Adaptive Rao–Blackwellized Particle Filter and Its Evaluation for Tracking in Surveillance

Xinyu Xu and Baoxin Li, *Senior Member, IEEE*

***Abstract*—Particle filters can become quite inefficient when being applied to a high-dimensional state space since a prohibitively large number of samples may be required to approximate the underlying density functions with desired accuracy. In this paper, by proposing an adaptive Rao–Blackwellized particle filter for tracking in surveillance, we show how to exploit the analytical relationship among state variables to improve the efficiency and accuracy of a regular particle filter. Essentially, the distributions of the linear variables are updated analytically using a Kalman filter which is associated with each particle in a particle filtering framework. Experiments and detailed performance analysis using both simulated data and real video sequences reveal that the proposed method results in more accurate tracking than a regular particle filter.**

***Index Terms*—Particle filter, Rao–Blackwellization, video-based surveillance, visual tracking.**

I. Introduction

VISUAL tracking is an important step in many practical applications including video-based surveillance. In recent years, particle-filter-based visual tracking has been extensively studied (e.g., [1]–[5]). Particle filtering has been shown to offer improvements in performance over some conventional methods such as the Kalman filter in nonlinear/non-Gaussian environments [6]. However, the large number of samples required to approximate the posterior density render its use difficult in high-dimensional state spaces.

In some cases, the system model may have a "tractable structure" such as with some components having linear dynamics that can be analytically estimated using exact filters conditional on other components in a sequential Monte Carlo (SMC) framework [7], [8] like particle filtering. In these cases, exact filters like Kalman filter, the HMM filter, or any other finite-dimensional optimal filters [10], could be exploited to marginalize out the linear dynamics. This technique is referred to as Rao–Blackwellization [11]. The resultant method is often called Rao–Blackwellized particle filter (RBPF).

Generally, suppose that we have an estimator $\zeta(R, L)$ depending upon two variables R and L, the Rao–Blackwell Theorem reveals that its variance satisfies [11]

$$\mathrm{Var}\left[\zeta(R,L)\right] = \mathrm{Var}\{E\left[\zeta(R,L)|R\right]\} + E\left\{\mathrm{Var}[\zeta(R,L)|R]\right\}. \quad (1)$$

Manuscript received November 7, 2005; revised September 11, 2006. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Zhigang (Zeke) Fan.

The authors are with the Department of Computer Science and Engineering, Arizona State University, Tempe, AZ 85287 USA (e-mail: xinyu.xu@asu.edu; baoxin.li@asu.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2007.891074

Since $E\{\mathrm{Var}[\zeta(R,L)|R]\}$ is non-negative, the variance of the estimator $\zeta' = E[\zeta(R,L)|R]$ is less than that of the original estimator $\zeta(R,L)$. The formal justification can be found in, for example, [8], [24]. One can interpret the Rao–Blackwell theorem by saying that the estimator obtained by the calculation of conditional expectation $E[\zeta(R,L)|R]$ is superior to the original one $\zeta(R,L)$, and the superiority manifests in the reduction in the variance of the estimates.

For the visual tracking problem, let X_t denote the state to be estimated and Z_t the observation, with subscript t the time index. The key idea of RBPF is to partition the original state-space X_t into two parts R_t (*root variables*) and L_t (*leaf variables*), such that $p(L_{1:t}|R_{1:t}, Z_{1:t})$ is a distribution that can be computed exactly conditional on the root variables, and the distribution $p(R_{1:t}|Z_{1:t})$ will be estimated using Monte Carlo methods such as particle filtering. The justification for this decomposition follows from the factorization of the posterior probability [10]

$$p(R_{1:t}, L_{1:t}|Z_{1:t}) = p(L_{1:t}|R_{1:t}, Z_{1:t})p(R_{1:t}|Z_{1:t}). \quad (2)$$

If the same number of particles is used in a regular particle filter and a RBPF, intuitively, the latter will provide better estimates for two reasons: first, the dimension of $p(R_{1:t}|Z_{1:t})$ is smaller than $p(R_{1:t}, L_{1:t}|Z_{1:t})$; second, optimal algorithms may be used to estimate the "tractable substructure." Reference [10] shows that $p(L_{1:t}|R_{1:t}, Z_{1:t})$ can be efficiently updated using Kalman filter when the initial uncertainty for leaves is Gaussian, and the conditional probability distributions of the observation model and system dynamics for the leaves are linear functions of the leaf states. In this paper, we will show how Kalman filter is combined with particle filtering to facilitate tracking in a surveillance application, with a physically meaningful dependency model for the relationship between the leaf and the root variables.

RBPF has been applied in some state estimation problems. For example, in [12], RBPF is used to integrate out the subspace coefficients in an Eigen Tracking problem. In the problem of tracking multiple people using a combination of anonymous and id sensors [13], RBPF is used to estimate the locations and identities of multiple objects, with Kalman filter being used to track an individual person. In [14], Freitas *et al.* combines Kalman filter with particle filter for fault diagnosis for a mobile robot, where Kalman filters are applied over continuous states and the samples are obtained over discrete states. In [15], the nonlinear ball motion model and robot location are tracked using particle filter while ball location and velocity are estimated by Kalman filter. In integrated aircraft navigation [16], Kalman filter is used to track the velocity components, and particle filtering is used to

estimate the position components. A more generalized discussion regarding marginalized particle filter for mixed linear/nonlinear state-space models can be found in [9].

Although RBPF has been studied in the aforementioned application areas, its application in tracking for surveillance has yet to be fully explored. In particular, we believe that a thorough quantitative performance comparison between RBPF and a regular particle filter (PF) will help us to understand the advantages of RBPF. The key contribution of this paper is, thus, two fold. First, with video-based surveillance as a case study, we utilize the constraints imposed by typical camera-scene configuration to partition the original state space into two subspaces, and a RBPF algorithm is proposed for tracking in surveillance. Second, we carry out thorough evaluation of the proposed RBPF algorithm in comparison with a regular PF [3] through experiments with both simulated data and real video. Our experiments show that the improvements of the proposed RBPF over a regular PF manifest in four aspects: increased estimation accuracy, reduced estimates variance, reduced number of particles required to achieve the same level of accuracy, and reduced weight variances.

The remaining of the paper is organized as follows. Section II presents the proposed algorithm. In Section III, we evaluate the performance of the proposed algorithm with comparison to a regular PF using simulated data. Section IV presents the tracking results on real video sequences. In Section V, related work on dimension reduction in visual tracking problems is discussed. Some important discussions are presented in Section VI, and, finally, we conclude in Section VII.

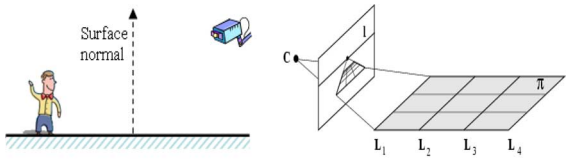
II. RBPF for Tracking in Surveillance

A. Partition the State Space

In typical surveillance applications, most of the time, the tracked objects are constrained to move on a dominant plane (e.g., the ground), and the camera is usually higher than the tracked object. Fig. 1 illustrates such a camera-scene configuration, where (b) is a geometric representation of (a). In Fig. 1(b), suppose a person is walking on the ground plane π, the ground is projected onto the image plane by camera C, with l the vanishing line for the ground plane. Any scene point projected onto the vanishing line l is at the same distance from plane π as the camera center [17]. If a scene point is farther from π than the camera is, then its image lies "above" the vanishing line and "below" if it is closer to the ground than the camera. So, if the moving object is not higher than the camera, the image of the object will always lie below the vanishing line, and when it moves towards the camera, the scale of the object on the image will get bigger as the y coordinate on image plane gets bigger, and vice versa. Fig. 1(b) clearly shows the dependence of the *scale change* on the y coordinate *change* in the image domain. In these situations, the constraints imposed by the camera-scene configuration can be exploited to deduce the dependency relationship among the state variables.

