Critical Comparative Study of Robot Path Planning in Grid-Based Environment

Dhaniya R D1*, Dr. Umamaheswari K M2*,

1Research Scholar, School of Computing, Department of Computer science and Engineering, SRM Institute of Science and Technology, Kattankulathur- 603203.
2Assistant Professor, School of Computing, Department of Computer science and Engineering, SRM Institute of Science and Technology, Kattankulathur- 603203.

dhaniyad@srmist.edu.in

Abstract. The path planning problem under two-dimensional maps is a fundamental issue in mobile robotics and other applications (unmanned vehicles, and so forth). The traditional algorithms (graph searching, artificial potential field and so forth) are used to find the shortest path that rely on grid-by-grid searching. The main disadvantage of the traditional methods is robot size is not considered and has much redundant in sharp curves which is difficult to guide a robot movement. In this paper, we compared the five baseline approaches of heuristic methods to find the optimal path which reduces the number of turns and time costs to attain a target. The experimental analysis shows that the second-order x derivative has an advantage over other baseline approaches that the search time and the number of turns are reduced.

Keyword: Path planning, Robot Navigation, Heuristic approach, Grid model

1. Introduction
Robotics plays an important role across a wide range of areas that perform exclusively for manufacturing works such as clinical centers, transportation, restaurants, military sites, and domestic homes [1]. Indeed, robotics growth will continue promptly in upcoming years, where autonomy and smart systems will integrate to provide more aspects of our world. The development of Robotics and the autonomous system was magnificent by following key benefits: (i) offer possibilities without human interaction; (ii) can perform complex actions; (iii) can be deployed instead of humans; (iv) can able to operate at high speeds; and (v) cost-effective [2,3]. Autonomous robots have to make intelligent decisions as well as plan optimally to attain targets depending on their tasks. Moreover, the bot needs to plan a collision-free path to reduce time, energy and distance in a given environment. In search of a collision-free path planning problem [4,5] faces in many disciplines such as chip design, tracking accomplishment, etc., but major in-disciplinary for autonomous bot’s navigation. Motion planning for autonomous bots is decomposed into path planning and trajectory planning. Path planning intention is to provoke a collision-free and optimized path concerning some criteria. Trajectory planning is to schedule the movement of bot along the planned path [6]. In this paper, we focus on the problem of path planning for autonomous
bots is an active area and covers an entire space in a static environment. Space is divided into a grid of cells and the environment addresses the bot’s navigation in a two-dimensional terrain that doesn’t intersect with any blocked cells. A vital role of the grid-based path planning method is to develop an algorithm to find the shortest path regardless of the shape of an obstacle although the different destinations in a known environment [7,8]. Our path planning algorithm figured on the subsist work that uses a heuristic search that coordinates to initiate a feasible solution. Hereby, we are comparing the five baseline approaches to find an optimal path from source to goal state in a grid environment. This paper is organized as follows. Section 2 contains the experimental results of the five baseline algorithms. The result analysis is compared based on performance metrics in section 3. Finally, we concluded this paper in section 4.

2. Experiments and Results

Experimental Setup: We have demonstrated the experiments on the developed robot as shown in figure 1. The robot is equipped with an ultrasonic sensor for identifying an obstacle by emitting sound waves. The sensor will detect an obstacle with a maximum distance up to 15 cm. The Arduino framework is used to integrate the test system and all the codes are running on a low-performance CPU having a configuration of Intel Core i5-5200 and RAM 8 GB. The bot testing has been done in a 4*4 grid-based environment (25cm*25cm inner grid), shown in figure 2. During the experiments, the robot should navigate from source to goal and finds the shortest path by detecting an obstacle in the grid environment.

Figure 1. Navigation robot

Figure 2. 4*4 grid environment

Navigation towards the goal:

Data Preparation: IR sensors are prone to external light source (Sunlight), so we have adopted an idea to use an ultrasonic sensor instead of IR sensor [9]. The ultrasonic sensors identify the obstacle while traversing through the 4*4 grid environment. We tried to evaluate the different algorithms by keeping the setup as shown in figure 3(a) and 3(b).

