Defence Science Journal, Vol. 71, No. 6, November 2021, pp. 748-761, DOI : 10.14429/dsj.71.16563

An Approach to Improve Multi-objective Path Planning for Mobile Robot Navigation using the Novel Quadrant Selection Method

K. Rajchandar*, R. Baskaran, K. Padmanabhan Panchu, and M. Rajmohan

Department of Industrial Engineering, College of Engineering Campus, Guindy Campus, Anna University, Chennai - 600 025, India
**E-mail: rajchandark@gmail.com*

ABSTRACT

Currently, automated and semi-automated industries need multiple objective path planning algorithms for mobile robot applications. The multi-objective optimisation algorithm takes more computational effort to provide optimal solutions. The proposed grid-based multi-objective global path planning algorithm [Quadrant selection algorithm (QSA)] plans the path by considering the direction of movements from starting position to the target position with minimum computational effort. Primarily, in this algorithm, the direction of movements is classified into quadrants. Based on the selection of the quadrant, the optimal paths are identified. In obstacle avoidance, the generated feasible paths are evaluated by the cumulative path distance travelled, and the cumulative angle turned to attain an optimal path. Finally, to ease the robot's navigation, the obtained optimal path is further smoothed to avoid sharp turns and reduce the distance. The proposed QSA in total reduces the unnecessary search for paths in other quadrants. The developed algorithm is tested in different environments and compared with the existing algorithms based on the number of cells examined to obtain the optimal path. Unlike other algorithms, the proposed QSA provides an optimal path by dramatically reducing the number of cells examined. The experimental verification of the proposed QSA shows that the solution is practically implementable.

Keywords: Optimal path planning; Grid approach; Mobile robot navigation; Multi-objective; Path smoothing; Alternative pathway

1. INTRODUCTION

Technological advancements and requirements of autonomous management activities are increasing day by day. The application of autonomous mobile robots in various situations encounter considerable problems such as simultaneous localisation, path-planning[1], navigation, collision avoidance, manipulating objects, communication, and mapping[2-3]. When operating a mobile robot in an active environment, the map changes very frequently[4]. Under such conditions, the successful navigation of mobile robots in a planned path is arduous. Researchers use a motion planning algorithm to avoid such situations, which leads to excess movement of robots[5]. Hence, the path planning (PP) and replanning have to be done rapidly for every updated map. Moreover, the present-day situation demands a path to optimise for more than one objective[6] simultaneously. To solve these issues, the development of an efficient algorithm that ensures an optimal path is essential.

Earlier, PP algorithms were used to increase traversal processing speed in the databases of tree data structures[7]. Later, these algorithms were used in video games and are currently used for mobile robot applications[8]. The PP problems are solved using classical optimisation techniques[9] or soft computing methods[10]. The classical optimisation techniques used the graph or grid map to approach the PP problem as the base environment. While planning the path, neighbourhood exploration techniques select the proper cell that provides the shortest route to reach the target position from the starting position[11]. Therefore, the appropriate selection of neighbours becomes the most vital step in planning a mobile robot path. In the early stages, breadth-first[12] and depth-first search[13] methods were popularly used to find feasible paths without guarantee for optimality. Later, the Bellman-Ford algorithm[14] was developed to provide an optimal path. Dijkstra's algorithm[15] and A* algorithm[16] are widely applied to find the optimal path. Dijkstra's algorithm examines more cells to find the optimal path than the A* algorithm..

In the widely practised A* algorithm, the function performs repeated investigation of selected neighbours (i.e., during the execution of each iteration of all the adjacent eight cells are evaluated for the minimum cost). Further, handling and avoiding the obstacles in the different environments becomes an exhaustive memory search, which leads to high computational effort even for simple situations[17]. JPS (Jump Point Search)[18] uses a technique to complete the grid's scanning to find the exact jump point in a quick time. Unlike the A* algorithm, JPS examines the same cell multiple times during each iteration[19]. In recent times, soft computing approaches such as evolutionary computing, fuzzy inference systems (FIS), probabilistic model, genetic algorithms (GA), Neural

Received : 11 December 2020, Revised : 11 February 2021
Accepted : 13 July 2021, Online published : 22 October 2021

networks optimisation, particle swarm optimisation algorithm (PSO), bio-inspired algorithms, and ant colony optimisation algorithm (ACO) are used for PP[10].

For multi-objective PP problems (MOPP)[20], soft computing approaches are generally preferable to random search, which results in computationally complex recursive cell processing with no guarantee of optimality[21]. Moreover, traditional approaches consume more significant memory and time-intensive operations for finding sub-target, which leads to computational complications in PP. Soft computing algorithms are further combined to solve PP complications to overcome the inadequacy of the traditional approach in PP problem-solving skills.

A combination of the PSO technique with a potential field method (PFM) was proposed to resolve the PP problem. The authors used PSO to optimise obstacle avoidance and PFM to obtain the target position[22]. PSO is used for finding optimal routing and Neural Network algorithms for obstacle avoidance[23]. Another method combines a traditional and intelligent approach to obtain PP by combining the PSO algorithm for global PP. At the same time, the probabilistic roadmap method (PRM) is used for collision avoidance in local PP [24]. A hybrid method combines ACO to find the optimal path and GA with a modified crossover operator to solve the local minima problem occurring during searching the optimal path[25]. The Max-Min Ant System was introduced to solve a path planning problem for exploratory tasks to maximise coverage[26]. Another hybridised PP algorithm using PSO with modified frequency bat (MFB) algorithm for path smoothness and local search (LS) algorithm for finding a viable path is developed[27]. Such a hybridised implementation of the algorithms drastically increases the computation effort in obtaining a path. A variation of the artificial bee colony algorithm with static and dynamic obstacle avoidance in a grid-based environment is proposed[28]. One more combination strategy is studied to resolve the navigation problems involved in a multi-agent system and multiple target tracking. In this study, an environment using an improved artificial bee colony (ABC) technique with evolutionary programming[29] was used. A modified ABC is employed to increase computing speed, and another food source method is incorporated into the base methodology to solve the path planning problem[30]. The above listed hybridised PP based on soft computing approaches provide an optimal path for any static and dynamic environment. However, more feasible paths are generated in a broader environment, leading to a higher computational effort to find one optimal path[31]. Most of the soft computing approaches are fit for the smaller static environment. The real-time implementation of such strategies is more difficult for the mobile robot to decide the optimal path in a split time instance.

A quadtree[32] data structured approach based on spatial/cells decomposition for mobile robot PP was introduced for the graph or grid-based environment. K-Framed Quadtrees approach is proposed for the PP of a mobile robot by utilising the traditional A* algorithm[33]. Hilbert curve traversal, neural network approach and greedy algorithm based PP algorithm developed upon quadtree segmentation[34]. The authors combined several PP techniques to obtain a sub-optimal path for mobile robot navigation. A long-distance PP methodology developed with modified A* combined with any angle PP algorithm in a quadtree structured environment is proposed[35]. The authors claim that quadtree structured cell decomposition is suitable for long distanced environments, i.e., over 100 million nodes, using a grid map.

Internal analysis based PP and obstacle representation method is proposed[36]. The authors repeatedly change the resolution of the quadtree to minimise the configuration space. Also, the authors use graph/interval based PP algorithm and obstacle representation as a separate procedure. The utilisation of such an approach is more complex and time-consuming. Moreover, the presence of excessive partitions, programming on a quadtree structured environment requires extensive coding and additional memory size[37]. Notably, the quadtree spatial decomposition changes its grid pattern for change in the environment. i.e., a separate quadtree decomposition program is essential for each environment that is hugely problematic to program for an environment like static and dynamic, which contains concentric obstacles and large-sized environments. Planning a path using a quadtree leads to more auxiliary turns, which cause redundant rapid bends to reach the target location. Due to a more significant number of turns, the path length becomes sub-optimal, expensive. It needs more memory to store the cluster of neighbouring nodes[38-39].

From the above-cited algorithms, the path generated by the mobile robot entirely relies on two or more approaches, which may be a stand-alone approach (i.e., either traditional or soft computing approach) or a hybrid approach. From the above-stated methods, we observe that every algorithm(s) can produce a successful path. However, most of the algorithms address objectives like (i) Planning an optimal path by minimising the number of sharp turns, (ii) Finding alternative paths, and (iii) Identifying safe paths. The cited algorithms take up ample memory space and more computational time for dealing with dynamic and outsized environments. Low-cost mobile robots find a broad spectrum of applications in various industries like commercial shopping malls, airports, warehouses, and hotels. Such low-cost mobile robots have limited processing capabilities, and hence the computational effort is a significant concern to be addressed. This uneasiness is our research's prime motivation to find an algorithm with the minimum computational effort by minimising the number of cells to be examined to find the optimal path with fewer sharp turns. A grid-based multi-objective PP algorithm [Quadrant selection algorithm (QSA)] is proposed in this research work. Our planning rules are made up of simple IF-Else statements, which any processor can comply with faster. The proposed QSA algorithm developed based on conditional statements consumes less time for finding an optimal path.