Formally, in our work we use the following 8-D ellipse model to describe the dynamics of the target (like [18])

$$\{x, \Delta x, y, \Delta y, H, \Delta H, W, \Delta W\}$$



(a) (b)

Fig. 1. (a) Typical camera configuration in surveillance. (b) The dominant plane is projected onto the image plane. If the person moves towards the camera, the image domain y coordinate and the size of the person both become larger.

where (x, y) represents the center of the ellipse, $(\Delta x, \Delta y)$ the motion velocity, (H, W) the horizontal and vertical half length of the ellipse axes, and $(\Delta H, \Delta W)$ the corresponding rates of scale change on the axes. With the above idea, the scale change of a moving object is related to its position along the y axis, so that the scale change can be estimated conditional on the location components. This facilitates the partition of the original 8-D state space into two groups: the root variables R containing the motion information and the leaf variables L consisting of the scale parameters, which are denoted by

$$R = \{x, \Delta x, y, \Delta y\}, \quad L = \{H, \Delta H, W, \Delta W\}.$$

B. Overview of the Method

In this work, the root variables are propagated by a first order system motion model defined by

$$R_t = TR_{t-1} + \boldsymbol{n}_{t-1} \tag{3}$$

where T is the transition matrix and $\boldsymbol{n}_{t-1}$ is random noise. Conditional on the root variables, the leaf variables forms a linear-Gaussian substructure specified by

$$L_t = AL_{t-1} + f(R_t, R_{t-1}) + \tau_{t-1}, \quad p(\tau) \sim N(0, P) \tag{4}$$

where f is a function encoding the conditional relation of L on R from $t-1$ to t. The image observations used for computing likelihood for both linear states and nonlinear states, denoted by Z_t, are the color histogram within a sample ellipse and the intensity gradients along the ellipse boundary. Since both the color histogram and gradient cues do not follow a linear-Gaussian relationship with the state variables, so the observation model is give in a general form

$$Z_t = \Psi(R_t, L_{t,}, \boldsymbol{m}_t) \tag{5}$$

where $\boldsymbol{m}_t$ is random noise and Ψ a nonlinear function. Due to the nonlinear measurement relation with the states and the non-Gaussian noise in measurements from the image, the object's states are typically better modeled by probabilistic multimodal densities; hence, we use particle filter for sampling the posterior density of the state variables.

Once the root variables are propagated one step ahead, the leaf variables can be computed by making use of the dependency between the leaf and the root using a Kalman filter. The reason why Kalman filter can be used is that: a) conditional on the root variables, the leaf variables form a linear-Gaussian substructure specified by (4); b) we introduce auxiliary observations O_t, which are simply (H, W), to serve as the observations used

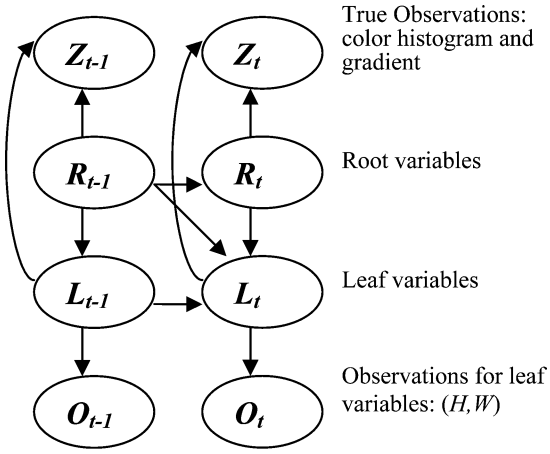


Fig. 2. Relationship among various components in the modeling, similar to a Dynamic Bayesian Network.

in the Kalman filter. Thus, the observations O_t form a linear relationship with state L

$$O_t = CL_t + \varsigma_t \quad p(\varsigma) \sim N(0, Q). \tag{6}$$

c) We assume that the system and observation model for the leaf variables are driven by Gaussian random noise $p(\tau) \sim N(0, P)$ and $p(\varsigma) \sim N(0, Q)$ with P and Q the system and the measurement noise covariance, respectively.

Fig. 2 illustrates the relationships between the state variables and the observations in a structure similar to a dynamic Bayesian network. The object motion at time t, R_t, only depends on the previous motion R_{t-1} as the motion evolution is described by a Markov process of order one. The scale of the ellipse at time t, L_t, depends on previous ellipse scale, L_{t-1}, previous object motion R_{t-1} and current motion R_t. The observations for the leaf at time t, O_t, only depends on current scale L_t. The true observations for deriving sample weight, Z_t, depends on both current object motion, R_t, and current scale L_t.

C. Proposed Algorithm

Fig. 3 illustrates the proposed RBPF algorithm. Just like regular particle filters, RBPF represents the posterior density by a set of weighted particles: $S_t = \{s_t^i, w_t^i | 1 \le i \le N\}$. Each particle maintains not only a sample from $p(R_t|Z_t)$, which we denote by R_t^i, but also a parametric representation of the distribution $p(L_t|R_t^i, Z_t)$, which consists of the mean vector of the leaf variables, $\mu_t^i = E[L_t]$, and the estimation error covariance for leaves $\sigma_t^i = E[(L_t^i - \mu_t^i)(L_t^i - \mu_t^i)]^T$ [10]. So each particle is represented by a triplet $s_t^i = \langle R_t^i, \mu_t^i, \sigma_t^i \rangle$. The proposed RBPF algorithm will sample the nonlinear non-Gaussian motion R_t^i using particle filter, while apply Kalman filter to estimate the scale parameters μ_t^i and σ_t^i conditional on the motion state. In the following, we detail the steps of the algorithm.

1) ***Propagate samples***

a) ***Sample the object motion*** according to

$$R_t^{i-} : p\left(R_t|R_{t-1}^i, Z_t\right) = p(Z_t|R_t)\, p\left(R_t|R_{t-1}^i\right). \tag{7}$$

The samples are propagated at each time step by (3). Most existing tracking algorithms, be the regular particle filter [3], ICondensation [19], or Auxiliary PF [20], the variance of $\boldsymbol{n}_t$ is typically static, making it difficult to track objects under dramatic and fast tracking motion. In the proposed RBPF algorithm, the variance of $\boldsymbol{n}_t$ is made adaptive, the detail will be presented at algorithm Step 6).

