SUGISAKA MASANORI

Journal of Advances in Artificial Life Robotics
Vol. 2(1); June (2021), pp. 6–11
ON LINE ISSN 2435-8061; ISSN-L 2435-8061
https://alife-robotics.org/jallr.html

Research Article

Graph-Based Path Generation for Robot Navigation in a Forest Environment

Ayumu Tominaga[1], Eiji Hayashi[1], Ryusuke Fujisawa[1], Abbe Mowshowitz[2]
[1]*Graduate School of Computer Science and Systems Engineering, Kyushu Institute of Technology, 680-4 Kawazu, Iizuka, Fukuoka, Japan*
[2]*Department of Computer Science, The City College of New York, 160 Convent Avenue, New York, NY 10031, USA*

ARTICLE INFO

Article History

Received 25 November 2019
Accepted 02 March 2021

Keywords

Path generation
Area coverage
Field robot
Navigation
Graph
Forestry

ABSTRACT

This study evaluates a trajectory generation method for the efficient navigation of autonomous mobile robots in forests. We propose a graph-based cycle generation method. A graph was generated using environmental landmarks as nodes, and the graph was modified to be Eulerian. The Hamiltonian cycle contained nodes that could be regarded as the midpoint between a pair of landmarks; an efficient path could then be found. We applied this method to an artificial forest to verify the feasibility.

1. Introduction

Path planning is an essential function for autonomous mobile robots and is required to automate work in many sectors, such as production, agriculture, fishing, and forestry. Path planning aims to provide an appropriate path and final destination for the robot, which will vary with the work being done and the working environment.

This study explores path planning for a field robot, specifically a forest industry robot. Forestry robots work in artificial and mountain forests. The robots are given tasks such as weeding and tree observation; it is expected that these tasks will become automated to compensate for a lack of workforce. Let us consider a robot tasked with weeding. Fig. 1 illustrates weeding done by an autonomous mobile robot in an artificial forest. The robot will be equipped with a weeding mechanism, and must be able to travel across the entire workspace to remove all weeds. In addition, the robot must be able to carry out its work without damaging the trees. Therefore, the path traveled must traverse all collision-free space.

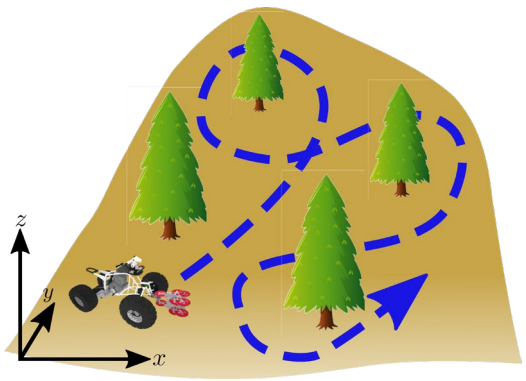

Fig. 1. Automatic weeding by an autonomous mobile robot

Corresponding author's E-mail: tominaga@mmcs.mse.kyutech.ac.jp, haya@mse.kyutech.ac.jp, fujisawa@ces.kyutech.ac.jp

Finding a path that visits all given coordinates is a common optimization problem, known as the Travelling Salesman Problem (TSP) [1]. In the TSP, the cost of moving between coordinates is given based on distance, and the minimum cost cycle will be found among the paths that visit all coordinates exactly once and return to the starting position. In some robotic applications, planning such a path is important to complete tasks in the shortest time and with the least energy consumption, or to achieve complete coverage of a work area [2]. However, the TSP is classified as an NP-hard problem for which an exact solution cannot be easily obtained; therefore, it is generally solved using an algorithm, resulting in a suboptimal solution [3],[4].

The aim of this work is to develop a method for generating such a suboptimal path in a forest environment. Previous methods have divided the workspace into small regions (cells), and created a graph structure using these cells as a set of nodes. A suboptimal path for visiting all nodes can be found from the graph by graph algorithms. Usually, the cells that divide the workspace are rectangles of equal size, and a map represented by such cells is called a grid map. The memory usage for the map representation exponentially growths with the area of the workspace, assuming that the resolution of this cell is constant [5]. Naturally, it is considered that the computational cost of path finding will be increased on the grid map represents a large-scale workspace such as a forest. Furthermore, it is desirable to be able to generate the path with smaller computational costs, since recalculation may be required due to changes in the environment caused by natural disasters. Therefore, in this study, we attempt a generation method based on the placement of environmental landmarks, instead of a region decomposition-based method, to obtain a sufficient solution. The proposed method is based on a characteristic of line graphs in graph theory, which has not been investigated thus far. The method only requires representative points of each environmental landmark (i.e., the center coordinate of the obstacle) as inputs, and the cycle is obtained by geometric computation and graph algorithms. The proposed method was verified through simulations and artificial forest data.

