Received: 23 February 2018 | Revised: 2 May 2018 | Accepted: 14 May 2018

DOI: 10.4218/etrij.2018-0041

SPECIAL ISSUE

WILEY ETRI Journal

Incremental hierarchical roadmap construction for efficient path planning

Byungjae Park[1] | **Jinwoo Choi**[2] | **Wan Kyun Chung**[3]

[1]SW·Contents Research Laboratory, Electronics and Telecommunications Research Institute, Daejeon, Rep. of Korea.

[2]Marine Robotics Laboratory, Korea Research Institute of Ships and Ocean Engineering, Daejeon, Rep. of Korea.

[3]Department of Mechanical Engineering, POSTECH, Pohang, Rep. of Korea.

Correspondence
Byungjae Park, SW·Contents Research Laboratory, Electronics and Telecommunications Research Institute, Daejeon, Rep. of Korea.
Email: bjp@etri.re.kr

Funding information
Institute for Information and Communications Technology Promotion, Grant/Award Number: 2017-0-00067; Korea Research Institute of Ships and Ocean Engineering, Grant/Award Number: PES9390; National Research Foundation of Korea, Grant/Award Number: 2011-0030075.

This paper proposes a hierarchical roadmap (HRM) and its construction process to efficiently represent navigable areas in an indoor environment. HRM is adopted to solve the path-planning problems of mobile robots in indoor environments. HRM has a multi-layered graphical structure that enables it to abstract and cover navigable areas using a smaller number of nodes and edges than a probabilistic roadmap. During the incremental process of constructing HRM, information on navigable areas is abstracted using a sonar gridmap when the mobile robot navigates an unexplored area. The HRM-based planner efficiently searches for paths to answer queries by reducing the search space size using the multi-layered graphical structure. The benefits of the proposed HRM are experimentally verified in real indoor environments.

KEYWORDS
environment representation, gridmap, mobile robot, path planning, probabilistic roadmap

1 | INTRODUCTION

Path planning is essential for mobile robots to perform various tasks. A path planner uses navigable area information about an environment to search for appropriate paths. These areas can be encoded using either a grid- or graph-based representation. Grid-based representation is commonly used for planning in a bounded two-dimensional (2D) indoor environment [1], and is useful for seeking safe and efficient paths because the navigable areas can be regularly divided and completely covered by rectangular grid cells. Besides, the navigable areas can be easily updated when a mobile robot acquires new sensor data. However, the computational complexity of a path planner using a grid-based representation may become too high when it is used in large environments or when the mobile robot has limited computational power and time.

A graph-based representation, such as a probabilistic roadmap (PRM) or a rapidly exploring random tree (RRT), is an alternative approach that requires less computation for a path planner in large environments [2,3]. A PRM is a single-layered undirected graphical structure that abstracts navigable areas efficiently, making it practical for a PRM-based path planner to answer multiple queries in less time by reducing the size of the search space. One node in a PRM corresponds to one unoccupied region of the environment, and one edge represents the connection between two adjacent nodes. A mobile robot can navigate between two nodes without colliding with obstacles if an edge is present between these nodes. However, there are limitations regarding the use of PRMs. When a mobile robot explores an unvisited area, a PRM must be expanded or reconstructed to abstract the navigable areas. Most PRM construction methods do not consider this issue because PRMs are usually applied to search for collision-free paths in a bounded high-dimensional space. Although a generalized Voronoi graph (GVG) can be used to construct a PRM incrementally, the GVG construction method is mainly suited to corridor-like environments [4]. Furthermore, a PRM cannot completely cover and evenly divide the navigable areas in an environment because most of the processes used to construct a PRM adopt random sampling. During the construction process of a PRM, nodes in unoccupied regions of the environment are generated through random sampling, and this causes the PRM to have unevenly distributed nodes. Therefore, it cannot cover the navigable areas completely, and thus a PRM-based planner may fail to find an appropriate path or may result in finding an inefficient path. Although such incomplete coverage can be mitigated using multiple nodes, this inevitably increases the computational complexity.

By extending our preliminary work [5], to overcome the limitations of a PRM, in this paper, we propose a hierarchical roadmap (HRM) and its incremental construction process using sonar sensors, which are widely used in indoor or home-service robots because of their low-cost and relatively accurate range readings.

Unlike PRM, HRM has a multi-layered graphical structure. Thus, it covers navigable areas more efficiently using a smaller number of evenly distributed nodes. In addition, the construction process of this method updates traversable regions when a mobile robot explores previously unvisited zones.

An HRM consists of two layers, as shown in Figure 1. The first layer is a region roadmap (RRM) that abstracts the connectivity of the subregions. In an RRM, a subregion in the environment is considered a region-node (RN). The second layer consists of a group of local roadmaps (LRMs). An LRM abstracts navigable areas in the corresponding subregions in an RRM. An LRM consists of three types of nodes: hub-nodes (HNs), which briefly abstract navigable areas in the corresponding subregions, sub-nodes (SNs) which almost completely cover navigable areas in the corresponding subregions, and junction nodes (JNs), which provide paths to the LRMs in neighboring RNs.

An HRM is constructed using the following procedure: (i) Sonar data are stacked in several consecutive steps, (ii) A local gridmap is constructed using the stacked sonar data, (iii) A subregion is extracted using the local gridmap and added to an RRM as a new RN, and (iv) An LRM is constructed in the extracted subregion.

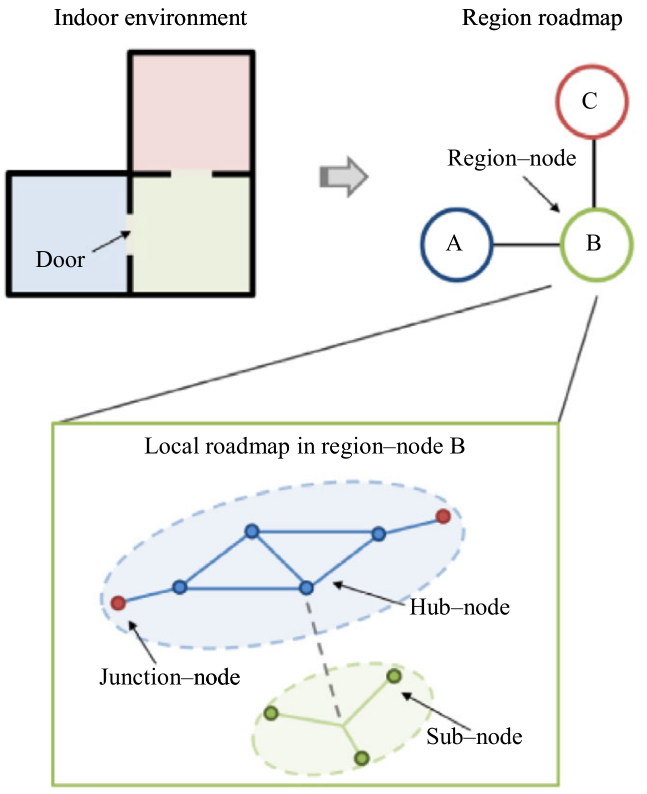


FIGURE 1 Region roadmap abstracts the connectivity of subregions in the environment. Each subregion is represented as a region-node in the region roadmap. Local roadmap represents navigable areas in the corresponding subregion. Hub-nodes briefly abstract navigable areas, sub-nodes almost completely cover navigable areas in the corresponding subregion, and junction-nodes provide paths to local roadmaps in the neighboring region-nodes

We also propose an HRM-based planner using a divide-and-search (DNS) approach, where it seeks appropriate paths to reach the given target locations efficiently by reducing the search-space size using a multi-layered graphical structure of an HRM. A path planner using an HRM searches for an appropriate path using the following procedure: (i) A global topological path is sought based on an RRM, (ii) Local metric paths are planned using the LRMs in the subregions that occur on the topological path, and (iii) A global metric local path is constructed by connecting local metric paths.

In a mobile robot path planner, an HRM has the following advantages over a PRM:

- In the incremental construction process of an HRM, the navigable areas are encoded using a sonar gridmap when a mobile robot navigates unexplored areas.
- An HRM uses a multi-layered graphical structure to represent the navigable areas efficiently.
- The computational complexity of a path planner is reduced owing to the smaller search-space size when using a multi-layered graphical structure.