2. PROBLEM STATEMENT

Path planning in the grid environment, the robot is assumed to occupy a grid cell and navigates from the starting to the end position using the planned path. A path defined as an ordered set of unit cell coordinates $\{(X_1^R, Y_1^R), (X_2^R, Y_2^R), \ldots\ldots, (X_n^R, Y_n^R), (X^T, Y^T)\}$; ($(X_i^R, Y_i^R)$ - Coordinate of an i^{th} cell where the robot is free to move. Where $i = 1$ *to* n.) is used by the robot to traverse

successively from one cell to another and reach the target cell. Ensure the robot's safety; the obstacles in the map are virtually inflated to a larger size to facilitate the free movement of robots at the adjacent cells of the obstacle. This research aims to attain the smooth optimal path from the starting location to the target location for mobile robot navigation with the minimum number of cells examined (N_{ce}).

2.1 Proposed Quadrant Selection Algorithm

In the grid environment, the robots move in any one of the eight directions, as represented graphically in Fig. 1. The proposed QSA provides a plan to move the robot towards the target from the current cell by searching and identifying the next adjacent cell. Figure 2 represents the significant steps followed in the proposed QSA.

Algorithm 1: logic for the selection of the quadrant in the gird environment

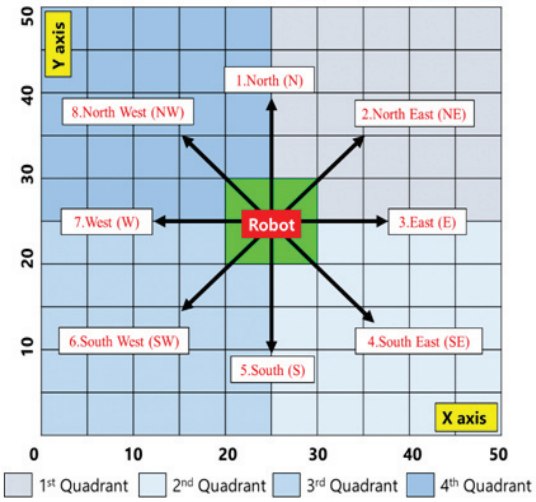

Figure 1. Possible positions, directions of movements (Index) of the mobile robot and target in the grid environment.

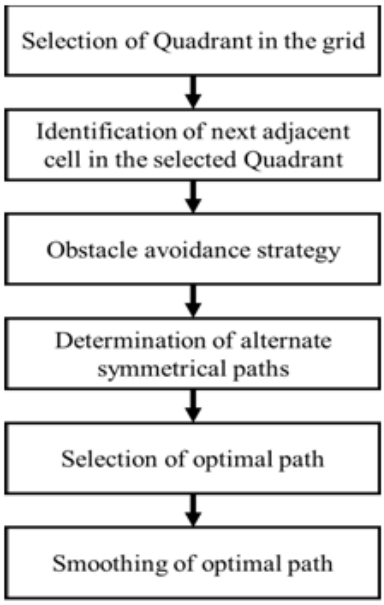

Figure 2. Major steps in the proposed QSA.

Input(s) : Robot, obstacles and target locations.
Output : Smoothened optimal path.

Notations:

$\left(X_i^R, Y_i^R\right)$- Coordinate of an i^{th} cell where the robot is free to move. where, $i = 1$ *to* n.

$\left(X_j^o, Y_j^o\right)$ - Coordinate of j^{th} obstacle. Where $j = 1$ *to* m.

$\left(X^T, Y^T\right)$- Coordinate of the target position.

$P = \{S_1^P, S_2^P, ..., S_a^P\}$- Set of the paths that contain 'a' different feasible path.

$S_u^P = \{(X_1^R, Y_1^R), (X_2^R, Y_2^R), ..., (X^T, Y^T)\}$-Path that contains an ordered set of coordinates. Where $u = 1$ *to* a.

Step 0: Initialise $\left(X_i^R, Y_i^R\right), \left(X_j^o, Y_j^o\right), and\ (X^T, Y^T)$, set $i = 1,\ u = 1$;

Step 1: Create an empty set of paths, $P = \{\varnothing\}$, and ordered open set of coordinates for the path u. $S_u^P = \{\varnothing\}$, where, $u = 1$ *to* a;

Step 2: Append the coordinates of the current position of the robot to set S_u^P. i.e., $S_1^P = \{(X_i^R, Y_i^R)\}$. Increment '$i$' by 1;

Step 3: Check whether the current location is the target location (X^T, Y^T)? , If yes, go to Step 11. Else, continue;

Step 4: Quadrant selection

The quadrants, possible robot positions and their associated target positions are represented in Fig. 1. Based on the logic of Algorithm 1, the next quadrant extension of the path is selected according to the current mobile robot and target position. Once a quadrant is selected, the active investigation for the latest appended coordinate of the path is restricted only to 3 adjacent cells (for example, in Fig. 1, if quadrant one is selected, the active investigation is limited to cells 1, 2 and 3 in quadrant one only) in the selected quadrant instead of all possible eight adjacent cells. This procedure eliminates the cells of other quadrants processed for the search of the path.

Step 5: Identifying the next adjacent cell of the path

Starting from the current position $\left(X_i^R, Y_i^R\right)$ to target positions (X^T, Y^T), the path is planned by appending the cell coordinates to the ordered set S_u^P. After selecting the quadrant, the adjacent cell identified using the logic given in Fig. 3(a).

Step 6: Check whether the chosen cell, occupied with the obstacle? If Yes, continue. Else go to step 2.

Step 7: There existed a possibility of another alternate path when an obstacle encountered. Hence, increment u by 1 and create a new empty path set $S_u^P = \{\ \}$, and the coordinates in the path S_{u-1}^P are copied to the new path set.

For example, in Fig. 3(a), Let us say the first path created is $S_1^P = \{(1,1),(2,2)\}$ and the next possible cell is (3,3) that contains an obstacle. The above step leads to alternate path generation and hence, a new empty path set $S_2^P = \{\ \}$ created and the coordinates from S_1^P are copied to the S_2^P as the path shared up to this point. Hereafter, the coordinates added to S_1^P and S_2^P Be different that represents different alternate paths.

Step 8: Obstacle avoidance sequence

(a) Create a set containing the obstacles encountered in the path planned $(OE) = (X_j^o, Y_j^o)$;

where $j = 1$ *to* m, and increment '*j*'.

(b) The obstacle avoidance sequence starts with the identification of the next possible adjacent cells to avoid the cell containing an obstacle using the logic given in step 5. The above step results in the identification of two potential cells, one at the clockwise direction and another at the counter-clockwise direction to the current cell. The dual direction leads to generating two different paths from the current cell whenever an obstacle is encountered.

(c) Choose the next cell in the clockwise direction and go to step 6.

(d) Remove the coordinate of the obstacle (X_j^o, Y_j^o) from the set OE, and decrement *j* by 1. Then, choose the cell in the counter-clockwise direction and go to step 6.

Step 9: Check whether $OE = \{\varnothing\}$. If Yes, continue; else go to step 8(d).

Step 10: Check whether $u > 1$; If Yes (there exists more than one feasible path), continue else, go to step 12.

Step 11: Optimal path selection

The optimal path selection strategy to find the best among the feasible paths discussed in steps 11a - 11h. The multi-objective parameters that decide the optimal path include the number of turns and path length, and the sum of turns and path length are in different metrics. To overcome this problem, path traversing time was chosen as a standard metric. During movement, the robot's speed was taken to be 0.24 m/s, and the time of turning was assumed as 0.02 sec/degree.

(a) For each feasible path in set *P*, the time required to traverse from the initial position to the target position is calculated. A set $T = \{t_1, t_2, ..., t_u\}$ created to represent the feasible paths' traverse time.

(b) Initialise a temporary variable $s = 1,\ i = 1, t_s = 0$;

(c) From the ordered path set S_u^P with the '*n*' coordinates, the coordinates $(X_i^R, Y_i^R,)and\,(X_{i+1}^R, Y_{i+1}^R)$ are chosen for the computation of time to traverse between the cells using Eqn (1).

$$t_s = t_s + \left[0.02 * \varnothing_i\right] + \left[0.24 * \sqrt{\left(X_i^R - X_{i+1}^R\right)^2 + (Y_i^R - Y_{i+1}^R)^2}\ \right] \quad \text{Eqn (1)}$$

where $\varnothing_i$ - Angle required to orient the current robot pose from the coordinate (X_i^R, Y_i^R) to coordinate (X_{i+1}^R, Y_{i+1}^R).

