Proceedings of the 2009 IEEE
International Conference on Mechatronics and Automation
August 9 - 12, Changchun, China

Particle Filter Based Robust Mobile Robot Localization

Dongsheng Wang
Department of Computer Engineering
Henan Polytechnic Institute
Nanyang, Henan Province, China, 473009
nywj@sina.com

Jianchao Zhao, Wei Wang
Department of Computer Engineering
Henan Polytechnic Institute
Nanyang, Henan Province, China, 473009
zjcl@sina.com, wangwei_ha@163.com

***Abstract* – Mobile robot localization is an important issue in the service robotics area, which is to determine the position of a robot given a map of its environment. In this paper, we described a robust self-localization approach for mobile robot based on particle filtering, which is a sophisticated model for robust estimation. In this method a large number of hypothetical current particles are initially generated to represent the possible robot position, with each sensor update, the probability that each hypothetical particle is updated based on Bayesian principle. Similarly, every robot motion is also applied in a statistical way to the particles based on the statistical motion model. Experimental results demonstrated good performance and robustness of our approach.**

Index Terms - Particle Filtering, Mobile Robot, Robust Localization.

I. Introduction

Mobile robot localization is the problem of determining a robot's pose from sensor data in the environment. The localization problem is a key problem in mobile robotics for long term reliable operation, it plays a pivotal role in various successful mobile robot systems [3][4][5]. The most simple localization problem, which has received by far the most attention in the literature is position tracking [6][7][8]. Here the initial robot pose is known, and the problem is to compensate incremental errors in a robot's odometry.

The vast majority of existing algorithms address only the position tracking problem. The nature of small, incremental errors makes algorithms such as Kalman filters[9] applicable, which have been successfully applied in a range of fielded systems [10]. Kalman filters estimate posterior distributions of robot poses conditioned on sensor data. Exploiting a range of restrictive assumptions, such as Gaussian noise and Gaussian distributed initial uncertainty, they represent posteriors by Gaussians. Kalman filters offer an elegant and efficient algorithm for localization. However, real data can be very complicated, typically involved elements of non-Gaussian, high dimensional and nonlinear, and also the single Extended Kalman Filters (EKF) approach so far does not really behave well in large environments, the big problem is that linearization errors compound and linearization starts to occur around points that are far from the true state. Broadly though, one of two schemes are adopted. The first one divides the world into local maps and uses uni-modal methods (EKF) within each map. The maps are then carefully glued back together to form a unified global map. Secondly uni-modal methods are abandoned altogether. Instead, Monte Carlo based particle filtering methods are used [11].

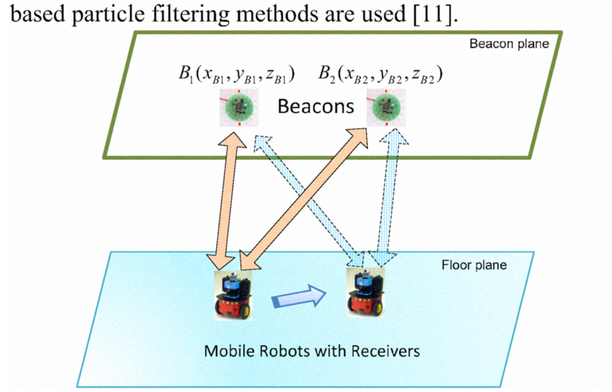


Fig. 1 Self-Localization based on active beacon system

Monte Carlo Localization is a family of algorithms for localization based on particle filters, which are approximate Bayes filters [12] that use random samples for posterior estimation. The basic idea is simple. We maintain lots of different versions of the state vector, all slightly different from each other. When a measurement comes in, we score how well each version explains the data. We then make copies of the best fitting vectors and randomly perturb them to form a new generation of candidate states. Collectively these thousands of possible states and their scores define the probability density distribution (*pdf*) what we are seeking to estimate. We never have to make the assumption of Gaussian noise or perform a linearization.

The remainder of this paper is organized as follows: In Section II, we briefly introduce the system of mobile robot localization by active beacon system. Section III demonstrates the particle filtering. In Section IV details of localization algorithm by particle filtering are explained, which includes prediction, resampling and final estimation. Experimental results are presented in Section V. Our conclusions follow in Section VI.