- An HRM is suitable for human-robot interactions because a multi-layered graphical representation is similar to the environmental perception of human beings.
- An incremental HRM construction process can be used with a wide variety of range sensors.

The remainder of this paper is organized as follows. Related subregion-extraction methods and PRM-based planning methods are explained in Section 2. Then, the RRM construction process is showed in Section 3, after which the construction process of an LRM is presented in Section 4. Section 5 describes a path planner using an HRM, and the experimental results are presented in Section 6. Section 7 concludes the paper.

2 | RELATED WORKS

2.1 | Subregion extraction

HRMs are constructed based on the extraction of subregions in an environment. These extracted subregions are then represented as a graphical model in an RRM. Several researchers have developed different methods of subregion extraction.

Voronoi diagrams are used to divide gridmaps into several subregions, which have been used to construct a topological graph for efficient planning [6]. Topological navigation in corridor environments can be achieved using the eigenvalue ratio of sonar sensor data to detect both node and edge regions [7]. Contour-based segmentation has been used to incrementally divide gridmaps [8]. Spectral clustering and cluster growing were applied to extract node regions from gridmaps by representing navigable areas of the environment as a graphical model [9,10]. Machine learning-based algorithms can be applied to identify the door, room, and corridor regions [11,12]. Although the above-mentioned methods can extract subregions in various environments, most of them are focused on topological modeling or localization and are not appropriate for solving path-planning problems. In addition, some of them are suitable only in corridor environments, where a few distinct locations, such as crossing points, are used as nodes.

2.2 | PRM-based planner

The navigable areas in each extracted subregion can be abstracted using the concept of a PRM to construct the multi-layered graphical structure of an HRM.

A PRM-based path planner solves the global path-planning problem. However, random sampling is used to construct a PRM, and thus its coverage becomes incomplete when a small number of nodes are used. Thus, it may fail to find an appropriate path.

Several methods have been proposed to improve the coverage of a PRM by replacing or improving the random sampling. Nonuniform sampling was proposed to classify unoccupied regions in the environment into several subregions and to assign weights to the classified subregions to increase the number of nodes generated in narrow subregions [13]. A conditional variational autoencoder was used to learn a nonuniform sampling distribution strategy from demonstrations [14]. The distribution of nodes can be improved using a quasi-random number generator [15]. Incremental roadmap generation and adaptive sampling were proposed to determine the appropriate number and arrangement of nodes [16,17]. As a different approach, post-processing methods have been proposed to improve the coverage of a PRM. Useful cycles can be added to improve the coverage of a PRM by adding nodes and reconnecting them to their neighbor nodes [18]. Additional local PRMs are constructed to connect narrow passages [19].

Although these methods may improve the coverage, nodes in the roadmap are still irregularly distributed. Moreover, they do not consider the exploration of unknown areas; most of the methods are mainly used to solve path-planning problems in bounded environments.

3 | REGION ROADMAP CONSTRUCTION

The first layer of an HRM is an RRM, which abstracts the connectivity of subregions in the environment, as shown in Figure 1. Navigable areas of the entire environment are divided into several subregions, which are used to construct the RRM. To do this, in the incremental construction process of the HRM, subregions are extracted from the local gridmap, as indicated in Figure 2A, which is generated by the accumulation of sonar data in several consecutive steps, and adjacent subregions are then connected to construct the RRM as a graphical model [20]. The extraction of subregions is achieved by first obtaining a reliable unoccupied region (RUR) using a local gridmap, and then using a normalized graph cut to extract new subregions.

3.1 | Reliable unoccupied region

3.1.1 | Confident grid-cell extraction

As shown in Figure 2A, the local gridmap contains noisy data owing to the multipath effect of sonar sensors, which need to be suppressed in order to acquire navigable areas in a reliable manner. To do this, a sonar sensor

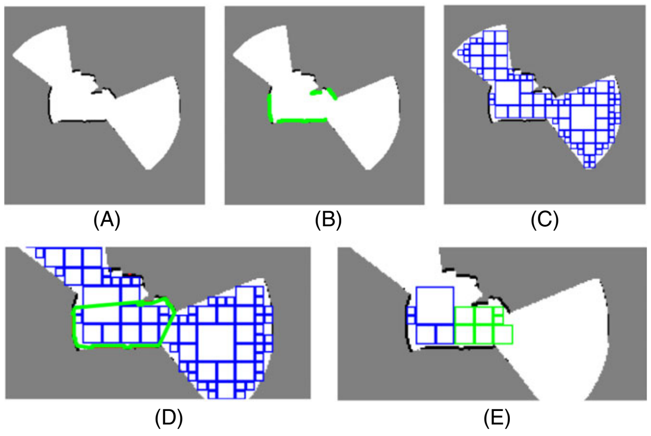


FIGURE 2 (A) Local gridmap, (B) confident grid cells (green), (C) decomposed squares (blue), (D) confident contour (green), and (E) tentative clusters (blue and green cells)

measurement is modeled to evaluate the grid confidence of each occupied grid cell [5,20]:

$$\mathrm{Conf}(i,j) = \sum_{s \in S_{\mathrm{occ}}(i,j)} P_{\mathrm{s}}(r,\theta), \tag{1}$$

where s indicates a sensor measurement, $S_{\mathrm{occ}}(i, j)$ is a set of sensor measurements that determine whether grid cell (i,j) is occupied, and P_{s} is the sound pressure of a sonar sensor measurement [21]. The grid confidence assesses the reliability of each occupied grid cell, as shown in Figure 2B. Grid cells whose confidence values exceed the average are considered confident grid cells.

3.1.2 | Quadtree decomposition

A quadtree decomposition is applied to extract navigable areas in confident grid cells [22]. A quadtree decomposition recursively divides each square into four smaller squares until every square contains only empty confident grid cells. Empty confident grid cells in the local gridmap can be represented using a set of decomposed squares (DSs) of various sizes after applying a quadtree decomposition, as shown in Figure 2C.

3.1.3 | RUR Extraction

The RUR of a local gridmap is extracted to represent navigable areas using the confident grid cells and DSs. The DSs that satisfy the following condition are considered the RUR:

$$Sq_i \cap E \neq \emptyset, \tag{2}$$

where Sq_i is one of the DSs, and E is a set of empty grid cells within the confident contour, which is defined by connecting the confident grid cells, as shown in Figure 2D.

3.2 | Subregion extraction

3.2.1 | Normalized graph cut

A normalized graph cut is used to divide an RUR tentatively into two subregions [23]. The RUR can be represented as a graphical model using the results of the quadtree decomposition. Each DS in the RUR is regarded as a node, and the edges are then defined according to the connectivity of the DSs. The weight value for each edge is set to the number of empty grid cells between adjacent DSs. Applying a normalized graph cut to the RUR divides a reliable region into two tentative subregions (Figure 2E). A normalized graph cut splits the RUR into two subgraphs to optimally minimize the inter similarity between them (3) as follows:

$$N_{\mathrm{cut}} = \frac{\sum_{i \in C_1, j \in C_2} w_{ij}}{\sum_{i \in C_1, j \in (C_1 \cup C_2)} w_{ij}} + \frac{\sum_{i \in C_1, j \in C_2} w_{ij}}{\sum_{i \in C_2, j \in (C_1 \cup C_2)} w_{ij}}, \tag{3}$$

where C_1 and C_2 are clusters, and w_{ij} is a weight value between node i and node j.

3.2.2 | New subregion extraction

Using the tentative subregions divided by the normalized graph cut, convexity measures are computed to determine whether the RUR can be considered a single subregion, or whether it can be divided into two subregions. The convexities are calculated as shown in (4) and (5), respectively [5, 20]:

$$C_1 = \frac{n(o \in \mathrm{Conv}(\mathrm{RUR}))}{\sum_{i=1}^{n} Sq_i}, \tag{4}$$

where o is an occupied confident grid cell, and Conv(RUR) is a convex hull of the RUR.

$$C_1 = \frac{n(o \in \mathrm{Conv}(Cl_1)) + n(o \in \mathrm{Conv}(Cl_2))}{\sum_{i=1}^{n} Sq_i}, \tag{5}$$

where Cl_1 and Cl_2 are divided tentative subregions. The above convexity measures are defined as the ratio of the number of occupied confident grid cells to the total number of confident grid cells in each convex hull. RURs with a large convexity are more concave than those with a small convexity.

