Sensor Planning for Mobile Robot Localization - A hierarchical approach using a Bayesian network and a particle filter -

Hongjun Zhou, *Member, IEEE,* Shigeyuki Sakane, *Member, IEEE,*

***Abstract*— In this paper we propose a hierarchical approach to solving sensor planning for the global localization of a mobile robot. Our system consists of two subsystems: a lower layer and a higher layer. The lower layer uses a particle filter to evaluate the posterior probability of the localization. When the particles converge into clusters, the higher layer starts particle clustering and sensor planning to generate an optimal sensing action sequence for the localization. The higher layer uses a Bayesian network for probabilistic inference. The sensor planning takes into account both localization belief and sensing cost. We conducted simulations and actual robot experiments to validate our proposed approach.**

***Index Terms*— Sensor planning, Localization, hierarchical approach, Particle filter, Bayesian Network**

I. Introduction

In recent years, the global localization of mobile robots has been an active area of research and various methods have been developed. However, most of the studies have taken a passive approach ([2]-[9]) in the sense that the robot's actions are random and are not planned explicitly for efficient localization. In this paper, we propose a hierarchical approach to solving sensor planning for global localization.

In the reports of ([2], [3]), the robot environment was represented by an occupancy grid map (cell size: 0.1 x 0.1 m^2), and the robot pose is updated after the robot's movement and sensor activation based on a hidden Markov model. A Monte Carlo version of global localization (MCL) has been reported in ([4], [9]). The particle filter resamples and updates the belief of localization, and estimates the maximal posterior probability density for the localization. The advantage of MCL is that arbitrary distribution can be represented by a set of samples, and the computational complexity of the Bayes filter[4] is reduced. As reported in ([2], [3], [4], [9]), the robot pose was represented by a small occupancy grid and high angle resolution. In [10] and [11], the environment was represented by a topological map, with corridors, T-junctions and doorways decomposed into larger cells (cell size: 1 x 1 m^2), and with each cell corresponding to a node of the topological map. The robot updates its pose based on the cells using the Partially observable Markov decision process (POMDP)[1]. Since these approaches assume that it is not necessary to know the robot's pose in detail, the pose of the robot is represented based on the larger cell and four compass directions. In the above approaches, since the robot moved randomly without sensor planning, the systems did not always assure efficient convergence of the global localization if the environment had locally similar sensing patterns.

[1]Partially observable Markov decision process (POMDP) is a technique for choosing optimal actions in partially observable stochastic domains.

The theory of POMDP is an extension of the Markov decision process (MDP). Using the viewpoint of a Bayesian network, we can put a POMDP into the domain of a hidden Markov model (HMM) using action control. POMDPs are well understood and widely applied to mobile robot applications. The observation node of a POMDP corresponds to the sensor data or its features, and the state node corresponds to the robot pose. However, in practice, the robot sensor information is not always only one type, and a real and complex environment may need multi-sensor data for representation. How to integrate such multi-sensor data will be a challenge for the traditional POMDP style model.

A Bayesian network is an extension of the HMM and Kalman filter. It allows one state node to have multiple observation nodes, and one observation node corresponds to multiple state nodes. In our system, we have employed a Bayesian network to represent and integrate the sensor information, the robot pose and the sensing actions. This model allows us to add multiple types of sensor data arbitrarily. Thus, our use of a Bayesian network is more general and extensible than the POMDP style model.

Active global localization is a research topic extending from passive global localization and addresses sensing action selection for efficient global localization. In the past decade, some active localization approaches ([13]-[23]) have been proposed. Jensfelt and Kristensen[13] used multiple hypothesis tracking based on Kalman filtering for active global localization. The environment of their system was represented by a topological graph. The robot pose hypotheses were tracked based on Kalman filtering and the robot sensing commands were generated by POMDP style planning. Tomatis et al.[19] also proposed integrating a Kalman filter-based metric map and a POMDP-based topological map for robot localization and navigation. These systems([13], [19]) are similar to our hierarchical system. We employ a particle filter to track the robot accurately. Our particle filter can deal with linear/non-linear dynamics with arbitrary noise. A basic Kalman filter is limited to a linear assumption. An Extended Kalman Filter (EKF) and Unscented Kalman Filter (UKF) can deal with non-linear dynamics, but need to assume the existence of Gaussian noise. We employ a particle filter only for robot localization in the lower layer, and it is not the core algorithm of our sensor planning for localization.

Basye et al.[20] and A. Cassandra et al.[21] have proposed a POMDP model for robot navigation and localization. As described in papers ([10],[11], [21]), since solving for optimal sensor strategies with accurate localization is intractable, their environments were represented by topological maps based on large cells. Their robots also could not achieve accurate localization.

To reduce the computational overhead of POMDP style planning in a large-scale environment, we represent the environment in a hierarchical architecture. In the higher layer, the large cells represent the topological information of the environment. In the lower layer, a particle filter estimates the accurate robot pose based on an occupancy grid map (grid size: 0.05 m). We can thus obtain sensing strategies based on the topological map in the higher layer using a Bayesian network, and localization accuracy through the lower layer using the particle filter.

A sensor planning ideal is not only being applied in active global localization, but it is also being employed by robot intelligent navigation([12], [24]), active vision[33]. Zhou and Sakane ([22], [23]) proposed two systems to perform sensor planning for localization using a Bayesian network. Mourikis et al.[38] proposed a method of optimal sensor scheduling for resource-constrained localization of a robot group.

Some studies have used Bayesian networks for modeling the mobile robot navigation environment. Asoh et al.[31] developed a system that combines local information for localization using a Bayesian network. However, their system did not actively plan how the mobile robot should gather sensor information, and the Bayesian network structure was manually designed.