After this step, we have $s_t^i = \langle R_t^{i-}, *, * \rangle$. The minus sign in the superscript denotes that the corresponding variable is *a priori* estimate and $*$ denotes an un-initialized value. In a regular PF, immediately after this step, we are supposed to weight each sample by the observation likelihood $p(Z_t|R_t)$, but here, Kalman prediction is performed before weighting each sample, which is intended to analytically predict a new mean and covariance for the leaf variables by making use of the dependency between L and R. See Step b) for details.

b) ***Kalman prediction for leaf states*** according to

$$L_t^{i-} : p\left(L_t|R_t^i, R_{t-1}^i, L_{t-1}^i, Z_t\right). \tag{8}$$

According to the Kalman filter model defined by (4) and (6), we project forward the state and error covariance using (9). The first four formulas in (9) perform prediction for the mean of the leaf variables, and the last two are the covariance and the observation prediction, respectively. The first formula means that each time the y coordinate of the object increases β pixels from $t-1$ to t, the half length of ellipse vertical axis will be increased α pixels accordingly. Note that, here, the predictions for the leaf variables have exploited the conditional relations between L and R, i.e., the scale increase from $t-1$ to t depends on the y coordinate increase from $t-1$ to t, and the location prediction y_t^{i-} sampled in Step a) has also been made use of. In the third formula, the scale change will keep the aspect ratio during tracking. In practice, parameter α and β should be adjusted according to the angle between the ground plane and the image plane: the larger the angle, and the higher the uncertainty of the scale change conditional on the location. The two parameters α and β influence the estimation accuracy of the proposed algorithm as they control the change rate of the sample ellipse size with respect to the object motion. The sensitivity of the estimation accuracy to the dependency model will be further discussed in Section III-G. After this step, we have $s_t^i = \langle R_t^{i-}, \mu_t^{i-}, \sigma_t^{i-} \rangle$

$$\begin{aligned}
H_t^{i-} &= H_{t-1}^i + \alpha\left(y_t^{i-} - y_{t-1}^i\right)/\beta \\
\Delta H_t^{i-} &\sim N(\mu_{\Delta H}, \sigma_{\Delta H}) \\
W_t^{i-} &= \frac{H_t^{i-}}{H_{t-1}^i} W_{t-1}^i \\
\Delta W_t^{i-} &\sim N(\mu_{\Delta W}, \sigma_{\Delta W}) \\
\sigma_t^{i-} &= A\sigma_{t-1}^i A^T + P \\
O_t^{i-} &= C\mu_t^{i-}.
\end{aligned} \tag{9}$$

2) ***Evaluate weight*** for each particle: $w_t^i = p(Z_t|R_t^{i-}, \mu_t^{i-}, Z_{1:t-1})$.

a) ***Compute the color histogram*** for each sample ellipse Γ characterized by ellipse center $c = (x_t^{i-}, y_t^{i-})$ and scale (H_t^{i-}, W_t^{i-})

$$p_c^{(u)} = f \sum_{\theta_i \in \Gamma} k\left(\frac{\|c - \theta_i\|}{a}\right) \delta\left[h(\theta_i) - u\right] \tag{10a}$$

where δ is the Kronecker delta function and $h(\theta_i)$ assigns one of the m bins of the histogram to a given color u at location θ_i. Pixels that are closer to the region center are given higher weights specified by

$$k(d) = \left\{ \begin{matrix} 1 - d^2 : d < 1 \\ 0, \text{ otherwise} \end{matrix} \right\}. \tag{10b}$$

This is the same model as proposed in [18].

b) ***Compute the gradient*** for each sample ellipse Γ characterized by ellipse center $c = (x_t^{i-}, y_t^{i-})$ and scale (H_t^{i-}, W_t^{i-}). The gradient of a sample ellipse is computed as an average over gradients of all the pixels on the boundary

$$g(\Gamma) = \frac{1}{N_\Gamma} \sum_{i=1}^{N_\Gamma} g(x_i, y_i) \tag{11a}$$

where the gradient at pixel (x_i, y_i) is set to the maximum gradient by a local search along the normal line (l_n) of the ellipse at location (x_i, y_i)

$$g(x_i, y_i) = \max_{(x_n, y_n) \in l_n} \left\{ g(x_n, y_n) \right\}. \tag{11b}$$

A simple operator is used to compute the gradient in x axis and y axis for pixel (x_n, y_n)

$$\begin{aligned} g_x(x_n, y_n) = {} & I(x_n - 2, y_n) + 2I(x_n - 1, y_n) \\ & - 2I(x_n + 1, y_n) - I(x_n + 2, y_n) \\ g_y(x_n, y_n) = {} & I(x_n, y_n - 2) + 2I(x_n, y_n - 1) \\ & - 2I(x_n, y_n + 1) - I(x_n, y_n + 2) \end{aligned}$$

and, finally, the gradient at point (x_n, y_n) is computed as

$$g(x_n, y_n) = \sqrt{g_x^2(x_n, y_n) + g_y^2(x_n, y_n)}. \tag{11c}$$

c) ***Compute the weight***. We first compute two weights for each sample. One is based on color histogram similarity between the hypothetical region and the target model

$$G_c^i = \frac{1}{\sqrt{2\pi}\sigma_c} \exp\left(- \frac{\left(1 - \rho\left[p_{\Gamma_i}, q\right]^2\right)}{2\sigma_c^2} \right) \tag{12a}$$

where

$$\rho[p, q] = \sum_{u=1}^{m} \sqrt{p^{(u)} q^{(u)}} \tag{12b}$$

and p stands for the color histogram of a sample hypothesis in the newly observed image, and q represents the color histogram of the target model. Equation (12b) essentially gives the similarity between the target histogram and the hypotheses histogram measured by Bhattacharyya coefficient. The larger ρ is, the more similar the distributions are $(0 \le \rho \le 1)$. Another weight is based on the gradient $g(\Gamma_i)$

$$G_g^i = \frac{1}{\sqrt{2\pi}\sigma_g} \exp\left(- \frac{\left(1 - g(\Gamma_i)^2\right)}{2\sigma_g^2} \right). \tag{12c}$$

Note that, before feeding the gradient into a Gaussian distribution to get a gradient weight, the gradient of a sample ellipse is divided by the maximum gradient of all the samples to normalize it into range [0, 1].

The final weight for each sample is given by

$$w_t^i = \eta G_c^i + (1 - \eta) G_g^i \quad \text{with } \eta = 0.5. \tag{12d}$$

If we assume that the color histogram cue is independent of gradient cue, the final weight could also be computed as $w_t^i = G_c^i \bullet G_g^i$.

As pointed out by the author in [9], only one Riccati instead of N Riccati recursions is needed if the linear states do not appear in the measurement relation, which will lead to a substantial reduction in the computation time, but this is not the case in our problem, because from Step 2a)–c), we see that both the predicted nonlinear states (x_t^{i-}, y_t^{i-}) and the linear states (H_t^{i-}, W_t^{i-}) have contributed to the computation of the weights, that is, both the nonlinear states and the linear states will appear in the measurement (5). Consequently, N Riccati recursions should be used in the Kalman time and measurement update (9) and (13). We think that this is why the computation cost of our proposed algorithm is slightly higher than the standard PF, as will be discussed in Section VI-B. Also, since both the predicted nonlinear states and linear states are involved in obtaining the weights, the weights represent the conditional likelihood of not only the nonlinear but also the linear states, i.e., the weights signify how well the predicted states "match" with the true measurement.

3) ***Select samples***. Resampling with replacement $\mathbf{Pr}(\langle R_t^i, \mu_t^{i-}, \sigma_t^{i-} \rangle = \langle R_t^{j-}, \mu_t^{j-}, \sigma_t^{j-} \rangle) = w_t^j$. The latest measurements will be used to modify the prediction PDF of not only the root variables but also the leaf variables. After this step, $s_t^i = \langle R_t^i, \mu_t^{i-}, \sigma_t^{i-} \rangle$.

