

# Path Planning Using 3D Grid Representation in Complex Environment

Biao ZHANG<sup>1,\*</sup>, Masaru ADACHI<sup>2</sup>, Qixin CAO<sup>3</sup>

<sup>1</sup>*Research Institute of Robotics, Shanghai Jiao Tong University, Shanghai 200240, China*

<sup>2</sup>*Yaskawa Electric Corporation, 12-1, Ohtemachi, Kokurakita-ku, Kitakyushu, Fukuoka 803-8530, Japan*

<sup>3</sup>*State Key Laboratory of Mechanical System and Vibration, Shanghai Jiao Tong University, Shanghai 200240, China*

## Abstract

To tackle with mobile robot path planning problem, we developed a novel path planning method utilized 3D grid map. Firstly we construct point cloud map using mobile robot equipped with 3D-LRF, then we transform point cloud into octree data structure, getting 3D grid map. We take mobile robots size and motion into consideration, develop a D\* lite algorithm based path planning method which can generate 3D trajectory of mobile robot in 3D maps directly. We constructed the 3D grid maps of different environments with many obstacles and generalized passable paths, we tested this method with both simulations and experiments, the results validated the reliability and practicality of this method.

*Keywords:* Path Planning; 3D Grid Map; Mobile Robot; 3D Point Cloud

## 1 Introduction

Autonomous mobile robots are playing an increasingly important role in our daily life, to carry out complex tasks they must have a detailed perception of environments as well as the ability to navigate themselves. Path planning is a fundamentally important issue in robot navigation; its purpose is to find the optimal path from a specific position to the destination in a given environment with minimum cost of the path.

With 3D sensor being more and more popular, constructing environment model and planning path with 3D information attracts a lot researchers [5, 6]. In [9], Jochen Kläß builds 3D grid maps and proposes Monte Carlo localization with probabilistic observation models using 2D and 3D sensor fusion data. In [11], Kai M. Wurm presents an approach for modeling 3D environments based on octree using probabilistic occupancy estimation. A relatively sophisticated planning approach for grid represented map in robotics researches is the A\* algorithm [14, 15]. It reduces

---

\*Corresponding author.

*Email address:* [zhang126biao@126.com](mailto:zhang126biao@126.com) (Biao ZHANG).

the computation cost of Dijkstra’s algorithm [16] by involving the heuristic searching. However, the A\* algorithm assumes a priori knowledge of the map, which makes it restrict to known environment. Anthony Stentz and Sven Koenig have done separately a lot of researches on D\* and D\* Lite family, which is able to operate in unknown or partial known environment. These approaches generate feasible path by assuming the unknown grid a certain value. And when the true status of grid contradicts the assumption, it is capable of re-planning with updated map information. Ideally, with the enrichment of map knowledge, the algorithm will always find a feasible path between start and goal grid if it does exist. However, these approaches are either treat the robot as an ideal point with no size or define a grid which is big enough hold a robot inside. In the latter cases, the resolution of grid will be too low to describe terrain features if the robot’s size is not small.

We approach this problem by designing a structured system. Firstly we collect point cloud data with 3D-LRF (lase range finder), then we register different point clouds together to obtain complete point cloud map, then octree structure is utilized to transform point cloud into 3D grid map. We develop a path plan method taking account of robot’s real size which is capable of navigation without a priori knowledge of the 3D grid map, and test it in both simulation and real conditions.

The reminder of the paper is structured as follows. The next section briefly describes the point cloud mapping procedure with 3D-LRF, followed by octree structure transformation. The path plan method in 3D grid map is introduced in section 3. Section 4 presents the simulation results as well as experiments. Finally, we conclude and plan for future work.

## 2 3D-LRF Based Mapping

### 2.1 Point cloud map building

The robot used in our experiment is mainly composed by a mobile base, a LRF for data collection and a motor used for rotating LRF as well as a laptop computer for calculation, processing and display, the robot platform is shown on the left side of Fig. 2.