Fox et al.[18] proposed an *Active Markov Localization* method to improve the efficiency of localization. Their method selects the optimal sensing action sequence based on an action selection function that takes into account the trade-off between the increase of global localization probability entropy and the sensing cost. The increase of the global localization probability entropy is evaluated by calculating the difference in the entropy for the localization belief and the belief in all free positions in the environment. The environment is represented by occupancy grids, and the cell size is approximately 0.1 x 0.1 m^2. Their policy of selecting the optimal sensing action seems similar to our system. However, compared to *Active Markov Localization*, there are two points that differ between our method and *Active Markov Localization*. First, the cell size of the sensor planning step is different. *Active Markov Localization* evaluates all of the occupancy grids (with a grid size of about 0.1 x 0.1 m^2) to select the optimal actions for localization. Also, our system has a hierarchical structure, and the system obtains the sensing strategy for localization from the higher layer using a Bayesian network based on larger cells that are decomposed upon the occupancy grid map. So the cell size of *Active Markov Localization* is smaller than that in our system. Since the sensor planning is performed on the occupancy grids, the actions optimized by *Active Markov Localization* are more precise than our method. However, in practice, the robot does not need such precise sensor planning

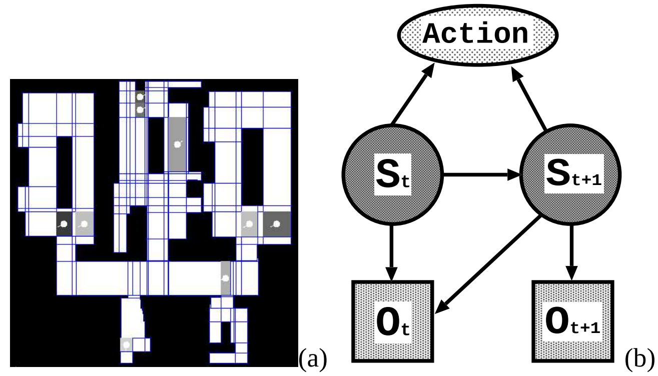

Fig. 1. (a) The initial status of the *higher layer control*. (b) The Bayesian network used in our system.

for localization, as the robot only needs to know in which room (or area) can it localize itself quickly, or which action is better for localization when it faces an intersection. For this reason, the sensing action path planned by our method will not be really optimal in the strictest sense. But it is adequate to increase the efficiency of the localization. The second point of difference is that we use a Bayesian network to estimate the gross localization and plan the sensing action. The HMM style model which is employed by the *Active Markov Localization* also is a type of Bayesian network. A Bayesian network is very expressive for representing the causal relationships between the robot pose, sensor action and sensor information. Since a Bayesian network has these advantages, it is more general and extensible than the HMM style model. Using a Bayesian network to represent multiple sensors (for example, cameras, lasers and so on) for localization is more natural than the HMM style model. In summary, the original contributions of our study and this paper are:

1) A general hierarchical approach for *active* and *precise* localization.
2) Unlike the traditional active localization approach, our sensor planning system uses a Bayesian network for robot *gross localization.*

II. Outline of the System

A. System Structure

Our mobile robot estimates its pose by the hierarchical integration of the two layers. We initially decompose the environmental map into some large cells (see Fig.1(a)), and perform clustering of each cell's geometrical features to discriminate the classes of the cells. The lower layer uses a particle filter to evaluate the posterior probability of the localization. When the particles converge into clusters, the higher layer starts particle clustering and sensor planning to generate an optimal sensing action sequence for the lower layer's localization. The higher layer consists of a Bayesian network[25] which represents the contextual relationships between the geometrical features of the cells, the sensing actions, and the global localization beliefs. The robot actively localizes itself precisely by the planned actions, which are obtained by the higher layer.

B. Optimal Path Selection and Sensing Cost Definitions

In our system, we define the robot's sensing cost by the distance between the center points of the larger cells. The planned action path indicates the direction when the robot faces an intersection or decides whether or not to move into a certain room (or area) for effective localization. So the optimal sensing cost is not related to a minimum distance of travel in a strict sense. Rather, the planned path is optimized based on the decomposed larger cell.

III. Cell Decomposition of the Environment

The gross localization is based on large cell decomposition. To obtain the geometrical features of the local environment, which a robot might come across during navigation, the system decomposes the environment into cells. Figure 1(a) shows an example of the exact cell decomposition[26] which is performed twice, i.e., from left to right and from top to bottom.

Our system uses a mobile robot (Pioneer 3) with two laser range sensors (SICK, LMS200). The range data is generated covering a 360-degree view. Since the system employs a cell-based representation, the robot has to recognize the cell that it currently occupies. To achieve this, the system performs clustering of the geometrical features of the cells. The robot gatherers range sensor data at the central point of every large cell in four compass directions, and classifies the range data into 20 classes using a K-means clustering algorithm [27]. Using these classified results, we can represent and recognize the geometrical features of the cells.

In our system, we do not need to obtain an accurate classified result in the K-means classification step, where the classified results only give us initial probabilities of the robot localization. The ambiguity of the localization will be reduced at the *gross localization* step using the Bayesian network and the lower layer, i.e., *particle filter*. We used the K-means clustering algorithm to classify the laser data and calculated the distances between the observed laser data and the center points of each classified class. The distances are normalized to probabilities, and the probabilities are the initial probability of the robot localization. The initial probability is input into the Bayesian network as soft evidence.

