Efficient particle filter algorithm for ultrasonic sensor based 2D range-only SLAM application

Po Yang

School of Computing, Science & Engineering,
University of Salford, Greater Manchester, United Kingdom
p.yang@salford.ac.uk

Abstract:

Simultaneously localization and mapping (SLAM) is a fundamental topic in robotics and mobile computing communities. Owning to low cost and relatively accurate measurement, ultrasonic sensors are widely used in range only SLAM applications. But these applications are challenging to accurately obtain an initial position given to the SLAM algorithm, and robustly localize the targets in a highly dynamic environment. The traditional approaches to solve these problems are Extended Kalman Filter (EKF) and Particle Filter (PF) algorithms. However, EKF approach suffers from the data association problem and map consistency, PF approach is limited by the highly computational complexity. This paper addresses an improved particle filter algorithm to solve the ultrasonic sensor based 2D range-only simultaneously localization and mapping (SLAM) problem with relatively good accuracy and robustness. A technique called Map Adjustment is proposed to increase the accuracy and efficiency of the algorithm. Using Map Adjustment, the proposed particle filter algorithm can either achieve improved localisation accuracy, or maintain the same accuracy but lower computational complexity. The feasibility and robustness of this algorithm is shown by experiments. The results demonstrate that the proposed algorithm can provide relatively good accuracy and robustness for ultrasonic sensor based 2D range SLAM applications.

1. Introduction:

Simultaneous Localisation and Mapping (SLAM) problem [1-3] refers if it is possible for a mobile robot at an unknown location in an unknown environment, to incrementally build a consistence map of the environment while simultaneously determining its location within this map. In order to know the information about robot environment, the sensor measurements deliver information about the bearing, distance, appearance etc. of nearby features in the environment. There have been various mobile computing systems providing in-door localization, using sensors like: Ultrasonic, Infrared, Laser or Radio frequency. Of the above sensors, due to the low cost and relatively accurate distance measurement, ultrasonic sensors are widely used in various range-only SLAM applications [4-9]. However, there are two major challenges for these ultrasonic localization systems, the first one is that the initial position of the transmitters needs to be known, or should be within a certain range of error; the second one is the strong tolerance ability to errors, which requires the system robust enough in a dynamic environment where many uncertainties might arise.

In order to overcome the above shortcomings, many researchers have attempted different approaches to solve them, which can be mainly classified into two categories, which are hybrid-sensor approaches or advanced localisation algorithms. Hybrid-sensor approaches attempted to hybrid other sensors with ultrasonic sensors to aid the localisation, so that the initial position of the transmitters can be determined, and the position errors can be corrected. Muller [9] developed an indoor localisation system by using a combination of ultrasonic and radio frequency, with one RF transmitter and four ultrasonic transmitters fixed on the ceiling. While this system can provide good accuracies as well as a fairly low cost, the setup of RF transmitters in this system is complicated, and the locations of the transmitters need to be measured manually. Peterllis [5] used infrared patterns to assist the ultrasonic sensors for the estimation of short distances observations. This approach can solve some cases where the distance under test are too short for ultrasonic sensors and the extension of the area covered. But localisation accuracy and system robustness of this approach have not been actually improved. Errington [10] proposed a Least-Squares approach to provide the initial position of the stationary vehicle to the SLAM algorithm, by using an array of RF identification tags placed at known positions. Whilst this approach illustrates the possibility to use RFID to provide relatively accurate and low-cost initial position estimation for SLAM applications, the practically achieved accuracy is low and instable, up to 20 centimetres.

The advanced localisation algorithms attempt to use probabilistic based localisation approaches [14] to overcome the uncertainty of a highly dynamic environment for ultrasonic range-only SLAM applications. Current advanced localisation algorithms mostly rely on a probabilistic framework: Bayesian Filter [15]. Extended Kalman Filter (EKF) [16] and Particle Filter [11] are two common and well-known approaches to the integration and implementation of Bayesian Filter. The major advantages of EKF based SLAM approaches are its capability of providing accurate non-linear estimation in some practical problems and easily implemented. However, EKF based SLAM approaches suffer from Data Association problem [17], which refers that the robot cannot identify each feature practically, especially when the mapping process is complicated. Another problem of EKF based SLAM approaches is the linear approximation of motion and observation model, which would produce the errors affecting the map consistency [18]. Compared to EKF based SLAM approaches, Particle Filter based SLAM approaches [11] have been shown to be more robust. The main strength of particle filter is its ability to solve non-linear problems and its robustness in dynamic environment. These strengths make it particularly suitable for the ultrasonic range-only SLAM applications because the observation model of the ultrasonic sensor is non-linear and not invertible due to the noisy of ultrasonic sensor reading. Another advantage of particle filter is that the initial system states do not need to be known, since the position errors would be converged by continuous sampling data.

However, there are still three major difficulties of applying particle filter into ultrasonic sensor based range-only SLAM applications: the first one comes from the nature of the ultrasonic sensors: the observation model is non-linear and not invertible. The second difficulty is the fact that the motion model of the mobile device or robot is very un-deterministic and without directional information. Great ambiguities arise because of the non-linearity. Finally, the computational complexity of Particle Filter has been a barrier that makes it intractable for SLAM. In localization the state space usually has just 3 or 4 dimensions, while in SLAM the number of features can easily be an order of hundreds. The number of particles needs to rise rapidly with the dimension of the state space, in order to achieve a satisfactory result. Consequently, for the standalone ultrasonic sensor based range-only SLAM applications, the efficient advanced localisation approach with good accuracy and robustness to dynamic environment is still a challenging task.

This paper proposed a particle filter algorithm to solve the ultrasonic sensor based 2D range-only SLAM problem. This algorithm uses the distance based straight observation model and 2D Gaussian based motion model to predict and update the state of localisation system with the capability of reducing the ambiguities at initialisation step. A novel approach called “Map Adjustment” is presented to reduce ambiguities and increase accuracies in this particle filter algorithm. This method exploits a structural property of the SLAM problem by simultaneously maintaining an estimation of the location information and map features in each particle. Hence when given enough sensor reading this approach can effectively estimate the path and the map features. The results demonstrate that the proposed algorithm can provide relatively good accuracy and robustness for ultrasonic sensor based 2D range SLAM applications. The main contributions of this paper are as follow:

(a) . An efficient particle filter algorithm with the distance based straight observation model and 2D Gaussian based motion model is build and implemented to solve the ultrasonic sensor based 2D range-only SLAM problem.

(b) . A technique called Map Adjustment is proposed to increase the accuracy and efficiency of the algorithm. Using Map Adjustment, the proposed particle filter algorithm can either achieve improved localisation accuracy, or maintain the same accuracy but lower computational complexity.

The rest of the paper is organized as follows. Section 2 describes the definition of system state and models. Section 3 present the proposed particle filter based approach, and section 4 shows their experimental validation results. Section 5 gives a summary of the conclusions and future work.

2. System State and Model:

This section provides a comprehensive description of the definition of system states and system models for ultrasonic sensor based 2D range SLAM applications.

