Remembering Exploration

Jungtae Kim
Department of Computer Science and Engineering,
Pohang University of Science and Technology, Korea
postman@postech.ac.kr

Daijin Kim
Department of Computer Science and Engineering,
Pohang University of Science and Technology, Korea
dkim@postech.ac.kr

Abstract - For autonomous mobile robots the robot needs to have some ability like navigation which is mainly consisted of localization and map-building. The method to find new area for map-building is called exploration. In this paper we introduce new exploration method which searches unknown area by reducing the redundant path. Unlike the previous study, which focuses on the trade-off between exploration and exploitation not on exploration itself, we suggest novel exploration method, Remembering Exploration (RE), which emphasizes exploration itself. For comparing our method with the previous methods we have an experiment in grid-world simulation, and for showing the possibility of RE method in real environment we also have 2D world simulation experiment. The experiment results show that our method has the priority over the previous method.

Keywords - Robot, Navigation, Exploration, Remembering Exploration

1. Introduction

The concerning about the mobile robot increases these days with development of robot technology. Traditionally the research about the mobile robot has two major part: one for the controlled mobile robot and the autonomous mobile robot. So far the controlled mobile robot rather than the other has been developed very much because of its a little easier development and the demand for the tool to help human's work. But these days people want to more active machines, which find out the necessary work and do it oneself as well as help people's behavior. This request makes the researcher to have more concerning about the autonomous mobile robot.

For the autonomous mobile robot the various abilities are needed, and one of them is navigation. The navigation ability also needs many detail technologies: for example, localization, mapping, planning, and obstacle avoidance. Our topic, exploration problem, is very important problem with exploitation in research area about planning. Exploration means finding the unknown information, and exploitation means getting the goal using the already known information as much as possible. In most case exploration is not useful but just consumes the resource, but with a little probability we get high profit from it. Therefore we must do exploration for unknown area not only exploitation for that.

Unlike the most research focussing on the trade-off between exploration and exploitation [1], in this paper we focus on the efficiency of exploration itself. This research area is sometimes called the coverage planning, and most well-known application of it is an auto-cleaning robot. Even though there is no map, the robot moves whole area without any missing space. To cover whole area the robot need to know where the passed place is and where the not yet passed place is. Moreover, the robot has to move as economically as possible because of its cost like time, electric, and etc.

There are several methods for the exploration problem; for example, Random Walk [2], Following a Left-Wall, Counter-based Exploration [3], Spiral Exploration [4], and Distance Transfer [5]. In this paper we suggest new exploration method, Remembering Exploration (RE), which can solve the previous problems; To cover all area, RE method remembers the space information that which space we know and which space we do not. To move following the optimal exploration path, RE method uses the well-known algorithms like breadth-first-search, depth-first-search, and depth-limited-search [6]. In next section, we explain it more detail.

2. Remembering Exploration (RE)

Our suggested exploration method finds the optimal path by reducing the redundant path as much as possible. We call this method Remembering Exploration because during the exploration we remember the unknown areas.

Let's explain about the RE method with Fig. 1. There is a grid-world; White room means empty space. Blue room means the blocked room, where if we go to the room, then we restart at initial position. Deep green with number shows the path. Light green without number shows the unknown area, where later we will come back here. If there is no blocked neighbor room, we can move to upper, lower, left, and right neighbor room with step 1.

In Fig. 1(a) the current position is the room with number 1, and as we see we can move to one of the left, right, and lower room but not the upper room, which is blocked. In the current position, the distances to the three unknown rooms are same, so we choose one randomly. Here we choose the right direction, and the result is Fig. 1(b). In Fig. 1(b) among the previous three unknown rooms we now known one room and find three another unknown rooms. From the room with number 2 the new three unknown rooms are the nearest unknown rooms. One of them is selected randomly, and the

