Path Extraction for Autonomous Mobile Robot Using Skeletonization

Ryuki Higuchi
Department of Electrical and Computer Engineering
Yokohama National University
Tel: +81–45–339–4174
Email: higuchi-ryuki-wc@ynu.jp

Yasutaka Fujimoto
Department of Electrical and Computer Engineering
Yokohama National University
Tel: +81–45–339–4174
Email: fujimoto@ynu.ac.jp

***Abstract*—One of the most important tasks regarding autonomous mobile robots is to plan a path for movement. Given a pre-built map, a robot is able to follow a path that is determined manually by setting way points on the map prior to exploration. However, for cases without pre-built maps, a robot has to derive its path while traveling autonomously and analyzing the surrouding environment. In this paper, we propose a method of path planning for autonomous movement under unknown environments in the case where a pre-built map does not exist. This system successfully determines way points on a road following the skeleton extracted by combining multiple data processing algorithms.**

***Index Terms*—autonomous mobile robot, image processing, skeletonization**

I. Introduction

In recent years, extensive attention has been paid to autonomous mobile robots including driverless cars. Autonomous mobile robots allow us not only to reduce the stress that comes with manual control, but also to access dangerous areas that we are generally unable to. In the terms of method behind autonomous movement, several studies have used a pre-built map for simultaneous localization and mapping (SLAM) to realize autonomous movement by estimating the position and orientation of the robot [1]. The robot will then move along a pre-planned path on a map to reach the destination. In addition, sensors that incorporate a great deal of information, such as three-dimensional light detection and ranging (3D-LiDAR) sensors are generally used to improve accuracy [2] [3]. However, since the fact that creating this kind of map is a very time-consuming process makes it difficult to use the robot in unknown locations. Thus, alternative methods that require little or no pre-built map have been also developed [4] [5]. Most of these methods are based on straight movement along roads and turns at intersections, which vary according to how the orientation is determined and how the intersections are detected. In [6] and [7], the direction of the road was detected based on the direction of a point cloud. In another study, a method that extracts the drivable region via a beam model was proposed [8].

In the current paper, we propose a method based on the extraction of a road skeleton undertaken while exploring in an unknown environment. Even in challenging environments such as outside areas, this method makes it possible to extracts passages using only a 2D laser sensor attached to the mobile robot developed in our laboratory, which is shown in Fig. 1. This is achieved by combining several image processing tools including skeleton pruning [9]. Our purpose is to realize autonomous movement for various environments including unknown and outside locations without a pre-built map.

Fig. 1. Exterior of the wheelchair type mobile robot

II. Proposed Method

As mentioned above, we do not use a pre-build map, which means that we have to simultaneously build an environment map and localize the robot on that map. This approach is known as the simultaneous localization and mapping (SLAM) method. Typical SLAM methods include the use of the Kalman filter [12] and FastSLAM [13] based on a particle filter. In this research, we used a SLAM method based on particle swarm optimization (PSO) [14], with an occupancy grid map being used to build a map based on scan matching via PSO. We

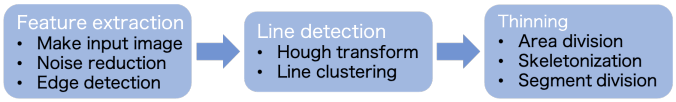


Fig. 2. Configuration of our method

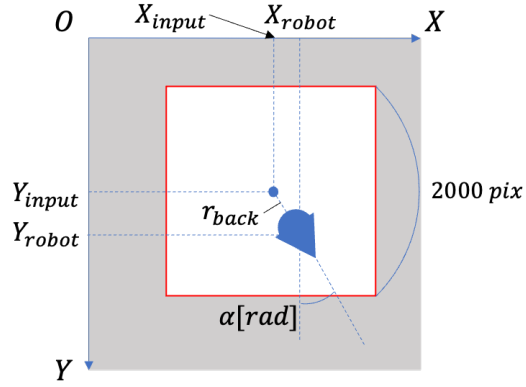


Fig. 3. The coordinate of the robot and the center of input image

also employed a number of image processing techniques to preprocess the image. The entire process consists of seven processes and is classified into three parts as outlined in Fig. 2. Each of them is described in detail below.

