Proceedings of the 2005 IEEE/ASME
International Conference on Advanced Intelligent Mechatronics
Monterey, California, USA, 24-28 July, 2005

Monte Carlo Multi-Robot Localization Based on Grid Cells and Characteristic Particles

Juncheng Liu[1], Kui Yuan[1], Wei Zou[1] , Qinmin Yang[2]
[1] Institute of Automation, Chinese Academy of Sciences
Beijing, 100080, China
E-mail: {jcliu, kuiyuan, wzou}@hitic.ia.ac.cn
[2] Electrical and Computer Engineering Department, University of Missouri Rolla
Rolla, MO65409, USA
E-mail: qyy74@umr.edu

Abstract—**In this paper, a Monte Carlo method based approach for multi-robot localization is described. In this approach, grid cells are used to describe the whole particle set which is used in MCL method to estimate the pose of robot. Then, the sizes of the grid cells are adjusted to capture the characteristic particles that can represent the property of all particles. The characteristic particles can be used to estimate the robot's position in its operation space. Because the number of the characteristic particles is much less than that of the total particles, this approach can reduce the computing time greatly. Simulation results are also given to show that this approach can obtain good localization performance in multi-robot system.**

Keywords-* *Monte Carlo Localization, Mult-robot Localization, Grid cells, Characteristic particles

I. INTRODUCTION

To be able to move freely in their work environment, mobile robots are required to obtain their position information during the work. Consequently, sensor-based robot localization has been seen as one of the most fundamental problems in mobile robotics [1]. In recent years, a great amount of work on localization of single robot has been reported [2]. However recently, it is critical to realize multi-robots cooperation in many robotic applications and the multi-robot systems are appealing for more attention. Thus, the main problem will be identifying and solving problems of multi-robot localization.

Multi-robot systems have some advantages over the single robot system. In the matter of fact, multi-robot system can collect and integrate multiple sensor information from different robots with different pose [3]. By fusing these multi-sensor data, the system can obtain better localization performance. Firstly, multiple robots can share their sensor information, which will increase the robustness of the localization algorithm for each robot. Secondly, the robots can exchange their pose information with each other, and use their geometric relationship to derive more reference information for localization [4]. Thirdly, as a possible implementation, the sensor architectures on some robots can be changed to lower the cost without influencing the performance of the whole system. In addition, different robots can be equipped with different type of sensors, so that the whole system can achieve more comprehensive environment description.

Recently, a lot of research work focus on integrating the sensor information from multiple sources to estimate the uncertainty of the absolute or relative position of robot [5]. The most commonly used approaches include triangulation, Kalman filter and MCL (Monte Carlo Localization). Roumeliotis and Bekey presented an approach in which the sensor data from a heterogeneous collection of robots are combined by a single Kalman filter to estimate the position of each robot in the team [6]. They also showed how this centralized Kalman filter can be divided into N separated Kalman filters (one for each robot) to allow for distributed processing. Fox etc. applied MCL algorithm, which is usually used on single-robot probabilistic localization, to solve the multi-robot localization problem [7]. In their approach, each robot maintains a probability distribution describing its own pose (based on odometer and environment sensing), and is able to be updated through the data from other robots. The authors used the density tree for each particle set to avoid the curse of dimensionality, and make the MCL in collaborative localization feasible.

In this paper, we present an approach for multi-robot localization based on MCL method. As stated in [7], MCL usually is used in single robot localization, and can not be directly used in collaborative localization because of the curse of dimensionality. Thus, instead of using the density trees used by Fox etc, we use grid cells to partition the whole particles into several areas. A changeable grid cells method is used to get the characteristic particles that represent the whole particles' property on estimating robot's pose. Because the characteristic particles can represent whole particles on calculating the weight of particles, this approach can reduce the computing complexity greatly without influencing the precision of the MCL algorithm.

The rest of this paper is organized as follows. Section 2 presents the mathematical details of the MCL algorithm, in section 3 we present the grid cells based MCL method and characteristic particles based MCL method on multi-robot localization. Simulation and experiments illustrating our application of this method are given in section 4. Finally, some conclusion remarks are given in section 5.

