Received August 28, 2020, accepted August 31, 2020, date of publication September 3, 2020, date of current version September 16, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3021605

Path Planning for Mobile Robot Navigation in Unknown Indoor Environments Using Hybrid PSOFS Algorithm

MOHD NADHIR AB WAHAB[1], (Member, IEEE), CHING MAY LEE[1], MUHAMMAD FIRDAUS AKBAR[2], (Member, IEEE), AND FADRATUL HAFINAZ HASSAN[1]

[1]School of Computer Sciences, Universiti Sains Malaysia, Penang 11800, Malaysia
[2]School of Electrical and Electronic Engineering, Engineering Campus, Universiti Sains Malaysia, Penang 14300, Malaysia

Corresponding authors: Mohd Nadhir Ab Wahab (mohdnadhir@usm.my) and Muhammad Firdaus Akbar (firdaus.akbar@usm.my)

This work was supported in part by the USM Short Term Grant PKOMP/6315262 and Grant PELECT/6315298, in part by the Ministry of Education Malaysia Fundamental Research Grant Scheme (FRGS) under Grant 203.PELECT.6071430, and in part by the Hubert Curien Partnership France-Malaysia Hibiscus (PHC-Hibiscus) Grant 203.PKOMP.6782005.

ABSTRACT Over the past few years, mobile robots are widely used in various industries because they can navigate in dynamic environments and carry out everyday tasks efficiently. Path planning undoubtedly plays an important role in mobile robot navigation, thus becoming one of the most researched topics in the field of robotics. The metaheuristic algorithms have been extensively studied in recent years to solve the path planning problems the same way as the optimization problems were solved, or in other words, plan the optimum path for the robot to navigate from the starting point to the destination. In this research, a mobile robot is launched to find its path from the starting point to the destination in three different simulated environments using a proposed hybrid metaheuristic algorithm between Particle Swarm Optimization and Fringe Search Algorithm, named PSOFS, and its simultaneous localization and mapping (SLAM) capability. During runtime, the path is optimized by considering the path length. The performance of PSOFS for local path planning is compared against two existing algorithms by evaluating the path smoothness as well as robot safety. The results showed that PSOFS has successfully generated shorter, smoother, and safer paths than the algorithms for mobile robot navigation in unknown indoor environments.

INDEX TERMS Local path planning, mobile robot navigation, unknown indoor environments, particle swarm optimization, fringe search algorithm, hybrid metaheuristics.

I. INTRODUCTION

Over the last few years, robotics and automation have gradually developed in parallel to carry out tasks in almost every area better and eventually replace manual labor, such as autonomous freight transport. Car manufacturers, as well as computer and technology companies, play an active role in the development of autonomous navigation.

Based on a systematic literature review by [1], autonomous navigation for indoor mobile robots, especially those that function based on vision, did not receive as much attention as autonomous navigation of road vehicles, specifically cars and trucks. The authors stated that cars and trucks have similar locomotion mechanisms, which is four wheels, and move in the same structured environment, which is roads. However, indoor mobile robots such as aerial and ground robots must consider different locomotion mechanisms and unstructured environments. For example, aerial robots fly in the sky with their wings and ground robots move on the ground with their wheels. These generated more complicated locomotion mechanisms with unpredicted environments that needed more in-depth study. Besides, autonomous robots that are built for a single purpose can be quite expensive. The robot vacuum cleaners map the environment, sweep the whole area, empty their compartments, and recharge when needed. The automated guided vehicles (AGV) are guided by markings and landmarks placed on the environment to transport materials and products in the warehouses. In order

The associate editor coordinating the review of this manuscript and approving it for publication was Xiangtao Li.

to accommodate robots in a specific environment or to enable robots to perform specific tasks, researchers often have to spend a lot of time and money building and configuring robot systems from scratch. Reconfiguration and further modifications may be required if there are any environmental changes. Regardless of the tolerance for conditions that new models can offer for those robots, robots still require a specialized configuration for their application.