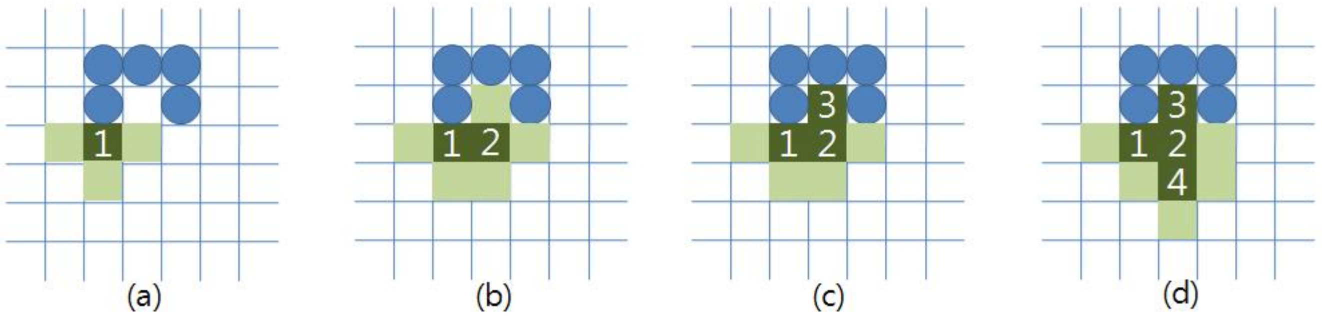

Fig. 1. Exploration algorithm in grid world; Blue circle means the blocked space, dark green with number means the path, and light green means unknown space. Move to the nearest unknown space.

result is like Fig. 1(c). In Fig. 1(c) the current position is the room with number 3. The upper, left and right rooms from the room with number 3 are blocked, and the lower room is already known, so there is no new unknown room. The nearest unknown rooms from the room with number 3 are the lower and right rooms of the room with number 2. Even if the two rooms are not neighbor of the room with number 3, there is no problem. Among the two nearest unknown rooms we choose one randomly. The lower room of the room with number 2 is selected, but because the room and the current position is not neighbor each other we can't move directly by 1 step. Therefore we have to find the shortest path. The two unknown rooms are neighborhood of known rooms, so here we use the known information to find the path from the current position to the neighbor room of the unknown rooms. For that we use well-known search method like breadth-first-search, depth-first-search, or depth-limited-search [6], then the path through the room with number 3 and the room with number 2 is the shortest path. Following the path, we get the result like Fig. 1(d). The room with number 4 is changed from an unknown room to a known room, and the right and lower rooms of the room with number 4 are added into the unknown rooms.

This procedure is repeated until all area is covered or known. When we find the final goal position we choose whether we try to this procedure more or we stop the procedure and exploit the already known information. RE method has the little redundant moves because of the following two reasons; RE searches the nearest unknown room and RE uses the shortest path to move to the nearest unknown room. In the next section we summary the previous procedure.

3. Algorithm

The pseudo-code of RE is like Algorithm 1. In the initial part the following procedure(2-6 lines) is done. There are a known area set (L_K) and an unknown area set (L_U), and we empty the two sets. The current position (x_c, y_c) is determined randomly or as a fixed position, and the target position(x_t, y_t) is set same to the current position (x_c, y_c). We register the current position (x_c, y_c) in the unknown area set (L_U).

After initializing, we repeat the following procedure(8-16 lines) until reaching the final position. Among the neighbor areas of the current position (x_c, y_c) we move one of the areas, which do not belong to the known area set (L_K), to the unknown area set (L_U). If the current position (x_c, y_c) reaches the target position (x_t, y_t), then we move the target position (x_t, y_t) from the unknown area set (L_U) to the known area set (L_K), and set the new target position (x_t, y_t) as the nearest area in the unknown area set (L_U). Going to the target position (x_t, y_t), we move to the neighbor area, which is in the union of known area set (L_K) and unknown area set (L_U). We then update the moved position as the new current position (x_c, y_c).

```
Algorithm 1. Remembering Exploration
INITIALIZE
    Empty the unknown land $L_U$.
    Empty the known land $L_K$.
    Set the current position $(x_c, y_c)$ randomly or fixed.
    Set the target position $(x_t, y_t)$ same to $(x_c, y_c)$.
    Register $(x_c, y_c)$ in $L_U$.
LOOP
    Register a neighbor land of $(x_c, y_c)$ in $L_U$ if the
    land $\notin L_K$.
    IF $(x_c, y_c) \equiv (x_t, y_t)$
        Transfer $(x_t, y_t)$ from $L_U$ to $L_K$.
        Set $(x_t, y_t)$ same to the nearest land $\in L_U$.
    END_IF
    Move to a neighbor land $\in L_K \cup L_U$ which is on the
    path toward $(x_t, y_t)$.
    Update $(x_c, y_c)$ by new current position.
END_LOOP
```

4. Experiments

We have two experiments like followings. The first one is to find the goal in grid world like Fig. 2. The second one is to explore area in 2D world like Fig. 3. From now we will explain about the two experiments in detail.

