P-SLAM: Simultaneous Localization and Mapping With Environmental-Structure Prediction

H. Jacky Chang, *Student Member, IEEE*, C. S. George Lee, *Fellow, IEEE*, Yung-Hsiang Lu, *Member, IEEE*, and Y. Charlie Hu, *Member, IEEE*

***Abstract*—Traditionally, simultaneous localization and mapping (SLAM) algorithms solve the localization and mapping problem in explored regions. This paper presents a prediction-based SLAM algorithm (called P-SLAM), which has an environmental-structure predictor to predict the structure inside an unexplored region (i.e., look-ahead mapping). The prediction process is based on the observation of the surroundings of an unexplored region and comparing it with the built map of explored regions. If a similar environment/structure is matched in the map of explored regions, a hypothesis is generated to indicate that a similar structure has been explored before. If the environment has repeated structures, the mobile robot can use the predicted structure as a virtual mapping, and decide whether or not to explore the unexplored region to save the exploration time. If the mobile robot decides to explore the unexplored region, a correct prediction can be used to speed up the SLAM process and build a more accurate map. We have also derived the Bayesian formulation of P-SLAM to show its compact recursive form for real-time operation. We have experimentally implemented the proposed P-SLAM on a Pioneer 3-DX mobile robot using a Rao–Blackwellized particle filter in real time. Computer simulations and experimental results validated the performance of the proposed P-SLAM and its effectiveness in indoor environments.**

***Index Terms*—Bayes procedures, environmental-structure prediction, simultaneous localization and mapping (SLAM).**

I. Introduction

SIMULTANEOUS localization and mapping (SLAM) is a fundamental and complex problem in mobile robotics research. Traditionally, in a SLAM problem, a mobile robot explores and senses an unknown region, constructs a map, and localizes itself in the map. Researchers encounter many problems, such as dead reckoning, noisy sensory measurements, failed data association, loop closing, and dynamic environments. They cause uncertainties and result in a search problem in high-dimensional spaces, which are composed of a large number of possible maps and robot trajectories. To make the search problem manageable, researchers have used various probabilistic techniques [1]–[12] to solve the SLAM problem in explored regions with little emphasis on unexplored regions.

This paper proposes a SLAM algorithm, called the prediction-based SLAM (P-SLAM) algorithm, which focuses on predicting the structure inside an unexplored region and using the predicted information for SLAM. The proposed P-SLAM has an environmental-structure predictor to generate hypotheses of unexplored regions before the mobile robot actually explores them. Specifically, P-SLAM focuses on unexplored regions that are in the neighborhood of the explored regions because they are the next exploration targets, and the mobile robot can use the information in these unexplored regions immediately. The prediction process is based on the observations of the surroundings of an unexplored region, comparing them with the built map of explored regions. If a similar structure is matched in the built map, a hypothesis is generated to indicate that a similar structure has been explored. If a mobile robot can predict the structure in an unexplored region correctly to a certain degree, then it can decide not to explore the unexplored region, and use the predicted structure as a virtual mapping to save the exploration time. If the robot decides to explore the unexplored region, then a correct prediction will reduce the search space to speed up the SLAM process and build a more accurate map. We have implemented the proposed P-SLAM with a Rao–Blackwellized particle filter (RBPF), and demonstrated that P-SLAM used fewer particles, resulting in a faster SLAM process.

The effectiveness of P-SLAM depends on the characteristics of an environment with repeated features, similar shapes, or symmetric structures. We usually can find an indoor environment with these characteristics (e.g., straight walls, right-angle corners, similar rooms, and symmetric layout in a building). With these characteristics, P-SLAM uses a built map to predict unexplored regions. If an environment is structurally irregular, then the environment is unpredictable. The consequence is that the built map will be difficult for use in predicting the environmental structure, and the mobile robot has no choice but to fully cover the unexplored regions to construct the map with a traditional SLAM algorithm.

A successful environmental-structure prediction can be used in many areas of mobile robotics research. In P-SLAM, we use it to save exploration time or construct a more accurate map and speed up the SLAM process. With correct predictions, a mobile robot can dynamically change its sensing frequency or exploration velocity to save energy. Our previous work has shown

Manuscript received November 28, 2005; revised June 17, 2006 and September 9, 2006. This paper was recommended for publication by Associate Editor J. Tardos and Editor L. Parker upon evaluation of the reviewers' comments. This work was supported in part by the National Science Foundation under Grant IIS-0329061. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. This paper was presented in part at the IEEE International Conference on Robotics and Automation, Orlando, FL, May 2006.

The authors are with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907-2035 USA (e-mail: chang26@purdue.edu; csglee@purdue.edu; yunglu@purdue.edu; ychu@purdue.edu).

Color versions of Figs. 3 and 13 are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2007.892230

that coordinating an ant-like, multirobot system by a simple environment estimation can explore an unknown area with energy and time efficiency [13]. Schroter *et al.* presented a detection and classification method of gateways in an office environment before a mobile robot fully observed these features [14]. This prediction scheme allows a mobile robot to obtain better localization and mapping results.

The mobile-robot exploration is another research area related to the unexplored-region prediction. During the exploration, a mobile robot needs to decide the next exploration target that provides the most information gain of an unexplored region. Therefore, the robot predicts and compares the information gain of different exploration targets. From this concept, the frontier-cell-based navigation method [15] is widely applied to the mobile-robot exploration. This method assumes that every frontier cell provides the same information gain, and the robot chooses a frontier cell as the next exploration target with the shortest traveling distance. One advantage of the frontier-cell-based navigation is that a mobile robot can explore every accessible region, but it suffers from longer exploration time due to rough information-gain predictions.

To improve the frontier-cell-based method, Grabowski *et al.* proposed "the next-best view" approach [16], [17], which provided a better information-gain prediction and thus speeded up the exploration process. For the multirobot exploration, it is even more urgent to correctly predict the information gain because it dominates the exploration time and energy. In [18] and [19], the authors employed a utility index to predict the information gain in target locations and used it to coordinate a team of mobile robots. In [20], frontier cells are clustered into groups and assigned to different mobile robots depending on the distance to targets and the predicted information gain. Although these exploration methods predict the information gain, they do not predict environmental structures inside unexplored regions and do not use the information to speed up the SLAM process.

To use the structure prediction in SLAM, we need to integrate the predictions in current SLAM algorithms. Most of the existing probabilistic approaches to the SLAM problem are all based on Bayesian filters [1], [21]. They use previous sensory information and motion commands to estimate mobile-robot poses (i.e., positions and orientations) and to build a map. The Bayesian formulation results in a compact recursive form suitable for real-time computation. In P-SLAM, we further use predicted maps for localization and mapping. Since P-SLAM differs from the traditional SLAM algorithms, it will be of interest to derive the Bayesian formulation of P-SLAM and compare it with the Bayesian formulation of traditional SLAMs. Like the Bayesian formulation of traditional SLAM algorithms, we show that the Bayesian formulation of P-SLAM also results in a compact recursive form, which incrementally solves the SLAM problem.

This paper is organized as follows. In Section II, we first formulate the prediction problem as a search problem and a similarity measurement problem, then we discuss how to use the predicted maps in SLAM. In Section III, we describe the environmental-structure predictor, including searching similar regions, similarity measurement, and hypothesis generation. In Section IV, we present the architecture and the Bayesian formulation of P-SLAM, and its implementation using a RBPF. We also discuss the worst-case scenarios and propose a recovery mechanism for P-SLAM. In Section V, computer simulation results and experimental results are presented and discussed. Conclusions are summarized in Section VI.

II. Problem Formulation

In this section, we formulate the environmental-structure prediction problem as two subproblems: how to perform the prediction and how to use predicted maps.

A. Prediction of an Unexplored Region With a Built Map

The central problem of the proposed P-SLAM is how to predict the structure of an unexplored region. To identify where the unexplored region is, P-SLAM adopts the occupancy-grid map, and this allows it to easily distinguish both the explored and unexplored regions. With the occupancy-grid map, the prediction process can be divided into four major steps:

1) locate a target frontier cell to predict;
2) collect structure information near the target region;
3) search for similar structures in the built map;
4) generate a hypothesis if a similarity match exists.

In the first step, a target cell f is selected for prediction. To prevent repeated or unnecessary predictions, P-SLAM adopts the following two criteria to select a target cell: the target is a frontier cell (i.e., a cell between an explored region and an unexplored region), and the cell is the next possible exploration goal. The first criterion guarantees that there is a nearby unexplored region, and the second criterion avoids unnecessary predictions because we want to use the predicted information immediately. If needed, the prediction process can be easily extended to predict several possible targets for coordinating the robot.

