Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2013, Article ID 853430, 8 pages
http://dx.doi.org/10.1155/2013/853430

Research Article

Sensor Scheduling with Intelligent Optimization Algorithm Based on Quantum Theory

Zhiguo Chen,[1] Yi Fu,[1,2] and Wenbo Xu[1]

[1] *Key Laboratory of Advanced Process Control for Light Industry, Ministry of Education, School of IoT Engineering, Jiangnan University, Wuxi 214122, China*
[2] *Research Centre of Environment Science and Engineering, Wuxi 214063, China*

Correspondence should be addressed to Zhiguo Chen; chenzg777@yahoo.com

Received 18 July 2013; Revised 3 September 2013; Accepted 4 September 2013

Academic Editor: Ming Li

The particle swarm optimization (PSO) algorithm superiority exists in convergence rate, but it tends to get stuck in local optima. An improved PSO algorithm is proposed using a best dimension mutation technique based on quantum theory, and it was applied to sensor scheduling problem for target tracking. The dynamics of the target are assumed as linear Gaussian model, and the sensor measurements show a linear correlation with the state of the target. This paper discusses the single target tracking problem with multiple sensors using the proposed best dimension mutation particle swarm optimization (BDMPSO) algorithm for various cases. Our experimental results verify that the proposed algorithm is able to track the target more reliably and accurately than previous ones.

1. Introduction

Nowadays, the optimization problems of real-world increase rapidly, and they are almost nonlinear and often have several local optima, so it is very hard to find the global optimal solution fast. For example in target tracking [1, 2] system, management of different kinds of sensors to get desired result within acceptable time is a very difficult task.

Heuristic search strategies can acquire faster solutions as they need not require the objective functions to be continuous or differentiable. For recent years the particle swarm optimization (PSO) algorithm in terms of social and cognitive behavior with heuristic search strategy has been successfully used in many real applications [3, 4] since it was proposed by Kennedy and Eberhart [5]. Like most biologically inspired algorithms, the PSO is population based, and the individuals which make up the population are referred to as particles. The main advantage of PSO algorithm is that it has fewer parameters to adjust, but when the search space is high its convergence speed will become very slow near the global optimum and get into local optima. PSO algorithm also shows poor quality results when it deals with large and complex data sets, and it often failed in searching for a global optimal solution when the objective function has a large number of dimensions. The reason for this phenomenon is not only the existence of the local optimal solutions especially for some multimodal functions, but also the velocities of the particles sometimes lapsed into degeneracy, so that the successive range was restricted in the subplain of the whole search space [6].

Although it was confirmed that the PSO algorithm has potential to find global optimal solution on some benchmark functions [7], it could get stuck in local optima. So the issue of local optima in PSO algorithm has been studied widely and many different variants of PSO algorithm were proposed. For example, by dynamically adjusting the inertia weight w and acceleration weights c_1, c_2 in PSO algorithm, the convergence speed is increased and the swarm is encouraged to escape from local optimal [8]. Some other mutation techniques are used to mutate the positions of some particles in order to diverge the swarm [9]. However, it is still difficult for the modified methods above to find the global solutions for some complex applications. In this paper, a new mutation method called best dimension mutation (BDM) is proposed, and it enables the particles to converge to global optimum fast and easily escape from local optima. A new perturbation technique inspired by quantum mechanics is also introduced

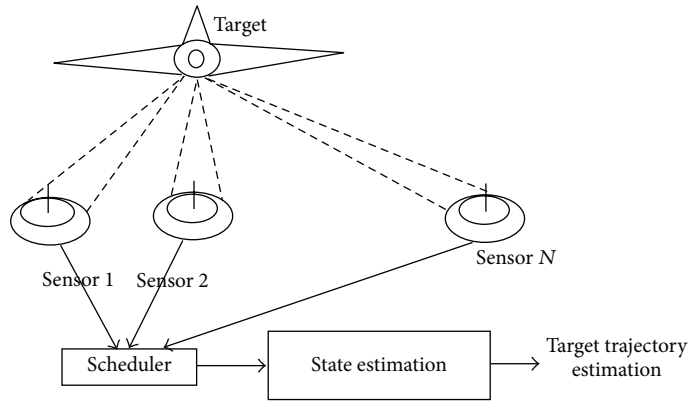

Figure 1: Single target tracking system with multiple sensors.

into original POS algorithm which is used to reinitialize the particles positions without distorting the ongoing search strategy when the particles are too close from each other. By combining the two proposed methodologies, the particles can escape from local optima more easily.

The rest of the paper is organized as follows. Section 2 discusses the tracking of single target using multiple noisy sensors. Section 3 presents the BDMPSO (best dimension mutated particle swarm optimization) algorithm. Section 4 shows our experimental results. Section 5 concludes the paper.