(d) If $i+1 \neq n$, increment *i* by 1 and go to 11(c). Else, continue;

(e) If $s \neq u$, increment *s* by 1. Else, continue;

(f) From the set *T*, the path S_x^P with minimum traversing time is chosen as the optimal path. The optimal path for the scenario is depicted in Fig. 3(a) and shown in Fig. 3(b).

Step 12: Path smoothing procedure

The optimal path obtained from step 11 contains sharp turns that force the robot to decelerate at every turning, increasing energy consumption and traversing time. To overcome these disadvantages, a path smoothing procedure is introduced Fig. 3(c) represents a smoothed optimal path for the path shown in Fig. 3(b).

(a) The obtained optimal path S_x^P Containing '*n*' ordered coordinates are selected for smoothing. Initialise *i* = 1 and Q = *n*.

(b) Check whether $n > 2$ (to ensure starting and ending coordinates are not the same). If Yes, continue. Else go to step 13;

(c) Check if any straight-line path without any turns exists between ith coordinate (X_i^R, Y_i^R) And Qth coordinate (X_y^R, Y_y^R) of S_x^P. If yes, remove all other coordinates in-between '*i*' and '*Y*' from the set S_x^P, and set *i* = *Q* and *Q* = *n*. Then, go to 12(b). Else, continue;

(d) Decrement *Q* by 1. Then, go to 12(b).

Step 13: The smoothed shortest path P with coordinates in S_x^P is sent to the robot for navigation.

Step 14: End

Function Quadrant _ select $\left((X_i^R, Y_i^R),(X^T, Y^T)\right)$

{

If $\left((X_i^R < X^T) \,\&\, (Y_i^R \le Y^T)\right)$

Quadrant _ select = 'Quadrant 1';

Elseif $\left((X_i^R \ge X^T) \,\&\, (Y_i^R \le Y^T)\right)$;

Quadrant _ select = 'Quadrant *2*';

Elseif $\left((X_i^R \ge X^T) \,\&\, (Y_i^R \ge Y^T)\right)$;

Quadrant _ select = 'Quadrant *3*';

Elseif $\left((X_i^R < X^T) \,\&\, (Y_i^R \ge Y^T)\right)$;

Quadrant _ select = 'Quadrant *4;*

End If

}

End Function

Algorithm 2: logic to find the exact next adjacent cell

Function exact _ next _ cell $\left((X_i^R, Y_i^R),(X^T, Y^T)\right)$

{

If $X_i^R = X^T \,\&\, Y_i^R < Y^T$

exact _ next _ cell $= (X_i^R, Y_i^R + 1)$; $Next\_position = 1$;

Elseif $X_i^R < X^T \,\&\, Y_i^R < Y^T$

exact _ next _ cell $= (X_i^R + 1, Y_i^R + 1)$;

$Next\_position = 2$;

Elseif $X_i^R < X^T \,\&\, Y_i^R = Y^T$

exact _ next _ cell $= (X_i^R + 1, Y_i^R)$; $Next\_position = 3$;

Elseif $X_i^R < X^T \,\&\, Y_i^R > Y^T$

exact _ next _ cell $= (X_i^R + 1, Y_i^R - 1)$; $Next\_position = 4$;

Elseif $X_i^R = X^T \,\&\, Y_i^R > Y^T$

exact _ next _ cell $= (X_i^R, Y_i^R - 1)$;

$Next\_position = 5$;

Elseif $X_i^R > X^T \,\&\, Y_i^R > Y^T$

exact _ next _ cell $= (X_i^R - 1, Y_i^R - 1)$; $Next\_position = 6$;

Elseif $X_i^R > X^T \,\&\, Y_i^R = Y^T$

exact _ next _ cell $= (X_i^R - 1, Y_i^R)$; $Next\_position = 7$;

Elseif $X_i^R > X^T \,\&\, Y_i^R < Y^T$

exact _ next _ cell $= (X_i^R - 1, Y_i^R + 1)$;

$Next\_position = 8$;

End If

}

End Function

3. EXPERIMENTAL VALIDATION OF THE PROPOSED QSA

Most of the simulation-based works fail when it comes to real-time implementation. To validate and check the application of the proposed QSA on real-time conditions, the FIREBIRD V® mobile robot research platform is utilised. The specifications of this robot are given in Table 1.

Table 1. Specifications of the FIREBIRD Robot with Technical details

Specification	Technical detail
Microcontroller	Atmel ATMEGA2560 microcontroller
I/O Communication	2.4GHZ - Wireless Communication (ZigBee)
Power	9.6V DC Supply
Dimension	Dia: 16cm; Height: 10cm
Weight	1.3Kgs
Motion	Two DC motors.
Speed	0.24 m/s
Load capacity	~ 2 kg on a flat surface
Control	Autonomous control / wired or wireless mode

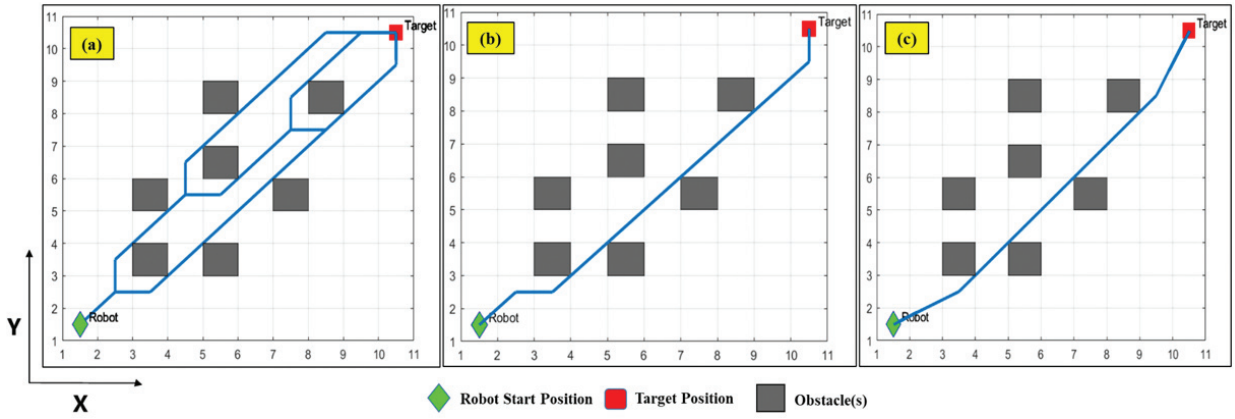

Figure 3. (a) Representation of the set of the feasible path, (b) Optimum path selected from the collection of feasible paths by proposed QSA (c) Smoothed optimal path

The proposed QSA is programmed and executed in the MATLAB R2018a environment installed in an X64-based PC with AMD Ryzen 3 2200U built-in 256SSD/4GBRAM. The sequence of operations is represented as a flow diagram in Fig. 4. The environment is supplied to the base station in the form of a topographical map. The topographical map changed to a grid map with virtually inflating the obstacles to guarantee the robot's safety. The user specifies the grid coordinates of the robot and target. The proposed QSA is executed and provides a path in the form of an ordered set of coordinates. These path coordinates are converted into motion commands by ARDUINO 1.8.12 software and sent to the robot through the Xbee transceiver. The robot receives the coordinates and starts navigating towards the target.

A floor constructed with obstacles in 1.0m x 1.2m space of the size. Occupancy and distribution of obstacles determine the complexity of the generated path. Hence, Low and high levels of obstacle occupancy environments were created to test the proposed QSA.

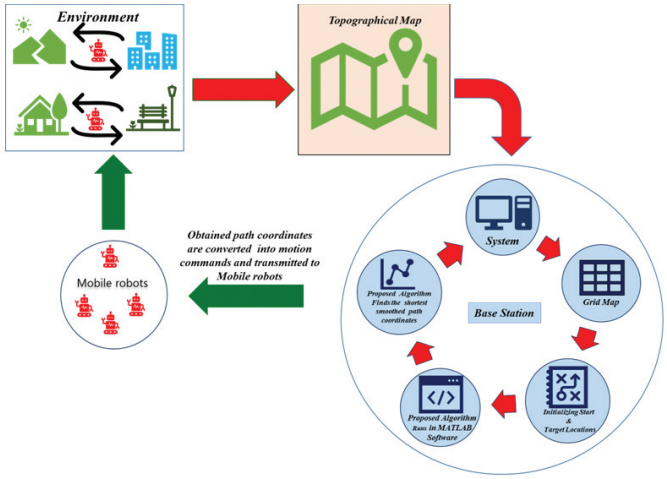

Figure 4. Flow of sequence of operation during experimentation.