IV. Lower Layer for Accurate Localization

In the past decade, particle filters[28] have been applied to various areas such as visual tracking[29] and mobile robot localization[30]. A particle filter estimates the posterior probabilities of unobservable state variables from sensor measurements. In the application to mobile robot localization, a state typically represents a pose of the robot in the environment. The particle filter approximates the posterior using a group of samples (or particles) with weights that correspond to the likelihood of the states. In every time step, the particles are resampled based on the weight values. A particle filter for mobile robot localization is shown in Fig. 2. The pose of each particle simulates the possible pose of the mobile robot. To perform global localization, the initial particles have a uniform distribution in the free space of the environment (Fig. 2(a)). The initial sensing action of this example is random

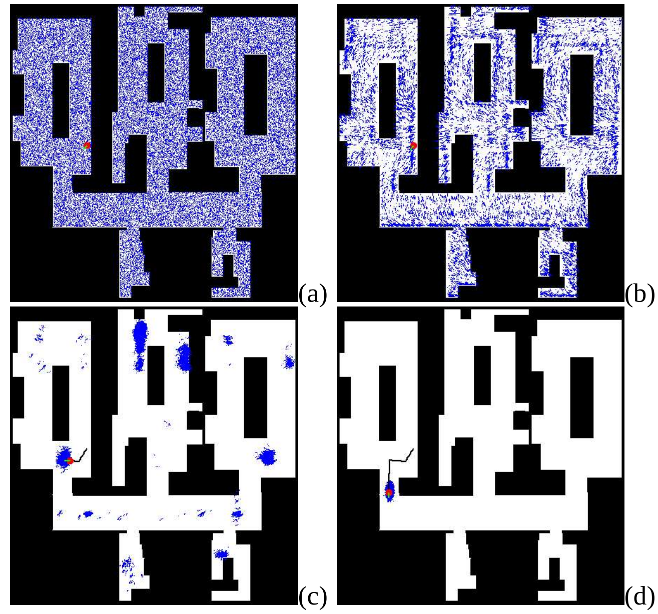

Fig. 2. A example of mobile robot localization using a particle filter by random actions.

without planned action. After a sequence of sensing actions, the particles converge into clusters (Fig. 2(b), (c)). When the mobile robot turns left at an intersection, the particles conveniently converge into one cluster (Fig. 2(d)).

V. Higher Layer Control

A. Bayesian network design

The higher layer employs a Bayesian network for sensor planning as shown in Fig. 1. The nodes S_{t+1} and S_t denote the states at time $t+1$ and t, respectively. A state is a pose of the mobile robot at each large cell, and a state transition occurs when the robot moves to the next cell or changes its orientation. The higher layer performs gross localization.

We quantize the robot orientation in the cell into four compass directions (east, west, north and south), so the robot has four states in each cell. The node $Action$ denotes the robot motion while moving from one cell to an adjacent cell. The nodes O_{t+1} and O_t denote the observation (sensor measurement) obtained at states S_{t+1} and S_t, respectively. The observation is not raw sensor data at each cell but is the classified results of the sensor data (see Section III).

In the following experiments, the environment is decomposed into 103 cells and the robot has four possible orientations at each cell, so the robot has $103 * 4 = 412$ states. The same number of random variables represents these states and each variable has a probability, i.e., a real value from 0 to 1. The node S_t includes these 412 variables and the node S_{t+1} has 413 random variables which correspond to the 412 states plus an error state (exception or obstacle). The node $Action$ includes four random variables, which correspond to four actions: go forward, go backward, turn left, and turn right. The nodes O_{t+1} and O_t have 20 random variables as defined in Section III. The initial values of the node S_t are generated

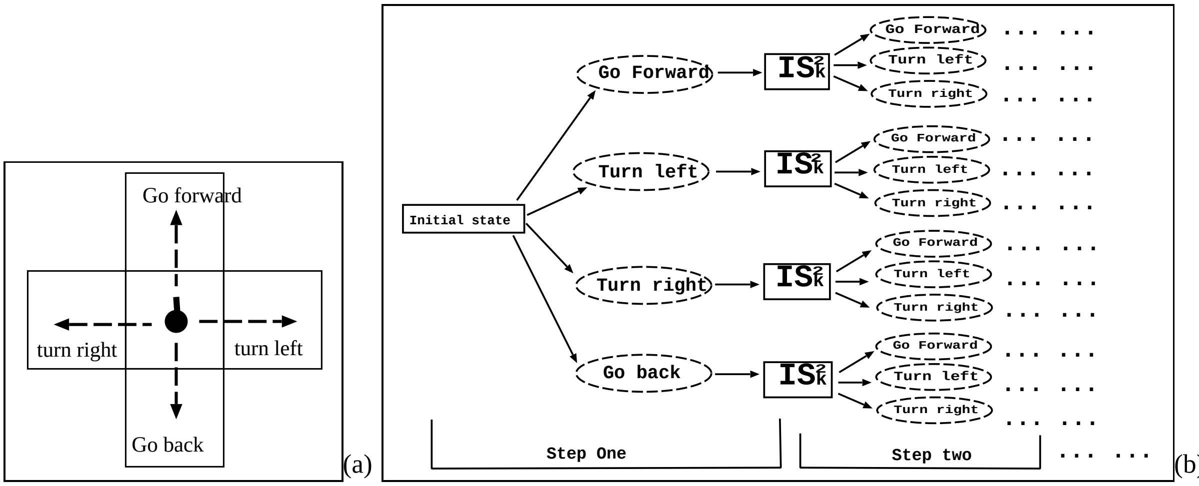

Fig. 3. (a) Four possible actions of the mobile robot in the sensor planning phase. (b) State transition during the sensor planning for mobile robot localization.

from the sum of the weights of the particles in each particle cluster.

After the particle clustering phase, the system locates some particle clusters. Cell-based sensor planning starts at this stage. The possible action sequences and state transition of each cluster during the sensor planning are shown in Fig. 3(b). In Fig. 3(b), $Initial\ state$ denotes one pose of the robot with a white circle (see Fig. 1(a)), which was found in the particle clustering phase. We use $IS_k \in \{IS_1, IS_2, ..., IS_K\}$ to denote one of $Initial\ state$. In the Fig. 1(a), there are nine $Initial\ state$s, so $K = 9$.

IS_k^2 is the pose of the robot, when the circle has finished a possible action, we call this step $Step\ one$. The white circles will move from the $Initial\ state$ to adjacent cells after $Step\ one$. When the circle takes a two-step action, the action sequences and state combinations are illustrated in $Step\ two$ of Fig. 3(b).

