Proceeding of the 11th World Congress on Intelligent Control and Automation
Shenyang, China, June 29 - July 4 2014

Mobile Robot Localization Based on Particle Filter

Fengbing Luo and Bianjing Du

SIASUN Robot & Automation Co., Ltd.

Jinhui Street No. 16, Hunnan District, Shenyang City, Liaoning Province, China

luofengbing@siasun.com, adu85@126.com

Zhen Fan

Shenyang Institute of Automation (SIA), Chinese Academy of Sciences

No.114 Nanta Street, Shenhe District, Shenyang City, Liaoning Province, China

fanzhen@sia.cn

***Abstract* -This paper proposes a self-localization algorithm for mobile robot based on particle filter algorithm. It uses the Monte Carlo method to solve the integral operation of the Bayesian estimation. In order to make the self-localization algorithm real-time, the sequence of importance sampling (SIS) method is introduced. Considering the actual environment, the grid map modal is created. In the inspection process of the robot, Environment map is updated by the Monte Carlo algorithm. This paper designs probability motion modal, detection modal and observation modal of robot, and make a simulation test. The results show that when the robot is on patrol, it can know its position and update the environment map in real time.**

Index Terms - mobile robot, self-localization, particle filter, Monte Carlo method.

I. INTRODUCTION

Location is the most basic aspects for mobile robot navigation. Depending on the complex work environment and various robot sensors, there are many different methods for mobile robot location [1]. In this paper, the robot is equipped with Inertia Navigation System and a laser range sensor. However, uncertainty, a very important issue in robot technology, makes the location unbelievable. The robot environment is high dynamics and uncertainty; the sensors of the robot have limitations and noise; the motors of the robot also have wear and noise. When the robot that lacking accurate information to do task, uncertainty will be generated.

With a large number of robots walking from industrial sites into the unstructured environment, whether to overcome these uncertainties is very important for the successful manufacture of robots. Now, a new approach to resolve the uncertainty of robot perception and action, probabilistic robotics [2-4], is proposed and attracted more and more attention. The main idea of probabilistic robotics is that using probabilistic to definitely describe uncertainty. Different from the past algorithms most likely depending on the event, probabilistic uses all possible scenarios to describe the probability distribution of the information .

To eliminate the interference and obtain a desired result, there have two methods, Gaussian filter and non-parametric filter. The Gaussian filter contains Kalman Filter, Extended Kalman Filter, Unscented Kalman Filter [2], which express the posterior probability of the state space by a Gaussian distribution, and update the mean and variance to calculate confidence and achieve a position estimate; Non-parametric filters do not rely on a fixed functional form of the posterior[5,6]. Instead, they approximate posteriors by a finite number of values, each roughly corresponding to a region in state space. The quality of the approximation depends on the number of parameters used to represent the posterior. As the number of parameters goes to infinity, non-parametric techniques tend to converge uniformly to the correct posterior. The non-parametric filters are well-suited to represent complex multimodal beliefs. When a robot has to cope with phases of global uncertainty, and when it faces hard data association problems that yield separate, distinct hypotheses [1,7,8]. In this paper, the mobile robot runs in the uncertain environment, we used a novel non-parametric filter method, particle filter, to solve mobile robot self-localization problem.

II. PARTICLE FILTER ALGORITHM

The particle filter is an alternative nonparametric implementation of the Bayesian filter. It uses the Monte Carlo method to solve the integral operation of the Bayesian estimation. Particle filters approximate the posterior by a finite number of parameters. The key idea of the particle filter is to represent the posterior by a set of random state samples drawn from this posterior.

A. Bayesian Estimation

The essence of Bayesian filter principle is to try to use all known information to construct the posterior probability density of the state variable [3]. Namely, use transition model to predict the prior probability density of the system state and then the most recent observation is used to correct the posterior probability density.