In the second step, P-SLAM collects the structure information near the target region. The information is defined as a region S_f surrounding the target cell f. The region S_f is formed within a range d, as shown in Fig. 1(a). The range d defines the surroundings and also determines the prediction range. Next, the third step is formulated as a search problem. Given f and S_f, search for a reference cell f' with the reference region $C_{f'}$ in the built map $\mathcal{M}$ such that

$$C_{f'} = \underset{C_{f'} \in \mathcal{M}, f' \neq f}{\arg\max} \left\{ \Psi\left(C_{f'}, S_f\right)\right\} \tag{1}$$

where $\Psi(\cdot)$ is a similarity measurement function. The corresponding f' and $C_{f'}$ are shown in Fig. 1(b). In the fourth step, if the similarity measurement is larger than a predefined threshold, then we say that a match exists and a hypothesis is generated. If several matches are above the threshold, the best match will be selected. As shown in Fig. 1(b), the generated hypothesis $H_{f'}$ corresponds to the unexplored region in Fig. 1(a). In Section III, we shall present the proposed environmental-structure predictor, including how to search for

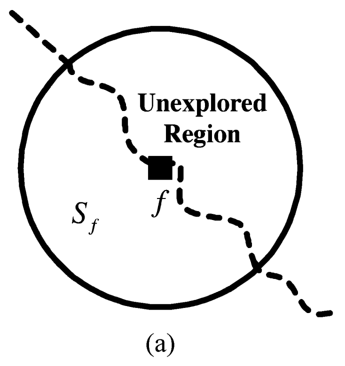

Fig. 1. Prediction of an unexplored region near a frontier cell. (a) A frontier cell f is marked as a bold square. The target region is inside the circle with a diameter d. The dashed line is the boundary between the unexplored and explored areas. The upper-right side of the boundary is indicated as the unexplored region. The surrounding region is S_f. (b) A reference cell in the built map is f', marked as a hollow square. The reference region $C_{f'}$ is inside the circle with a diameter d. The dashed line corresponds to the boundary of the target region in (a). The $H_{f'}$ is used to generate the hypothesis of the unexplored region in (a).

reference regions, how to design a similarity measurement function, and how to generate a hypothesis.

B. *Use of Predicted Maps*

With the predicted map of an unexplored region, the next question is how to use it in SLAM. For mapping purposes, P-SLAM uses the predicted map as a virtual map (look-ahead mapping). If the environment has repeated structures, the robot can decide not to explore the predicted regions to save the exploration time. For the SLAM purpose, P-SLAM uses the predicted map to update the particle weights when the robot first enters the predicted region, which is different from the traditional SLAM approaches. Most of the current SLAM algorithms process sensory readings and localize a robot in a built map. Thus, the correctness of the localization process heavily depends on the overlapping of sensor readings and the built map. If there is no overlapping, the robot only can use the noisy odometry readings to localize itself. In some cases, especially when the robot suddenly has a sharp turn or moves into a new unexplored region, the overlapping between the built map and the new sensor measurements could be very small. This may result in an inaccurate localization and an inconsistent map. The localization with an accurate prediction of an unexplored region eliminates this problem. Nevertheless, we shall be very careful to apply the prediction-based localization, because it may also induce extra noise due to inaccurate predictions.

The major difference between traditional SLAM algorithms and the proposed P-SLAM is the look-ahead mapping, as shown in Fig. 2. The details on how to merge the look-ahead mapping into a traditional SLAM will be discussed in Section IV.

III. Environmental-Structure Predictor

In this section, we introduce the proposed environmental-structure predictor. Since the prediction was formulated as a search problem and a similarity-measurement problem, the predictor consists of three major functions: search for a reference region; similarity measurement; and hypothesis

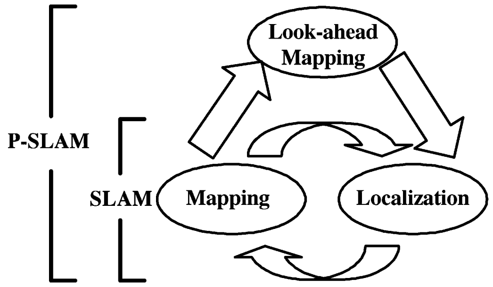

Fig. 2. Difference between the proposed P-SLAM and a traditional SLAM. P-SLAM has a look-ahead mapping, which is not included in a traditional SLAM algorithm.

generation. The details of these functions are presented in the following subsections.

A. *Search for a Reference Region*

Because the possible reference region could exist in any location in the built map (i.e., position and rotation), the search space could be enormously large and could overwhelm our algorithm. For instance, the total number of cells is 10^4 in a 10 m$\times$10 m region with 0.1 m resolution. If the comparison resolution in rotation is 1° and each comparison takes 1 ms, then the search time for one prediction is 3.6×10^3 s. Although the theoretical computational bound is proportional to the size of the map, it is impractical to realize it in real time, especially in an embedded mobile platform.

To overcome this searching problem, we use image-processing techniques in image registration to search for a reference region. The image registration is a process of establishing point-by-point correspondence between two images of a scene [22]. The image registration consists of feature extraction, feature correspondence, and calculating transformation matrices. Because we use the features for the search for a reference region, the computational cost now is proportional to the number of extracted features. In a typical indoor environment, the number of features is around hundreds, and usually can be handled by a computer in real time. Although we greatly scale down the computational requirement by using extracted features, the feature number still can be proportional to the size of the environment. One possible way to solve the problem is to set a fixed searching bound and obtain a suboptimal solution. Next, we shall briefly describe the steps of the image-registration method for this searching problem.

1) Feature Extraction: Lines and corners are selected as extracted features because they are the elementary representation of walls, hallways, or rooms. The line features are extracted from Hough transform, and the corner features are extracted from image gradients. It should be pointed out that the proposed method is not bounded to a corner or a line feature. As long as the extracted feature or the registration algorithm can provide the information to align the target and reference regions (i.e., obtain the homogenous transformation matrix), then the prediction process can proceed. Fig. 3 shows an example of the feature extraction from a partial mapping result.

2) Homogenous Transformation Matrix: From the extracted corner features, we obtain a pair of control points, where one

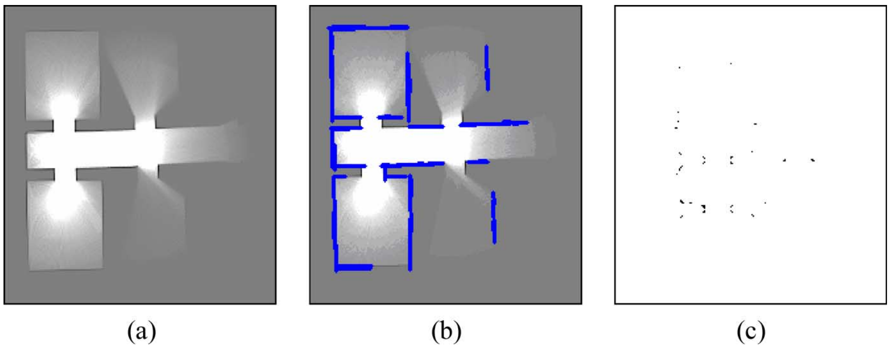

Fig. 3. Feature extraction. (a) The original occupancy-grid map. (b) The extracted line features. (c) The extracted corner features.

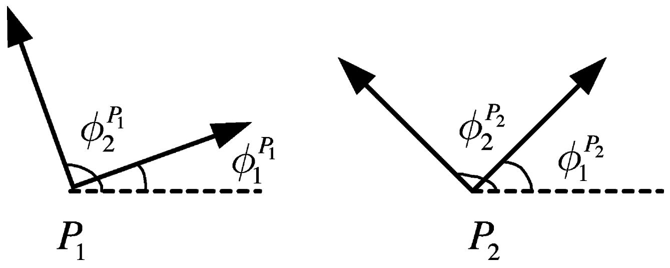

Fig. 4. P_1 and P_2 are control points extracted from corner features. The angles $\phi_1^{P_1}$, $\phi_2^{P_1}$, $\phi_1^{P_2}$, and $\phi_2^{P_2}$ are extracted from the line features.

point is in the target region and the other is in the reference region. With a pair of control points, we form 3×3 homogenous-coordinate-transformation matrices to align these two control points. For example, as shown in Fig. 4, let $P_1 = (p_x^1, p_y^1)$ and $P_2 = (p_x^2, p_y^2)$ be a pair of control points from extracted corner features. The θ represents an angle to align the orientation of these two control points, where θ is obtained from the extracted line features connected to the control points. Because a corner feature is formed by two line segments, a control point usually has two different angles for alignment. For instance, the control point P_1 has two angles, $\phi_1^{P_1}$ and $\phi_2^{P_1}$. For the control point P_2, it also has two angles, $\phi_1^{P_2}$ and $\phi_2^{P_2}$. Thus, we have four different θ's for alignment. For each θ, we can establish and use a homogenous-coordinate-transformation matrix Γ for alignment

$$\Gamma = \begin{bmatrix} \cos\theta & -\sin\theta & p_x^1 - p_x^2 \\ \sin\theta & \cos\theta & p_y^1 - p_y^2 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2}$$

We use the transformation matrix Γ and the pair of control points to obtain one possible reference cell $\tilde{f}'$ with aligned reference region $\tilde{C}_{\tilde{f}'}$ for similarity measurement. Note that the selected target cell may not be the control point. Therefore, we search for the nearest corner feature as the control point in a preset range. If there is no corner feature found, we will not start the prediction process. This prevents an unnecessary search process if there is no appropriate feature providing correspondence information.

B. Similarity Measurement

Once we obtain one possible reference region $\tilde{C}_{\tilde{f}'}$ and a target region S_f, the next step is to calculate the similarity of these two regions. Most of the existing similarity measurements, such as the sum of absolute differences, cross-correlation, and invariant moments do not fit our needs. This is because the major part of the map is identified as empty cells in an ordinary map. If we simply compare two maps by these methods, the results would be highly similar to each other due to the presence of many empty cells. Hence, instead of counting all the cells, we only consider occupied cells which are the structures of an environment. Consider the filters $I(\cdot)$ and $J(\cdot)$ for filtering out the occupied cells

$$I(i,j) = \begin{cases} 1, & \text{if } S_f(i,j) \text{ and } \tilde{C}_{\tilde{f}'}(i,j) \text{ are occupied} \\ 0, & \text{else} \end{cases} \tag{3}$$

$$J(R,i,j) = \begin{cases} 1, & \text{if } R(i,j) \text{ is occupied} \\ 0, & \text{else} \end{cases} \tag{4}$$

where $R(i,j)$ is an input region, and i, j are the position indexes of the cell. Using (3) and (4), the similarity measurement of these two regions, S_f, and $\tilde{C}_{\tilde{f}'}$ can be established as

$$\Psi(S_f, \tilde{C}_{\tilde{f}'}) = \frac{2 \times \sum_{i=1}^{d_s} \sum_{j=1}^{d_s} I(i,j)}{\sum_{i=1}^{d_s} \sum_{j=1}^{d_s} J(S_f,i,j) + \sum_{i=1}^{d_s} \sum_{j=1}^{d_s} J(\tilde{C}_{\tilde{f}'},i,j)} \tag{5}$$

where d_s is the size of the input region obtained from $d_s = \lceil d/\text{scale} \rceil$, and the scale is the resolution of the map.