2. Sensor Scheduling for Target Tracking

Figure 1 illustrates such a scene where a single target moves across a geographical area and several sensors are used to track the target simultaneously. The target state $X_k = [x_k \ \dot{x}_k \ y_k \ \dot{y}_k]'$ is unknown and the brief target model is considered as

$$X_{k+1} = AX_k + B\omega_k, \tag{1}$$

where ω_k is white Gaussian process noise with covariance matrix Q. The target state kinematics equation is modeled by system matrix A and the noise intensity is determined by matrix B. The target state is observed by several sensors and the measurement results are impaired by Gaussian noise. At given time k, the measurement from the nth sensor is a column vector given by

$$Z_k^n = H^n X_k + v_k^n, \tag{2}$$

where v_k^n is the measurement noise of each sensor, which is assumed to be independent of other sensors and of the process noise. We make an assumption that covariance matrix R^n and observation matrix H^n are previously known, and the Kalman filtering techniques [10–13] are used to estimate the state of the target. For a given initial error covariance P_0, the estimated error covariance of the state is given by

$$P_{k|k} = P_{k|k-1} - P_{k|k-1}H' S_k^{n-1} H P_{k|k-1}, \tag{3}$$

where

$$\begin{aligned} P_{k|k-1} &= AP_{k-1|k-1}A' + BQB', \\ S_k^n &= HP_{k|k-1}H' + R_n. \end{aligned} \tag{4}$$

The base station (BS), which schedules the sensors to be active or asleep, collects the latest information about the deleted and newly added sensors as soon as the parameters of the target have been obtained. In our work, we assume that only one sensor is scheduled to estimate the state of the target at any time. Therefore, there is only one sensor to be activated by the BS at a time. Let N describe the number of latest available sensors to take measurements and $\mu(T) = \{u_1, u_2, \ldots, u_T\}$ represent the sensor scheduling sequence for $k = 1, 2, \ldots, T$, and let $u_k \in \{1, \ldots, N\}$ be the activated sensor at time step k. Our goal is to minimize the total estimated error of the target states given by the Kalman filter when $k = 1$ to T and to weigh the total sensor usage cost. Calculating the trace of a matrix provides a cheaper and more effective means of defining a scaler based on the estimated error covariance than those of the other ways. Therefore, the trace of the error covariance matrix is considered as our scaler. By using a positive weighting factor w, we can weigh the cost between error and usage. If the cost function J_T is defined to measure the performance of the state's cumulative root mean square error and to weigh sensor usage cost for the time horizon $\{1, 2, \ldots, T\}$, then

$$J_T = \sum_{k=1}^{T} \left\{ P\left(u_k\right) + wc_k\left(u_k\right) \right\}, \tag{5}$$

where $P(u_k) = \sqrt{\text{trace}(P_{k|k}(u_k))}$ represents the square root of the error covariance associated with the selected sensor u_k at instant time k. The nth sensor usage cost at kth instant time is $c_k(n)$ and $\omega \in R^+$ is a constant weight coefficient. Our objective is to find an optimal sensor sequence which can produce the minimum cost at the entire time horizon.

The optimal sensor sequence is described as $\mu^*(T) = \{u_1^*, u_2^*, \ldots, u_T^*\}$:

$$\mu^*(T) = \arg\min_{\forall u_k} \left\{ \sum_{k=1}^{T} \{P(u_k) + wc_k(u_k)\} \right\}. \quad (6)$$

From (6), we can see that there are total N^T possible combinations to schedule the sensors. Up to now many optimal algorithms have been proposed to solve this sensor sequence problem. Dynamic programming (DP) and branch-and-bound (BB) were used to solve such problem, but the computational cost rose exponentially along with the increase in the number of sensors and the sequence length [14].

Rollout Algorithm (RA) is closely approximate to a dynamic programming method which can overcome the exponential computational cost problem [15], and it can produce attractive suboptimal results not worse than traditional heuristic methods.

Sliding Window (SW) is another suboptimal method and its window size will balance the computational cost and final results.

The PSO algorithm can be used to solve above target tracking problem [16], but the quality will dramatically decrease with the increment of the sensors. An improved PSO algorithm will be proposed in Section 3 and it will produce acceptable global optimization result.

3. Best Dimension Mutated Particle Swarm Optimization

3.1. Best Dimension Mutation. Our goal is to develop a methodology which is able to converge faster than PSO algorithm while ensuring the solution is near optimal. In GA (genetic algorithm) or any hybridized PSO algorithm, dimensions of particles are just selected by an assigned probability P for mutation. The performance of PSO algorithm varies with different values of P for different cases. High value of P may result in the swarm divergence and misleading results in some applications. Our methodology aims to speed up the searching process with simple and useful mutation ways. The proposed technique inspired from quantum theory is the key measure to improve the performance of PSO algorithm which can perturb the particles when they are too close from each other.