2.1 System State:

In terms of the dimensions of 2D range-only SLAM, the state space where this SLAM algorithm operates is two-dimensional. Hence all the features in the map as well as the location of the targeted object can be represented by Cartesian coordinates. Several ultrasonic transmitters are assumed to be mounted around the surrounding of the robot or human carrying a mobile device equipped with an ultrasonic receiver. Each feature of the map actually represents an ultrasonic transmitter, as shown in Fig.1.

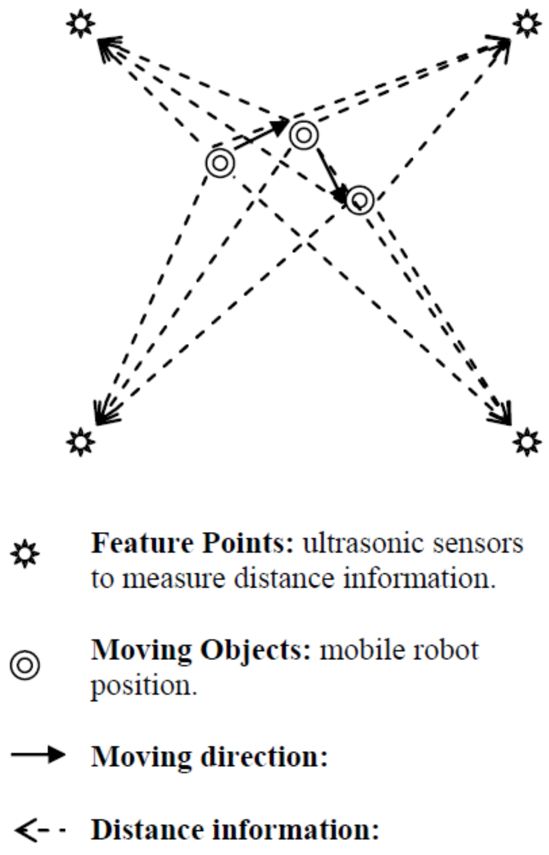
Feature Points: ultrasonic sensors to measure distance information.
Moving Objects: mobile robot position.
Moving direction:
Distance information:

Fig. 1 ultrasonic sensor based 2D range SLAM applications

Each feature of the map actually represents a node of ultrasonic sensor transmitter, which are denoted as f_n, where *n* is an index of ultrasonic sensor transmitters. The location state represents the position of targeted object, is defined as *S*: where *n* is index of transmitters:

$$f_n = \begin{bmatrix} x_f \\ y_f \end{bmatrix}, \qquad s = \begin{bmatrix} x_s \\ y_s \end{bmatrix}, \qquad (1)$$

The system state, at time *t*, is then defined as x_t:

$$x_t = \begin{bmatrix} s_t \\ f_{1,t} \\ f_{2,t} \\ \cdots \\ f_{n,t} \end{bmatrix} \qquad (2)$$

Given the above overview of system state, the targeted object starts moving from an initial position S_0 without prior knowledge of the sensor nodes $f_{1,} f_2 ... f_n$. As the targeted object keeps moving it receives relative range data from the ultrasonic sensor transmitters. By using these sensor data, the particle filter based SLAM algorithm tries to estimate the path $S_0, S_1, ... S_t$ of the targeted object.

2.2 System models:

There are two models that need to be implemented, namely the observation model and motion model. Their specific implementation is characterized by the nature of the ultrasonic sensor system and the motion kinematics. Bayesian filter can be defined as a probabilistic distribution: $\Pr(d_t \mid s_t)$, where d_t, s_t are the targeted object location state and ultrasonic sensor reading over time *t* respectively.

The observation model tells the probability of obtaining a mobile robot position at a certain location state. Unlike most other SLAM problems that use range-bearing sensors, the characteristic of the ultrasonic sensor is that it can only provide relative distance information but not bearing information. Also the distance information contains some noise which is caused by errors of transmitters. Reflection of the ultrasonic on walls or other obstacles will also bring noise. In this paper, it only considers the major error caused from ultrasonic sensor transmitters, the external noise like reflections and obstacles is concerned for future development. The straight observation model is given by the following equation:

$$d_s = g(f_s, s) = \sqrt{(x_f - x_s)^2 + (y_f - y_s)^2} + w \qquad (3)$$

Where: (x_f, y_f) is the coordinate of a feature

(x_s, y_s) is the coordinate of the targeted object

d_n is the relative distance from the targeted object to a feature *n*

w is the Gaussian noise characterizing the errors of the sensors

At each time step, the sensor attached to the targeted object receives observation information from all features.

The motion model characterizes the targeted object location states over time. It helps to predict the next targeted object location state given the most current one. When implementing the motion model, the target mobile object trajectory is associated with direction or speed of the movement that is random. 2D Gaussian model is used to approximate the motion regarding as its ability to cover all possible motion directions. When given the location state S_t at the time step *t*, to predict the location state S_{t+1} at the time *t+1*, a number of particles are randomly distributed from a 2D Gaussian distribution with zero-mean. These particles form a circle with origin at S_t and its radius is determined by the standard deviation of the 2D Gaussian distribution.

3. Algorithm description:

This section would represent the proposed particle filter algorithm in this SLAM solution, which is based on the similar mathematical framework of FastSLAM [18]. The estimation of the system states is factorized into the estimation of the location state and the estimation of the feature states conditioned on the targeted object's path, also the estimation of every feature is independent of each other; the location state estimation is calculated using a particle filter. By exploiting the fact that each feature is conditions on the path, each particle requires maintaining its own estimation of the whole map. Based on above the framework, the data structure of M particles is illustrated in Fig.2:

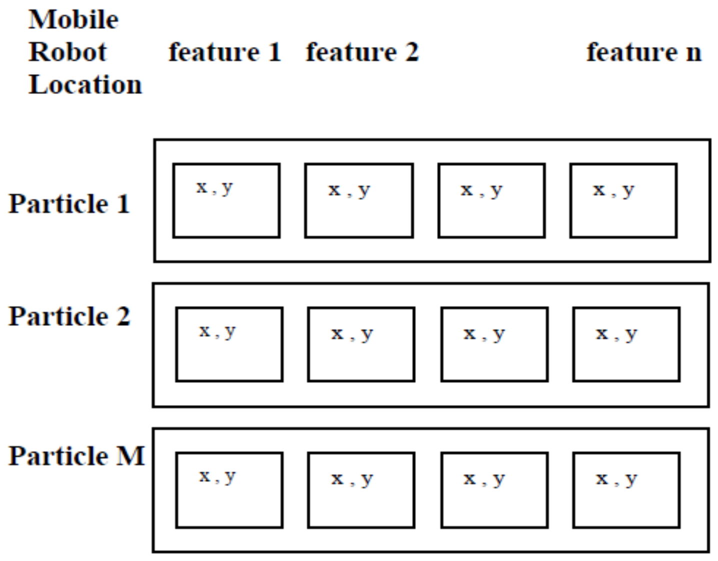

Fig.2 Data structure of Particles

Each particle has *2 (n + 1)* states: 2 location states and *2n* feature states. In a mathematical form, each particle is:

$$\begin{aligned} x_t^m &=< s_t^m, f_{1,t}^m, f_{2,t}^m, \ldots f_{n,t}^m > \\ &=< (x,y)_t^m, (x,y)_{1,t}^m, (x,y)_{2,t}^m, \ldots (x,y)_{n,t}^m > \end{aligned} \qquad (4)$$

Where: *m* is the index of the particle

t indicates the time step

s_t^m is the location of the targeted object (mobile robot)

$f_{n,t}^m$ represents feature *n.*

The particle filter algorithm operates on a set of particles x_t^m. Each iteration of the algorithm can be divided into the following stages: [1] Initialization [2] Apply motion model and apply observation model and weight all the particles [3] Map Adjustment [4] Resampling.