A. Feature extraction

1) Creating the input image: Although a 2000×2000 px map image is generated in this autonomous robot based on SLAM, we resize this image to 300×300 px to reduce the complexity of calculation. This down-sampling of the image is necessary to process it in real time. The boundary between the road and the impassable area is an important feature in terms of extracting the skeleton of the road. Thus the priority of the pixel value is determined as follows. First, cases in which the pixel value is greater than 0.7 and less than 0.05 are the highest and second highest priority, respectively. All other pixel values have the lowest priority. In addition, there is the possibility of not being able to extract sufficient number of features in the map because this would lead to the unobserved area being included in front of the robot. Therefore, the center of the input image would be a certain distance behind the robot instead of over the robot itself. Suppose that the coordinate of the robot is X_{robot}, Y_{robot}, as shown in Fig. 3. The center of the input image is then represented as follows:

$$\begin{bmatrix} X_{input} \\ Y_{input} \end{bmatrix} = \begin{bmatrix} X_{robot} - r_{back} \sin\alpha \\ Y_{robot} - r_{back} \cos\alpha \end{bmatrix} \quad (1)$$

In Eq. (1), the variable r_{back} is the distance between the robot and the center of the input image and α is the angle of the robot represented by $-\pi \leq \alpha \leq \pi$.

2) DBSCAN: Density-based spatial clustering of applications with noise (DBSCAN) [11] is employed for noise reduction during the first step of our method. This is one of a general method of data-clustering algorithm and is used to classify a point cloud based on the density of points. This clustering method makes it possible to extract the noise points. For this process, there are two main parameters to be determined: the distance within which points will be recognized as part of the same cluster and the minimum number of points in one cluster. The algorithm begins from an arbitrary point, and points within the specified distance are classified into the same cluster. The points that are far away from any cluster or the points in clusters that have less than the specified number of points are labeled as noise points. For example, the point N shown in Fig. 4 is a noise point because there is no other point within the designated radius.

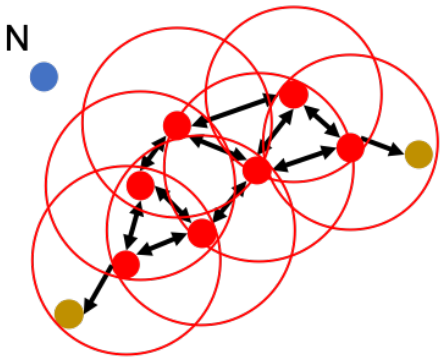


Fig. 4. Overview of DBSCAN

3) Canny edge detection: After the noise reduction, edge detection is executed through Canny edge detection [10], which is a robust and popular method for edge detection. The process can be divided into four steps. First, a smoothing process is conducted using a Gaussian filter derived via Eq. (2). Second, the gradient vector at each pixel value is derived by a convolving Sobel filter. As a third process, thinning is undertaken through comparing the adjacent vector. Finally, the edges are detected by setting a number of thresholds, and even weak edges can be detected based on the those two thresholds.

$$\begin{aligned} \theta(x,y) &= \frac{1}{2\pi\sigma^2} \mathrm{e}^{\frac{-(x^2+y^2)}{2\sigma^2}} \\ &\approx \frac{1}{16} 2^{|x|+|y|-2} \end{aligned} \quad (2)$$

In this equation, the filter size is 3 ($-1 \leq x \leq 1$, $-1 \leq y \leq 1$), $\sigma = \sqrt{\frac{2}{\pi}}$. The x and y represent the horizontal and vertical axes in the image, respectively.