The PSO algorithm performs very rapid convergence rate at the beginning and will slow down during the remaining searching process [17], and sometimes the velocity of a particle will be zero at the end. This brings about the particles stagnated in local optima [18]. The new mutation technique will relocate the particles to new positions and keep moving around the searching space. The conventional mutation is carried out by selecting one or more dimensions of a particle probabilistically and then added a disturbance factor with special distribution [19, 20].

We developed a new mutation technique called best dimension mutation (BDM) which will find the best dimension to perform mutation at each iteration step, and the selected dimension will be replaced by a random value from the searching space as follows:

(1) randomly select one particle X_i from the swarm population $X_{i_t} = (X_i^1, X_i^2, \ldots, X_i^D)$;

(2) randomly select s number of dimensions in position vector of the selected particle and perform mutation for all selected dimensions separately. The mutated particles X_M are given by $X_M = (X_{M_1}, X_{M_2}, \ldots, X_{M_s})$, where $X_{M_m} = (X_i^1, \ldots, X_i^{d-1}, r, X_i^{d+1}, \ldots, X_i^D)$ and $r = \text{rand}_i^d(\max X - \min X) + \min X$;

(3) select the particle denoted by $X_{M_m}^*$ with the best performance out of s number of mutated particles X_M and the parent particle X_{i_t}, considering problem $X_{M_m}^* = \arg\min_{\forall X} f(X), X \in \{X_{i_t}, X_M\}$;

(4) update position of the particle by the best mutated one. The velocity of the particle can be calculated as $X_{i_{t+1}} = X_{M_m}^*$, $v_{i_{t+1}} = X_{i_{t+1}} - X_{i_t}$.

Figure 2 illustrates the flowchart of the proposed BDM procedure. It was observed that if more number of dimensions were selected at the beginning of the search process, the particles would find better solution in a short time period and the divergence of the swarm was more efficient. However, the computational complexity will increase with more dimensions. Therefore, we experimentally found that the performance and computational efficiency of the algorithm can be increased by selecting all of dimensions for a short period and gradually decreases toward the end of the search process. Figure 3 illustrates the selection of number of dimensions as number of iterations increase. The maximum number of iterations is denoted by IMAX. The mutation is carried out only for one particle in the swarm and it makes an additional s number of fitness evaluation at each iteration step.

3.2. Refreshing Particles Based on Quantum Theory. The performance of the BDM methodology above can be increased by selecting all of the dimensions for a short period of time at the beginning and gradually decreasing toward the end of the searching process. However, the problem of getting stuck in local optima would occur inevitably. Therefore, a perturbation technique is applied to the particles when they are too close from each other. The particle's position can be reinitialized with two different strategies as follows.

(1) Use Gaussian distribution with the mean as global best solution G_t and the variance as L% of the search space to generate the particles:

$$X_{i_t} \underset{X_{i_t} \neq G_t}{\sim} N(G_t, L). \quad (7)$$

(2) Use the delta potential well distribution to generate the particles as used in [21]:

$$X_{i_t} \underset{X_{i_t} \neq G_t}{\sim} G_t \pm \frac{\lg(1/u)}{2g \ln \sqrt{2}} L, \quad (8)$$

where u is a random number in the range $(0, 1]$ and g is a constant variable.