II. Self-Localization System of Mobile Robot

One of most reliable solutions to the localization problem is to design and deploy an active beacon system specifically for the target environment [1]. This is the preferred technique of ensuring the highest possible reliability of localization. The paradigm of the active beacon system is shown in Fig. 1[2], in the system there are ultrasonic beacons receivers. The beacons are placed on the ceiling and the receiver is fixed in

978-1-4244-2693-5/09/$25.00 ©2009 IEEE

the mobile robot. The trick here is to send simultaneously Radio Frequency (RF) signal and ultrasonic pulse, then to measure the delay between both's impacts, and beacon send distance information to the mobile robot with RF. So, mobile robot can estimate its position based on triangle relationship, the number of beacon can be changed based on the real environment. But the number should be kept at least two. In the active beacon system, system and measurement equations are given as follows as (1) and(2):

$$r_k = f(r_{k-1}, w_{k-1})$$
$$\begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \end{bmatrix} + \begin{bmatrix} \cos\theta_{k-1} & 0 \\ \sin\theta_{k-1} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} d \\ \Delta\theta_{k-1} \end{bmatrix} + w_k \tag{1}$$

$$z_k = h(r_k^i, v_k) = \begin{bmatrix} (x_k^i - x_{Bi})^2 + (y_k^i - y_{Bi})^2 + (z_k^i - z_{Bi})^2 \\ (x_k^i - x_{Bj})^2 + (y_k^i - y_{Bj})^2 + (z_k^i - z_{Bj})^2 \end{bmatrix} + v_k \tag{2}$$

Where $r_k = [x_k\ y_k\ \theta_k]^T$ and z_k are state and measurement of k^{th} step, where w_k and v_k are mutually independent noise of k^{th} step, $(x_i\ y_i\ z_i)$ is the position of beacon. Input $[d\ \Delta\theta_{k-1}]^T$ represents displacement of position and heading.

III. PARTICLE FILTERING

Many problems in science require estimation of the state of a system that changes over time using a sequence of noisy measurements made on the system. In order to analyze and make inference about a dynamic system, at least two models are required: First, a model describing the evolution of the state with time (the system model) and, second, a model relating the noisy measurements to the state (the measurement model). Particle filters had their beginning in the 1940s with the work of Metropolis, and Norbert Wiener suggested something much like particle filtering as early as 1940. But only since the 1980s computational power has been adequate for their implementation. Particle filtering goes by many names, including sequential importance sampling, bootstrap filtering, interacting particle approximations, Monte Carlo filtering, sequential Monte Carlo filtering, and survival of the fittest. The particle filter aims to estimate the sequence of hidden parameters, x_k for $k=0,1,2,3,\ldots$, based only on the observed data y_k for $k=0,1,2,3,\ldots$. All Bayesian estimates of x_k follow from the posterior distribution $p(x_k|y_{0:k})$.

Particle filtering is a probability based state estimator, which using Bayesian approach for state estimation. Actually, particle filter implement the Bayesian estimation numerically. The main idea is very simple and intuitive. At the beginning of the estimation problem, particles (state vector) are randomly generated with a given number N based on the initial *pdf* $p(x_0)$ (which is assumed to be known), and then each particle should be evaluated (assign probability) based on the measurement, the evaluated particles are called posterior states, given the posterior particles, we can compute any desired statistical measure of the *pdf*, but normally we typically are most interested in computing the mean and covariance. The particle filter can be summarized as follows:

1. The system and measurement equations are given as in (3):

$$\begin{cases} x_{k+1} = f_k(x_k, \omega_k) \\ y_k = h_k(x_k, v_k) \end{cases} \tag{3}$$

Where ω_k and v_k are mutually independent and identically distributed (i.i.d) sequences with known probability density functions, frequently used parameters are define in Table. 1.