II. MONTE CARLO LOCALIZATION

MCL method is a probabilistic method, in which the current localization is modeled as a set of weighted density particles (or samples). Each particle can be seen as a hypothesis that the robot is located at a certain position and is represented by a robot pose element, which consists of the robot's rotation and x/y-coordinates etc. The weight denotes the possibility that the robot is located at that corresponding position. Meanwhile, MCL method requires both a motion model and an observation model. The former models the change of the probability for certain actions to move the robot to certain positions [9], while the latter describes the probability for certain measurements taken at a certain location.

As a recursive algorithm, the localization approach works as follows for each loop:

Step1: Position changing of all particles according to previous motion. At time t, suppose the change of the robot's position is (ΔX_t , ΔY_t) with uncertainties of ($\Delta P_{Xt}, \Delta P^i_{Yt}$), which can be measured by odometer. Then we can compute every particle's new position by:

$$\begin{cases} X_t^i = X_{t-1}^i + \Delta X_t + \Delta P_{Xt} \times rand \times W_{t-1}^i \\ Y_t^i = Y_{t-1}^i + \Delta Y_t + \Delta P_{Yt} \times rand \times W_{t-1}^i \end{cases} \quad (1)$$

where (X_{t-1}^i, Y_{t-1}^i) is the position of particle *i* at time *t-1*, while (X_t^i, Y_t^i) is the new position of particle *i* obtained from the motion model, *rand* is a random number $\in$ [-1,1], while W_{t-1}^i is the weight of particle *i* at time *t-1*. If we know the robot's initial position, we can use Equation 1 to estimate its position in runtime. However, the uncertainty will accumulate over time. Thus, it is necessary to decrease the amplitude of the uncertainty by fusing sensor data into the estimation.

Step2: Weights computation of particles after acquiring environment information S . The observation model describes the probability of taking a certain measurement S^i at a certain location [10]. Hence, supposing that the robot is located at a given particle's position, we can estimate the sensor data that should obtain by the observation model. And the posterior probability [11] $Q(S,P_i)$ of particles P_i can be obtained by matching S and S^i:

$$Q(S,P_i) = ke^{-q(S-S^i)^2}, \quad (2)$$

where k and q are properly selected parameters.

Since the effect of the particle's previous weight on the posterior probability must be taken into consideration, the calculation of $Q(S,P_i)$ can be adjusted to:

$$Q(S,P_i) = W_i^{t-1} ke^{-q(S-S^i)^2} \quad (3)$$

where W_i^{t-1} is the previous weight of the particle. Therefore, the posterior probability of the particle can be seen as the similarity of the sensor information S and the expected information S^i . Furthermore, MCL requires that the sum of all particles' weights equals 1, so the particle's weight achieved by:

$$W_t^i = Q(S,P_i) / \sum Q(S,P_j) \quad (4)$$

Step3: Update (or resample). In this step, the distribution of particles is updated, based on their weight. The particles are copied from the old distribution to a new distribution. The frequency in the new distribution depends on the weight of each particle, so more probable particles are copied more often than less probable ones, and improbable particles are removed. That is to say, more probable particles can "survive" with more probability than less probable ones, and improbable particles are removed instead. This process can be seen as a probabilistic random search for the best position. Because the particles that are moving closer to the real position of the robot will be rewarded by better posterior probabilities during the updating steps, it will be more possible that they can remain in the distribution in future distributions [12].

Step4: Pose Estimation. The pose of the robot is calculated from the particles distribution in two steps. First, the cluster with most particles is found, and then the current pose is calculated as the average of all particles in that cluster.

Finally, the algorithm will go back to step 1, and by this means, the all particles will move closer and closer to the real position of the robot.

III. MCL MULTI-ROBOT LOCALIZATION BASED ON GRID CELLS AND CHARITISTIC PARTICLES

A MCL Multi-robot Localization Based on Grid Cells

Multi-robot system has multiple advantages over single-robot system, since robots can detect each other to get relative geometric information, and hence obtain more reference data for localization. Furthermore, they can make use of this geometric relationship to fuse all sensor information for localization among the robots. Particularly in MCL algorithm, this fused information can be useful to synchronously update the particles of robots.

We investigate the multi-robot localization problem under the following assumptions:

1. All robots run on a 2D flat platform.

2. All robots are capable of exchanging their sensor data, and the time for communicating is insignificant and can be neglected.

3. Robots are equipped with sensors that can be used to detect each other (such as visual sensors). And when a robot detects another one, the sensors can provide their geometric relationship.

