IMPROVED PARTICLE FILTERING SCHEMES FOR TARGET TRACKING

Zhe Chen [1], *Thia Kirubarajan* [1], *Mark R. Morelande* [2]

1. Communications Research Lab, McMaster University, Canada
2. Center for Sensor, Signal and Information Processing, University of Melbourne, Australia

ABSTRACT

In this paper, we propose two improved particle filtering schemes for target tracking, one based on gradient proposal and the other based on Turbo principle. We present the basic ideas and derivations and show detailed results of three tracking applications. Favorable experimental findings have shown the efficiency of our proposed schemes and their potential in other tracking scenarios.

1. INTRODUCTION

Recent years have witnessed ever-growing efforts in applying particle filters to signal processing, communications, and machine learning [1]. Bearing the nature of recursive Bayesian estimation and sequential Monte Carlo sampling, particle filtering has demonstrated its potential in various nonlinear, non-Gaussian, non-stationary sequential estimation problems. Among many, target tracking problem provides a testbed for particle filter. As well known, the conventional Bayesian bootstrap [3] or SIR filtering (using prior proposal) has a drawback of ignoring the most recent observation. In this paper, we propose two improved particle filtering schemes to overcome this weakness and apply them in several tracking applications. The first improvement scheme is to use gradient information of the measurement model. The idea of gradient proposal is very heuristic, but it is very simple to implement and turns out to be quite efficient in practice [4]. We also propose another new particle filtering method based on Turbo principle (motivated from Turbo decoding in communications). Basically, we use one filter (so-called *slave filter*) to produce a first-stage (rough) estimate; and we run another filter (*master filter*) in parallel to yield the second-stage (ultimate) estimate, which uses its current as well as previous estimate for importance weights update in a recursive particle filtering fashion.

The rest of the paper is organized as follows: In section 2, we briefly discuss the Bayesian bootstrap filter and then introduce the improved schemes for particle filtering. Section 3 is devoted to two simulated and one real-life tracking applications, followed by concluding remarks in Section 4.

2. PARTICLE FILTERING AND IMPROVED SCHEMES

2.1. State-Space Model and Bayesian Bootstrap Filter

Consider a generic discrete-time nonlinear state space model:

$$\mathbf{x}_{n+1} = \mathbf{f}(n, \mathbf{x}_n, \mathbf{d}_n), \quad (1a)$$

$$\mathbf{y}_n = \mathbf{g}(n, \mathbf{x}_n, \mathbf{v}_n), \quad (1b)$$

where $\mathbf{d}_n$ and $\mathbf{v}_n$ characterize the dynamic and measurement noise processes, respectively. The state equation (1a) characterizes the state transition probability $p(\mathbf{x}_{n+1}|\mathbf{x}_n)$, whereas the measurement equation (1b) describes the probability $p(\mathbf{y}_n|\mathbf{x}_n)$ which is further related to the measurement noise model.

Simply say, particle filter uses a number of *independent* random variables called particles, sampled directly from the state space, to represent the posterior probability, and update the posterior by involving the new observations; the "particle system" is properly located, weighted, and propagated recursively according to the Bayesian rule. Specifically, using a sequential important sampling (SIS) scheme, it can be shown [2] that the importance weights update has the following recursive form:

$$W_n^{(i)} = W_{n-1}^{(i)} \frac{p(\mathbf{y}_n|\mathbf{x}_n^{(i)}) p(\mathbf{x}_n^{(i)}|\mathbf{x}_{n-1}^{(i)})}{q(\mathbf{x}_n^{(i)}|\mathbf{x}_{0:n-1}^{(i)}, \mathbf{y}_{0:n})}. \quad (2)$$

where $W_n^{(i)} = p(\mathbf{x}_n^{(i)})/q(\mathbf{x}_n^{(i)})$ denotes the importance weight, and $q(\mathbf{x}_n^{(i)}|\mathbf{x}_{0:n-1}^{(i)}, \mathbf{y}_{0:n})$ represents the proposal distribution. Choosing a proper proposal often has a crucial effect on the particle filtering performance.