Parameter	Meaning
$x_{k,i}^-$	the *ith* priori particle before measurement y_k
$x_{k,i}^+$	the *ith* posterior particle after measurement y_k

Table.1 Meaning of some parameters in PF

2. Assume that the *pdf* of the initial state $p(x_0)$ is known, randomly generate N initial particles on the basis of the *pdf* $p(x_0)$, These particles are denoted as $x_{0,i}^+$ $(i = 1,2,\ldots, N)$. N is the number of particles as in Fig. 2, N should be chosen as a trade-off between the computational effort and estimation accuracy. In our application, N is equal to 200;

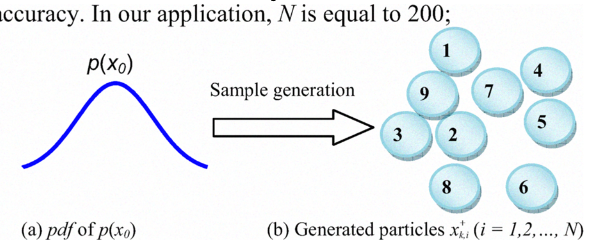


Fig. 2 Particles initialization

For k=1,2, … , perform the following steps.

3. Perform the time propagation step to obtain a priori particle $x_{k,i}^-$ $(i = 1,2,\ldots, N)$, using the known dynamic process equation and the known *pdf* of the process noise as in Fig. 3.

$$x_{k,i}^- = f_{k-1}(x_{k-1,i}^+, \omega_{k-1}^i) \quad (i = 1,\ \ldots,\ N) \tag{4}$$

Where each ω_{k-1}^i noise vector is randomly generated on the basis of the known *pdf* of ω_{k-1}.

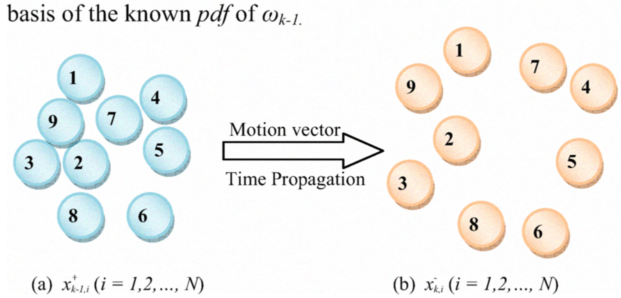


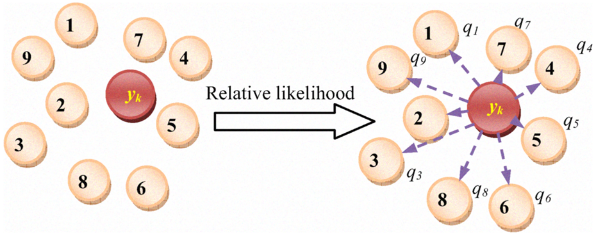
Fig. 3 Time Propagation of Particles

4. Compute the relative likelihood q_i of priori particle $x_{k,i}^-$, conditioned on the measurement y_k; this is done by evaluating the *pdf* $p(y_k|x_{k,i}^-)$ on the basis of the nonlinear measurement equation and the *pdf* of the measurement noise as in Fig. 4.

5. Normalize the relative likelihood q_i obtained in the previous step as follows:

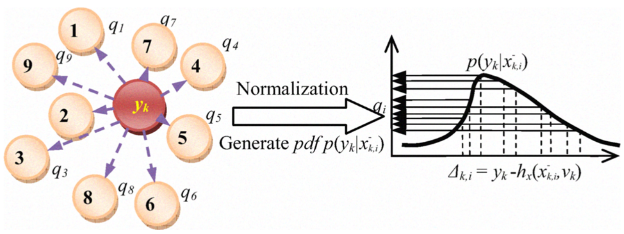
$$q_i = \frac{q_i}{\sum_{j=1}^{N} q_j} \tag{5}$$

Now the summation of all q_i is equal to one as in Fig. 5.