3.1 Real-world Scenario with the Low and High Level of Obstacles Occupancy

A real-world experiment was conducted using the FIREBIRD V robot constructed with a low level of obstacle occupancy. (i.e., a minimum number of obstacles) for experimentation, as shown in Fig. 8 (L-I -Starting position; L-II - the initial level of obstacle avoidance; L-III -Smooth travelling of a robot by avoiding the obstacles and L-IV - Successful reach of the target position. Similarly, the experiments were conducted for high-level Occupancy (Fig. 5). During the experimentation, the FIREBIRD V robot's path was almost the same as the path obtained by the QSA. It shows that there was a slight time difference when comparing the experimental output with the calculated time. This brings factors like the robot's wheel slips and error in positional sensing into consideration. However, the results obtained were encouraging, and ease of implementation was witnessed.

4. RESULTS AND DISCUSSION

Every proposed graph-based PP algorithm has its method of addressing the PP problem. As these methods vary, the computational effort also varies. The extent of variation depends on the factors such as the number of cells examined to identify the target and the evaluation requirements for selecting the same cell as the proposed algorithm incorporates a novel change in the primary search methodology to reduce the computational effort and compared with the traditional algorithms and as well as the improved algorithms. Almost all the enhanced algorithms are modified versions of any conventional algorithms to meet the specific application needs. However, the necessary search procedure remains the same, and hence their computational efforts also stay the same. Moreover, the research work that addresses the reduction in computation effort does not clearly state that the decrease is due to high-end computing facilities or improved techniques. Hence to make a fair comparison between the proposed QSA and recent PP algorithms, a novel scheme was developed to evaluate the computational effort.

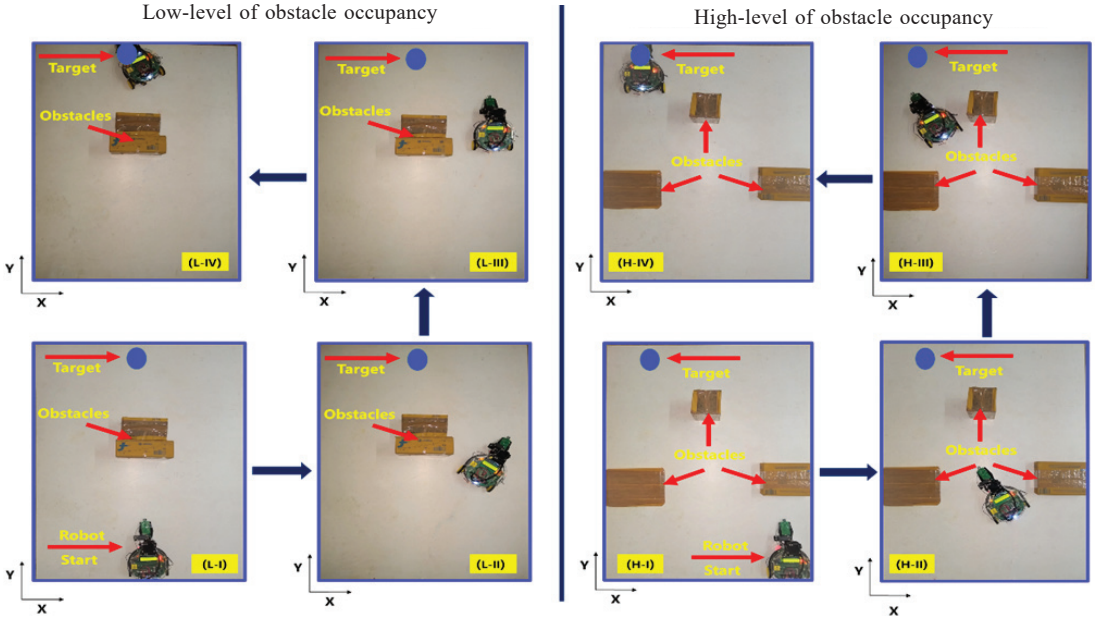

Figure 5. Navigation of FIREBIRD V Robot in the real-time environment.

4.1 Comparison with Traditional PP Algorithm

The widely used traditional PP algorithm like A*(1968), Dijkstra (1977), Breadth-First Search (1984), Best First search (1984), Jump point Search (2012), and Orthogonal JPS (2015) considered verifying best the competence of the proposed algorithm in simulated environments.

The operating procedure of traditional algorithms was found to induce N_{ce} while finding the optimal path. However, the N_{ce} also varied with the type of distribution of obstacles in the environment. In general, specific algorithms that perform well in one kind of environment may not be helpful in the other types. Hence, to test the performance of proposed QSA and compare it with other algorithms, six different grid environments have been taken into consideration. Specifically, (1) Spiral grid, (2) Caved obstacle occupancy grid, (3) Warehouse alike grid, (4) Zigzag obstacles occupancy grid, (5) Random obstacle occupancy grid, and (6) Narrow way grid of size 10 × 10 square units.

The performance measures such as N_{ce} (includes revisiting the same cells multiple times in search of the optimal path), PL, and the percentage of additional cells examined (PACE) concerning the proposed QSA considered for the comparison. The N_{ce} is needed for the optimal path in the environment with Narrow way-obstacle occupancy by the six different algorithms obtained from[40] and shown in Fig. 6(a). The N_{ce} by the proposed QSA is graphically presented in Fig. 6(b) for the same environment. The proposed QSA was tested in the specified six different environments, and the obtained smoothed path is shown in Fig. 7. The performance measures of the proposed QSA were compared with the other algorithms, as shown in Table 2.

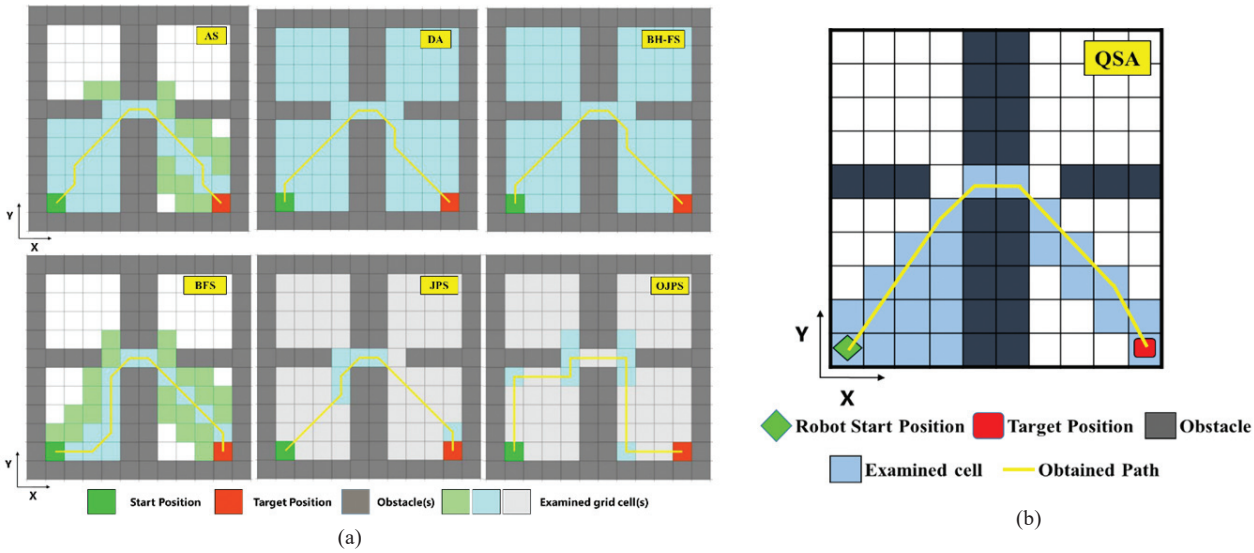

Figure 6. (a) N_{ce} for obtaining an optimal path by different algorithms in narrow way - obstacle occupancy environment and (b) N_{ce} for obtaining an optimal path by the proposed QSA in narrow way - obstacle occupancy environment.

