Two Layer Path Planning for Multi-Map Navigation

(다중 지도 네비게이션을 위한 이중 레이어 경로 계획)

지도교수 : 권태경

이 보고서를 공학학사 학위 논문 대체 보고서로 제출함.

2025년 4월 18일

서울대학교 공과대학

컴퓨터공학부

장준서

2025년 8월

Two Layer Path Planning for Multi-Map Navigation

(다중 지도 네비게이션을 위한 이중 레이어 경로 계획)

지도교수 : 권태경

이 보고서를 공학학사 학위 논문 대체 보고서로 제출함.

2025년 4월 18일

서울대학교 공과대학

컴퓨터공학부

장준서

2025년 8월

Abstract

This paper presents a hierarchical path-planning approach designed for robots operating across multiple maps. The proposed method utilizes a hybrid map structure consisting of two layers: an upper-layer graph that represents navigational waypoints and transitions between maps, and a lower-layer occupancy grid that provides detailed spatial information for precise movement.

Path planning is performed in three stages. First, the robot's current position is added as a node to the upper-layer graph. Second, a high-level path is computed using a shortest-path algorithm to determine the waypoints the robot must pass through. Finally, the lower layer is used to generate a detailed navigation path between waypoints while dynamically considering real-time positioning and obstacles.

This approach enables seamless navigation in large spaces, including multi-story buildings, by effectively managing map transitions through designated horizontal map change points and elevator nodes. The method ensures scalability and flexibility, allowing robots to operate efficiently across extensive areas. Future work will explore automatic generation of upper-layer nodes and real-time updates to further enhance adaptability.

Keywords: Path Planning, Autonomous Mobile Robot, Multi Map Navigation

Contents

List of Tables

List of Figures

제 1 장 Introduction

With advancements in autonomous driving technology, various autonomous robots are being utilized in everyday life. Representative examples include service robots that shuttle between kitchens and tables to deliver food and drinks [1], cleaning robots that roam specific areas to perform cleaning and disinfection tasks [2], and logistics robots that transport goods in large warehouses [3].

These robots typically operate within designated areas, performing simple, repetitive tasks. As the demand for autonomous robots continues to grow, the size of the areas each robot must manage has also expanded. Whereas traditional service robots were confined to operating within the boundaries of a small restaurant, they are now required to transport food or goods across large shopping malls with multiple restaurants or multi-story office buildings [4].

The expansion of operational areas has introduced several technical challenges. One such challenge is the inability to cover all areas using a single map. Autonomous robots rely on pre-constructed maps to determine their current location and calculate routes to their destinations. However, if a large operational area is represented using a single map, the computational load becomes excessively high, leading to inefficiency. A solution to this problem is dividing the operational area into multiple maps. This approach reduces the size of each map, enabling more efficient localization and easier maintenance.

However, when the starting point and destination are located on different maps, conventional path-planning methods such as A* or RRT cannot compute a path. These methods operate under the assumption that both points lie within the same map. Therefore, a path-planning algorithm for new data structure and multiple maps, is necessary. The paper [5] proposes a path-planning method for navigating between different floors of multi-storey buildings. In this study, the robot's operational area is represented as a hybrid map composed of multiple

layers. The upper layers contain topological information, such as the building and floor details of specific locations, while the lowest layer holds metric information, such as an occupancy grid map. Using this hybrid map, the robot performs hierarchical path planning to identify the waypoints required for navigation.

Based on the aforementioned study, this report explains the implementation of a path-planning method for navigating between arbitrary start and destination points across multiple maps. This is one of the tasks the author worked on at the compay "Bear Robotics". The path-planning process consists of two main stages: 1) High-level path planning, which operates on a graph composed of nodes that connect different maps. This stage determines which maps to traverse and which nodes to pass through to move from the starting point to the destination. 2) Low-level path planning, which calculates the path within a single map to travel between nodes. By combining high-level and low-level path planning, the method ensures efficient navigation across multiple maps while managing the complexity of larger operational areas.