B. Line detection

1) Probabilistic Hough transform: In this section, we describe the method used for line detection: probabilistic Hough transform. As shown in Fig. 5, arbitrary straight line L can be represented as $x\cos\theta + y\sin\theta = \rho$. In this equation, ρ and θ are the distance from the origin point and normal angle of L, respectively. For general Hough transform, the pairs of ρ and θ that pass through the point are found at every point of the image, and a sufficient number of points in the same pair

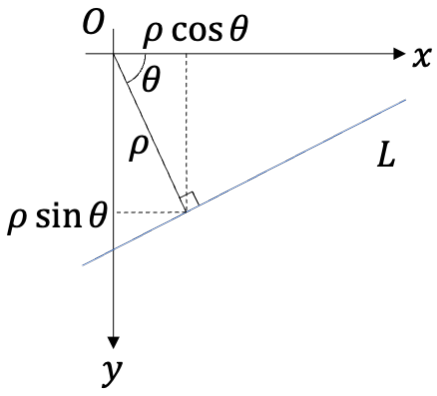


Fig. 5. Expression of line in hough transform

can be detected as being part of a straight line. In this case, every single point must be searched. In contrast, probabilistic Hough transform is sequential process and works according to following steps:

1) Choose the unscanned point randomly and initialize the accumulator.
2) Search for combinations that have the maximum number of votes on the accumulator and exceed the threshold. Otherwise back to 1.
3) Calculate the coordinates of both ends of a line segment.
4) If the length of the line segment is greater than the threshold, it is added to the detection target. Otherwise back to 1.
5) Decrement the accumulator value corresponding to the points contained in the line segment.
6) If there is unscanned point yet, back to 1. Otherwise end.

2) Line clustering: Thus, the point cloud in the image is transformed into line segments. Every line that can be derived by probabilistic Hough transform is sorted according to distance, and the line segments that are sufficiently close to one another are classified into the same cluster. In this paper, any two line segments within a distance of 20 px from each other, corresponding to 2 m, are merged into the same cluster. This process is applied to every line segment detected as described in the previous section, and single line can be classified into multiple clusters. Owing to various influences, such as unstable scanning, a number of inappropriate clusters might exist. Therefore, we set certain conditions and executed processing to disable clusters with following features:

- less than three line segments; or
- a total line segment distance (of all segments included in the cluster) of less than 2 m.

C. Thinning

1) Area division: Before extracting the map skeleton, the map should be divided into multiple areas, with each area corresponding to a region dominated by a cluster, to obtain the borders among these areas. The dominant area of each cluster is composed of the pixels closest to that cluster among any others.

2) Skeletonization: The pixel which has more than two kind of areas in the vicinity of 8 pixels become a candidate points of skeleton, and the skeleton is expanded searching these points by BFS (Breadth-First Search). We can have a skeleton of the road with 2 pixels width at this moment. The skeleton is supposed to be the path for autonomous movement, so it has to be a 1 pixel width using thinning. In this method, we apply the Hilditch thinning algorithm [15]. The skeleton expressed by the thickness of 1 pixel is connected by four neighborhoods, not eight neighborhoods, because this expression makes branch detection easy. By applying these processing, we can finally get a binary image of a 1 pixel width passage.

Algorithm 1 Thinning algorithm

Input: $input\_map$[300][300]: An image of the map created by line clustering
m: Amount of line segments
$p[m][2]$: Endpoints of line segments
Output: $output\_map$[300][300]: An image of thinned path
for all (i, j) such that $0 \leq i \leq 300, 0 \leq j \leq 300$ **do**
Find the nearest valid line segment cluster $r(i, j)$
end for
queue.push((150,150))
repeat
$p_{search} \Leftarrow$ queue.front
if p_{search} has not been searched yet **then**
if $n(R(p_{search})) \geq 2$ **then**
$p_{start} \Leftarrow p_{search}$
else
queue.push($N_8(p_{search})$)
end if
end if
until p_{start} is found
queue.push(p_{start})
while not queue.empty do
$p_{search} \Leftarrow$ **queue.front**
if p_{search} **has not been searched yet then**
if $n(R(p_{search})) \geq 2$ **then**
$output\_map[p_{search}.y][p_{search}.x] \leftarrow 1$
queue.push($N_8(p_{search})$)
else
$output\_map[p_{search}.y][p_{search}.x] \leftarrow 0$
end if
end if
end while

