Available *online* at : http://bit.ly/InfoTekJar

InfoTekJar : Jurnal Nasional Informatika dan Teknologi Jaringan

ISSN (Print) 2540-7597 | ISSN (Online) 2540-7600

Path Planning, Shortest Path

Reverse Tracking Graph Based on Dynamics Path Planning

Devanta Abraham Tarigan [1], *Muhammad Zarlis* [1*], *Rahmat Widia Sembiring* [2]

[1] *Faculty of Computer Science Universitas Sumatera Utara, Medan, 20155, Indonesia*
[2] *Department of Computer Engineering Politeknik Negeri Medan, Medan 20155, Indonesia*

KEYWORDS

Reverse Tracking, Dijkstra Algorithm, Depth First Search Algorithm, pre path planning, availability path planning

CORRESPONDENCE

Phone: +62 823 6166 2729

E-mail: m.zarlis@usu.ac.id

A B S T R A C T

This paper gives substance to Dynamic Path Planning focusing on reverse tracking method. The development of this method are proposed and expected to reduce the algorithm scanning the whole graph repeatedly. In this paper, several approachment will be presented sequentially. First, analysis and modeling of the obstacle and environment, pre-path planning, Depth First Search for availability path planning, and improvement of the Dijkstra algorithm for the shortest path. There-in the proposed model is defined by adopting the reverse feature in the Depth First Search algorithm in the finding of the availability path on the graph.

INTRODUCTION

Dynamic path planning has many puzzles to be solved. Reverse tracking indispensable in Graph technology. In the graph, the start node is given by the root node and the target should be the destination node. Reverse tracking provides for finding the global path planning specifically on dynamic path planning. Why there is Dynamic Path Planning? In recent years, the path planning concept being applied in many areas specifically in intelligent features. In robotics, avoiding the obstacle is one of the most important things of the implementation of this technology. The path-planning allows the robot to effectively avoid the obstacle and guide to find the destination or target [1]. Path-planning observation result is to be able to find a collision-free path on a graph from the starting node to the target based on a specific certain evaluation in the environment with obstacles [2]. Many routes have been tracked to achieve the target, the problem in recent years the obstacle has a way of being able to dynamically change position.

The main focus of the reverse track is to return the route to reach the target which has been detected. The proposed method of development reverse tracking firstly to analyze and modeling obstacle environment. Modeling of environmental obstacles as a basic requirement to research the path planning, the advantage, and disadvantages of obstacle modeling automatically affect the effect of path planning [4]. Pre-path planning offers a path based on the graph from the prior-known obstacle environment [5]. In this paper, there are several sub-research concepts offered to help understand the method flow of the research. Depth First Search algorithm as the finder availability path planning in the overall node in graph focus consider the path should be taken of the shortest path algorithm. Depth First Search (DFS) tracing all the branches in a graph, and doing reverse feature if endpoint not found in the graph. After reverse, DFS repeats the process until finding the endpoint [8]. DFS is a method of undirected graph scanning that has been widely known as a powerful method for solving graph problems[9].

In the research for the shortest path, Dijkstra’s algorithm is the most famous shortest pathfinder method and proposed in 1959 by Edsger Dijkstra [11]. Dijkstra is a kind of greedy algorithm and the most used algorithm for finding the shortest path [3]. Dijkstra as the main focus algorithm to be improved shortest pathfinder to add by adding the feature adopted by the Depth First Search algorithm because in conventional Dijkstra Algorithm, as the famous method for finding the shortest path there is still a problem for finding the path if the graph changed according to the target reverse tracking based on dynamic path planning.

In this article, we presented the pre-path planning to imaged possibility path after analysis and modeling obstacle environment clear. This will help full to reduce repetitive work of improved algorithm. In the proposed Depth First Search algorithm, the reverse feature in this algorithm will be presented as instructed the Dijkstra algorithm method will help Dijkstra not to tracking the shortest path. In Actually work, the reverse method in Depth First Search to applied to check whether edges are connected to find the possible path to reach the endpoint or target, one starts at the root node (based on graph DFS selecting arbitrary node as the root)[10], but there is still a problem if Depth First Search Algorithm overall applied in this case specifically with the cost of time complexity, because each will be tested.