C. Hypothesis Generation

With extracted features and homogenous transformation matrices, we search for every possible reference region and calculate their similarity measurements. We then obtain the reference region $C_{f'}$ with the highest similarity measurement value and use it to generate a hypothesis. If the similarity measurement of every possible reference region is below the threshold, the prediction process is terminated until the next target is determined. An example of hypothesis generation is shown in Fig. 5. The original built map is shown in Fig. 3(a). The selected target region and the found reference region are shown in Fig. 5(a) and (b), respectively. The calculated similarity measurement value is 0.53. To obtain the hypothesis, the transformation matrix is used to align the reference region and the target region. Then the hypothesis is generated from the reference region, which covers the unexplored parts in the target region. The generated hypothesis is shown in Fig. 5(c). In Fig. 5(d), the prediction result is "merged" into the built map for comparison with Fig. 3(a). In Fig. 5(d), the scantily explored

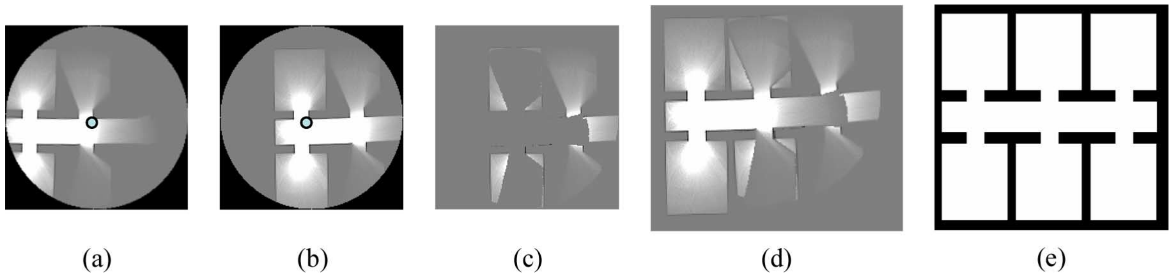

Fig. 5. (a) The selected target region. The small circle is the target control point. (b) The found reference region. The small circle is the reference control point. (c) The generated hypothesis for the selected target region. The hypothesis of the leftmost two rooms only covers the unexplored region, and thus the room centers are indicated in gray. (d) The merger of hypothesis and Fig. 3(a). The hypothesis is merged into the built map for comparing with Fig. 3(a). Since the scantily explored parts are still considered "explored regions," they are not covered by the hypothesis. (e) The simulation environment (23 m×20 m).

parts are not covered by the hypothesis because they are still considered "explored regions." As indicated in Section II-A, P-SLAM only predicts unexplored regions, therefore, only the unexplored parts are covered by the hypothesis. Compared with the simulated environment in Fig. 5(e), the prediction result shows that the environmental-structure predictor correctly predicts the structures in the unexplored rooms and part of the corridor.

To prevent low-quality predictions, a threshold value of similarity measurement is selected between 0.4 and 0.7 to warrant the quality prediction. If a robot does not have *a priori* information about the environment, the threshold value can be set conservatively to 0.7, which usually yields a good result in an indoor environment. This is based on our observations from extensive computer simulations. We also found that a similarity measurement value above 0.4 usually indicates that a reasonable-quality similar structure is found.

IV. P-SLAM

The proposed P-SLAM consists of two major components as shown in Fig. 6: the environmental-structure predictor (i.e., the look-ahead mapping); and a traditional SLAM algorithm working with generated hypotheses. The proposed P-SLAM differs from traditional SLAM algorithms in the use of predicted maps for localization and mapping. We shall derive the Bayesian formulation of P-SLAM and compare it with the Bayesian formulation of traditional SLAM algorithms to see the effect of predicated maps to traditional SLAM algorithms. Like the Bayesian formulation of traditional SLAM algorithms, the Bayesian formulation of P-SLAM also results in a compact recursive form, which incrementally solves the SLAM problem. We also combine our environmental-structure predictor with a RBPF to realize the proposed P-SLAM in our computer simulations and experimental work.

A. Bayesian Formulation of P-SLAM

We shall first review the Bayesian formulation of traditional SLAM algorithms and then extend it to derive the Bayesian formulation of P-SLAM. Consider the set of robot locations, motion commands, and observations, respectively, as follows:

$$X_k = \{x_1, x_2, \ldots, x_k\} = \{X_{k-1}, x_k\} \quad (6)$$

$$U_k = \{u_1, u_2, \ldots, u_k\} = \{U_{k-1}, u_k\} \quad (7)$$

$$Z_k = \{z_1, z_2, \ldots, z_k\} = \{Z_{k-1}, z_k\} \quad (8)$$

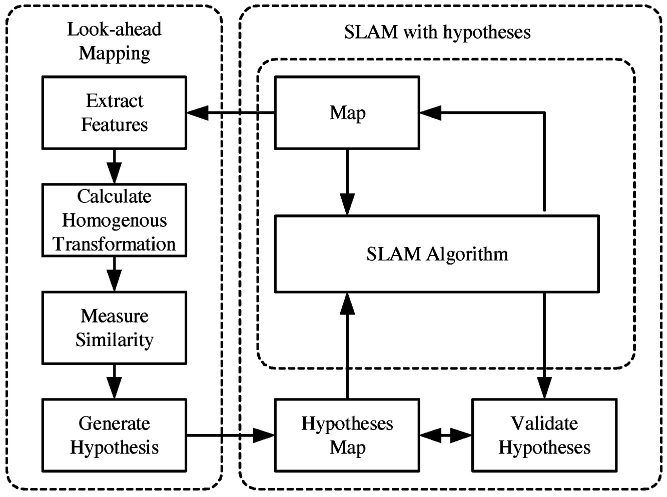

Fig. 6. P-SLAM structure.

where k denotes the discrete-time index, x_k denotes the robot pose, u_k denotes the motion command, and z_k denotes the observation. Given the motion commands U_k and the observations Z_k, the SLAM problem is to calculate the distribution as

$$P(X_k, \mathcal{M}_k | Z_k, U_k, x_0) \quad (9)$$

where $\mathcal{M}_k$ is the built map at time k and x_0 is the initial location of the robot. One possible way to estimate (9) is to consider all the possible maps and poses. However, the high-dimensional spaces in maps and poses make the SLAM problem intractable. To reduce the dimensionality of the search space, two assumptions are made: the motion model is Markov, and the environment is stationary. Then (9) can be derived from the Bayesian formulation in a recursive form [23]

$$\begin{aligned} P(x_k, \mathcal{M}_k | Z_k, U_k, x_0) \\ = \kappa \times P(z_k | x_k, \mathcal{M}_k) \times \Lambda \\ \Lambda = \int P(x_k | x_{k-1}, u_k) P(x_{k-1}, \mathcal{M}_{k-1} | Z_{k-1} U_{k-1}, x_0) dx_{k-1} \end{aligned} \quad (10)$$

where $P(x_k, \mathcal{M}_k | Z_k, U_k, x_0)$ is the posterior probability at time k, κ is a normalizing constant, $P(z_k | x_k, \mathcal{M}_k)$ is the sensor model, $P(x_k | x_{k-1}, u_k)$ is the motion model, and $P(x_{k-1}, \mathcal{M}_{k-1} | Z_{k-1}, U_{k-1}, x_0)$ is the posterior probability at time $k-1$. The Bayesian formulation incrementally solves the SLAM problem, making it solvable in real time.