A new subregion is extracted when the following conditions are satisfied [5,20]:

$$C_1 > c_{\mathrm{t}} \;\&\; C_2 < \frac{C_1}{2}, \tag{6}$$

where c_{t} is a threshold value; otherwise, the robot continues to accumulate sonar data to generate a local gridmap.

The extraction of new subregions, which is shown in Figure 3, proceeds as follows. If the tentative subregions do not satisfy the extraction conditions in (6), the robot continues to

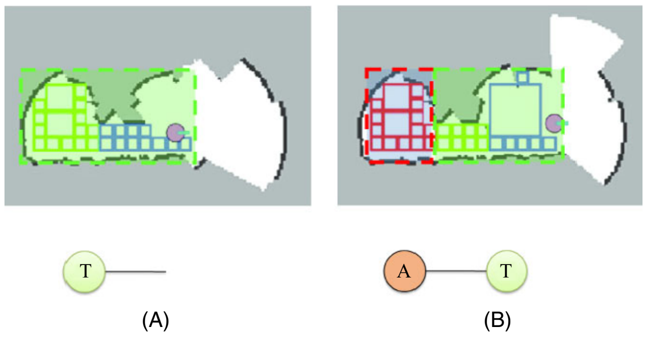


FIGURE 3 Example of incremental subregion extraction for region roadmap construction. (A) If the tentative subregions (blue and green cells in green boxes) do not satisfy the extraction conditions in (6), the robot continues to accumulate sonar data. (B) New subregion is extracted when the extraction conditions are satisfied. This subregion is added to the RRM as a new node (red cells in the red box)

accumulate sonar data to generate the local gridmap, as shown in Figure 3). A new subregion is extracted when the extraction conditions are satisfied, and the new subregion is then added to the RRM as a new RN, as shown in Figure 3B.

4 | LOCAL ROADMAP CONSTRUCTION

An LRM is constructed in each RN of an RRM, and is composed of three types of nodes: HNs, SNs, and JNs (Figure 1).

Hub-nodes briefly represent navigable areas in the corresponding RN. SNs cover navigable areas almost completely, and are connected with the nearest HNs to constitute a hub-and-spoke topology, and JNs provide paths to neighboring RNs in the RRM. An LRM is constructed using the following procedure: (i) generate and rearrange nodes to cover navigable areas in the corresponding RN, (ii) classify the nodes into HNs and SNs, (iii) connect HNs and SNs to construct hub-and-spoke topologies, and (iv) add JNs to connect neighboring RNs.

4.1 | Node generation

The first step in the construction of an LRM in an RN is to generate nodes to cover navigable areas in the RN. We initially used the information of an RUR to generate nodes and then rearranged the positions of the nodes to increase the uniformity of their distribution. The nodes are initially generated using the DSs in RUR, and the positions of these nodes are set as centroids of the DSs.

Centroidal Voronoi tessellation (CVT) is used to rearrange the generated nodes in order to improve their distribution [24]. CVT is a particular type of general Voronoi tessellation (GVT). A Voronoi region in GVT is defined as follows [5]:

$$V_i = \{x \in \Omega_{\text{occ}}^C : D(x, n_i) \leq D(x, n_j), i \neq j\}, \tag{7}$$

where n_i is the Voronoi node of V_i, n_j is another Voronoi node not included in v_j, $D(x, n_i)$ is the distance between x and n_i, and Ω_{occ} is the occupied region in the environment. CVT is defined only when every node satisfies the following equation [5]:

$$n_i = \frac{\int_{V_i} x\mathrm{d}x}{\int_{V_i} \mathrm{d}x}. \tag{8}$$

General Voronoi tessellation is converted into CVT by iteratively updating the positions of the nodes. The position of a node is updated in each iteration using the following equation [5,25]:

$$n_i^* = \frac{((1-\alpha) \times j_i + (1-\beta))}{j_i + 1} \times n_i + \frac{(\alpha \times j_i + \beta)}{j_i + 1} \times \bar{w}_i, \tag{9}$$

where n_i^* is the updated position of n_i, j_i is the number of iterations, α and β are damping coefficients, and $\bar{w}$ is the centroid of a group of nearest neighbors of n_i, namely, $W(n_i)$, which is defined as follows [5]:

$$W(n_i) = \{y \in Y : D(y, n_i) \leq D(y, n_j), i \neq j\}, \tag{10}$$

where Y is a set of random samples, and is expressed as follows:

$$Y = \{y_{k=1}^q \in \Omega_{\text{occ}}^c\}. \tag{11}$$

These random samples are used to estimate the Voronoi center of each node without calculating the area of the Voronoi region. Ideally, the nodes remain stationary after several iterations. However, the movements of the nodes do not stop because of the inaccurate estimation of the Voronoi centroid using the unevenly distributed random samples.

To mitigate this problem, Halton sampling was adapted to generate better evenly distributed random samples [26, 27]:

$$y_k = [S_r \times \Phi_{p1}(k), S_c \times \Phi_{p2}(k)]^T, \tag{12}$$

where s_r and s_c are the row and column sizes of the local gridmap of the RN, respectively, and $\Phi_p(k)$ is a function to determine a random number, which is expressed as follows:

$$\Phi_p(k) = \sum_{i=0}^{r} \frac{a_i}{p^{i+1}}, \tag{13}$$

where p is a prime number, and k is a nonnegative integer that can be expanded using the prime base p, and is expressed as follows:

$$k = \sum_{i=0}^{r} a_0 \times p^i. \tag{14}$$

The initially generated nodes are iteratively moved until their positions are converged [5]. The initial position is set

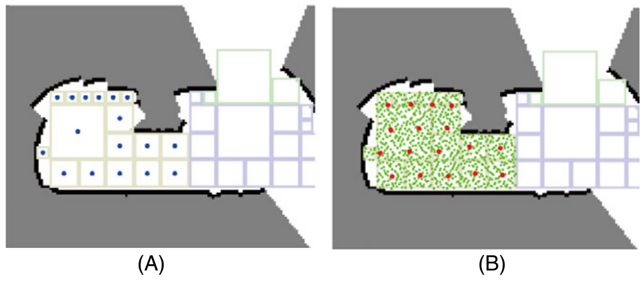


FIGURE 4 Example of a node rearrangement: (A) 17 initial nodes are generated in the extracted RN and (B) Updated nodes after 20 iterations. Blue and red dots represent the initial and updated nodes, respectively, and the green dots represent random samples generated using Halton sampling

using the DSs in the RUR, as shown in Figure 4A. The nodes are then rearranged, as shown in Figure 4B. The rearranged nodes almost completely cover and regularly divide the navigable areas in an RN. The mean distance updates of the nodes in each iteration using the random samples generated by pseudorandom and Halton sampling are shown in Figure 5. The nodes updated using Halton sampling converged smoothly, as opposed to the nodes using the pseudorandom method, which continued to move even after numerous iterations.

4.2 | Node classification

Rearranged nodes are classified into HNs and SNs, and are connected to construct hub-and-spoke topologies.

To classify these rearranged nodes, affinity propagation clustering (APC) [28] was applied to group them into several subsets and identify the representatives of those subsets. The representative of each subset is set as its HN, and the other nodes are considered SNs. The APC identifies the HNs and SNs by measuring the similarities between the rearranged nodes. Each rearranged node exchanges messages with other rearranged nodes recursively until the HNs and SNs are identified.