2. Graphs and Line Graphs

In this study, a graph, $G = (V, E)$, is constructed from a set of nodes, V, and edges, E. An individual node ($v_i \in V$) represents a position on a two-dimensional workspace, and an edge ($e_j \in E$) represents a line that joins two nodes. Here, $e_j = \{v_s, v_t\}$ represents e_j which joins v_s and v_t. Our proposed path generation method uses line graphs; a transformation in graph theory, defined as follows. The line graph of a graph, G, is denoted by $L(G)$ in this paper:

- $L(G)$ has a set of nodes that correspond to the edges of G.
- Two nodes, $v_s, v_t \in V(L(G))$, are adjacent if the corresponding edges in G are adjacent.

As shown in the example of a line graph (Fig. 2), $L(G)$ is made by replacing the edges of G with nodes, and

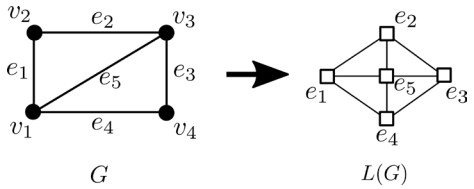

Fig. 2. Example of a line graph; G is an Eulerian graph including Euler path, $L(G)$ is the line graph of G and Hamiltonian graph with Hamilton path

represents the edge joining of G [1, 5]. It is known that a line graph has the following characteristic: if G is an Eulerian graph, $L(G)$ will be Hamiltonian. Eulerian graphs always have a cycle that traverses every edge exactly once. In the case of G in Fig. 2Fig. 2, by starting at v_1 and visiting nodes with passing through each edge in the order $\{v_1, e_1, v_2, e_2, v_3, e_3, v_4, e_4, v_1, e_5, v_3\}$, every edge can be crossed $(e_1, e_2, e_3, e_4, e_5)$. Hamiltonian graphs contain a path that visits every node exactly once. In the case of $L(G)$ in Fig. 2, by starting at e_1 and traveling $\{e_1, e_2, e_3, e_4, e_5\}$, all nodes can be visited. Using the TSP on the Hamiltonian graph will result in the shortest path being used to visit every position. Using this proposed method, the Hamiltonian cycle can be found rapidly by an autonomous moving robot. The optimal solution may not always be obtained,

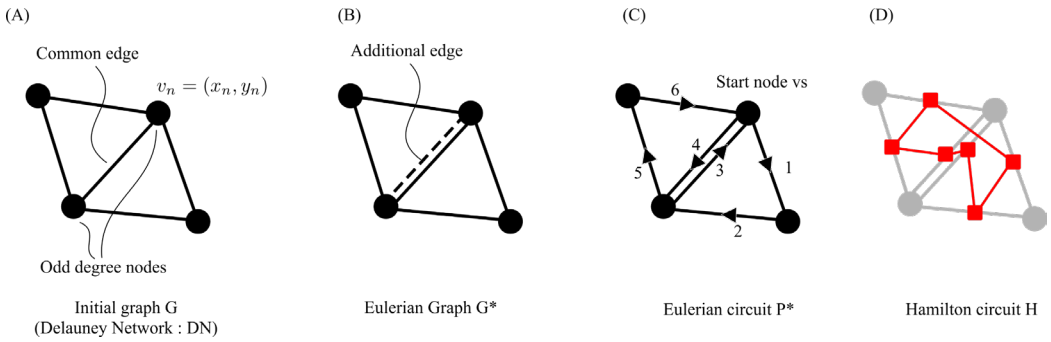

Fig. 3 Procedure of the graph-based path generation

but we will always find an appropriate path in the given environment through this method.

3. Procedure of Path Generation

In this proposed method, the geography of the workspace was obtained in advance (i.e., size of the work area and the location of all landmarks). By referring to the location of the landmarks, the initial graph G was generated (Fig. 3A). G is made into an Eulerian graph by the addition of edges (Fig. 3B). Finally, the Eulerian path was found (Fig. 3C), and the Hamiltonian path computed from this (Fig. 3D). The Hamiltonian cycle represents a trajectory that can traverse the entire work area. Details of each step are described below.