The remainder of this report is organized as follows: Section **??** of chapter **??** details the data structure to represent operational areas and path planning method across multiple maps. Section **??** of chapter **??** presents examples of hybrid map and calculatd path. Section **??** of chapter **??** discuss the pros and cons of the method. Finally, Chapter **??** concludes the report.

제 2 장 Body

제 1 절 Methodology

The implemented path-planning method operates on a two-layer hybrid map. The lower layer consists of occupancy grid maps, each of which is uniquely identified by its ID and has its own independent coordinate system. During navigation, the robot uses a map corresponding to its current position which is represented by the coordinate system of that map. If the horizontal area is too large and must be divided into multiple maps, overlapping regions are introduced between adjacent maps. These common regions ensure seamless localization when the robot switches from one map to another, allowing for accurate position estimation even after switching maps.

The upper layer consists of a graph, where nodes are categorized into three types:

1. Destination: Represents final destinations such as tables in a restaurant or storage locations in a warehouse.

2. Horizontal Map Change Point: Represents transition points between different maps on the same floor. These points exist within the overlapping regions of two maps and are conceptually identical locations but have separate nodes for each map. Multiple points can connect two specific maps, and a single point may connect three or more maps.

3. Elevator: Represents the location of a specific elevator on a given floor. Although an elevator may serve multiple floors, a separate node is defined for each floor's boarding point.

Each node contains various information such as location coordinates, floor number and the map ID to which it belongs.

To make a graph, nodes should be connected by edges. The conditions for creating edges between nodes are as follows:

1. Within the same map: Nodes on the same map are connected. The edge weight between two nodes depend on the length of the path between them. The path is calculated using an occupancy grid map in the lower layer and traditional path planning methods such as A*.

2. Between horizontal map change points: Nodes representing the same location on different maps are connected. As the actual distance between the two nodes is 0, the edge weights between these nodes are set to 0.

3. Between elevator nodes: Nodes representing the same elevator across different floors are connected. Because the robot does not move while elevator is moving, the edge weights between these nodes are also set to 0.

An example of this graph structure can be seen in Figure **??**.

Path planning consists of three main stages. The first stage is to add the current location as a new node of the "destination" type to the upper-layer graph. The edge is connected between this new node and other nodes in the same map according to the edge generation rule. This step can be skipped if the starting point is not the current position but an existing node.

In the second step, the shortest path from the initiating node to the destination node is calculated in the upper layer. For this a single-source single-destination shortest path algorithm, such as Dijkstra' s algorithm, is used. This stage determines the waypoints the robot must pass through to reach its destination. Importantly, this step is performed only once, before the robot begins its journey.

The third stage is to perform global path planning in the lower layer to navigate from the current position to the next waypoint. This step is similar to a single map navigation that calculates global paths and local paths using real-time information, such as the robot's current location and obstacles. This process continues until the robot reaches the final destination.