3.1 Initialization:

In EKF-based SLAM, its task is to initialize the mean and covariance matrix for the state vector, while in this particle filter based SLAM it is to initialize the location state and feature states in each particle. The initialization process can be quite difficult when a single measurement is not enough to constrain a feature's location in all dimensions. This problem leads to great ambiguities about the feature states at the beginning of this algorithm. In this paper, an approach is employed to reduce ambiguities by using the first two measurements to obtain a rough idea of where the next location states should be, i.e. in which quadrant the state is. Then a random point is chosen in that quadrant to be the next location state, as shown in Fig.4.

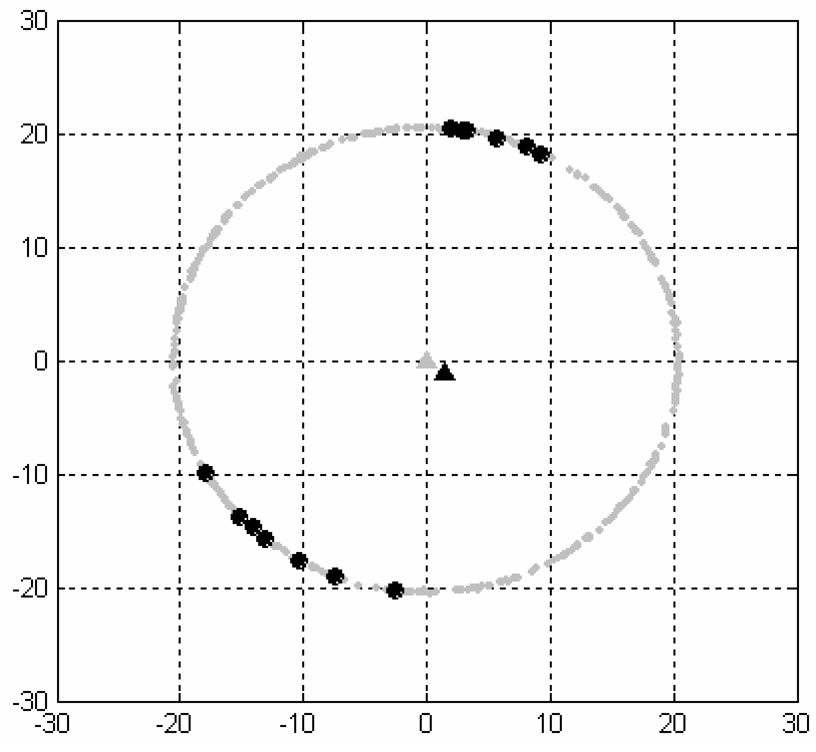

Fig. 3 The initialization process.

In Fig.4, at the beginning (time step 1), the distance measurement (here it is 20cm) of feature A is received. Hence we put feature states of all particles on a circle (the grey points) with a radius of 20 to approximate feature A, and put location state of particles on the origin to be the initial location state (the grey triangle). At time step 2 we put a random point to be the next location state (the black triangle) and based on this point to estimate feature A (the black points). The ambiguity about feature A is reduced from a circle to some points.

3.2 Weighting

After the initialization, the motion model is applied to all particles. The location state of each particle will be replaced with a new one generated from the motion model while the feature state of each particle will remain unchanged. Fig.4 illustrates an example showing one particle being applied the motion model. Before applying the motion model, the particle has an estimation of the mobile robot location state at (x_s, y_s) and estimation of Feature 1 at (x_{f1}, y_{f1}). After applying the motion model the location state is replaced with $(x_s, y_s)'$ while the estimation to Feature 1 remains unchanged. Only applying the motion model to all particles does not represent the true posterior of the path and features since it does not incorporate the observation. Therefore the weighting process is required which gives individual particle a weight to reflect the observation. Before describing how to implement the weighting process, we need to define some terms: At time step *t*, before receiving the observation, each particle has its estimation to the location state and feature states. Then it defines 'predicted location state' as the location state after being applied the motion model, and defines 'predicted observation' as the distance measurement from the predicted location state to a feature.

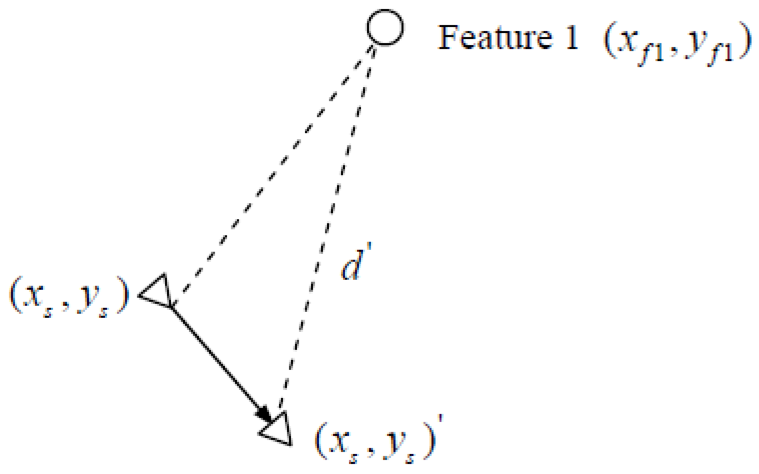

Fig.4 The weighting process.

Regarding as Fig.4, (x_s, y_s) is the predicted location state and *d* is the predicted observation. Then the weight of each particle should be determined by the difference of the predicted observation and real observation. If the predicted location state $(x_s, y_s)'$ and feature state (x_{f1}, y_{f1}) is very close to the real states. Then the predicted observation *d* will be very close to the real observation. Hence this particle will have a high weight. In a probabilistic math form, the weight of each particle is given by:

$$w^m = \int Pr(d_t \mid f_n, s_t^m)\Pr(f_n \mid s_{0:t-1}^m, d_{0:t-1})df_n \qquad (5)$$

Where: *m* is the index of the particle,

t is time step,

f_n is *feature n*,

d_t is the observation.