Figure 3 (a). Source position of abot

Figure 3 (b). Goal position of abot
Evaluation. The bot has planted in the grid environment by providing the start and goal state. The bot's traversal is biased towards the forward direction because of its default position setup. We have evaluated the motion of the bot using five different grid-based algorithms:

| Function type | Description |
|---------------|-------------|
| Heuristic Function | Function heuristic(node)  
   Dx=abs (node.x - goal.x)  
   Dy=abs (node.y - goal.y)  
   Return D*(dx+dy)  
   While (locx != goalx || locy != goaly):  
   costarray[3] = g + heuristic(loc x)[locy - 1];  
   costarray[2] = g + heuristic(loc x + 1)[locy];  
   costarray[1] = g + heuristic(loc x)[locy + 1];  
   costarray[0] = g + heuristic(loc x - 1)[locy]; |
| First order x Derivative | Function heuristic(node)  
   Dx=abs (node.x - goal.x)  
   Dy=abs (node.y - goal.y)  
   Return D*(dx+dy)  
   While (locx != goalx || locy != goaly):  
   costarray[2] = g + heuristic(loc x + 1)[locy];  
   costarray[1] = g + heuristic(loc x)[locy + 1];  
   costarray[3] = g + heuristic(loc x)[locy - 1];  
   costarray[0] = g + heuristic(loc x - 1)[locy]; |
| First order y Derivative | Function heuristic(node)  
   Dx=abs (node.x - goal.x)  
   Dy=abs (node.y - goal.y)  
   Return D*(dx+dy)  
   While (locx != goalx || locy != goaly):  
   costarray[0] = g + heuristic(loc - 1)[locy];  
   costarray[1] = g + heuristic(loc)[locy + 1];  
   costarray[2] = g + heuristic(loc + 1)[locy];  
   costarray[3] = g + heuristic(loc)[locy - 1]; |
| Second order x Derivative | Function heuristic(node)  
   Dx=abs (node.x - goal.x) +50  
   Dy=abs (node.y - goal.y)  
   Return D*(dx)  
   While (locx != goalx || locy != goaly):  
   costarray[3] = g + heuristic[60][locy - 1];  
   costarray[2] = g + heuristic[60+1][locy];  
   costarray[1] = g + heuristic[60][locy + 1];  
   costarray[0] = g + heuristic[60 - 1][locy]; |
| Second order y Derivative | Function heuristic(node)  
   Dx=abs (node.x - goal.x)  
   Dy=abs (node.y - goal.y) +50  
   Return D*(dx)  
   While (locx != goalx || locy != goaly):  
   costarray[3] = g + heuristic(loc x)[60 - 1];  
   costarray[2] = g + heuristic(loc x + 1)[60];  
   costarray[1] = g + heuristic(loc x)[60 + 1];  
   costarray[0] = g + heuristic(loc x - 1)[60]; |

Figure 4. Pseudo Codes of different algorithms.
Results in a Grid Environment: The bot is evaluated in a real time by the user by providing the input data for example the source pt. (1, 1) and the goal pt. (4, 4). The bot’s adaptability is identified by providing a dynamic environment. The evaluation has been carried out based on the following metrics.

- Time: Time taken to reach from source to destination.
- Distance: The distance covered by the bot while traversing from source to destination
- No of turns: The total number of turns made by the bot while traversing from source to destination.

We have carried out the evaluation of bot’s motion in three different scenarios as shown in figure 5(a), 5(b) and 5(c). The results of the three scenarios has been tabulated in the table 2, 3 and 4 for comparative purpose.

Scenario 1: Grid without obstacle

Scenario 2: Grid with single obstacle

Scenario 3: Grid with multiple obstacles

![Figure 5(a). Grid without obstacle](image1)

![Figure 5(b). Grid with single obstacle](image2)

![Figure 5(c). Grid with multiple obstacles](image3)
Table 1. Grid without obstacle

| METRICS     | TYPES OF ALGORITHMIC TECHNIQUES |
|-------------|----------------------------------|
|             | Heuristic Value                  | First order x Derivative | First order y Derivative | Second Order x Derivative | Second Order y Derivative |
| Time (s)    | 24                               | 24                       | 18                      | 12                       | 12                       |
| Distance (cm)| 187                              | 187                      | 160                     | 154                      | 154                      |
| No. of Turns| 4                                | 4                        | 4                       | 1                        | 1                        |