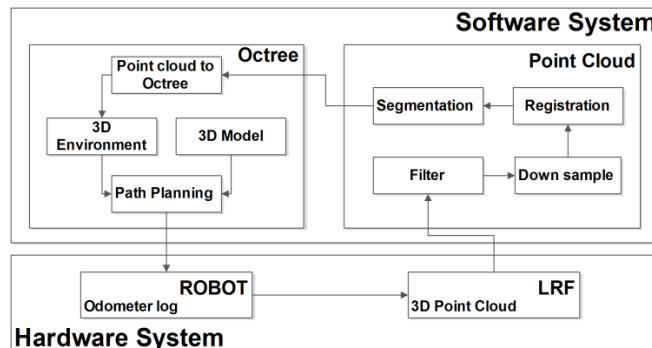


Fig. 1: Functional modules of map building system

The function modules of our map building system is shown in Fig. 1, there are mainly two sub-systems: hardware system and software system. Software system includes point cloud processing flow and octree processing flow while hardware system is composed by Mobile robot and LRF.

The robot was driven through our building and was stopped at some certain places to record the odometer log and point cloud data, while moving to the next place it would process the data collected from last position. The point cloud map is built through registering point clouds gathered from different positions using NDT (Normal Distributions Transform) algorithm [4], the right side of Fig. 2 is the point cloud map generated by mobile robot, points with different colors are gathered from different positions.

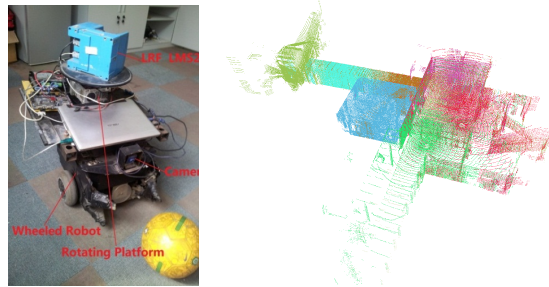


Fig. 2: 3D-LRF equipped Robot platform and 3D point cloud map

## 2.2 Octree based grid map construction

Point cloud is an inferior structure for describing environment which contains a lot of redundant information and consumes a lot of memory. We use octree structure to optimize the environment modeling method.

An octree is a hierarchical tree-based data structure for managing sparse 3D data [12]. Each node in an octree represents the space contained in a cubic volume, usually called a voxel. This volume is recursively subdivided into eight sub volumes until a given minimum voxel size is reached. In a point region octree, the node stores an explicit 3-dimensional point, which is the center of the subdivision for that node; the point defines one of the corners for each of the eight children. In an MX octree, the subdivision point is implicitly the center of the space the node represents. The root node of a point region octree can represent infinite space; the root node of an MX octree must represent a finite bounded space so that the implicit centers are well-defined [12]. We use MX octree considering it is highly memory efficient.

We transform the complete point cloud map generated from 2.1 into octree structure, as shown in Fig. 3, the original point cloud data is shown on the left while octree structure data is shown on the right side. A clearer version of 3D grid map is shown in Fig. 4. We performed a qualitative analysis of the 3D grid map, after down sampling the point cloud map in Fig. 2 is about 31MB without apparent reduce in accuracy, and the 3D grid map in Fig. 3 is approximately 937KB, which can be handle easily by most mobile robots.

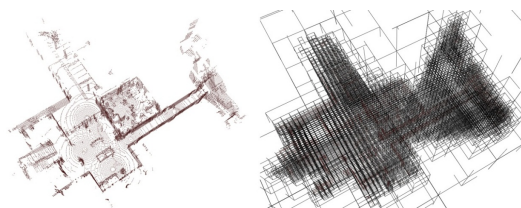


Fig. 3: Original point cloud map with roof removed (left) and its octree structure (right)

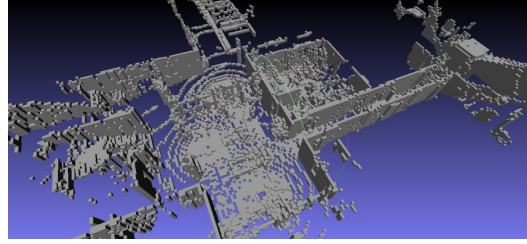


Fig. 4: Generated 3D Grid map with roof removed

### 3 Path Planning

We use the 3D grid map presented to represent the environment, the grid-based path planning algorithm operating on this 3D grid map is originated from D\* lite algorithm.

#### 3.1 Path planning algorithm in ideal condition

First, we ignore the shape and motion of mobile robot and take it as a leaf grid. Let  $G = (S, E)$  be the set representing the grid map, where  $S$  is a set of grids which represent possible robot position,  $E$  is a set of edges indicating transitions between these positions. Function  $adj(s)$  defines the set of grids that is adjacent to grid  $s$ .  $Succ(s) \subseteq S$  Denotes the set of successors of grid  $s$  and  $Pred(s) \subseteq S$  denotes the set of predecessors of grid  $s$ .  $h(s)$  Indicates the average height of grid  $s$  while  $v(s)$  denotes whether robot has perceived grid  $s$ .