In each iteration, a rearranged node exchanges two types of messages with its counterparts; each node sends a responsibility message to every other node individually, which responds with an availability message. A responsibility message $r(n_i, n_i)$ is sent from a node n_i to a neighboring node n_k, and is used to measure whether n_i can be a representative for n_k, which is expressed as follows [28]:

$$r(n_i,n_k)=s(n_i,n_k)-\max(\{a(n_i,n_k')+s(n_i,n_k')\}),\quad n_i,n_k'\in N(n_i),n_k'\neq n_k, \tag{15}$$

where $s(n_i, n_k)$ is a similarity between n_i and n_k, and $N(n_i)$ represents n_i's neighboring nodes. This similarity is defined as follows [28]:

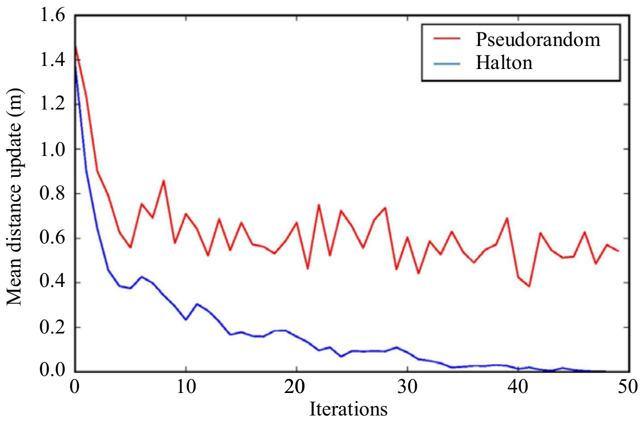


FIGURE 5 Halton sampling generates unbiased random samples to update the positions of initial nodes. Blue and red lines show the mean distance updates using pseudorandom and Halton samplings, respectively, over various iterations

$$s(n_i,n_k)=-|n_i-n_k|^2. \tag{16}$$

An availability message $a(n_k, n_i)$ is sent from a neighboring node n_k to a node n_i, and is a value that is used to measure whether k_i can be a member of a subset represented by n_k [28]:

$$a(n_i,n_k)=\min\left(0,r(n_k,n_k)+\sum_{n_i'\notin(n_i,n_k)}\max\left(0,r(n_i',n_k)\right)\right), \tag{17}$$

where $r(n_k, n_k)$ is a self-availability message, and is expressed as follows [28]:

$$a(n_i,n_k)=\sum_{n_i'\neq n_k}\max\left(0,r(n_i',n_k)\right). \tag{18}$$

The responsibility and availability messages are exchanged to identify representative nodes and their subsets among all rearranged nodes. After classifying the HNs, SNs are connected with their HN to make a hub-and-spoke topology, as shown in Figure 6.

4.3 | Junction node

After constructing the hub-and-spoke topology, JNs are added to provide passages to the neighboring RNs. The location of a JN is set on the boundary between two subregions, as shown in Figure 6.

5 | HIERARCHICAL PATH PLANNING

5.1 | Divide-and-Search Approach

A PRM-based path planner uses various graph-search algorithms to find a path for a given query. When the PRM

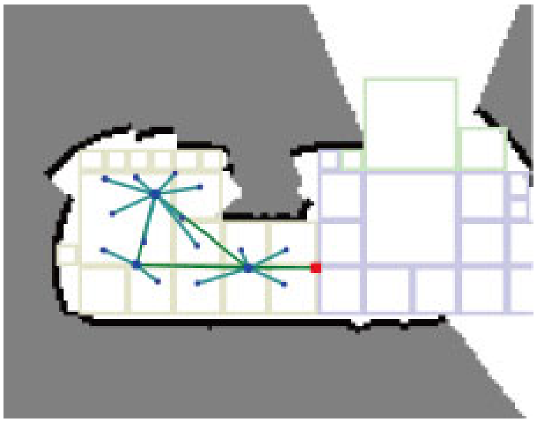


FIGURE 6 LRM construction in an RN. The large blue dots represent HNs, the small blue dots represent SNs, the red dots represent JNs, and the green lines represent edges

has a large number of nodes and edges, the computational complexity of finding the minimum-cost path increases because this complexity depends on the number of nodes and edges considered in the PRM. If the PRM has a large number of nodes and edges, or if the computing power of a mobile robot is limited, then queries may not be answered promptly owing to the high computational complexity.

An HRM-based planner can seek a path for a given query with less computational complexity than a PRM-based planner by using the multi-layered graphical structure of HRM. The multi-layered structure helps to reduce the number of nodes and edges that are touched by a graph-search algorithm using a DNS approach. This approach seeks a global topological route using an RRM, and local metric paths are then sought using LRMs.

The RNs, which include the initial and target locations, are set as the initial and target RNs, respectively.

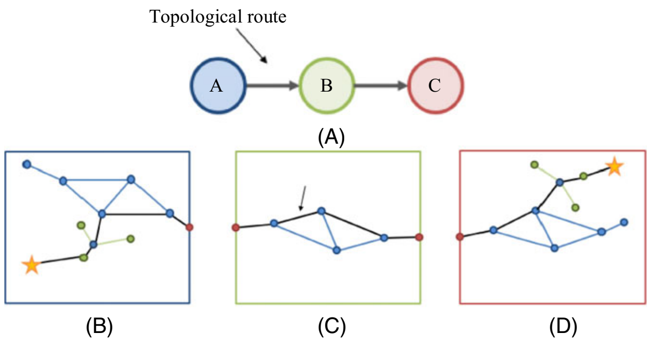


FIGURE 7 HRM-based divide-and-search approach: (A) global topological route from RN A to RN C (the black arrows represent the global topological route), (B) local metric path in the initial RN (the yellow star is the initial location), (C) local metric path in RN B, and (D) local metric path in the target RN (the yellow star is a target location, and the black lines represent local metric paths)

A global topological route from the initial RN to the target RN, as shown in Figure 7A, is sought based on the RRM.

After finding the global topological path, local metric paths in the RNs on the topological route are sought in the LRMs. To search for the local metric paths, local initial and target locations are set in each RN. In the initial RN, as shown in Figure 7B, the JN connected to the next RN on the global topological route is set as the local target location. In the target RN, the JN connected to the previous RN is set as the local initial location, as shown in Figure 7D. In the intermediate RNs on the global topological route, the JNs connected to the previous and next RNs on the global topological route are set as the local initial location and the local target location, respectively. Unlike the initial and target RNs, only HNs are considered to search for the metric paths in the intermediate RNs, as shown in Figure 7C.

5.2 | Post-processing

Although an HRM-based planner efficiently searches for an appropriate path, it has redundant sections that increase the total path length, and has sharp curves that may cause unstable motion or slippage [29]. To improve the quality of the path-planning result, iterative path-pruning and path-smoothing techniques are applied. The iterative path-pruning [30] approach eliminates redundant sections by searching for shortcuts based on the path-planning result. This method randomly selects two waypoints on the planned path to iteratively search for shortcuts; then, if it is possible to generate between two selected waypoints a shortcut that does not cause a collision with obstacles, it removes all waypoints between the two selected waypoints. continuous cubic Bézier curve (G1CBC) path smoothing was adopted to eliminate abrupt directional changes or slippage [31].

6 | EXPERIMENTAL RESULTS

We tested our proposed HRM construction and path-planning method using a self-acquired indoor environment dataset, as shown in Figure 8. The size of the environment was 11.4 m × 8.4 m. In addition, the gridmap resolution was 5 cm × 5 cm. The dataset was acquired using a Pioneer 3DX robot with 12 sonar sensors.

6.1 | HRM construction

An HRM represents the navigable areas incrementally when a mobile robot navigates unknown areas using sonar gridmaps [32].

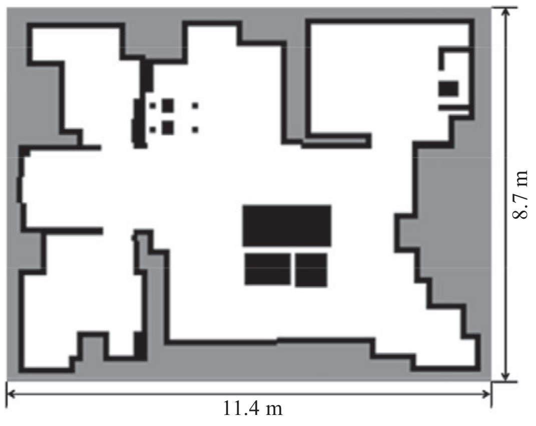


FIGURE 8 Home environment

The RRM was constructed using incremental topological modeling, as shown in Figure 9. The RURs were successfully obtained by removing noisy sonar data, as shown in Figures 9A,D. The environment was segmented into 10 subregions: three rooms were extracted as three different subregions, and the living room was extracted into several subregions (because the sofa and table were located in the center of the living room).

The extracted subregions were encoded in RRM, as shown in Figures 9E–H, in which each extracted subregion was set as an RN. Tentative subregions are represented by green circles in these RRMs, and do not satisfy the extraction conditions in (6). Edges in these RRMs represent the connectivity of adjacent RNs.