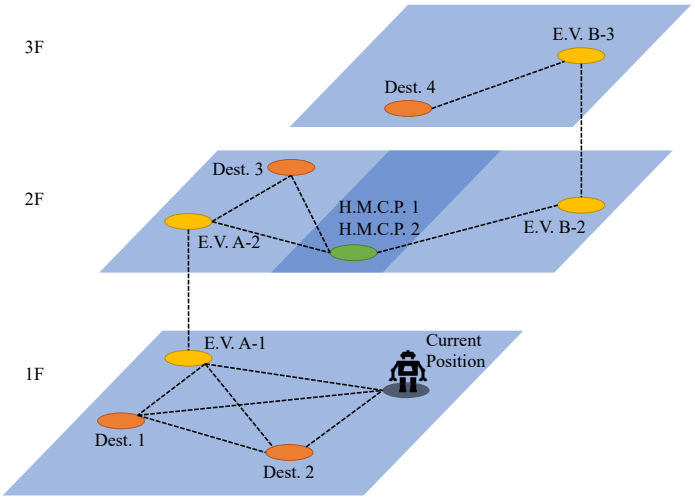

Figure 2.1: This figure illustrates an example of the upper layer of a hybrid map for a three-story building. The building is structured such that the 1st and 2nd floors are connected by Elevator A, while the 2nd and 3rd floors are connected by Elevator B. On the 1st floor, where the robot is currently located, the map includes two destination nodes and one elevator node representing Elevator A. The black node represents the robot's current location. Since it is a temporary node, it will be removed when the path planning is finished. The 2nd floor is divided into two separate maps: the left map contains one destination node, one elevator node for Elevator A, and one horizontal map change point (H.M.C.P. 1) that connects the left and right maps. The right map includes one elevator node for Elevator B and one horizontal map change point (H.M.C.P. 2) that connects the two maps on this floor. Although H.M.C.P. 1 and H.M.C.P. 2 are defined in the coordinate systems of their respective maps, they represent the same physical location. On the 3rd floor, the map includes one elevator node for Elevator B and one destination node.

제 2 절 Example of Implementation

An example is illustrated in Figure **??**. Suppose the destinations are given as Dest. 4, Dest. 3, Dest. 2, and Dest. 1. First, the robot's current position is added to the graph, as shown in the figure. Then, Dijkstra' s algorithm is used to calculate the shortest path from the current position to Dest. 4. The resulting node array is "E.V A-1, E.V A-2, H.M.C.P 1, H.M.C.P. 2, E.V B-2, E.V B-3, Dest. 4". The process is repeated by updating the starting point to the current destination and calculating the shortest path to the next destination. The results of these calculations can be found in Table **??**.

After high-level path planning is completed, low-level path planning is performed. Figure **??** illustrates the robot performing path planning to reach the first waypoint, E.V A-1. Low-level path planning continues iteratively until no waypoints remain.

제 3 절 Discussion

Unlike [5], this path-planning method uses only two layers. In the previous study, nodes were structured at several levels, such as buildings, floors, and regions. On the other hand, this implementation uses a single semantic layer, where each node contains all the necessary information, such as building ID, floor number, map ID, etc. The reason for not using multiple levels is to improve scalability. As the number of properties such as floor and region increases, it will become increasingly difficult to distinguish hierarchical levels and manage topological graphs. To avoid this complexity, all semantic nodes are placed at the same level, and the relationships between them are handled through edge connections. This approach makes the system more flexible and scalable.

This method allows navigation system to easily integrate with other elements by using waypoints. The robot's screen can display the current waypoint as the destination, making users to intuitively understand its movement. Additionally, specific actions to reach a given waypoint can be easily created using finite state machines (FSM) or behavior trees. For example, if the next two waypoints are elevators on the first floor and the second floor, a behavior tree could be structured as follows: "Move", "Board Elevator", "Press 2nd Floor Button" → "Wait" → "Exit Elevator".

However, there are drawbacks. One major problem is that it takes a long time to create the upper layer in advance. Horizontal map change points and elevators must be specified manually. Moreover, since all nodes within the same map must be interconnected, the graph size increases significantly as the number of nodes increases. As the number and type of nodes increase, system maintenance will become more difficult. Additionally, if a starting point is the robot's current position, robot needs time to add a new node to the graph. It makes loading time longer and leads to a bad user experience. Therefore, it is important to divide the space using a reasonable number of maps, ensuring that there are not too many nodes within a single map while still avoiding excessive maps overall.

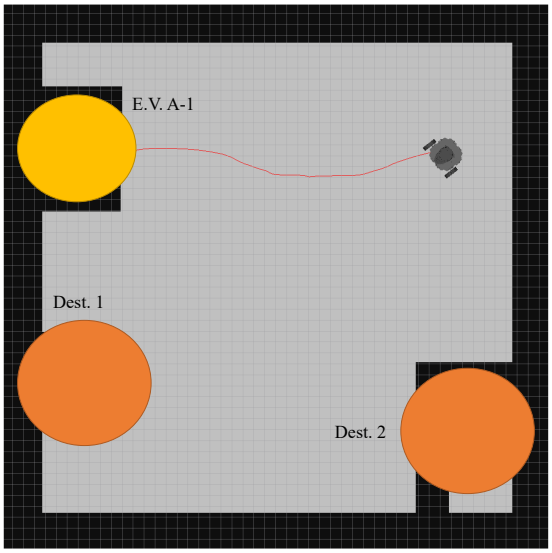

Figure 2.2: Example of low-level path planning from ”Current Position” to the first waypoint, ”E.V. A-1”.