B. Learning Conditional Probability Tables of the Bayesian Network

A Bayesian network is represented by two factors[25], the graph structure and conditional probability tables (CPTs). The graph structure of a Bayesian network represents the causal relationships between the nodes. The graph structure of our Bayesian network was designed manually (see Fig. 1). The CPTs of the nodes can be learned from case data directly. Since our environment information data is complete, we use the $Maximum\ Likelihood\ Estimation$[32] method to learn the CPTs. For example, the conditional probability of node X's value x_i is $P(X = x_i | Z_1 = z_1, Z_2 = z_2, ..., Z_k = z_k)$, and $Z_1, Z_2, ..., Z_k$ is K parents nodes of node X. $z_1, z_2, ..., z_k$ are denoted by the values of nodes $Z_1, Z_2, ..., Z_k$. So the conditional probability $P(X = x_i | Z_1 = z_1, Z_2 = z_2, ..., Z_k = z_k)$ can be calculated as the following:

$$P(X = x_i | Z_1 = z_1, Z_2 = z_2, ..., Z_k = z_k)$$
$$= \frac{P(X = x_i, Z_1 = z_1, Z_2 = z_2, ..., Z_k = z_k)}{P(Z_1 = z_1, Z_2 = z_2, ..., Z_k)}$$
$$\simeq \frac{N(X = x_i, Z_1 = z_1, Z_2 = z_2, ..., Z_k = z_k)}{N(Z_1 = z_1, Z_2 = z_2, ..., Z_k)}$$

$N(X = x_i, Z_1 = z_1, Z_2 = z_2, ..., Z_k = z_k)$ denotes the number of training data cases, which has $X = x_i, Z_1 = z_1, Z_2 = z_2, ..., Z_k = z_k$. $N(Z_1 = z_1, Z_2 = z_2, ..., Z_k = z_k)$ denotes number of the data case that has $Z_1 = z_1, Z_2 = z_2, ..., Z_k = z_k$ in the total environment data.

C. Soft evidence of Bayesian Network

As described in Section III, our system has classified the geometrical features of the cells using K-mean clustering. However, if the geometrical features of each cell are similar, the class boundaries may be ambiguous for recognizing the class. If we use the classification results directly as evidences for the Bayesian network, the inference could be too sensitive to estimate the robot pose. Since the Bayesian network allows us to use *soft evidence*[25] for inference, we use the nodes O_{t+1} and O_t as soft evidences to cope with the uncertainties of the boundaries.

For example, we assume the node O_t has sensor data obtained by the robot in a cell i with an orientation m. We also assume the sensor measurement (denoted by P_{im}) is classified into a class n. The system calculates the distances between the P_{im} and a center point of each class, then sorts all of the distances. By normalization, the system converts the distances to beliefs or probabilities of the classification to the cell. The probabilities are the soft evidences of the observation node O_t.

D. Inference for Sensor Planning

Our goal is to select an optimal sensing action sequence for localization. The evaluation of the action sequence is performed based on the inference of the Bayesian network and a utility function (Eq. 10).

As shown in the following procedure, sensor planning for localization of the higher layer has four steps:

- Calculating joint probability of nodes
- Estimating possible actions

- Predicting possible states
- Sensor planning for localization

1) Joint probability of Nodes Calculation: Initially, we calculate the joint probability of the Bayesian network's nodes (Eq. 1). The joint probability of all the nodes is calculated as the following formulations.

$$\begin{aligned} & P(A, S_t, S_{t+1}, O_t, O_{t+1}) && (1) \\ \overset{\text{CR}}{=} \; & P(A) \times P(S_t|A) \times P(S_{t+1}|S_t, A) \times \\ & P(O_t|A, S_t, S_{t+1}) \times P(O_{t+1}|A, S_t, S_{t+1}, O_t) \\ \overset{\text{CI}}{=} \; & P(A) \times P(S_t|A) \times P(S_{t+1}|S_t, A) \times \\ & P(O_t|S_t, S_{t+1}) \times P(O_{t+1}|S_{t+1}) && (2) \end{aligned}$$

Based on the chain rule (CR) of probability and the conditional independence (CI) relationships between the nodes, the joint probability Eq. 1 can be rewritten as Eq. 2. In Eq. 2, the conditional probabilities are learned by the $Maximum\ Likelihood\ Estimation$ method which is described in section V-B.

2) Estimating Possible Actions: Using the joint probability we can estimate the conditional probability distribution of the node $action$ given the evidences of the observation nodes O_t and current state node S_t as Eq. 4.

$$\begin{aligned} P(A = a|\hat{s}_t, \hat{o}_t) \overset{\text{CI}}{=} \; & P(A|S_t = \hat{s}_t) && (3) \\ \overset{\text{Bayes}}{=} \; & \frac{P(A = a, S_t = \hat{s}_t)}{P(S_t = \hat{s}_t)} && (4) \end{aligned}$$

where:

$$\begin{aligned} P(A = a, S_t = \hat{s}_t) = & \sum_{(\hat{o}_t, \hat{o}_{t+1}, \hat{s}_{t+1})} P(S_{t+1} = \hat{s}_{t+1}, A = a, \\ & S_t = \hat{s}_t, O_t = \hat{o}_t, O_{t+1} = \hat{o}_{t+1}) \end{aligned}$$

and

$$\begin{aligned} P(S_t = \hat{s}_t) = & \sum_{(a, \hat{o}_t, \hat{o}_{t+1}, \hat{s}_{t+1})} P(S_{t+1} = \hat{s}_{t+1}, A = a, \\ & S_t = \hat{s}_t, O_t = \hat{o}_t, O_{t+1} = \hat{o}_{t+1}) \end{aligned}$$

In Eq. 3, a, $\hat{s}_t, \hat{o}_t$ and $\hat{o}_{t+1}$ are the evidences of the nodes $Action$, S_t, O_t and O_{t-1}, respectively. We define the possible actions in a list $ActionList = \{a_1^\star, ..., a_N^\star\}$. Each possible action $a_n^\star \in ActionList$ must satisfy the following condition:

$$P(A = a_n^\star|S_t = \hat{s}_t, O_t = \hat{o}_t) > 0 \quad (5)$$

3) Predicting Possible States: The goal of this step is to predict the next states when the robot takes a possible action $a_n^\star \in ActionList$. Initially, we calculate the conditional probability distribution of node S_{t+1} given the evidences of the nodes S_t, O_t and one of possible action $a_n^\star \in ActionList$ using the following formulations.