For each RN of every RRM, an LRM was constructed to cover navigable areas, as shown in Figures 10A–D. The LRM abstracted navigable areas very efficiently because the connectivity of navigable areas in each RN was represented by a sparse graphical structure. Nodes of these LRMs were almost evenly distributed in the navigable areas, and they helped to improve the coverage of the

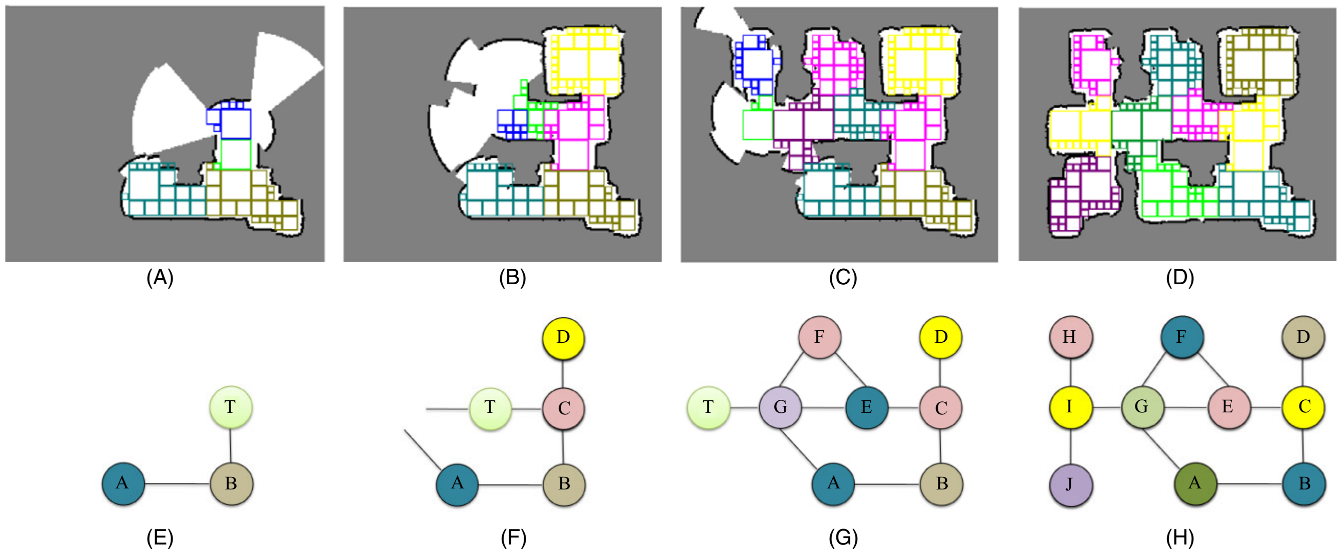


FIGURE 9 Region roadmap construction: (A) sequence 23, (B) sequence 49, (C) sequence 69, and (D) sequence 122 autonomous subregion extraction (each subregion is shown in a different color, and the blue and green cells are tentative subregions). A single sequence consists of 30 consecutive odometry and sonar readings. (E) sequence 23, (F) sequence 49, (G) sequence 69, (H) sequence 122 constructed RRMs (the greed and grey circles represent tentative and added RNs, respectively)

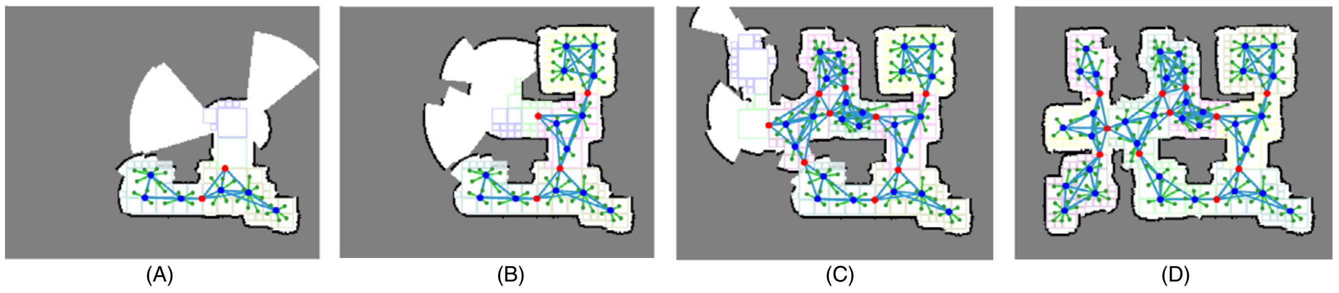


FIGURE 10 Local roadmap construction: (A) sequence 23, (B) sequence 49, (C) sequence 69, and (D) sequence 122 local roadmaps are constructed in the corresponding subregions, as shown in Figure 9. The blue dots and green dots represent HNs and SNs, respectively, and the red dots represent JNs. The positions of the JNs are set on the boundary between two adjacent RNs

TABLE 1 Results of map construction using a single-layered PRM and an HRM method (average of 100 runs)

Statistic	Coverage (%)	Regularity
PRM	98.0	60.1
HRM	99.6	8.2

HRM and increase the chance of finding efficient paths to solve arbitrary queries. JNs are equivalent to the edges of the RRM and are used to connect two LRMs of adjacent RNs.

The incremental process of constructing the HRM responds to changes in a sonar gridmap when a mobile robot revisits the previously extracted subregion. If the sonar gridmap of the revisited subregion changes, the LRM of the corresponding RN is reconstructed. The LRM in RN A, as shown in Figure 10A, was reconstructed, as indicated in Figure 10D, when the mobile robot revisited RN A because the sonar gridmap of the subregion had changed. However, the RRM did not change as the RN A was already encoded to it previously. Hence, the topological relationship between the RN A and its neighboring RNs remained intact. The total number of nodes in the HRM was 208.

The construction results of the HRM were compared with those of the PRM, which has a single-layered graphical structure, to verify whether HRM covered and evenly divided most of the navigable areas in an environment. PRM takes the number of nodes as a hyperparameter, whereas HRM optimizes it automatically. In our experiments, for PRM, this number was directly taken from the output of HRM; hence, for all of our results, the coverage and the regularity of an environment are obtained in both methods using the same number of nodes.

The coverage C of the PRM and HRM was calculated as follows:

$$C = \frac{\sum_{i=1}^{k} \int_{\mathrm{VisV}(n_i)} x\mathrm{d}x}{\int_{\Omega_{\mathrm{occ}}^{C}} x\mathrm{d}x}, \tag{19}$$

where VisV is the visible region [33] of each node, which is the area that can be reached from the node without encountering obstacles, and is calculated as follows:

$$\mathrm{VisV}(n_i) = \{x \in V(n_i) \cap \mathrm{Vis}(n_i)\}. \tag{20}$$

HRM almost completely (99.6%) covered navigable areas in the environment, whereas PRM covered 98.0%, as shown in Table 1.