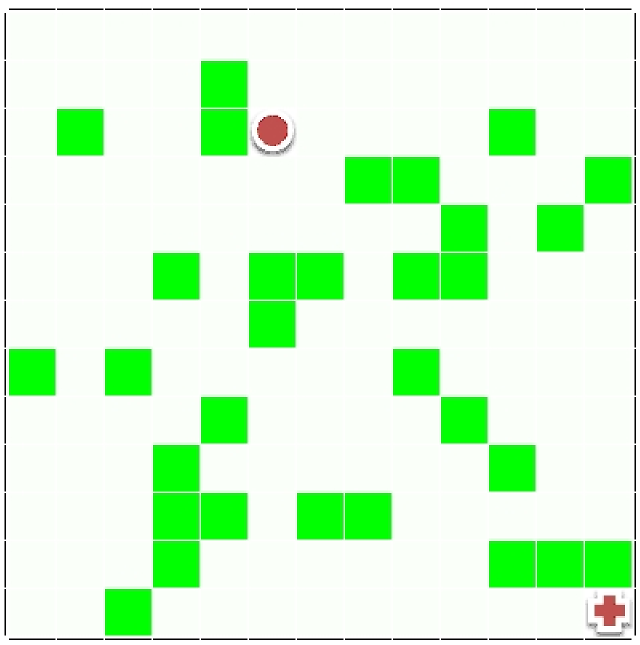

Fig. 2. Grid world with 13x13 rooms: green for the blocked room, red circle for the current position, and red cross for the target position.

4.1. *Grid world experiment*

The grid world experiment is an algorithm test on the 13x13 rooms, totally 169 rooms. In the grid world the white room means the empty space, the green room means the blocked space, the red cross means the final goal, and the red circle means the current position (Fig. 2). The initial position and final position are determined at initial time randomly. If we are not on the border rooms, we can select one of the upper, lower, left, and right direction. Whenever moving, we can move only to a neighbor room, that is, the movable distance is always 1. If the number of moves is over than maximum of moves or we move to the blocked room, we must restart at initial position.

In the condition that the current position is determined and the information about the target position and the blocked rooms is unknown, we compare the performance of our suggested algorithm and the previous method [1], [3], which are like the following; First, Random Walk (RW) algorithm [2] enables to move along one of the upper, lower, left, and right direction (except the case when on the border) randomly. The algorithm is the simplest and shows the various paths. Second, Following a Left-Wall (FLW) algorithm is a well-known method in finding the exit in maze. As we known from the name, we just continue to follow the left-wall from the initial position. Here we assume that the initial position and the final position is connected by the wall. Even though we get the same result if we follow the right-wall, we unify the name for the easy. Third, Counter-based Exploration (CE) algorithm [3] is that whenever we visit a room we increase the number of visit to the room, and then when we choose the next direction we consider the lowest number of visit among the neighbor rooms. Unlike the previous methods which uses only the information of neighborhood and determines the next direction, CE algorithm holds the information about the whole world and updates and uses the information.

For each method, the complexities are following; Let N be the total number of rooms. For the RW method, we can't expect the next position, that is, it is impossible to know the maximal moves. So we skip the analysis of the complexity of RW. For the FLW method, it can move all N area at most. Therefore, its complexity is $\mathcal{O}(N)$. For the CE method, it can cover all N areas as known areas, and to reach each area it can move N steps at most. Therefore, it has $\mathcal{O}(N^2)$ complexity. For the our suggested RE method, it is similar to the CE method. So in the complex analysis RE's complexity is same to the CE's one, $\mathcal{O}(N^2)$. All above results are summarized in the last column of Table 1.

We have a test on the three previous method and our method for finding the path from the initial position to the final position. The performance comparison is done by the criterion of how fast the method finds out the final position and how short the length of the final path, which the method found out, is. After all methods reach the final position, the total moves, the retrial number, and the final path length are recorded and new grid world is rebuilt. The average of the 100 times for this procedure is summarized in the Table 1.

TABLE I

PERFORMANCE COMPARISON WITH THE PREVIOUS METHODS AND OUR METHOD; RANDOM WALK (RW), FOLLOWING A LEFT-WALL (FLW), COUNTER-BASED EXPLORATION (CE), AND REMEMBERING EXPLORATION (RE). EACH COMPLEXITY IS SHOWN IN THE LAST COLUMN.