We assumed that the initial value of the probability density is known.

$$p(x_0 \mid z_0) = p(x_0)$$

Then its prior probability density formula shows as follow.

$$p(x_t \mid z_{1:t-1}) = \int p(x_t \mid x_{t-1}) p(x_{t-1} \mid z_{1:t-1}) dx_{t-1} \quad (1)$$

After obtaining the observed value z_t at time t, the posterior probability is derived by Bayesian formula.

$$p(x_t \mid z_{1:t}) = \frac{p(z_t \mid x_t) p(x_t \mid z_{1:t-1})}{p(z_t \mid z_{1:t-1})} \quad (2)$$

We can get a recursive method to seek the posterior probability from formula (1) and (2), but only a theoretical approach. In fact, the integration in formula (1) is difficult to achieve, then the precise analysis is impossible too. Now the multiple integrals Monte Carlo method is proposed to solve high-dimensional integration problems. The method is suitable for solving high-dimensional integration problems. The Monte

Carlo method is introduced to the sub-sequence of Bayesian estimation, to form a sequential Monte Carlo method [1], also called the particle filter theory.

B. Particle Filtering Principle

Monte Carlo (MC) integration possesses integral value as a mathematical expectation of a random variable, estimates such value by sampling methods. We can get the integration value of an arbitrary function by formula (3).

$$E(g(x_{0:t})) = \int g(x_{0:t})p(x_{0:t} \mid z_{1:t})dx_{0:t} \tag{3}$$

Then we can use $\overline{E(g(x_{0:t}))}$ to approximate $E(g(x_{0t}))$.

$$\overline{E(g(x_{0:t}))} = \frac{1}{N}\sum_{i=1}^{N} g(x_{0:t}^{i}) \tag{4}$$

The discrete samples $\{x_{0:t}^{i}, i = 0...N\}$ are independent and identically distributed sequence from the posterior probability distribution in the generated function. It is proved that when N is big enough, $\overline{E(g(x_{0:t}))}$ is absolute convergence to $E(g(x_{0t}))$.

According to the principle of importance sampling, the importance function $q(x_{0t} \mid z_{1t})$ is introduced to approximate $p(x_{0t} \mid z_{1t})$ by weighting the sampling points that are reference distribution. To offset this difference between p and q, particles $x_{0:t}$ are weighted by the quotient

$$w_t(x_{0:t}) = \frac{p(z_{1:t} \mid x_{0:t})p(x_{0:t})}{q(x_{0:t} \mid z_{1:t})} \tag{5}$$

Then the formula (3) is now carried out with

$$E(g(x_{0:t})) = \int \frac{g(x_{0:t})w_t(x_{0:t})q(x_{0:t} \mid z_{1:t})}{\int w_t(x_{0:t})q(x_{0:t} \mid z_{1:t})dx_{0:t}} dx_{0:t} \tag{6}$$

In order to make the application of real-time, we need to calculate the weights recursively. The sequence importance sampling (SIS) method is introduced [4,5]. By Bayesian formula, we get

$$q(x_{0:t} \mid z_{1:t}) = q(x_t \mid x_{0:t-1}, z_{1:t})q(x_{0:t-1} \mid z_{1:t-1}) \tag{7}$$

Combining Equation (5),

$$w_t(x_{0:t}) = \frac{p(z_{1:t} \mid x_{0:t})p(x_{0:t})}{q(x_t \mid x_{0:t-1}, z_{1:t})q(x_{0:t-1} \mid z_{1:t-1})} \tag{8}$$

Then

$$w_t(x_{0:t}) = w_{t-1}(x_{0:t-1})\frac{p(z_t \mid x_t)p(x_t \mid x_{t-1})}{q(x_t \mid z_{0:t-1}, z_{1:t})} \tag{9}$$

From the sampling of the importance function $q(x_t \mid z_{0:t-1}, z_{1:t})$, and through state equation and observation equation we can calculate $p(x_t \mid x_{t-1})$ and $p(z_{1:t} \mid x_t)$. The task of Sequential importance sampling is to generate the initial sample set, and to calculate weights recursively according to the formula (9).