(a) Measurement y_k is available (b) Compute the relative likelihood q_i

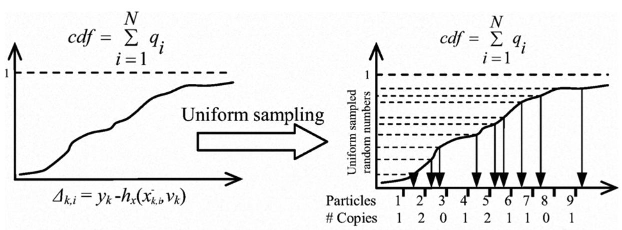
Fig. 4 Compute the relative likelihood q_i of each particle $x^{-}_{k,i}$



(a) Relative likelihood q_i (b) Normalized relative likelihood q_i

Fig. 5 Compute the normalized relative likelihood q_i of each particle $x^{-}_{k,i}$

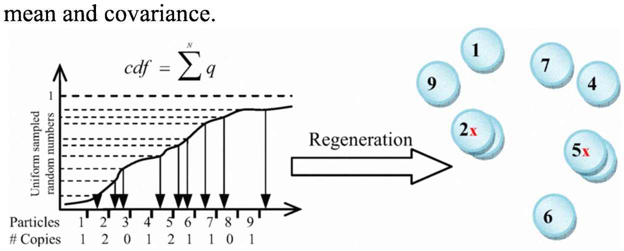
6. Generate a set of posterior particles $x^{+}_{k,i}$ ($i = 1,2,\ldots, N$), on the basis of the relative likelihood q_i, this is the resampling step (for example, by inversion of *cdf*, order statistics, stratified sampling and residual sampling, systematic sampling, etc), Fig. 6 shows the resampling by inversion of *cdf*.



(a) *cdf* of $p(y_k|x^{-}_{k,i})$ (b) sampling via inversion of *cdf*

Fig. 6 Resampling via inversion of *cdf*

7. Now that we have a set of particles $x^{+}_{k,i}$ ($i = 1,2,\ldots, N$) as in Fig. 7, which shows 2nd and 5th particle are sampled with two times, that are distributed according to the *pdf* $p(x_k|y_k)$. Normally we typically are most interested in computing the mean and covariance.



(a) Resampling via inversion of *cdf* (b) Posterior particles $x^{+}_{k,i}$ ($i = 1,2,\ldots, N$)

Fig. 7 Generate posterior particles $x^{+}_{k,i}$ via resampling

IV. Detail of Implementation Algorithm

During the exploration and localization of the unknown environment, the robot maintains a set of hypotheses with regard to their position and the position of the different objects around them, the input for updating these beliefs comes from the various sensors, An "optimal estimator" can be employed in order for the mobile robots to update their beliefs as accurately as possible. More specifically, the position of the obstacle observed in the past can be updated every time when more information becomes available. Moreover, after an action, the estimate of the pose of the robot can be updated based on the information collected up to that point in time, which is the filtering process, this is the main contribution of our particle filtering. Our approach for this simulation is to use landmarks (including computational vision, sonar or laser range finding) in the environment in order to localize frequently and thus reduce the odometry error. We select a collection of landmarks in known positions and inform the robot beforehand.

A. *Robot Vehicle model and odometry*

This is a good point to write down a simple motion model for mobile robotic vehicle. We allow the vehicle to move on a 2D surface (a floor) and point in arbitrary directions. We parameterize the vehicle pose x_v (the joint of position and orientation) as $x_v = [x_v\ y_v\ \theta_v]^T$, Fig. 8 is a diagram of non-holonomic (local degrees of freedom less than global degree of freedom) vehicle with steering. the angle of the steering wheel is given by φ and the instantaneous forward velocity is V. with reference to Fig. 8, we immediately see that:

$$\dot{x}_v = V\cos(\theta_v) \qquad \dot{y}_v = V\sin(\theta_v) \tag{6}$$

Using the instantaneous center of rotation we can calculate the rate of change of orientation as a function of steering angle:

$$L/a = \tan(\varphi),\ a\dot{\theta}_v = V \Rightarrow \dot{\theta}_v = V\tan(\varphi)/L \tag{7}$$