In this work, we are not just describing the reverse feature to be adopted in Dijkstra, but sequencely describe the method for dynamic path planning. It is assumed as follows :

1) *Analysis and Modeling Obstacle Environment:* In the modeling obstacle environment we implement a grid to localization the obstacle to be analyzed. The starting point starts from A and the endpoint is Z (in this case, the trajectory collides with the obstacles).

2) *Route Searching:* In this sub-research, DFS applied in Availability or APP (in this case, this will show the original trajectory to reach the endpoint). Pre-path planning will propose the new graph according to the obstacle in the graph. Then the improved Dijkstra Algorithm will execute the graph. If the condition change, it will turn on the reverse feature on this method.

ANALYSIS AND MODELING OBSTACLE ENVIRONTMENT

Modeling the obstacle environment has many advantages to designing pre-path planning in every subject of the smart control. We assume, this kind of modeling reduce at least of time operation of the algorithm. The main purpose is to design obstacle collision avoidance control against the trajectory [12]. The advantages of this method will affect the applied algorithm to analyze the environment. The reverse tracking very dependent with the modeling. In this research, we proposed the obstacle defined in 6*5 grid. The localization of the obstacle randomly fulfill the grid. In this sub-research, each obstacle is transformed into a circular shape inside of the grid as shown below.

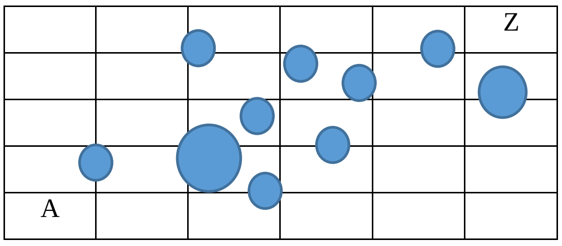


Figure 1. Analysis and Modeling obstacle environment

By analyzing the obstacle, we can see it can be divided into 3 bases, firstly the starting point, endpoint, and obstacle. If the starting point indicates as "A" and endpoint as "Z" and the line indicate as first shortest path. In the search of trajectory from A → Z, this method allows pathfinder learns the appropriate pathways to patrol in unfamiliar space. It will also built the map simultaneously [6]. In dynamic path planning the obstacle indicated change the location automatically. In this case, figure out the sensor has been applied and imaged the obstacle environment.

PRE PATH PLANNING

Pre-path planning is a multi disciplines research topic. Artificial Field Potential (AFP) is the frequently method for planning the trajectory in the environment. The construction of the pre-path planning will obtain the potential path planning. In this article, we study to design pre-path planning of this research. Pre-path planning focus on the decision that must be taken in dynamic path planning. We assume this method very is indispensable for imaging the overall path.

Hong Shen in *"Unmanned Aerial Vehicle (UAV) Path Planning based on Improved Pre-planning Artificial Potential Field Method"* we proposed of Pre-path planning as follows. Firstly, define the starting point or root node as x_{init} in obstacle environment. Then scatter randomly a point in the environment to get point x_{rand} , in this case its aims to x_{rand} collides with obstacle. If x_{rand} not in there, then connect x_{init} and x_{rand} to get line segment *L*. If does not collide any obstacle, let x_{init} move to certain distance in the direction of x_{rand}, In this case the next x_{new} became the new point of x_{rand} Then a simples tree with two node will be made to avoid the obstacle [7]. In this paper, we proposed the flowchart of the pre-path planning analysis in the figure 2. The flowchart aim is to describe the flow process easily.