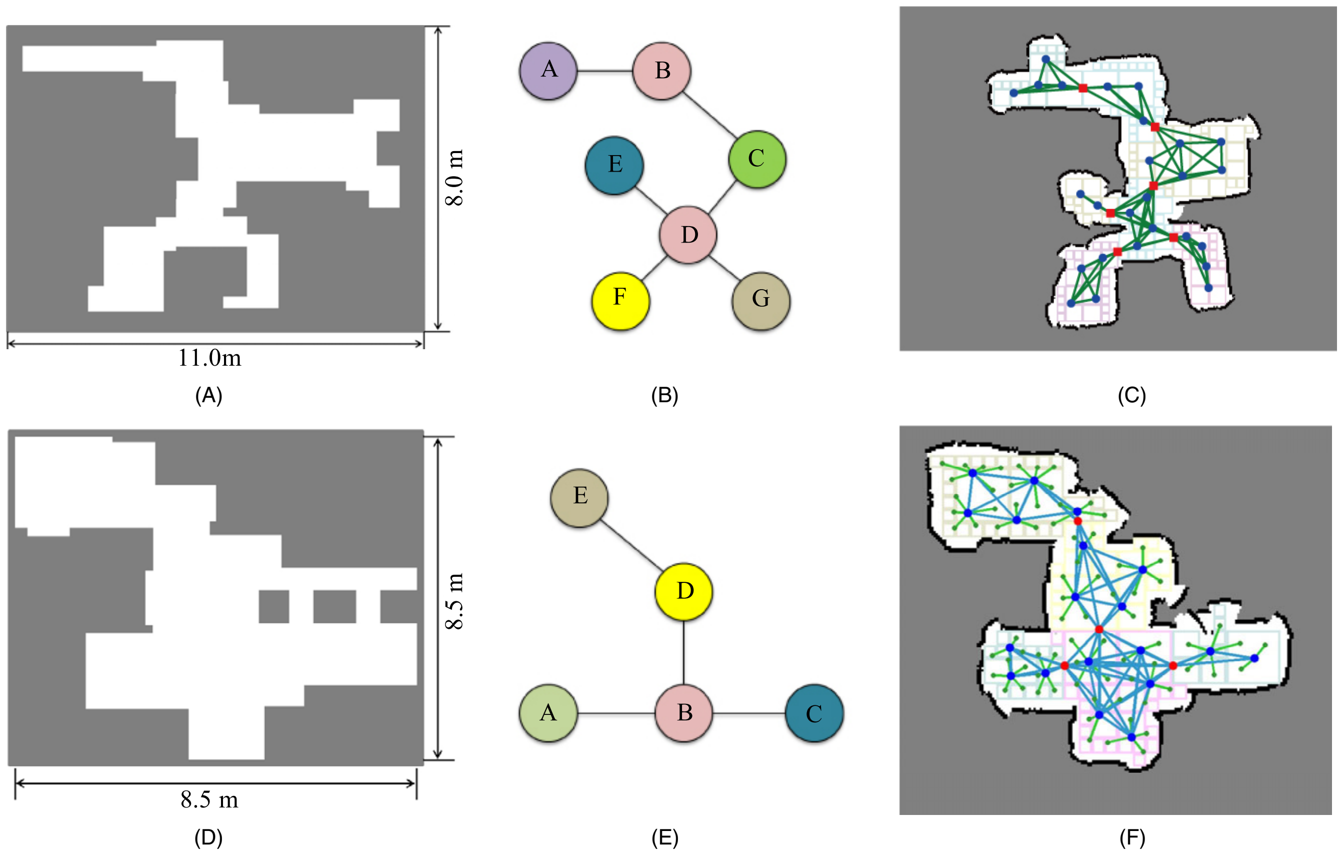


FIGURE 11 Experimental results: (A,D) apartment and guesthouse environments, (B,E) region roadmaps, and (C,F) local roadmaps

The distributions of nodes in HRM and PRM are very different. To measure these distributions, the regularity χ in a d-dimensional space was measured [34].

$$\chi = \left(\max_{i=1,\ldots,k} \chi(n_i)\right) - 1, \chi(n_i) = \frac{\sqrt{d}h(n_i)}{\gamma(n_i)}, \tag{21}$$

where

$$h(n_i) = \max_{x \in \mathrm{VisV}}(n_i) D(x, n_i) \tag{22}$$

is the point distribution norm, and

$$\gamma(n_i) = \min_{n_i \neq n_j} D(n_i, n_j) \tag{23}$$

is the point covariance. The regularity decreases when the nodes are evenly distributed and increase otherwise. It should be noted that the regularity of the HRM was much smaller than that of the PRM, as shown in Table 1, which verifies that HRM was much more efficient in distributing the nodes evenly.

The experimental results in two different indoor environments also show that the incremental HRM construction process appropriately abstracted the navigable areas, as shown in Figures 11. The process yielded 142 and 101 nodes for apartment and guesthouse environments, respectively.

6.2 | Path planning the hierarchical roadmap

An HRM-based planner uses the DNS approach to seek an appropriate path for given initial and target locations, as shown in Figure 12A.

Region-nodes H and B were set as initial and target RNs, respectively, to search for a global topological path. The topological path was searched in the order of H→I→G→A→B, as shown in Figure 12B, according to the approximated edge costs of the RRM.

Local metric paths were sought in the RNs falling on the global topological path. Within the initial and target RNs, LRMs were used to seek the local metric paths. Meanwhile, for the other RNs, only the HNs were used because the local metric paths in these RNs have only a small effect on the total path length. SNs were not considered for these computations. The local metric paths that were found were connected to make the global metric path. To remove redundancy and sharp turns in the global metric path, iterative path pruning and G1CBS path-smoothing techniques were applied, as in Figure 12C.

Benefits of the DNS approach using the multi-layered graphical structure of HRM were verified by comparing it to a single-layered PRM with the same number of nodes, as shown in Figure 12D.

Figures 13 and 14 also show path planning results using both planners in the apartment and guesthouse environments.

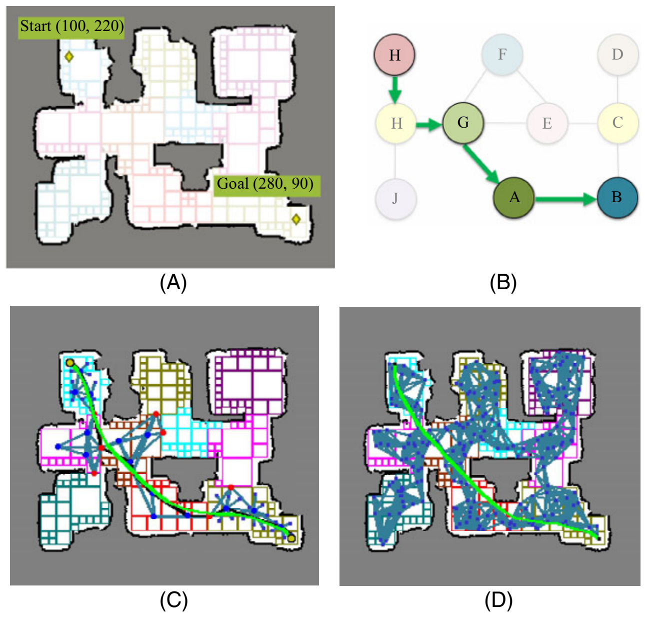


FIGURE 12 Planning result of the example query: (A) initial (yellow star, upper left) and target (yellow star, lower right) locations, (B) global topological path (H is the initial RN. I, G, and A are intervening RNs. B is the destination RN.), (C) global metric path, and (D) global metric path sought based on the single-layered PRM

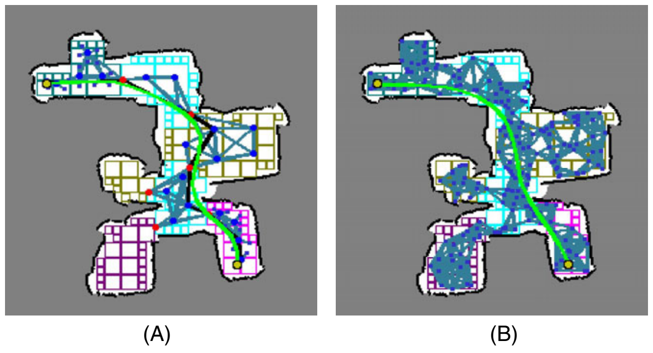


FIGURE 13 Planning results in the apartment environment: (A) HRMbased planner, and (B) PRM-based planner. Total number in the PRM is the same as that in the HRM: 142

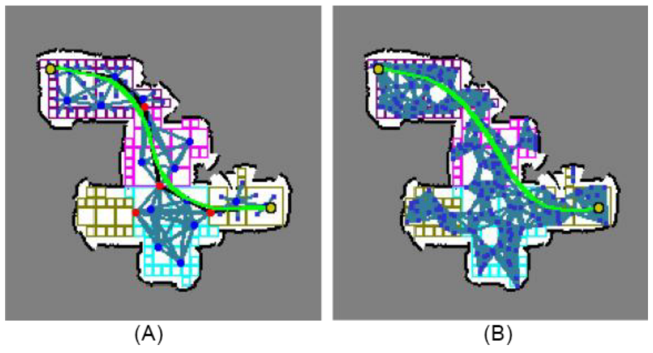


FIGURE 14 Planning results the guesthouse environment : (A) HRM-based planner and (B) PRM-based planner. Total number in the PRM is the same as that in the HRM: 101

TABLE 2 Query times (ms) of the example query shown in Figure 14 using a single-layered PRM and HRM (average of 100 runs)

	Overhead	Search	Postprocess	Total
PRM	0.57	75.21	3.14	78.91
HRM	1.09	8.68	3.51	13.28

Note. The total number in the PRM is the same as that in the HRM: 208.