The well-known SIR and Bayesian bootstrap filter [3] use a transition prior as proposal, i.e. $q(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathbf{y}_{0:n}) = p(\mathbf{x}_n|\mathbf{x}_{n-1})$; it then simplifies (2) to

$$W_n^{(i)} = W_{n-1}^{(i)} p(\mathbf{y}_n|\mathbf{x}_n^{(i)}), \quad (3)$$

which essentially neglects the effect of recent observation $\mathbf{y}_n$. Despite its appealing simplicity, this proposal distribution is far from optimal and its resulted performance can be quite poor even a large number of particles are used. Many improved schemes (such as the auxiliary variable) have been developed in the literature [1]. In the following, we will describe two improved schemes in an attempt to efficiently incorporate the observation information into the sampling step.

2.2. Particle Filtering Using Gradient Proposal

In order to use the recent observation, we propose to use the gradient information of (1b) to select the "informative" particles [4]. The main idea behind it is to introduce a MOVE-step to sampling for the proposal distribution, which is plugged in before the sampling step in the conventional SIR filter. This new algorithm essentially calculates the gradient information from the likelihood model and guides the particles towards the low-error region, along the gradient-descent direction; by assuming an additive measurement noise model in (1b), the MOVE-step is described by

$$\hat{\mathbf{x}}_{n|n-1} = \hat{\mathbf{x}}_{n-1|n-1} - \eta \frac{\partial (\mathbf{y}_n - \mathbf{g}(\mathbf{x}))^2}{\partial \mathbf{x}}\bigg|_{\mathbf{x}=\hat{\mathbf{x}}_{n-1|n-1}} \quad (4)$$

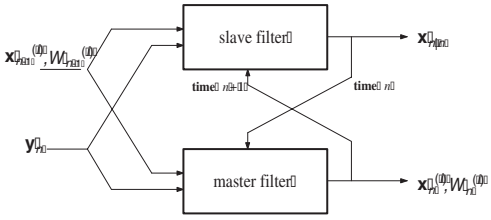

Fig. 1. A schematic diagram of Turbo particle filtering.

where $\eta \in [0.001, 0.01]$ is a small-valued step-size parameter. As expected from (4), the inclusion of current observation $\mathbf{y}_n$ and the calculation of gradient information will tend to push the samples to a high-likelihood region, thereby providing more reliable predictive samples for the next step. In summary, the improved particle filtering with gradient proposal reads as follows:

1. For $i = 1, \cdots, N_p$, sample $\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0)$, $W_0^{(i)} = 1/N_p$.
2. For each sample $\{\mathbf{x}_{n-1}^{(i)}\}$, update the sample via (4).
3. Importance sampling: $\hat{\mathbf{x}}_n^{(i)} \sim p(\mathbf{x}_n|\hat{\mathbf{x}}_{n|n-1}^{(i)})$.
4. Importance weights update:

$$W_n^{(i)} = W_{n-1}^{(i)} p(\mathbf{y}_n|\hat{\mathbf{x}}_n^{(i)}) \frac{p(\hat{\mathbf{x}}_n^{(i)}|\hat{\mathbf{x}}_{n-1|n-1}^{(i)})}{p(\hat{\mathbf{x}}_n^{(i)}|\hat{\mathbf{x}}_{n|n-1}^{(i)})}.$$

5. Calculate effective sample size $\hat{N}_{eff}$, if $\hat{N}_{eff} > N_p/2$, resampling; otherwise go to Step 2.

2.3. Turbo Particle Filtering

A schematic diagram of Turbo particle filtering is illustrated in Fig. 1. In Fig. 1, there are two filters in parallel, to be run iteratively. The slave filter, being an extended Kalman filter (EKF) here, is used to produce a rough estimate $\hat{\mathbf{x}}_{n|n}$, given the current observation $\mathbf{y}_n$ and previous state estimates $\mathbf{x}_{n-1}^{(i)}$. This is done by the typical EKF equations. Note that in the *prediction step*, every particle $\mathbf{x}_{n-1}^{(i)}$ is passed through the state equation, where the predicted covariance can be estimated by the sample covariance, $\hat{\mathbf{P}}_{n|n-1}$, or calculated through linearization; in the *filtering step*, instead of using all of the samples $\{\hat{\mathbf{x}}_{n|n-1}^{(i)}\}$, we only use its mean value, $\hat{\mathbf{x}}_{n|n-1} = \langle \hat{\mathbf{x}}_{n|n-1}^{(i)} \rangle$, to perform the EKF update:

