groups, namely *Target Representation and Localization* algorithms and *Filtering and Data Association* algorithms [1].

The Mean Shift (MS) algorithm [1] is a non-parametric method which belongs to the first group. MS is an iterative kernel-based deterministic procedure which converges to a local maximum of the measurement function under certain assumptions on the kernel behaviors. On the one hand, MS is a low complexity algorithm, which provides a general and reliable solution independently from the features representing the target. On the other hand, MS fails in tracking small and fast moving targets and in recovering a track after a total occlusion [1]. Particle Filter (PF) is a parametric method which belongs to the second group. PF solves non-linear and non-Gaussian state estimation problems [2, 3, 4] and can deal with multi-modal *pdf*s. The ability to recover from lost tracks makes PF one of the most used tracking algorithm. Unfortunately, PF is heavily dependent on the parameter settings which in turns depends on the scene content. Moreover, the number of particles needed to model the variations of the underlying *pdf* increases exponentially with the dimensionality of the state space, thus dramatically increasing the computational load.

In this paper, we propose a hybrid PF and MS tracking algorithm that overcomes the above mentioned drawbacks of MS and PF. Moreover, we use a new adaptive state transition model with updating variances. The update process is driven by the variability of the target in the previous frames. Hybrid methods have already been presented in the literature. In [5, 6] a mode detection MS procedure is applied over a continuous *pdf* generated from the particle weights. The novelty of our hybrid approach is that the tracker first generates the particles using a zero-order model and then uses an independent MS procedure that drives each particle in the position state sub-space.

The paper is organized as follows. Section 2 introduces the target model, existing tracking algorithms and the proposed hybrid tracking with adaptive state transition model. Experimental results are presented in Section 3. Finally, in Section 4 we draw the conclusions.

2. TWO STAGE HYBRID TRACKER WITH ADAPTIVE STATE TRANSITION MODEL

2.1. Target Representation

A widely used form of target representation is the color histograms, because of its independence from scaling and rotation and its robustness to partial occlusions [1, 3].

Let us define the target model as its normalized color histogram, $q = \{q_u\}_{1,\dots m}$, where m is the number of bins. The normalized color distribution of a target candidate $p(\mathbf{y}) = \{p_u(\mathbf{y})\}_{1,\dots m}$ centered in $\mathbf{y}$ can be calculated as

$$p_u(\mathbf{y}) = C_h \sum_{i=1}^{n_h} k\left(\left\| \frac{\mathbf{y}-\mathbf{x}_i}{h} \right\|^2\right) \delta\left[b(\mathbf{x}_i) - u\right], \qquad (1)$$

where $\{\mathbf{x}_i\}_{i=i,\dots n_h}$ are the n_h pixel locations of the target candidate in the target area, $b(\mathbf{x}_i)$ associates the pixel $\mathbf{x}_i$ to the histogram bin, $k(x)$ is the kernel profile with bandwidth h, and C_h is a normalization function defined as

$$C_h = \frac{1}{\sum_{i=1}^{n_h} k\left(\left\| \frac{\mathbf{y}-\mathbf{x}_i}{h} \right\|^2\right)}. \qquad (2)$$

The same equations are used to obtain the target model q.

In order to calculate the likelihood of a candidate we need a similarity function which defines a distance between the model and the candidate. A metric can be based on the Bhattacharryya coefficient [7], defined between two normalized histograms $p(\mathbf{y})$ and q as

$$\rho\left[p(\mathbf{y}), q\right] = \sum_{u=1}^{m} \sqrt{p_u(\mathbf{y}), q_u}. \qquad (3)$$

Hence we define the distance as

$$d\left[p(\mathbf{y}), q\right] = \sqrt{1 - \rho\left[p(\mathbf{y}), q\right]}. \qquad (4)$$

2.2. Mean Shift (MS)

The MS algorithm is an iterative process that aims at minimizing the distance in Eq. (4). The process is initialized with the location of the target in the previous frame, $\mathbf{y}_0$. The shape of the kernel is chosen so that the distance becomes a smooth function [1]. Then, based on gradient information, the MS algorithm converges to the nearest local minimum. Looking at Eq. (4) and Eq. (3), it is possible to notice that minimizing Eq. (4) corresponds to maximizing Eq. (3). To this end, by computing the Taylor expansion of the Bhattacharryya coefficient around the starting position $\mathbf{y}_0$ we obtain, using the Eq. (1), the following expression

$$\rho\left[p(\mathbf{y}), q\right] \approx \frac{1}{2}\sum_{u=1}^{m}\sqrt{p_u(\mathbf{y}_0), q_u} + \frac{C_h}{2}\sum_{i=1}^{n_h}\omega_i k\left(\left\|\frac{\mathbf{y}-\mathbf{x}_i}{h}\right\|^2\right). \tag{5}$$