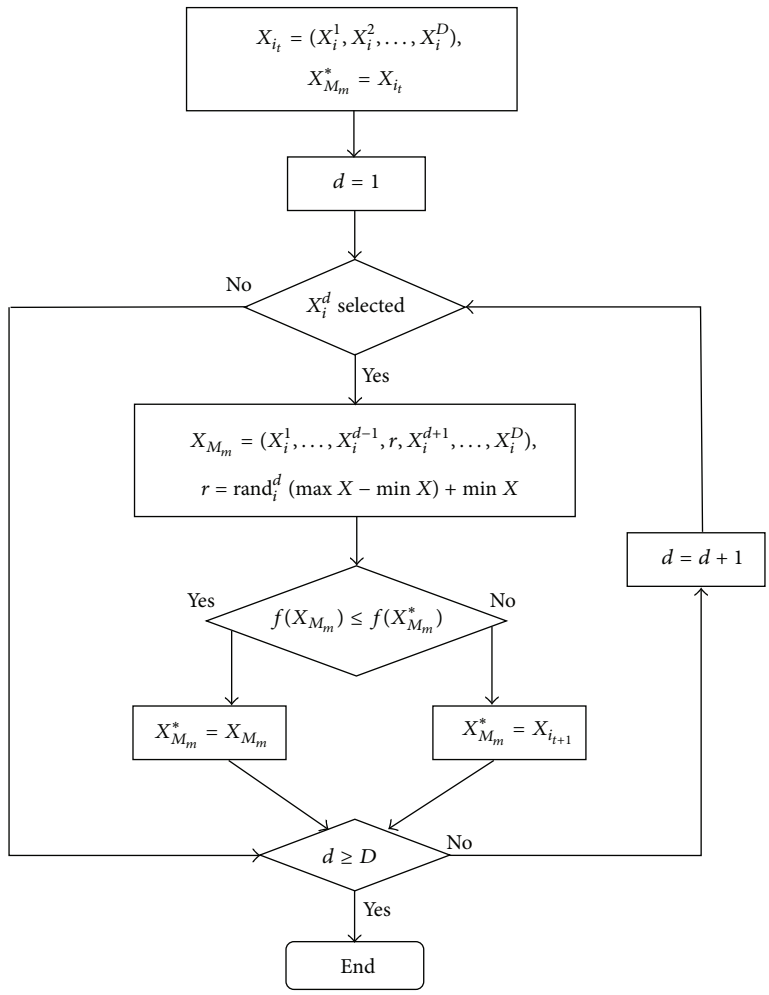

Figure 2: The flowchart of the proposed method of best dimension mutation (BDM).

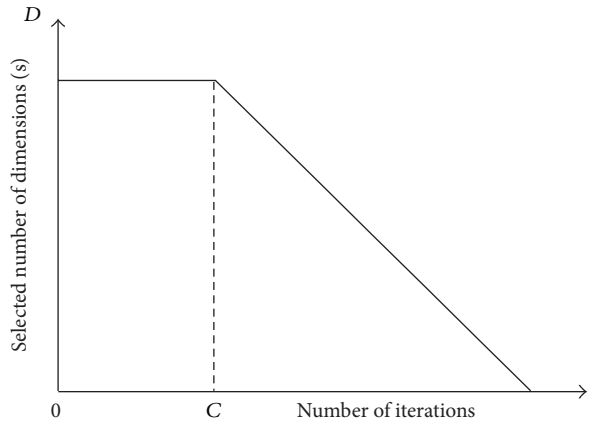

Figure 3: Selection of number of dimensions with number of iterations.

The particles generated by delta potential well distribution and Gaussian distribution are denoted by Qps and Gps, respectively, in Figure 4. Both quantum distribution (delta potential well) and Gaussian distribution have a great chances to generate particles close to *gbest* (global best) position as shown in Figure 4. However, the delta potential distribution produces some particles significantly far away from *gbest* position. This is the tunneling effect; that is, particles have a great chance to reach the global optima even if the global optima should be far away from the local optima [21]. We tested the performance of the PSO algorithm with both particles generating techniques. We experimentally found that the particles generated with quantum based technique in PSO algorithm produce better results than those produced with Gaussian techniques. Therefore, we will use delta potential well to generate the particles when they are too close to each other in this paper.

The parameter L decides the position of the particles around *gbest* position. We can see from Figure 5 that the particles generated with larger L value are comparatively far away from *gbest* (where $G_t = 0$). We assigned larger L from the start and linearly reduced it to zero with the increment of iterations to ensure the convergence of the particles. The algorithm will be very slow if the positions of the particles are calculated in each iteration, so we refresh the particles after t_{cap} predefined iterations. We investigated the impact of t_{cap} on final results and the value was chosen empirically.

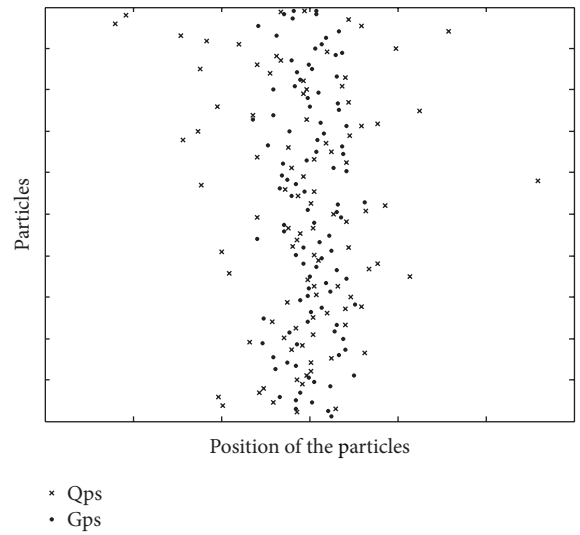

Figure 4: Particles generated with Gaussian distribution and Delta potential well distribution ($L = 3$, $G_t = 0$, and $g = 2.5$).