Equation 5 is implemented to calculate the real observation d_t under a Gaussian with mean $d_t^{'}$ and standard deviation σ determined by the observation noise. The weight of each particle is calculated using the following equation:

$$w = \int_{allfeatures} (2\pi\sigma^2)^{1/2} e^{-\frac{(d-d^{'})^2}{2\sigma^2}} \tag{6}$$

3.3 Map Adjustment

Map Adjustment is novel techniques invented in this paper. The basic idea of Map Adjustment is: For each particle, after applying the motion model and weighting, when the observation is received, each feature's state is then adjusted so that the difference between the predicted observation and real observation becomes smaller. Fig.5 shows one particle example of the Map Adjustment:

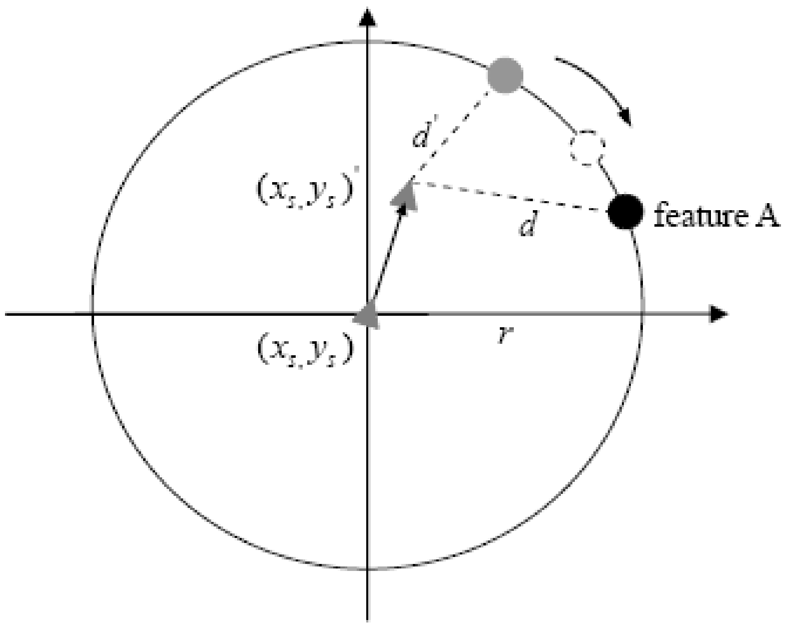

Fig. 5 Illustration of the Map Adjustment

At the beginning, a distance measurement of feature A is received hence we put its estimation (the grey circle) on a circle (with a radius of the distance r). Then the motion model is applied which moves the location state from (x_s, y_s) to $(x_s, y_s)'$ (the grey triangle). If the black circle is the real location of feature A, then a new observation d will be received. Then we compare the real observation d with the predicted observation d'. Typically, d is larger then d' so the estimation to feature A is moved to the dashed circle. By doing so the estimation to feature A will be closer to the real one. How far the grey circle should be moved depends on the difference between d' and d, and the radius *r*. In this implementation, the following equation is used to calculate the movement:

$$movement = p * \frac{(d - d')}{r} \tag{7}$$

Where p is a parameter which must be specified manually based on experiments. By using the Map Adjustment, the accuracy of the estimation to features can be greatly improved, or can be maintained but fewer particles are required.

3.4 Resampling

Resampling is the last step in each iteration. This step is very much the same as the one in Particle Filter Localization. In this process, those particles with large weight will be duplicated while those with small weight will be deleted.

The summary of the whole particle filter localization algorithm for ultrasonic sensor-based 2D range-only SLAM program, as shown in Fig.6:

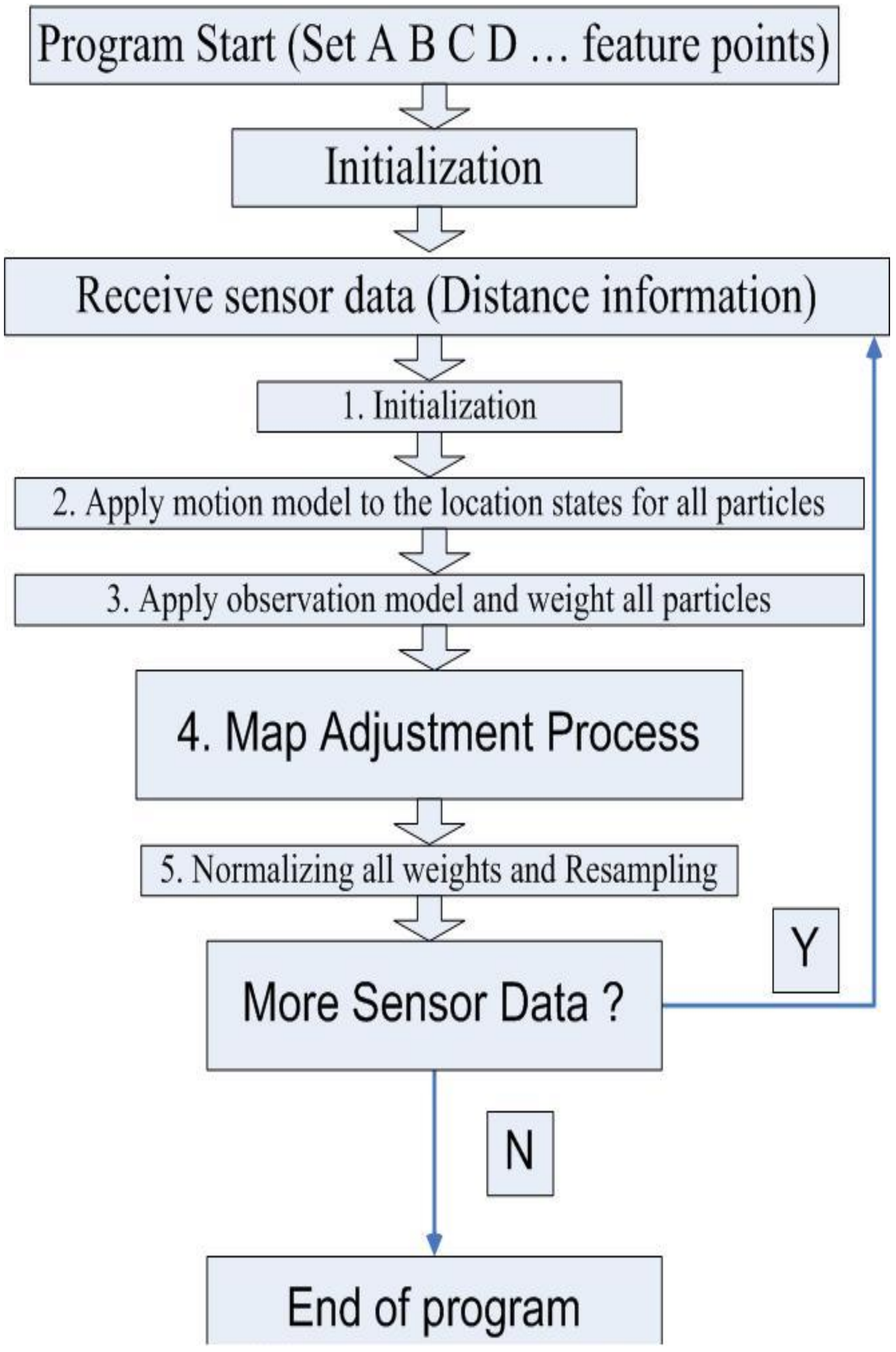
Program Start (Set A B C D … feature points)
Initialization
Receive sensor data (Distance information)
1. Initialization
2. Apply motion model to the location states for all particles
3. Apply observation model and weight all particles
4. Map Adjustment Process
5. Normalizing all weights and Resampling
More Sensor Data ?
Y
N
End of program

Fig. 6 The flow chart of proposed algorithm.