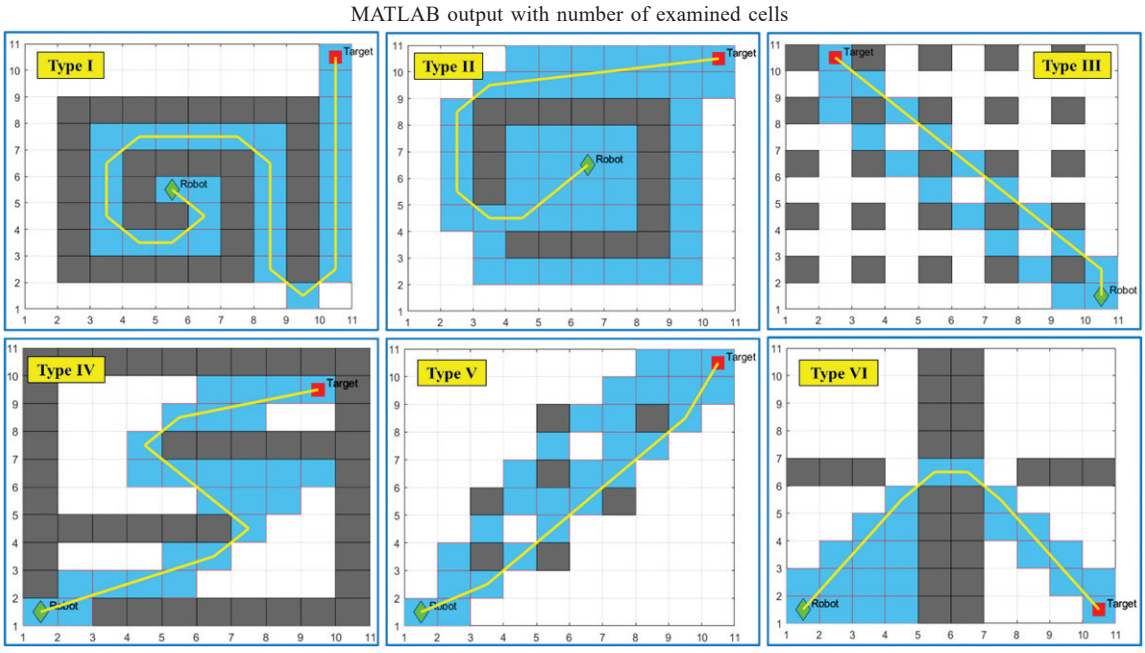

Figure 7. Smoothened optimal path obtained by the QSA for different environments.

Table 2. Performance Comparison of the QSA with some popular existing algorithms

Environment	Performance measure	A* (AS)	Dijkstra (DA)	Breadth-first search (BH-FS)	Best first search (BFS)	Jump point search (JPS)	Orthogonal JPS (OJPS)	Proposed quadrant selection algorithm (QSA)
Type I	N_{ce}	68	92	92	66	92	70	49
	PL	27.9	27.9	27.9	27.9	27.9	32	27.9
	PACE	27.94	46.73	46.73	25.75	46.73	30	-
Type II	N_{ce}	109	164	164	77	118	156	68
	PL	17.07	17.07	17.07	17.07	17.07	20	16.72
	PACE	37.61	58.53	58.53	11.68	42.37	56.41	-
Type III	N_{ce}	43	150	145	43	67	174	21
	PL	12.31	12.31	12.31	12.31	12.31	17	12.313
	PACE	56.41	86	85.51	51.16	68.65	87.93	-
Type IV	N_{ce}	92	112	112	71	75	120	60
	PL	17.31	17.31	17.31	17.31	17.31	22	16.731
	PACE	34.78	46.42	46.42	15.49	20	50	-
Type V	N_{ce}	68	186	186	52	110	313	62
	PL	13.31	13.31	13.1	13.9	13.31	18	12.95
	PACE	8.82	66.66	66.66	-	43.63	80.19	-
Type VI	N_{ce}	76	152	152	53	94	117	60
	PL	14.31	14.31	14.31	15.49	14.31	19	13.82
	PACE	21.05	60.52	60.52	-	36.17	48.71	-

From Table 2, the path lengths for environments I and III are almost the same compared with the other algorithms. The above similarity is because there is no possibility of path smoothing due to the nature of obstacle occupancy in the environments. The PL obtained by the proposed QSA provides better results for the types II, IV, V, and VI environments where smoothing is possible. The proposed QSA uses minimum numbers of cells for examination (shown in Fig. 8) to obtain the optimal path except for the BFS algorithm in type V and VI environments. In this case, though the BFS algorithm examines the minimum number of cells, it does not provide an optimal path. The percentage of additional cells examined between the proposed QSA method and traditional algorithms is shown in Fig. 9.

4.2 Scalability Test

For measure the scale-up capability, various grid resolution has experimented in proposed QSA. Table 3 shows the operating time for the completion of path planning in different grid sized environments. The proportional increase in the computational time is less for a commensurate increase in the grid size, which shows that the algorithm is competent in scalable environments. The set of benchmark grid maps with different resolutions is listed in Figs. 10 (a-l).

4.3 Comparison with Recent MOPP Algorithms

The computational effort of solving path planning problems depends on sequential steps involved in searching the path and grid size. Most modified algorithms use the

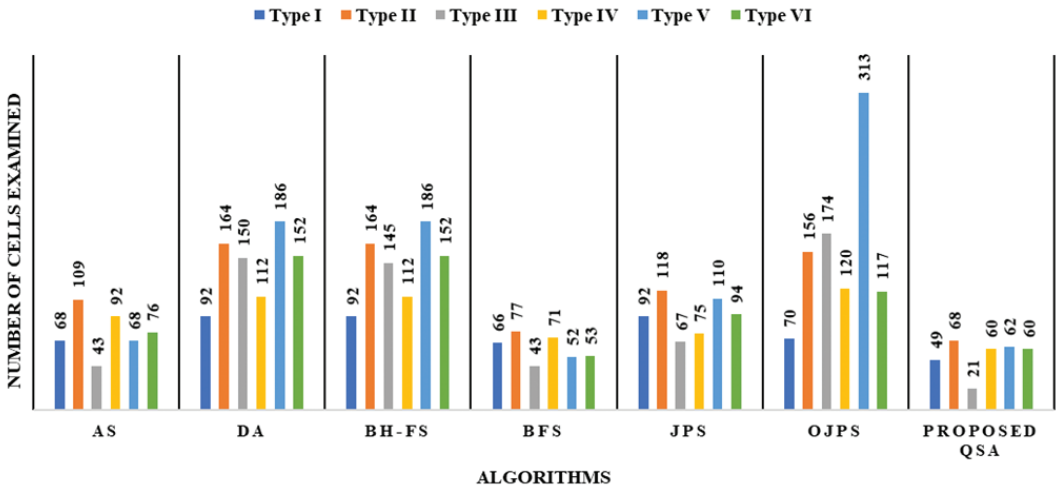

Figure 8. Evaluation of the proposed QSA with traditional algorithms concerning the N_{ce}.

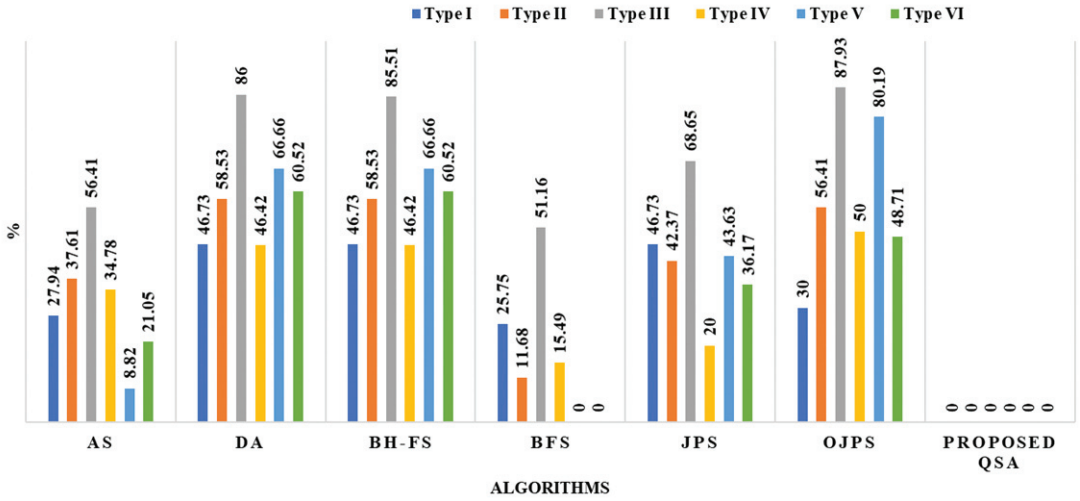

Figure 9. Evaluation of the proposed QSA with traditional algorithms concerning the percentage of additional cells examined.