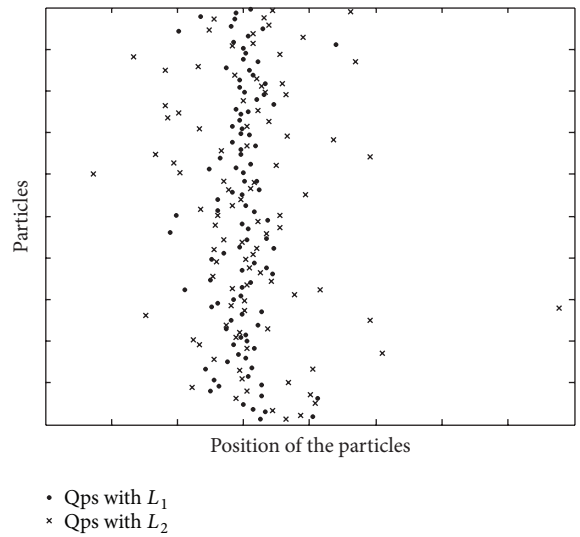

Figure 5: Particles generated with different L values ($L_1 = 2$, $L_2 = 5$, $G_t = 0$, and $g = 2.5$).

3.3. Sensor Scheduling by BDMPSO. We assumed that we did not know the target parameters such as system matrix A, noise intensity B, and covariance matrix Q until the target appeared in the sensor field. Once the target has been detected the base station will awaken available sensors for target observing. After all information about the target and sensors has been collected, there will be a short period of time to schedule the sensor sequence. Therefore, the BDMPSO algorithm will be applied to produce optimal sensor sequence with the minimum iterations. As the number of sensors is discrete, we use rounding method to discretize the dimensions. Also the replacement mechanism is used when a particle goes beyond the search space as shown below:

$$X_{i_t}^d = \text{round}\left[\text{rand}_i^d \left(\max X - \min X\right) + \min X\right]. \quad (9)$$

The flowchart of the proposed BDMPSO is illustrated in Figure 6.

The entire procedure of BDMPSO procedure for sensor scheduling can be described as below.

(1) At $t = 1$ randomly select particles from $\{x_{i_1} \in \{u_1, u_2, \ldots, u_T\}\}_{i=1}^{n_p}$, and each of which indicates a corresponding feasible sensor sequence. Initialize velocity V_{i_0}, own best position p_{i_0}, and global best position G_0, and set temp = 0.

(2) At iteration t, if temp = t_{cap} (a predefined number of iterations), refresh all the particles except the global best one using delta potential well distribution and set temp = 0; else goto Step 3.

(3) Randomly select a particle X_{i_t} from the swarm population and perform the BDM and go to Step 4.

(4) Evaluate the fitness value and update the own best p_{i_t}, global best G_0, velocity V_{i_t}, and position X_{i_t} for all n_p particles. Increment temp by one and go to Step 5.

(5) If $t = I_{\text{MAX}}$ then go to Step 6; else increment t by one and go to Step 2.

(6) The optimal sensor sequence of the problem is returned by the global best particle at the end of the $\{u_1^*, u_2^v, \ldots, u_T^*\} = G_{IMAX}$.

4. Simulation Results

In this section, we tested a single target tracking in a 2-dimensional Cartesian space, and the state of the target evolved with linear Gaussian dynamics. Our experiments used the constant velocity in target model. Therefore, the covariance matrix Q and system matrix A can be described as follows:

$$A = \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & \Delta t & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$Q = q\Delta t = \begin{bmatrix} \frac{\Delta t^2}{4} & \frac{\Delta t}{2} & 0 & 0 \\ \frac{\Delta t}{2} & 1 & \frac{\Delta t}{2} & 0 \\ 0 & 0 & \frac{\Delta t^2}{4} & \frac{\Delta t}{2} \\ 0 & 0 & \frac{\Delta t}{2} & 1 \end{bmatrix}, \quad (10)$$

where the scalar quantity q dominates the intensity of system noise and the state updates with time step Δt. We assigned $q = 20$ and $\Delta t = 1$ for all of the experiments. The sensor measurement showed a linear correlation of the target state and was impaired by white Gaussian noise in (2). All of the sensors are homogeneous, and the error covariances and usage costs differ from one another. The observation matrix H is a unit matrix with 4 by 4 for all sensors. There are 24 sensors used