$$\hat{\mathbf{x}}_{n|n} = \hat{\mathbf{x}}_{n|n-1} + \mathbf{K}_n(\mathbf{y}_n - \mathbf{g}(\hat{\mathbf{x}}_{n|n-1})), \quad (5)$$

$$\mathbf{P}_{n|n} = \hat{\mathbf{P}}_{n|n-1} + \mathbf{K}_n \mathbf{C}_n \hat{\mathbf{P}}_{n|n-1}, \quad (6)$$

where $\mathbf{K}_n = \hat{\mathbf{P}}_{n|n-1}\mathbf{C}_n^T(\mathbf{C}_n\hat{\mathbf{P}}_{n|n-1}\mathbf{C}_n^T + \Sigma_{\mathbf{v}})^{-1}$, and $\mathbf{C}_n$ is the linearized Jacobian matrix of the measurement equation, $\mathbf{P}_{n|n}$ is the filtered state covariance. Note that the filtered estimate $\hat{\mathbf{x}}_{n|n}$ is more accurate than the predicted estimate $\hat{\mathbf{x}}_{n|n-1}$, since it utilizes the observation $\mathbf{y}_n$. In the meantime, the master filter, given $\mathbf{y}_n$ and the previous simulated samples $\{\mathbf{x}_{n-1}^{(j)}\}$, as well as the first-stage estimate $\hat{\mathbf{x}}_{n|n}$, runs a particle filtering procedure with a constructed suboptimal proposal distribution, and further produces a second-stage posterior estimate $\mathbf{x}_n^{(i)}$. After a complete step, the master filter propagates its samples to the slave filter for the next iteration. Essentially, two filters are trying to solve the same filtering problem but looking at it from different perspectives; each one takes advantage of the result of the other at the previous step and thereby produces the solution in a cooperative way. Due to its similarity to Turbo decoding, we call the proposed filter structure as *Turbo particle filter* (TPF). In what follows, we will derive the update equation mathematically in detail.

Let us write the filtering posterior in a slightly different way:

$$\begin{aligned} p(\mathbf{x}_n|\mathbf{y}_{0:n}) &= p(\mathbf{x}_n|\mathbf{y}_n, \mathbf{y}_{0:n-1}) \\ &= \frac{p(\mathbf{y}_n|\mathbf{x}_n, \mathbf{y}_{0:n-1})p(\mathbf{x}_n|\mathbf{y}_{0:n-1})}{p(\mathbf{y}_n|\mathbf{y}_{0:n-1})} \\ &\propto p(\mathbf{y}_n|\mathbf{x}_n)p(\mathbf{x}_n|\mathbf{y}_{0:n-1}). \end{aligned} \quad (7)$$

Next, suppose we can draw samples $\{\mathbf{x}_n^{(i)}\}$ from a proposal distribution, we need to find the importance ratios to appropriately weight the samples. Using the importance sampling trick, we have

$$W_n^{(i)} = \frac{p(\mathbf{y}_n|\mathbf{x}_n^{(i)})p(\mathbf{x}_n^{(i)}|\mathbf{y}_{0:n-1})}{q(\mathbf{x}_n^{(i)}|\mathbf{y}_n)}, \quad (8)$$

where $q(\mathbf{x}_n^{(i)}|\mathbf{y}_n)$ is the proposal distribution. Assuming that at time n, we have the simulated particles for approximating the posterior of time $n-1$:

$$p(\mathbf{x}_{n-1}|\mathbf{y}_{0:n-1}) \approx \sum_{j=1}^{N_p} \tilde{W}_{n-1}^{(j)} \delta(\mathbf{x}_{n-1} - \mathbf{x}_{n-1}^{(j)}), \quad (9)$$

where $\tilde{W}_{n-1}^{(j)}$ are the normalized importance weights with the sum equal to unity. Hence, we can have

$$\begin{aligned} p(\mathbf{x}_n^{(i)}|\mathbf{y}_{0:n-1}) &= \int p(\mathbf{x}_n^{(i)}|\mathbf{x}_{n-1})p(\mathbf{x}_{n-1}|\mathbf{y}_{0:n-1})d\mathbf{x}_{n-1} \\ &\approx \sum_{j=1}^{N_p} \tilde{W}_{n-1}^{(j)} p(\mathbf{x}_n^{(i)}|\mathbf{x}_{n-1}^{(j)}). \end{aligned} \quad (10)$$