Now, we can consider the SLAM problem with hypotheses (i.e., predicted maps). The set of hypotheses is expressed as

$$H_k = \{h_1, h_2, \ldots, h_k\} = \{H_{k-1}, h_k\}. \tag{11}$$

The posterior probability of P-SLAM at time k is expressed as

$$P(x_k, H_k, \mathcal{M}_k | Z_k, U_k, x_0). \tag{12}$$

To distinguish observations in explored and unexplored regions, z_k is decomposed into two parts

$$z_k = z_k^e + z_k^u \tag{13}$$

where z_k^e is part of the observation z_k overlapped with the built map $\mathcal{M}_{k-1}$ (i.e., in the explored region), and z_k^u is the observation in the unexplored region which overlaps with hypothesis h_k. From our assumption that the environment is stationary, the observation z_k^e in the explored region shall have been already represented in $\mathcal{M}_{k-1}$. For observation in the unexplored region, z_k^u will become a new part of the built map, therefore, $\mathcal{M}_k$ can be obtained as

$$\mathcal{M}_k = \mathcal{M}_{k-1} + z_k^u. \tag{14}$$

From (13) and (14), we factor out z_k^e in (12) and obtain

$$\begin{aligned} &P(x_k, H_k, \mathcal{M}_k | z_k^e, z_k^u, Z_{k-1}, U_k, x_0) \\ &\quad \propto P\left(z_k^e | H_k, \mathcal{M}_k, z_k^u, Z_{k-1}, U_k, x_0\right) \\ &\quad \times P\left(H_k, \mathcal{M}_{k-1}, x_k | z_k^u, Z_{k-1}, U_k, x_0\right). \end{aligned} \tag{15}$$

In (15), z_k^e is independent from H_k, Z_{k-1}. In addition, because of the Markov assumption, z_k and U_k are conditionally independent given x_k. Thus, the term $P(z_k^e | H_k, \mathcal{M}_k, z_k^u, x_k, Z_{k-1}, U_k, x_0)$ can be simplified as $P(z_k^e | \mathcal{M}_k, z_k^u, x_k)$ or $P(z_k^e | \mathcal{M}_k, x_k)$ using (14). Using the product rule, and applying that x_k and z_k^u are conditionally independent given U_k and x_0, the last term of (15) can be derived as

$$\begin{aligned} &P\left(h_k, H_{k-1}, \mathcal{M}_{k-1}, x_k | z_k^u, Z_{k-1}, U_k, x_0\right) \\ &\quad = P\left(h_k | H_{k-1}, \mathcal{M}_{k-1}, x_k, z_k^u, Z_{k-1}, U_k, x_0\right) \\ &\quad \times P(H_{k-1}, \mathcal{M}_{k-1}, x_k | Z_{k-1}, U_k, x_0). \end{aligned} \tag{16}$$

The term $P(h_k | H_{k-1}, \mathcal{M}_{k-1}, x_k, z_k^u, Z_{k-1}, U_k, x_0)$ depicts how likely it is that one would observe h_k. To further simplify this term, we observe that h_k is independent from H_{k-1}, Z_{k-1}, U_k, and x_0. Furthermore, although h_k is generated from $\mathcal{M}_{k-1}$, the similarity measurement or the hypothesis-generation process cannot actually tell us how likely we would observe h_k in an unexplored region. For the other two variables, z_k^u and x_k, the new observation z_k^u directly overlaps with h_k in the unexplored region, which depends on the current robot pose x_k. Hence, by assuming that h_k is independent from $\mathcal{M}_{k-1}$, (16) can be rewritten as

$$P\left(h_k | z_k^u, x_k\right) \times P(\mathcal{M}_{k-1}, H_{k-1}, x_k | Z_{k-1}, U_k, x_0). \tag{17}$$

Finally, using (15) and (17), $P(x_k, H_k, \mathcal{M}_k | Z_k, U_k, x_0)$ can be obtained as

$$\begin{aligned} &P(x_k, H_k, \mathcal{M}_k | Z_k, U_k, x_0) \\ &\quad \propto P\left(z_k^e | \mathcal{M}_k, x_k\right) \times P\left(h_k | z_k^u, x_k\right) \\ &\quad \times P(\mathcal{M}_{k-1}, H_{k-1}, x_k | Z_{k-1}, U_k, x_0). \end{aligned} \tag{18}$$

The recursive form of P-SLAM is derived from the total probability theory and expressed as

$$\begin{aligned} &P(x_k, H_k, \mathcal{M}_k | Z_k, U_k, x_0) \\ &\qquad = \kappa' P\left(z_k^e | \mathcal{M}_k, x_k\right) P\left(h_k | z_k^u, x_k\right) \Lambda' \\ &\Lambda' = \int P(x_k | x_{k-1}, u_k) P(x_{k-1}, H_{k-1}, \mathcal{M}_{k-1} | Z_{k-1}, \\ &\qquad U_{k-1}, x_0) dx_{k-1} \end{aligned} \tag{19}$$

where κ' is a normalizing constant.

Equation (19) is the P-SLAM Bayesian formulation. Comparing it with the traditional SLAM in (10), P-SLAM has an additional term $P(h_k | z_k^u, x_k)$. This term only exists when a robot explores a region covered by the hypothesis h_k. After the exploration, the hypothesis will no longer exist in the explored region. This means that $P(h_k | z_k^u, x_k)$ will only take effect when the robot enters a region the first time. The physical meaning of $P(h_k | z_k^u, x_k)$ is how likely it is that one can observe h_k when x_k and z_k^u are given. It also can be considered to shrink the search space of maps and poses; in other words, P-SLAM speeds up the SLAM process. However, it is not easy to obtain $P(h_k | z_k^u, x_k)$ directly, thus, we use the likelihood function $P(z_k^u | h_k, x_k)$ to approximate $P(h_k | z_k^u, x_k)$. By treating z_k^u as sensing data and h_k as a built map, we now can reuse the same sensor model in $P(z_k^e | \mathcal{M}_k, x_k)$ to obtain $P(z_k^u | h_k, x_k)$.

In summary, similar to (10), the P-SLAM Bayesian formulation is also in a recursive form. The result indicates that P-SLAM has the same structure as the original SLAM Bayesian formulation, and incrementally solves the SLAM problem. Equation (19) also shows that the predicted maps can be easily fused into any Bayesian filter technique by adding $P(h_k | z_k^u, x_k)$. For example, if we consider H_k to be the predicted information, such as an object, a feature, or a landmark, the formulation generalizes to using H_k in a landmark-based SLAM algorithm (e.g., extended-Kalman-filter-based SLAM).

B. P-SLAM With an RBPF

To implement the proposed P-SLAM, we need to consider two important requirements: real-time processing and a clear map representation. The real-time processing is usually required in an exploration task, and the clear map representation allows a robot to easily distinguish explored and unexplored regions. Here, we use an RBPF as the backbone of P-SLAM [24], [25]. The RBPF has been successfully applied in real-time SLAMs such as FastSLAM [10] and DP-SLAM [26]. In addition, the RBPF also works well with the occupancy-grid map representation [27], which satisfies the second requirement.

The RBPF has two components to compute the joint posterior distribution over possible maps and possible trajectories of a robot. The first component represents a distribution over possible trajectories, and this distribution is estimated by a particle filter in which each particle represents a possible trajectory. The second component represents the distribution over possible maps and hypotheses, which are conditioned on the trajectories given by the particle filter. One can consider that RBPF computes an individual map for each particle based on a known trajectory. Because RBPF only applies particles on the robot trajectory and not on the map estimation, the required number of particles can be greatly reduced. Moreover, the raw odometry reading is stabilized by a laser-scan matching algorithm before sending to the RBPF [28].

In the particle filter, each particle (n) is a tuple of $\langle x_k^{(n)}, m_k^{(n)}, h_k^{(n)}, w_k^{(n)} \rangle$, where $x_k^{(n)}$ is the robot pose at time k, $m_k^{(n)}$ is the constructed map, $h_k^{(n)}$ is the hypothesis map, $w_k^{(n)}$ is the particle weight for resampling, and T_k is the exploration target at time k. With action command u_{k-1} and observation z_k, the updates of each particle at time k are

$$\begin{aligned} x_k^{(n)} &= A\left(u_{k-1}, x_{k-1}^{(n)}\right) \\ m_k^{(n)} &= M\left(z_k, x_k^{(n)}\right) + m_{k-1}^{(n)} \\ h_k^{(n)} &= V\left(m_k^{(n)}, h_{k-1}^{(n)}\right) + H\left(m_k^{(\text{best})}, x_k^{(\text{best})}, x_k^{(n)}, T_k\right) \\ w_k^{(n)} &= S\left(z_k, x_k^{(n)}, m_{k-1}^{(n)}, h_k^{(n)}\right) w_{k-1}^{(n)} \end{aligned} \tag{20}$$

where $A(\cdot)$ and $M(\cdot)$ are the action and map models, respectively, $V(\cdot)$ is the hypothesis-validation model, and $H(\cdot)$ is the hypothesis-generation model, which is the look-ahead mapping. More details about models $A(\cdot)$ and $M(\cdot)$ can be found in [29].

For the hypothesis-validation model $V(\cdot)$, the validation is done by comparing $m_k^{(n)}$ and $h_{k-1}^{(n)}$, and removing explored parts from the hypothesis. For the hypothesis-generation model, $H(\cdot)$, it uses the built map from the current best particle, denoted by the superscript (best), and the next movement target T_k. Note that the hypothesis is only generated when a new target is selected, and the similarity measurement value is higher than the threshold. In addition, the generated hypothesis for each particle is also adjusted to the relative pose of the best particle. This avoids extensive prediction in each particle, since the map in each particle is similar to the others.

In [29], $S(\cdot)$ represents the sensor model $P(z_k^e|\mathcal{M}_{k-1}, x_k)$, which equals $P(z_k^e|\mathcal{M}_k, x_k)$, due to z_k^e and z_k^u being conditionally independent given x_k and $\mathcal{M}_{k-1}$. For $S(\cdot)$ in P-SLAM, we have an additional term $P(h_k|z_k^u, x_k)$ in (19), which is approximated by $P(z_k^u|h_k, x_k)$. Therefore, the weight in the nth particle is updated as

$$w_k^{(n)} = w_{k-1}^{(n)} P\left(z_k^e|x_k^{(n)}, m_{k-1}^{(n)}\right) P\left(z_k^u|h_k^{(n)}, x_k^{(n)}\right) \tag{21}$$

where we assume that the proposal distribution used in the RBPF is the action model as in FastSLAM [10]. Then $S(\cdot)$ can be rewritten as

$$\begin{aligned} S\left(z_k, x_k^{(n)}, m_{k-1}^{(n)}, h_k^{(n)}\right) = S_1&\left(z_k^e, x_k^{(n)}, m_{k-1}^{(n)}\right) \\ &\times S_2\left(z_k^u, x_k^{(n)}, h_k^{(n)}\right) \end{aligned} \tag{22}$$

where $S_1(\cdot)$ is the same sensor model in [29], $S_2(\cdot)$ is the same function as $S_1(\cdot)$, and z_k^u and $h_k^{(n)}$ are substituted for $m_{k-1}^{(n)}$ and z_k^e, respectively.