$$\begin{aligned} & P(S_{t+1} = \hat{s}_{t+1}|A = a_n^\star, \hat{s}_t, \hat{o}_t) && (6) \\ \overset{\text{Bayes}}{=} \; & \frac{P(S_{t+1} = \hat{s}_{t+1}, A = a_n^\star, \hat{s}_t, \hat{o}_t)}{P(A = a_n^\star, \hat{s}_t, \hat{o}_t)} \end{aligned}$$

where:

$$\begin{aligned} & P(A = a_n^\star, \hat{s}_t, \hat{o}_t) \\ = & \sum_{[\hat{s}_{t+1}, \hat{o}_{t+1}]} P(\hat{s}_{t+1}, A = a_n^\star, \hat{s}_t, \hat{o}_t, \hat{o}_{t+1}) \end{aligned}$$

and

$$\begin{aligned} & P(S_{t+1} = \hat{s}_{t+1}, A = a_n^\star, \hat{s}_t, \hat{o}_t) \\ = & \sum_{[\hat{o}_{t+1}]} P(\hat{s}_{t+1}, A = a_n^\star, \hat{s}_t, \hat{o}_t, \hat{o}_{t+1}) \end{aligned}$$

Based on the inference of Eq. 6, we can obtain the most-probable explanation $\hat{s}_{t+1}^\star$ that satisfies

$$\hat{s}_{t+1}^\star = argmax(P(\hat{s}_{t+1}|a_n^\star, \hat{s}_t, \hat{o}_t))$$

And the conditional probability of the most-probable explanation $\hat{s}_{t+1}^\star$ is calculated as the following formulation.

$$\begin{aligned} Prob(\hat{s}_{t+1}^\star)_{a_n^\star} &= P(\hat{s}_{t+1}^\star|a_n^\star, \hat{s}_t, \hat{o}_t) && (7) \\ &= max(P(\hat{s}_{t+1}|a_n^\star, \hat{s}_t, \hat{o}_t)) && (8) \end{aligned}$$

We define the most-probable explanation $\hat{s}_{t+1}^\star$ as the predicted next state, and $Prob(\hat{s}_{t+1}^\star)_{a_n^\star}$ is the probability of the predicted next state when the robot takes a possible action $a_n^\star$. Using the above formulations we can calculate all of $Prob(\hat{s}_{t+1}^\star)_{a_n^\star}$ which correspond to each $a_n^\star \in ActionList$.

4) Sensor Planning for Localization: Based on the $Prob(\hat{s}_{t+1}^\star)_{a^\star}$ and sensing cost, the mobile robot can find an optimal sensing action sequence for the global localization.

The system assumes that the white circles perform all of the possible action sequences from the robot's initial pose, and estimates the gross global localization probabilities of the white circles using the Bayesian network, when the actions have finished. We denote the average probability of all of the white circles to be '$AveBelief$', and '$Cost$' is defined by the distance that the circle moved. For example, when the robot takes a possible action $a_n^\star$, the $AveBelief$ is calculated in the following equation.

$$AveBelief = \frac{\sum_{k \in K} Prob(\hat{s}_{t+1}^\star)_{(a_n^\star, IS_k)}}{K} \quad (9)$$

We use $Prob(\hat{s}_{t+1}^\star)_{(a_n^\star, IS_k)}$ to denote the predicted global localization belief of the robot, when the $Initial\ state$ is IS_k, the Bayesian network is given soft evidences of the current state (S_t), observations (O_t, O_{t+1}), and robot takes a possible action $a_n^\star$. The calculation of $Prob(\hat{s}_{t+1}^\star)_{(a_n^\star, IS_k)}$ is described in detail in section V-D.

$$\begin{aligned} Utility &= (1 - \theta) \times AveBelief + \frac{\theta}{Cost} && (10) \\ where: & \quad 0 < \theta < 1 \end{aligned}$$

Based on Eq. 10, by taking into account the balance between $AveBelief$ and $cost$, the system calculates the '$Integrated\ Utility$', and selects an optimal action sequence for the mobile robot localization based on the $Integrated\ Utility$. The θ of Eq. 10 is a parameter for balancing the *AveBelief* and *Cost*.

E. Computational Complexity

The computational cost of our sensor planning algorithm is determined by two factors. One is the Bayesian network inference, and the other is sensor planning that uses that Bayesian network inference. The exact inference of a large-scale Bayesian network is an NP-hard problem[36]. Fortunately, in our research, the number of nodes of our Bayesian network does not expand with larger environment sizes. So the environment size does not affect the computational complexity of the Bayesian network inference. However, the computational time for finding the optimal sensing actions depends on the number of cells. The time complexity of an exhaustive search to depth d is $O(D^d \times E^d \times K^d)$, where E is the number of possible states, D is the number of available actions and K is the number of available observations. The time complexity of a traditional POMDP solution is also $O(D^d \times E^d \times K^d)$[37], and looks quite similar to our algorithm. However, since our algorithm decomposes the occupancy grid map into larger cells, and plans the sensor actions based on those decomposed larger cells, the search depth d for our sensor planning is smaller than the d of the occupancy grid based on POMDP planning. So the computational cost of our proposed hierarchical approach will be smaller than the cost of occupancy grid-based POMDP planning.

VI. Simulation Experiments

A. Particle Clustering

Our system does not perform sensor planning at all times. Sensor planning starts when the particles converge into clusters. The system checks the averages and variances of the weights of the particles in all cells and at every resampling cycle. When the average and the variance exceed certain thresholds, the system starts to perform particle clustering and sensor planning. The threshold for particle clustering is an experiential result. We did not attempt to learn the threshold from the particles this time, but we may consider it as a task for future study.