Substituting (10) into (8) yields the importance weights update:

$$W_n^{(i)} = \frac{p(\mathbf{y}_n|\mathbf{x}_n^{(i)}) \sum_{j=1}^{N_p} \tilde{W}_{n-1}^{(j)} p(\mathbf{x}_n^{(i)}|\mathbf{x}_{n-1}^{(j)})}{q(\mathbf{x}_n^{(i)}|\mathbf{y}_n)}. \quad (11)$$

Using the Gaussian approximation proposal (from the EKF), we can draw samples $\{\mathbf{x}_n^{(i)}\}$ from $q = \mathcal{N}(\hat{\mathbf{x}}_{n|n}, \mathbf{P}_{n|n})$, where $\mathbf{P}_{n|n}$ is obtained from (6). Now equation (11) naturally combines the prior and likelihood information, using previous weights and samples as well as observation $\mathbf{y}_n$. Finally, the master filter produces a (second-stage) mean estimate $\mathbf{x}_n = \sum_{i=1}^{N_p} \tilde{W}_n^{(i)} \mathbf{x}_n^{(i)}$.

In summary, a complete-step of TPF runs as follows:

1. At time n, for $j = 1, \ldots, N_p$, given $\mathbf{x}_{n-1}^{(j)}$ (obtained from the master filter in the previous step) and $\mathbf{y}_n$, run the EKF updates (for the slave filter) to calculate the approximated Gaussian sufficient statistics $(\hat{\mathbf{x}}_{n|n}, \mathbf{P}_{n|n})$.
2. Draw N_p samples $\{\mathbf{x}_n^{(i)}\}$ from $\mathcal{N}(\hat{\mathbf{x}}_{n|n}, \mathbf{P}_{n|n})$.
3. For the master filter, for $i = 1, \ldots, N_p$, calculate the importance weights via (11), and normalize them to get $\{\tilde{W}_n^{(i)}\}$.
4. Calculate the second-stage estimate: $\mathbf{x}_n = \sum_{i=1}^{N_p} \tilde{W}_n^{(i)} \mathbf{x}_n^{(i)}$.
5. Calculate $\hat{N}_{eff}$, if $\hat{N}_{eff} < N_p/2$, perform the resampling.
6. Copy the N_p particles to the slave filter for the next step.

Table 1. Experimental results of bearing-only target tracking based on 50 Monte Carlo runs (excluding the divergence trials).

filter	MSE	NMSE	diver. rate
EKF	0.0026	0.0168	0/50
UKF	0.0045	0.0232	0/50
SIR-prior ($N_p = 100$)	0.0021	0.0150	3/50
SIR-gradient ($N_p = 100$)	0.00014	0.0078	1/50
TPF-EKF ($N_p = 30$)	0.00006	0.0009	0/50
TPF-UKF ($N_p = 30$)	0.00006	0.0008	0/50

3. TRACKING APPLICATIONS

Bearing-Only Tracking: First, we consider a bearing-only tracking benchmark problem [3]. Let $(\nu, \hat{\nu}, \eta, \hat{\eta})$ denote the $x-y$ positions and velocities of a moving target. The state-space equations are described as follows:

$$\begin{aligned} \mathbf{x}_{n+1} &= \mathbf{F}\mathbf{x}_n + \mathbf{C}\mathbf{d}_n, \\ y_n &= \tan^{-1}(\eta_n/\nu_n) + v_n, \end{aligned}$$

where $\mathbf{x}_n = [\nu_n, \dot{\nu}_n, \eta_n, \dot{\nu}_n]^T$, $\mathbf{d}_n = [d_{1,n}, d_{2,n}]^T$, and