Table 2. Grid with single obstacle

| METRICS     | TYPES OF ALGORITHMIC TECHNIQUES |
|-------------|----------------------------------|
|             | Heuristic Value                  | First order x Derivative | First order y Derivative | Second Order x Derivative | Second Order y Derivative |
| Time (s)    | 30                               | 24                       | 20                      | 12                       | 12                       |
| Distance (cm)| 253                              | 214                      | 187                     | 154                      | 154                      |
| No. of Turns| 4                                | 4                        | 3                       | 1                        | 1                        |

Table 3. Grid with multiple obstacles

| METRICS     | TYPES OF ALGORITHMIC TECHNIQUES |
|-------------|----------------------------------|
|             | Heuristic Value                  | First order x Derivative | First order y Derivative | Second Order x Derivative | Second Order y Derivative |
| Time (s)    | 42                               | 36                       | 30                      | 20                       | 24                       |
| Distance (cm)| 424                              | 360                      | 295                     | 154                      | 253                      |
| No. of Turns| 5                                | 5                        | 2                       | 1                        | 1                        |

Table 2 shows the evaluation of the bot’s navigation in the grid environment with no obstacle using five different algorithms. The heuristic approach and the first order x derivative consumed a distance of 187 cm from source to destination during the traversal, the time taken for the traversals 24 seconds and the traversal had 4 turns. The first orderly derivative consumed a distance of 160 cm from source to destination during traversal, the time taken for the traversals 18 seconds and the traversal had 4 turns. These condor derivatives consume a distance of 154 cm from source to destination during the traversal, the time taken for the traversal is 12 seconds and the traversal had 1 turn. The second order x derivative consumed a distance of 154 cm from source to destination during the traversal, the time taken for the traversal is 12 seconds and the traversal had 1 turn.

Similarly, Table 3 and Table 4 shows the evaluation of the bot’s navigation in the grid environment with single obstacle and multiple obstacles using five different algorithms. The heuristic approach consumed
a distance of 253 cm from source to destination during the traversal for a single obstacle whereas a distance of 424 cm has utilized in the multiple obstacle environment. The exerted time for single obstacle is 30 seconds whereas for multiple obstacles traversal time is 42 seconds. While traversal, the bot had 4 turns for a single obstacle environment although for a multiple obstacle environment the bot had 5 turns. The first order x derivative approach manipulated a distance of 214 cm from source to destination for a single obstacle though a distance of 360 cm traversed by a bot in multiple obstacle environment. The bot traversal requires 24 seconds of time for single obstacle whereas time taken for traversing to reach a goal in multiple obstacles is 36 seconds. The bot traversal made 4 turns for single obstacle although in multiple obstacle environment betakes 5 turns. The first order y derivative approach exploits a distance of 187 cm from source to destination during traversal for single obstacle although for multiple obstacle environment the traversal distance is 295 cm. The time taken for traversal in single obstacle environment is 20 seconds whereas in multiple obstacles traversal time is 30 seconds. The bot had 3 turns during traversal in single obstacle circumstance and for multiple obstacles circumstance the bot had 2 turns. The second order x derivative approach consumed a traversal distance of 154 cm from source to goal for both single obstacle and multiple obstacle environment. The time utilized for single obstacle environment is 12 seconds whereas for multiple obstacle environment traversal time is 20 seconds. The bot traversal had 1 turn for both single obstacle and multiple obstacle environment. The second order y derivative approach utilized a traversal distance of 154 cm from source to goal for single obstacle although the traversal distance of 253 cm for multiple obstacle environment. The time taken by the bot for traversing from source to destination is 12 seconds for single obstacle whereas for multiple obstacles the bot takes 24 seconds. The traversal of the bot had 1 turn for both single and multiple obstacle environment.

3. Analysis

Based on the results obtained from the bot’s navigation schemas, the analysis is concluded by concerning the performance metrics.

The table 2 represents the obstacle schema in which, the heuristic approach and the first order x derivative are not providing an encouraging result in terms of the performance metrics. Comparatively, the second-order x derivative and the second-order y derivative approaches yield better results among the other approaches. In the second order x derivative, x is constant whereas y serve as center for the navigation of the bot. In the second order y derivative, y is constant whereas x serve as center for the navigation of the bot is shown in figure 6.