The query times, as shown in Table 2, for the HRM and single-layered PRM were recorded to measure the computational complexity. Dijkstra's graph-search algorithm was used on top of the HRM and PRM to search for an appropriate path using one example query. The overhead of the PRM-based planner involved connecting the given initial and target locations to the PRM. However, to search for an appropriate path using an HRM, local initial and target locations must first be set, after which they are then connected to the corresponding LRMs. Notwithstanding its higher number of operations, an HRM-based planner requires a similar amount of time as does a PRM-based planner because the HRM-based planner considers fewer nodes and edges to connect the initial and target locations. Post-processing times of an HRM-based planner are similar to those of a PRM-based planner. The graph-searching time of an HRM-based planner is much shorter because of the use of the DNS approach. Nodes and edges of the LRMs, which are not included in the topological path, are also not touched by the graph-search algorithm. This exclusion reduces the computational complexity of the HRM-based planner, compared with the PRM-based planner, which touches all nodes and edges between the initial and target locations.

Figure 15 shows total query times of the HRM- and PRM-based planners in home, apartment, and guesthouse environments. In each environment, 10 different queries were used to measure the query times. It should be noted that the query times of the HRM-based planner did not increase much, even after the number of nodes was increased with the increase in size or structural complexity of an environment. However, the query times of the PRM-based planner increased with the expansion in the environment. The standard deviation of the total query times of the PRM-based planner is much higher than that of an HRM-based planner. The coverage and distribution of nodes are different each time a PRM is constructed, especially when a limited number of nodes are used, whereas the HRM does not change much for any number of reconstructions.

As shown in Table 3, the success rates and planning times for 10 queries in each environment were measured to compare the robustness of the HRM- and PRM-based planners. The success rate is defined as the number of times that a path planner successfully retrieved results to solve given queries. The HRM-based planner searched for appropriate paths more robustly than the PRM-based planner because of its better coverage and regularity.

The total path lengths that resulted from the HRM- and PRM-based planners were also measured in the three environments, as shown in Figure 16. The total path lengths output by the HRM-based planner were slightly longer than

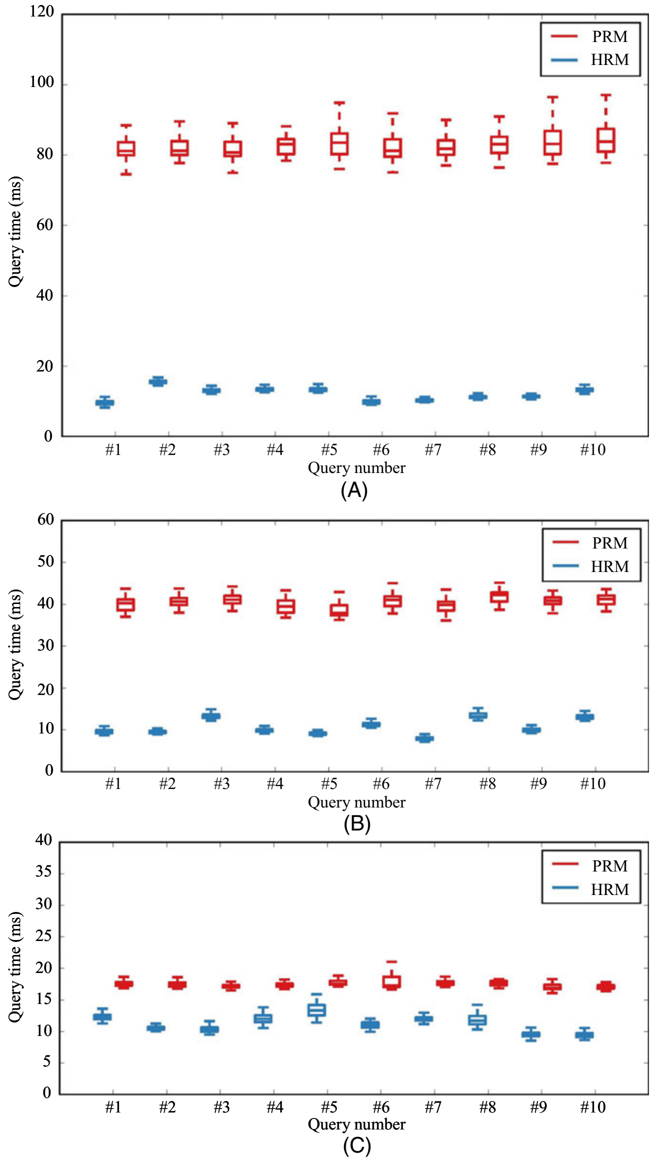


FIGURE 15 Query time (ms) for 10 queries using a single-layered PRM and an HRM (average of 100 runs): (A) HOME environment (208 nodes), (B) apartment environment (142 nodes), and (C) guesthouse environment (101 nodes)

TABLE 3 Success rates (%) for 10 queries using a single-layered PRM and HRM (average of 100 runs for each query)

Environment	Home	Apartment	Guesthouse
Number of nodes	208	142	101
PRM	89.8	91.4	92.6
HRM	100.0	100.0	100.0

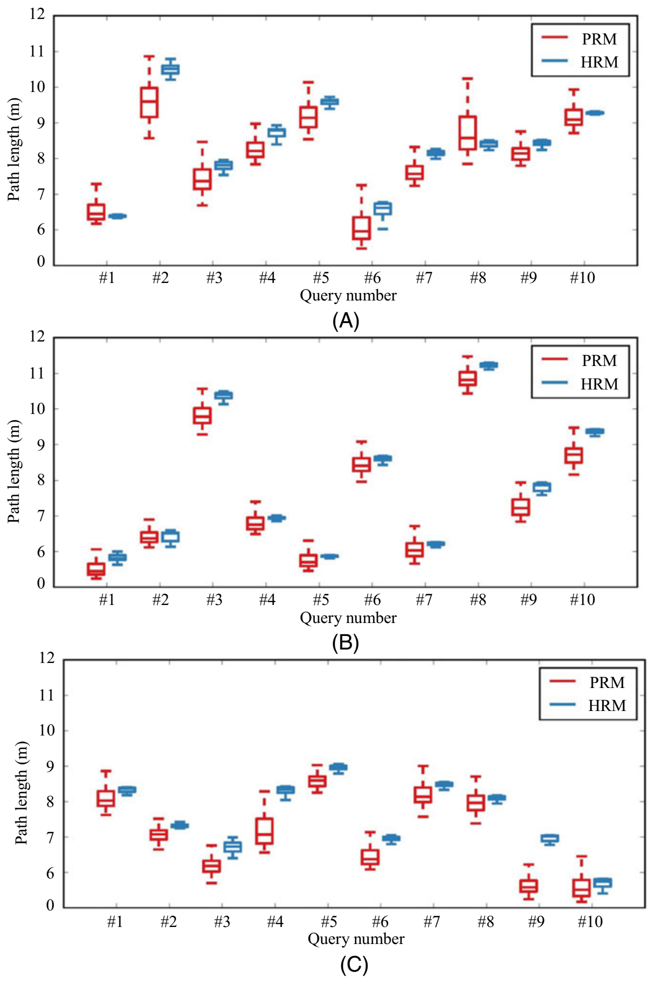


FIGURE 16 Path length (m) for 10 queries using a single-layered PRM and an HRM (average of 100 runs): (A) home environment (208 nodes), (B) apartment environment (142 nodes), and (C) guesthouse environment (101 nodes)

those of the PRM-based planner because the DNS approach in these planners only considers HNs in RNs on global topological routes.

In summary, our experimental results verify that the HRM efficiently abstracts the navigable areas. In addition, the HRM-based planner efficiently searches for appropriate paths for the given queries using a multi-layered graphical structure and the DNS approach to reduce the size of the search space.