$$\mathbf{F} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}.$$

The observation is a noisy bearing, and $\mathbf{d}_n \sim \mathcal{N}(\mathbf{0}, 0.001^2\mathbf{I})$, $v_n \sim \mathcal{N}(0, 0.005^2)$. The goal is to reconstruct the trajectory $\mathbf{x}_{0:n}$ given the observed bearings $y_{0:n}$ and initial condition $\mathbf{x}_0 = [-0.05, 0.001, 0.7, -0.055]^T$. The priors of particle filters are set up as $p(\mathbf{x}_0) \sim \mathcal{N}(\mathbf{0}, \text{diag}\{0.05^2, 0.005^2, 0.03^2, 0.01^2\})$ and $\mathbb{E}_{p(\mathbf{x}_0)}[\mathbf{x}_0] = [-0.06, 0.0015, 0.65, -0.05]^T$. Note that here the prior and likelihood are both peaked.

The experimental comparisons are conducted between (i) EKF; (ii) unscented Kalman filter (UKF); (iii) SIR filter with a prior proposal; (iv) gradient proposal particle filter; (v) TPF with an EKF as slave filter; and (vi) TPF with a UKF as slave filter. A total of 50 Monte Carlo experiments with different random seeds are performed for each filtering scheme, using different number of particles. The error metrics of interest here are the mean-squared error (MSE), the normalized MSE (NMSE), as well as the divergence rate.[1] Experimental results are summarized in Table 1. As seen, the two proposed particle filtering schemes outperform the conventional bootstrap filter. In this example, the TPF produces the best tracking result; but no big difference is observed between (v) and (vi), partially because the state equation is linear and Gaussian. Using the same number of particles, the gradient proposal obviously produces better results than the prior proposal; however, as expected, when N_p gradually increases, the difference between them will be reduced; this has been confirmed in our experiments (see Fig. 2).

Tracking with Coordinate Turn: Next, we consider a typical target tracking through a coordinated turn (CT), where the state-space model is described by a stochastic differential equation (SDE) and approximated by a 2nd-order weak Taylor approximation. See [5] for background and details. Let $\mathbf{x}_t = [\xi_t, \varsigma_t, s_t, \theta_t, \omega_t]^T$ denote the state vector containing target position in x and y coordinate, target speed, heading, as well as turn rate. Under the SDE theory, the constant speed and turn rate (in the ideal CT model) will be instead modified as a Wiener process.

[1] By divergence we mean the filter deviates from the target trajectory and is unable to come back to the true track.

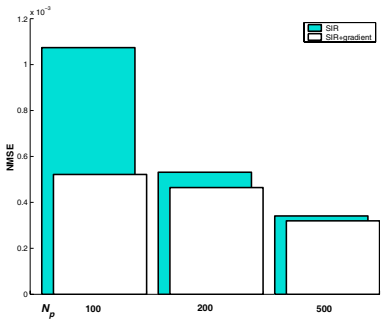

Fig. 2. Performance (NMSE) comparison between the SIR and gradient proposal particle filters using varying numbers of particles, each based on 20 independent Monte Carlo runs.

Specifically, the 2nd-order Taylor approximation of the continuous-time state equation is described as [5]:

$$\mathbf{x}_t = \mathbf{f}(\mathbf{x}_\tau) + \mathbf{G}(\mathbf{x}_\tau)\mathbf{w}_t \tag{12}$$