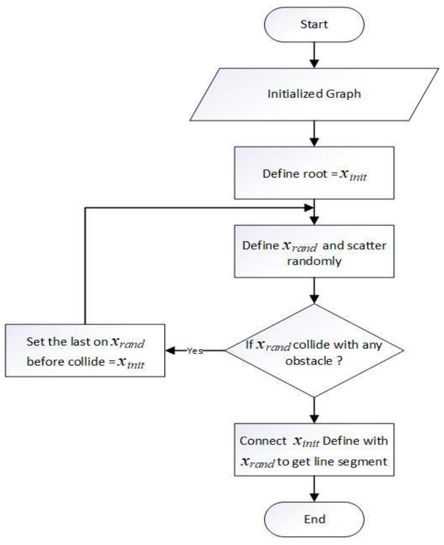

Start
Initialized Graph
Define root = x_{init}
Define x_{rand} and scatter randomly
If x_{rand} collide with any obstacle ?
Yes
Set the last on x_{rand} before collide = x_{init}
Connect x_{init} Define with x_{rand} to get line segment
End

Figure 2. Pre-path planning flowchart

By analyzing the flowchart of this method. Pre-path planning contruct the potential path planning to avoid the obstacles. We can figure out the the pre-path planning as shown as below

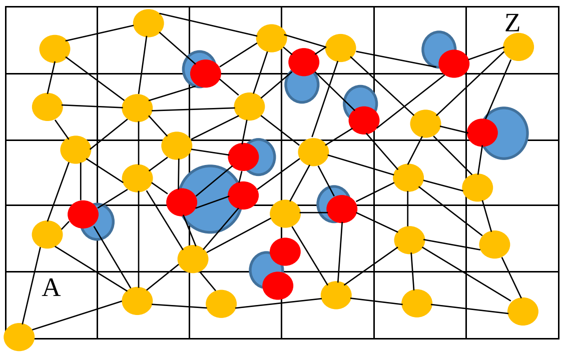


Figure 3. Pre-path Planning Analyzed

According to the figure above, the yellow circle is the passable node, in this case, the red circle is the collision node, and the blue circle is the obstacle. We can see the x_{init} from the root node create randomly node to get point x_{rand}

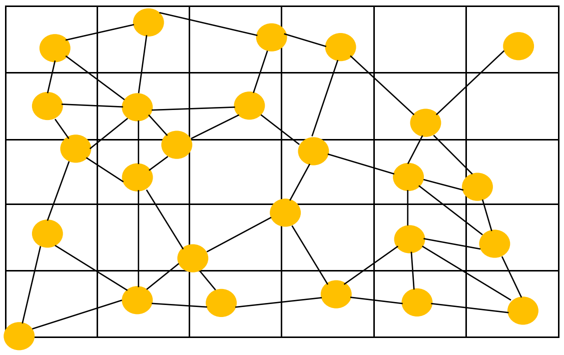


Figure 4. Free collision node

In figure 4, the free collision node is produced, then the next process is to track all the paths to reach the endpoint. In this paper, we implemented the Depth First Search (DFS) for finding the availability path in the graph to reach the endpoint.

PROPOSED METHOD

In this paper, we proposed Dijkstra to be improved by adding the reverse feature to tracking the last one of the passable node. We assumed this method will reduce the Depth First Search or DFS tracking the node repeatedly from the start point. In edge relaxation of conventional Dijkstra algorithm, the edge relaxation is the operation to achieve the target by calculating each node to reaching the cost to the vertex lower. Below is the figure of conventional edge relaxation.

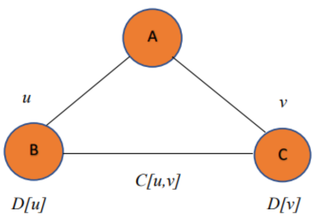


Figure 5. Node Relaxation in Dijkstra

According to the fig.1, edge relaxation will update $D[v]$ became $D[u] + C[u.v]$ if less than $[v]$. Below the edge relaxation has represented. Node “C” has an infinity value. It can be explained from the vertex u to v if the distance u adds the value between A→B = 2 or smaller than infinity then the distance of v is equal to the distance of u add by $u + v$. There can be a problem if there is a disturbance that causes the arrangement of nodes in the graph to automatically change.

In the proposed method, we using Depth First Search’s reverse feature to set the track to the last passable node. It aims to help Dijkstra to reduce the algorithm, compare if DFS track the overall graph from the start point. In the figure below we imaged the flow process according to the result of pre-path planning.