4. Robots are equipped with sensors that can provide their motion data (such as odometer).

5. Robots are equipped with compass that can obtain

their azimuth information in real time.

Based on these assumptions, when *robot1* detects *robot2*, they can achieve their relative distance *d* from sensor readings (as show by Figure 1). According to MCL method, we can use such information to synchronously update the particles of *robot1* and *robot2*. Suppose the position of particle *i* of *robot1* is (x_{1i} , y_{1i}), and the position of particle *j* of *robot2* is (x_{2j} , y_{2j}), then the distance between the two particles is:

$$d_{ij} = \sqrt{(x_{1i} - x_{2j})^2 + (y_{1i} - y_{2j})^2} \qquad (5)$$

Then, both of the posterior probability of particle *i* and *j* can be written as:

$$Q(d, P_{ij}) = ke^{-q(d-d_{ij})^2} \qquad (6)$$

Assume there are N particles in *robot1*'s particles set, and M particles in *robot2*'s set. According to Equation 6, if we want to obtain the posterior probability of particle *i*, Equation 6 should be calculated for particle *i* with every particle in *robot2*, and the maximum posterior probability will be chosen to calculate its weight.

$$Q(d, P_i) = \max(Q(d, P_{ij})) \quad \text{j=1..M} \qquad (7)$$

The same process should be applied on particle *j*. So, if we want to obtain the posterior probabilities of all particles, we need to calculate them for N×M times, which will take a large amount of time. Thus, the MCL method in multi-robot localization can not be implemented straightforwardly.

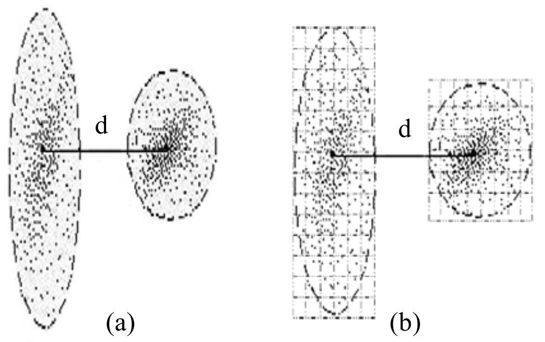

Fig.1 Using grid cells to describe the particle sets

The posterior probability of particle denotes the possibility that the robot is located at a certain position [13][14]. If the particles are very close to each other, the difference of their posterior probabilities will be very small. Considering the sensor reading error, we assume that the posterior probabilities of adjacent particles are equal when the distance between these particles is less than the sensor reading error. So the posterior probabilities of all of the particles adjacent to a certain particle can be obtained by only calculating that particle's posterior probability. Thus, all of the particles can be assigned into a number of grid cells. Assuming that the posterior probability of all of the particles in a grid cell is equal to the posterior probability of that grid cell, the posterior probabilities of particles can be calculated using the defined grid cells.

Because the number of grid cells is much less than that of the particles, the computing efficiency can be improved significantly. The algorithm works as following.

1. Determining the size of grid cell.

Because all of the particles in the same grid cell have the same posterior probability, their estimated position error should be less than the maximum error of sensor reading ΔP. On the other hand, the particles whose positions are very close with each other may be in different grid cells with different posterior probability and different estimated position error. We should also make sure that this kind of error must be less than ΔP. So, we chose the length of the cells *L* as:

$$L = \Delta P / 2\sqrt{2} \qquad (8)$$

2. Disvide all of the particles into grid cells.

Suppose the position of particle *i* belong to *robot1* is (x_{1i} , y_{1i}), particle *i* will locate in grid cell (*x1row*, *y1row*):

$$x1row = floor(x_{1i} - \min(P(x_{1i}))) / L) + 1 \qquad (9)$$

$$y1col = floor(y_{1i} - \min(P(y_{1i}))) / L) + 1 \qquad (10)$$

The number of grid cells *Pnum* is calculated as:

$$Binsx = round((\max(P(x_i)) - \min(P(x_i)))/L) \qquad (11)$$

$$Binsy = round((\max(P(y_i)) - \min(P(y_i)))/L) \qquad (12)$$

$$Pnum = Binsx \times Binsy \qquad (13)$$

where $\max(P(x_i))$ is the maximum *x* of all particles, while $\min(P(x_i))$ is the minimum position of all particles on *x* axis. Similarly, $\max(P(y_i))$ and $\min(P(y_i))$ corresponds to the maximum and minimum value on *y* axis.