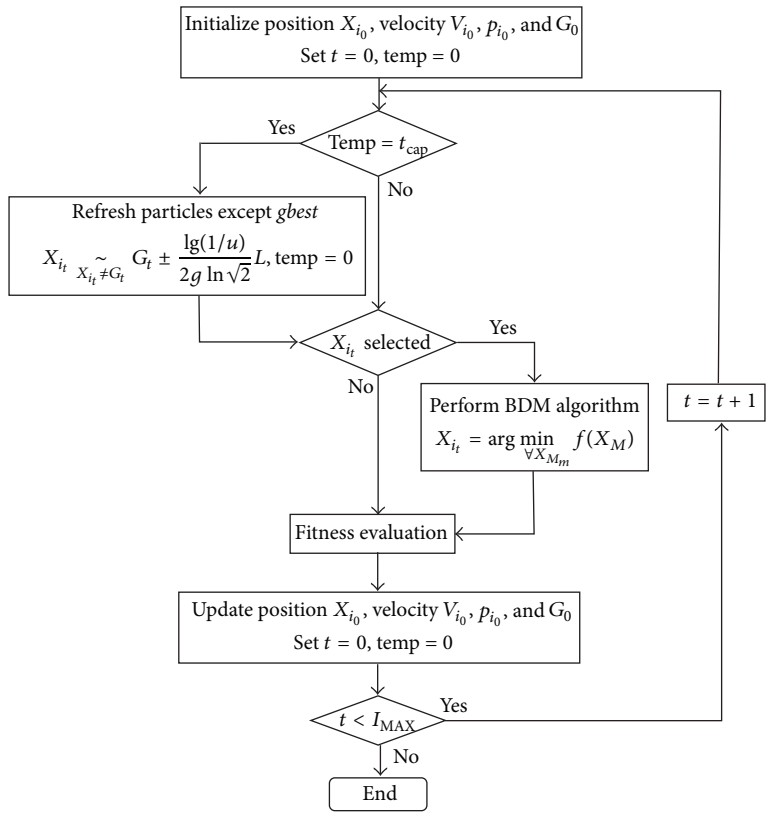

Figure 6: The flowchart of the proposed procedure for the BDMPSO algorithm.

for the experiments all together and the error covariances of which can be described as follows:

$$R_1 = 10^3 \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 1.5 & 0 & 0 \\ 0 & 0 & 200 & 0 \\ 0 & 0 & 0 & 1.5 \end{bmatrix},$$

$$R_2 = 10^3 \begin{bmatrix} 2.5 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 0.5 \end{bmatrix},$$

$$R_3 = 10^3 \begin{bmatrix} 150 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 \\ 0 & 0 & 1.5 & 0 \\ 0 & 0 & 0 & 1.5 \end{bmatrix},$$

$$R_4 = 10^3 \begin{bmatrix} 200 & 0 & 0 & 0 \\ 0 & 1.5 & 0 & 0 \\ 0 & 0 & 200 & 0 \\ 0 & 0 & 0 & 1.5 \end{bmatrix}, \quad (11)$$

where the other sensors' error covariance eigenvalues are larger than R_4. Among the entire time horizon, the usage costs are defined as $c(1) = 65$, $c(2) = 25$, $c(3) = 40$, and $c(n) = 60$, for all $n \geq 4$. Therefore, the optimal solution including R_1, R_2, R_3, and R_4 produced by sensors such that $n > 4$ should be the same as that produced by four sensors only. The intention in generating these covariances is to change the landscape of the problem with the global solution unchanged. For all experiments the weighting factor w in (5) is assigned 1. We have measured the performance of the proposed method in contrast to PSO with time varying parameters (PSO-tp), Comprehensive Learning PSO [22] (CLPSO), Rollout Algorithm using PSO (RA1), Rollout Algorithm using one step-look-ahead (RA2), and Sliding Window (SW) approach. The set values of BDMPSO are $t_{cap} = 30$, $L = 2\%$, $c = 40$, and $n_p = 30$ and the rest parameters are the same as in [23].

Firstly, we tested the ability of proposed algorithms with the length of the sensor scheduling sequences varying. Table 1 shows the average results over 30 independent runs for all six different methods in different time horizons (the best costs are highlighted). For the algorithms above, the maximum fitness evaluation (FE) is 20000 with the exception of BDMPSO, being 15000. We can see from Table 1 that the performances of the RA2, SW, and PSO-tp decrease with the sequence length increasing. However, the performances of BDMPSO will not be affected by the increasing number of dimensions. By combining best dimension mutation and new refreshment technology, the performance of the BDMPSO algorithm makes a much more enhancement than others.

Secondly, we analyzed the performance of the PSOs (PSO algorithms) by varying numbers of sensors. Table 2 shows the average results over 30 independent runs for (the best costs are highlighted) all six different methods. The fitness evaluations (FEs) are assigned to 20000 for all the PSOs. As the SW and RA2 are deterministic means, the results are the same for different cases. The performances of RA2 and

Table 1: Comparison of cumulative cost for different time horizons.