	# of moves	# of epoch	final path len.	cmplx.
RW	5000	3000	500	- -
FLW	100	10	40	$\mathcal{O}(N)$
CE	800	100	28	$\mathcal{O}(N^2)$
RE	600	50	28	$\mathcal{O}(N^2)$

Among the three previous methods and our suggested method, RW method has a worst result. In total moves and retrial number, the numbers are much bigger than the others, and the final path is also not an optimal path. FLW method finds the final position very fast because it never takes moves more than 169, which is the total number of rooms. But the final path is also not an optimal path. CE method shows prior than RW or FLW method; It finds an almost optimal path. But during the exploration it also moves some redundant path.

The suggested RE method also finds out the optimal path like CE. Moreover, the time which is took to find out the final path is surprisingly 25% shorter than CE method. From this experiment we know that our suggested method has a priority on the previous methods.

4.2. *2D world experiment*

The 2D world experiment is about the performance when the virtual robot with a laser sensor in the real building map

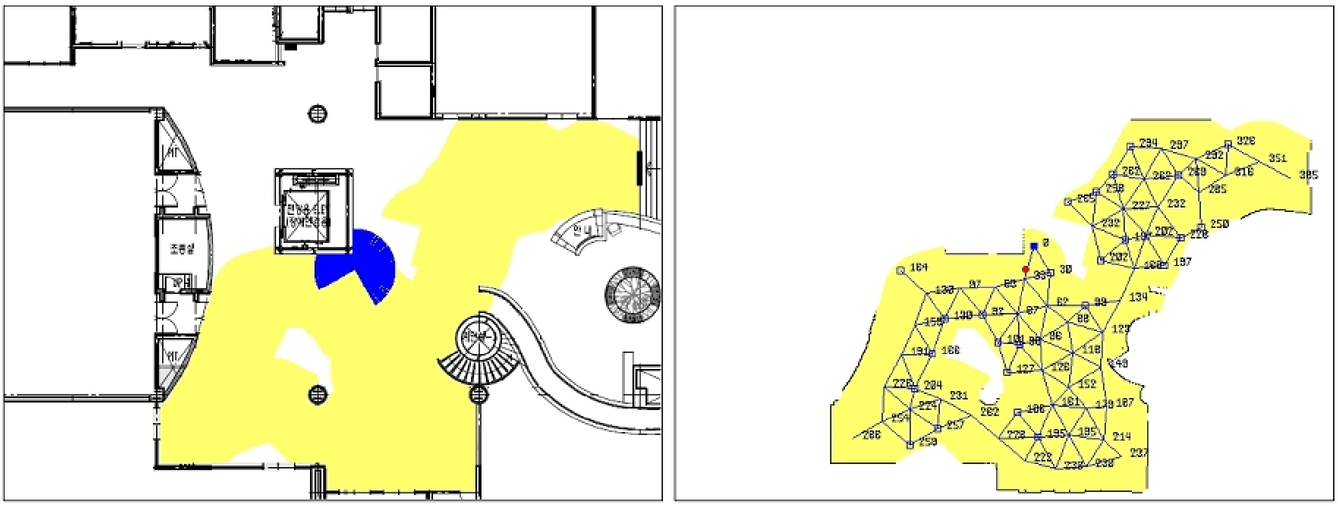

Fig. 3. Exploration on 2D map of real building. [Left] Blue pie shows that the robot scans neighborhood area with laser scanner. Yellow areas mean the passed areas of the robot. [Right] Yellow area shows the known area of the robot. Blue line shows the mesh model. Blue square means the unknown area and red circle means the current position of the robot. The number on the mesh shows the distance from the target position.

moves to explore the area (Fig. 3[Left]). The differences with the previous grid-world experiment are following; First, the number of possible position becomes very bigger or the space becomes the continuous space ($\Re^2$). Therefore, it is impossible to memorize all of the area, so we must find out the method which represents all area with a little information. There are several well-known methods for that, for example the method using the grid model [7] or the method using the mesh model [8]. In this experiment we use the mesh model method (Fig. 3[Right]). Second, unlike the grid-world experiment, we can know the information which exists far away. In this experiment we simulate the robot with a laser sensor, URG-04LX [9] model, which scans 4 meters away at most. The laser sensor scan around 135 degree, so as you see in Fig. 3[Left] the robot can not recognize whole 360 degree area but only know the front 135 degree area. Finally, the goal of this experiment is not to find the specific final position but to know how much the robot covers whole area, that is, the problem is changed from point-to-point planning to coverage planning. It is because usually the exploration work cooperates with map-building, so we want to make the total map as wide as possible.