Another fundamental problem existed in SIS algorithm is degeneracy problem, that after a few iterations recurrence, many particle weights become very small, only a few particles have large weight, and a lot of computing weights are wasted in the small particles. Reference [9] has pointed out that the variance of the weight increase over time, so the degradation can not be avoided.

Resampling principle is proposed to solve the degradation problem [1]. The basic idea is to generate new set of points by re-sampling the posterior probability density M times, then retaining or copying particles with larger weights, excluding particles with smaller weights.

III. POSITIONING ALGORITHM BASED ON PARTICLE FILTER

A. The Models of the Particle Filter Self-Positioning

The models of the particle filter self-positioning contain robot motion model, probability map model, sensor model and particle observation model.

1) Robot Probability Motion Model

(x, y) represents the coordinates of robot centroid in the world coordinate system; θ represents the angle between the direction of robot and the x-axis; provided with robot translational speed v, rotation speed ω, two-wheel mobile robot model is

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \cos\theta \\ \sin\theta \\ 0 \end{pmatrix} v + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \omega \tag{10}$$

In reality, robot motion is subject to noise. The actual velocities differ from the commanded ones (or measured ones, if the robot possesses a sensor for measuring velocity).

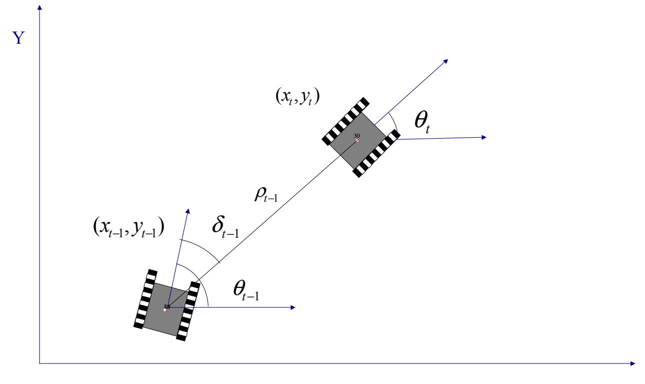


Fig. 1 Model of robot motion

We will model this difference by a zero-centered random variable with finite variance. More precisely, let us assume the actual velocities are given by

$$\begin{aligned} x_t &= x_{t-1} + (\rho_{t-1} + \omega_\rho)\cos(\theta_{t-1} + \omega_{\delta 1} - \delta_{t-1}) \\ y_t &= y_{t-1} + (\rho_{t-1} + \omega_\rho)\sin(\theta_{t-1} + \omega_{\delta 1} - \delta_{t-1}) \\ \theta_t &= \theta_{t-1} + \omega_{\delta 2} - \delta_{t-1} \end{aligned} \tag{11}$$

Here ω_ρ, $\omega_{\delta 1}$, $\omega_{\delta 2}$ are robot translational motion noise, translational motion heading angle noise, rotary motion heading angle noise.

The process of sampling robot probability motion model is shown here.

```
$Algorithm\ sample\_motion\_model(u_t, x_{t-1}^{[m]});$
$\overline{X_t} = f, \omega_\rho = o, \omega_\sigma = 0$
$for$ i=1 to M
    $\omega_\rho = sample\_motion\_distribution(\alpha_1 \rho_{t-1} + \alpha_2 \delta_{t-1} + \psi)$
    $\omega_{\delta 1} = sample\_motion\_distribution(\alpha_3 \rho_{t-1} + \alpha_4 \delta_{t-1} + \psi)$
    $\omega_{\delta 2} = sample\_motion\_distribution(\alpha_5 \rho_{t-1} + \alpha_6 \delta_{t-1} + \psi)$
    $x_t^i = x_{t-1}^i + (\rho_{t-1} + \omega_\rho)\cos(\theta_{t-1} + \omega_{\delta 1} - \delta_{t-1})$
    $y_t^i = y_{t-1}^i + (\rho_{t-1} + \omega_\rho)\sin(\theta_{t-1} + \omega_{\delta 1} - \delta_{t-1})$
    $\theta_t^i = \theta_{t-1} + \omega_{\delta 2} - \delta_{t-1}$
    add$(x_t^i, y_t^i, \theta_t^i)$ to $\overline{X_t}$
$endfor$
return $\overline{X_t}$
```