Different from the traditional RBPF-based SLAM, $S(\cdot)$ includes a hypothesis-observation model $S_2(\cdot)$. Hence, the hypothesis h_k assists the robot to update the importance weight of each particle, and the update takes effect in localization and mapping after the resampling process. If the hypothesis correctly represents the structure inside the unexplored region, then the hypothesis minimizes the search space of SLAM. Therefore, we can have fewer particles to estimate the distributions of the robot poses. Moreover, the number of particles also determines the required computational power in a particle filter. Hence, a correct hypothesis speeds up the SLAM process. In some cases, if the hypothesis is incorrect, this prediction results in an inaccurate weight updating of particles. To avoid this situation, we shall present a recovery mechanism to recover from incorrect predictions.

C. Worst-Case Scenarios and Recovery Mechanism

As mentioned in Section II-B, we need to be cautious of errors induced from the prediction process. We identify two different cases of worst-case scenarios in P-SLAM. The first one is that an environment is totally irregular without any common structure. This results in no knowledge that can be used from a built map, and no hypothesis can be generated. In this extreme case, the penalty is the computational cost of the look-ahead mapping of the P-SLAM algorithm. The mapping result will be exactly the same with the embedded SLAM algorithm, since we will only have $S_1(\cdot)$ in the RBPF.

The second worst-case scenario is that the prediction is inaccurate or incorrect and the robot uses it in (20). To prevent this situation from happening, we propose a recovery mechanism for P-SLAM. The key idea of the mechanism is that if a prediction is unlikely to an actual observation, we should not use the prediction in the RBPF. For example, if a robot predicts a room structure, but the real sensory data represents a hallway, then the robot should not use the prediction in the RBPF. To implement this idea, we propose a hit-ratio index to measure how far it is from a prediction to an observation. For the nth particle, the index $hr_k^{(n)}$ is considered to be a proportion of the intersection between $h_k^{(n)}$ and z_k^u as

$$hr_k^{(n)} = \frac{\left|h_k^{(n)} \bigcap z_k^u\right|}{|z_k^u|}. \tag{23}$$

We check $hr^{(n)}$ before we use the prediction in an RBPF. If $hr_k^{(n)} < hr_{\text{Th}}$ (i.e., a hit-ratio threshold) in all particles, we will not use the prediction in the RBPF. This prevents incorrect predictions from influencing the localization and mapping results.

V. Experimental Results

We performed extensive computer simulations as well as experiments on an ActivMedia Pioneer 3-DX mobile robot (P3-DX) to evaluate the performance of the proposed P-SLAM. The computer simulations verified the look-ahead mapping, the reduction of exploration time, and the correctness of generated hypotheses. The experiments demonstrated the improved mapping results in the same number of particles compared with traditional particle-filter-based SLAM algorithms in a real indoor environment.

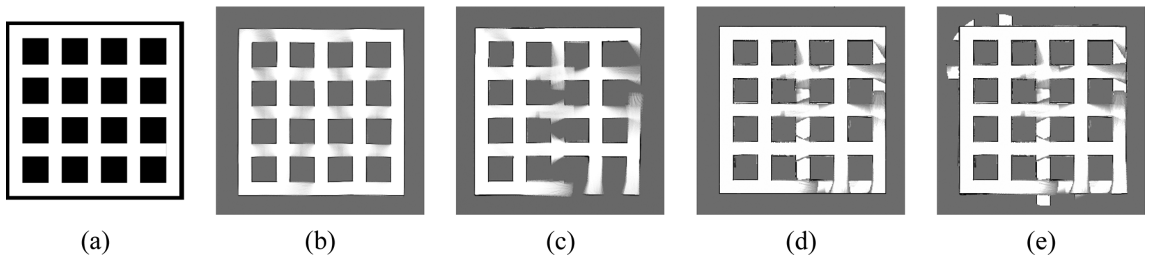
Fig. 7. (a) Square-maze environment used in computer simulations. (b) The mapping result of a full coverage with a scan-line exploration strategy. (c) The P-SLAM mapping result with designated robot movements. (d) The hypotheses are "merged" to the P-SLAM mapping result with the environmental boundary known. (e) Same as (d) when the hypotheses are "merged" to the P-SLAM mapping result but without knowing the environmental boundary.

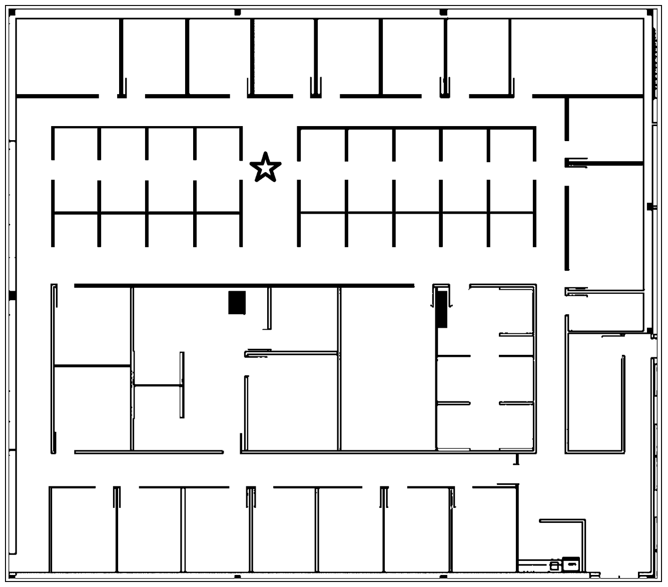
Fig. 8. Office environment used in the computer simulation. The star indicates the initial location of the mobile robot.

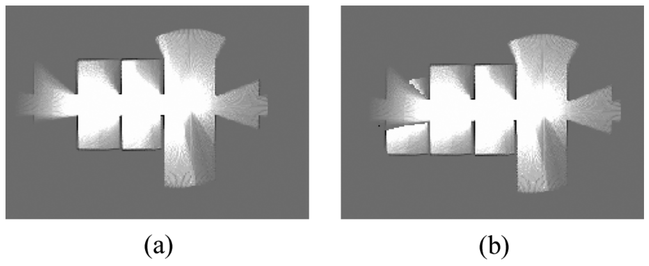
Fig. 9. (a) Built map when the first hypothesis was generated. (b) Built map with the first generated hypothesis.

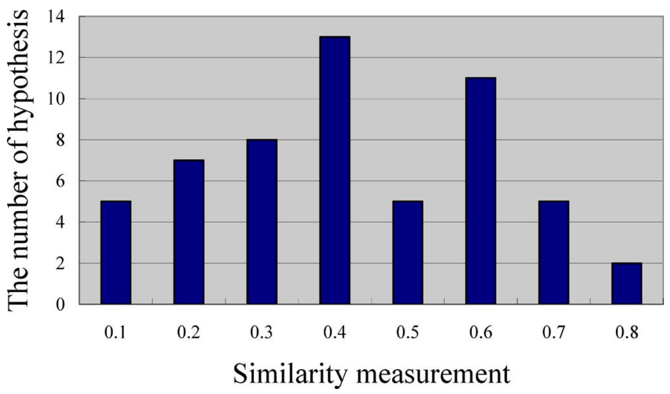

Fig. 10. Distribution of the number of hypotheses and the similarity measurement with the built map.

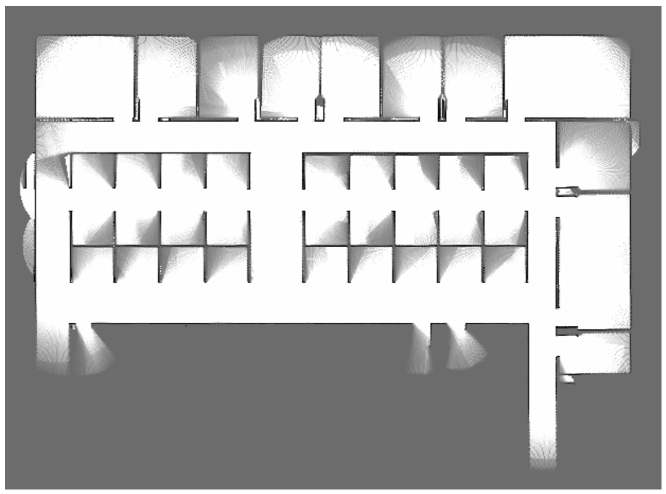
Fig. 11. Map generated by the proposed P-SLAM of the upper-office environment in Fig. 8. The predictions in the unexplored regions were also added in this map, such as the leftmost part, which cannot be reached by the robot.

A. Computer Simulations

To validate the effectiveness and correctness of the proposed P-SLAM, we have performed computer simulations on Player/Stage [30], which is a popular mobile-robot simulator and control software. We simulated a P3-DX mobile robot with a SICK LMS-200 laser ranger, running at a top speed of 0.2 m/s. For the mapping parameters, the resolution of the occupancy grid map was 0.1 m, the surrounding range d for prediction was 10 m, and the threshold of the similarity measurement was 0.6. A symmetric square maze was selected as a testing environment to demonstrate the look-ahead mapping. An indoor-office environment was selected to test the correctness of the predictions. Both of them have many common features/structures, which can be used in P-SLAM.

