Mobile Robots Global Localization Using Adaptive Dynamic Clustered Particle Filters

Zhibin Liu, Zongying Shi, Mingguo Zhao, Wenli Xu

Department of Automation, Tsinghua University, Beijing, P.R.China
liu-zb04@mails.tsinghua.edu.cn

I. INTRODUCTION

Global localization problem considers how to globally localize the robot without any knowledge about the robot’s initial position, when the map of the environment is given.

In literature, there are mainly three families of algorithms providing a solution to global localization: multi-hypothesis Kalman filters [1, 2], grid-based Markov localization [3, 4] and Monte Carlo localization (MCL) [5, 6, 7]. MCL is the most popular method and has been shown to be effective in many situations. It is robust to arbitrary noise distributions and non-linear system dynamics, and its computational complexity is medium amongst the three families of algorithms. But, it faces some important problems.

One of the key problems is the selection of number of particles. The efficiency and accuracy of the particle filter depends mostly on this number. An effective way to tackle this problem is to adapt the number of particles to the underlying state uncertainty [8, 9].

Another key drawback of standard MCL for solving global localization is: the samples often converge to a single, high-likelihood pose hypothesis too quickly. When there are multiple pose hypotheses with nearly equal probability (as the case in symmetric environments), there will be some chances that MCL incorrectly converges to one of those hypotheses which is not the robot’s real pose. In order to overcome this limitation, a higher diversity of the samples should be kept. In [10], the resampling step are constrained according to effective sample size (ESS), and in [11] a more sophisticated method termed Clustered Particle Filtering (CPF) is proposed, which classifies the particles into different clusters according to their spatial similarity. Each cluster is considered to be a pose hypothesis and all clusters are independently tracked. The resulting localization method is termed Cluster-MCL. This method is sufficient to keep sample diversity between different clusters, but it is not capable of keeping diversity of samples within the cluster.

This article presents an adaptive dynamic clustered particle filtering method for tackling global localization problem. It classifies the particles into a number of clusters which are dynamically evolved by merging or splitting operations with adaptive sample sizes. More specifically, we first derive a dynamic clustered particle filtering method which is an extension of CPF, by introducing the idea of merging and splitting the previously formed clusters. Then, in order to keep better balance between the efficiency and accuracy, we investigate the lower bound of the number of particles needed for each cluster so as to limit the estimation error, by utilizing the central limit theorem in multi-dimensional space and the statistic theory of Importance Sampling (IS). Finally, the problem about how to tune the density of particles in each cluster according to the derived lower bound is discussed.

This paper is organized as follows. Section II presents the theoretical derivation and implementation procedure of adaptive dynamic clustered particle filtering method. Then, in Section III the effectiveness of our method has been tested by simulation. Finally, conclusions are drawn in Section IV.

II. ADAPTIVE DYNAMIC CLUSTERED PARTICLE FILTERING

For the sake of clarity, we describe our approach with three subsections in succession. First, a logical subroutine of the overall method, i.e. dynamic clustered particle filtering, is presented. Then, the lower bound of sample size for each cluster is derived, with the corresponding method for tuning the inner-density of clusters brought forth. Finally, the whole global localization algorithm is outlined.

A. Dynamic Clustered Particle Filtering (DCPF)
1) Theoretical Derivation: Suppose the whole particle set is partitioned into $M$ clusters, and the number of particles belonging to $j$th cluster is denoted as $n_j$, $j = 1, \ldots, M$. $N$ is the total number of particles summed over all clusters. Denote $B_{j,t}$ as the weight of $j$th cluster at time $t$, which can be considered as the probability that cluster $j$ contains the actual robot position. We have the following equations:

$$\sum_{j=1}^{M} B_{j,t} = 1 \quad \forall j \in \{1, \ldots, M\}, \quad \sum_{j=1}^{n_j} w_{j,t}^{i} = 1 \quad (2)$$