T	SW (10^3)	RA1 (10^3)	RA2 (10^3)	PSO-tp (10^3)	CLPSO (10^3)	BDMPSO (10^3)
30	2.576	2.641	2.498	2.479	2.542	**2.396**
40	3.295	3.592	3.426	3.192	3.252	**3.159**
50	4.315	3.971	4.201	3.859	3.883	**3.736**
60	4.855	4.732	5.076	4.621	4.597	**4.527**
70	5.662	5.319	6.125	5.302	5.304	**5.204**
80	6.174	6.201	6.823	6.343	5.902	**5.842**
90	6.822	6.924	7.418	6.912	**6.587**	6.721
100	7.252	7.427	8.421	7.724	7.403	**7.196**
110	8.961	8.093	9.381	8.581	8.067	**8.036**
120	9.382	8.801	10.24	9.304	8.812	**8.512**

Table 2: Comparison of cumulative cost for different numbers of sensors.

N	SW (10^3)	RA1 (10^3)	RA2 (10^3)	PSO-tp (10^3)	CLPSO (10^3)	BDMPSO (10^3)
3	7.476	7.324	7.389	7.395	7.557	**7.242**
6	7.476	7.406	**7.389**	7.413	7.562	7.392
9	7.476	7.364	7.389	7.630	7.642	**7.331**
12	7.476	7.426	7.389	7.598	7.585	**7.329**
15	7.476	7.524	7.389	7.621	7.612	**7.326**
18	7.476	7.631	7.389	7.655	7.620	**7.346**
21	7.476	7.630	7.389	7.623	7.632	**7.363**
24	7.476	7.762	7.389	7.692	7.593	**7.379**

BDMPSO are better than the others'. However, it can be seen from Table 2 that the performances of PSO-tp and CLPSO decrease when the number of sensors increases, and easily get stuck in local optima. The BDMPSO produces the better results for all of the cases.

Thirdly, we compared the performance of PSOs with different numbers of FEs. The parameters were assigned to the same numbers except for the FEs which varied from 10000 to 40000. Table 3 shows the average results over 30 independent runs for (the best costs are highlighted) all six different methods with sequence length of 100 units. It can be seen from Table 3 that the proposed BDMPSO achieves better result than other variant PSO algorithms and it converges to desired results in fewer iterations, but other PSOs will need considerably more numbers of FEs to achieve comparative result. Therefore, the proposed BDMPSO algorithm is more suitable for nearly linear problems in less time.

5. Conclusions

We have presented an improved PSO algorithm and applied it to the target tracking problem. The best dimension mutation method with quantum based reinitialization technique makes the particles escape from the local optima correspondingly easier and accelerates the convergence rate. Relatively this

Table 3: Comparison of cumulative cost for different FEs.

FEs	PSO-tp (10^3)	CLPSO (10^3)	BDMPSO (10^3)
Number of sensors $N = 3$			
10000	7.812	7.509	**7.348**
15000	7.712	7.496	**7.337**
20000	7.712	7.400	**7.315**
30000	7.702	7.327	**7.298**
40000	7.642	7.307	**7.297**
Number of sensors $N = 6$			
10000	7.713	8.904	**7.405**
15000	7.532	7.987	**7.359**
20000	7.501	7.640	**7.313**
30000	7.487	7.462	**7.297**
40000	7.486	7.462	**7.297**
Number of sensors $N = 9$			
10000	9.387	9.846	**7.409**
15000	9.346	9.105	**7.412**
20000	8.615	8.510	**7.406**
30000	8.559	7.559	**7.413**
40000	8.578	7.542	**7.357**
Number of sensors $N = 12$			
10000	11.762	10.824	**7.768**
15000	11.516	9.637	**7.422**
20000	11.327	9.120	**7.425**
30000	10.678	8.249	**7.516**
40000	10.519	**7.348**	7.496
Number of sensors $N = 15$			
10000	12.513	11.305	**8.098**
15000	12.249	10.190	**7.416**
20000	12.301	9.598	**7.387**
30000	12.209	8.315	**7.375**
40000	11.879	7.627	**7.375**
Number of sensors $N = 18$			
10000	14.138	11.467	**8.304**
15000	14.030	10.654	**7.412**
20000	13.975	10.213	**7.327**
30000	13.567	8.649	**7.310**
40000	13.439	7.781	**7.309**
Number of sensors $N = 21$			
10000	14.725	11.496	**8.398**
15000	14.098	10.912	**7.406**
20000	14.134	10.146	**7.378**
30000	13.320	8.697	**7.294**
40000	13.140	8.416	**7.292**
Number of sensors $N = 24$			
10000	15.412	11.849	**8.792**
15000	14.678	11.102	**7.509**
20000	14.355	10.534	**7.376**
30000	14.297	8.912	**7.324**
40000	14.295	8.246	**7.324**

methodology does not increase the computational cost significantly compared with the original PSO algorithm and also it is more easily to implement. To sum up, the proposed BDMPSO algorithm shows better performance and achieves the results faster than other existing ones for target tracking problem. We plan to focus on shortening the convergence time of BDMPSO algorithm in further work.