4 Experimental Validation

A number of experiments have been carried out, using different ultrasonic datasets from both practical observation and system simulation, different numbers of particles and other various settings. The goal of these experiments is to evaluate the accuracy, robustness and efficiency of this particle filter based SLAM solution, and to investigate if this algorithm has been successfully implemented for ultrasonic sensor based mobile robot 2D range position estimation and tracking.

4.1 Experiments with Different Ultrasonic Datasets

In this experiment, ultrasonic datasets are observed within some continuous time steps which reflect the sensor noise and the motion kinematics of a human walking at normal speed are used. In particular, in the following datasets, a Gaussian with zero mean and a standard deviation of 0.6 is used as the noise in the observation model. In both of the two datasets there are four simulated *features* in the map with the following *states*: Feature A: (10 dm,10 dm) Feature B: (21.92 dm, -0.65 dm) Feature C: (-11.95 dm, -16.6 dm) Feature D: (-5 dm, 15 dm) The above feature states are called real feature states in the following sections.

In the first dataset, the real path of the mobile robot is tested deliberately to avoid ambiguities and should have some varieties in all directions. The experiment results in Fig. 8 show that: at the beginning the path estimation is not correct, and neither do the feature estimations. This is due to the fact that there are a lot of ambiguities about each feature. For instance, at time step 3, there are several estimations to feature C, which are distributed quite depressively (the grey circles). These ambiguities cause the path estimation to be 'twisted' (the blue line). However as the mobile robot keeps moving, at time step 60, both the feature estimations and path estimation converge to the real ones. Fig.8 (b) shows the errors of the location estimation over time.

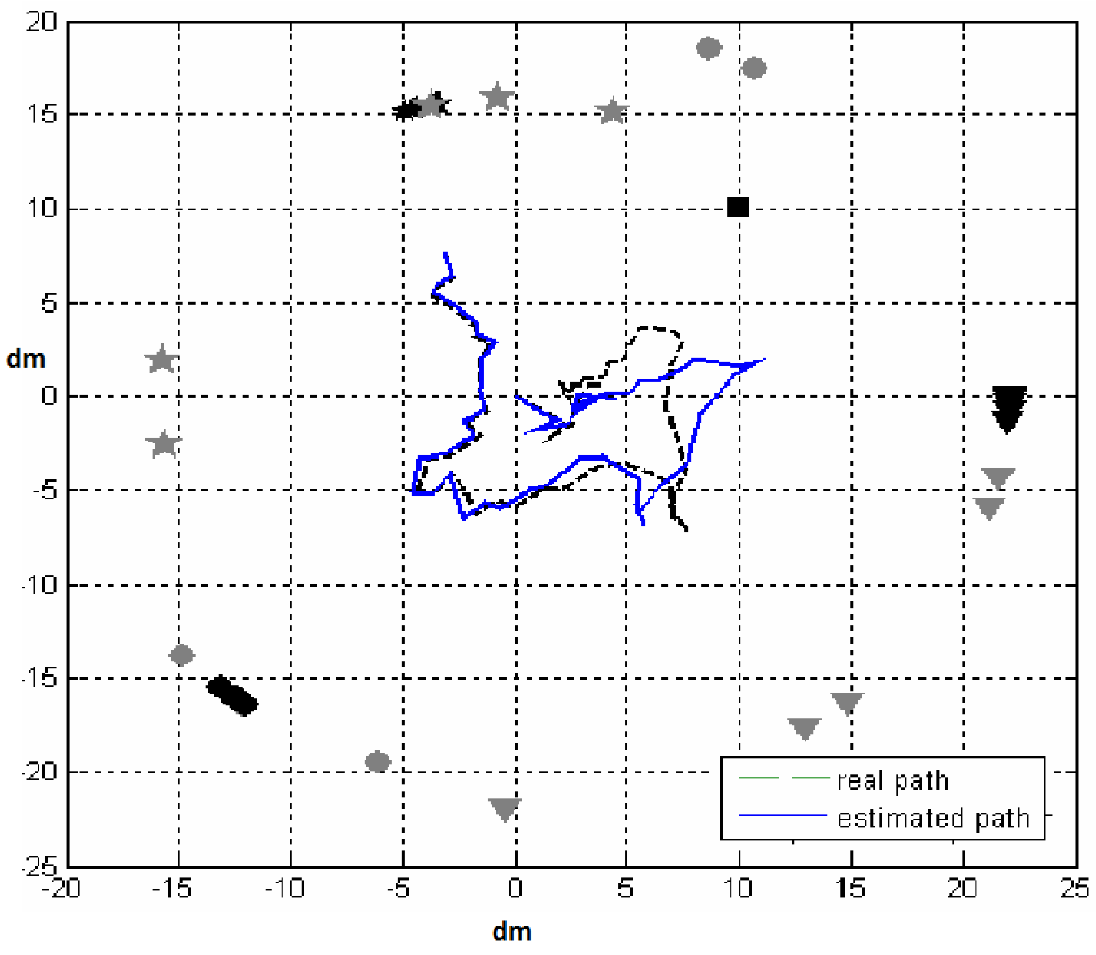

Fig. 8 (a) Regular Dataset Experiment Results

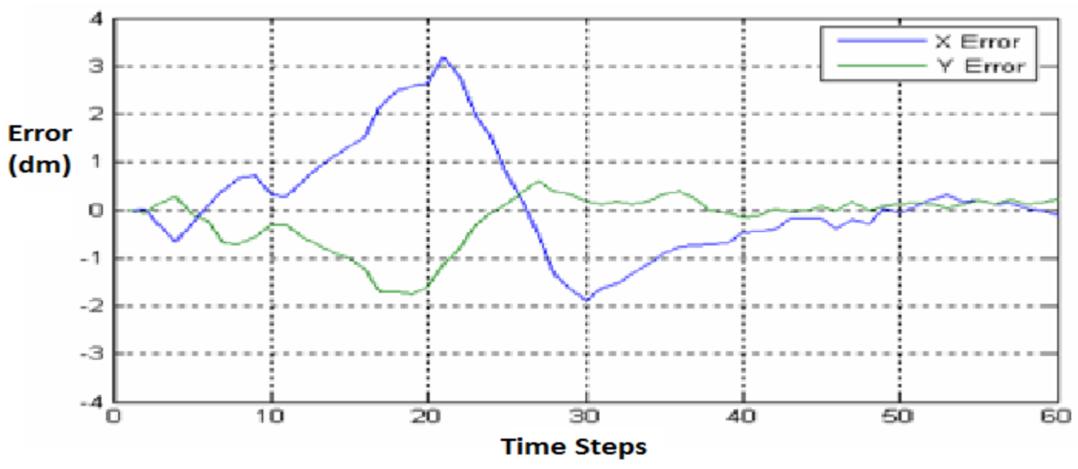

Fig. 8 (b) Errors of the location estimation over time

Fig.8 illustrates the errors of the path estimation from time step 0 to 60. At time step 0 since it assumes that the estimated location and the real location are both at the origin, there are only small errors at the beginning. As the algorithm keeps iterating, a significant error occurs at about time step 20. Compared to Fig.9, it concludes that this is because the mobile robot changes its direction at that time. Nevertheless, as the mobile robot keeps moving, the errors get smaller and smaller and finally converge. Fig.8 (b) also shows that compared with the real feature state, the errors are very little. At time step 60 the estimations to each feature are (calculated using the mean of the corresponding feature estimation of all particles):

Estimation of feature B: (22 dm, -0.074 dm)	real: (21.92 dm,-0.65 dm)
Estimation of feature C: (-12.25 dm, -16.23 dm)	real: (-11.95 dm,-16.6 dm)
Estimation of feature D: (-4.336 dm, 15.29 dm)	real: (-5 dm, 15 dm)

Compared with the real feature state the errors are very little.

