Mobile Robot Exploration Based on Rapidly-exploring Random Trees and Dynamic Window Approach

Taiping Zeng[1,2,3,4], Bailu Si[4]
[1]State Key Laboratory of Robotics, Shenyang Institute of Automation,
Chinese Academy of Sciences, Shenyang 110016, China
[2]Institutes for Robotics and Intelligent Manufacturing,
Chinese Academy of Sciences, Shenyang 110016, China
[3]University of Chinese Academy of Sciences, Beijing 100049, China
[4]School of Systems Science, Beijing Normal University, Beijing 100049, China
e-mail:zengtaiping.ac@gmail.com, bailusi@bnu.edu.cn

***Abstract*—Exploration is a critical function for autonomous mobile robots. Traditionally, the entire map has to be processed to extract frontiers and perform path planning. However, as the robot explores the environment, the map grows over time, and increasing computational resources are required, especially for large-scale environments. Moreover, only a few methods focus on the exploration on point cloud maps. Here, I present a new practical method to autonomous mobile robot exploration based on a sparse, relatively small-size point cloud local map, which combines Rapidly-exploring Random Tree (RRT) and dynamic window approach (DWA) algorithm together. The local map is built from the consecutive inputs of raw point clouds using an inexpensive 3D sensor, i.e. Kinect V2. Frontiers are effectively detected and local path planning is performed by RRT algorithm directly on unordered point cloud local maps. Motion planning is performed online by DWA to avoid obstacles and direct a nonholonomic mobile robot towards frontiers separating known environments from unknown environments. Embedded with simultaneous localization and mapping (SLAM) system of my previous research, the performance of the proposed method is evaluated in a large-scale customized virtual environment with a size of $33 \times 29 \times 6$m using Gazebo as the robotic simulator. The results suggest that the proposed algorithm can accurately direct the nonholonomic mobile robot to unexplored environments in real time. Also, it successfully helps build a coherent semi-metric topological map. The proposed algorithm shows great efficient performance and is suitable for both static and dynamic environments. Combination of RRT and DWA algorithm on point clouds, as a general approach, can be extended to generic 3D nonplanar physical environments. Videos of the experiments can be found at https://youtu.be/0i766fhs9Ds.**

***Index Terms*—Mobile Robots, Autonomous Exploration, Point Clouds, Motion Planning, Rapidly-exploring Random Tree, Dynamic Window Approach**

I. Introduction

Mobile robot exploration is essential for autonomous mobile robots safely and efficiently operating in many challenging application scenarios, such as transportation, inspection, surveillance, and search and rescue. The main goal of mobile robot exploration is to answer the question of choosing where a robot should go next, and select appropriate control actions to reach the next desired position. It leads to increase the knowledge of the robot by visiting unknown or uncertain environments [1].

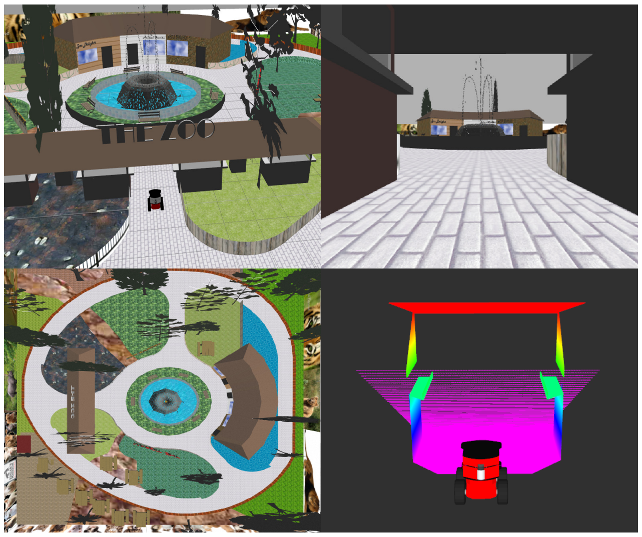


Fig. 1. An instant at the beginning of the simulation experiment. The top left image depicts the robot, i.e. Pioneer 3-AT, at the gate. The top right shows what the robot sees. The bottom left image is the top-down shot of the simulation environment. Point clouds are presented on the bottom right corner, when the robot sees the environment through Kinect V2.

Although many works have been proposed to solve the indoor exploration problem using laser scanner and deal with navigation problem on point cloud maps, only a few methods focus on the exploration on point cloud maps. An online computed random tree is applied to find the best branch and make micro aerial vehicle (MAV) follow the first edge of this branch to explore unmapped space for 3D environments [2]. A point cloud map of environment is built from stereo cameras on the MAV. The algorithm is tested in two simulation scenarios, the apartment setup and the bridge model. It is also evaluated in a closed room with a size of $9 \times 7 \times 2m$. However, the online random tree still needs to search the whole 3D occupancy map. As the map grows, the MAV with limited resources will be unaffordable. Also, it is only evaluated in a closed room in the physical environment.