Table 3. Operating time for proposed QSA

Grid size	Percentage of increase in the number of the grid cell	Average operating time (sec)
10X10	-	0.0325
20X20	300	0.0336
30X30	800	0.0351
40X40	1500	0.0367
60X60	3500	0.0368

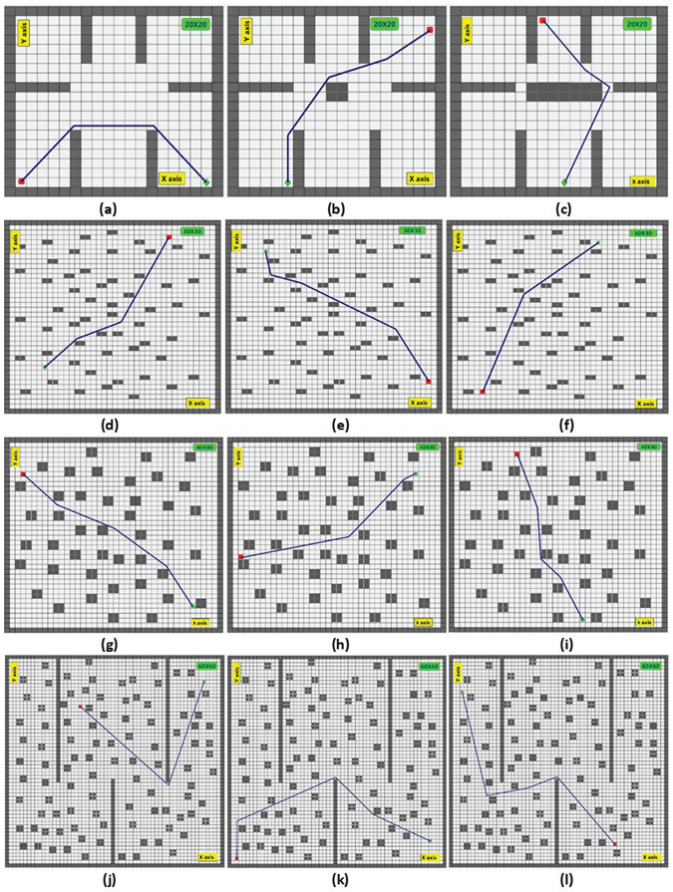

Figure 10. Set of benchmark grid maps with various resolutions (a-l) used for performance evaluation.

traditional algorithm as a base method of investigation, and novelty is meant for specific objectives in multi-objective problems. The modified algorithms were developed using the latest computing technology at that time. Also, researchers find it difficult to compare the computational time of algorithms as it is necessary to create the program for available algorithms in common platforms. Hence, it is proposed to evaluate the computational effort based on the sequence of the operations involved in searching the path.

Standards for evaluation of algorithms (Table 5) based on the percentage of cells examined. The assessment is classified as High - More than 80% or Medium -50% to 80% and Low - less than 50%. Also, each cell's examination requires operational requirements such as investigation (i.e., check for presence or absence of obstacles) and mathematical computations (i.e., distance calculation). Based on each cell's operational requirements, the levels are classified as High, Medium, and Low (High: Operational investigation + Mathematical computation; Medium: Only mathematical computation; Low: Only operational investigation). The computational effort is represented in Table. 5, based on the percentage of cells examined, operational requirements, and the same evaluation as per the details given in Table 4.

Based on the comparison, we found that the algorithms[41-44] are simple modifications made in the A* algorithm, and it has a separate function(s) to solve additional objective(s). In[41], the authors proposed Waypoint Refining Path Smoother (WRPS) for smoothing the path obtained. In[42], the authors modified the A* algorithm to get the smoothed path. In[43], Adaptive Window Approach function for path smoothness and safety navigation,

Table 4. Scheme developed for the evaluation of computational effort

		% of examined cells		
		H	**M**	**L**
Operational requirement	H	*VH*	*H*	*M*
	M	*H*	*M*	*L*
	L	*M*	*L*	*VL*

H- High; M-Medium; L-Low; VH– Very High; Very Low-VL

Table 5. Comprehensive comparison of recent improved/modified algorithms with proposed QSA algorithm

Methods and algorithm	Year	No. of cells examined (N_{ce})	Operational requirements	Computational effort	Smoothing	Alternative paths	Safety
Smoothed A* algorithm with and without WRPS[41]	2019	M	H	H	YES	NA	NA
Smoothed A* algorithm[42]	2019	M	M	M	YES	NA	NA
Safe A* algorithm and adaptive window approach[43]	2020	M	M	M	YES	NA	YES
A* with FSVM and GRNN[44]	2017	M	H	H	YES	NA	NA
Improved Dijkstra's algorithm (IDA)[45]	2019	H	H	VH	NA	NA	NA
Multi-objective hybrid PP algorithm GRTOP using Dijkstra's algorithm[46]	2019	H	H	VH	NA	YES	NA
Multi-objective Dijkstra's algorithm[47]	2019	H	H	VH	NA	YES	NA
Multi-objective route optimisation by using Dijkstra's algorithm[48]	2020	H	H	VH	NA	NA	NA
Adopted JPS for local PP[49]	2017	M	H	H	NA	NA	NA
SD-JPS PP algorithm[50]	2019	L	L	VL	NA	NA	NA
Combination of neural network and fuzzy techniques[51]	2020	H	H	VH	NA	NA	NA
Improved ant colony algorithm[52]	2017	M	H	H	NA	NA	NA
Improved ant colony algorithm[53]	2019	M	H	H	NA	NA	NA
Artificial bee colony algorithm[54]	2020	H	H	VH	NA	NA	NA
Hybrid PSO-MFB optimisation algorithm[55]	2020	H	H	VH	YES	NA	NA
MOPP using enhanced genetic algorithm[56]	2019	H	H	VH	YES	NA	YES
Proposed QSA	-	L	L	VL	YES	YES	YES

and in[44], fuzzy support vector machine (FSM) and general regression neural network (GRNN) functions incorporated with A* algorithm. Still, the heuristic function in the A* algorithm itself has high computation time and integrating additional functions leads to an increase in run time and more memory requirements. In[45-47] a well-recognised Dijkstra algorithm was used. In which the fundamental operation is searching for an entire area for finding adjacent nodes. Though[45] stated an improved Dijkstra's algorithm, the computational effort and operational requirements were found to be very high. There is no evidence in the methodology that claims to reduce the high computational effort of basic Dijkstra's algorithm.

Moreover, achieving the multi-objective goal requires dedicated colossal memory space and increased run time to the most famous traditional approaches like A* and Dijkstra's algorithm. It is also challenging to implement such memory hungry algorithms in small mobile robots with less processing capability. The proposed QSA MOPP algorithm be the best alternative for currently available traditional approaches with less computational effort. The additional main advantage is to provide alternative paths apart from the optimal path for better navigational purposes during execution.

Later, a well-known time-optimal JPS algorithm[48-49] which possesses fundamentals A* algorithm's heuristic function, is compared with the proposed algorithm. In improved/modified JPS as a Safe Distance (SD) JPS[50] algorithm, the searching ability is superior to other traditional algorithms with fewer cells examined. However, there are no instances reported with the application of JPS to a multi-objective scenario. The operational requirements are very high if objectives like smoothing, alternative paths, or safety are incorporated further. Finally, some recent soft computing approaches to solve MOPP problems[51-56] were compared. These soft computing approaches are more suitable for complex/uncertain environments and target tracking purposes, where time is not a significant constraint. For an austere environment, several alternative paths are generated. It is a time-consuming process to identify the optimal path among the available paths. Suppose the environment becomes a little complex or broader in size. In that case, the computational time for finding the optimal is very high. An enhanced genetic algorithm[56] has shown some multi-objective capacity. However, the number of cells examined, computational efforts, and operational efforts are very high compared to the proposed QSA.

The design of this study is to confirm whether the proposed MOPP algorithm shows an improved performance in minimising cell examined, computational and operational efforts. For this purpose, two different experimental grids (i.e., low level and high-level obstacle occupancy) were constructed. Followed by the proposed MOPP algorithm tested

in six different environments and the performance compared with traditional algorithms. During the evaluations, we found that every PP algorithm has successfully achieved the optimal PP. However, most of the algorithms repeatedly examine adjacent grid cells to find their sub-target(s). This recurrent operational activity impacts the higher computational effort and more operational requirements for solving any simple or moderated PP problems.

As the proposed QSA needs less operational requirement and uses a minimum number of cells for examination to find the optimal path, it is found to be competitive in terms of computational effort compared with the other algorithms (shown in Fig. 10). The proposed QSA offers several alternate feasible paths with a minimum number of examined cells that are predominantly useful in multi-robot situations where the robots can use different paths to avoid congestion and traffic delays. The alternative paths are very useful in emergency handling situations where the optimal path is not operable due to real-time constraints.

5. LIMITATIONS

The proposed path planning approach is evidenced to be more effective and capable of solving the path planning problem with a minimised examination of neighbour cells. However, this research does not provide a complete picture of the assessment in different standards or specifications. Here some examples are, (1) When the number of turns in an obtained optimal path is higher in number may lead to unattainable/failure in the smoothening procedure. (2) Similarly, in some instances, there are identical optimal paths obtained with the clockwise and counter-clockwise direction of movement to the obstacle(s) position. In such examples, human intervention is required to resolve the ambiguity. (3) Furthermore, there was a limitation on having less significant prior research on minimising neighbouring cells' examination. Among various available methods, SD-JPS[50] is the only method that significantly reduced the number of neighbouring cells' examinations to the A* algorithm. In contrast, other examined cells [arbitrarily selected cells] were considered. (4) With various cross-platforms and ambiguous data and information, the preparation of comparative study becomes tedious.