3.1. *Generation of initial graph*

Let N be the total number of landmarks in the environment. G has a set of nodes, $V = \{v_n \,|n = 1,2,\dots,N\}$, where each node, v_n, is given two-dimensional coordinates (x_n, y_n). Each landmark would serve as an obstacle for the robot to avoid. Based on the neighborhood of the nodes, a set of edges is constructed appropriately. Every edge represents a segment that the robot should cross. Therefore, all edges should be connected between pairs of nodes that satisfy the line-of-sight condition. The Delaunay triangulation method was used to do this as it is simple. This made a Delaunay Network, and every Delaunay edge was added to G, as shown in Fig. 3 (A).

3.2. *Creating an Eulerian graph*

To ensure that $L(G)$ contained the Hamiltonian path (i.e., G was an Eulerian graph), every node of G had to be connected to an even number of edges (even degree nodes). Therefore, edges were appended or removed from G. In graph theory, editing a given graph into an Eulerian graph such that it has Eulerian paths is called the Chinese Postman Problem [1]. For general graphs, Eulerian graphs can be created by increasing some edges in the graph to two, based on the following method:

(i) Find all nodes with odd degree in a given graph. Note that the number of nodes is always even due to the handshaking lemma [6].

(ii) Pair the above nodes in a way that minimizes distance traveled.

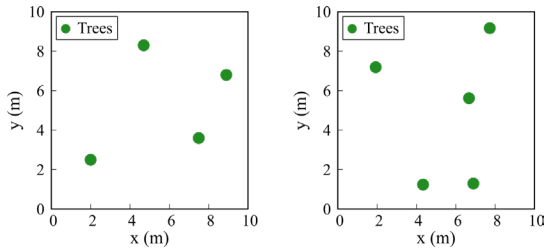

Fig. 4. Simulated tree locations

(iii) For each of these pairs, find the shortest path between nodes of its edges, and append the edges contained in it to the graph.

Here, the Eulerian graph, G^*, obtained by the method described above is a multigraph (graph which allows multiple edges to join the same node pair).

3.3 *Computation of Hamiltonian cycle*

The Eulerian cycle in G^* can be found by Fleury's algorithm [6], and is denoted by $E^* = \{e_1, e_2, \dots, e_m\}$. Finally, new nodes, $v_m^l \in V(L(G))$, defined by $v_m^l = h(e_m)$ are obtained, and the order of visiting of $v_m^l \in V(L(G))$ should obey E^*. Thus, we can obtain the Hamiltonian path of the environment. We imposed the

natural geometry condition on $h(e_m)$ (Eq. (1)). Here, (x_s, y_s) and (x_t, y_t) are the positions given to the nodes $v_s, v_t \in V(G)$. In other words, the nodes of $L(G)$ represent the midpoints of every edge in G^*.

$$h(e_m) = \left\{\frac{x_s+x_t}{2}, \frac{y_s+y_t}{2}\right\} \quad (1)$$

4. Experimental Results

This section describes the results of the proposed cycle generation method. The path was generated by following the procedure described in section 3, using Python and the Network X package.

4.1. *Simulated path generation*

This subsection describes a simulation of the proposed path generation. We verified the proposed method in the following conditions:

- The workspace was a two-dimensional Euclidean space of ten x ten m squares (0.01 hectare).
- Simulations were carried out for four and five trees.
- The trees were represented by circles with diameters of 0.3 m.

These conditions were chosen because the density of harvestable trees (> 50 y) is approximately 500 per hectare, and they have an average stem diameter of 0.3 m.

Fig. 4 shows the locations of the landmarks in the simulation, and Fig. 5 shows the paths created. In the case of $N = 4$, the five initial edges of G were created by Delaunay Triangulation. Edges were added between trees that were in the line-of-sight of each other (Fig, 5A, $N = 4$). Here, v_2 and v_4 were odd degree nodes. To make G into an Eulerian graph, the edge $e_5' = e_5 = \{v_2, v_4\}$ was selected as the edge which should be appended (Fig. 5B, $N = 4$). The Eulerian path $\{e_4, e_3, e_2, e_5', e_1\}$ was found and the start node, v_s, was chosen to be v_1 (Fig. 5C). Finally, the Hamiltonian cycle, H, could be computed using Eq. (1) (Fig. 5D). Every node of H was generated to be the midpoint of a pair of trees and were connected by edges. Here, e_5 and e_5' in Eulerian graph represent same segment, so the nodes suffering these two edges were set on the same midpoint between v_2 and v_4. The proposed method also provided a path which allowed the robot to travel all