In this work, I present a new practical method to autonomous mobile robot exploration on a sparse, relatively small-size point cloud map. The input of raw point clouds is provided by an inexpensive 3D sensor, i.e. Kinect V2. The raw point clouds are filtered to a sparse one. Then it is concatenated with the previous local map. After cropped to limited size, the output point cloud is the current local map. The RRT algorithm directly searches on the unordered point cloud local map to detect the frontier points and traversible paths. The mean-shift algorithm [3] is applied to cluster the frontier points without assigning the number of clusters, which largely discards many frontier points which are close to each other. Motion planning is performed by DWA method [4] to online avoid obstacles and control a nonholonomic mobile robot to the desired frontier point, which separates known environments from unknown environments. The proposed method is implemented in C++ based on ROS framework. Embed with my previous SLAM system [5], the performance is evaluated in a large-scale customized virtual environment with a size of $33 \times 29 \times 6$m (See Fig. 1), which uses Gazebo as the robotic simulator. The experimental results suggest that, in each local point cloud map, the proposed method can accurately direct the nonholonomic robot to unknown environments in real time. Moreover, it successfully helps build a coherent semi-metric topological map.

In summary, this paper makes the following contributions:

- Frontiers are detected by RRT algorithm on a local point cloud map in 3D environments. Since the local map always keeps similar size, computational complexity would not increase in this local exploration process, as the exploration time goes.
- The proposed method helps build a semi-metric topological map in a large-scale customized virtual environment with a size of $33 \times 29 \times 6$m using Gazebo as the robotic simulator.
- Combination algorithm of RRT and DWA on point clouds is proposed, which can be considered as a general approach to deal with local planning problems.
- An inexpensive 3D sensor, i.e. Kinect V2, and the car-like nonholonomic robot, i.e. Pioneer 3-AT are chosen in the system for future practical applications, in which a limited range and limited field of view range are considered.
- The point cloud registration is implemented by using robot odometry at close range, not by iterative closest point (ICP) [6], which largely improves the performance.

The remainder of this paper is organized as follows: Section III presents the proposed algorithm in detail. The robotic implementation is described in Section IV. Evaluation in simulation experiment is described in the Section V. Section VI summarizes and concludes the presented work.

II. Related Work

Many algorithms are proposed to solve the exploration problem. Topological methods construct the explored environment as a connectivity graph [7], [8]. It can identify which distinctive place is reached and recognize the explored environments through graph vertices. Potential methods can also be adapted for exploration by gradient descent in a continuous vector field with appropriate boundary conditions [9]. Another method is to use random search techniques to sample space and extend edges in tree-like connectivity. The Sensor-based Random Tree is a typical example [10], [11], which is a variant of Rapidly-exploring Random Tree [12]. Frontier-based exploration is the most common method for mobile robot exploration [13]–[15]. The idea is to move to the boundary between explored and unexplored environments, where the robot can use sensors to perceive more information and expand the knowledge of workspace. The RRT algorithm can also be employed to detect the frontier points on the 2D occupancy grid map, which is generated from laser scanner [16]. After finding the desired frontier point, ready-make ROS packages, the gmapping package [17] and the ROS navigation stack [18], are applied to perform mapping and path planning, respectively. It is only limited to 2D exploration, and the whole 2D occupancy grid map is needed to processed to detect frontiers. As the map grows, it will consume more and more computational resources [19].

Point cloud is a canonical 3D representation. Point cloud can be recorded from LiDAR, depth sensor, and converted from stereo cameras. Other 3D representations can also be converted to point cloud, such as the representation of mesh, volumetric, and depth map [20]. Also, point cloud is close to raw sensor data, and point cloud is a more compact 3D representation than voxel. Many researchers focus on the path planning problem given a 3D point cloud map. The 3-D tensor voting framework is used to solve the 2.5-D navigation problem using raw point cloud as input for mobile robots [21]. A practical approach is presented in [22] to perform global motion planning and terrain assessment on point cloud maps for mobile robots in generic 3D environments. The trajectories are directly computed on unordered point cloud maps using RRT algorithm. After trajectory optimization, the robot is navigated to the goal by following the planned path.