6. CONCLUSION

This research develops a new effortless multi-objective path planning algorithm to achieve the smoothed optimal path by examining the minimum number of cells. The performance of the proposed QSA in different environments was compared with six popular all-time algorithms and validated through real-time experimentations. It is found that the proposed QSA outperformed all the other algorithms in various environments for providing the optimal path with a reduced number of cells examination. This algorithm also offered some feasible alternate paths that are useful in multi-robot situations to avoid traffic congestions. The developed algorithm is valuable for time-critical emergency handling situations where the optimal path is not an operational constraint. In the future, this approach can continue solving multi-objective optimal PP problems in complex and dynamic 3D environments.

REFERENCES

1. Han, J. A surrounding point set approach for path planning in unknown environments. *Comput. Indust. Eng.,* 2019, **133**, 121-130. doi: 10.1016/j.cie.2019.05.013
2. Latombe, J.C. Robot motion planning. *Springer Science & Business Media,* 2012, (**124**).
3. Bugmann, G.; Lu, S. & Chung, J.H. Weighted path planning is based on collision detection. *Industrial Robot: An International Journal*, 2005. doi: 10.1108/01439910510629208
4. Thrun, S. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence,* 1998, **99**(1), 21-71. doi: 10.1016/S0004-3702(97)00078-7
5. Seda, M. & Pich V. Robot motion planning using generalized Voronoi diagrams., 2008. Target, 1, p.q2.
6. Mittal, S. & Deb. Three-dimensional offline path planning for UAVs using multi-objective evolutionary algorithms. *IEEE,* 2007, pp. 3195-3202. doi: 10.1109/CEC.2007.4424880
7. Horowitz, S.L. & Pavlidis T. Picture segmentation by a tree traversal algorithm. *Journal of the ACM (JACM),* 1976, **23**(2), 368-388. doi: 10.1145/321941.321956
8. Daniel, K.; Nash, A.; Koenig, S. & Felner, A. Theta*: Any-angle path planning on grids. *Journal of Artificial Intelligence Research*, 2010, **39**, 533-579. doi: 10.1613/jair.2994
9. Schouwenaars, T.; De Moor, B.; Feron, E. & How, J. Mixed-integer programming for multi-vehicle path planning. *In* European Control Conference (ECC), 2001, pp. 2603-2608. IEEE. doi: 10.23919/ECC.2001.7076321
10. Zhang, H.M. Path planning methods of mobile robots based on soft computing techniques, 2011. *Advanced Materials Research,* **216**, 677-680. doi: 10.4028/www.scientific.net/AMR.216.677
11. Drugan, M.M. & Thierens, D. Stochastic Pareto local search: Pareto neighborhood exploration and perturbation strategies. *Journal of Heuristics,* 2012, **18**(5), 727-766. doi: 10.1007/s10732-012-9205-7
12. Bundy, A. & Wallen, L. Breadth-first search. In Catalogue of artificial intelligence tools 1984, pp. 13-13. *Springer,* Berlin, Heidelberg. doi: 10.1007/978-3-642-96964-5_1
13. Tarjan, R. Depth-first search and linear graph algorithms. *SIAM Journal on Computing*, 1972, **1**(2), 146-160. doi: 10.1137/0201010
14. Bellman, R. On a routing problem. *Quarterly of applied Mathematics,* 1958, **16**(1), 87-90. doi: 10.1090/qam/102435
15. Knuth, D.E. A generalization of Dijkstra's algorithm. *Information Processing Letters,* 1977, **6**(1), 1-5. doi:10.1016/0020-0190(77)90002-3
16. Hart, P.E.; Nilsson, N.J. & Raphael, B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 1968,

4(2), 100-107.
doi: 10.1109/TSSC.1968.300136
17. DuchoĖ, F.; Babineca, A.; Kajana, M.; BeĖoa, P.; Floreka, M.; Ficoa, T. & Jurišicaa, L. Path planning with modified a star algorithm for a mobile robot. *Procedia Engineering*, 2014, **96,** 59-69.
doi: 10.1016/j.proeng.2014.12.098
18. Witmer, N. Jump point search explained. 2013-09-22 J. http://zeruwidth.com/2013/05/05/jump-point-search-explained, HTML.
19. Harabor, D. & Grastien, A. Improving jump point search. *In* Proceedings of the International Conference on Automated Planning and Scheduling, 2014, **24**(1).
20. Wang, Z.; Li, Y. & Yao, Y.-A.Motion and path planning of a novel multi-mode mobile parallel robot based on chessboard-shaped grid division. *Industrial Robot*, 2018, **45**(3), 390-400.
doi: 10.1108/IR-01-2018-0001
21. Zadeh, L.A. Soft Computing and fuzzy logic. In Fuzzy Sets, Fuzzy Logic, and Fuzzy Systems: Selected Papers by Lotfi a Zadeh, 1996. pp. 796-804
doi: 10.1142/9789814261302_0042
22. Mandava R.K.; Bondada S. & Vundavilli P.R. An Optimized Path Planning for the Mobile Robot Using Potential Field Method and PSO Algorithm. In: Bansal J., Das K., Nagar A., Deep K., Ojha A. (eds) Soft Computing for Problem Solving. Advances in Intelligent Systems and Computing, 2019, **817**. Springer, Singapore.
doi: 10.1109/ICUAS.2019.8798031
23. Tavoosi, V; Marzbanrad, J. & Golnavaz, M. Optimized path planning of an unmanned vehicle in an unknown environment using the PSO algorithm. *In* IOP Conference Series: *Mater. Sci. Eng.*, 2020, **671**(1), 012009. IOP Publishing.
doi: 10.1088/1757-899X/671/1/012009
24. Masehian E. & Sedighizadeh, D. A multi-objective PSO-based algorithm for robot path planning. *In* IEEE International Conference on Industrial Technology, 2010, pp. 465-470.
doi: 10.1109/ICIT.2010.5472755.
25. Châari, I.; Koubâa, A.; Bennaceur, H.; Trigui, S. & Al-Shalfan, K. smartPATH: A hybrid ACO-GA algorithm for robot path planning. *In* 2012 IEEE Congress on Evolutionary Computation, Brisbane, QLD, 2012, pp. 1-8.
doi: 10.1109/CEC.2012.6256142.
26. Valéria de C. Santos, Fernando E. B. Otero, Colin Johnson, Fernando S. Osório, & Cláudio F. M. Toledo. Exploratory path planning for mobile robots in dynamic environments with ant colony optimization. *In* Proceedings of the 2020 Genetic and Evolutionary Computation Conference (GECCO '20). Association for Computing Machinery, New York, NY, USA, 2020, 40–48.
doi:10.1145/3377930.3390219
27. Ajeil, F.H.; Ibraheem, I.K.; Sahib, M.A. & Humaidi, A.J. Multi-objective path planning of an autonomous mobile robot using a hybrid PSO-MFB optimization algorithm. *Applied Soft Computing,* 2020, **89**, 106076.
doi: 10.1016/j.asoc.2020.106076
28. Goel, P. & Singh, D. Efficient ABC algorithm for dynamic path planning. *Int. J. Comput. Appl.,* 2014, **88**(2).
29. Faridi, AQ.; Sharma, S.; Shukla, A.; Tiwari, R. & Dhar, J. Multi-robot multi-target dynamic path planning using artificial bee colony and evolutionary programming in unknown environment. *Intelligent Service Robotics.* 2018, **11**(2), 171-86.
doi: 10.1007/s11370-017-0244-7
30. Xu, F.; Li, H.; Pun, C.M.; Hu, H.; Li, Y.; Song, Y. & Gao, H. A new global best guided artificial bee colony algorithm with application in robot path planning. *Applied Soft Computing*, 2020, **88**, p.106037.
doi: 10.1016/j.asoc.2019.106037
31. Dunbabin, M. & Marques, L. Robots for environmental monitoring: Significant advancements and applications. *IEEE Robotics Automation Mag.,* 2012, **19**(1), 24-39.
doi: 10.1109/MRA.2011.2181683
32. Noborio, H.; Naniwa, T. & Arimoto, S. A quadtree-based path-planning algorithm for a mobile robot. *Journal of Robotic Systems,* 1990, **7**(4), 555-574.
doi: 10.1002/rob.4620070404
33. Rodrigues, A.; Costa, P. & Lima, J. The K-Framed Quadtrees Approach for Path Planning Through a Known Environment. In: Ollero A., Sanfeliu A., Montano L., Lau N., Cardeira C. (eds) ROBOT 2017: Third Iberian Robotics Conference. ROBOT 2017. *Advances in Intelligent Systems and Computing*, 1990, **693**, *Springer*, Cham.
doi: 10.1007/978-3-319-70833-1_5
34. Ma, Y.; Sun, H.; Ye, P. & Li, C. Mobile robot multi-resolution full coverage path planning algorithm. *In* 5th International Conference on Systems and Informatics (ICSAI), Nanjing, 2018, pp. 120-125.
doi: 10.1109/ICSAI.2018.8599478.
35. Shah B. C. & Gupta, S. K. Long-distance path planning for unmanned surface vehicles in complex marine environment. *IEEE J. Oceanic Eng.,* 2020, **45**(3), 813-830.
doi: 10.1109/JOE.2019.2909508.
36. Mäenpää, P.; Aref, M.M. & Mattila, J. FORMI: A Fast Holonomic Path Planning and Obstacle Representation Method Based on Interval Analysis. *In* IEEE International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM), Bangkok, Thailand, 2019, pp. 398-403.
doi:10.1109/CIS-RAM47153.2019.9095822.
37. Mora, E. G.; Cagnazzo, M. & Dufaux, F. AVC to HEVC transcoder based on quadtree limitation. *Multimedia Tools and Applications,* 2017, **76**(6), 8991-9015.
38. Petres, C.; Pailhas, Y.; Patron, P.; Petillot, Y.; Evans J. & Lane, D. Path planning for autonomous underwater vehicles. *IEEE Trans. Robotics,* 2007, **23(**2), 331-341.
doi: 10.1109/TRO.2007.895057.
39. Hwang, Joo Young; Kim, Jun Song; Lim, Sang Seok & Park, Kyu Ho. A fast path planning by path graph optimization. *In* IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, 2003, **33**(1),