3. Calculating the posterior probabilities of grid cells.

When *robot1* detected *robot2*, they can get their relative distance *d* from sensor readings. Meanwhile, suppose the center position of cell *i* of robot1's particle sets is (x_{1i} , y_{1i}), while the center position of cell *j* of robot2's particle set is (x_{2j} , y_{2j}), then the distance between the two cells is

$$d_{ij} = \sqrt{(x_{1i} - x_{2j})^2 + (y_{1i} - y_{2j})^2} \qquad (14)$$

Obviously, both of the posterior probability of cell *i* and *j* are:

$$Q(d, P_{ij}) = ke^{-q(d-d_{ij})^2} \qquad (15)$$

Suppose that there are *Pnum1* cells in *robot1*'s particles set, and *Pnum2* cells in *robot2*'s set. According to Equation 14, if we want to find the posterior probability of cell *i*, Equation 15 should be calculated for cell *i* with every cells in *robot2*, and the maximum posterior probability will be chosen as the posterior probability of cell *i*. Same process should apply on cell *j*.

4. Calculating the posterior probabilities of particles.

All of the particles locate in a grid cell have the same posterior probabilities as that of the cell. Hence we normalize all particles' posterior probabilities to get the weight of each particle, and then implement MCL to estimate the position of the robots.

B MCL Multi-robot Localization Based on Characteristic Particles

In grid based MCL method, the central position of a cell is use to calculate the posterior probability of particles in that cell, so that the precision of estimated position may be degraded. Furthermore, if the space of particle set is large, from Equation 8, we can also find that the number of cells will also be large. If we adjust the size of the cells to reduce the number of cells, the precision of estimated position may be degraded further.

The estimated position of robot is mainly determined by the particles whose weights are large. At the same time, according to MCL algorithm, in updating the distribution of particles set, the particles with higher probability have more chance to be retained. By this means, all particles will be clustered by the particles that have larger weights. That is to say, the larger the weight of the particle has, the more influence it will put on the estimate result. Consequently, we should pay more attention on the particles that has larger weights, and reduce the effect of particles with lower weights on the estimate result. Thus, as a solution, we use changeable grid cells to describe the particles set when we assign particles set to grid cells.

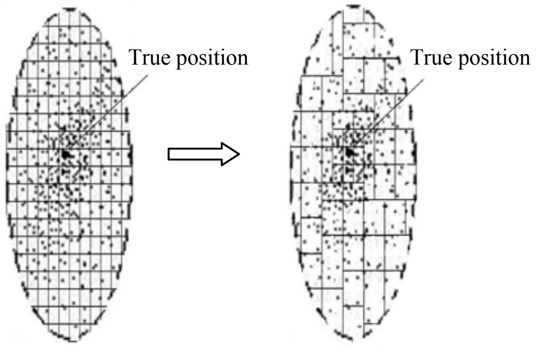

Fig.2 Using changeable grid cell to describe the particle set

When calculating the posterior probability of a cell in grid cell based MCL method, if a cell with small posterior probability, all particles locate in this cell will put less influence on estimate result. So if we properly enlarge the size of the cell, the accuracy of estimate result will not decrease largely, but the amount of cells will be reduced (as showed in Figure 2). A certain particle that near to the central position of the cell can represent the properties of that cell on localization estimation, and can then be used to calculate the posterior probability of that cell. These particles are called characteristic particles, and the characteristic particles set can also replace the whole particle set in calculating the posterior probabilities of particles.

We use grid cells whose size is determined by the Equation 8 as basic cells. When a characteristic particle locates in a certain cell, the characteristic particle will occupy this cell. If a characteristic particle has a small posterior probability, the cell it occupied will be properly enlarged. That is to say, some other basic cells around the characteristic particle will be occupied by this characteristic particle. Consequently, the number of the characteristic particles will be greatly reduced. This approach will be more computing efficient.

This approach works as following:

1. Determine whether characteristic particles need to be used.

After computing the sizes of grid cells by Equation 8, these cells will be used to describe the particles set. If the amount of cells is small enough,

$$Pnum = Binsx \times Binsy < K_p \quad (16)$$

which means that the space of the particles set is small, and all of the particles are around the true position, we need not change the cells' size. K_p is the threshold value. If *Pnum* is larger than K_p, the characteristic particles should be used to reduce the computing complexity.