III. Proposed Approach

The exploration strategy in local maps can be split into four modules: the local mapping module, the RRT-based frontier detection module, the frontier filter module, and DWA-based motion planning module. The overall schematic diagram of the exploration strategy is shown in Fig. 2. Kinect V2 serves as eyes of the robot to see the environment. The point cloud is provided to construct a local point cloud map. Then frontiers are detected by the RRT algorithm and filtered by the frontier filter module. Finally, the DWA-based motion planning module sends the action command to the robot, which continuously controls the robot in the environment.

A. Local Mapping

The local mapping module only maps the nearby environment of the robot. The local mapping process is shown in Fig. 3. The raw point cloud is recorded from Kinect V2.

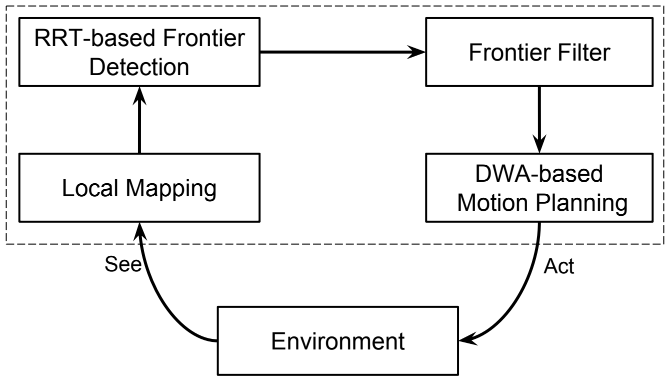


Fig. 2. Overall schematic diagram of the mobile robot exploration algorithm. The robot sees the environment through a 3D sensor by the representation of point clouds. Local map is built from consecutive inputs of point clouds. The RRT algorithm is performed on the local point cloud map to detect frontiers. The frontier points are clustered and assigned as the next designed point. The DWA-based motion planning directs the robot following the traversible path towards the unknown environments.

To ensure the computational perform, point clouds are downsampled to sparse point clouds. According to the odometry information, the movement of the robot from the previous time step to the current time step can be calculated, which is used to transform the coordinate of the previous local map to the current coordinate of the robot. Then the downsampled point

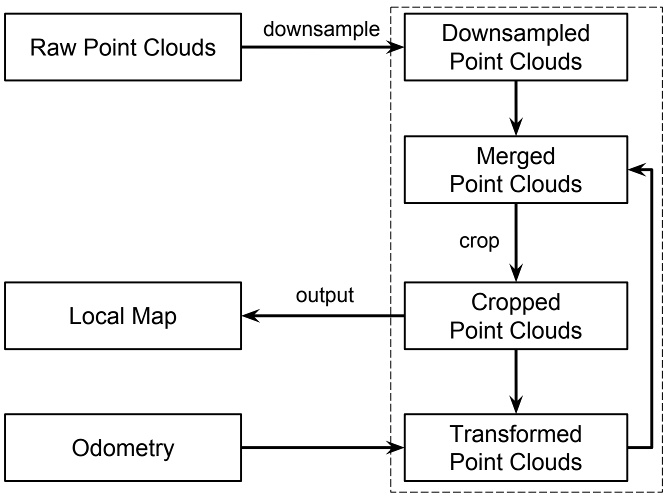


Fig. 3. The local map building process. The raw point cloud is downsampled. According to odometry information, coordinates of the previous cropped point clouds, namely previous local point cloud map, are transformed. Then the current downsampled point clouds are merged with the previous transformed local map. Finally the merged point clouds are cropped and output as the current local map.

cloud is merged with the previous transformed local map. To limit the size of the local map, the merged point clouds are cropped from three different dimensions at each iteration. It only keeps the familiar high of the robot. Finally, the cropped point clouds output as the current local map.

B. RRT-based Frontier Detection

The RRT-based frontier detection aims at finding the frontier points and traversible paths on the local point cloud map. To ensure the requirement of computational resources, the RRT algorithm only searches on the local point cloud map. In order to improve the performance, a variant of RRT, namely RRT* [23], is actually employed in the work. For simplicity, only the RRT-based frontier detection algorithm is shown in Algorithm 1.

The RRT-based frontier detection algorithm starts from a single initial vertex $V = x_{init}$, where the initial point is the origin point. The edge is set to empty $E = \phi$. For each loop, a random point x_{rand} is generated from a defined range. Here, the sample range is a rectangular area in the robot. Then, the nearest point $x_{nearest}$ is found from the current tree. A new point is generated by **Steer** function according to a predefined step length from $x_{nearest}$ and x_{rand}. The **ObstacleFreeCheck** function checks whether there are obstacles between $x_{nearest}$ and x_{new}. Afterwards the **PointCloudCheck** function checks whether x_{new} is close to the local point cloud map. After fixed iterations, a random tree is generated on the local point cloud map.