We can now discretise this model by inspection:

$$x_v(k+1) = f_k(x_v(k), u_k);\quad u_k = [V(k)\quad \varphi(k)]^T \tag{8}$$

$$\begin{bmatrix} x_v(k+1) \\ y_v(k+1) \\ \theta_v(k+1) \end{bmatrix} = \begin{bmatrix} x_v(k) + V(k)\cos(\theta_v(k)) \\ y_v(k) + V(k)\sin(\theta_v(k)) \\ \theta_v(k) + V(k)\tan(\varphi(k))/L \end{bmatrix} \tag{9}$$

Note that we have started to lump the throttle and steer into a control vector, which makes sense if we think about the controlling actions of a human driver. Equation (9) is a model for a perfect, noiseless vehicle. Clearly this is a little unrealistic, so we need to model uncertainty. One popular way to do this is to insert noise terms into the control signal u such that $u(k) = u_n(k) + \omega(k)$, where $u_n(k)$ is a nominal (intended) control signal and $\omega(k)$ is a zero Gaussian distributed noise vector as in (10),

$$v(k) \sim \mathrm{N}\left(0, \begin{bmatrix} \sigma_V^2 & 0 \\ 0 & \sigma_\varphi^2 \end{bmatrix}\right),\quad \mathrm{u}(k) \sim \mathrm{N}\left(\mathrm{u}_n(k), \begin{bmatrix} \sigma_V^2 & 0 \\ 0 & \sigma_\varphi^2 \end{bmatrix}\right) \tag{10}$$

This completes a simple probabilistic model of a robot vehicle. we shall now see how particle filtering affects the estimation of the robot vehicle location.

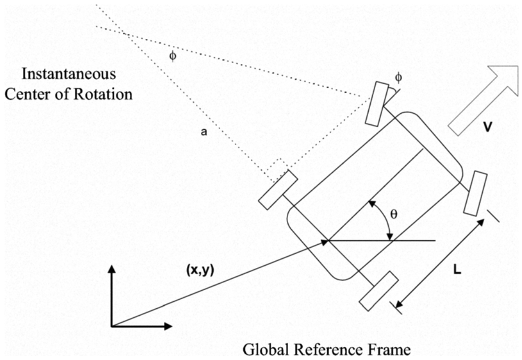


Fig. 8 A non-holonomic robot vehicle with steering

B. Particle Filtering Localization Algorithm

Since we already generate the system equation and uncertainty assignment above, the detail of the particle filtering localization algorithm as follows:

1. Initialize *n* particles in a map. Each particle is a 3 by 1 state vector of the vehicle

2. Apply the plant model to each particle. In contrast to the prediction step in the Kalman approach we actually inject the process noise as well, i.e, we add a random vector to the control vector *u*.

3. For each particle predict the observation. Compare this to the measured value. Use a likelihood function to evaluate the likelihood of the observation given the state represented by each particle. This will be a scalar *L* which is associated with each particle. This scalar is referred to as a "weight" or "importance".

4. Select the particles that best explain the observation. One way to do this would be to simply sort and select the top candidates based on *L* but this would reduce the number of particles. One solution to this would be to copy from this "winning set" until we have n particles again. This has a hidden consequence though: it would artificially reduce the diversity (spread) of the particle set. Instead we randomly sample from the samples biased by the importance values. This means that there is a finite chance that particles that really didn't explain the observation well will be reselected. This may turn out to be a good plan because they may do a better job for the next observation.

5. Goto 2

A common question is "what is the estimate?". Well technically it is the distribution of particles themselves which represents a *pdf*. If we want a single vector estimate, the mode, mean and median are all viable options.

V. Experimental Results

Firstly, we will examine how an initial uncertainty in vehicle pose increases over time as the vehicle moves when only the control signal u is available. Fig. 9 shows the uncertainty evolution. This is an important point to make here that we must understand. In actual real life the real robot is integrating the noisy control signal. The true trajectory will therefore always drift away from the trajectory estimated by the algorithms running inside the robot. This is exactly the same as closing our eyes and trying to walk across University Parks. Your inner ears and legs give you u which you pass through your own kinematic model of your body in your head. Of course, one would expect a gross accumulation of error as the time spent walking "open loop" increases. The point is that all measurements such as velocity and rate of turn are measured in the vehicle frame and must be integrated, along with the noise on the measurements.