7 | CONCLUSIONS

This paper proposed an HRM and its incremental construction process. The proposed HRM efficiently represents navigable areas using a multi-layered graphical structure and comprises an RRM and a set of LRMs. The RRM abstracts the connectivity of subregions in the environment, whereas LRMs represent navigable areas within the corresponding subregions. The construction process incrementally represents the navigable areas when a mobile robot visits unknown areas by autonomously extracting new subregions using sonar data.

The HRM-based planner using the DNS approach can seek appropriate paths to rapidly solve arbitrary queries. The DNS approach reduces the size of the search space using the multi-layered graphical structure of an HRM.

The advantages of the HRM were experimentally verified in real environments. Compared to a single-layered PRM, an HRM covered nearly all navigable areas in the indoor environments, and they were divided evenly. Moreover, a method for constructing the HRM can be used with a wide variety of range sensors. In addition, we developed a method for constructing an HRM for sonar sensors; our proposed HRM can also be applied to any sensor that can be used to build a gridmap, such as a laser rangefinder.

ACKNOWLEDGEMENTS

This work was supported by IITP grant (No. 2017-0-00067) and NRF grant (No. 2011-0030075) funded by MIST, and a grant from an endowment project (PES9390) funded by KRISO. The author would like to thank Mr. M. Zaigham Zaheer for his constructive comments.

ORCID

Byungjae Park http://orcid.org/0000-0002-8952-0736

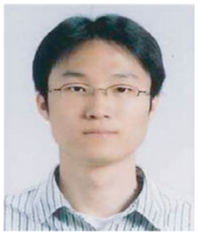
REFERENCES

1. D. Dolgov et al., *Path planning for autonomous vehicles in unknown semi-structured environments*, Int. J. Robot. Res., **29** (2010), no. 9, 485–501.
2. L. Kavraki et al., *Probabilistic roadmaps for path-planning in high-dimensional configuration spaces*, IEEE Trans. on Robot. and Autom., **12** (1996), no. 4, 566–580.
3. J. van den Berg et al., Anytime path-planning and replanning in dynamic environments, *IEEE Int. Conf. Robot. Autom.*, Orlando, FL, USA, May 15–19, 2006, pp. 2366–2371.
4. A. Vázquez-Otero et al., *Reaction diffusion Voronoi diagrams: From sensors data to computing*, Sens., **15** (2015), no. 6, 12736–12764.
5. B. Park et al., An efficient mobile robot path planning using hierarchical roadmap representation in indoor environment, *IEEE Int. Conf. Robot. Autom.*, St. Paul, MN, USA, May 14–18, 2012, pp. 180–186.
6. S. Thrun, *Learning metric-topological maps for indoor mobile robot navigation*, Artif. Intell., **99** (1998), no. 1, 21–77.
7. K. Lee et al., Topological navigation of mobile robot in corridor environment using sonar sensor, *IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, Beijing, China, October 9–15, 2006, pp. 2760–2765.
8. L. Fermin-Leon et al., Incremental contour-based topological segmentation for robot exploration, *IEEE Int. Conf. Robot. Autom.*, Singapore, May 29–June 3, 2017, pp. 2554–2561.

9. H. Wu et al., *Spatial semantic hybrid map building and application of mobile service robot, Robot*. Auton. Syst., **62** (2014), no. 6, 923–941.
10. F. Blöchliger et al., *Topomap: Topological mapping and navigation based on visual SLAM maps*, arXiv, 1709.05533, 2018.
11. O. M. Mozos and W. Burgard, Supervised learning of topological maps using semantic information extracted from range data, *IEEE Int. Conf. Robot. Autom.*, Beijing, China, October 9–15, 2006, pp. 2772–2777.
12. L. Shi and S. Kodagoda, *Towards generalization of semi-supervised place classification over generalized Voronoi graph*, Robot. and Auton. Syst., **61** (2013), no. 8, 785–796.
13. J. van den Berg and M. Overmars, *Using workspace information as a guide to non-uniform sampling in probabilistic roadmap planners*, Int. J. Robot. Res., **24** (2005), no. 2, 1055–1071.
14. B. Ichter et al., *Learning sampling distributions for robot motion planning*, arXiv, 1709.05448, 2018.
15. M. Branicky, Quasi-randomized path-planning, *IEEE Int. Conf. Robot. Autom.*, Seoul, Rep. of Korea, May 21–26, 2001, pp. 1481–1487.
16. D. Xie et al., *Incremental map generation*, Algo. Found. Robot. VIII, **47** (2008), 53–68.
17. S. Rodriguez et al., *RESAMPL: A region-sensitive adaptive motion planner*, Algo. Found. of Robot. VIII, **47** (2008), 285–300.
18. R. Geraerts and M. Overmars, Creating high-quality roadmaps for motion planning in virtual environments, *IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, Beijing, China, October 9–15, 2006, pp. 4355–4361.
19. H. Lee and B. Lee. *Path planning based on probabilistic roadmap for initial deployment of Marsupial robot team*, J. Control Rob. Syst., **24** (2018), no. 1, 80–89.
20. J. Choi et al., *Autonomous topological modeling of a home environment and topological localization using a sonar gridmap*, Auton. Robot., **30** (2011), no. 4, 351–368.
21. L. Kleeman and R. Kuc, Sonar sensing, *Handbook on Robot.*, Springer-Verlag, 2008.
22. N. Katevas et al., *The approximate cell decomposition with local node refinement global path-planning method: Path nodes refinement and curve parametric interpolation*, J. of Intell. Robot. Syst., **22** (1998), no. 3–4, 289–314.
23. J. Shi and J. Malik, *Normalized cuts and image segmentation*, IEEE Trans. Pattern Anal. Machine Intell., **22** (2000), no. 8, 888–905.
24. Q. Du et al., *Centroidal Voronoi Tessellations: Applications and algorithms*, SIAM Review, **41** (1999), no. 4, 637–676.
25. B. Park and W. K. Chung, *Conf. Intell. Robot. Syst.*, St. Louis, MO, USA, October 11–15, 2009, pp. 4399–4405.
26. T. T. Wong et. al., *Sampling with Hammersley and Halton points*, J. Graph. Tool., **2** (1997), no. 2, 9–24.
27. B. Park and W. K. Chung, *Efficient environment representation for mobile robot path-planning using CVT-PRM with Halton sampling*, Electron. Lett., **48** (2012), no. 22, 1397–1399.
28. B. Frey and D. Dueck, *Clustering by passing messages between data points*, Science, **315** (2007), no. 5184, 972–976.
29. E. Magid et al., Spline-based robot navigation, *IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, Beijing, China, October 9–15, 2006, pp. 2296–2301.
30. R. Geraerts and M. Overmars, *Creating high-quality paths for motion planning*, Int. J. Robot. Res., **26** (2007), no. 8, 845–863.
31. K. Yang and S. Sukkarieh, *An analytical continuous-curvature path-smoothing algorithm*, IEEE Trans. Robot., **26** (2010), no. 3, 561–568.
32. K. Lee and W. K. Chung, *Effective maximum likelihood grid map with conflict evaluation filter using sonar sensors*, IEEE Trans. Robot., **25** (2009), no. 4, 887–901.
33. T. Simeón et al., *Visibility-based probabilistic roadmaps for motion planning*, Adv. Robot., **14** (2012), no. 6, 477–493.
34. H. Nguyen et al., *Constrained CVT meshes and a comparison of triangular mesh generators*, Comp. Geom., **42** (2009), no. 1, 1–19.

AUTHOR BIOGRAPHIES

Byungjae Park received his PhD degree in mechanical engineering from POSTECH in 2013. He is currently a senior researcher at ETRI. His main research interests are robotic perception, sensor fusion, mapping, and localization.


Jinwoo Choi received his PhD degree in Mechanical Engineering from POSTECH in 2011. He is currently a senior researcher at KRISO. His main research interests are underwater navigation, mapping, and localization.

Wan Kyun Chung received his PhD in production engineering from KAIST in 1987. He is a professor in the Department of Mechanical Engineering, POSTECH (he joined the faculty in 1987). In 1988, he was a visiting professor at the Robotics Institute of Carnegie-Mellon University. In 1995, he was a visiting scholar at the University of California, Berkeley. His research interests include autonomous navigation, robust control, and medical robots. He served as an editor for the IEEE Transactions on Robotics, and he is currently an editor-in-chief for Intelligent Service Robotics.