3) Segment division: If a skeleton is extracted from a noisy and unstable environmental map, passages that do not actually exist are often detected accidentally. Therefore, for the final step of process , the skeleton in the image is reviewed. To prevent the robot from traveling on such a path, a process

to delete the wrong passage is implemented. Before partially removing the passage, it is necessary to divide the skeleton based on the intersection. The skeleton is divided into the same segment until the branch point follows the passage. The pixel which has more than three valid pixels in four neighborhoods (right, left, up and down) is judged as a branch.

After dividing each passage into a number of segments, each segmented passage is judged as either appropriate or inappropriate based on the original map image. The map image created by SLAM expresses the passable area and the impassable area, which might include an observed wall. Accordingly, a judgment is made based on which area each passage passes through on the map. In this paper, the decision is made via the threshold processing, which deletes the passages that pass through more than a specified ratio of impassable areas. In other words, passages that pass through impassable areas at a rate greater than the threshold are removed from the branch.

4) Setting way point: Finally, the robot realizes autonomous movement by setting way point on the skeleton derived above. Excluding the area behind the robot, each way point is searched on a circumference with a radius of 2 m and set on a skeleton that exists within that range. To prevent situations in which multiple way points are set extremely close to each other, the number of way points in each passage is limited to one. Regarding path planning, a global path planning, following definition described in [16], is represented by specifying intersections which robot should turn. On the other hand, a local path planning using a method based on A* algorithm [17] is employed. This makes it possible to avoid obstacles and minimize travel distance during next way point.

III. Simulation

To verify the accuracy of the outlined skeletonization process during running, we implemented it in a simulation environment. By reading the laser range finder scan data acquired during past movement, it was possible to reproduce the constructing a map. The image obtained by cutting out the circumference of the robot from the constructed map and resizing it was used as the input for skeletonization. The robot began to run from the bottom in Fig. 6 constructing the map and ended in the upper left corner of the image. The way points derived by skeletonization were applied to path planning. Accordingly, we can see the robot moved autonomously in this simulation. As can be seen by path the robot traveled, indicated red line in Fig. 6, the robot moved successfully throughout the entire simulation. Each way point was set 2 m in front of the robot each time the robot came within 40 cm of previously determined way point.

Regarding the image processing in each moment, we confirmed it worked successfully. Multiple paths existed, as shown in Fig. 7, which exhibits each every skeleton corresponding to each extracted path. At the end of the process, the way points indicated by the green circles were set in front of the robot, expressed as a red circle in Fig. 7i. However, in the figure, an intersection that existed in the real environment could not be recognized because there was an obstacle that was not included. To conduct a simulation that included turning, we would have to fix this by specifying the intersection beforehand.