where $\delta = t - \tau$ (we use $\delta = 1$ in discretization), and

$$\begin{aligned} \mathbf{f}(\mathbf{x}_\tau) &= \begin{pmatrix} \xi_\tau + \delta s_\tau \cos(\theta_\tau) - \delta^2 s_\tau \omega_\tau \sin(\theta_\tau)/2 \\ \varsigma_\tau + \delta s_\tau \sin(\theta_\tau) + \delta^2 s_\tau \omega_\tau \sin(\theta_\tau)/2 \\ s_\tau \\ \theta_\tau + \delta\omega_\tau \\ \omega_\tau \end{pmatrix}, \\ \mathbf{G}(\mathbf{x}_\tau) &= E(\mathbf{x}_\tau)V_\delta, \quad \text{with} \\ E(\mathbf{x}_\tau) &= \begin{pmatrix} \sigma_s\cos(\theta_\tau) & 0 & 0 & 0 \\ \sigma_s\sin(\theta_\tau) & 0 & 0 & 0 \\ 0 & 0 & \sigma_s & 0 \\ 0 & \sigma_\omega & 0 & 0 \\ 0 & 0 & 0 & \sigma_\omega \end{pmatrix}, \\ V_\delta &= \begin{pmatrix} \sqrt{\delta^3/3} & 0 \\ \sqrt{3\delta}/2 & \sqrt{\delta}/2 \end{pmatrix} \otimes \mathbf{I}_{2\times 2} \end{aligned}$$

where $\otimes$ denotes the Kronecker product; $\mathbf{w}_t$ is a Wienner process approximated by standardized white Gaussian noise $\mathcal{N}(\mathbf{0}, \mathbf{I}_{4\times 4})$.

The measurement equation consists of a "range-bearing" pair:

$$\mathbf{y}_t = \left[\sqrt{\xi_t^t + \varsigma_t^2},\ \tan^{-1}(\varsigma_t/\xi_t)\right]^T + \mathbf{v}_t \tag{13}$$

where the Gaussian measurement noise $\mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, \Sigma_\mathbf{v})$ is independent on the initial state and $\mathbf{w}_t$. The data trajectory was generated using the Euler approximation with sampling period of 1 second and 1000 intervals per sampling instant. Measurements are collected for 200 seconds with a constant sampling period. The noise and initial parameter setup in our experiment is as follows: $\sigma_s^2 = 1/5$, $\sigma_\omega^2 = 5 \times 10^{-7}$, $\Sigma_\mathbf{v} = \text{diag}\{100, (\pi/180)^2\}$, $\mathbf{x}_0 \sim \mathcal{N}(\boldsymbol{\mu}_0, \Sigma_0)$ with

$$\begin{aligned} \boldsymbol{\mu}_0 &= [1000, 2650, 150, \pi/2, -\pi/45]^T, \\ \Sigma_0 &= \text{diag}\{400, 400, 25, (5\pi/180)^2, (0.2\pi/180)^2\}. \end{aligned}$$

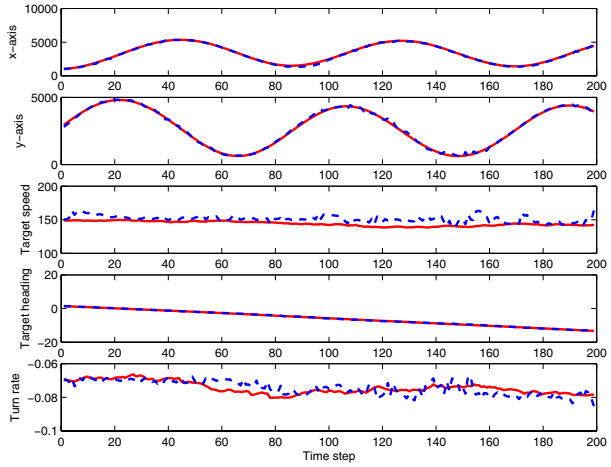

Fig. 3. One typical result in tracking with CT experiment using gradient proposal particle filter ($N_p = 500$, RMSE=87.5).

Note that here the dynamic noise is peaked whereas the measurement noise is rather flat. Table 2 shows the comparison between the bootstrap filter (with prior proposal) and the SIR filter (with gradient proposal) and TPF, with varying number of particles. Fig. 3 shows one typical tracking result.

MIMO Wireless Channel Tracking: Finally, we study a real-life MIMO wireless channel tracking problem (actually it is a channel/sybmol joint estimation problem but we ignore the symbol decoding part due to space limitation) [4]. The real-life wireless narrowband MIMO channel data were recorded in midtown Manhattan, New York city, January 2001. In particular, the state equation of the channel can be described by a first-order AR model driven by non-Gaussian noise (such as the mixture of Gaussians), whereas the measurement equation is described as [4]:

$$y_{j,n} = \sum_{k=1}^{\#receivers} s_{k,n} x_{jk,n} + v_{j,n},$$

where $s_{k,n}$ is the block of encoded symbols radiated by the kth transmitter at time n; $x_{jk,n}$ is the channel coefficient from the kth transmitter to the jth receiver at time n; $y_{j,n}$ is the signal observed at the input of the jth receiver; and $v_{j,n}$ is the measurement noise at the input of the jth receiver at time n, which is modeled by a *Middleton Class A noise model*. See [4] for more details.