In this form, the first term of the r.h.s. does not depend on $\mathbf{y}$. Therefore we need to minimize only the second term of Eq. (5). At each step of the iterative process, the estimated target moves from $\mathbf{y}_0$ to the new location $\mathbf{y}_1$, defined as

$$\mathbf{y}_1 = \frac{\sum_{i=1}^{n_h}\mathbf{x}_i\omega_i g\left(\left\|\frac{\mathbf{y}-\mathbf{x}_i}{h}\right\|^2\right)}{\sum_{i=1}^{n_h}\omega_i g\left(\left\|\frac{\mathbf{y}-\mathbf{x}_i}{h}\right\|^2\right)}. \tag{6}$$

If $g(x) = -k'(x)$, then $\mathbf{y}_1 - \mathbf{y}_0$ is in the gradient direction. The iterative process stops when $\|\mathbf{y}_1 - \mathbf{y}_0\| < \epsilon$. Usually $\epsilon = 1$ pixel.

In Eq. (6) it is possible to notice that the maximum area where the target can be correctly localized is the kernel size. For this reason, if the center of the object shifts more than this size in two consecutive frames, then the MS vector is no more correlated with the object itself and therefore the track is likely to be lost.

2.3. Particle Filter (PF)

The PF algorithm belongs to the *Filtering and Data Association* class of tracking algorithms. PF solves the tracking problem based on the state equation

$$\mathbf{x}_t = \mathbf{f}_t(\mathbf{x}_{t-1}, \mathbf{v}_t), \tag{7}$$

and on the measurement equation

$$\mathbf{z}_t = \mathbf{h}_t(\mathbf{x}_t, \mathbf{n}_t), \tag{8}$$

where f_t and h_t are non-linear and time-varying functions. $\{\mathbf{v}_t\}_{t=1,\ldots}$ and $\{\mathbf{n}_t\}_{t=1,\ldots}$ are assumed to be independent and identically distributed stochastic processes. The problem consists in calculating the *pdf* $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ at each time instant t. This *pdf* can be obtained recursively in two steps, namely prediction and update. The *prediction step* uses the state Eq. (7) to obtain the prior *pdf* as

$$p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})d\mathbf{x}_{t-1}, \tag{9}$$

with $p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})$ known from the previous iteration and $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ determinated by Eq. (7). When the measurement $\mathbf{z}_t$ is available, it is possible to perform the *update step* using the Bayes' rule

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = \frac{p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1})}{\int p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1})d\mathbf{x}_t}. \tag{10}$$

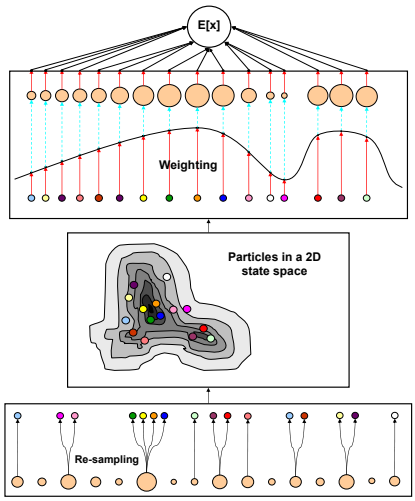

Fig. 1. Schematic representation of the Particle Filter algorithm

PF approximates the densities $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ with a sum of N_s Dirac functions centered in $\left\{\mathbf{x}_k^i\right\}_{i=1,\ldots N_s}$ as

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \sum_{i=1}^{N_s}\omega_t^i\delta\left(\mathbf{x}_t - \mathbf{x}_t^i\right), \tag{11}$$

where ω_t^i are the weights associated to the particles and are calculated as

$$\omega_t^i \propto \omega_{t-1}^i\frac{p(\mathbf{z}_t|\mathbf{x}_t^{i-1})p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i)}{q(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i, \mathbf{z}_t)}. \tag{12}$$

$q(.)$ is the importance density function which is often chosen to be $p(\mathbf{x}_t|\mathbf{x}_{t-1}^i)$ and this leads to $\omega_t^i \propto \omega_{t-1}^i p(\mathbf{z}_t|\mathbf{x}_t^{i-1})$.