Denote the robot’s state at time instant $t$ as $x_t$, $z_t$ is the perceptual data (observation) at time $t$, and $u_{t-1}$ is the odometry data (control measurement) between time $t$ and $t-1$. Additionally, define $U_{0:t-1} = \{u_0, u_1, \ldots, u_{t-1}\}$ the history of control inputs and $Z_{0:t} = \{z_0, z_1, \ldots, z_t\}$ the set of all observations up to time $t$. Then the entire posterior probability distribution is presented by:

$$p(x_t | U_{0:t-1}, Z_{0:t}) \approx \sum_{j=1}^{M} B_{j,t} \sum_{i=1}^{n_j} w_{j,t}^{i} \delta(x_t - x_t^{i}) \quad (3)$$

Since each cluster is independently tracked by particle filtering algorithm, the weight updating process for the particles in $j$th cluster ($\forall j \in \{1, \ldots, M\}$) can be given by:

$$w_{j,t}^{i} = \eta_{j,t} w_{j,t-1}^{i} p(z_t | x_t^{i}, u_{t-1}) p(x_t^{i} | x_{t-1}^{i}, u_{t-1}) \quad (4)$$

where $q(x_t^{i} | x_{t-1}^{i}, u_{t-1}, z_t)$ is the proposal distribution, $\eta_{j,t}$ is a normalizing factor which is equal to:

$$\eta_{j,t} = \left[ \sum_{i=1}^{n_j} \frac{w_{j,t-1}^{i} p(z_t | x_t^{i}) p(x_t^{i} | x_{t-1}^{i}, u_{t-1})}{q(x_t^{i} | x_{t-1}^{i}, u_{t-1}, z_t)} \right]^{-1} \quad (5)$$

After updating the particles’ weights in each cluster, we have to update the weight of each cluster $B_{j,t}$. First, the posterior probability distribution in (3) can be represented by:

$$p(x_t | U_{0:t-1}, Z_{0:t}) = \sum_{j=1}^{M} B_{j,t} \cdot p_j(x_t | U_{0:t-1}, Z_{0:t}) \quad (6)$$

where $p_j(x_t | U_{0:t-1}, Z_{0:t})$ is component density and satisfy $p_j(x_t | U_{0:t-1}, Z_{0:t}) \approx \sum_{i=1}^{n_j} w_{j,t}^{i} \delta(x_t - x_t^{i}) \quad (7)$

According to Bayesian estimation framework, the posterior distribution after the updating step can be obtained:

$$p(x_t | U_{0:t-1}, Z_{0:t}) = \int \frac{p(z_t | x_t) p(x_t, U_{0:t-1}, Z_{0:t-1})}{\int p(z_t | x_t) p(x_t, U_{0:t-1}, Z_{0:t-1}) dx_t} \quad (8)$$

On the other hand, in equation (6), $p_j(x_t | U_{0:t-1}, Z_{0:t})$ can be transformed using Bayes rule:

$$p_j(x_t | U_{0:t-1}, Z_{0:t}) = \frac{p(z_t | x_t) p(x_t | U_{0:t-1}, Z_{0:t})}{\int p(z_t | x_t) p(x_t | U_{0:t-1}, Z_{0:t}) dx_t} \quad (9)$$

Substitute (9) into (6) and compare with (8), we have:

$$B_{j,t} = \frac{\int p(z_t | x_t) p_j(x_t | U_{0:t-1}, Z_{0:t}) dx_t}{\sum_{j=1}^{M} B_{j,t-1} \int p(z_t | x_t) p_j(x_t | U_{0:t-1}, Z_{0:t}) dx_t} \quad (10)$$

Now, we can update $B_{j,t}$ according to equation (10), where

$$\int p(z_t | x_t) p_j(x_t | U_{0:t-1}, Z_{0:t}) dx_t = \int \int p(z_t | x_t) p(x_t, U_{0:t-1}, Z_{0:t-1}) p_j(x_t | U_{0:t-1}, Z_{0:t}) dx_t dx_{t-1} \approx \sum_{i=1}^{n_j} \frac{w_{j,t-1}^{i} p(z_t | x_t^{i}) p(x_t^{i} | x_{t-1}^{i}, u_{t-1})}{q(x_t^{i} | x_{t-1}^{i}, u_{t-1}, z_t)} = \eta_{j,t} \quad (11)$$

Substituting (11) into (10), we get the final form of weight updating equation:

$$B_{j,t} = B_{j,t-1} \eta_{j,t}^i / \sum_{j=1}^{M} (B_{j,t-1} \eta_{j,t}^i) \quad (12)$$