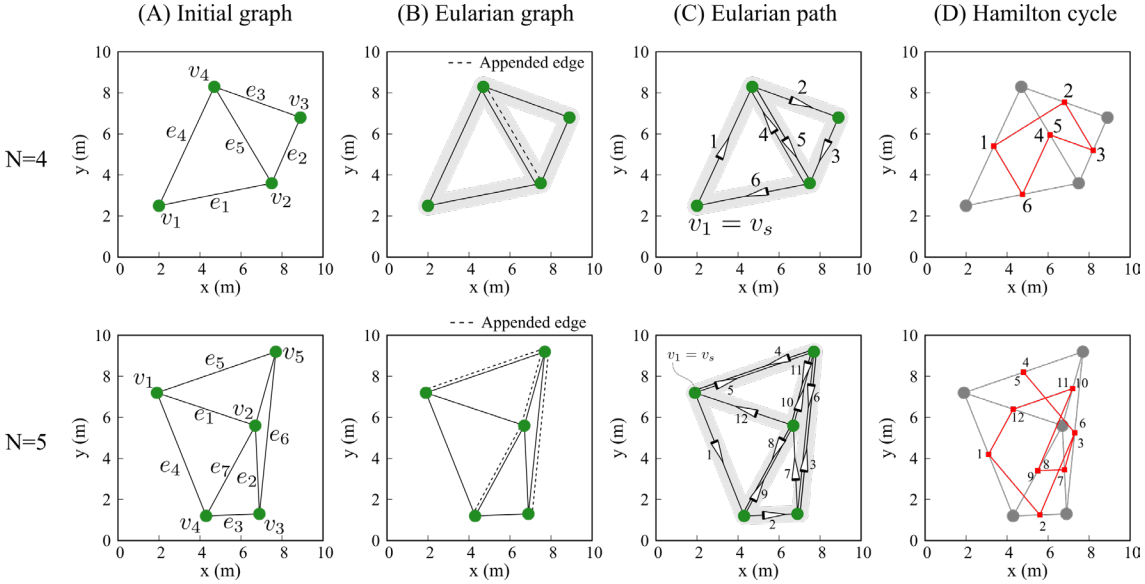

Fig. 5. Simulated path generation results

midpoints of tree pairs for $N = 5$ (Fig. 5). Here, four additional edges were arisen, and there are four

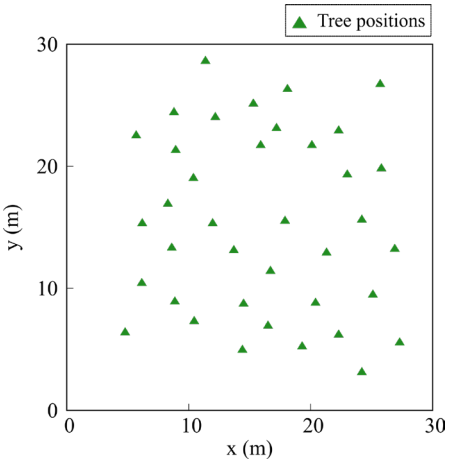

Fig. 6. Tree positions in the artificial forest

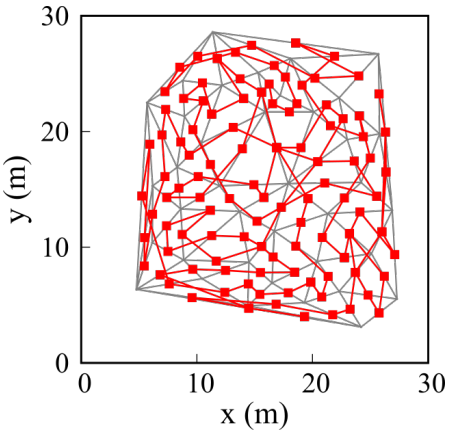

Fig. 7. Generated path in the artificial forest

overlapping midpoints in the Hamiltonian cycle as the result.