Some important remarks are due regarding to Steps 2) and 3). A common problem with the particle filter is the degeneracy problem: after a few iterations, some of the particle weights may become dominantly larger than other negligible weights. In the literature, there are two primary methods to overcome this problem [2]: one is a good choice of importance density [3], [21], the other is the use of resampling [3], [22], [23]. Although resampling can effectively lessen the degeneracy problem in low-dimensional state estimation, its capacity for alleviating degeneracy in high-dimensional state estimation becomes very limited. In Section IV, we will show that Rao–Blackwell technique can do a very good job in reducing degeneracy in such a situation.

4) ***Kalman update*** for leaf variables. Kalman update is accomplished by (13a)–(13c) over the selected sample set. K_t^i is the Kalman gain which aims at minimizing the posterior error covariance. As the measurement error covariance Q approaches zero, the Kalman gain weights the measurement O_{t-1} more heavily while the predicted measurement $C\mu_t^{i-}$ is "trusted" less and less. On the other hand. as the *a priori* estimate error covariance σ_t^{i-} approaches zero, the predicted measurement is trusted more than the measurement. Equation (13b) incorporates a measurement O_{t-1} into the *a priori* leaf state estimate to obtain an improved *a posteriori* leaf state estimate. One may question the feasibility of $O_{t-1} = \{H, W\}_{t-1}$ acting as observations since $\{H, W\}$ are calculated from their previous values and root variables in (9). This is practically not a problem since O_{t-1} has gone through the entire estimation process and, thus, already incorporates the information in the root variables and the true measurements (color histogram and gradient). Essentially, the auxiliary observation $O_{t-1} = \{H, W\}_{t-1}$ can be viewed as being indirectly linked to the root variables and the true measurements at $t-1$ through a non-Gaussian nonlinear function $O_t = F(R_t, Z_t)$

$$K_t^i = \sigma_t^{i-} C^T \left(C\sigma_t^{i-} C^T + Q\right)^{-1} \tag{13a}$$

$$\mu_t^i = \mu_t^{i-} + K_t^i \left(O_{t-1} - C\mu_t^{i-}\right) \tag{13b}$$

$$\sigma_t^i = \sigma_t^{i-} - K_t^i C\sigma_t^{i-}. \tag{13c}$$

After this step, we have $s_t^i = \langle R_t^i, \mu_t^i, \sigma_t^i \rangle$. Note that if the leaf prediction uncertainty is large (e.g., σ_t^{i-} is large), the Kalman update should be done immediately after the Kalman prediction step, so that the inaccurate prediction of leaf variables can be immediately corrected by incorporating a new observation. On the other hand, if the leaf prediction uncertainty is small, but the dynamic motion model (3) is not accurate, then Kalman update is due here so that the Kalman update operates on only those "good samples" (those samples in the vicinity of the true object location) which have been selected by the **Selection** step.

5) ***Compute the mean state*** at time t. Since resampling has been done by **Selection**, so, now, the mean state can be simply computed as the average of the state particles

$$E[S_t] = \sum_{i=1}^{N} s_t^i \Big/ N. \tag{14}$$

6) ***Compute the new noise variance***

If fixed-noise variance is employed in the system model to track objects moving with dramatically and fast changing velocity, the range of the samples may not be sufficient to cover a large motion change. This argument has been confirmed by the following simulations. In these simulations, a point moving on a 2-D plane with a certain velocity generates an actual path; noise is then added to the actual path to simulate a noisy measurement of the actual path. After that the proposed RBPF with Steps 1)–5) is invoked to track the actual path. We then measure the estimation mean-square error (MSE) for horizontal velocity under different noise variances for three motion settings: moving with small constant velocity, moving with nonconstant velocity within a small range, moving with dramatically changing velocity. We found that when the velocity is small and constant, we only need a small noise variance to reach the smallest MSE, but if the velocity changes dramatically, we need a much larger noise variance to reach the lowest MSE.

These experiments show that adaptive noise is needed to track objects with dramatically and fast changing motion. Intuitively, the noise variance should be proportional to the prediction error (since it determines the quality of tracking). If the prediction error is small, we only need noise with small variance to absorb the residual motion; if prediction error is large, we need a large noise variance to cover potentially large variations in the system state. In our work, the noise $\boldsymbol{n}_t$ assumes a normal distribution $\boldsymbol{n}_t \sim N(\mathbf{0}, \boldsymbol{\varepsilon}_t)$, where $\boldsymbol{\varepsilon}_t$ denotes the noise variance that needs to be made adaptive.

In a tracking problem, it is hard to directly compute the prediction error since we never know the true states of the target object. So, in both the real data experiment and the simulations, a similarity measure φ_t is computed which is inversely proportional to the prediction error. Intuitively, if this similarity measure is large, it means that with the current noise variance the algorithm still maintains a good tracking, and, thus, we only need to adjust the variance a little bit. Otherwise, if the similarity measure is small, this may well be due to the object's dramatic velocity change that has resulted in poor tracking, and, thus, the noise variance needs to be enlarged. Hence, the noise variance $\boldsymbol{\varepsilon}_t$ is inversely proportional to this similarity measure, and $\boldsymbol{\varepsilon}_t$ is computed by

$$\varepsilon_t = \max\left(\min(\lambda_0 \sqrt{1/\varphi_t}, \lambda_{\max}), \lambda_{\min}\right) \tag{15}$$

where $\lambda_{\min}$ is the lower bound to maintain a reasonable sample coverage and $\lambda_{\max}$ is the upper bound to constrain the variance below a certain threshold if φ_t is very small.

The similarity in the real data experiments is computed by (12a) and (12b) as a Gaussian likelihood between the histogram of the previous state (i.e., target model) and the histogram of the current mean state. The similarity in the synthetic data experiments is computed by (16) as the Mahanalobis distance from the estimated current mean location to the current observed location. Finally, we plug φ_t into (15) to obtain the new variance

$$\varphi_t = \exp\left((-0.5)\left(\mu_{L_t} - Z_t\right)^T \Sigma^{-1} \left(\mu_{L_t} - Z_t\right)\right). \tag{16}$$

In (16), μ_{Lt} is the mean location estimated by RBPF, Z_t the observation at time t, and Σ the estimation error covariance of $\{x_t, y_t\}$.

Note that not only the noise variance of the Monte Carlo part can be made adaptive, the noise variance of the Kalman filter (e.g., the system noise covariance P and the measurement noise covariance Q) can also be made adaptive, which should capture the similarity between the size of a sample ellipse at the current time step and the size of a sample ellipse at the previous time step.

Input::$S_{t-1} = \{< R^i_{t-1}, \mu^i_{t-1}, \sigma^i_{t-1} > | \, i = 1, ..., N\}$ and Z_t.
for $i = 1 : N$ do
1. Propagate samples.
a) Sample object motion:
$R^{i-}_t \sim p(R_t \mid R^i_{t-1}, Z_t) = p(Z_t \mid R_t) p(R_t \mid R^i_{t-1})$
b) Kalman prediction:
$L^{i-}_t \sim p(L_t \mid R^i_t, R^i_{t-1}, L^i_{t-1}, Z_t)$
2. Evaluate weight
a) Compute the color histogram: (10a)-(10b)
b) Compute the gradient: (11a)-(11c)
c) Compute the weight: (12a)-(12d)
end for loop
3. Select samples.
for $i = 1 : N$ do
4. Kalman update: (13a)-(13c)
end for loop
5. Compute the mean state: (14)
6. Compute new system noise: (15)-(16)

Fig. 3. RBPF algorithm for tracking in surveillance.

III. Evaluation of the Proposed RBPF Algorithm

In this section, we evaluate the performance of proposed adaptive RBPF algorithm by comparative study between regular particle filter [3] and the proposed adaptive RBPF algorithm.