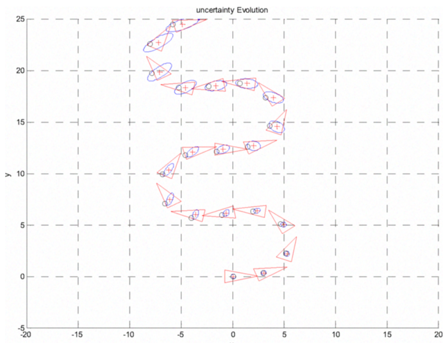


Fig. 9 Uncertainty evolution. The circles are the true location of the mobile robot whereas the crosses mark the estimated locations. The orientation of the vehicle is made clear by the orientation of the triangles. The uncertainty of each estimated location is represented as ellipses.

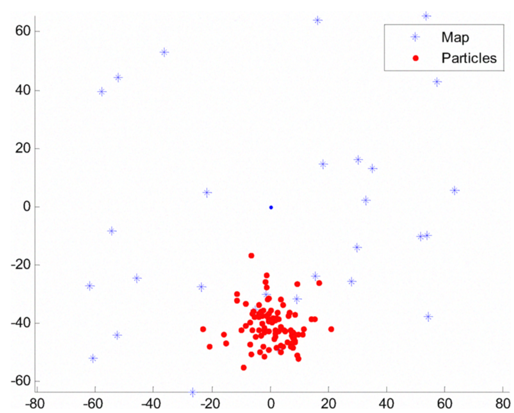


Fig. 10 Initial map and first particles are generated, green is the map, red is the particles, map means the some sensors that can communicate with robots, which used to get observations.

We suppose that the world is populated by a set of discrete landmarks or features whose location/orientation and geometry (with respect to a defined coordinate frame) can be described by a set of parameters, We are given a map *M* containing a set of features and a stream of observations of measurements as in Fig. 10 between the mobile robot and these features, we assume to begin with that an oracle is telling us the associations between measurements and observed features. We have already had the prediction equations from previous discussions.Fig. 11 shows the trajectory of the mobile robot as it moves through a field of

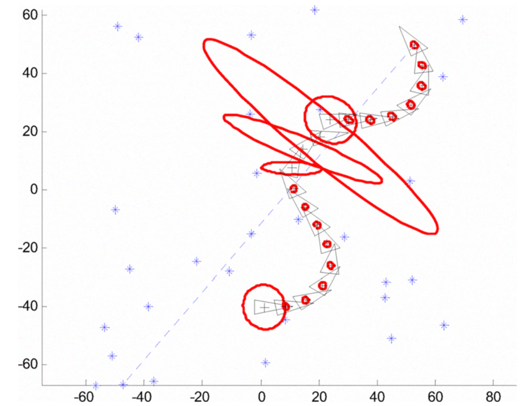


Fig. 11 Mobile robot localization based on particle filtering

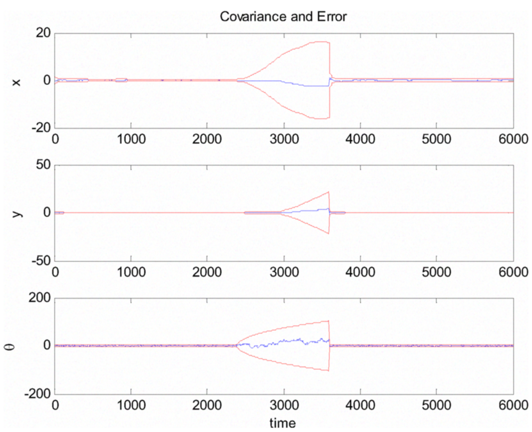


Fig. 12 Feature based localization innovation and error/covariance plots. Notice how when no measurements are made the vehicle uncertainty grows rapidly but reduces when features are re-observed. The covariance bounds are 3-sigma and 1-sigma-bound on state error and innovation plots respectively.