Acknowledgments

This work was supported in part by the Fundamental Research Funds for the Central Universities under Grant no. JUSRP111A46 and the National Natural Science Foundation of China under Grant no. 61170119.

References

[1] H. S. Meyerhoff, F. Papenmeier, and M. Huff, "Object-based integration of motion information during attentive tracking," *Perception*, vol. 42, pp. 119–121, 2013.

[2] G. Jahn, J. Wendt, M. Lotze, F. Papenmeier, and M. Huff, "Brain activation during spatial updating and attentive tracking of moving targets," *Brain and Cognition*, vol. 78, no. 2, pp. 105–113, 2012.

[3] S. Rana, S. Jasola, and R. Kumar, "A review on particle swarm optimization algorithms and their applications to data clustering," *Artificial Intelligence Review*, vol. 35, no. 3, pp. 211–222, 2011.

[4] A. A. A. Esmin and S. Matwin, "HPSOM: a hybrid particle swarm optimization algorithm with genetic mutation," *International Journal of Innovative Computing, Information and Control*, vol. 9, no. 5, pp. 1919–1934, 2013.

[5] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the 1995 IEEE International Conference on Neural Networks*, pp. 1942–1948, December 1995.

[6] R. Thangaraj, M. Pant, A. Abraham, and V. Snasel, "Modified particle swarm optimization with timevarying velocity vector," *International Journal of Innovative Computing, Information and Control*, vol. 8, no. 1, pp. 201–218, 2012.

[7] S. Baskar and P. N. Suganthan, "A novel concurrent particle swarm optimization," in *Proceedings of the Congress on Evolutionary Computation (CEC '04)*, vol. 1, pp. 792–796, June 2004.

[8] Y. V. Pehlivanoglu, "A new particle swarm optimization method enhanced with a periodic mutation strategy and neural networks," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 3, pp. 436–452, 2013.

[9] Y. Zhang and L. Wu, "A hybrid TS-PSO optimization algorithm," *Journal of Convergence Information Technology*, vol. 6, no. 5, pp. 169–174, 2011.

[10] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME*, vol. 82, no. 1, pp. 35–45, 1960.

[11] L. Xie, P. Yang, T. Yang, and M. Li, "Dual-EKF-based real-time celestial navigation for lunar rover," *Mathematical Problems in Engineering*, vol. 2012, Article ID 578719, 16 pages, 2012.

[12] M. Li, "Approximating ideal filters by systems of fractional order," *Computational and Mathematical Methods in Medicine*, vol. 2012, Article ID 365054, 6 pages, 2012.

[13] M. Li, S. C. Lim, and S. Chen, "Exact solution of impulse response to a class of fractional oscillators and its stability," *Mathematical Problems in Engineering*, vol. 2011, Article ID 657839, 9 pages, 2011.

[14] A. C. Nearchou, "Solving the single machine total weighted tardiness scheduling problem using a hybrid simulated annealing algorithm," in *Proceedings of the 2nd IEEE International Conference on Industrial Informatics (INDIN '04)*, pp. 513–516, June 2004.

[15] D. P. Bertsekas, *Dynamic Programming and Optimal Control. Volume 1*, Athena Scientific, Nashua, NH, USA, 3rd edition, 2005.

[16] H. T. Yao, H. Q. Chen, and T. F. Qin, "Niche PSO particle filter with particles fusion for target tracking," *Applied Mechanics and Materials*, vol. 239, pp. 1368–1372, 2013.

[17] M. J. A. Hasan and S. Ramakrishnan, "A survey: hybrid evolutionary algorithms for cluster analysis," *Artificial Intelligence Review*, vol. 36, no. 3, pp. 179–204, 2011.

[18] F. S. Levin, *An Introduction to Quantum Theory*, Cambridge University Press, New York, NY, USA, 2002.

[19] Z. Chen, Y. Bo, P. Wu, and W. Zhou, "A new particle filter based on organizational adjustment particle swarm optimization," *Applied Mathematics & Information Sciences*, vol. 7, no. 1, pp. 179–186, 2013.

[20] Y. Bo, Z. Chen, J. Zhang, and J. Zhu, "New clone particle swarm optimization-based particle filter algorithm and its application," *Applied Mathematics & Information Sciences*, vol. 7, no. 1, pp. 171–177, 2013.

[21] S. M. Mikki and A. A. Kishk, "Quantum particle swarm optimization for electromagnetics," *IEEE Transactions on Antennas and Propagation*, vol. 54, no. 10, pp. 2764–2775, 2006.

[22] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, 2006.

[23] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240–255, 2004.