1) Simulation in a Symmetric Square Maze: In this simulation, we demonstrated the prediction capability of P-SLAM when the environment is highly regular. In addition, we showed that a robot can reduce the exploration time by leaving out some predicted regions from exploration. As shown in Fig. 7(a), the selected square maze has an area of 625 m^2 (25 m$\times$25 m) and has many repeated similar structures. The width of the square obstacles is 3.5 m, and the corridor width is 2.2 m.

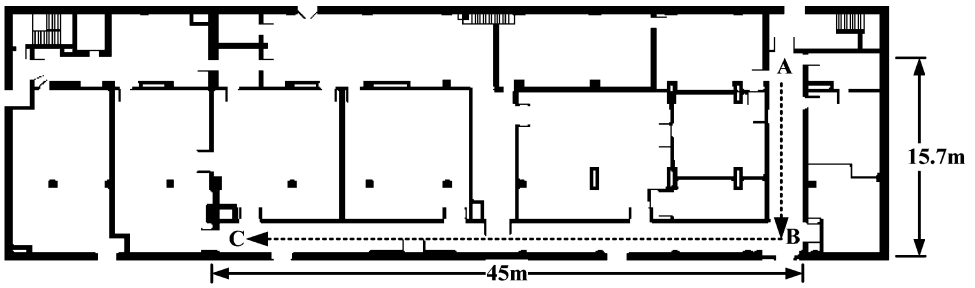

Fig. 12. Blueprint of the experimental environment. The mobile robot explored the L-shaped corridor starting from location A and ending at location C.

We performed a full coverage with a scan-line exploration strategy. It took 684 s to explore the environment, and the mapping result from a traditional SLAM is shown in Fig. 7(b). Instead of the full coverage, we navigated the robot movement to avoid exploring the predicted regions generated by the proposed P-SLAM. The exploration took 457 s, and the mapping result is shown in Fig. 7(c). In Fig. 7(d), the hypotheses are "merged" to the P-SLAM mapping result assuming that the environmental boundary is known. Fig. 7(d) shows that P-SLAM predicted the structures inside the unexplored regions reasonably well. Since we did not explore the predicted regions, the exploration time was reduced by 227 s, which is a 33% reduction compared with the full exploration. If the environmental boundary is unknown, then the mapping result is shown in Fig. 7(e), where some hypotheses are outside the square maze (e.g., upper-left corner). This is because the environmental-structure predictor did not know the boundary, and considered those regions as unexplored regions and generated hypotheses for them.

2) Simulation in an Office Environment: In order to verify the proposed P-SLAM, we simulated an indoor office environment to test the correctness of generated hypotheses from different kinds of structures. The simulated environment was a blueprint of a regular office environment selected from [31], as shown in Fig. 8. The robot was navigated to explore all accessible regions of the upper-half office environment.

During the exploration, 56 hypotheses were generated, and the covered area of the upper-half office environment was about 2000 m^2. Fig. 9(a) shows the map for generating the first hypothesis, and in Fig (9b), the hypothesis is "added" to the map and correctly predicts the structure in the unexplored region. The hypothesis also shows that the environmental-structure predictor is not designed for any specific structure such as the symmetric square maze in the previous simulation.

The total predicted size of the 56 hypotheses was 605.76 m^2, which was 30% of the covered area. To verify the correctness of the predicted hypotheses, we navigated the robot to explore the predicted regions. When we obtained the built map of the predicted area, we compared the map and the hypotheses by using the similarity measurement described in Section III-B. The similarity measurement values of these hypotheses and the final built map varied from 0.10 to 0.86, and the distribution is shown in Fig. 10. In addition, there were 36 hypotheses with similarity-measurement values higher than 0.4 and the average similarity-measurement value was 0.51, which indicated the predictions were very effective in this office environment. In addition, most of the successful predictions happened at the location of repeated structures such as the rooms or the hallways. As shown in the final map in Fig. 11, some of the predicted regions cannot be reached by the robot; however, these regions can be removed by postprocessing with a path-planning algorithm.

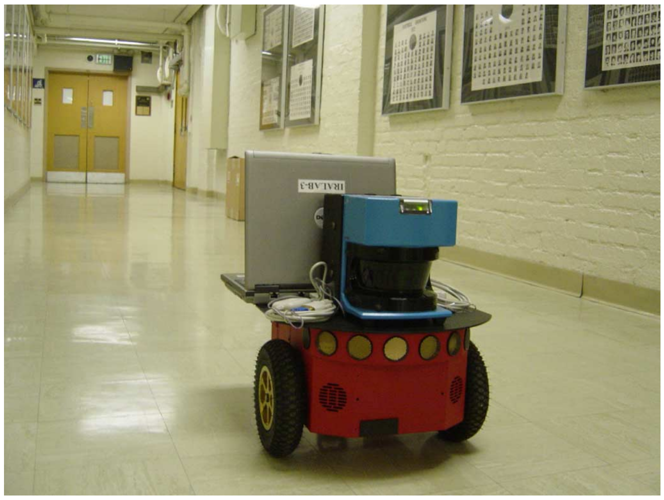

Fig. 13. Experimental environment and the Pioneer 3-DX robot with a SICK LMS-200 laser ranger. The boxes were placed to create corner features for prediction.

B. Experiments on a Pioneer 3-DX Robot

In this subsection, three experiments were performed to show how the proposed P-SLAM improved the accuracy of a map and speeded up the SLAM process. The first two experiments were conducted at a corridor and a floor inside an office building at Purdue University (West Lafayette, IN). The third experiment was performed by running a data set downloaded from the Robotics Data Set Repository (Radish) [31]. In these experiments, the mobile robot was equipped with a SICK LMS-200 laser ranger and obtained odometry readings from two 500 ticks encoders. A laptop computer with a 2.0 GHz Pentium-M processor and 2 GB memory handled all the required computations from the environmental-structure predictor and the RBPF. Since the prediction process was only executed when the next exploration target was selected, we used a different program thread to parallel process the prediction. This allowed the mobile robot to execute the prediction and the RBPF in real time.

1) Experiment in a Corridor Environment: The first testing environment was a corridor inside an office building, as shown in Fig. 12. The corridor only has very few corner features and similar segments of parallel lines, which were examined through

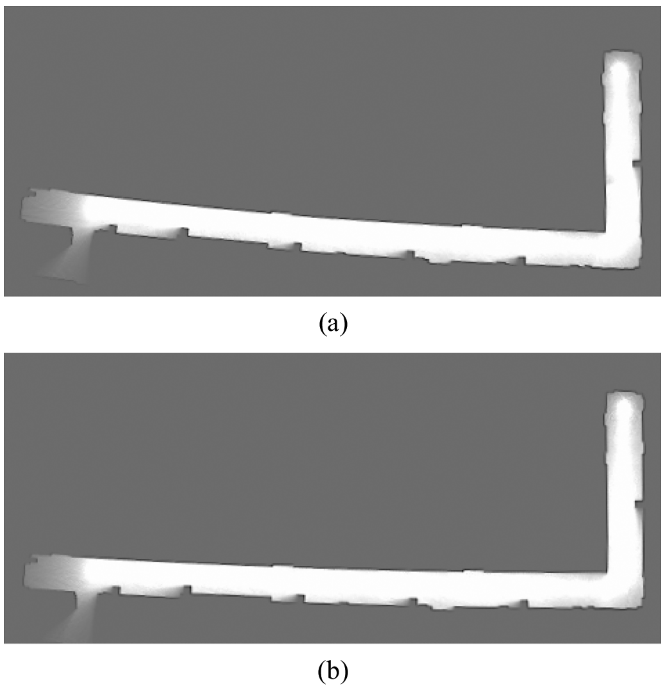

Fig. 14. (a) Final map generated by the RBPF with scan matching. (b) Final map generated by P-SLAM. The map of the corridor $\overline{BC}$ shows less bending compared with (a).

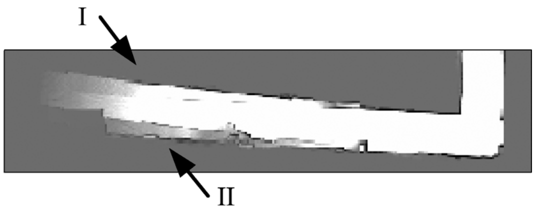

Fig. 15. The location marked by I indicates the built map without considering the hypothesis generated by P-SLAM. The location marked by II indicates the generated hypothesis, which looks like the shadow of the built map at location I.

static laser measurements with a resolution in 10 mm. This resulted in no similar structures being found due to the lack of corner features. Thus, we arbitrarily broke down the parallel lines into several segments by manually adding boxes in the corridor, as shown in Fig. 13. These boxes provided additional corner features, which were used as control points in P-SLAM. With these boxes, P-SLAM could detect the corridor with repeated similar structures.

Fig. 14(a) and (b) shows the final maps generated by the RBPF and by P-SLAM, respectively. Comparing these two final maps, P-SLAM provided a better result with less bending. The reason is that the map of corridor $\overline{AB}$ in Fig. 12 provides a good template for future predictions in corridor $\overline{BC}$. Hallway $\overline{AB}$ is a good template because $\overline{AB}$ is shorter, and the laser ranger can have additional laser readings at location B for scan matching. Fig. 15 shows how the generated hypothesis assists a traditional SLAM. The location marked by "I" indicates the built map without considering the hypothesis, and the location marked by "II" indicates the generated hypothesis. Here, the hypothesis affected the importance weight update of the particle filter and resulted in a better map. Since we need the template $\overline{AB}$ to correctly predict the corridor $\overline{BC}$, it reveals that a well-built map is critical to the proposed P-SLAM.