2. Determining the amount of basic cells that the characteristic particles occupy.

The number of basic cells that the characteristic particles occupy is determined by:

$$deltad = \mathrm{int}(k \times PNum \times (Q\max - Q(S, P_i)) / (Q\max - Q\min)) \quad (17)$$

where *k* is the weights coefficient, *Qmax* is the maximum posterior probabilities of all particles, and *Qmin* is the minimum posterior probabilities of all of the particles. Assuming particle *i* is located in basic cell (*k,l*), all of the basic cells around cell (*k,l*) with radius *deltad* will be occupied by particle *i*. When multiple particles occupy one single basic cell, there should be a competition for these particles. For simplicity, the basic cell will be occupied by the particle with larger posterior probability.

3. Determining if the particle is a characteristic particle.

If the basic cell that a particle located in is exclusive (not occupied by other particles), this particle will be characteristic particle, or else we will go to step 4 for that.

4. Changing the number of basic cells that a characteristic particle occupies.

When a particle is located in a basic cell that already occupied by another characteristic particle with smaller posterior probability, rule (a) and (b) will be applied to determine whether the particle with larger posterior probability will be a characteristic particle. If a particle with larger posterior probability is added into the characteristic particles set, the number of the basic cells occupied by the characteristic particle with smaller posterior probability should be adjusted. If a basic cell is occupied by two characteristic particles simultaneously, the particle with larger posterior probability will occupy this cell.

Rule (a) - If the radius *deltad* of the particle with larger posterior probability is larger than the size of the basic cell and the distance between two particles is larger than a certain distance *d*, which can be obtained by Equation 18, the particle with larger posterior probability is taken to be a characteristic particle.

$$d = K_d \times deltad \quad (18)$$

where K_d is the weight coefficient.

Rule (b) - If two particles do not satisfy rule (a), but they are not located in the same basic cell, and the particle with larger posterior probability satisfies the Equation 19, it is a characteristic particle.

$$Q\max - Q(S,P_i) < K_q \tag{19}$$

where K_q is a threshold parameter.

The detailed description of this algorithm is showed in the following.

```
Suppose there are M particles belong to robot1:
If (PNum<Kp)
{ For(i=1,i<=M,i++)
   { row=particle(i).row;
     col=particle(i).col; //get the position of particle i in grid cells;
     if(cell(row,col) is empty)
        cellocuppy(row,col,i);//particle i occupying the cells;
     elseif(the $Q(S,P_i)$ of particle i larger than the $Q(S,P_i)$ of particle occupying this cell)
        {calculate the distance d between particle i and the particle occupying this cell;
           If(distance satisfy (a) or $Q(S,P_i)$ satisfy (b))
              cellocuppy(row,col,i);
        }
   }
}
cellocuppy(row,col,particleindex);
{ put particle particleindex into characteristic particles sets.
  Calculate deltad by the weight of particle particleindex.
  Search all cells locate in the space that around (row,col)with radius deltad;
  { if(the cell is empty)
      {this cell occupied by particle particleindex;}
      else if(the $Q(S,P_i)$ of particle particleindex larger than the $Q(S,P_i)$ of particle occupying this cell)
         { this cell occupied by particle particleindex;}
  }
}
```

Algorithm 1 Obtaining the characteristic particles

IV. SIMULATIONS AND EXPERIMENTS

A Simulation of obtaining the characteristic particles

When using MCL method to estimate the position of a robot, we use 500 particles in this simulation. At a certain time, suppose robot is located at (45,60), while the position of particles are shown as Figure 3. The robot can use sensor readings to get its true position, and the sensor reading noise is a zero mean Gaussian random variable with variance 2. Using algorithm 1, we can obtain the characteristic particles set shown as Figure 4. In this experiment, the number of characteristic particles is 71. Since the total number of particles is greatly reduced, the computing time can be reduced accordingly in turn. Even then, because there are enough particles around the true position used to estimate the pose of the robot, quite good estimation result can be obtained.