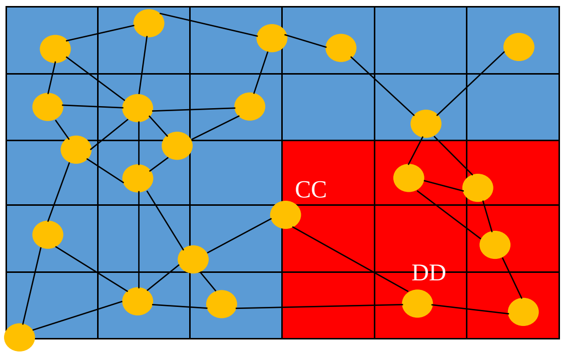


Figure 6. Changed Free Collision Node

According to the figure above, We can see there are several node has change the position. Focus on node “DD” is the current location node, and “CC” is the last one of passable node. In this case, We design the method for tracking back into the last one of the passable node as shown below.

Step1. We define the current node as $G\ (Xnode)$, In this case G is the main graph, and $Xnode$ as the current location. This aims to remark the last position in blind graph.

Step2. Secondly, DFS run the reverse feature as $G\ (Xnode \rightarrow Xlast)$. In this case, the movement traverse back into $Xnode$, this option always repeat after the $G\ (Xnode)$ has different planning base on pre-path planning,

Step3. Finally, the improved Dijkstra will find the shortest path in Dynamic Path Planning.

In this research, the advantage of this feature is can reduce the regular check from DFS to traverse all the node from the start point. We assume, this method will much reduce the time complexity of algorithm. In this paper, we designed the flowchart of the proposed method as below.

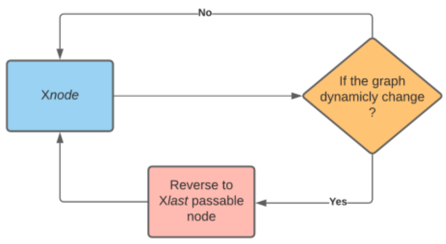
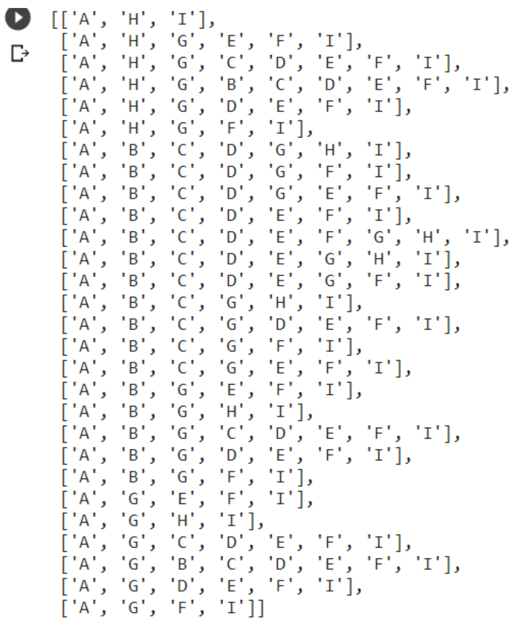