Algorithm 1: RRT-based Frontier Detection Algorithm

```
1  $V \leftarrow x_{init}; E \leftarrow \phi$
2  for $i$ = 1,...,$n$ do
3      $x_{rand} \leftarrow$ SampleFree;
4      $x_{nearest} \leftarrow$ Nearest$(G = (V, E), x_{rand})$;
5      $x_{new} \leftarrow$ Steer$(x_{nearest}, x_{rand})$;
6      if ObstacleFreeCheck$(map, x_{nearest}, x_{new})$ then
7          if PointCloudCheck$(map, x_{new})$ then
8              $V \leftarrow V \cup x_{new}; E \leftarrow E \cup (x_{nearest}, x_{new})$
9          end
10     end
11 end
```

C. Frontier Filter

As the random search algorithm will generate too many frontier points which are extremely close to each other, it needs to discard the redundant frontier points and find the appropriate frontier points. The frontier filter receives the detected frontier points from the RRT-based frontier detection module. Firstly, the frontier points are clustered by mean-shift algorithm without assigning the number of clusters. The desired frontier point, which is nearest to the center of each cluster, is selected. The other frontier points are removed. Sometimes, on the intersection of roads, multiple desired frontier points can be found. The coordinates of the frontier points relative to the position of the robot would be as selection criteria, as the coordinates provide the orientation information of the frontier points. During the high-level decision making process, the preferred conditions would determine which frontier points would be chosen.

The mean-shift algorithm is detailed described in the following. The frontier points x_i, $i = 1, ..., n$ on a 2-dimensional space R^2. The kernel density estimate can be defined as

$$f(x) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\left\|\frac{x - x_i}{h}\right\|^2\right) \quad x \in R^2 \tag{1}$$

with window radius h, and kernel $K(x)$. The kernel $K(x) = c_{k,d}k(\|x\|^2)$, and $c_{k,d}$ assures $K(x)$ integrates to 1. When the gradient function $\nabla f(x) = 0$, the mean shift item can be found

$$\mathbf{m}_h(x) = \frac{\sum_{i=1}^{n} x_i g\left(\left\|\frac{x-x_i}{h}\right\|^2\right)}{\sum_{i=1}^{n} g\left(\left\|\frac{x-x_i}{h}\right\|^2\right)} - x. \tag{2}$$

For each iteration, the mean shift vector $\mathbf{m}_h(x^t)$ is computed. Then the points are updated by

$$x_{t+1} = x^t + \mathbf{m}_h(x^t). \tag{3}$$

The gradient makes the mean shift vector points toward the direction of maximum increase in the density, until convergence to a point.

D. DWA-based Motion Planning

The dynamic window approach is a velocity-based local planner [4]. As the dynamic window approach is derived from the dynamics of the robot, it directly calculates the optimal collision-free velocity for a robot to reach the goal. It translates a frontier point into a series of velocity (v, w) commands for a mobile robot. The DWA-based motion planning algorithm is presented in algorithm 2. Three basic information is given, namely robot pose, the frontier point, and the robot kinematic. In this work, the robot is on the origin. The robot kinematic sets the max translational and rotational velocities, accelerations of translation and rotation, resolutions of translational and rotational velocities. The dynamic step time and predicted duration serve as the parameters of the dynamic window. The point cloud is provided by the local mapping module.

The main goals of DWA-based motion planning algorithm are to calculate a valid velocity search space, and then select the optimal velocity command for the mobile robot. The allowable velocity is calculated from the robot pose and robot kinematic, which also ensures the robot continuously travels in the environment. The set of velocities generate safe trajectories, which allow the robot to stop before collision. The set of velocities can be achieved in the next time slice given the dynamics of the robot, namely dynamic window. The optimal velocity is chosen to maximize the function of the velocity, the heading closest to the goal, and the overlap between the straight line to the goal and possible trajectories (See algorithm 2).

Algorithm 2: DWA-based Motion Planning Algorithm

```
Given robotPose, frontierPoint, robotKinematic
pointcloud = readPointcloud()
allowable_v = generateWindow(robotPoseV, robotKinematic)
allowable_w = generateWindow(robotPoseW, robotKinematic)
for each v in allowable_v do
    for each w in allowable_w do
        breakDist = calculateBreakingDistance(v)
        dist = find_dist(v,w,pointcloud,robotKinematic)
        if dist $\geq$ breakDist then
            heading = calcHeadingEval(v,w,frontierPoint)
            path = calcStraightLineEval(v,w,frontierPoint)
            clearance = (dist-breakDist)/(dmax - breakDist)
            cost = costFunction(clearance,heading,v,w,path)
            if cost $\geq$ optimal then
                best_v = v
                best_w = w
                optimal = cost
            end
        end
    end
end
```

IV. Robotic Implementation

The proposed mobile robot exploration on point clouds algorithm is implemented in the C++ language and is run in the Robot Operating System (ROS) Indigo on Ubuntu 14.04 LTS (Trusty). The SLAM system of my previous work in [5] is embedded in the implemented system. The SLAM system receives images and odometry as the input, and builds a semi-metric topological map, also called cognitive map, when the proposed exploration algorithm consecutively sends a set of action commands to control the robot operation in the environment.