Table 2.1: Results of high-level path planning of given destinations.

Start Node	End Node	Waypoints
”Current Position”	”Dest. 4”	”E.V. A-1”, ”E.V. A-2”, ”H.M.C.P. 1”, ”H.M.C.P. 2”, ”E.V. B-2”, ”E.V. B-3”, ”Dest. 4”
”Dest. 4”	”Dest. 3”	”E.V. B-3”, ”E.V. B-2”, ”H.M.C.P. 2”, ”H.M.C.P 1”, ”Dest. 3”
”Dest. 3”	”Dest. 2”	”E.V. A-2”, ”E.V. A-1”, ”Dest. 2”
”Dest. 2”	”Dest. 1”	”Dest. 1”

제 3 장 Conclusion

This report explains the path planning method I implemented at Bear Robotics. The method was used in situations where multiple maps were required, such as in multi-floor buildings. The proposed approach utilizes a new map structure consisting of two layers. By performing hierarchical path planning within each layer, the system can calculate paths between different maps. High level path planning provides the necessary waypoints to move to other maps, such as the elevator. Low level path planning provides paths or trajectories using existing methodologies such as A* or RRT. This method allows the robot to navigate freely across large spaces and reach their destinations accurately, even in multi-floor buildings. In addition, the waypoints from high level path planning can clearly show the robot's path to users. In the future, this approach can be further improved by automating the generation of upper-layer nodes or dynamically updating the upper layer with real-time position data.

References

[1] "Servi +." Accessed: 2025-01-11. (), [Online]. Available: https://www.bearrobotics.ai/servi-plus (visited on 01/11/2025).

[2] J. Bok. "병원 방역도 인공지능 로봇이." Accessed: 2025-01-11. (2022), [Online]. Available: http://sjbnews.com/news/news.php?number=749827 (visited on 01/11/2025).

[3] "Carti 100." Accessed: 2025-01-11. (), [Online]. Available: https://www.bearrobotics.ai/carti (visited on 01/11/2025).

[4] "Enhanced hospitality with servi lift." Accessed: 2025-01-11. (), [Online]. Available: https://www.bearrobotics.ai/hotels (visited on 01/11/2025).

[5] Q. Zhang, X. Wu, B. Liu, A. H. Adiwahono, T. A. Dung, and T. W. Chang, "A hierarchical topological planner for multi-storey building navigation," in *2015 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, 2015, pp. 731–736. DOI: 10.1109/AIM.2015.7222624.

국문초록

본 보고서에서는 여러 개의 지도를 활용하는 로봇을 위한 계층적 경로 계획 기법을 제안한다. 제안된 방법은 두 개의 layer로 구성된 하이브리드 맵 구조를 사용한다. 상위 layer는 그래프 구조로, 지도 간의 전환과 주요 경유지를 나타내며, 하위 layer는 점유 그리드 맵을 활용하여 정밀한 이동을 가능하게 한다.

경로 계획은 세 단계로 이루어진다. 첫째, 로봇의 현재 위치를 상위 layer의 그래프에 새로운 노드로 추가한다. 둘째, 최단 경로 알고리즘을 사용하여 로봇이 거쳐야 할 waypoint를 계산한다. 셋째, 하위 layer에서 실시간 위치 및 장애물 정보를 고려하여 waypoint 간의 세부 이동 경로를 생성한다.

이 방법은 수평 지도 전환 지점(Horizontal Map Change Point)과 엘리베이터 노드를 활용하여 넓은 공간 및 다층 건물에서도 원활한 이동을 가능하게 한다. 또한 확장성과 유연성을 보장하여 로봇이 광범위한 환경에서도 효과적으로 작동할 수 있도록 한다. 향후 연구에서는 상위 layer의 노드를 자동으로 생성하거나 실시간으로 업데이트하는 방식을 도입하여 적응성을 더욱 향상할 예정이다.

주요어: 경로 계획, 자율 이동 로봇, 다중 지도 네비게이션