In the second experiment, a dataset observed with longer time steps (120 time steps) is used, to test the stability of this algorithm. Result is shown in Fig 10 path estimation with long time steps. Fig 9 illustrates the estimation error on path over time. From Fig.9, the limitation of X and Y axis are separately (-20 dm, 10 dm) and (-10 dm, 20 dm), thus we would get the max X and Y error on unit percentage is about 1/30.

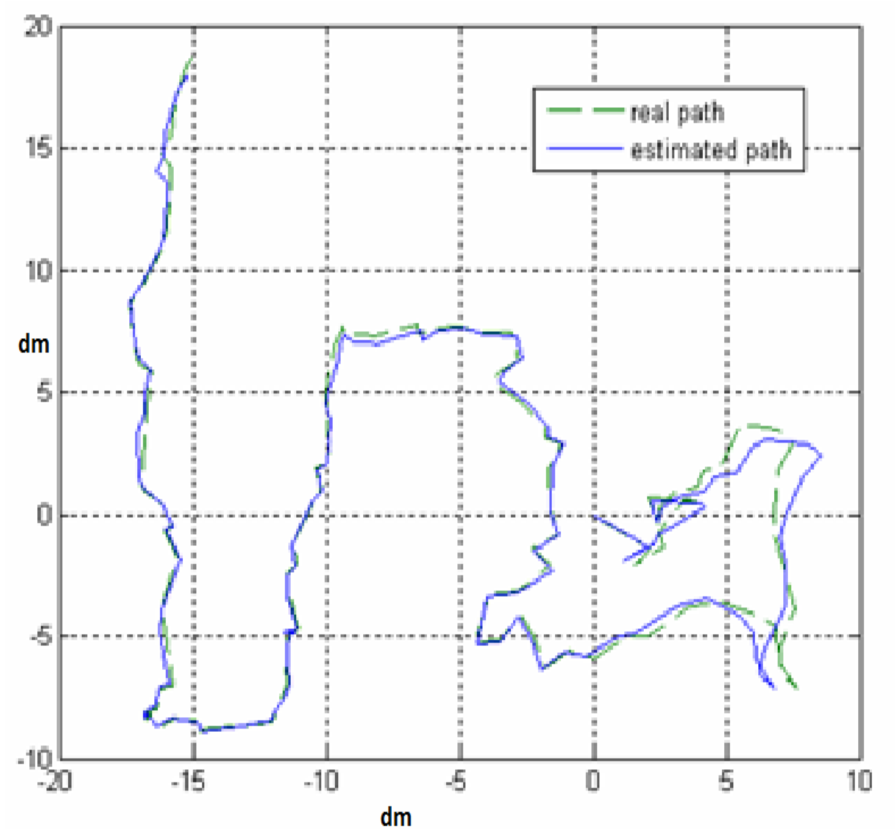

Fig.9 (a) Path estimation with long time steps

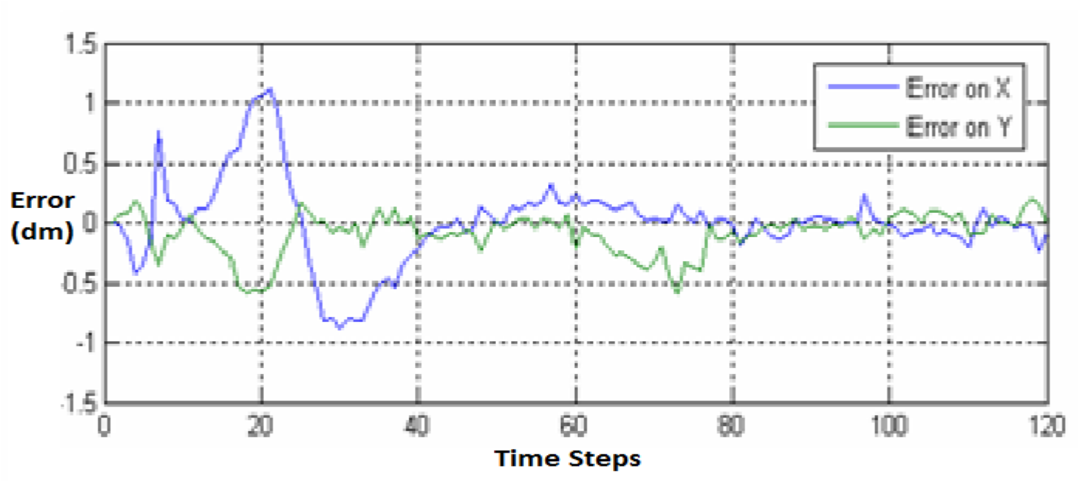

Fig. 9 (b) Error on path estimation over time

Fig.9 shows that after time step 40, the error converges to a stable level and remains relatively small.

4.2 Map Adjustment Improvement the accuracy

Map Adjustment technique proposed in this paper can help to improve the accuracy, or can maintain the same accuracy but fewer particles are required. Fig 10 illustrates a comparison of two experiments using 200 particles. The Figure 10 (a) is the error of the path estimation over time without the Map Adjustment, while the Figure 10 (b) is with Map Adjustment. Clearly after applied the Map Adjustment the estimated path converges more quickly to the real path. The reason is that the map adjustment gives a limitation on the predicted particle estimation, and shortens the time of mobile robot tracking from unstable state to stable state.