A re-sampling algorithm can then be applied to avoid the degeneracy problem [2]. In this case the weights are set to $\omega_{t-1}^i = 1/N_s \;\forall\, i$, therefore

$$\omega_t^i \propto p(\mathbf{z}_t|\mathbf{x}_t^{i-1}). \tag{13}$$

The weights are therefore proportional to the likelihood function, i.e. to the Bhattacharryya coefficient ρ (Eq. (3)). Figure (1) shows an example of PF with re-sampling. The re-sampling step derives the particles depending on the weights of the previous step, then all the new particles receive a starting weight equal to $1/N_s$ which will be updated by the next frame likelihood function.

The best state at the time t is derived based on the discrete approximation of Eq. (11). The most common solution is the Monte Carlo approximation of the expectation

$$\mathbb{E}[\mathbf{x}_t|\mathbf{z}_{1:t}] \approx \frac{1}{N_s}\sum_{i=1}^{N_s}\omega_t^i\mathbf{x}_t^i. \tag{14}$$

PF is much more complex than MS and, *if the parameter are set properly*, it can track fast small targets and can recover a track even after a total occlusion.

The major limit of PF is the limited capability of the particles to describe the *pdf* when the state space is not densely sampled. To overcome this problem, a large number of particles is required thus increasing the computational load.

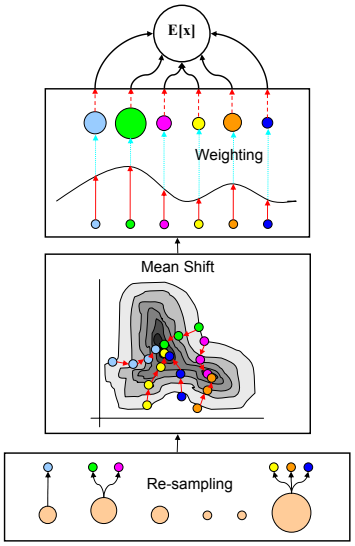

Fig. 2. Schematic representation of the Two Stage Hybrid Tracker

In the next section we propose an alternative method which requires about 80% less samples than PF while improving its performance.

2.4. Hybrid Tracker (TSHT)

We can divide the proposed tracker in four main steps (Fig. (2)). The *first step* generates particles using Eq. (7) and re-samples them; the *second step* applies MS independently to each particle until all particles have reached a stable position. The *third step* recalculates the weights using the Bhattacharyya coefficients. Finally, the *fourth step* calculates the weighted average to obtain the best state as in Eq. (14).

The selection of a state transition model is an important issue for the *first step* of the proposed algorithm. Most of the PF approaches learn the model from a training set or based on the previous frames. These solutions are often implemented using first- or second-order models [3]. The posterior probability (Eq. 11) is well approximated when the particles are concentrated in a small area. However, if the concentration of the particles is too high, the flexibility of the algorithm is reduced and it becomes difficult to deal with maneuvering targets. A first attempt to overcome this limit using an adaptive motion model was presented in [4]. Here a first-order model is coupled with adaptive Gaussian noises dependent on the *state prediction error* in the previous frame.

In order to stabilize the behavior of PF in the presence of false targets and to handle unexpected events like fast changes in target speed or direction, we employ a zero-order adaptive model that learns the *variability of the target state*. This model is based on calculating the average state velocity in the previous k frames as

$$\mathbb{E}_k[\mathbf{\Delta x}] = \frac{1}{k}\sum_{n=t-k}^{t} |\mathbf{x}_n - \mathbf{x}_{n-1}|\,, \tag{15}$$

therefore the state transition model can be defined as

$$\mathbf{x}_t = \mathbf{x}_{t-1} + \mathbf{C}_t\mathbf{v}_t, \tag{16}$$

where $\mathbf{C}_t \propto \mathbb{E}_k[\mathbf{\Delta x}]$, and $\mathbf{v}_t = \mathcal{N}(0, I)$ is jointly Gaussian.

We can better explain the *second step* of the algorithm defining the Mean Shift operator $\mathbb{MS} : R^d \rightarrow R^d$, where d is the state space dimensionality. Now we can rewrite Eq. (11) as

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \sum_{i=1}^{N_s} \omega_t^i \delta\left(\mathbf{x}_t - \mathbb{MS}(\mathbf{x}_t^i)\right). \tag{17}$$

Note that $\mathbb{MS}$ is a d-dimension operator, but in the proposed algorithm it operates on the bi-dimensional position sub-space only. Each MS procedure guides the generated particle over the sub-space independently from all the others.