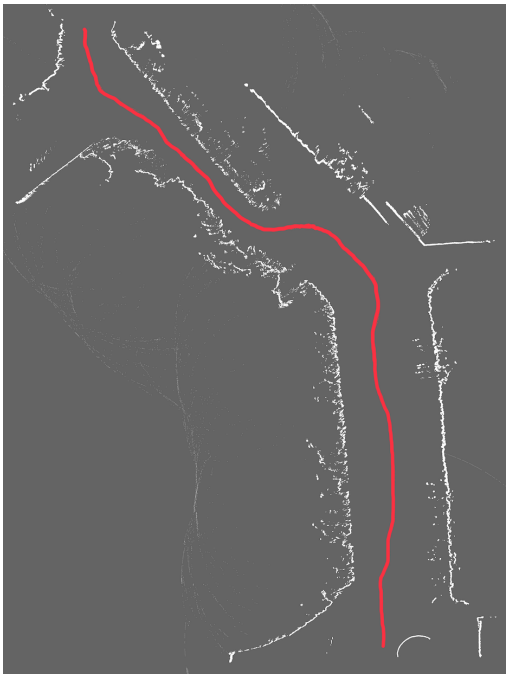


Fig. 6. The map constructed after the simulation and the trace of the robot

IV. Conclusion

In this paper, we implemented skeletonization to determine the setting way points for path planning in an autonomous robot. Even though the map was constructed real time, the autonomous movement that we expected was confirmed. Fully autonomous movement including turning at intersections specified beforehand can be realized by path planning and path selection. However, a number of problems remained, especially during the early part of the journey. We can improve the situation by considering the behavior of the robot when the map is not completed, such as through exploration to construct a more detailed map. Finally, we are planning to apply this method to autonomous movement in a real environment.

References

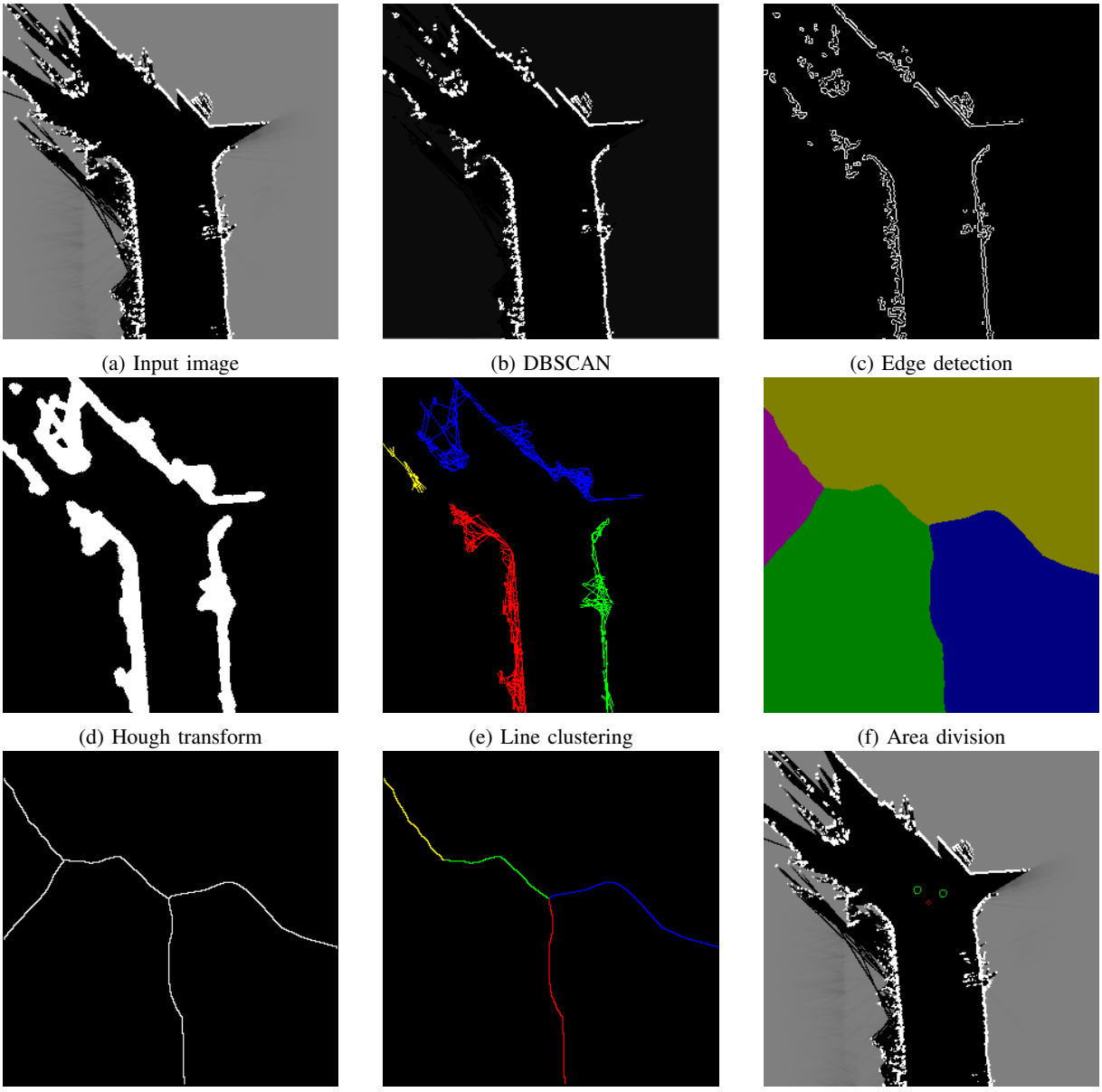
[1] J. Saarinen, H. Andreasson, T. Stoyanov and A. J. Lilienthal, "Normal distributions transform Monte-Carlo localization (NDT-MCL)," *IEEE/RSJ International Conference on Intelligent Robots and Systems,* pp. 382–389, 2013.