The *higher layer* starts sensor planning after the particle clustering. The possible locations of the robot are focused on the cells with particle clusters. The probabilities of the possible locations are evaluated by the sum of the particles' weights in the cluster. The initial status of the *higher layer* is illustrated in Fig. 1(a). The candidate robot poses (white circles) are shown in the centers of the cells that have clusters, and the orientation of the robot is obtained from the average of the particles in the cluster. The probabilities of the clusters are calculated by the sum of the particles' weights. The gradation of the cell color in Fig. 1(a) shows the probability of the robot pose. That is, the darker intensity corresponds to a higher probability of the global localization, and the brighter intensity corresponds to a lower probability.

B. Higher Layer Control for Sensor Planning

The particle filter performs filtering, predicting and smoothing the sensing data. However, it cannot control the actions of the mobile robot. If we only implement the particle filter for localization and do not consider action control, random sensing actions may cause slow convergence of the particles. Therefore, to solve this problem, the *higher layer* is necessary to improve the efficiency of the localization.

Action	Average Probability (AveBelief)
Go forward	0.8010
Turn left	0.6754
Turn right	0.8630
Go back	0.8652

TABLE I

Average probabilities of all of the particle clusters inferred by the BN when the system performed one action.

For example, in Fig. 1(a), there are nine white circles that correspond to nine possible robot poses, and each possible pose corresponds to an $Initial\ state$ (see Fig. 3(b)). Initially, the system assumes that all circles perform '$Step\ one$'. As shown in Fig. 3, in this step, the robot has four possible actions. The average probabilities of the nine circles (AveBelief) are shown in Table. I. We can see that the $AveBelief$ is the largest, if the robot takes the action $go\ back$. If the largest $AveBelief$ does not exceed a certain threshold, the system has to perform the '$Step\ two$' or more, until it finds an $AveBelief$ that is larger than the threshold. In our experiments, we used 0.9 for the threshold. If the system finds a $AveBelief$ that is larger than the threshold, an optimal action sequence will be selected by calculating the $Integrated\ Utility$ using the Eq. 10. The optimal action sequence must hold a largest $Integrated\ Utility$. Since, in the Table I, the largest $AveBelief$ does not exceed the threshold (0.9), the system has to perform $Step\ two$. The system calculates the $AveBelief$s of the nine circles when they have finished "$Step\ one$ and $Step\ two$"(see Fig. 3). The calculated results are shown in Table. II. We can see that the $AveBelief$ is the largest when the nine circles performed '$turn\ left$' and '$go\ forward$', and the largest $AveBelief$ (0.9460) exceeds the threshold (0.9). In Table II, we can find only one $AveBelief$ that exceeds the threshold. So the action sequence, "$turn\ left$ then $go\ forward$", will be the optimal action sequence for localization. The simulation results in Fig. 2 show that our sensor planning is effective for the global localization.

VII. Actual Robot Experiments

As shown in Fig.4, we conducted actual robot experiments on sensor planning for mobile robot localization. The sub-figures on the left in Fig.4 show the real scenario of these experiments. Two laser range sensors were mounted on the mobile robot (Pioneer III), and sensor planning and localization calculations were conducted using a laptop computer (IBM X41). The environment for the mobile robot was two large rooms and two small rooms, and we measured the real environment and built a map (right-side sub-figure in Fig.4) using the measured data.

Initially, the particles were generated based on uniform distribution (see Fig.4(a)). The particles of the environment map represent the possible poses of the mobile robot. When

Action 1	Action 2	Average Probability (AveBelief)
Go forward	go forward	0.8862
Go forward	turn left	0.8558
Go forward	turn right	0.8010
Turn left	go forward	0.9460
Turn left	turn left	0.6754
Turn left	turn right	0.7623
Turn right	go forward	0.8803
Turn right	turn left	0.8761
Turn right	turn right	0.8735
Go back	go forward	0.7704
Go back	turn left	0.8091
Go back	turn right	0.7704

TABLE II

AVERAGE PROBABILITIES OF ALL OF PARTICLE CLUSTERS INFERRED BY THE BN WHEN THE SYSTEM PERFORMED TWO ACTIONS.

the robot performed some random sensing actions, the particles converged into clusters (Fig.4(b)). Based on the method described in section VII-A, our system detected these particle clusters (see Fig.4(c)). The circles in the right-side sub-figure (Fig.4(c)) show the location of the particle clusters. As shown in Fig.4(c), when the robot faces an intersection, it has to determine an action for localization. The sensor planner of the higher layer is run at this stage. Through sensor planning by taking into account the trade-off between global localization beliefs and sensing cost, the system obtains the optimal sensing action sequence for localization (see Fig.4(d)). The particles converge into one cluster completely when the mobile robot completes the optimal sensing actions. Our experimental results demonstrate the effectiveness of our proposed system.

VIII. DISCUSSION AND CONCLUSION

A. Discussion of Complex Environment

In Section II, we have described methods for map building and smaller cell selection. Using these semiautomatic methods, we can build and decompose a more complicated map.

Our proposed sensor planning algorithm does not depend on the shape of the map and the objects. If the environment is more complex, the sensor data of every decomposed cell will be more ambiguous than the rectangular map of our system. However, in K-means classification, the classified results give us only the initial probabilities of the robot localization, and the ambiguity of the localization will be reduced by the *gross localization* step using the Bayesian network and the lower layer, i.e., *particle filter*. We use the K-means algorithm to classify the laser data, then the robot calculates the distance between the observed laser data and the center point of each classified class. The distances are normalized to probability, and the probability is the initial probability of the robot pose. The initial probability is input into the Bayesian network as soft evidence (see section V-C). Thus, while our proposed algorithm allows the classification results to have ambiguity, the ambiguity will be reduced and revised by the *gross localization* step using the Bayesian network. Since this ambiguous classification does not effect the sensor planning and since the map construction (map building and cell decomposition) is a semiautomatic process, our proposed algorithm can deal effectively with more complex environments.