In real robot experiment whenever exploring the unknown area the robot must do the map-building, localization or both like Simultaneous Localization and Mapping (SLAM). But in this paper we only focus on the exploration, so we assume that other necessary abilities are work well; There is no error in making a map, that is, the map-building is well done, and we know the current position exactly, that is, the localization is well done. Moveover, we assume that our experiment has no sudden obstacle and change, that is, this is an exploration experiment in stationary environment. Without other bad effect, we have this experiment.

In Fig. 3[Right] there is a mesh model with blue lines; The vertex of the mesh shows the position of the robot, and the edge shows the path of the robot. The vertex connected to the other vertex means that they are neighborhood. The blue square means that it is unknown area, and the red circle means indicates the current position.

In the grid world the four neighbor rooms become the neighbor area, but in the 2D world we can not make the concept of the upper, lower, left, and right, so we regard the vertex in the mesh model as the neighbor area and apply the Algorithm 1. In Algorithm 1, the neighbor area of the current position is determined by selecting the middle points which have no obstacle around area. In this experiment we divide the scan area 270 degree into 5 parts, that is, 54 degree for each part. So if there is no obstacle in continuous 54 degree area, we select the middle point as the neighbor position.

In the mesh model the distance is defined by Euclidian distance, and the distance from the goal position is recorded in the each vertex like Fig. 3[Right]. The robot do not move exactly on the edge because the space is continuous and the control of the robot is not perfect. The robot moves with satisfying the following conditions as possible; First, the robot uses as a nearer vertex as possible. Second, the robot prefers the unknown area, which is not far from the current position by the Euclidian distance. Third, the robot prefers the smaller different angle from the current angle to the angle of any unknown area(vertex). When the robot selects the next position, it uses the upper three condition as criterion.

Because the performance comparison of the other algorithm and our algorithm is already done in the previous experiment, and its performance is same in the 2D world experiment, so we need not test about that anymore. Instead of that, we only test about the percentage of coverage area. In this experiment we know that our RE method can cover 95% of whole movable area. The extra 5% area can be reduced by tuning the range of finding the neighbor area.

5. Discussion and Future Work

In this paper we introduce the new exploration method, Remembering Exploration (RE). For comparing with the previous exploration methods we have an experiment for finding the path to the goal in grid world and compare the total number of moves and the final path length. In the result we see that our method is prior to the previous ones. Moreover, to see how well our method works in pseudo-real environment we have an experiment in the 2D world environment, and we get the result that RE method can cover almost all over the region even within very short time.

For the easy of 2D world experiment, we assume that the explored environment is stationary as well as there is no error in making the map. But in real world this assumption is not suitable. So we have to study about the environment where some obstacle are appeared or the arrangement is changed sometimes. With that, we also need to study about the adjustment work for the mesh model when there occurs an loop-closing during making the map.

6. Acknowledgements

The authors would like to thank the Ministry of Education of Korea for its financial support toward the Division of Mechanical and Industrial Engineering, and the Division of Electrical and Computer Engineering at POSTECH through BK21 program.

References

[1] S. B. Thrun, "Efficient exploration in reinforcement learning," Tech. Rep. CMU-CS-92-102, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, 1992.

[2] D. Nguyen and B. Widrow, *The truck backer-upper: an example of self-learning in neural networks*. Cambridge, MA, USA: MIT Press, 1990.

[3] S. B. Thrun, "The role of exploration in learning control," in *Handbook for Intelligent Control: Neural, Fuzzy and Adaptive Approaches*, Florence, Kentucky: Van Nostrand Reinhold, 1992.

[4] E. Gonzalez, O. Alvarez, Y. Diaz, C. Parra, and C. Bustacara, "Bsa: A complete coverage algorithm," in *Proceeding of IEEE Int. Conf. on Robotics and Automation*, (Barcelona, Spain), April 2001.

[5] H. Choset, "Coverage for robotics - a survey of recent results," *Annals of mathematics and Artificial Intelligence*, pp. 11–126, 2001.

[6] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms, Second Edition*. Cambridge, MA, USA: MIT Press, 2001.

[7] A. W. Moore and C. G. Atkeson, "The parti-game algorithm for variable resolution reinforcement learning in multidimensional state spaces," *Machine Learning*, vol. 21, no. 3, pp. 199–233, 1995.

[8] J. C. Latombe, *Robot Motion Planning*. Kluwer Academic, Boston, 1991.

[9] H. Kawata, A. Ohya, S. Yuta, W. Santosh, and T. Mori, "Development of ultra-small lightweight optical range sensor system," in *Proceeding of IEEE Int. Conf. on Intelligent Robots and Systems (IROS '05)*, pp. 3277–3282, 2005.