When given an initial grid  $s_{start} \in S$  to a goal grid  $s_{goal} \in S$ , the planning method should be able to generate the shortest path between these two grids. To do this, an estimate  $g(s)$  of the path cost from the initial state to each state  $s$  is stored.

$$g(s) = \begin{cases} 0 & \text{if } s = s_{start} \\ \min_{s' \in Pred(s)} (g(s') + c(s', s)) & \text{otherwise} \end{cases} \quad (1)$$

Meanwhile, a heuristic estimate of its path cost to the goal  $h(s, s_{goal})$  is used to force the search to go towards the goal, which can reduce computation of grids which actually is not needed when planning the path. Function  $0 < c(s, s') \leq \infty$  is used to store the cost it takes for robot to transit from grid  $s$  to  $s'$ .  $c(s, s') = \infty$  Means the edge between grids  $s$  and  $s'$  is not traversable. Obviously,  $h(s, s_{goal})$  and  $c(s, s')$  should obey the triangle inequality.

$$h(s, s_{goal}) \leq h(s', s_{goal}) + c(s, s') \quad (2)$$

Initially, for all  $s \in S$ ,  $h(s)$  is set to be 0 and  $v(s)$  is set to be *false*, which indicates the grid states are all unknown for robot. At the beginning of the search,  $g(s_{start}) = 0$  and  $s$  is labeled as *CLOSED* which means the grid is not needed to be visited again. For the entire grid  $s \in adj(s_{start})$  we calculate  $g(s)$  and  $h(s, s_{goal})$ . If grid  $s$  is not labeled as *CLOSED*, then it is added into a list named *OPEN* list. The *OPEN* list stores grids the robot is to visit. It is sorted by the  $k(s)$ .

$$k(s) = g(s) + h(s, s_{goal}) \quad (3)$$

The grid with minimum  $k(s)$  is stored at the top of the OPEN list. After  $k(s)$  where  $s \in adj(s_{start})$  has been pushed into OPEN list. The grid  $s$  at the top of the list is popped out and its state is labeled as CLOSED. For  $s' \in adj(s)$ , we calculate  $g(s')$  and  $h(s', s_{goal})$ , then add those not labeled as CLOSED into OPEN list. After reordering the OPEN list, we pop out the grid with minimum  $k(s)$  and the search continues. Repeat these steps and stop until  $s_{goal}$  is labeled which means path has been found or OPEN list is empty which means there is no feasible path between  $s_{start}$  and  $s_{goal}$ . Finally, we can generate the path by visiting  $Pred(s)$  from  $s_{goal}$ .

### 3.2 Robot involvement

The Ideal searching algorithm treats the robot as a leaf grid with no real size. However, the size of robot will put a critical influence on the result of path planning and it should be taken into account during the search. The robot status includes two main contents: size and motion.

With robot size taken into account in path planning algorithm, we use 3D model of mobile robot simulate real conditions, we also simplify the 3D model by transform the model into octree structure as displayed in Fig. 5. The 3D grids on the right represent the robot on the left during path planning to check if there is enough space in the propagated path.

The second thing to consider is robot motion, Fig. 6 shows a simplified 2D case, in which the width is only two thirds of the length of the robot. Although the traversability normally depends on the width of the robot, if we use a circle whose radius is robots width, there exist potential risks when robot tries to change its direction. On the other hand, if we use a circle whose radius is robots length, we will abandon possible paths during path planning.

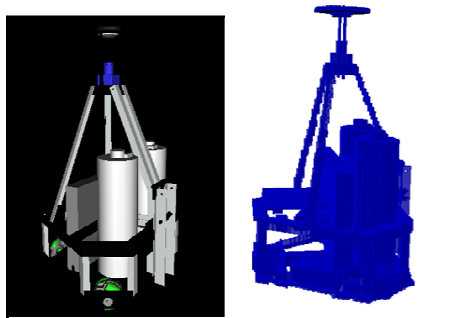


Fig. 5: Mobile robot model (left) and 3D grid model (right)

To avoid such situations, we utilize 3D collision check between grid representation of robot and 3D grid map to judge whether there is enough space for robot to carry out planned motion. The modified process of path planning can be concluded as steps below:

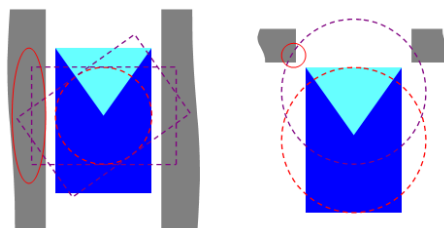


Fig. 6: Collision result from robot motion in 2D condition

**Step 1:** Propagate a traversable grid among the adjacent grids of current grid and go to **Step 2**.

**Step 2:** Check if the propagated grid is to the diagonal direction from the current grid. If it is, then go to **Step 3**. Otherwise go to **Step 4**.