In the mobile robot exploration algorithm, the local mapping module receives point cloud messages directly from Kinect V2 mounted on the robot, which is separately implemented as a ROS node. The local mapping module outputs the local point cloud map as a ROS message. The RRT-based frontier detection module and the frontier filter module are together implemented in a separate ROS node. It receives the local point cloud map, and searches frontier points on the local point cloud map, then filters to find the desired frontier point with preferred conditions. The desired frontier point can be autonomously selected according to threshold of the distance and the rotation angle for each intersection, or just at random. Here, for this robotic implementation, the preferred conditions, like left or right, are sent manually during the simulation process, which would lead the algorithm to find the optimal frontier point. Therefore, the system is actually a semi-autonomous exploration robotic system. The DWA-based motion planning module is implemented as another ROS node, which receives the next desired frontier point and local point cloud map as the inputs. Derived from the mobile robot kinematics, it directly sends the optimal velocity command to the robot.

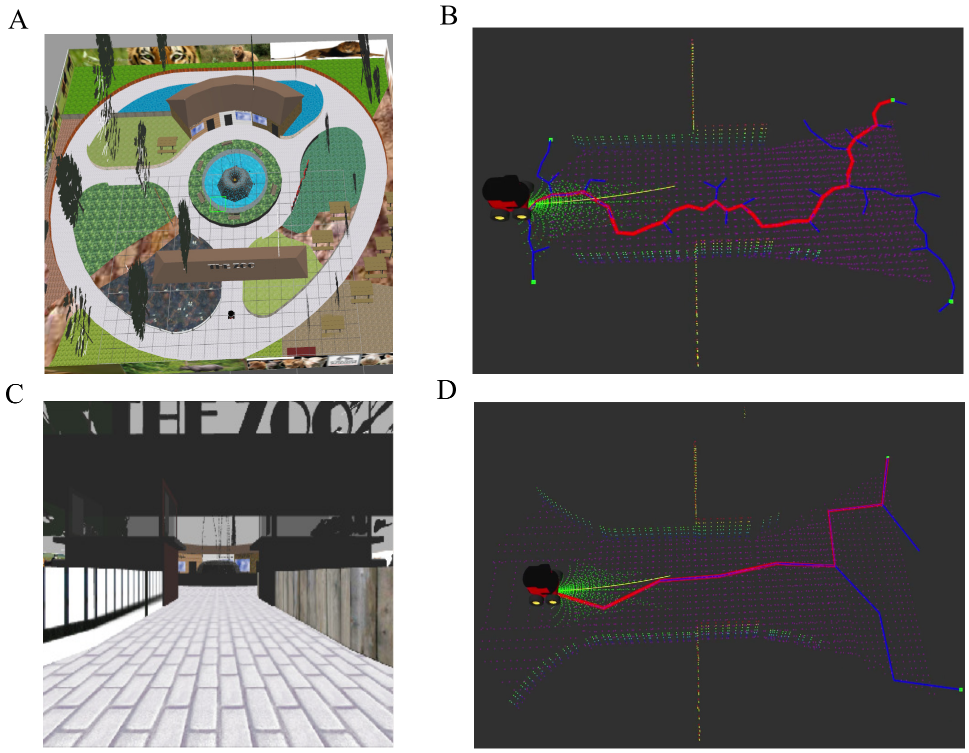


Fig. 4. Screenshots of mobile robot exploration algorithm. (A) The position of the robot is in front of the gate. (B) The mobile robot exploration algorithm with a smaller random step is shown in rviz provided by ROS. The point cloud shows the local map. The blue line is generated by RRT-based frontier detection algorithm. The green points are frontiers after mean-shift algorithm filtering. The red line is the selected traversible path according to the desired frontier point. The little green points in front of the robot are trajectories generated by DWA-based motion planning algorithm. The yellow one is generated the optimal velocity. (C) The image is seen by the robot. (D) The mobile robot exploration algorithm with a larger random step is shown in rviz.