No
Xnode
If the graph dynamicly change ?
Yes
Reverse to Xlast passable node

Figure 7. Flowchart of Proposed method

In this simulation, we give weight to the edges of the graph. Firstly, Search the availability of path planning based. In this case, We proposed the “A” is the starting point and the “I” is the endpoint. Depth First Search will propose several paths planning to reach the target as shown below.



```
[['A', 'H', 'I'],
 ['A', 'H', 'G', 'E', 'F', 'I'],
 ['A', 'H', 'G', 'C', 'D', 'E', 'F', 'I'],
 ['A', 'H', 'G', 'B', 'C', 'D', 'E', 'F', 'I'],
 ['A', 'H', 'G', 'D', 'E', 'F', 'I'],
 ['A', 'H', 'G', 'F', 'I'],
 ['A', 'B', 'C', 'D', 'G', 'H', 'I'],
 ['A', 'B', 'C', 'D', 'G', 'F', 'I'],
 ['A', 'B', 'C', 'D', 'G', 'E', 'F', 'I'],
 ['A', 'B', 'C', 'D', 'E', 'F', 'I'],
 ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I'],
 ['A', 'B', 'C', 'D', 'E', 'G', 'H', 'I'],
 ['A', 'B', 'C', 'D', 'E', 'G', 'F', 'I'],
 ['A', 'B', 'C', 'G', 'H', 'I'],
 ['A', 'B', 'C', 'G', 'D', 'E', 'F', 'I'],
 ['A', 'B', 'C', 'G', 'F', 'I'],
 ['A', 'B', 'C', 'G', 'E', 'F', 'I'],
 ['A', 'B', 'G', 'E', 'F', 'I'],
 ['A', 'B', 'G', 'H', 'I'],
 ['A', 'B', 'G', 'C', 'D', 'E', 'F', 'I'],
 ['A', 'B', 'G', 'D', 'E', 'F', 'I'],
 ['A', 'B', 'G', 'F', 'I'],
 ['A', 'G', 'E', 'F', 'I'],
 ['A', 'G', 'H', 'I'],
 ['A', 'G', 'C', 'D', 'E', 'F', 'I'],
 ['A', 'G', 'B', 'C', 'D', 'E', 'F', 'I'],
 ['A', 'G', 'D', 'E', 'F', 'I'],
 ['A', 'G', 'F', 'I']]
```

Figure 9. Availability Path Planning

There are 28 paths to reach the “I” or endpoint based on figure 8. In this research, the Availability path planning became the first. We simulated, after change the connectivity edges between *“E, F, G, H, I”*,

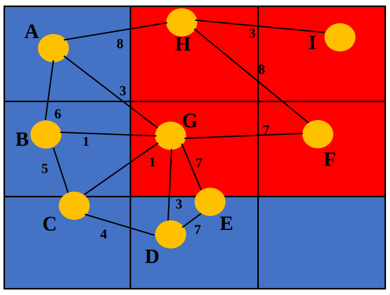


Figure 9. Changed Path Planning

The blue zone is marked as an unchanging zone and the red zone is marked as the changed zone. In this case, we simulate the current location of the root node in “F” to perform the reverse feature of is check the one last passable node which has not changed is “G”

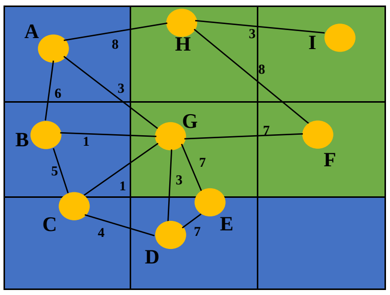
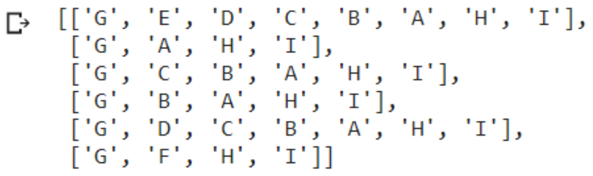


Figure 10. “G” is the root node

After simulated the new graph, we have got the result as below.



```
[['G', 'E', 'D', 'C', 'B', 'A', 'H', 'I'],
 ['G', 'A', 'H', 'I'],
 ['G', 'C', 'B', 'A', 'H', 'I'],
 ['G', 'B', 'A', 'H', 'I'],
 ['G', 'D', 'C', 'B', 'A', 'H', 'I'],
 ['G', 'F', 'H', 'I']]
```

Figure 11. Availability Path Planning of Proposed Method

There are 6 possibility path planning in new path and the trajectory start from the root “G” to reach “I” as the endpoint. The last evaluation is the shortest path planning to reach the target. The table below shows the distance of the path.

Table 2. Table of Shortest Path Distance

Posibility Path	**Distance to Endpoint**
G→E→D→C→B→A→H→I	39
G→A→H→I	14
G→C→B→A→H→I	21
G→B→A→H→I	18
G→D→C→B→A→H→I	29
G→F→H→I	18

According to the proposed shortest path algorithm, the optimal distance is *G→A→H→I*, and the weight is 14.