4.2. *Artificial forest path generation*

The proposed method was applied to an artificial forest. The experimental forest on the campus of the Kyushu Institute of Technology consists of 37 trees spread over a 30 × 30 m square measuring 0.12 hectares. There are no objects, other than the trees, in this area that might impede the movement of a robot. The position of the trees was measured using a Terrestrial Laser Scanner (Leica, 3D Disto), as shown in Fig. 6. The positive y- and x -axes are oriented to true north and east, respectively. Fig. 7 shows the results of path generation in the forest. There were 38 nodes (trees), resulting in 140 waypoints. The total length of the path was 398.4 m. Using a computer with an intel core i7, 2.6 GHz CPU, the time from creating the initial graph to obtaining the path was 0.064 s. Moreover, the proposed method succeeded in generating a Hamiltonian path from an Eulerian graph based on the characteristics of the line graph. In addition, using the neighborhood graph that satisfies the condition that any pair of trees is prospectively line-of-sight, the path passed through all midpoints of tree pairs corresponding to waypoints that the robot will visit. In summary, the proposed method can generate a path for robot navigation using the coordinate information of trees in a forest as an input.

5. Conclusion

This paper proposed a graph-based path generation method for autonomous forestry robots. A final path in the given workspace is computed systematically from an initial geographic map containing the positions of trees. In addition, the proposed method provides a Hamiltonian cycle that can traverse all midpoints of tree pairs by simple geometrical computation and graph algorithms [7]. However, to improve the safety of the robot, the collision of the segments in the Hamiltonian cycle and the obstacles in the environment should be considered both offline and online. Moreover, to be used in a wider range of applications, testing in urban environments, underwater environments, etc. are necessary in the future.

References

[1] B. Steinberg, *Covering walks in graphs*, Springer-Verlag, New York, NY, 2014.

[2] E. Galceran and M. Carreras, A survey on coverage path planning for robotics, *Rob. Auton. Syst.*, **61**(12) , 2013, pp. 1258–1276.

[3] H. Van Pham, P. Moore, and D. X. Truong, Proposed smooth-STC algorithm for enhanced coverage path planning performance in mobile robot applications, *Robotics*, **8**(2), 2019.

[4] Y. Gabriely and E. Rimon, Spanning-tree based coverage of continuous areas by a mobile robot, *Ann. Math. Artif. Intell.*, **31**(1–4) , 2001, pp. 77–98.

[5] S. Thrun, Learning metric-topological maps for indoor mobile robot navigation, *Artif. Intell.*, **99** (1) 1998, pp. 21–71.

[6] A. M. Hobbs, Powers of graphs, line graphs, and total graphs in *Theory and Applications of Graphs*.271-285, Springer, Berlin, Hiedelberg, 1978.

[7] R. Diestel, *Graph Theory*, 173, Springer Berlin Heidelberg, 2017.

Author Introductions

Mr. Ayumu Tominaga

Mr. Tominaga received his M.S. degree from the Department of Interdisciplinary Informatics at Kyushu Institute of Technology, Japan, in 2017. He is pursuing a PhD at Kyushu Institute of Technology, Department of Computer Science and Systems Engineering under the supervision of Prof. Eiji Hayashi. His research interests include intelligent robots and field robots.

Prof. Eiji Hayashi

Professor Hayashi works in the Department of Intelligent and Control Systems at Kyushu Institute of Technology. He received his PhD (Dr. Eng.) degree from Waseda University in 1996. His research interests include intelligent mechanics, mechanical systems and perceptual information processing. He is a member of The Institute of Electrical and Electronics Engineers (IEEE) and The Japan Society of Mechanical Engineers (JSME).

Dr. Ryusuke Fujisawa

Dr Fujisawa is an Associate Professor in the Department of Intelligent and Control Systems at Kyushu Institute of Technology. He received his PhD (Dr. Eng.) degree from the University of Electro-Communications, Tokyo, in 2009. His research interests include swarm intelligence, swarm robotics, and ethology. He is a member of The Information Processing Society of Japan (IPSJ).

Prof. Abbe Mowshowitz

He received the PhD degree from University of Michigan in 1967. He has been professor of computer science at the City College of New York and member of the doctoral faculty at the Graduate Center of the City University of New York since 1984. His current research interests lie in two areas are organizational and managerial issues in computing, and network science. In addition to teaching and research, He has acted as consultant on the uses and impacts of information technology (especially computer networks) to a wide range of public and private organizations in North America and Europe.