2) Dynamically Evolving the Clusters: There are two distinct operators for evolving clusters: merging and splitting. The motivations behind this are:

- In ideal cases, there would be one cluster for each of the modes of the multimodal target posterior distribution. So, when there are more than one clusters matched to the same mode, it is better to merge them.
- During the localization process, the status and the number of the modes in target posterior distribution will change. Sometimes, a mode is likely to become quite diffuse or even split into some sub-modes. In these cases, it is reasonable to split the corresponding clusters as to maintain better diversity of the particles.

i) Merging: The distances between the clusters are checked at every iteration. Once the distance between any pair of clusters become less than a certain threshold, the merging operation is called to merge all of the overlapping clusters.

Suppose we already have $M$ clusters at time $t$, and there are $K$ clusters should be merged, whose weights are denoted by $B_{k,j,t}$, where $k, j \in \{1, \ldots, M\}$ for each $j = 1, \ldots, K$. Denote the newly generated cluster after merging as $\chi$, and its weight $B_{\chi,t}$. Suppose the weight for $k$th particle in cluster $j$ is $w_{k,j,t}^{(i)}$, and its new weight after being merged into cluster $\chi$ is $w_{\chi,t}^{(i)}$. Our aim is to calculate $B_{\chi,t}$ and $w_{\chi,t}^{(i)}$. According to the idea of DCPF, the following conditions should be satisfied:

$$\sum_{j=1}^{K} \sum_{i=1}^{n_{k,j}} w_{\chi,t}^{(i)} = 1 \quad (13)$$

$$\sum_{j=1}^{K} \sum_{i=1}^{n_{k,j}} w_{\chi,t}^{(i)} \delta(x_t - x_t^{(i)}) = B_{\chi,t} \sum_{j=1}^{K} \sum_{i=1}^{n_{k,j}} w_{k,j,t}^{(i)} \delta(x_t - x_t^{(i)}) \quad (14)$$

Equation (14) indicates the contribution of the clusters to the entire posterior distribution should not change before and after merging. Substituting equation (2) and (13) into (14), we have:

$$B_{\chi,t} = \frac{\sum_{j=1}^{K} B_{k,j,t}}{\sum_{j=1}^{K} B_{k,j,t}} \quad w_{\chi,t}^{(i)} = w_{k,j,t}^{(i)} B_{k,j,t} / B_{\chi,t} \quad (15)$$

ii) Splitting: The effective sample size (ESS) for each cluster is checked at every iteration. Denote the ESS for $j$th cluster as $N_{eff,j}$, which is defined by $[12, 13]$:

$$N_{eff,j} = 1 / \sum_{i=1}^{n_j} (w_t^{(i)})^2$$

When any cluster’s ESS becomes less than $n/2$ ($n$ is the number of particles included in that cluster) or any cluster becomes too diffuse, the clustering algorithm (due to lacking of space, the algorithm concerning how to classify the
particles into different clusters is not included in this paper) will be called to re-cluster the particles in that cluster. If the particles are indeed classified into a few different clusters with significant gap between each other, the splitting operator will be applied to adjust the weights of each new cluster as well as the particles assigned to them.

Suppose the cluster \( \chi, \chi \in \{1, \cdots, M\} \), is going to split into \( K \) new clusters denoted \( k_j, j = 1, \cdots, K \). The weight of cluster \( \chi \) is \( B_{\chi,\chi} \), and \( B_{\chi,k_j} \) represents the weight of new cluster. Denote the new weight for \( i \)th particle in cluster \( k_j \) as \( w^{(i)}_{t(j)} \), and its previous weight is \( w^{(i)}_t \) when it belongs to the original cluster \( \chi \). Our aim is to calculate \( B_{\chi,k_j} \) and \( w^{(i)}_{t(j)} \).