A. Simulation Settings

In the simulation, a true path and a noisy path are generated: a point moving on a 2-D plane with nonconstant velocity generates an actual path; noise is then added to the actual path to simulate a noisy measurement of the actual path. Then RBPF and PF are invoked to "track" the true path using the noisy path as the measurement. The state of the moving point at any time is given by $s_t = \{x_t, u_t, y_t, v_t\}$, in which $\{x_t, y_t\}$ corresponds to the horizontal and vertical position and $\{u_t, v_t\}$ the horizontal and vertical velocity. When generating the true path, we assume $v_t = 6u_t$. This is to facilitate utilizing dependency relation between the root and the leaf variables to apply RBPF. The state is projected forward by a nonlinear non-Gaussian model (17), where $rand$ denotes uniform random noise

$$\begin{aligned} x_t &= (u_{t-1})^{1.5} + \text{rand} \\ u_t &= u_{t-1} + \text{rand} \\ y_t &= \sqrt{v_{t-1}} + \text{rand} \\ v_t &= 6u_t. \end{aligned} \tag{17}$$

The noisy measurement Z_t is created by (18), where ς_t is a vector of uniform random noise distributed within $[-1\ 1]$

$$Z_t = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} s_t + \varsigma_t. \tag{18}$$

After the true and noisy paths are generated, both the proposed RBPF and a regular PF are invoked ten times to track the true path with random initializations using the noisy path as the measurement. 200 particles are used in the two algorithms. With the actual path as the ground truth, the performance of these two algorithms can be quantitatively compared. In the RBPF estimation, the root, the leaf and the observations are: $R_t = \{u_t\}$, $L_t = \{x_t, y_t, v_t\}$ (in order to maintain the linear substructure of the leaf variables, u_t is also incorporated in the computation of the mean of the leaf variables as a dummy element), $Z_t = \{x_t, y_t\}$. The root is estimated by regular particle filtering, and the leaf variables are estimated using Kalman filter by assuming the dependency $v_t = 5u_t$, which is different from the true dependency $(v_t = 6u_t)$ used in the path generation. This is intended to test if the proposed algorithm still works fine even if the exact dependency model between the state variables is not available.

B. Increased Estimation Accuracy

Fig. 4 shows the true path and the average path estimated by RBPF and PF over ten Monte Carlo simulations. In Fig. 5(a), the location root mean-square error (RMSE) obtained by PF and RBPF are compared. The location RMSE at time t is computed by (19), where R_{MC} is the number of MC simulations, l^{true}_t denotes the true location states at time t, and $\hat{l}^j_t$ denotes the estimated location states in the $j-th$ simulation at time t. Fig. 5(a) reveals that the proposed RBPF outperforms the regular PF in reducing estimation errors by a large margin

$$l^{\text{RMSE}}_t = \sqrt{(1/R_{MC}) \sum_{j=1}^{R_{MC}} \left\| l^{\text{true}}_t - \hat{l}^j_t \right\|_2^2}. \tag{19}$$

C. Reduced Estimates Variance

One advantage of Rao–Blackwellization is that it can reduce the variance of the state estimates. To verify this, the standard deviation (STD) of the state over 10 MC simulations at each time instant is computed [(20) gives the way of computing STD for y_t]. Fig. 5(b) shows the STD comparison for the vertical location y_t. We found that the variance of the RBPF estimates is much lower than that of the PF for any one of the four state components

$$y^{\text{STD}}_t = \sqrt{(1/(R_{MC}-1)) \sum_{j=1}^{R_{MC}} \left(\hat{y}^j_t - \overline{y}_t \right)^2} \tag{20a}$$

$$\overline{y}_t = \frac{1}{R_{MC}} \sum_{j=1}^{R_{MC}} \hat{y}^j_t. \tag{20b}$$

D. Reduced Particle Numbers

Another advantage of Rao–Blackwellization is that it can reduce the volume of the space over which we need to sample. This in turn brings the benefit that far fewer particles are needed to reach the same level of accuracy, compared with a conventional PF. Table I illustrates such results from the proposed algorithm. In Table I, as the number of particles increases in PF,

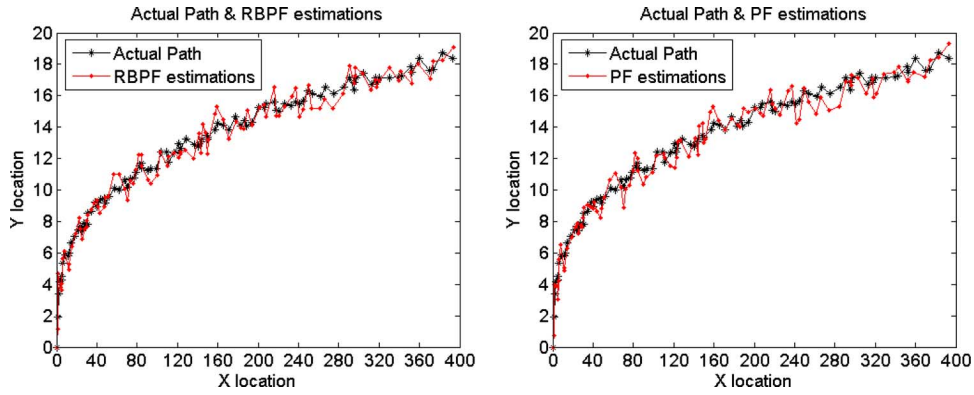


Fig. 4. Left: True path versus path estimated by the proposed RBPF. Right: True path versus path estimated by a particle filter.

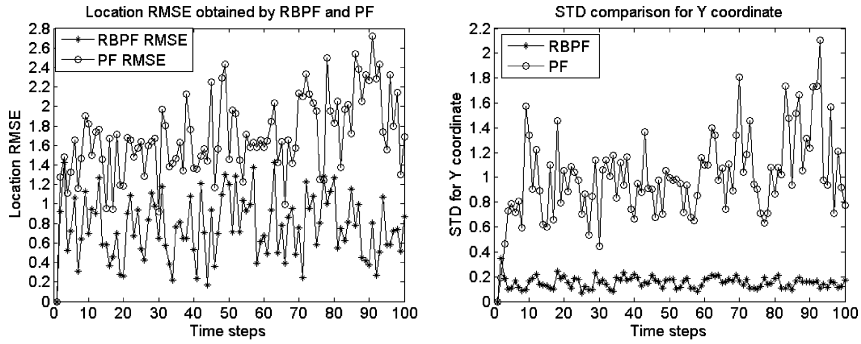


Fig. 5. (a) Comparison of location RMSE obtained by RBPF and PF. (b) Comparison of standard deviation for the y coordinate.

TABLE I
LOCATION RMSE FOR RBPF/PF RUNNING WITH VARIOUS NUMBERS OF PARTICLES

Algorithm	Location RMSE
PF 200 particles	1.5568
PF 300 particles	1.3923
PF 400 particles	1.2605
PF 500 particles	1.1548
RBPF 200 particles	0.81262

the location RMSEs given by (21) are decreasing. However, the location RMSE obtained by PF cannot be decreased any more by simply increasing the number of particles when the number reaches 500. At that time the location RMSE obtained by PF are still larger than that obtained by RBPF with 200 particles. Thus, we may conclude that for this simulation setting, the proposed RBPF only needs 200 particles to reach the same of level of estimation accuracy obtained by PF running with 500 particles

$$l^{\mathrm{RMSE}} = \sqrt{(1/T)\sum_{t=1}^{T}(1/R_{MC})\sum_{j=1}^{R_{MC}}\left\|l_t^{\mathrm{true}} - \hat{l}_t^j\right\|_2^2}. \quad (21)$$

E. Reduced Weight Variances

It has been revealed [23] that the variance of true weights (the weights before resampling) can be decreased after applying Rao–Blackwellization. To verify this in our algorithm, we compute the standard deviations of weights of 200 particles in the RBPF and PF at each time step. Fig. 6(a) clearly shows that the standard deviation obtained by RBPF is evidently less than that of PF. The significance for the weight variance reduction is discussed in depth in Section IV.