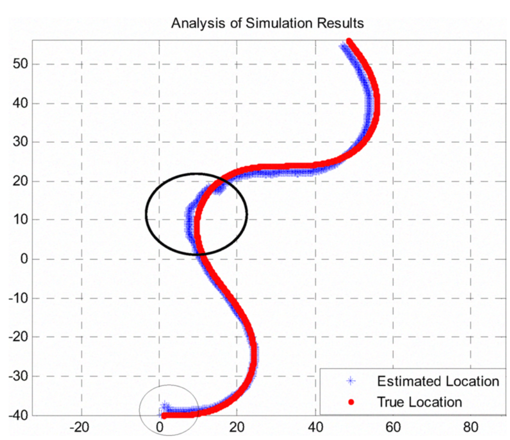


Fig. 13 Simulation results analysis , the difference between the estimated and true location is big in the ellipse area, because this part simulate the mobile robot cannot get the measurement since the sensor failure or other reasons which make the measurement cannot be obtained.

random point features. Note that our approach simulates a sensor failure for the middle twenty percent of the mission. During this time the mobile robot becomes more and more lost. When the sensor comes back on line there is a jump in estimated mobile robot location back to one close to the true position as in Fig. 12 shows. And the difference between the ground truth trajectory and the estimated trajectory is shown in Fig. 13. Particle filtering system can get back to the right estimation from the sensor failure. So the experimental results indicate the potential of our PF for robust localization of mobile robot.

VI. Conclusion

This paper introduced a robust mobile robot localization algorithm, called particle filtering based localization, particle filtering is one of sequential Monte Carlo (SMC) methods. Experimental results demonstrated good performance and robustness of mobile robot localization. In summary, the particle Filter approach is very elegant and very simple to implement. A crucial point is that it does not require any linearization assumptions like EKF and there are no Gaussian assumptions. It is particularly well suited for problems, with small state spaces, the experiments here has a state dimension of three of mobile robot localization.

References

[1] Zhou, H.J. and S. Sakane, Sensor planning for mobile robot localization - A hierarchical approach using a Bayesian network and a particle filter. Ieee Transactions on Robotics, 2008. 24(2): p. 481-487.

[2] Mottaghi, R. and R. Vaughan, An integrated particle filter and potential field method applied to cooperative multi-robot target tracking. Autonomous Robots, 2007. 23(1): p. 19-35.

[3] Li, M.H., B.R. Hong, and R.H. Luo, Mobile robot simultaneous localization and mapping using novel Rao-Blackwellised particle filter. Chinese Journal of Electronics, 2007. 16(1): p. 34-39.

[4] Duan, Z.H., et al., An adaptive particle filter for mobile robot fault diagnosis. Journal of Central South University of Technology, 2006. 13(6): p. 689-693.

[5] Fujimaki, R., T. Yairi, and K. Machida, Localization of intravehicular flying robot using particle filter algorithm. Transactions of the Japan Society for Aeronautical and Space Sciences, 2006. 48(162): p. 199-204.

[6] Yin, B., Z.Q. Wei, and X.D. Zhuang, Robust mobile robot localization using an evolutionary particle filter. Computational Intelligence and Security, Pt 1, Proceedings, 2005. 3801: p. 279-284.

[7] Hu, X.L., T.B. Schon, and L. Ljung, A basic convergence result for particle filtering. Ieee Transactions on Signal Processing, 2008. 56(4): p. 1337-1348.

[8] Brooks, A., A. Makarenko, and B. Uperoft, Gaussian Process Models for Indoor and Outdoor Sensor-Centric Robot Localization. Ieee Transactions on Robotics, 2008. 24(6): p. 1341-1351.

[9] B. Ristic, S. Arulampalam, N. Gordon, *Beyond the Kalman filter*, Artech House Publishers, Boston, 2004

[10]Caballero, F., et al., A probabilistic framework for entire WSN localization using a mobile robot. Robotics and Autonomous Systems, 2008. 56(10): p. 798-806.

[11]S. Maskell, N. Gordon, “A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking,” IEEE Transactions on Signal Processing, Vol.50, pp. 174-188, 2002

[12]Fujun, P., C. Pingyuan, and C. Yangzhou. Adaptive MCMC Particle Filter for Nonlinear and Non-Gaussian State Estimation. in Innovative Computing Information and Control, 2008. ICICIC '08. 3rd International Conference on. 2008.