121-129.
doi: 10.1109/TSMCA.2003.812599.
40. Xueqiao (joe) xu, Q.I.A.O. 2014. PathFindingjs. [Online]. [15 May 2020]. Available from: https://qiao.github.io/PathFinding.js/visual/
41. Song, R.; Liu, Y. & Bucknall, R. Smoothed a* algorithm for practical unmanned surface vehicle path planning. *Applied Ocean Research*, 2019, **83**, 9-20. doi: 10.1016/j.apor.2018.12.001
42. Gunawan, S.A.; Pratama, G.N.P.; Cahyadi, A.I.; Winduratna, B.; Yuwono, Y.C.H. & Wahyunggoro, O. Smoothed A-star algorithm for nonholonomic mobile robot path planning. *In* International Conference on Information and Communications Technology (ICOIACT), Yogyakarta, Indonesia, 2019, pp. 654-658. doi: 10.1109/ICOIACT46704.2019.8938467.
43. Zhong, X.; Tian, J.; Hu, H. & Peng, X. Hybrid path planning based on safe A* Algorithm and adaptive window approach for mobile robot in large-scale dynamic environment. *Journal of Intelligent & Robotic Systems,* 2020, pp.1-13. doi: 10.1007/s10846-019-01112-z
44. Chen, J.; Jiang, W.; Zhao, P. & Hu, J. A path planning method of anti-jamming ability improvement for autonomous vehicles navigating in off-road environments. *Industrial Robot: An International Journal,* 2017. doi: 10.1108/IR-11-2016-0301
45. Wenzheng, L.; Junjun, L. & Shunli, Y. An Improved Dijkstra's Algorithm for Shortest Path Planning on 2D Grid Maps. *In* IEEE 9th International Conference on Electronics Information and Emergency Communication (ICEIEC), 2019, pp. 438-441. doi: 10.1109/ICEIEC.2019.8784487
46. Fink, W.; Baker, V.R.; Brooks, A.J.W.; Flammia, M.; Dohm, J.M. & Tarbell, M.A. Globally optimal rover traverse planning in 3D using Dijkstra's algorithm for multi-objective deployment scenarios. *Planetary and Space Science*, 2019, **179**, p.104707. doi: 10.1016/j.pss.2019.104707
47. Schäfer, L.E. & Ruzika, S. On Variants of the Single-criterion and Multiobjective Near-Shortest Paths Problem. *In* Multikriterielle Optimierung und Entscheidungsunterstützung, 2019, pp. 17-30, doi: 10.1007/978-3-658-27041-4_2
48. Hossain, M.A.; Ahmedy, I.; Harith, M.Z.M.; Idris, M.Y.I.; Soon, T.K.; Noor, R.M. & Yusoff, S.B. Route Optimization by using Dijkstra's Algorithm for the Waste Management System. *In* Proceedings of 2020 The 3rd International Conference on Information Science and System, 2020, pp. 110-114. doi: 10.1145/3388176.3388186
49. Zhou, K.; Yu, L.; Long, Z. & Mo, S. Local path planning of driverless car navigation based on the jump point search method under the urban environment. *Future Internet,* 2017, **9**(3), p.51. doi:10.3390/fi9030051
50. Zheng, X.; Tu, X. & Yang, Q. Improved JPS algorithm using new jump point for path planning of mobile robot. *In* IEEE International Conference on Mechatronics and Automation (ICMA) 2019, pp. 2463-2468. doi:10.1109/ICMA.2019.8816410
51. Martínez, F.; Penagos, C. & Pacheco, L. Scheme for motion estimation based on an adaptive fuzzy neural network. *Telkomnika,* 2020, **18**(2), 1030-1037. doi: 10.12928/TELKOMNIKA.v18i2.14752
52. Liu, J.; Yang, J.; Liu, H.; Tian, X. & Gao, M. An improved ant colony algorithm for robot path planning. *Soft Computing*, 2017, **21**(19), 5829-5839. doi: 10.1007/s00500-016-2161-7
53. Luo, Q.; Wang, H.; Zheng, Y. & He, J. Research on path planning of mobile robots based on an improved ant colony algorithm. *Neural Computing and Applications,* 2020, **32**(6), 1555-1566. doi: 10.1007/978-3-030-37436-5_11
54. Xu, F.; Li, H.; Pun, C.M.; Hu, H.; Li, Y.; Song, Y. & Gao, H. A new global best guided artificial bee colony algorithm with application in robot path planning. *Applied Soft Computing*, 2020, **88**, 106037. doi: 10.1016/j.asoc.2019.106037
55. Ajeil, F.H.; Ibraheem, I.K.; Sahib, M.A. & Humaidi, A.J. Multi-objective path planning of an autonomous mobile robot using a hybrid PSO-MFB optimization algorithm. *Applied Soft Computing*, 2020, **89**, 106076. doi: 10.1016/j.asoc.2020.106076
56. Nazarahari, M.; Khanmirza, E. & Doostie, S. Multi-objective multi-robot path planning in a continuous environment using an enhanced genetic algorithm. *Expert Systems with Applications,* 2019, **115**, 106-120. doi: 10.1016/j.eswa.2018.08.008

CONTRIBUTORS

Mr K. Rajchandar received a Master of Engineering in Industrial Engineering from Sudharsan Engineering College (Affiliated to Anna University, Tiruchirappalli, T.N., India), in 2011. He also finished a Master of Business Administration in Production and Operation Management from Alagappa University, Tamil Nadu, in 2017. He is an active Research Scholar towards receiving a PhD in the Department of Industrial Engineering, Anna University, Chennai, India. His common research interests include Robot path/motion planning, multi-robot collaboration, and autonomous mobile robot navigation optimisation.
His contribution towards this research is the design of the algorithm, preparation of experimental setup and data collection, primary manuscript preparation.

Prof. R. Baskaran currently working as Professor in the Department of Industrial Engineering, College of Engineering Guindy, Anna University, Chennai. He did his Bachelors in Mechanical Engineering from Madras University (1991), Masters in Industrial Engineering (1994) from College of Engineering Guindy, Anna University, Chennai. He also did his doctoral thesis in the performance evaluation of bus depots and bus routes for metro cities (2011). His research interests include Route optimisation, vehicle scheduling, and sustainability in higher education.
He contribute in this study is the identification of multiple objectives related to path planning and evaluation of the objectives.

Prof. K. Padmanabhan Panchu currently working as Assistant Professor in the Department of Industrial Engineering, College of Engineering Guindy, Anna University, Chennai. He did his Bachelors in Production Engineering from Government College of Technology, Coimbatore (2004), Masters in Industrial Engineering (2008) from College of Engineering Guindy, Anna University, Chennai. He did his doctoral thesis in multi-objective optimisation of multi-robot task allocation. His research interests include Multi-robot systems, path planning and scheduling, and operations management.
He has contributed to this article in identifying the problem, experimental design and algorithm design.

Prof. M. Rajmohan currently working as Professor in the Department of Industrial Engineering, College of Engineering Guindy, Anna University, Chennai. He did his Bachelors in Agricultural Engineering from College of Agricultural Engineering, Tamilnadu Agricultural University (1993 - 1998), Masters in Industrial Engineering (1999 - 2001) from College of Engineering Guindy, Anna University, Chennai. He did his doctoral work in vehicle route optimisation with time windows. His research interests include Vehicle routing optimisation, logistic and supply chain management, meta-heuristics and design of experiments.
He has contributed to this article in identifying the comparison metrics for evaluation and analysis of results.