B. Conclusion

This paper presented our hierarchical approach for solving sensor planning for the global localization of a mobile robot. The environment is represented by the integration of a large cell based topological map and an occupancy grid map (grid size: 0.05 m). Our system consists of two subsystems: a lower layer and a higher layer. The lower layer uses a particle filter to evaluate the posterior probability of localization based on the occupancy grid map. When the particles converge into clusters, the higher layer starts particle clustering and sensor planning to generate an optimal sensing action sequence for the localization based on the topological map. The higher layer uses a Bayesian network that represents contextual relationships between the geometrical features of the local environment, the sensing actions and the global localization beliefs. The sensor planning generates optimal action sequences by taking into account both the localization belief and the sensing cost. We demonstrated the effectiveness of our proposed approach in simulations and actual robot experiments. Further improvements in actual environments are planned for future study.

REFERENCES

[1] S. Thrun, "Probabilistic Algorithms in Robotics," *AI Magazine*, 21(4):93-109, 2000.

[2] S.Thrun, "Bayesian Landmark Learning for Mobile Robot Localization," *Machine Learning* 33, pp.41-76, 1998.

[3] N. Roy, W. Burgard, D. Fox, and S. Thrun, "Coastal navigation: Robot navigation under uncertainty in dynamic environments," *Proc. of IEEE Int. Conf. on Robotics and Automation*, 1999.

[4] S.Thrun, D.Fox, W.Burgard, and F.Dellaert, "Robust Monte Carlo Localization for Mobile Robots," *Artificial Intelligence*, 128(1-2), 2001.

[5] J. Borenstein, B. Everett, and L. Feng. "Navigating Mobile Robots: Systems and Techniques", A.K. Peter, Ltd., Wellesley, MA, 1996.

[6] L. Guibas, R. Motwani, and P. Raghavan. "The Robot Localization Problem", In K. Goldberg, D. Halperin, and J. C. Latombe, editors, Algorithmic Foundations of Robotics. A. K. Peters, 1995.

[7] J.A. Castellanos and J. D. Tardos. "Mobile robot localization and map building: a multisensor fusion approach", Kluwer Academic Publishers, 1999.

[8] J.S. Gutmann, W. Burgard, D. Fox, and K. Konolige, "An experimental comparison of localization methods". In Proc. International Conference on Intelligent Robots and Systems (IROS98), 1998.

[9] F. Dellaert, D. Fox, W. Burgard, S. Thrun, "Monte Carlo Localization for Mobile Robots," in Proc. ICRA 1999, pp.1322-1328.

[10] S.Koenig, R.G.Simmons, "Xavier: A Robot Navigation Architecture Based on Partially Observable Markov Decision Process Models," *Artificial Intelligence and Mobile Robot,* D.Kortenkamp, R.P.Bonasso, and R.Murphy, Eds: AAAI Press/The MIT Press, 1998, pp.91-122.

[11] R.Simmons, S.Koenig, "Probabilistic Robot Navigation in Partially Observable Environments," in Proc. 14th IJCAI, Montreal, Canada, August 1995, pp.1080-1087.

[12] S. Kristensen, "Sensor Planning with Bayesian Decision Analysis," *PhD thesis*, Faculty of Technology and Science, Aalborg University, Aalborg, Denmark, 1996.

[13] P. Jensfelt and S. Kristensen, "Active Global Localization for a Mobile Robot Using Multiple Hypothesis Tracking," *IEEE Transactions on Robotics and Automation*, Vol. 17, No. 5, pp.748-760, 2001.

[14] L. Romero, E. Morales and E. Sucar, "An Hybrid Approach to Solve the Global Localization Problem For Indoor Mobile Robots Considering Sensor's Perceptual Limitations", IJCAI'01.

[15] G. Dudek, K. Romanik, and S. White-sides. "Localizing a robot with minimum travel", SIAM J. Comput, 27(2):583-604, 1998.

[16] L. Romero, E. Morales, and E. Sucar. "A robust exploration and navigation approach for indoor mobile robots merging local and global strategies", In M. C. Monard and J.S. Sichman, editors, Advances in Artificial Intelligence, LNAI 1952. Springer, 2000.

[17] S. Thrun, A. Bucken, W. Burgar, et al. "Map learning and high-speed navigation in rhino". In D. Kortenkamp, R. P. Bonasso, and R Murphy, editors, Artificial Intelligence and Mobile Robots. AAAI Press/The MIT Press, 1998.

[18] D. Fox, W. Burgard, and S. Thrun, "Active Markov Localization for Mobile Robots," *Robotics and Autonomous Systems*, Vol.25, pp.195-207, 1998.

[19] N. Tomatis, I. Nourbakhsh, K. Arras and R. Siegwart, "A Hybrid Approach for Robust and Precise Mobile Robot Navigation with Compact Environment Modeling," *Proc. of IEEE Int. Conf. on Robotics & Automation*, pp.1111-1116, 2001.

[20] K. Basye, T. Dean, J. Kirman, and M. Lejter, "A Decision-Theoretic Approach to Planning, Perception, and Control," *IEEE Expert*, Vol.7, No.4, pp.58-65, 1992.

[21] A. Cassandra, L. P. Kaelbling, J. Kurien, "Acting under Uncertainty: Discrete Bayesian Models for Mobile-Robot Navigation," in Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, 1996.

[22] H. Zhou and S. Sakane, "Sensor Planning for Mobile Robot Localization using Bayesian Network Representation and Inference," *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp.440-446, 2002.

[23] H. Zhou and S. Sakane, "Learning Bayesian Network Structure from Environment and Sensor Planning for Mobile Robot Localization," *Proc. 2003 IEEE Int. Conf. on Multisensor Fusion and Integration for Intelligent System (MFI2003), pp.76-81, 2003*.

[24] J. Miura and Y. Shirai, "Vision-Motion Planning for a Mobile Robot considering Vision Uncertainty and Planning Cost," *Proc. of 15th Int. Joint Conf. on Artificial Intelligence*, pp.1194-1200, 1997.