We have compared different trackers (including Kalman filter, mixture Kalman filters, and particle filters) [4], but here we only highlight the efficiency of the gradient proposal in this real-life application. As seen from Table 3, the proposed gradient proposal particle filter produces much better results (in terms of MSE as well as symbol error rate) than the conventional SIR filter, especially when using a small number particles. Namely, the particles drawn from the gradient proposal are more informative. This phenomenon has also been evidenced in the previous two applications. In terms of *relative* complexity and performance gain, compared to the Kalman filter (with complexity 1), the relative complexity factors of the SIR (with 100 particles) and our gradient SIR filters (with 20 particles) are 3.2 and 1.5, respectively; while their performance gains are 2.6 and 2.7, respectively! This huge gain is quite significant when complexity issue is concerned in industrial practice.

Table 2. The RMSE $\equiv \sqrt{|\xi - \hat{\xi}|^2 + |\varsigma - \hat{\varsigma}|^2}$ performance comparison (based on 20 Monte Carlo runs) with varying N_p.

Filter	RMSE
SIR-prior ($N_p = 500$)	144.3
SIR-prior ($N_p = 1000$)	91.6
SIR-prior ($N_p = 2000$)	81.9
SIR-gradient ($N_p = 200$)	142.3
SIR-gradient ($N_p = 500$)	90.2
SIR-gradient ($N_p = 1000$)	74.3
TPF-EKF ($N_p = 100$)	158.7

Table 3. The MSE of 2-by-2 MIMO wireless channel estimate and symbol error rate (SER) for various numbers of particles (at 10dB SNR) based on 100 Monte Carlo runs.

Number of	SIR-prior		SIR-gradient	
Particles, N_p	MSE	SER	MSE	SER
10	0.0615	0.0681	0.0233	0.0353
20	0.0431	0.0460	0.0206	0.0305
40	0.0338	0.0387	0.0196	0.0291
100	0.0257	0.0301	0.0188	0.0275
200	0.0227	0.0285	0.0183	0.0272

4. CONCLUDING REMARKS

We have proposed two improved particle filtering schemes and demonstrated their potential merits in three tracking applications. In particular, the ad-hoc gradient proposal is quite efficient in various noise scenarios (esp. with small $\Sigma_{\mathbf{v}}$); whereas the TPF usually works well when variances $\Sigma_{\mathbf{d}}$ and $\Sigma_{\mathbf{v}}$ are both small (since the deterministic EKF gradually reduces the state-error variance). Eventually, the issue of performance and complexity tradeoff of particle filtering is central: choosing a better proposal distribution (with more computation per step) might ironically reduce the total complexity (in terms of required particle number simulation). In practice, we might design certain decision criteria for using different particle filters in different scenarios; in other words, we have to study the problem first and then choose a problem-specific filter.

Acknowledgements: The authors thank Kris Huber for assistance in wireless channel tracking investigation. Z. C. was financially supported by a NSERC grant of Prof. Simon Haykin.

5. REFERENCES

[1] A. Doucet, N. de Freitas, and N. Gordon, Eds. *Sequential Monte Carlo Methods in Practice*, Springer, 2001.

[2] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statist. Comput.*, vol. 10, pp. 197–208, 2000.

[3] N. Gordon, D. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-gaussian Bayesian state estimation," *IEE Proc.-F*, vol. 140, pp. 107–113, 1993.

[4] S. Haykin, K. Huber, and Z. Chen, "Bayesian sequential state estimation for MIMO wireless communication," *Proc. IEEE*, vol. 92, no. 3, pp. 439–454, March 2004.

[5] M. R. Morelande and N. J. Gordon, "Target tracking through a coordinated turn," manuscript submitted to ICASSP2005.