V. Simulation Based Evaluation

In order to systematically evaluate the performance of the proposed mobile robot exploration algorithm, simulation experiments are performed. The proposed mobile robot exploration algorithm using an inexpensive 3D sensor could freely guide the online robot exploration in the simulation environment with a size of $33 \times 29 \times 6$m, and successfully build a coherent topological map (see videos of the experiments at https://youtu.be/0i766fhs9Ds).

A. Simulation Setup

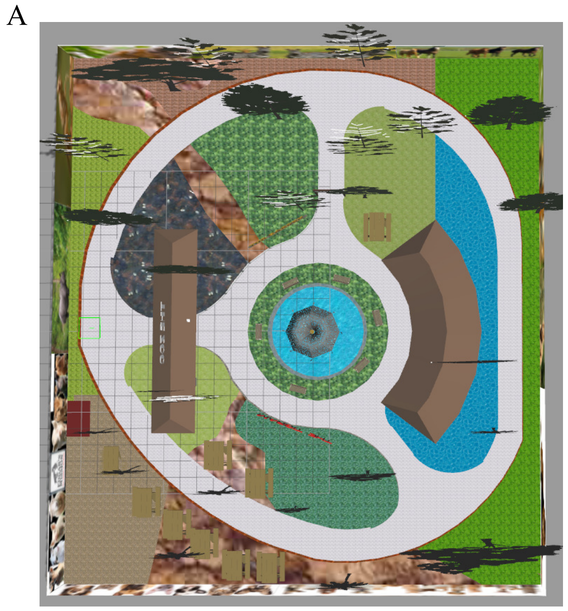
As the proposed mobile robot exploration algorithm works in a closed loop with the robot's perception, local mapping, local path planning, and motion control, a detailed and realistic simulation is needed. The Gazebo simulation environment is used along with a nonholonomic robot, i.e. Pioneer 3-AT. The nonholonomic robot is car-like robots, which is more significant for practical applications. Kinect V2 is chosen as a 3D sensor to see the environment, as the price is much cheaper than another 3D sensor, like LiDAR. It also leads to consider the limited range and limited field of view range of the sensor during the operation process of the robot. Kinect V2 can also provide visual information. The proposed mobile robot exploration algorithm together with my previous SLAM work is evaluated on a personal computer with 3.4 GHz six-core Intel i7 processor and 64 GB memory.

The simulation scenario refers to a $33 \times 29 \times 6$m outdoor environment. The customized environment simulated in Gazebo is a typical park scenario, which includes the zoo gate, trees with different color, size and shape, ticket office, fountain, walls, paths, etc. The whole environment is closed by walls, which attaches different animal images, like tiger, lion, cat, rhinoceros, etc. Hence, the environment is called "**The Zoo**".

B. Simulation Results

I evaluated the proposed mobile robot exploration algorithm in the simulation environment, i.e. **The Zoo**. The mobile robot performs motion planning on the local point cloud map. A set of motion command is sent to the motion control module to make the mobile robot explore the simulation environment towards the unknown space.

Fig. 4 visually shows how the proposed mobile robot exploration algorithm works during the simulation process. The robot stands in front of the gate shown in Fig. 4A. What the robot sees is shown in Fig. 4C. The mobile robot exploration algorithm with a smaller random step is presented in rviz, shown in Fig. 4B. The traversible paths are generated by RRT-based frontier detection algorithm, shown in blue lines. The paths avoid the obstacles, and are just on the local point cloud map explored area. The random step is set to 0.2m. The total iterations for the RRT-based frontier detection are 300. After clustering the frontier points by the mean-



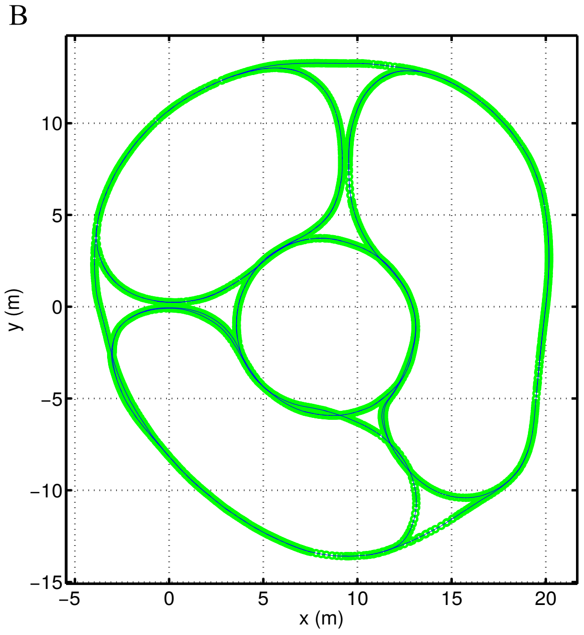


Fig. 5. An overhead view of the simulation environment and the semi-metric topological map. (A) The evaluation environment is a 33 × 29 × 6m outdoor environment. The white part is the runnable path. Other colors show the buildings, trees, walls, gardens, etc. (B) The semi-metric topological map. The green thick line comprises topological graph vertices, and the blue thin line consists of links between connected vertices.