The *third* and *fourth step* are the same as in the standard PF, but operate on particles that are close to local maxima. Therefore, the approximation of Eq. (14) is performed using samples better representing the state space than those generated by PF. For this reason, these particles are more efficient than those in the standard PF. The TSTH algorithm is more than a simple multiple initialization of MS since the particles are filtered and selected by PF depending on their likelihood. This feature gives to TSHT the possibility to treat multi-modal *pdf*s.

3. EXPERIMENTAL RESULTS

In this section we present the comparison between PF, MS, and TSHT. Figure (3) shows sample results from the test sequences *Highway*, *Table tennis*, and *Soccer*. All the sequences are in CIF at 30 Hz, except *Highway* which is at 25 Hz. The ground truth information of the coordinates and sizes of each target has been generated. The three algorithms under test are initialized using the ground truth. The histograms are calculated in the RGB space with 10x10x10 bins. MS runs 5 times with different kernel sizes up to +/-10% that of the previous frame. In order to have a fair comparison with MS, we use a 3-dimensional state model for PF and TSHT. The state model is composed of the target position, (x, y), and the target size, h. PF uses a zero-order motion model with fixed $\sigma_x = \sigma_y = 5$ and $\sigma_h = 0.05$. (Note that the scale change is a percent while the position is in pixels.) The initial values for the adaptive state transition model are $\sigma_x^0 = \sigma_y^0 = 14$ and $\sigma_h^0 = 0.13$; $k_p = 10$ frames for the position and $k_s = 5$ frames for the size. PF uses 150 samples and TSHC uses 30 samples only.

In Fig. (3)(a) MS and TSHT successfully track the car, whereas PF loses the target because the values of the variances are too large for the object behavior. This problem is not encountered by TSHT thanks to the adaptive state transition model. In Fig. (3)(b) TSHT only succeeds in maintaining the track of the ball. PF recovers the target position for few frames after the bouncing of the ball but then fails. A comparison of the tracking accuracy of the three algorithms for *Table tennis* is reported in Fig. (4). In Fig. (3)(c) MS is not able to track the ball kicked by the player. TSHT is faster than PF in reacting to the abrupt shift of the ball, and it is better in recovering the ball size (frame 54). In *Soccer* and *Table tennis* the target is moving in unexpected directions with shifts that are larger than the kernel size, moreover during the fast movements the object shape is affected by camera blur, hence the effectiveness of the MS vector is decreased. In TSHT the multiple MS initialization created by PF solves the problem. Table (1) summarizes the results presented in this section. TSHT achieves better performance than PF for both position and size recovery. Furthermore, TSHT is 32% faster than PF. MS looses the track in *Table tennis* and *Soccer* even though it is 46% faster than TSHT. Hence at the price of a slower speed TSHT achives an improved reliability.