2) Particle Observation Model

We use map diffusion method to simulate the work of distance sensor in the actual work environment. For each pose particles on, in the direction corresponding to the actual distance of the sensor array, the pixel of that pose expand on the map, until it encounters a pixel that is greater than a certain threshold point. Since the laser range finder has a resolution of three 1°, 0.5°, 0.25°, the information thus measured to be very much, in order to reduce the amount of computation, we tend to extract only a portion of the direction to be calculated, as shown in Figure 2. The picture shows the obtained simulation environment map information by a position and orientation of the particles. The information contains 37 data evenly distributed within the 180° angle range.

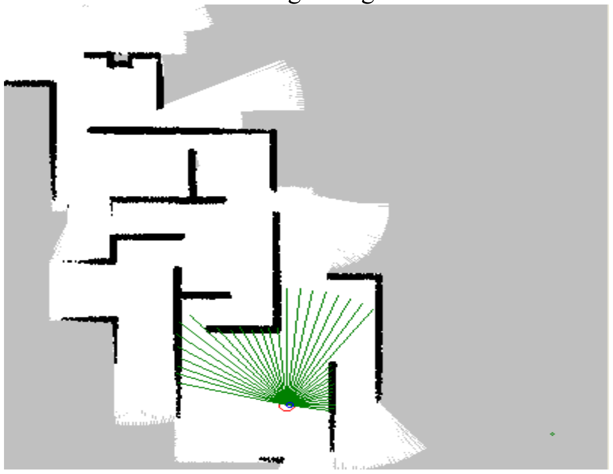


Fig. 2 Model of particle observations

The comparison of similarity between actual observation information and particle observation information is the most important factor in calculating the weight of the particles.

$$\omega_t^{[m]} = \eta p(z_t \mid x_t^{[m]}) \tag{12}$$

Here η is a parameter of normalization. z_t is a pose observed by particles, $p(z_t \mid x_t^{[m]})$ is the observed probability.

In this paper we use the Euclidean distance to express this similarity, namely,

$$D(X_{real}, X_{particle}) = \sqrt{\sum_{i=1}^{M} (X_{real}^i - X_{particle}^i)^2} \tag{13}$$

Here M is the number of columns of distance information from the particles. Then the weight ω_i is adjusted based on the actual simulation results.

$$\omega_i = \frac{1}{D(X_{real}, X_{particle})^4 + 1} \tag{14}$$

Here the range of ω_i is (0 to 1).

B. The Process of Particle Filter Localization Algorithm

The flow of mobile robot localization algorithm based on particle filter is as follow.

```
1.$Input$: $X_{t-1} = \{< x_{t-1}^i, \omega_{t-1}^i > | i = 1,...,M\}$, $u_t, z_t$
2.$\overline{X_t} = X_t = \Phi$; $a = 0$
3.$for\ m = 1\ to\ M\ do$
4.    $x_t^{[m]} = sample\_motion\_model(u_t, x_{t-1}^{[m]})$ //sampling
5.    $\omega_t^{[m]} = p(z_t \mid x_t^m)$ //caculate the weights
6.    $a = a + \omega_t^{[m]}$ //update normalization factor
7.    $\overline{X_t} = \overline{X_t} + < x_t^{[m]}, \omega_t^{[m]} >$; //insert sample
8.$end\ for$
9.$for\ i = 1\ to\ M\ do$ //normalizaton
10.   $\omega_t^{[i]} = \omega_t^{[i]} / a$
11.$end\ for$
12.$for\ m = 1\ to\ M\ do$ //resampling
13.   draw i from $\overline{X_t}$ with probability $\propto \omega_t^{[m]}$
14.   $X_t = X_t + < x_t^{[m]}, \omega_t^{[m]} >$
15.$end\ for$
16.$Output$: $X_t$
```

IV. Map update and create

In the inspection process of the robot, we can get the latest information of the environment. The information can be used to continuously update the known map, and create the latest map of the environment. While the robot completes the inspection and arrives to the pre-set target point, the map is updated and reconstruction.