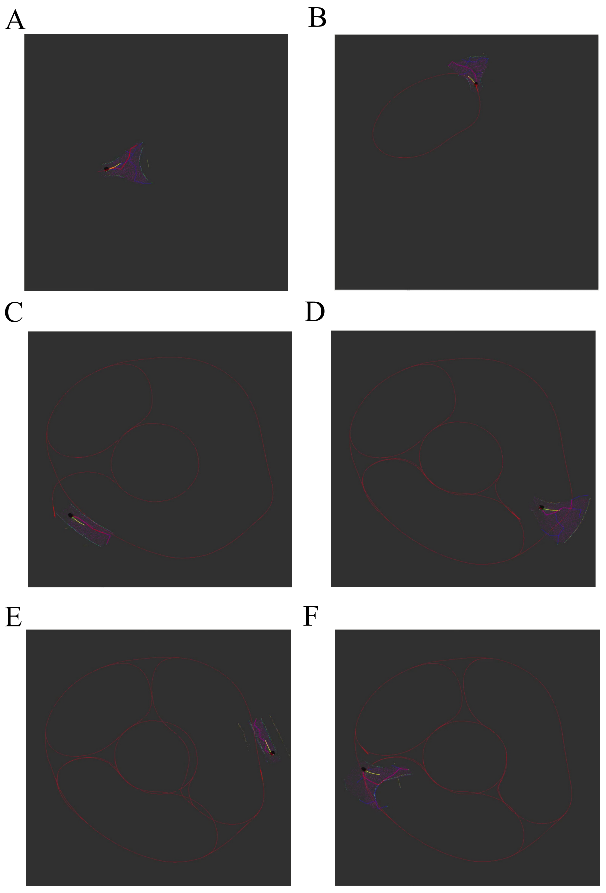


Fig. 6. Topological map evolution for the simulation environment visualized with ROS rviz. The odometry is poor due to the accumulated error and regularly leads to localization errors which are corrected by loop closures.

shift algorithm, the frontier points are green points on the edge of the local point cloud map. The preferred conditions are given manually, which guides the robot to choose which frontier point at each intersection. According to the desired frontier point, the selected path is shown by the red line. DWA-based motion planning algorithm generates the achievable trajectories displayed by little green points. The yellow line presents the optimal velocity trajectory. After all this process, the robot receives movement commands towards the desired frontier points. While, in the practical application, a small random step would consume more computational resources, a larger step usually is chosen. In Fig. 4D, the random step size of the RRT-based frontier detection is set to 1.0m.

Fig. 6 shows the evolution of the topological map for the simulation environment, for intervals of one-sixth of the simulation, ending in the final map shown in Fig. 6F. The topological map is displayed by red lines within ROS rviz. Since the odometry is poor due to the accumulated error, it regularly leads to localization errors which are corrected by loop closures. Fig. 6A shows the mobile robot in the beginning position. The mobile robot is located in the intersections shown in Fig. 6B and D, which can find multiple runnable paths on the local point cloud maps using the proposed algorithm. However, the mobile robot can find only one runnable way when it runs in the long and narrow lane shown in Fig. 6C and E.

Together with my previous SLAM work, the semi-metric topological map is created shown in Fig. 5B. The green thick line comprises by topological vertices, and vertices are connected by the blue thin line. As the physical distance is considered, the map is actually a semi-metric topological map.

The performance of the proposed algorithm can be visually compared with the overhead view of the ground truth map (Fig. 5A) by naked eyes. The overall layout of the road network of the simulation environment is captured by the semi-metric topological map. The semi-metric topological map correctly represents all loop closures, curves, and intersections. As the odometry is with uncertainty, the map is slightly different with the ground truth map. In summary, the semi-metric topological map is consistent with the ground truth map of the simulation environment.