**Step 3:** Check if there is enough space for robot to carry out planned motion. Abandon propagated grid if robot is not able to rotate or move at current position and go to **Step 1**. If motion is possible, go to **Step 4**.

**Step 4:** Check if there is enough space for robot to stop at the propagated grid. If it is, then go to **Step 5**. Otherwise go back to **Step 1**.

**Step 5:** Move to the propagated grid in the previous step. If the current grid is goal grid, the mission is accomplished. Go to **Step 6**. Otherwise, jump back to **Step 1**.

**Step 6:** Generate path from goal grid to start grid.

## 4 Simulations and Experiments

Experiments were conducted to demonstrate the advantage of using 3D grid representation for path planning in complex environments. First we implement our aforementioned 3D path planning algorithm according to and run the simulated navigation in pre-generated 3D Grid map. According to the results, cost function was modified and employed in the real experiments.

As illustrated in Fig. 7, the real scene on the left is located in our laboratory, segmented into 53240 grids with each grid representing an area of 0.10.10.1m<sup>3</sup>. This 3D grid map was then utilized to drive the robot from a same position to a specified target position, in our case the robot was assigned a task to move from room to hall through the door. The trajectory generated by our planning method for mobile robot is shown in right side of Fig. 7 with corresponding positions marked in red.

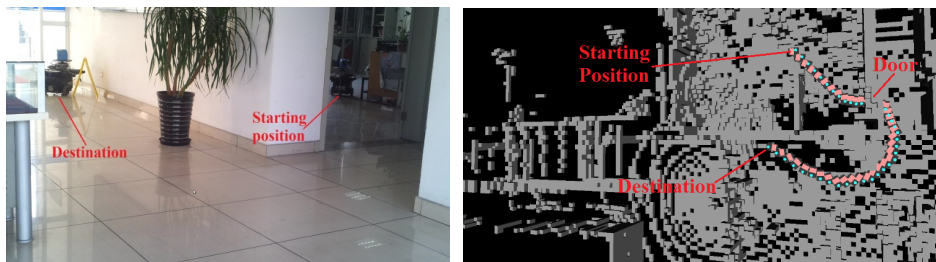


Fig. 7: Mobile robot model (left) and 3D grid model (right)

Additionally, we conducted out several path planning experiments to testify our planning method, one of them was implemented in a relatively more complex environment with 6 obstacles randomly arranged in a 3.23.8m<sup>2</sup> area, the experiment setup is shown on the left side of Fig 7. The three-wheeled RoboCup soccer robot equipped with HSV and its 3D model we used are shown respectively in the upper left corner and upper left corner. After we appointed the start and destination in 3D grid map, two passable paths were calculated, shown in Fig. 8. These paths avoid collision with main obstacle 3 from different direction, guide robot move towards destination and avoid other encountered obstacles.

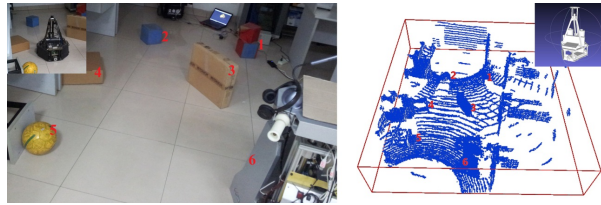


Fig. 8: The real scene (left) and corresponding 3D grid map (right)

As is shown in Fig. 7 and Fig. 8, utilizing 3D point cloud and taking robot size and motion into consideration show great advantages than traditional planning algorithms. On one hand, the environment structure can be extracted using grid represented map, saving a lot of memory space without losing structured information. On the other hand, this method can plan path and avoid collision with obstacles in the environment at the same time.