Again, the following conditions should be satisfied:

\[
\sum_{i=1}^{n_{k_j}} w^{(i)}_{t(j)} = 1, \quad \forall j \in \{1, \cdots, K\} \tag{16}
\]

\[
\sum_{j=1}^{K} B_{\chi,k_j} \sum_{i=1}^{n_{k_j}} w^{(i)}_{t(j)} \delta(x_i - x^{(i)}_t) = B_{\chi,\chi} \sum_{j=1}^{K} w^{(i)}_t \delta(x_i - x^{(i)}_t) \tag{17}
\]

Substituting (16) into (17), we obtain:

\[
B_{\chi,k_j} = B_{\chi,\chi} \sum_{i=1}^{n_{k_j}} w^{(i)}_{t(j)} / \sum_{i=1}^{n_{k_j}} w^{(i)}_{t(j)} \tag{18}
\]

B. Adapting the Sample Size

One of the most elegant methods for adapting the number of particles is KLD-sampling algorithm developed by Fox, et al. in [8]. However, the problem with KLD-sampling is the derivation of the bound of sample size using the empirical distribution, which has the implicit assumption that the samples come from the true posterior distribution. This is not the case for particle filters where the samples come from an importance function. In [9], a revised bound for KLD-sampling is given based on the variance of Importance Sampling (IS). But, when the posterior distribution is multimodal, the expectation and variance of the state vector are not adequate for describing its distribution and the improvement becomes less significant.

As to our dynamic clustered particle filtering method presented above, the multimodality of posterior distribution is properly represented in form of clusters, and each distinct mode is expected to be represented by one cluster only. This property is encouraging, because the overall distribution is naturally decomposed into a set of unimodal component distributions, and expectation and covariance matrix will be effective tools for describing the status of these component distributions, which are approximated by clusters of particles. This inspires us to deduce the lower bound of sample size for each cluster, and then the lower bound of the whole distribution can be easily obtained by summing up the lower bounds of individual clusters.

i) Lower Bound of Sample Size for Clusters. At each cycle of the filtering algorithm, we estimate the number of particles needed for each cluster so as to, with a certain level of confidence, bound the approximation error. Denote the expectation and covariance of \( p_j(x_t | U_{0:t-1}, Z_{0:t}) \) as \( E_p(x_t) \) and \( C_p(x_t) \) respectively. Assume there are \( n \) samples \( x^{(i)}_t \), \( i = 1, \cdots, n \), sampled from \( p_j(x_t | U_{0:t-1}, Z_{0:t}) \), and denote the empirical mean as \( M_n = (x^{(1)}_t + \cdots + x^{(n)}_t)/n \). Then, according to the theory of statistics, we have:

\[
E(M_n) = E_{p_j}(x_t), \quad Var(M_n) = Var_{p_j}(x_t)/n \tag{19}
\]

Further, since \( x^{(i)}_t, i = 1, \cdots, n \), are i.i.d., with mean \( E_{p_j}(x_t) \) and covariance \( C_{p_j}(x_t) \), according to the central limit theorem in multi-dimensional space (because \( x_t \) is a multi-dimensional random vector), when \( E_{p_j}(x_t) \) and \( C_{p_j}(x_t) \) are finite, we have:

\[
\sqrt{n}(M_n - E_{p_j}(x_t)) \xrightarrow{n \to \infty} N(0, C_{p_j}(x_t)) \tag{20}
\]

That is when the number of samples \( n \) goes infinite, the random vector \( \sqrt{n}M_n \) will converge in distribution to a multi-dimensional Gaussian distribution with mean \( \sqrt{n}E_{p_j}(x_t) \) and covariance \( C_{p_j}(x_t) \). In mobile robot localization, the state vector \( x_t \) is usually 3-dimensional, \( x_t = (x_t, y_t, \theta_t) \), we denote \( E_{p_j}(x_t) \) as \( \mu = (\mu_x, \mu_y, \mu_\theta) \), \( C_{p_j}(x_t) \) as \( C \). Then, if we want to bound the error of \( M_n \) in approximating the true expectation \( \mu \) within a given hyper-rectangle region \( \Theta = \{(x, y, \theta) | x \in [-\epsilon_x, \epsilon_x], y \in [-\epsilon_y, \epsilon_y], \theta \in [-\epsilon_\theta, \epsilon_\theta]\} \) with a required confidence level \( 1 - \delta (\delta \text{ is a small positive number within the interval (0,1)} \), the following equation should be satisfied:

\[
P((M_n - \mu) \in \Theta') \geq 1 - \delta \tag{21}
\]