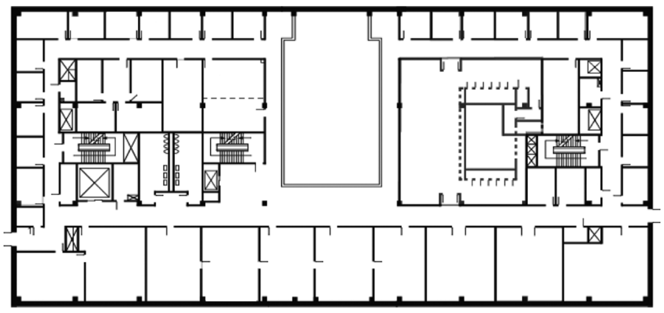

Fig. 16. Second-floor blueprint of the Materials and Electrical Engineering Building at Purdue University.

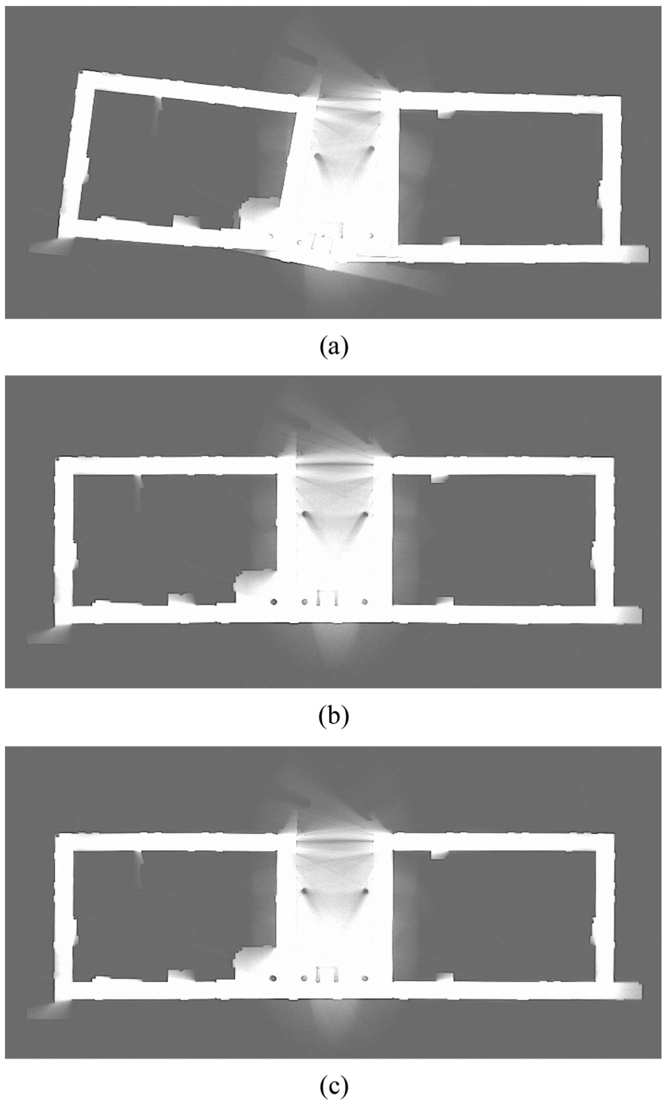

Fig. 17. (a) Online mapping result generated by a RBPF with 200 particles. (b) Online mapping result generated by P-SLAM with 200 particles. (c) Offline mapping result generated by a RBPF with 1000 particles.

2) Experiment in an Office Building Environment: The second testing environment was the second floor of the Materials and Electrical Engineering Building at Purdue University with an area of size 3000 m^2 (75 m$\times$40 m). The floor blueprint

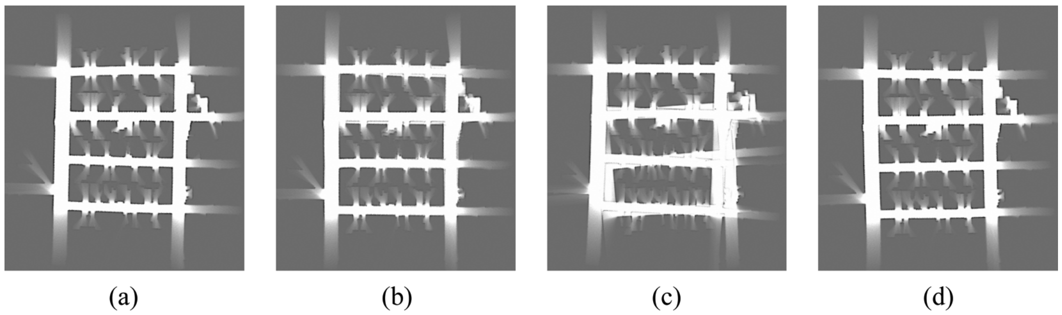

Fig. 18. Mapping results of the Intel Laboratory in Hillsboro, OR. (a) The final map generated by P-SLAM with 40 particles. $\mathrm{D_{KL}} = 1.55 \times 10^4$. (b) The final map generated by P-SLAM with 40 particles without the recovery mechanism. $\mathrm{D_{KL}} = 1.64 \times 10^4$. (c) The mapping result generated by RBPF with 40 particles. $\mathrm{D_{KL}} = 2.16 \times 10^5$. (d) The mapping result generated by RBPF with 200 particles.

is shown in Fig. 16. The threshold of the similarity measurement was set to 0.7, the number of particles is set to 200. Fig. 17(a) and (b) shows the final maps generated by an RBPF and by P-SLAM, respectively. In Fig. 17(a), the inconsistent global map was caused by incorrect angle estimations of the corners. The reason is that when the robot was turning, it injected some large rotation uncertainty into the system state. If the number of particles is not large enough to cover the spread of the system state, the particle filter will not be able to close the loop. On the other hand, in Fig. 17(b), P-SLAM successfully closed the loop due to correctly predicting the corner structures, which means that P-SLAM reduced the required number of particles. To see how many particles are needed to obtain a convergence result in an RBPF, we increased the number of particles by 100 each time. The offline simulation results showed that we needed 1000 particles to obtain the converged mapping result, which is shown in Fig. 17(c).

To quantitatively show the map improvement in accuracy, we used the Kullback–Leibler (KL) divergence to measure the distribution distance from one map to another. In (24), we used the map built by 1000 particles as the ground-truth distribution $P(i)$, and one built map as given distribution $Q(i)$. The KL divergence of the map in Fig. 17(a) is 3.06×10^4, and the KL divergence of the map Fig. 17(b) is 9.08×10^2 (the lower the better). The result shows the significant improvement made by P-SLAM

$$\mathrm{D_{KL}}(P\|Q) = \sum_i P(i) \log\left(\frac{P(i)}{Q(i)}\right). \qquad (24)$$

3) Experiment in a Nonempty Office Environment: In this experiment, we ran P-SLAM on a data set downloaded from the Radish. This data set is the raw sensory data acquired by a robot during a tour of part of the Intel Laboratory in Hillsboro, OR. The laboratory is an environment of multiple loops and has nonempty office rooms, which were selected to test the proposed recovery mechanism. The threshold of the similarity measurement was set to 0.7. In Fig. 18(a), P-SLAM with the proposed recovery mechanism performed well with 40 particles. Next, we deactivated the recovery mechanism, and ran the data set again. The mapping result is shown in Fig. 18(b), and part of the mapping results were inconsistent in the central corridors. The reason is that the office rooms were nonempty and irregular, some of the predictions for these office rooms were incorrect, and they influenced the RBPF. On the other hand, Fig. 18(c) shows the inconsistent mapping result of a traditional RBPF with 40 particles. To obtain a consistent mapping result, the traditional RBPF requires 200 particles, and the map is shown in Fig. 18(d). We also calculated the KL divergences to show the accuracy of these mapping results. We used the map built by 200 particles as the ground truth, and the KL divergences of Fig. 18(a), (b), and (c) are 1.55×10^4, 1.64×10^4, and 2.16×10^5, respectively. These KL divergences showed that P-SLAM with the recovery mechanism indeed gave a better result.

VI. Conclusions

In this paper, we have presented the proposed P-SLAM algorithm based on the environmental-structure prediction. With P-SLAM, a mobile robot can predict an unexplored region for a look-ahead mapping. This allows a mobile robot to reduce the exploration time and to speed up the SLAM process. In addition, we have also derived the Bayesian formulation of P-SLAM, which merges the predicted maps into the traditional SLAM Bayesian formulation. The Bayesian formulation generalizes how to use the predicted information in the SLAM problem. Computer simulations have shown that P-SLAM is very effective in an indoor environment. Finally, we implemented P-SLAM on a Pioneer 3-DX mobile robot in real time, and the experimental results showed that P-SLAM improved the mapping results.

For future work, we will address two possible improvements for the prediction process. First, one can consider using a set of control points to obtain a more accurate transformation matrix. The matrix will improve the accuracy of the predictions when we align a reference region and a target region. The second possible improvement is an object-based prediction. Instead of using an occupancy grid map, the object-based prediction estimates a structure by recognizing objects such as doors, chairs, or rooms. To perform the object-based prediction, a robot requires a database containing the structure information of these objects. The database can be given before an exploration task begins, or the robot can build the database online during the exploration task. Finally, for the improvement of the robot exploration, current frontier-cell-based exploration strategies can be incorporated into the predicted maps to estimate more accurate information gain. This will allow a robot or a team of robots to explore an environment more efficiently.

References

[1] S. Thrun, "Robotic mapping: A survey," in *Exploring Artificial Intelligence in the New Millenium*, G. Lakemeyer and B. Nebel, Eds.. San Mateo, CA: Morgan Kaufmann, 2002.