F. Dependence Analysis

As we mentioned in Section II-C, the dependency relationship between the leaf and the root affects the performance of the adaptive RBPF algorithm. In practice, it is usually difficult to obtain an exact dependency model unless the surface normal of the dominant plane and the image plane are known. Thus, it is necessary to analyze the impact of an inaccurate dependency model on the performance of the proposed algorithm. In the evaluation for this purpose, the true path is the one in Fig. 4, with the dependency being $v_t = 6u_t$. Then, 15 different dependencies are used to estimate the actual path, with $v_t = Su_t$, $S = 1, 2, \ldots, 15$. Fig. 6(b) shows the estimated location RMSE computed by (21) under various dependencies. We can tell that the location errors fluctuate within a small range when the dependency scalars fall into a relatively large range around the true dependency scalar 6. The estimation error has a peak when the tentative dependency is far away (e.g., $S = 15$) from the true dependency. This result validates our conjecture that the proposed RBPF is not very sensitive to the choice of dependency since within a relatively large margin of the true dependency, the estimation error of RBPF does not increase too much.

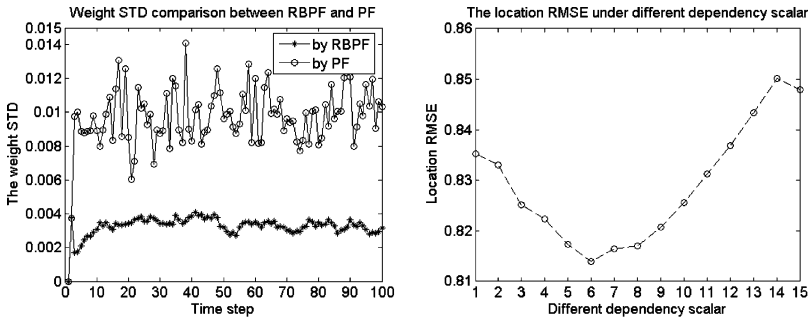


Fig. 6. (a) Weight STD comparison between RBPF and PF. (b) Location RMSE estimated by RBPF under different dependency models.

Fig. 7. Outdoor human tracking. Every ninth frame is shown to cover the entire sequence. Top row: RBPF tracker. Bottom row: PF tracker. The green dots are the gradient contours around the mean state. PF tracker lags behind the true location, whereas RBPF tracker can maintain tracking all the way.

IV. Real Data Experiment

Extensive real data experiments have been carried out to evaluate the performance of proposed RBPF algorithm and compare it with a regular particle filter.

One surveillance scenario is outdoor human tracking. The test sequence for this scenario is from the CAVIAR project (http://www.homepages.inf.ed.ac.uk/rbf/CAVIAR/). The ground truth data were also provided accompanying the video. Fig. 7 illustrates the RBPF and PF tracking results on regularly spaced frames extracted from a 35-frames-long sequence where a person moves toward the window, stops there for browsing, and continues moving along the corridor. We may notice that the PF tracker lags behind the true location when the person transits from *standing* to *moving* and it takes the PF tracker several frames to catch the person. In contrast, the RBPF tracker maintains good tracking during this process which involves large motion velocity and orientation transition. We also found that, when the person gets further from the camera and consequently, the size becomes very small, the PF tracker would not realistically capture the scale change of the object and tend to deviate from the object; the RBPF algorithm, in contrast, still maintains good tracking since it can analytically update the scale change using the location information.

In addition, the reduction of the weight variance is confirmed using the sequence in Fig. 7. This, in turn, brings the great benefit of degeneracy alleviation. A suitable measure of degeneracy, the effective sample size N_{eff} [23], [25], well explains why the reduction of weights variance alleviates degeneracy. It is defined by $N_{\text{eff}} = N/(1 + \text{Var}(w_t^{*i}))$ [3] where w_t^{*i} refers to the weights before resampling. Notice that the effective sample size N_{eff} is less or equal to the particle number N, and if the variance of weights is large, N_{eff} would be small, implying severe degeneracy. The researches by Kong *et al.* [25] and Doucet [28] revealed that the variance of weights can only increases (statistically) over time and, thus, eventually leads to very small effective sample size. A straightforward yet brute force approach to reducing this effect is to use a very large N [3], but that is often impractical in the case of high-dimensional state space. Some other good approaches to avoiding degeneracy include choosing good importance density and using resampling [2], [28]. In the subsequent parts, we show that applying the Rao–Blackwell technique can effectively reduce the weights variance and in turn reduce the effect of degeneracy in the case of high-dimensional state-space estimation.

To verify that the weight variance is decreased by RBPF, both the RBPF and the regular PF with 200 particles are invoked ten times with different random initializations to track the person in Fig. 7. We compute the variance of the 200 normalized weights at each time step. Finally, the weight variances at each instance obtained by the ten runs are averaged, as shown in Fig. 8(a). During the tracking process, the RBPF weight variances are ob-

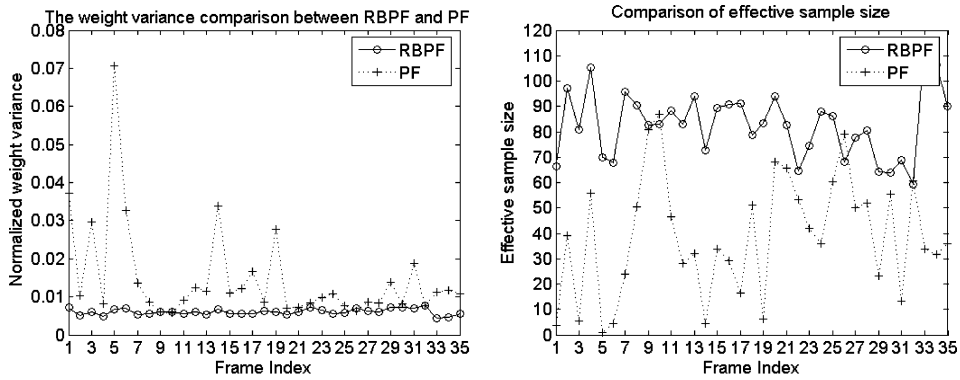


Fig. 8. (a) Weight variance obtained by RBPF and PF using real data. (b) Effective sample size obtained by RBPF and PF.