It is equivalent to:

\[
P(\sqrt{n}(M_n - \mu) \in \Theta') \geq 1 - \delta \tag{22}
\]

where \( \Theta' = \{(x, y, \theta) | x \in [-\sqrt{n}\epsilon_x, \sqrt{n}\epsilon_x], y \in [-\sqrt{n}\epsilon_y, \sqrt{n}\epsilon_y], \theta \in [-\sqrt{n}\epsilon_\theta, \sqrt{n}\epsilon_\theta]\} \). According to equations (20) and (22), we obtain the lower bound of the number of particles \( n^* \), which is the root of the following equation:

\[
\int_{-\sqrt{n}\epsilon_x}^{\sqrt{n}\epsilon_x} \int_{-\sqrt{n}\epsilon_y}^{\sqrt{n}\epsilon_y} \int_{-\sqrt{n}\epsilon_\theta}^{\sqrt{n}\epsilon_\theta} \frac{1}{(2\pi)^{1/2}|C|^{1/2}} \exp \left[ -\frac{x^T C^{-1} x}{2} \right] dx dy d\theta = 1 - \delta \tag{23}
\]

However, in particle filters the particles are never sampled from the true density \( p_j(x_t | U_{0:t-1}, Z_{0:t}) \), but from an importance function. According to [14, 9], for an one dimensional random variable \( v \) whose true distribution is \( p(v) \), when the samples come from an importance function \( q(v) \), denote the empirical mean of the samples as \( n_{IS} E_{IS}(v) \), \( n_{IS} \) is the number of samples. \( E_p(v) \) and \( E_q(v) \) are the expectations of \( v \) under distribution \( p(v) \) and \( q(v) \) respectively. We have:

\[
Var[\hat{v}^n E_p(v)] = E_q[(v - E_p(v))^2] / n_{IS} = \sigma_{IS}^2 / n_{IS} \tag{24}
\]

where \( \sigma(w(v)) \) is the importance factor (or weight), and \( \sigma^2_{IS} \) is the variance of Importance Sampling. Compare (24) with (19), if we want to achieve similar levels of accuracy, the variance of both estimators should be equal. So, we have:

\[
n_{IS} = \sigma^2_k \cdot n / Var_{p}(v) \tag{25}
\]
In order to utilize equation (25) which quantifies the equivalence between samples from the true and proposal densities, we will investigate all dimensions of $x_t$ separately. Take dimension $y$ for example. Since $p_j(x_t|U_{0:t-1}, Z_{0:t})$ is represented by a cluster of weighted particles, we have to calculate the following terms using MC integration:

$$E_{p_j}(y_t) \approx \sum_{i=1}^{n} w_t^{(i)} y_t^{(i)}$$
$$\text{Var}_{p_j}(y_t) = E_{p_j}(y_t^2) - (E_{p_j}(y_t))^2$$

The similar lower bound for dimension $x$ and $\theta$ can be derived analogically. Finally, the lower bound of the number of particles needed for this cluster is determined by:

$$n^*_{IS} = \max \{n^*_{IS,x}, n^*_{IS,y}, n^*_{IS,\theta} \} \quad (28)$$

Then the lower bound of particle number for dimension $y$ is:

$$n^*_{IS,y} = \sigma^2_{IS,y} \cdot n^*/\text{Var}_{p_j}(y_t) \quad (27)$$

So, for $j$th cluster, we have to sample $n \geq n^*_{IS}$ particles. The needed number of particles for each cluster can be calculated similarly.

2) Tuning the Sample Size for Clusters. Now we consider how to shrink or enlarge the size of the current sample set in order to satisfy the lower bound. A most intuitive method is to resample $[n^*_{IS,j}]$ samples from the current sample set. But, the diversity of the samples might be seriously affected, and consequently, the lower bound of sample size in the next time instance will also be affected.

Here we present a method for combining and incising the original particles to form a new target particle set, and prevent the size of new particle set from being dramatically smaller than the original particle set.

i) Case $n^*_{IS,j} < n_j$: In this case, we have to combine some of the particles. At first, all of the particles in original sample set are sorted according to their weights in ascending order, and the new target sample set is initialized to be $\Phi$. The size of the target sample set is set to be $n_j = \max \{n^*_{IS,j}, n_j/2 \}$. Then, the last $2n_j - n_j$ particles (with larger weights) in original set are directly picked out and injected into the target sample set. Finally, the other $2(n_j - n_j)$ particles in original set are randomly combined in pairs to form $n_j - n_j$ new particles and injected into the target sample set.