[2] Y. Zhang, J. Wang, X. Wang, C. Li and L. Wang, "A real-time curb detection and tracking method for UGVs by using a 3D-LIDAR sensor," *IEEE Conference on Control Applications,* pp. 1020–1025, 2015.

[3] J. Wang and Y. Fujimoto, "High Accuracy Real-Time 6D SLAM with Feature Extraction Using a Neural Network," *IEEJ Journal of Industry Applications,* vol. 10, no. 5, 2021.

[4] R. Toshimitsu and Y. Fujimoto, "Transformation Between Simple and Detailed Maps Based on Line Matching for Robot Navigation," *proc. IEEE International Conference on Industrial Informatics,* 2019.

[5] A. Watanabe, S. Bando, K. Shinada, *et al.*, "Road following based navigation in park and pedestrian street by detecting orientation and finding intersection," *International Conference on Mechatronics and Automation,* pp. 1763–1767, 2011.



(a) Input image (b) DBSCAN (c) Edge detection

(d) Hough transform (e) Line clustering (f) Area division

(g) Skeletonization (h) Segment division (i) Way point

Fig. 7. A result of Skeletonization

[6] R. Higuchi and Y. Fujimoto, "Road and Intersection Detection Using Convolutional Neural Network," *IEEE International Workshop on Advanced Motion Control*, 2020.

[7] R. Toshimitsu and Y. Fujimoto, "Detection of Road Direction Using 2D LRF in Unknown Environment," *IEEJ Workshop on Sensing, Actuation, Motion Control, and Optimization*, 2018.

[8] S. Thrun, W. Burgard and D. Fox, "Probabilistic robotics," MIT Press, 2005.

[9] W. Shen, X. Bai, R. Hu, H. Wang, and L. J. Latecki, "Skeleton growing and pruning with bending potential ratio," *Pattern Recognition*, vol. 44, no. 2, pp. 196–209, 2011

[10] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* pp. 679–698, 1986.

[11] M. Ester, H. P. Kriegel , J. Sander, X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, pp. 226–231, 1996.

[12] M. W. M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte and M. Csorba, "A solution to the simultaneous localization and map building (SLAM) problem," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 229–241, June 2001.

[13] M. Michael *et al.*, "FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," *International Joint Conference on Artificial Intelligence*, pp. 1151–1156, 2003.

[14] T. Matsuno and Y. Fujimoto, "Experimental Verification of Localization by Optimization Considering Occupancy," *IEEJ Workshop on Sensing, Actuation, Motion Control, and Optimization*, 2018.

[15] C. J. Hildich, "Linear Skeleton from Square Cupboards," *Machine Intelligence*, vol. 4, pp. 403–420, 1969.

[16] W. Khaksar *et al.*, "A review on mobile robots motion path planning in unknown environments," *IEEE International Symposium on Robotics and Intelligent Sensors,* pp. 295–300, 2015.

[17] Y. Hasegawa and Y. Fujimoto, "Experimental Verification of Path Planning with SLAM," *IEEJ Journal of Industry Applications*, vol. 5, no. 3, pp. 253–260, 2016.