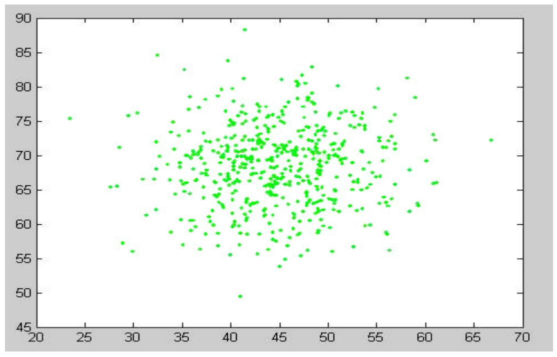

Fig.3 The particles set

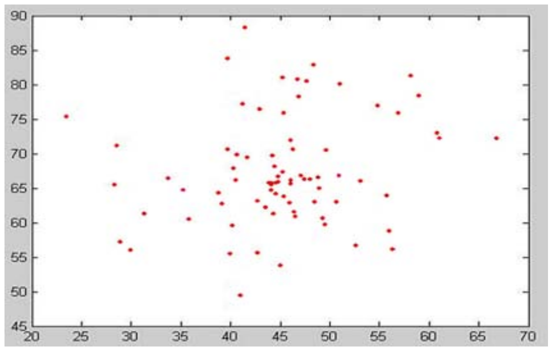

Fig.4 The characteristic particles set

B Simulation of collaborative localization

p'pIn this simulation, we will use two robots. Assuming the motion model of *robot1* can be written as:

$$\begin{cases} X_t = 2.1\times\cos(1.2\times t) + X_{t-1} + 0.2 + w_{1t} \\ Y_t = 2.1\times\sin(1.2\times t) + Y_{t-1} + 0.3 + w_{2t} \end{cases} \tag{20}$$

The motion model of *robot2* is:

$$\begin{cases} X_t = 2.5\times\cos(1.2\times t) + X_{t-1} + 0.2 + w_{1t} \\ Y_t = 2.5\times\sin(1.2\times t) + Y_{t-1} + 0.3 + w_{2t} \end{cases} \tag{21}$$

where w_{it} is a Gaussian noise with mean of 0.2, and variance of 0.5. *robot1* and *robot2* are equipped with some sensors such as visual sensors that can be used for detecting each other and can obtain their relative distance. In our simulation experiment, robots can always detect each other during experiment, and the measurement model is given as following:

$$L = \sqrt{(X_{1t} - X_{2t})^2 + (Y_{1t} - Y_{1t})^2} + v_t \tag{22}$$

where v_t is a zero mean Gaussian random variable with variance 2. Figure 5 shows the comparison of the estimation errors obtained by several different localization methods.

Firstly, only the robot motion model Equation 1 is used to estimate robots position. Due to the process noise, the estimation error goes upward fast over time. Then we implemented grid cell based MCL method. The amount of grid cells is limited to 100. When the number of grid cells is large than 100, the size of the cell is changed to make the number equal to 100. Since the sizes of the cells near the true position may be larger than *L*, the result may be not accurate enough. Thirdly, the grid cells based MCL method is used with the *L* left unchanged. Despite the number of the basic cells is very large, its error is reduced. But because the amount of cells may be large, the method may be very complicated. The characteristic particles method described in section 3 is also used to estimate the position of the robots. The estimation error is similar to that of the *L* unchangeable grid cell based MCL method, but the computing efficiency is proved a lot since the number of the characteristic particles is smaller.

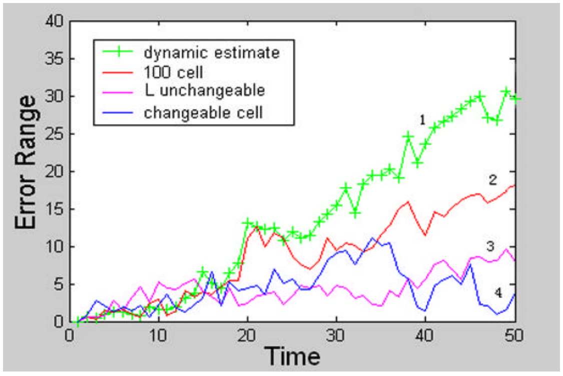

Fig.5 Comparing the errors of different estimation method

V. CONCLUSIONS

As a common approach in robot localization, MCL method can not be directly used in collaborative localization due to the curse of dimensionality. In our approach, instead of the density trees, a kind of grid cell is used to partition the whole particles into sub-sets, and then adjust the size of the grid cells accordingly to obtain the characteristic particles, which can represent the whole particles' property on estimating robot's pose. Because the number of characteristic particles is much less than that of the whole particles, this approach can improve the computing efficiency greatly without loss of estimating accuracy. Simulation results show that this approach can achieve good localization performance in multi-robot system.