[25] F. V. Jensen, "Bayesian Networks and Decision Graphs," *Springer*, 2001.

[26] J.C. Latombe, "Robot Motion Planning," *Kluwer Academic Publishers*, 1996.

[27] Richard O. Duda, Peter E. Hart and David G. Stork, "Pattern Classification, 2/Edition," *Published by John Wiley & Sons Inc*, 2001.

[28] J. Liu and R. Chen. "Sequential monte carlo methods for dynamic systems. Journal of the American Statistical Association", 93:1032-1044, 1998.

[29] M. Isard and A. Blake. "Condensation: conditional density propagation for visual tracking. International Journal of Computer Vision", 29(1):5-28, 1998. In press.

[30] A. Doucet, J.F.G. de Freitas, and N.J. Gordon, editors. "Sequential Monte Carlo Methods In Practice". Springer Verlag, New York, 2001.

[31] H. Asoh, Y. Motomura, I. Hara, S. Akaho, S. Hayamizu, and T. Matsui, "Combining Probabilistic Map and Dialog for Robust Life-long Office Navigation," *Proc. of Int. Conf. on Intelligent Robots and Systems (IROS'96)*, pp.880-885, 1996.

[32] D. Heckerman, "A Bayesian Approach to Learning Causal Networks," Technical Report MSR-TR-95-04, Microsoft Research, March, 1995.

[33] R.Rimey and C.Brown, "Control of Selective Perception using Bayes Nets and Decision Theory," *Int. Journal of Computer Vision*, Vol.12, pp.173-207, 1994.

[34] Immanuel Ashokaraj, A. Tsourdos, P. Silson, B. White, "Sensor Based Robot Localisation and Navigation: Using Interval Analysis and Unscented Kalman Filter", IEEE/RSJ International Conference on Intelligent Robots and Systems 2004 (pp. 7-12)

[35] Vandi Verma, Geoff Gordon, and Reid Simmons, "Efficient Monitoring for Planetary Rovers", Proceedings of the International Symposium on Artificial Intelligence and Robotics in Space (iSAIRAS), May, 2003.

[36] Cooper, G. F., "The Computational Complexity of Probabilistic Inference using Bayesian Belief Networks, Artificial Intelligence," Vol. 42, pp. 393-405, 1990.

[37] S. Thrun, W. Burgard, and D. Fox, "Probabilistic Robotics," MIT Press, Cambridge, MA, 2005.

[38] A.I. Mourikis and S.I. Roumeliotis, "Optimal Sensor Scheduling for Resource Constrained Localization of Mobile Robot Formations," IEEE Transactions on Robotics, 22(5), Oct. 2006, pp. 917-931.

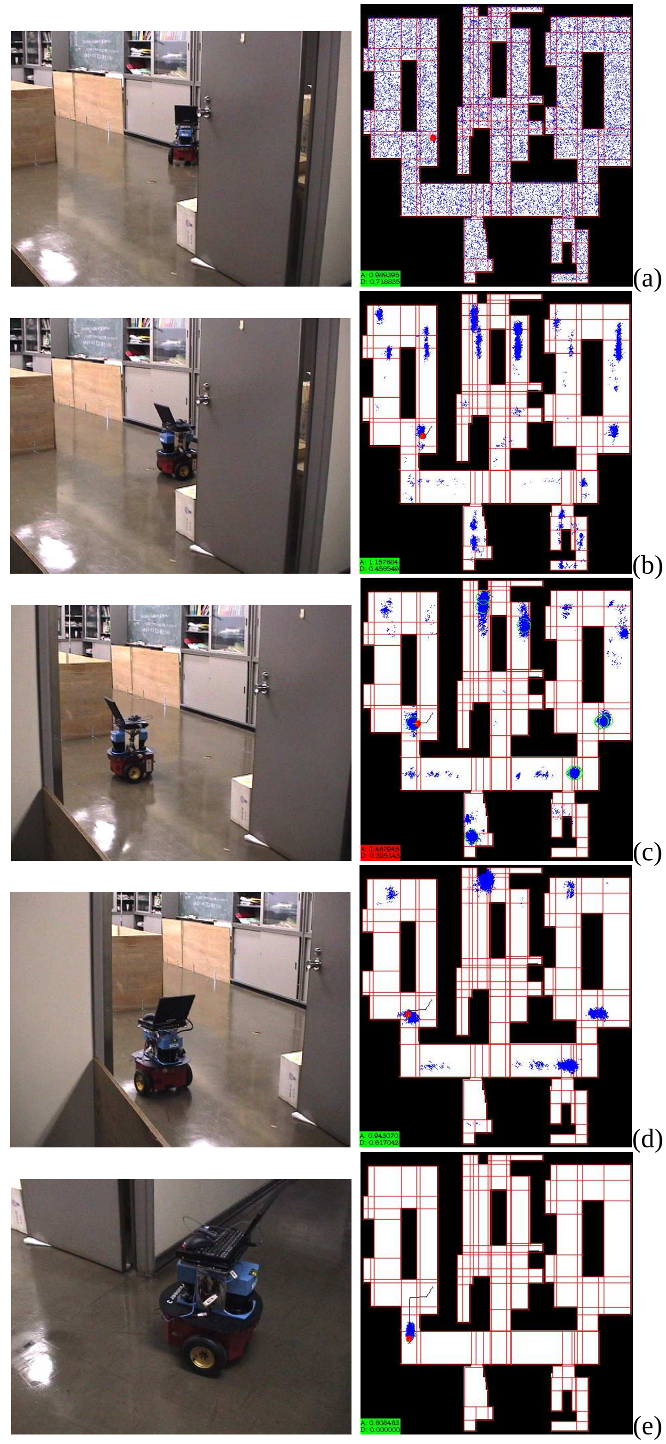

Fig. 4. Sensor planning for mobile robot localization using an actual robot.