CONCLUSIONS

In this research. The last passable node is the main point to taking decision of the optimal path planning and the problem is when the last passable node changed this method. In future research, we improve the method to solve this problem in various ways to produced more optional nodes to be taken as the main point. There are several methods to improve in future work, especially in Dynamic Path Planning. Analysis and Obstacle modeling will be better if using the Kalman filter to localization the obstacle in the environment. This research will be better if the obstacle can be executed by training the obstacle, cluster, and classification the obstacle. In the future work, we will divide the research into several research according to the method which has been adopted in this research area.

REFERENCES

[1] L. Liu et al., "Global Dynamic Path Planning Fusion Algorithm Combining Jump-A* Algorithm and Dynamic Window Approach", IEEE Access, vol. 9, 2021, pp. 19632-19638.

[2] Z. Zhu, J. Xie and Z. Wang, "Global Dynamic Path Planning Based on Fusion of A* Algorithm and Dynamic Window Approach", Chinese Automation Congress (CAC), 2019, pp. 5572 – 5576.

[3] Risald, A. Mirino and Suyoto, "Best routes selection using Dijkstra and Floyd-Warshall algorithm", 11th International Conference on Information & Communication Technology and System (ICTS), 2017, pp. 155 -158.

[4] Qian, C., Qisong, Z. and Li, H., 2018. Improved artificial potential field method for dynamic target path planning in LBS. *Chinese Control And Decision Conference (CCDC)*, 2018, pp.2710 -2714.

[5] Z. Wang and X. Xiang, "Improved Astar Algorithm for Path Planning of Marine Robot", *2018 37th Chinese Control Conference (CCC)*, 2018, pp. 5410 – 5414.

[6] C. Wu, D. Liaw and H. Lee, "A Method for Finding the Routes of Mazes", *2018 International Automatic Control Conference (CACS)*, 2018.

[7] H. Shen and P. Li, "Unmanned Aerial Vehicle (UAV) Path Planning Based on Improved Pre-planning Artificial Potential Field Method", *2020 Chinese Control And Decision Conference (CCDC)*, 2020, pp. 2727 – 2732.

[8] A. Hidayatullah, A. Jati and C. Setianingsih, "Realization of depth first search algorithm on line maze solver robot", 2017 International Conference on Control, Electronics, Renewable Energy and Communications (ICCREC), 2017, pp. 247 – 251

[9] S. Eng, O. Penangsang, R. Wibowo, I. Suryawati and C. Chhlonh, "Distribution System Restoration Using Spanning Tree Based on Depth First Search Visual in GUI", 2019 5th International Conference on Science and Technology (ICST), 2019.

[10] E. Žunić, A. Djedović and B. Žunić, "Software solution for optimal planning of sales persons work based on Depth-First Search and Breadth-First Search algorithms," 2016 39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2016, pp. 1248-1253.

[11] G. Qing, Z. Zheng and X. Yue, "Path-planning of automated guided vehicle based on improved Dijkstra algorithm," 2017 29th Chinese Control And Decision Conference (CCDC), 2017, pp. 7138-7143.

[12] J. Ji, A. Khajepour, W. Melek and Y. Huang, "Path Planning and Tracking for Vehicle Collision Avoidance Based on Model Predictive Control With Multiconstraints", IEEE Transactions on Vehicular Technology, vol. 66, no. 2, pp. 952-964, 2017.

NOMENCLATURE

$D[v]$ meaning of the next *node* from $D[v]$, which has infinity value from the *root* node.
$D[u]$ meaning of the next *node* which has been connected to the *root node* and can be calculated
$C[u, v]$ meaning of the distance between *node* which has been connected.
$Xnode$ meaning of the current position of, which has been marked.
$Xlast$ meaning of the last passable *node* before the current position.

AUTHOR(S) BIOGRAPHY

Devanta Abraham Tarigan
Author achieve the Diploma Degree from Politeknik Negeri Medan in 2016 in Computer Engineering, and Bachelor Degree from Unversitas Pembangunan Pancabudi Medan in 2018. In the late of 2018 until now the Author is the student in Universitas Sumatera Utara Major Computer Science