In this paper, we use grid map to simulate the robot environment. To eliminate the uncertainties caused by many factors, map updated and created by the Monte Carlo algorithm. Through constantly calculating the probability of the grid map, we use the gray scale to show the map. Updating grid map has twofold. First, incremental update the global map by the local map. Second, update the relative positions of partial maps of all time previously observed in the global map according to the desired path generated by the particle filter method.

To simplify the notation, we instead use the p instead of writing $p(S_{i,j} = Occupy)$ to indicate the probability of the grid been occupied. So the first step, global map is incremental updated by formula (15).

$$p_t = p_{t-1} + \alpha(p_v - p_{t-1}) \times |p_v - 0.5| \qquad (15)$$

Here p_t is the probability of grid been occupied at time t. p_v denotes the probability estimated by laser at time t. Then we use Monte Carlo method to approximate the probability p_v.

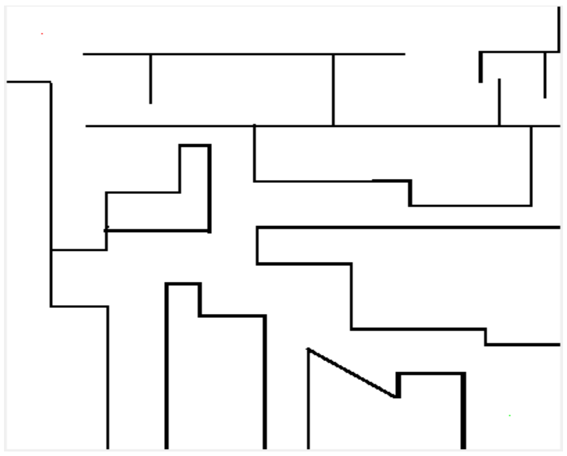
The position of the grid map is represented by $m_{ij}(i \in [0,m], j \in [0,n])$, Laser rangefinder observation data recorded as $z_{0:t} : \{z_0, z_1, z_2, \cdots, z_t\}$, Position and orientation of the robot is denoted $x_{0:t} : \{x_0, x_1, x_2, \cdots, x_t\}$, each time we have L particles $\{x_t^0, x_t^1, x_t^2, \cdots x_t^l\}$, the probability of updated map is $p(m_{ij} \mid x_{0:t}, z_{0:t})$, According to observations z_t, the estimated probability of each grid is $p(m_{ij} \mid x_t, z_t)$. The Monte Carlo method is as follows:

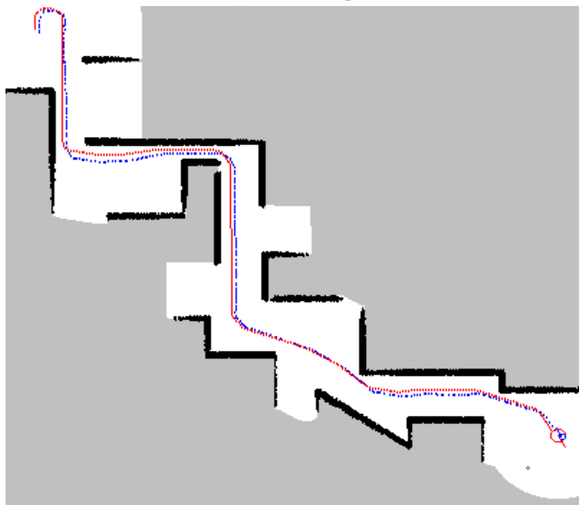
```
$MCM\ (p(m_{ij} \mid x_{0:t}, z_{0:t}), x_{0:t}, z_{0:t})$
  for $k = 1$ to $K$ do
      sample $\xi_k \sim P(x_t)$
      for each grid_cell $m_{ij}$ do
          $p(m_{ij} \mid \xi_k, z_t)$ by laser_probability_model
      endfor
  $p(m_{ij} \mid x_t, z_t) = p(m_{ij} \mid x_t, z_t) + \frac{1}{K} p(m_{ij} \mid \xi_k, z_t)$
  endfor
  $p(m_{ij} \mid x_{0:t}, z_{0:t}) = p(m_{ij} \mid x_{0:t-1}, z_{0:t-1})$
          $+ (\alpha |p(m_{ij} \mid x_t, z_t) - p(m_{ij} \mid x_{0:t-1}, z_{0:t-1})|)$
          $\times (p(m_{ij} \mid x_t, z_t) - 0.5)$
  return $p(m_{ij} \mid x_{0:t}, z_{0:t})$
```