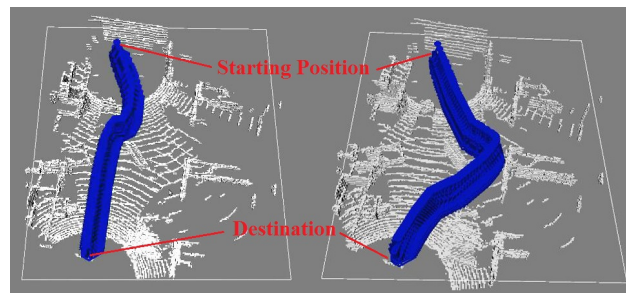


Fig. 9: Two passable trajectories of mobile robot in 3D grid map

## 5 Conclusion and Future Work

We have presented a comprehensive system for path planning using 3D grid map based on 3D point clouds. The point cloud map is built through registering point clouds gathered from different positions with 3D-LRF. After transforming the point cloud into octree structure, we get 3D grid map which consumed a lot less memory and would be a lot more pragmatic for mobile robots. Our path planning method take robot size and motion into account, after 6-step process it will generate the passible trajectories in 3D grid represented map for mobile robot.

The current path planning method uses only 3D-LRF as sensor to perceive environment, the color information and other sensor message are ignored which becomes a defect of our current system. In our future work we will try to employ sensor fusion information for both map construction and path planning.

## Acknowledgement

This work is supported by National Natural Science Foundation of China (Grant No. 61273331), National 863 Key Program on Advanced Manufacturing Technology Field of China (Grant No. 2012AA041403) and Yaskawa Electric Corporation.

## References

- [1] S. Thrun, W. Burgard, and D. Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping. in Proc. of IEEE Intl. Conference on Robotics and Automation (ICRA), San Francisco, pp. 321-328, 2000.
- [2] HU Ruifei, YIN Guofu, TAN Ying, CAI Peng. Cooperative clustering based on grid and density. *Chinese Journal of Mechanical Engineering*. 19 (4), pp. 544-547, 2006.
- [3] Jiajun Gu; Qixin Cao, Path planning for mobile robot in a 2.5-dimensional grid-based map, *The Industrial Robot* vol. 38, no. 3, pp. 315, 2011.
- [4] M. Magnusson. The Three-Dimensional Normal-Distributions Transform an Efficient Representation for Registration, Surface Analysis, and Loop Detection, [D]. dissertation, Örebro University. 2009.
- [5] Yong Wang, Weidong Chen, Jingchuan Wang. Hybrid map-based navigation for intelligent wheelchair. in Proc. 2011 IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, May 9-13. 2011.
- [6] Radu Bogdan Rusu, Zoltan Csaba Marton, Nico Blodow, Andreas Holzbach, Michael Beetz. Model-based and Learned Semantic Object Labeling in 3D Point Cloud Maps of Kitchen Environments. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2009.
- [7] W Wang, B Shen, X Wang. Research and implementation of a hogging algorithm. *Chinese Journal of Mechanical Engineering*. 18 (2), pp. 307-312, 2005.
- [8] Jiajun Gu, Qixin Cao, Path planning using hybrid grid representation on rough terrain , *Industrial Robot*, 2009 , Vol. 36, No. 5, pp. 497-502.
- [9] I. Dryanovski, W. Morris, and J. Xiao. Multi-volume occupancy grids: An efficient probabilistic 3D mapping model for micro aerial vehicles, in Proc. IROS, 2010, pp. 1553-1559. 2010.
- [10] E. Fong, Representing a 3-D environment with a 2.5-D map structure. in Proc. of IEEE/RSJ Intl. Conference on Intelligent robots and Systems, Las Vegas, pp. 2986-2991, 2003.
- [11] K. M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, and W. Burgard. OctoMap: A probabilistic, flexible, and compact 3D map representation for robotic systems. In Proc. of the ICRA 2010. Anchorage, AK, USA, May 2010.
- [12] J. Wilhelms and A. Van Gelder. Octrees for faster isosurface generation, *ACM Trans. Graph.*, vol. 11, no. 3, pp. 201-227, 1992.
- [13] H. P. Moravec. Robot spatial perception by stereoscopic vision and 3d evidence grids. Technical Report CMU-RI-TR-96-34, CMU Robotics Institute, 1996.
- [14] Hart, P.; Nilsson, N.; and Rafael, B.. A formal basis for the heuristic determination of minimum cost paths. *IEEE trans. Sys. Sci. and Cyb.* 4, pp. 100-107, 1968.
- [15] Nilsson, N., Principles of Artificial Intelligence. Tioga Publishing Company, 1980.
- [16] E. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, pp. 269-271, 1959.