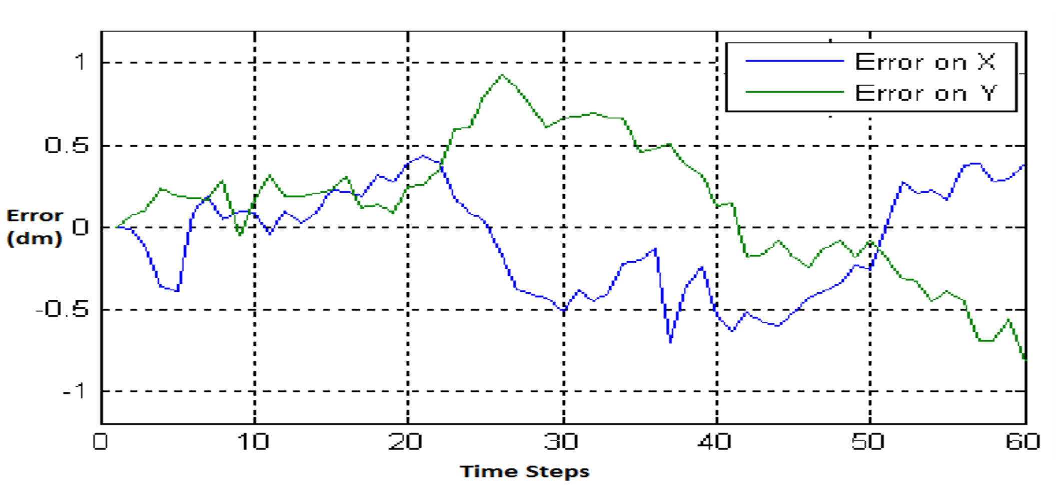

Fig. 10(a) Algorithm without Map Adjustment

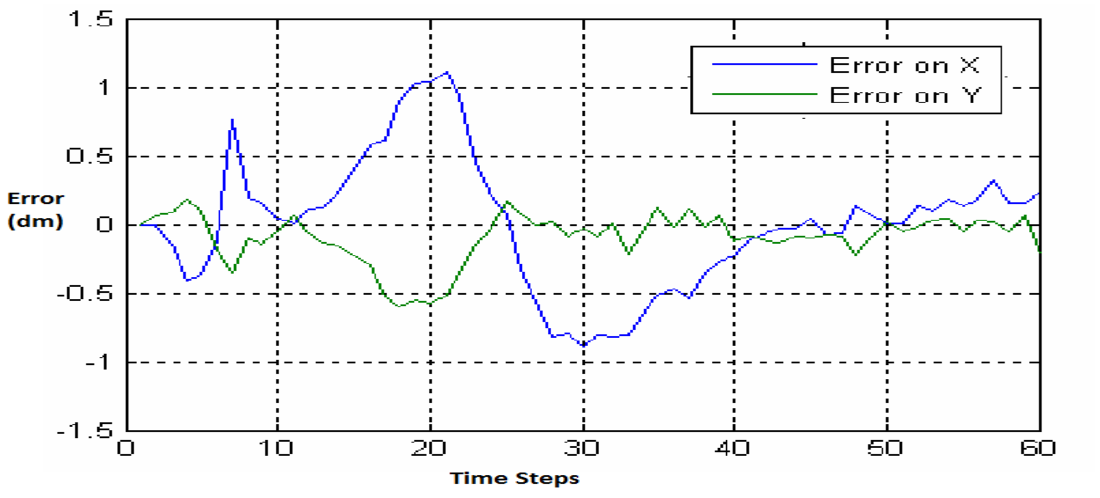

Fig. 10 (b) Algorithm with Map Adjustment

4.3 Performances with different Number of Particles

The number of particles used in this algorithm will significantly affect its performance. Few experiments are carried out using 100, 200, 300, 400, 800 and 1000 particles, respectively. Fig.11 and 12 show the performance of this algorithm in difference number of particles. (CUP 2.4 GHz, RAM 1GB). The platform of algorithm running is the common experiment environment on windows XP and Visual C++. Based on the above experiments, it appears see that the time this algorithm takes is linear to the number to particles, and is also linear to the number of *feature*s in the map. However, as the number of particles reach about 400, the position error would be not improved and remain a stable level.

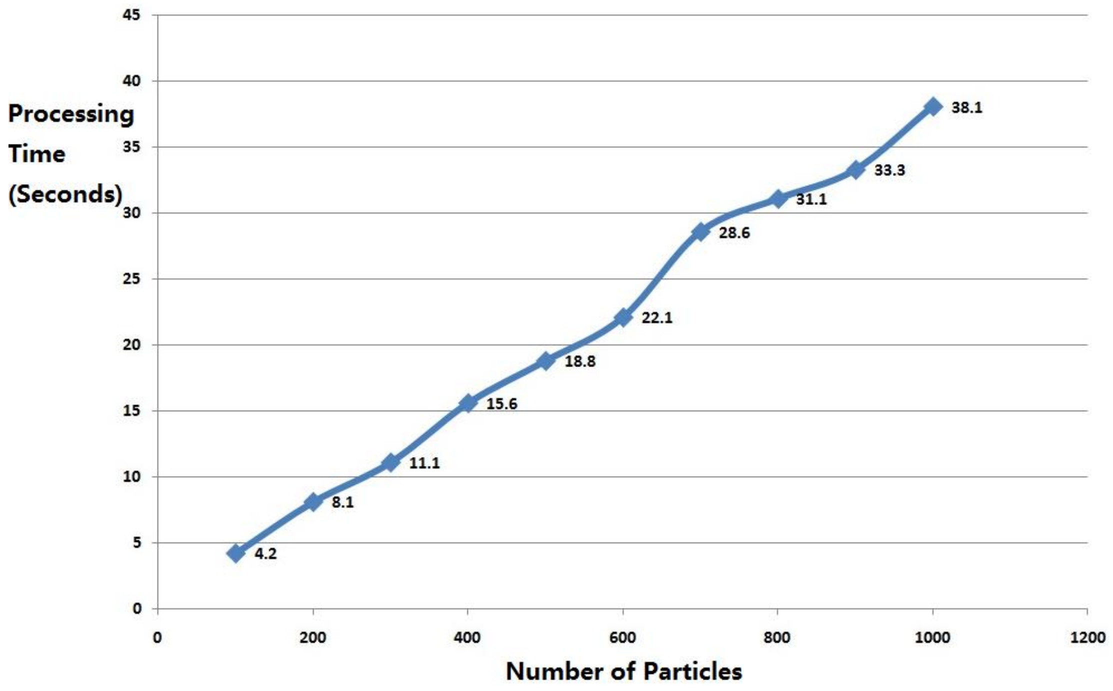

Fig.11 Processing time with increasing number of particles

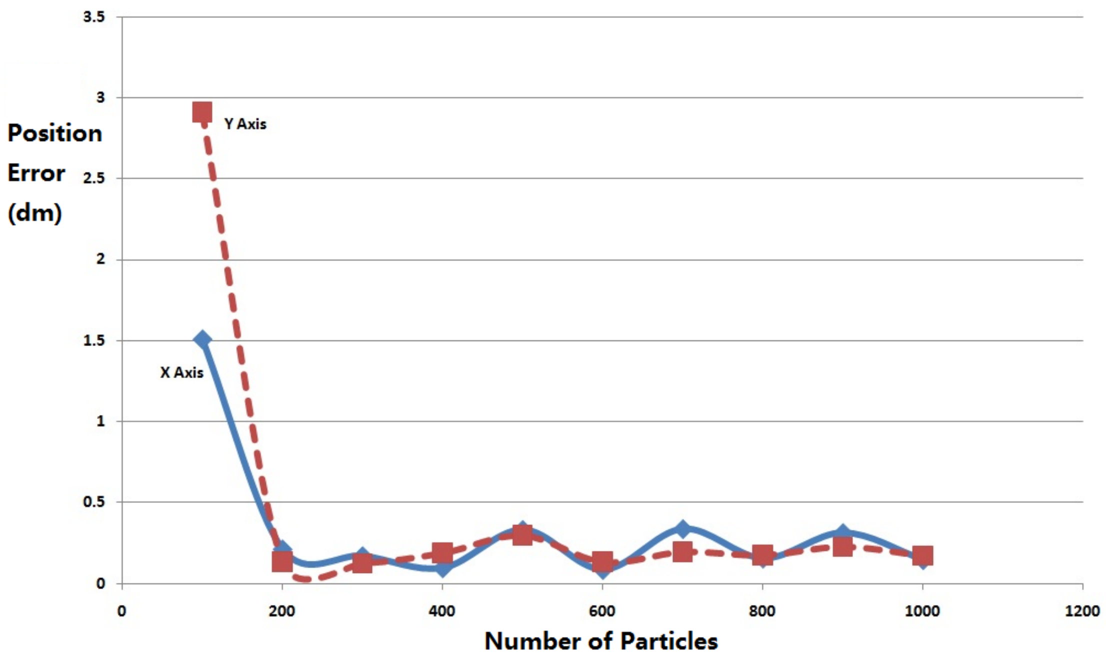

Fig.12 Position error with increasing number of particles.