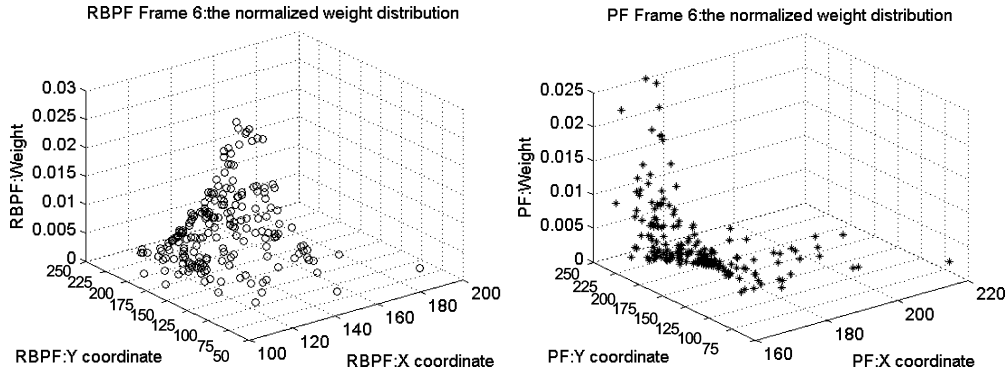


Fig. 9. Particle weights distribution for the third RBPF (left) tracking frame and (right) third PF tracking frame in Fig. 7. A large number of particles in PF receive very small weights and, hence, will not have offspring in the resampling stage, while the RBPF weights distribution presents reasonably large effective sample size.

viously lower than those from the PF. Based on the weight variance obtained in Fig. 8(a), the effective sample size for each frame is calculated by (22) which is an estimate of N_{eff} introduced in [2]. Fig. 8(b) shows that PF results in more severe degeneracy due to the relatively smaller effective sample size

$$\hat{N}_{\text{eff}} = 1 \Big/ \sum_{i=1}^{N} \left(w_k^i\right)^2. \tag{22}$$

Now we want to show why large effective sample size will help alleviate degeneracy. To this end, we are interested at knowing the weight of a given state sample. Fig. 9 illustrates the distributions of particle weights for the third frame in Fig. 7. The x axis and y axis denote the person's horizontal and vertical coordinate, respectively, and the z axis represents the normalized weight. We can see that, in the right plot from PF, all but a few particles have negligible weights and, hence, will not be selected in the resampling procedure. Thus, these particles are actually wasted and those few particles with larger weights are reselected many times. This process will lead to inadequate representation of the posterior density within a few iterations. Comparing to PF, the proposed RBPF produces much larger effective sample size (left plot) since the shape of the weight distribution has a relatively good spread instead of a narrow peak like in PF.

A more challenging human tracking sequence is shown in Fig. 10. The task is made difficult by the reflections from the ground, the opposing window, occlusion due to the text on the window, etc. Nevertheless, the RBPF algorithm obtained good tracking, while the PF tracker was distracted by the similar color of the ground when it encountered the red text.

Another test scenario in surveillance is vehicle tracking. One sample result is shown in Fig. 11, in which the car is undertaking complex motions like turning (rotation), translation and large scale change. Particularly, we observe that the PF does not reflect the scale change realistically when the object becomes very small due to the increasing distance from the camera, whereas the proposed method is able to maintain good tracking in face of the dramatic scale change.

V. Related Work

A few techniques have been proposed to make particle filtering applicable in high-dimensional state space. In [26], Deutscher *et al.* developed an annealed particle filtering for search in the high-dimensional space in the context of articulated 3-D human motion tracking. Essentially, a series of weight distribution functions were arranged into layers, from a relatively flat distribution to distributions with local and global peaks. Initially they sampled a broad distribution with reduced sensitivity and gradually they introduced the narrow peak so that the sampling migrated towards the global peaks. Although annealing works well in some applications, e.g., [26] and [27], it has limitations as a general method for reducing dimensionality. The main problem is that it samples indiscriminately within a certain energy band, regardless of whether the points sampled are likely to lead out of the basin towards another

Fig. 10. Person moving in front of a shop window, shown for every 15th frame extracted from the whole sequence. Top row: RBPF tracker. Bottom row: PF tracker. Frame index is 65, 80, 95, and 118. PF tracker was distracted by the ground when it encountered the red text whose color differs from the initial grey color of the target object.

Fig. 11. Vehicle tracking. Top row: RBPF tracker; bottom row: PF tracker. When the size of the object becomes small as the vehicle moves far away from the camera, the PF tracker cannot scale accurately and tends to be distracted by the background texture. On the other hand, RBPF tracker maintains good tracking, capturing the scale change accurately.

local maximum, or whether they simply lead further up an ever-increasing potential peak [29].

Other related techniques, instead of working directly on reducing the size of the state-space, focus on adopting efficient sampling scheme. For example, a well-known improved variant of regular PF is ICondensation introduced in [19]. It is extended from the original Condensation algorithm by including high level information as an additional source of knowledge to sample the posterior. This method was used to track deformable contours in highly cluttered environments.

Another variant of PF is the auxiliary particle filter (APF), proposed by Pitt and Shephard [20]. It aims at enhancing the effectiveness of importance sampling. The advantage of the APF is that it produces samples which, conditional on the current observations, are most likely to be consistent with the true state [2]. If the process noise is small, then the APF is often not so sensitive to the outliers as sequential importance resampling (SIR) and the weight variance is decreased; but if the process noise is large, the use of the APF then degrades performance due to the poor approximation of $p(X_t|X_{t-1})$ [2].

The proposed RBPF algorithm differs from the current techniques in that it makes use of the "structural" information inherent in the problem itself to analytically infer the state of the linear dynamics, thus leaving only the nonlinear parts to be sampled. This directly leads to increased estimation accuracy.

VI. Discussion

A. Failure Cases

In our work, the conditional relationship between various components of the state space is abstracted into a constraint between the location and the scale components which is warranted by the typical camera configuration in typical surveillance applications. Our basic assumption is that if the camera is mounted relatively higher than the target object, the scale of the object will become bigger as the y coordinate of the object gets larger. If the real scenario does not satisfy this assumption, although the proposed RBPF may still work, the desired advantages of RBPF might not be achieved and the performance might be even poorer than a PF. On the other hand, if the aforementioned basic assumption is satisfied, we found that the proposed RBPF performs better than the regular PF especially when the object moves with fast changing velocity between two successive frames, or the object scale is changing relatively fast.

B. Computation Cost

Our real data experiments run at a 3.00-GHZ Pentium 4 CPU with 200 samples. For typical image with resolution $320 * 240$, the processing frame rate is roughly 5 Hz. More work is needed to improve the speed performance for real time processing, e.g., through optimizing the color histogram computation. In our

problem, if equivalent number of particles is used for RBPF and PF, the RBPF algorithm is computationally more expensive than PF (but the estimation accuracy of RBPF is higher than PF), because our original target model has only eight dimensions, and the dimensionality reduction from 8 to 4 is not enough to offset additional computation required by RBPF over PF. On the other hand, with the increasing dimensionality of the target model, the computation cost of PF is increasing exponentially; thus, the application of Rao–Blackwell technique in this case is supposed to greatly improve the estimation efficiency. Our conclusion is that, to achieve the same level of estimation accuracy, RBPF needs far fewer particles than PF does; hence, it is more efficient than PF.

C. Future Improvements to the Proposed Algorithm

When applying the proposed RBPF algorithm, an important issue is to determine the dependency relationship between the root and leaf variables. In general, this dependency could be a conditional probability function or a deterministic scalar. In our case, we simplify the problem by using a deterministic scalar, and set it as a fixed value during the entire tracking process for a particular scene, and this scalar has to be tuned for different scenes. A better strategy to determine this dependency is to dynamically detect the surface normal with respect to the camera through vanishing lines. In addition, we use an ellipse to model the object shape (although the gradient-related measurements go beyond the basic ellipse), which is not always a good model. Deformable contours used in [5] may be explored in our future work.