![Figure 6: Analysis of the grid without obstacle for five different approaches](image-url)
The table 3 represents the single obstacle schema in which, the heuristic approach and the first order x derivative are not providing an encouraging result in terms of the performance metrics. Comparatively, the second-order x derivative and the second-order y derivative approaches yield better results among the other approaches. By having x value as a constant value in second order x derivative, y value shows a flexible result for the bot navigation. Where as in second order y derivative, y values take constant and x shows a flexible result for the bot navigation is shown in figure 7.

**Figure 7:** Analysis of the grid with single obstacle for five different approaches

**Figure 8:** Analysis of the grid with multiple obstacles for five different approaches
The table 4 represents the multiple obstacles schema in which, the heuristic approach and the first order x derivative are not providing an encouraging resulting term of the performance metrics. Comparatively, the second-order x derivative approach yield better results among the other approaches. In the second order x derivative, x value is constant whereas y values reveal centre for the navigation of the bot is shown in figure 8. Finally, the most import an aspect to be pointe do there that the second-orders x derivative and the second-order y derivative approaches performs well in the grid environment. The second order x derivative yields a better result compared to other approaches in terms of shortest path selection, time consumption and the number of turns made, because the value of y derivative shows a flexible path traversal by keeping the x derivative e value constant and also furnish an optimal path selection nontematised modelling the bot’ behaviour.

4. Conclusion
The path planning problem of the navigation robot in a dynamic environment is a complex optimization problem with multiple obstacles and constraints. The most important aspect of the path planning problem is to choose the shortest path and collision- free environment to attain a goal. The model is defined as a grid graph with 4*4 size including multiple obstacles. A comparison of five base line approaches in this paper has strong adaptability for navigation robot, which reduces the number of turns and distances to attain a goal. The experiment is designed to verify the performance of five baseline algorithm based on metrics. The results show that the second-order x derivative algorithm is effective and efficient for the path planning problem in the grid environment.

References
[1] Jeddissaravi K., Alitappeh R. J., Pimenta L. C. A. and Guimaraes F. G., “Multi-objective approach for robot motion planning in search tasks”, Applied Intelligence, vol. 45, 2016, pp.305–321.
[2] Tajti F., Burdelis M., “A novel potential field method for path planning of mobile robots by adapting animal motion attributes”, Robotics & Autonomous Systems, vol. 82, 2016, pp. 24–34.
[3] Liu J. H., Yang J. G., Liu H. P., Geng P. and Gao M., “Robot global path planning based on ant colony optimization with artificial potential field”, Transactions of the Chinese Society of Agricultural Machinery, vol. 46, 2015, pp. 18–27.
[4] X. G. Zhang, J. Y. Wang, L. L. Yin, On the future development of the old age for the elderly in China, Business Review, Vol. 23, No. 8, 3-7, 2016.
[5] Ministry of Industry and Information, Development and Reform Commission, Ministry of Finance, Robot industry development plan (2016- 2020 years), Department of Industry and Information, No. 109, 2016.
[6] Santos, Luis, Filipe N. Santos, Sandro Magalhães, Pedro Costa, and Ricardo Reis. "Path Planning approach with the extraction of Topological Maps from Occupancy Grid Maps in steep slope vineyards." In 2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC), pp. 1-7. IEEE, 2019.
[7] Zhu D. Q., Yan M. Z., “Survey on technology of mobile robot path planning”, Control and Decision, vol. 25, 2018, pp. 961–967.
[8] Liu Y. S., Wei N., Sun Y. M., “Path planning algorithm based on grid method for virtual human”, Computer Engineering and Design, vol. 5, 2016, pp. 1229–1230+1267.
[9] Bayili S., Polat F., “Limited-Damage A*: A path search algorithm that considers damage as a feasibility criterion”, Knowledge-Based Systems, vol. 24, 2016, pp. 501–512.
[10] Stentz A., “Optimal and efficient path planning for partially-known environments”, IEEE International Conference on Robotics and Automation. Piscataway, USA: IEEE, 2017: 3310–3317.
[11] Kamran Yeganegi et al 2020 J. Phys.: Conf. Ser. 1530 012110.