Suppose a pair of samples $s^{(k)}_t$ and $s^{(r)}_t$ are to be combined to form a new sample $s'_t$, we have:

$$s'_t = \frac{x^{(r)}_t w^{(r)}_t + x^{(k)}_t w^{(k)}_t}{w^{(r)} + w^{(k)}}, \quad w'_t = w^{(k)} + w^{(r)}$$

ii) Case $n^*_{IS,j} > n_j$: In this case, we have to incise some of the particles. However, the process of generating new target sample set is a little more complex than that of combining particles. The motivation is that: the particles with larger weight should have more descendants. The algorithm for incising particles is presented in Table I.

C. Outline of the Localization Algorithm

The procedure of the overall global localization algorithm using adaptive dynamic clustered particle filtering (we term it ADC-MCL) is outlined in Table II.

| TABLE I |
| ALGORITHM FOR INCISING PARTICLES TO GENERATE NEW SAMPLE SET |
| 1. Input: the original particle set of the cluster $s^* = \{s^{(i)}_t = (x^{(i)}_t, w^{(i)}_t)|i = 1, \ldots, n_j\}$; the size of the target sample set $n^*_{IS}$. |
| 2. Initialize the target sample set $s^*_t = s^*$, target sample set size $n_j = n_j$. |
| 3. while ($n_j < n^*_{IS}$) do |
| 4. Find the max-weight particle $s^{(m)}_t = (x^{(m)}_t, w^{(m)}_t)$ in $s^*_t$. |
| 5. In the case, we have to combine some particles in $s^*_t$ into a new target particle set. |
| 6. End while |

| TABLE II |
| OUTLINE OF OVERALL METHOD |
| Initial Steps: |
| 1. initialize the sample set according to the initial distribution (usually uniform as in case of global localization); |
| 2. iterate several steps through ordinary MCL; |
| 3. clustering the particles based on their spatial similarity; |
| 4. create a particle filter for each cluster; |

Normal Steps:

In normal operation of ADC-MCL after the first time of clustering, we update each cluster’s particle filter separately, with the weights of both particles and clusters updated according to equations (4) and (13). Then, we dynamically evolve the clusters:

1) merge the clusters that are too similar to each other; |
2) split the clusters which are too diffuse or their effective sample sizes are lower than a certain threshold, by re-clustering the particles in these clusters; |

Validating Step:

If the sum of the weights of particles over all clusters before normalization is lower than a certain threshold, a separate new instance of MCL will run at global level to find likely clusters of particles that are dissimilar to those in current filter. This enables ADC-MCL to cope with the kidnapped robot problem.

III. SIMULATION RESULTS

In this section, we illustrate the performance of our ADC-MCL algorithm compared with those of standard MCL and Cluster-MCL by 2-D simulation. Two simulated environments are established: one is a simple symmetric environment; the other is an indoor environment.

A. In a Simple Symmetric Environment

As shown in Figure 1, the environment in this scenario is a $4m \times 6m$ rectangular region with two different types of landmarks, which are marked small solid squares and circles respectively. Initially, the robot is placed at the position $(−2400mm, 500mm)$, which is not known to the robot. The simulation is carried out by conducting the robot to move around the field counterclockwise. There is a monocular...
vision system mounted on this robot, which enables it to
detect and recognise the landmarks. The robot’s field of
view is 50°, and the detectable distance of all landmarks are
limited to 2 meters. The motion and sensor noises are
Gaussian. The initial sample sizes for the three algorithms
are 3000.

The experiment results of these algorithms are shown in
Figure 2. The robot’s real position is marked by a small
hollow square. The short green line indicates the orientation
of the robot, and the two longer blue lines plot out the
robot’s field of view. Figure 2(a1) and (a2) show the
localization results of standard MCL, where the tiny grey
dots show the distribution of the particles. It is clear that
standard MCL failed in this case, since all of the particles
have converged to the position which is nearly symmetric to
the robot’s real position in (a2). In contrast to standard MCL,
both Cluster-MCL and our ADC-MCL successfully kept
track of the robot’s real pose, with the results shown in (b1),
(b2) and (c1), (c2) respectively. The groups of particles with
different colour correspond to the different clusters in these
two algorithms.