VII. Conclusion

In this paper, we proposed an adaptive RBPF for surveillance tracking. We discussed how the dependency between state variables imposed by typical surveillance application could be utilized to improve the accuracy of a regular particle filter. Essentially, this is accomplished by partitioning the state variables into separate groups, with the linear parts being computed by Kalman filter and nonlinear part being estimated by particle filter. Extensive comparative studies using both simulated and real data have demonstrated the improved performance of the proposed RBPF over regular particle filtering.

Rao–Blackwellization can be applied to a broader context than surveillance tracking. Two key problems arise when applying RBPF to a real problem: the first is how to partition the state space into two (or more) meaningful groups; the second is what analytical filter should be used to efficiently estimate the leaf variables conditional on the root variables. Often, these two issues need to be solved based on the nature of the specific application. As a possible future working direction, we are investigating the potential of using learning approaches to find a proper dependency model from a large number of state variables.

References

[1] A. Doucet, J. Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*. New York: Springer-Verlag, 2001, ch. 1.

[2] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for on-line non-linear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 174–188, Feb. 2002.

[3] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *Proc. Inst. Elect. Eng. F*, vol. 140, no. 2, pp. 107–113, 1993.

[4] B. Li and R. Chellappa, "A generic approach to simultaneous tracking and verification," *IEEE Trans. Image Process.*, vol. 11, no. 5, pp. 530–544, May 2002.

[5] M. Isard and A. Blake, "Contour tracking by stochastic propagation of conditional density," in *Proc. Eur. Conf. Computer Vision*, 1996, pp. 343–356.

[6] S. Arulampalam and B. Ristic, "Comparison of the particle filter with Range parameterized and modified polar ekf for angle-only tracking," *Signal Data Process. Small Targets*, vol. 4048, pp. 288–299, 2000.

[7] A. Doucet, N. de Freitas, and N. Gordon, "An introduction to sequential Monte Carlo methods," in *Sequential Monte Carlo Methods in Practice*, A. Doucet, N. de Freitas, and N. Gordon, Eds. New York: SpringerVerlag, 2001, pp. 3–13.

[8] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statist. Comput.*, vol. 10, no. 3, pp. 197–208, 2000.

[9] T. Schon, F. Gustafsson, and P.-J. Nordlund, "Marginalized particle filters for mixed linear/nonlinear state-space models," *IEEE Trans. Signal Process.*, vol. 53, no. 7, pp. 2279–2289, Jul. 2005.

[10] K. Murphy and S. Russell, "Rao–Blackwellised particle filtering for dynamic Bayesian networks," in *Sequential Monte Carlo Methods in Practice*, A. Doucet, N. Freitas, and N. Gordon, Eds. New York: Springer-Verlag, 2001, ch. 24.

[11] G. Casella and C. P. Robert, "Rao Blackwellisation of sampling schemes," *Biometrika*, vol. 83, no. 1, pp. 81–94, 1996.

[12] Z. Khan, T. Balch, and F. Dellaert, "A Rao–Blackwellized particle filter for Eigen tracking," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, Washington, DC, 2004, vol. 2, pp. 980–986.

[13] D. Schulz, D. Fox, and J. Hightower, "People tracking with anonymous and id-sensors using Rao–Blackwellised particle filters," presented at the Int. Joint Conf. Artificial Intelligence, Acapulco, Mexico, 2003.

[14] N. de Freitas, R. Dearden, F. Hutter, R. Morales-Menendez, J. Mutch, and D. Poole, "Diagnosis by a waiter and a Mars explorer," *Proc. IEEE*, vol. 92, no. 3, pp. 455–468, Mar. 2003.

[15] C. Kwok and D. Fox, "Map-based multiple model tracking of a moving object," in *Proc. RoboCup*, 2004, pp. 18–33.

[16] P.-J. Nordlund and F. Gustafsson, "Sequential Monte Carlo filtering techniques applied to integrated navigation systems," presented at the Amer. Control Conf., Arlington, VA, Jun. 2001.

[17] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, Mar. 2004, p. 220.

[18] K. Nummiaro, E. Koller-Meier, and L. V. Gool, "An adaptive color-based particle filter," *Image Vis. Comput.*, vol. 21, pp. 99–110, 2002.

[19] M. Isard and A. Blake, "Icondensation: Unifying low-level and high-level tracking in a stochastic framework," in *Proc. 5th Eur. Conf. Computer Vision*, 1998, vol. 1, pp. 893–908.

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

[21] A. Doucet, "On sequential monte carlo methods for bayesian filtering," Tech. Rep., Dept. Eng., Univ. Cambridge, Cambridge, U.K., 1998.

[22] G. Kitagawa, "Monte Carlo filter and smoother for non-Gaussian nonlinear state space models," *J. Comput. Graph. Statist.*, vol. 5, pp. 1–25, 1996.

[23] J. S. Liu and R. Chen, "Sequential Monte Carlo methods for dynamic systems," *J. Amer. Statist. Assoc.*, vol. 93, pp. 1032–1044, 1998.

[24] A. Doucet, N. J. Gordon, and V. Krishnamurthy, "Particle filters for state estimation of jump Markov linear systems," *IEEE Trans. Signal Process.*, vol. 49, no. 3, pp. 613–624, Mar. 2001.

[25] A. Kong, J. S. Liu, and W. H. Wong, "Sequential imputations and Bayesian missing data problems," *J. Amer. Stat. Assoc.*, vol. 89, no. 425, pp. 278–288, 1994.

[26] J. Deutscher, A. Blake, and I. Reid, "Articulated body motion capture by annealed particle filtering," in *Proc. IEEE Conf. Computer Vision Pattern Recognition*, 2000, vol. 2, pp. 126–133.

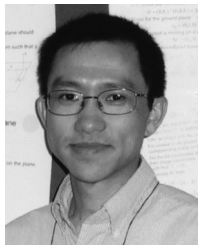
[27] R. M. Neal, "Annealed importance sampling," *Statist. Comput.*, vol. 11, pp. 125–139, 2001.

[28] A. Doucet, N. D. Freitas, K. P. Murphy, and S. J. Russell, "Rao–Blackwellised particle filtering for dynamic Bayesian networks," in *Proc. 16th Conf. Uncertainty in Artificial Intelligence*, 2000, pp. 176–183.

[29] Soto, "A probabilistic approach for the adaptive integration of multiple visual cues using an agent network," Ph.D. dissertation, Robotics Institute, Carnegie Mellon Univ., Pittsburgh, PA, Sep. 2002.

Xinyu Xu received the B.S. degree from Qingdao University in 2001 and the M.S. degree from the Beijing University of Posts and Telecommunications in 2004, both in computer science and engineering. She is currently pursuing the Ph.D. degree at the Center for Cognitive Ubiquitous Computing and the Department of Computer Science and Engineering, Arizona State University, Tempe.

She was with the Alcatel Shanghai Bell, Co., Ltd., as a Mobile Service Solution Specialist from April 2004 to July 2004. Her research interests include computer vision, cognitive visual analysis, machine learning, medical image analysis, and pattern recognition.



Baoxin Li (S'97–M'00–SM'04) received the Ph.D. degree in electrical engineering from the University of Maryland, College Park, in 2000.

He is currently an Assistant Professor of computer science and engineering with Arizona State University, Tempe. He was previously a Senior Researcher with SHARP Laboratories of America (SLA), Camas, WA, where he was the Technical Lead in developing SHARP's Hi-Impact Sports Technologies. He was also an adjunct faculty member with the Portland State University, Portland, OR, from 2003 to 2004. His research interests include pattern recognition, computer vision, multimedia processing, and statistical methods in computing.