ACKNOWLEDGMENTS

This work is funded by research grants from the Nation High Technology Research and Development "863" Program of P.R. China (Grant No. 2001AA422200) and National Natural Science Foundation of P.R. China (Grant No.60375037).

REFERENCES

[1] J. M. Armingol, L. Moreno, A. de la Escalera and M. A. Salichs. Landmark Perception Planning for Mobile Robot Localization. proceedings of IEEE International Conference on Robotics and Automation[C], IEEE Computer Society, Leuven, Belgium, 1998, vol.4, pp. 3425 – 3430

[2] Hyun-Deok Kang,Kang-Hyun Jo. Self-localization of mobile robot using Omni-direcional vision. Proceedings of The 7th Korea-Russia International Symposium on Science and Technology, Korea, vol.2, No.28, 2003, pp. 86 – 91

[3] Stergios I. Roumeliotis, George A. Bekey. Distributed Multirobot Localization. IEEE Transactions on Robotics and Automation, Vol. 18, No. 5, 2002, pp. 781 - 795

[4] John R. Spletzer Camillo J. Taylor. A Bounded Uncertainty Approach to Multi-Robot Localization. IEEE/RSJ International Conference on Intelligent Robots and Systems, Las Vegas, USA , vol. 2, 2003, pp. 1258-1264

[5] Andrew Howard, Maja J Matari´c, and Gaurav S Sukhatme. Localization for Mobile Robot Teams: A Distributed MLE Approach. Experimental Robotics VIII, Bruno Siciliano and Paulo Dario (eds), Springer-Verlag, 2003, pp. 146 – 155

[6] S. I. Roumeliotis and G. A. Bekey. Collective localization: a distributed kalman filter approach. In Proceedings of the IEEE International Conference on Robotics and Automation,vol.3, San Francisco, U.S.A, 2000, pages 2958–2965.

[7] Dieter Fox, Wolfram Burgard, Hannes Kruppa, Sebastian Thrun. A Probabilistic Approach to Collaborative Multi-Robot Localization. Autonomous Robots on Heterogeneous Multi-Robot Systems, 2000, 8(3), pp. 325 – 344.

[8] Dieter Fox. Kld-sampling: Adaptive particle filters and mobile robot localization. Proceedings of the 2003 IEEE International Conference on Robotics and Automation(ICRA 2003), Taipei, Taiwan, 2003, pp. 2836 – 2841

[9] Thomas Rofer, Matthias Jungel.. Vision-Based Fast and Reactive Monte-Carlo Localization. Proceedings of IEEE International Conference on Robotics and Automation (ICRA 2003), vol.1, 2003, pp. 856 - 861

[10] Ioannis Rekleitis, Gregory Dudek, Evangelos Milios. Probabilistic cooperative localization and mapping in practice. Proceedings of IEEE International Conference on Robotics and Automation (ICRA 2003), vol.2, 2003, pp. 1907 – 1912

[11] David C.K, Yuen and Bruce, A. MacDonald. An Evaluation of the Sequential Monte Carlo Technique for Simultaneous Localisation and Map-building. Proceedings of IEEE International Conference on Robotics and Automation (ICRA 2003), vol.2, 2003, pp. 1564 - 1569

[12] Cody Kwok, Dieter Fox, Marina Meil a. Adaptive Real-time Particle Filters for Robot Localization. Proceedings of IEEE International Conference on Robotics and Automation (ICRA 2003), vol.2, 2003, pp. 2836 - 2841

[13] Emanuele Menegatti, Mauro Zoccarato, Enrico Pagello, Hiroshi Ishiguro. Image-Based Monte-Carlo Localisation without a Map. Advances in Artificial Intelligence Proc. of 8th Congress of the Italian Association for Artificial Intelligence, Springer, Pisa, Italy, 2003, pp. 423-435

[14] Michael Montemerlo, Sebastian Thrun, William Whittaker. Conditional Particle Filters for Simultaneous Mobile Robot Localization and People-Tracking. Proceedings of IEEE International Conference on Robotics and Automation (ICRA 2002), vol.1, 2002, pp. 695 - 701