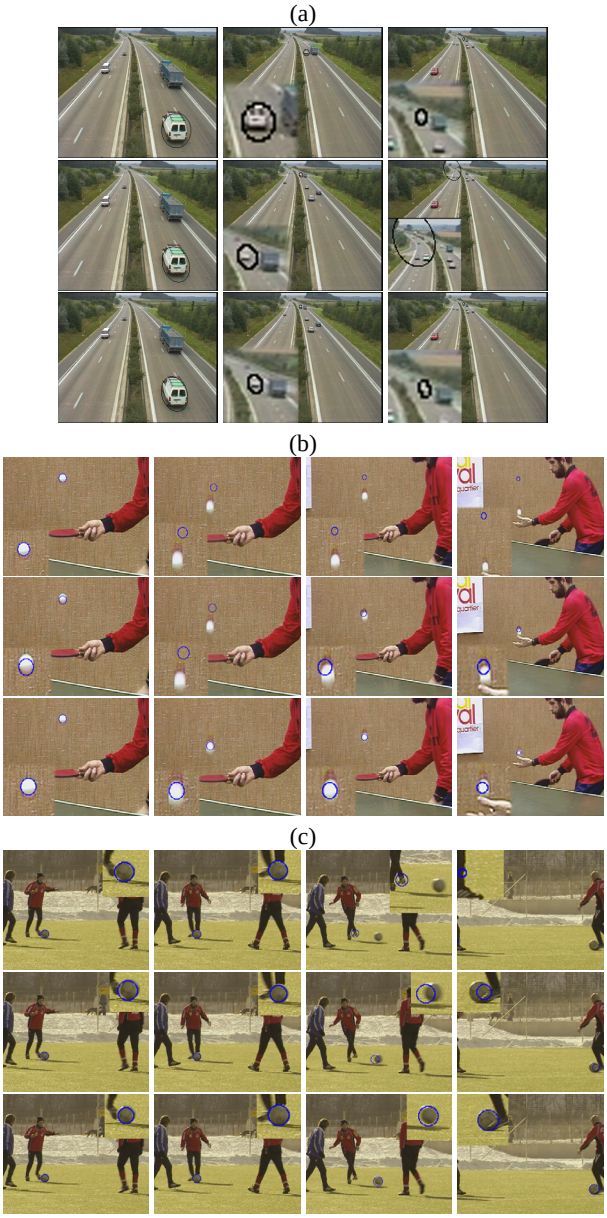

Fig. 3. Comparison of tracking performance between MS,PF and TSHT. (a) *Highway* (frames 101, 245, 301), (b) *Table tennis* (frames 1, 9, 37, 53) (c) *Soccer* (frames 1, 7, 13, 54). In each group of pictures, the top row reports the results of MS, the middle row the results of PF, and the bottom row the results of TSHT. The videos with the results are available at http://www.elec.qmul.ac.uk/staffinfo/andrea/TSHT.html

4. CONCLUSIONS

We presented a hybrid tracking algorithm based on a combination of Particle Filter and Mean Shift with an adaptive state transition model. The Mean Shift is inserted in a Particle Filter framework in order to make each particle independent and therefore more flexible to local conditions. The adaptive state transition model stabilizes the Particle Filter behavior. This effect is obtained by adapting the searched area of the state space to the average state velocity. Experimental results showed that the proposed algorithm is faster and more accurate than Particle Filter and more reliable than Mean Shift with fast moving objects. Future work includes investigating a new target representations scheme with spatial information.

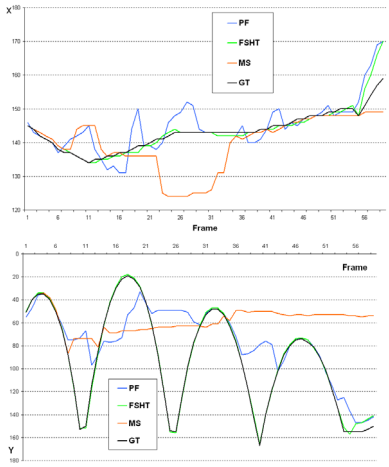

Fig. 4. Comparison of tracking accuracy for *Table tennis*.

		MS	PF	TSHT
"Highway"	APE	0.95	12.8(TL)	0.88
	ASE	2.74	22.3(TL)	3.58
"Table tennis"	APE	43.2(TL)	24.1(TL)	2.0
	ASE	6.7(TL)	3.3(TL)	2.8
"Soccer"	APE	242(TL)	3.9	3.2
	ASE	18.2(TL)	10.8	9.8

Table 1. Deviation from the ground truth of the three trackers. TL: Track Lost; APE: Average Position Error (pixels), ASE: Average Size Error. ASE=$\sqrt{W^2 + H^2}$. W: Width Error, H: Height Error.

5. REFERENCES

[1] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564–577, May 2003.

[2] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, Feb. 2002.

[3] K. Nummiaro, E. Koller-Meier, and L. Van Gool, "A color-based particle filter," in *Proc.of the 1st Workshop on Generative-Model-Based Vision*, June 2002, pp. 53–60.

[4] S. Zhou, R. Chellappa, and B. Moghaddam, "Appearance tracking using adaptive models in a particle filter," in *Proc. of Asian Conf. on Computer Vision*, June 2002.

[5] B. Han, D. Comaniciu, Y. Zhu, and L. Davis, "Incremental density approximation and kernel-based bayesian filtering for object tracking," in *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, June – July 2004, vol. 1, pp. 638–644.

[6] C. Chang and R. Ansari, "Kernel particle filter: iterative sampling for efficient visual tracking," in *Proc. of IEEE International Conf. on Image Processing*, Sept. 2003, vol. 3, pp. III–977–80.

[7] T.Kailath, "The divergence and bhattacharryya distance measures in signal selection," *IEEE Trans. Comm. Technology*, vol. 15, pp. 52–60, 1967.