[2] M. G. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba, "A solution to the simultaneous localization and map building (SLAM) problem," *IEEE Trans. Robot. Autom.*, vol. 17, no. 3, pp. 229–241, Jun. 2001.

[3] R. Madhavan, K. Fregene, and L. E. Parker, "Terrain aided distributed heterogeneous multirobot localization and mapping," *Auton. Robots*, vol. 17, pp. 23–39, 2004.

[4] H. Feder, J. Leonard, and C. Smith, "Adaptive mobile robot navigation and mapping," *Int. J. Robot. Res.*, vol. 18, no. 7, pp. 650–668, Jul. 1999.

[5] S. Thrun, D. Koller, Z. Ghahramani, H. Durrant-Whyte, and A. Ng, "Simultaneous mapping and localization with sparse extended information filters: Theory and initial results," in *Proc. 5th Int. Workshop Algorithmic Found. Robot.*, J.-D. Boissonnat, J. Burdick, K. Goldberg, and S. Hutchinson, Eds., Nice, France, 2002, pp. 363–380.

[6] J. Leonard and H. Durrant-Whyte, "Simultaneous map building and localization for an autonomous mobile robot," in *Proc. IEEE Int. Workshop Intell. Robots Syst.*, 1991, pp. 1442–1447.

[7] J. Neira and J. Tardos, "Data association in stochastic mapping using the joint compatibility test," *IEEE Trans. Robot. Autom.*, vol. 17, no. 6, pp. 890–897, Dec. 2001.

[8] D. F. Wolf and G. S. Sukhatme, "Mobile robot simultaneous localization and mapping in dynamic environments," *Auton. Robots*, vol. 19, no. 1, pp. 53–65, Jul. 2005.

[9] H. J. Chang, C. S. G. Lee, Y.-H. Lu, and Y. C. Hu, "A computational efficient SLAM algorithm based on logarithmic-map paritSitioning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2004, pp. 1041–1046.

[10] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," in *Proc. AAAI Nat. Conf. Artif. Intell.*, Edmonton, AB, Canada, 2002, pp. 593–598.

[11] B. Stewart, J. Ko, D. Fox, and K. Konolige, "The revisiting problem in mobile robot map building: A hierarchical Bayesian approach," in *Proc. 19th Annu. Conf. Uncertainty Artif. Intell.*, 2003, pp. 551–558.

[12] A. Ranganathan, E. Menegatti, and F. Dellaert, "Bayesian inference in the space of topological maps," *IEEE Trans. Robot. Autom.*, vol. 22, no. 1, pp. 92–107, Feb. 2006.

[13] H. J. Chang, C. S. G. Lee, Y.-H. Lu, and Y. C. Hu, "Energy-time-efficient adaptive dispatching algorithms for ant-like robot systems," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2004, vol. 4, pp. 3294–3299.

[14] D. Schroter, T. Weber, M. Beetz, and B. Radig, "Detection and classification of gateways for the acquisition of structured robot maps," in *Proc. 26th Pattern Recog. Symp.*, Aug. 2004, pp. 553–561.

[15] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proc. IEEE Int. Symp. Comput. Intell. Robot. Autom.*, 1997, pp. 146–151.

[16] R. Grabowski, P. Khosla, and H. Choset, "Autonomous exploration via regions of interest," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2003, pp. 1691–1696.

[17] H. Gonzalez-Banos, E. Mao, J. Latombe, T. Murali, and A. Efrat, "Planning robot motion strategies for efficient model construction," in *Proc. 9th Int. Symp. Robot. Res.*, 2000, pp. 345–352.

[18] R. Simmons, D. Apfelbaum, W. Burgard, D. Fox, M. Moors, S. Thrun, and H. Younes, "Coordination for multi-robot exploration and mapping," in *Proc. AAAI Nat. Conf. Artif. Intel.*, Austin, TX, 2000, pp. 852–858.

[19] W. Burgard, M. Moors, C. Stachniss, and F. Schneider, "Coordinated multi-robot exploration," *IEEE Trans. Robot.*, vol. 21, no. 3, pp. 376–386, Jun. 2005.

[20] A. Solanas and M. Garcia, "Coordinated multi-robot exploration through unsupervised clustering of unknown space," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2004, pp. 852–858.

[21] C.-C. Wang, C. Thorpe, and S. Thrun, "Online simultaneous localization and mapping with detection and tracking of moving objects: Theory and results from a ground vehicle in crowded urban areas," in *Proc. IEEE Int. Conf. Robot. Autom.*, Sep. 2003, pp. 842–849.

[22] A. A. Goshtasby, *2-D and 3-D Image Registration for Medical, Remote Sensing, and Industrial Applications*. New York: Wiley, 2005.

[23] S. Thrun, M. Beetz, M. Bennewitz, W. Burgard, A. Cremers, F. Dellaert, D. Fox, D. Hähnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz, "Probabilistic algorithms and the interactive museum tour-guide robot minerva," *Int. J. Robot. Res.*, vol. 19, no. 11, pp. 972–999, 2000.

[24] K. P. Murphy, "Bayesian map learning in dynamic environments," in *Advances in Neural Information Processing Systems*. Cambridge, MA: MIT Press, 2000, pp. 1015–1021.

[25] A. Doucet, J. de Freitas, K. Murphy, and S. Russel, "Rao-Blackwellized particle filtering for dynamic Bayesian networks," in *Proc. Conf. Uncertainty Artif. Intell.*, 2000, pp. 499–516.

[26] A. Eliazar and R. Parr, "DP-SLAM 2.0," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2004, vol. 2, pp. 1314–1320.

[27] G. Grisetti, C. Stachniss, and W. Burgard, "Improving grid-based SLAM with Rao-Blackwellized particle filters by adaptive proposals and selective resampling," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2005, pp. 2443–2448.

[28] D. Hähnel, W. Burgard, D. Fox, and S. Thrun, "An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2003, pp. 27–31.

[29] A. Howard, "Multi-robot simultaneous localization and mapping using particle filters," in *Proc. Robot.: Sci. Syst.*, Cambridge, MA, Jun. 2005, pp. 201–208.

[30] R. T. Vaughan, B. P. Gerkey, and A. Howard, "On device abstractions for portable, reusable robot code," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2003, pp. 2121–2427.

[31] A. Howard and N. Roy, The Robotics Data Set Repository (Radish) 2003 [Online]. Available: http://www.radish.sourceforge.net/

H. Jacky Chang (S'04) is working toward the Ph.D. degree in the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN. He received the M.S. degree from the Department of Electrical and Control Engineering and the B.S. degree from the Department of Control Engineering, both from the National Chiao Tung University, Hsinchu, Taiwan, R.O.C.

His research interests include simultaneous localization and mapping, multirobot systems, and sensor fusion.

C. S. George Lee (S'71–M'78–SM'86–F'93) received the B.S. and M.S. degrees in electrical engineering from Washington State University, Pullman, in 1973 and 1974, respectively, and the Ph.D. degree from Purdue University, West Lafayette, IN, in 1978.

He is a Professor of Electrical and Computer Engineering at Purdue University, West Lafayette, Indiana. Currently, he is serving as Program Director for the Robotics Program at the National Science Foundation (NSF). His current research focuses on humanoid robots, robotic SLAM, and neuro-fuzzy systems. He has authored or co-authored over 150 publications in these areas, in addition to two graduate textbooks: *Robotics: Control, Sensing, Vision, and Intelligence* (New York: McGraw-Hill, 1986) and *Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems* (Englewood Cliffs, NJ: Prentice-Hall, 1996), and 20 book chapters.

Dr. Lee was an IEEE Computer Society Distinguished Visitor in 1983–1986, and the Organizer and Chairman of the 1988 NATO Advanced Research Workshop on Sensor-Based Robots: Algorithms and Architectures. He had also served as Secretary and Vice-President for Technical Affairs of the IEEE Robotics and Automation Society in 1988–1990 and 1990–1995, respectively. He was General Chair of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems in Las Vegas, NV, and was Program Chair of the 1996 IEEE International Conference on Robotics and Automation in Minneapolis, MN, and the 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems in Victoria, BC, Canada. He served as General Co-Chair of the 2006 IEEE International Conference on Robotics and Automation, held in Orlando, FL, in May 2006. He is a recipient of The IEEE Third Millennium Medal Award and the Distinguished Service Award from the IEEE Robotics and Automation Society.

Yung-Hsiang Lu (S'90–M'03) received the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA, in 2002.

He currently is an Assistant Professor with the School of Electrical and Computer Engineering, and by courtesy, the Department of Computer Science, at Purdue University, West Lafayette, IN. His research focuses on power management and energy conservation for computer systems, mobile robots, sensor networks, and image processing.

In 2004, Dr. Lu received an NSF Career Award for studying energy conservation by operating systems.

Y. Charlie Hu (S'91–M'98) received the M.S. and M.Phil. degrees from Yale University, New Haven, CT, in 1992, and the Ph.D. degree in computer science from Harvard University, Cambridge, MA, in 1997.

Currently, he is an Assistant Professor of Electrical and Computer Engineering and Computer Science, Purdue University, West Lafayette, IN. From 1997 to 2001, he was a Research Scientist with Rice University, Houston, TX. His research interests include operating systems, distributed systems, networking, and parallel computing. He has published more than 65 papers in these areas.

Dr. Hu received the NSF CAREER Award in 2003. He served as a TPC Vice Chair for the 2004 International Conference on Parallel Processing (ICPP-04), and a Co-founder and TPC Co-Chair for the First International Workshop on Mobile Peer-to-Peer Computing (MP2P'04). Dr. Hu is a member of USENIX and ACM.