Base on results from the above experiments, it can conclude that:

1. The real path and the number of time steps will affect the robustness of this algorithm. Adding more varieties along the path, and increasing the time steps will be helpful in terms of accuracy. In some circumstances where the path is too symmetric and the number of time steps is too small the algorithm may fail.

2. The estimated path and map are ‘relative’. There is no fixed orientation of the estimated map and path, i.e., their orientation is determined by how the first observed feature is initialized.

3. The number of particles has a great impact on the performance. More particles will bring better accuracy but worse efficiency. Whereas using fewer particles can improve efficiency but then this algorithm may not be able to obtain accurate estimation. Overall using 400 particles is fairly enough to achieve the balance between spend and accuracy in the above experiments.

5. Conclusion and Future Work

SLAM has been a fundamental topic during the last decade as it allows the deployment of mobile systems within an unknown or partially unknown environment. Ultrasonic based SLAM application yields great potentials in various fields like robotic navigation, mapping, and mobile computing localization.In this paper; an efficient particle filter algorithm has been designed and implemented to solve the problem of ultrasonic sensor based 2D range only SLAM for mobile robot tracking. The particle filter approach can improve the efficiency by factoring the high-dimensional SLAM problem into a product of several low-dimensional estimation problems. Thus the high-dimensional SLAM problem is possible to be solved using particle filter. The experiment results show that the algorithm would achieve the good accuracy and robustness to dynamic environment, with the capable of dealing with noisy observations on ultrasonic sensors. The strengths and limitations that arise throughout the progress of this work will lead to the following issues that warrant future research. Firstly, the current implementation of the motion model only makes use of previous location state and totally ignores the past states. Possible improvements can be made to consider the historical location states so that we can obtain some directional information. Secondly, it would be interesting if this algorithm can be extended to solve a 3D range mobile robot tracking. The estimation of orientation of mobile robot would be considered in further research.

References

1. Andrade-Cetto, J. and Sanfeliu, A. : ‘Environment Learning for Indoor Mobile Robots: Stochastic State Estimation Approach to Simultaneous Localization and Map Building’. Berlin, Germany: Springer-Verlag, 2006.

2. Durrant-Whyte, H. and Bailey, T.: ‘Simultaneous localisation and mapping (SLAM): Part I: The essential algorithms’, IEEE Robot. Autom. Mag., vol. 13, no. 2, pp. 99–110, Jun. 2006.
3. Bailey, T. and Durrant-Whyte, H.: ‘Simultaneous localisation and mapping (SLAM): Part II: State of the art’, IEEE Robot. Autom. Mag., vol. 13, no. 3, pp. 108–117, Sep. 2006.
4. Lin, H.H., Tsai, C.C., and Hsu, J.C.: ‘Ultrasonic localization and pose tracking of an autonomous mobile robot via fuzzy adaptive extended information filtering’, IEEE Trans. Instrum. Meas., vol. 57, no. 9, pp. 2024–2034, Sep. 2008.
5. Petrellis, N., Konofaos, N., and Alexiou, G. P. : ‘Target localization utilizing the success rate in infrared pattern recognition’, IEEE Sensors J., vol. 6, no. 5, pp. 203–210, Oct. 2006.
6. Bank, D.: ‘A novel ultrasonic sensing system for autonomous mobile system’, IEEE Sensors J., vol. 2, no. 6, pp. 597–605, Dec. 2002.
7. Krammer, P. and Schweinzer, H.: ‘Localization of object edges in arbitrary spatial positions based on ultrasonic data’, IEEE Sensors J., vol. 6, no. 1, pp. 203–210, Feb. 2006.
8. Ashokaraj, I. A. R. P., Silson, M. G., Tsourdos, A., and White, B. A,: ‘Robust sensor-based navigation for mobile robots’, IEEE Trans. Instrum. Meas, 58(3): pp551–556, 2009.
9. Randell, C., and Muller, H.: ‘Low cost indoor positioning system’’ In G. D. Abowd, editor, Ubicomp 2001: Ubiquitou sComputing, pp 42–48. Springer-Verlag, 2001.
10. Errington, A. F. C., Daku, B. L. F. and Prugger, A. F.: ‘Initial Position Estimation Using RFID Tags: A Least-Squares Approach’, IEEE Trans. Instrum. Meas, 59(11): pp2863–2871, 2010.
11. Doucet, A.: ‘On sequential simulation-based methods for Bayesian filtering’. Technical report, Signal Processing Group, Departement of Engeneering, University of Cambridge, 1998.

12. Williams,S.B., Newman,P., Dissanayake, G., and Durrant-Whyte, H.: ‘Autonomous underwater simultaneous localisation and map building’, in Proc. IEEE Int. Conf. Robotics and Automation, San Francisco, CA, 24-28 Apr. 2000, pp. 1793–1798.
13. Brooks, R. A.: ‘A robot that walks: Emergent behavior from a carefully evolved network’, IEEE Journal of Robotics. and Automation, 2: pp. 253–262, 1989.
14. Thrun, S., Fox, D., Burgard, D. and Dellaert, F.: ‘Robust monte carlo localization for mobile robots’, Artificial Intelligence, vol. 128, no. 1-2, pp. 99–141. 2001.
15. Degroot, M. H.,and Schervish, M.J.: ‘Probability and statistics’, 3rd ed: Addison-Wesley. 2002.
16. Huang, S.D., and Dissanayake, G., ‘Convergence and Consistency Analysis for Extended Kalman Filter based SLAM,’ IEEE Transaction on Robot., vol. 23, no. 5, pp. 1036–11044, Oct. 2007.
17. Castellanos, J.A., Neira, J. and Tard´os, J.D.: ‘Limits to the consistency of EKF-based SLAM’. in 5th IFAC Symposium on Intelligent Autonomous Vehicles, Lisbon, Portugal, 5-7 Jul. 2004.
18. Thrun, S., Montemerl, M., Koller, D., Wegbreit, B., Nieto, J., and Nebot, E., ‘FastSLAM: An efficient solution to the simultaneous localization and mapping problem with unknown data association’, J. Machine Learning Research, 2004,
19. Dellaert,F., Thrun, S., Burgard, W., and Fox, D.: ‘Monte Carlo localization for mobile robots,’ in Proc. IEEE Int. Conf. Robotics and Automation (ICRA’99), Detroit, MI, May 1999, pp. 1322–1328.