Table III shows the result of 100 independent runs of
these three algorithms. Though in ideal cases the standard
MCL would be expected to converge to the robot’s real pose
due to the slight dissymmetry of the environment, it fails 35
times. On contrary, it is obvious that both ADC-MCL and
Cluster-MCL can perfectly keep track of multiple pose
hypotheses when globally localizing the robot.

However, the computational efficiency of ADC-MCL and
Cluster-MCL differ dramatically (Figure 3). The particle
number of standard MCL and Cluster-MCL are fixed, equal
to 3000. While, in ADC-MCL, the sample size can
adaptively enlarge and shrink as the uncertainty of estimated
robot pose arise and resolved. All in all, the computational
cost of ADC-MCL is far less than that of Cluster-MCL and
standard MCL.

B. In Symmetric Indoor Environment

In this scenario, we focus on the comparison between
ADC-MCL and Cluster-MCL. The environment is shown in
Figure 4. It is a 20m × 30m symmetric indoor environment.
 Initially, the robot is located in one of the rooms, and during
the localization process it will move to another room. The
trajectory of the robot is shown by the red dotted line.

Figure 5 shows the localization results of Cluster-MCL
algorithm. After a few steps, the particles converged to two
distinct hypothetical positions with two clusters formed
accordingly (Fig. 5(a)). As the robot moves, the particles in
both clusters diffuse. Then, as shown in Fig. 5(c), the door of
the target room is detected. However, because the doors of
the two neighbouring rooms look almost the same, the
particles in each cluster will actually assemble around two
similar positions. But, unfortunately, due to the fact the
diversity of the particles in the same cluster can't be well kept by Cluster-MCL, only one hypothesis survived for each cluster in Fig. 5(d). This leads to the failure of Cluster-MCL.

Our ADC-MCL overcomes the drawbacks of Cluster-MCL mentioned above. The result is shown in Figure 6. The situations described by Figure 6(a) and (b) are similar to those of Cluster-MCL. But, as indicated by Figure 6(c), when the door of the target room is detected, the particles in each cluster are partitioned into two different clusters. That is, the previously formed two clusters split into four descendant clusters. Seen from Figure 6(d), the robot’s true pose is correctly tracked till the end of the test.

Table IV shows the comparison result of 100 independent runs of Cluster-MCL and ADC-MCL, which indicates that ADC-MCL is more robust and effective.

![Fig. 5 Global localization using Cluster-MCL.](image1)

![Fig. 6 Global localization using ADC-MCL.](image2)

<table>
<thead>
<tr>
<th></th>
<th>Result of 100 Independent Runs of Cluster-MCL and ADC-MCL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster-MCL</td>
<td>Times failed: 32</td>
</tr>
<tr>
<td>ADC-MCL</td>
<td>Times failed: 0</td>
</tr>
</tbody>
</table>

IV. CONCLUSION

An adaptive dynamic clustered particle filtering method for mobile robot global localization is proposed. First, the particles are classified into a number of clusters based on their spatial similarity, and all these clusters are dynamically evolved by merging the overlapping ones and splitting those become too diffuse or the particles in them gather to some sub-clusters inside. An attractive feature of this method is that one cluster usually corresponds to one distinct mode of the multimodal target distribution. This inspires the method for adapting the sample size of each cluster to its underlying state uncertainties, in order to promote computational efficiency without sacrificing estimation accuracy. The theoretical lower bound of particle number for any cluster is derived, based on the central limit theorem in multi-dimensional space and the statistic theory of Importance Sampling (IS). In addition, the method for robustly tuning the sample size of each cluster to satisfy the derived lower bound is proposed. The effectiveness of our ADC-MCL has been shown by simulation, where it can keep track of the robot’s real pose throughout the experiments in symmetrical environments. Both the number of clusters and particles are adaptively tuned according to the property of the posterior distribution, which leads to significant improvement in both robustness and computational efficiency.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China under Grant NO. 60674017 and the Education Foundation of Tsinghua University with Grant NO. 202025001.

REFERENCES