VI. Summary & Conclusion

Within this work, a mobile robot exploration algorithm is proposed that is capable of exploring a previously unknown environment, only directly on a sparse, relatively small-size point cloud local map. Since the proposed method performs RRT-based frontier detection only on the local map, the requirement of computational resources would not increase as the map expands over time. A semi-metric topological map is successfully constructed of the explored environment, which is a customized virtual environment. Compared with the ground truth environment, the map captures the overall layout of the road in the simulation environment. The mobile robot exploration algorithm online computes good exploration paths, finds the desired frontier point, and helps the robot avoid obstacles, finally towards to the unknown environment. The nonholonomic robot and an inexpensive 3D sensor are used for more practical application in the future. Furthermore, considering the real-time performance of the proposed mobile robot exploration algorithm, it is suitable for both static and dynamic environments, and also, as a general method, it can be extended for generic 3D nonplanar physical environments.

In the future, the proposed algorithm will be considered as a basis to further develop autonomous mobile robots system. The proposed mobile robot exploration algorithm can help the mobile robot find possible runnable paths and safely travel in the dynamic environments based on the local point cloud map, even when the mobile robot gets lost in the global localization. Also, it can be used as the underlying algorithm to ensure the normal operation of the mobile robot, and receives high-level commands, such as take the left side, or go straight, like humans. Together with the cognitive map, it is very likely to develop fully autonomous exploration and navigation system at low computational cost and complexity.

Acknowledgment

This work is supported by the National Key Research and Development Program of China (No. 2016YFC0801808).

References

[1] R. J. Shade, "Choosing where to go: mobile robot exploration," Ph.D. dissertation, University of Oxford, 2011.

[2] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon" next-best-view" planner for 3d exploration," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1462–1468.

[3] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE transactions on pattern analysis and machine intelligence*, vol. 17, no. 8, pp. 790–799, 1995.

[4] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[5] T. Zeng and B. Si, "Cognitive mapping based on conjunctive representations of space and movement," *Frontiers in neurorobotics*, vol. 11, p. 61, 2017.

[6] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," *Image and vision computing*, vol. 10, no. 3, pp. 145–155, 1992.

[7] G. Dudek, M. Jenkin, E. Milios, and D. Wilkes, "Robotic exploration as graph construction," *IEEE transactions on robotics and automation*, vol. 7, no. 6, pp. 859–865, 1991.

[8] B. Kuipers and Y.-T. Byun, "A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations," *Robotics and autonomous systems*, vol. 8, no. 1-2, pp. 47–63, 1991.

[9] E. P. e Silva Jr, P. M. Engel, M. Trevisan, and M. A. Idiart, "Exploration method using harmonic functions," *Robotics and Autonomous Systems*, vol. 40, no. 1, pp. 25–42, 2002.

[10] G. Oriolo, M. Vendittelli, L. Freda, and G. Troso, "The srt method: Randomized strategies for exploration," in *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, vol. 5. IEEE, 2004, pp. 4688–4694.

[11] L. Freda and G. Oriolo, "Frontier-based probabilistic strategies for sensor-based exploration," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. IEEE, 2005, pp. 3881–3887.

[12] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," 1998.

[13] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Computational Intelligence in Robotics and Automation, 1997. CIRA'97., Proceedings., 1997 IEEE International Symposium on*. IEEE, 1997, pp. 146–151.

[14] B. Yamauchi', "Frontier-based exploration using multiple robots," in *Proceedings of the second international conference on Autonomous agents*. ACM, 1998, pp. 47–53.

[15] Y. Wang, A. Liang, and H. Guan, "Frontier-based multi-robot map exploration using particle swarm optimization," in *Swarm Intelligence (SIS), 2011 IEEE Symposium on*. IEEE, 2011, pp. 1–6.

[16] H. Umari and S. Mukhopadhyay, "Autonomous robotic exploration based on multiple rapidly-exploring randomized trees," in *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE, 2017, pp. 1396–1402.

[17] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[18] E. Marder-Eppstein, E. Berger, T. Foote, B. Gerkey, and K. Konolige, "The office marathon: Robust navigation in an indoor office environment," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 300–307.

[19] P. Senarathne, D. Wang, Z. Wang, and Q. Chen, "Efficient frontier detection and management for robot exploration," in *Cyber Technology in Automation, Control and Intelligent Systems (CYBER), 2013 IEEE 3rd Annual International Conference on*. IEEE, 2013, pp. 114–119.

[20] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, vol. 1, no. 2, p. 4, 2017.

[21] M. Liu, "Robotic online path planning on point cloud," *IEEE transactions on cybernetics*, vol. 46, no. 5, pp. 1217–1228, 2016.

[22] P. Krüsi, P. Furgale, M. Bosse, and R. Siegwart, "Driving on point clouds: Motion planning, trajectory optimization, and terrain assessment in generic nonplanar environments," *Journal of Field Robotics*, vol. 34, no. 5, pp. 940–984, 2017.

[23] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.