The core idea of the algorithm is that the Monte Carlo method approximates error distribution of the robot pose $(x, y, \theta)^T$, and gets the estimate probability of each grid $p(m_{ij} \mid x_t, z_t)$, then updates probability of each grid in the new global map by probability formula.

V. EXPERIMENTS AND RESULTS

In simulation environment, experiment of this method is made in maze map. This experiment proves the effectiveness and robustness of particle filter self-localization for global positioning.


(a) Actual map


(b) Robot location

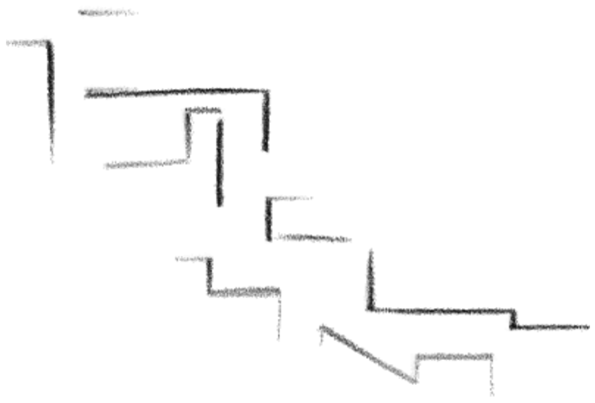

(c) Created map

Fig. 3 Simulate of robot location and map creating

In Figure 3, Fig 3.a is the actual map. In the Fig 3.b, the gray area means the area that no sensor has detected, and area of these places will not be updated. The red line indicates robot actual track, and the blue one is a track estimated by particle filter. Fig 3.c shows the map created by robot after a route running along a grid map. The walls and obstacles in the figure are not particularly clear, this is because the probability is not very high and caused by uncertain factors.

REFERENCES

[1] Sebastian Thrun, Dieter Fox, Wolfram Burgard. Probabilistic Robotics. Cambridge, MA: The MIT Press. 2005

[2] Cody Kwork, Dieter Fox, Marina Meila. Adaptive Real-time Particle Filters for Robot Localization. Proc. of the IEEE International Conference on Robotics and Automation (ICRA)

[3] N.J. Gordon, D.J. Salmond, C. Ewing. Bayesian state estimation for tracking and guidance using the bootstrap filter. Journal of Guidance, Control and Dynamics, Vol.18, No. 6, 1995, 1434-1443

[4] A. Doucet, S. Godsill, and C. Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. Statist. Computer, Vol.10, 2000, 197-208

[5] A. Doucet, J. de Freitas, N. Gordon. Sequential Monte Carlo Methods in Practice. New-York: Springer-Verlag. 2001

[6] P. Del Moral. Measure valued processes and interacting particle systems. Application to nonlinear filtering problems. Ann. Appl. Probab, 1998

[7] Chatila R, Laumond J-P. Position referencing and consistent world modeling for mobile robots. IEEE international Conference on Robotics and Automation. 1985, 138-145

[8] A. Lambert, D. Gruyer, and G. S. Pierre, A fast Monte Carlo algorithm for collision probability estimation, in Int. Conf. on Control, Automation, Robotics and Vision (ICARV), 2006, pp. 406–411

[9] Simon Godsill, Tim Clapp. Improvement strategies for Monte Carlo particle filters [D]. Signal Processing Group, University of Cambridge, 1998