Generally, there are four steps towards mobile robot navigation: perception, localization, cognition and path planning, and motion control. The first step is perception; the robot collects the environmental information from its sensors and followed by localization; reminds the robot of its position and environmental orientation for each iteration. Next is cognition and path planning; the robot plans its path to the destination. The last step is for the robot to track its path by controlling the motion. Besides, there are two types of navigation: global and local. Global navigation deals with completely known environments, while local navigation deals with partly known or unknown environments. Therefore, safe path planning is required by detecting and avoiding obstacles, then navigating from the starting point to the destination while meeting certain requirements such as distance and smoothness of the traveled path.

The existing path planning methods can be categorized as follows: classical methods and heuristic methods [26]. Classical methods such as cell decomposition method, potential field method, and roadmap method are often used in global navigation since the publication of [2]. However, these algorithms have limited intelligence. Evolutionary algorithms such as Artificial Neural Network (ANN) [3]–[5], Genetic Algorithm (GA) [6]–[8] and Particle Swarm Optimization (PSO) [9]–[11] algorithm are widely used in local navigation in recent years because they are more intelligent to control and execute a plan autonomously.

Reference [27] is an example of previous work that focuses on a heuristic method to solve a path planning problem where they proposed a path planning algorithm based on multi-objective Particle Swarm Optimization known as MOPSO for multi-robots. The path is constructed based on the sensors and experiences shared among the robots to achieve the best path, which is short, safe, and smooth. However, path planning is not limited to mobile robot application only, for instance, [28] utilized modified classical algorithm named as Improved Rapidly-exploring Random Tree (RRT) algorithm to generate a path for an industrial robot in a complex environment. As Deep Reinforcement Learning is merging, people are starting to utilize this algorithm in path planning. Reference [14] combining FastSLAM with Deep Reinforcement Learning to construct a path in a complex unknown environment for an application in a mobile robot. They are also using the Robot Operating System (ROS) as their platform for data gathering.

Ultimately, the motivation for this research is to enable collision-free navigation of a mobile robot in unknown indoor environments. As mentioned earlier, autonomous navigation for indoor mobile robots did not receive as much attention as autonomous navigation of road vehicles, and autonomous robots that are built for a single purpose can be quite expensive. The future robot must also be able to locate itself, find its way to the destination and avoid obstacle simultaneously after leaving the starting point in any indoor environment so that it can be used for various applications.

This research is done in three simulated indoor environments with a virtual robot, which is TurtleBot3 Burger. The simulated environments with the virtual robot are created using the Robot Operating System (ROS) and Gazebo.

Furthermore, the research problem is the problem of planning the optimum path for a mobile robot to navigate in unknown indoor environments. An optimum path refers to a path that is shorter, smoother in terms of the number of turns the robot needs to make to the destination, and safer in terms of the robot's ability to avoid obstacles when navigating in an unknown environment. Even though there are many algorithms proposed for solving local path planning problem in recent years, researchers are constantly enhancing the algorithms and proposing their best algorithm to solve the problem.

In this paper, a hybrid method between Particle Swarm Optimization and Fringe Search is utilized to create a collision-free path between the starting point and the endpoint is presented. This paper is organized as follows: Section I introduces the subject related to path planning and recent methods developed for this subject matter; section II discusses the comparison methods for our proposed method which are PSOA and PSOD; section III discusses the proposed techniques; section IV describes the experimental setup; section V reports the results obtained after running the proposed method on 3 different layouts measuring their path length, safety, and smoothness. Finally, section VI gives an overall conclusion for this research.

II. STATE OF THE ART: PATH PLANNING

The existing path planning methods are divided into two main categories: classical methods and heuristic methods [26]. The classical methods include the Cell Decomposition approach (CD), the Probabilistic Road Map approach (PRM), the Potential Field approach (PF), and the Rapidly exploring Random Tree approach (RRT). The heuristic methods include Dijkstra's algorithm, A* search algorithm, and D* algorithm. The evolutionary algorithms include Artificial Neural Network (ANN), Genetic Algorithm (GA), and Particle Swarm Optimization (PSO). All these approaches and algorithms are commonly utilized for mobile robot path planning. Since the last two decades, many heuristic methods such as Dijkstra's algorithm, A* search algorithm, and D* algorithm were introduced and used for new robotic systems such as humanoid robots and space rovers. Generally, these algorithms are informed search strategy algorithms which are used for graph search, and they can find a path faster than the classical methods because of their completeness, efficiency, and optimality.

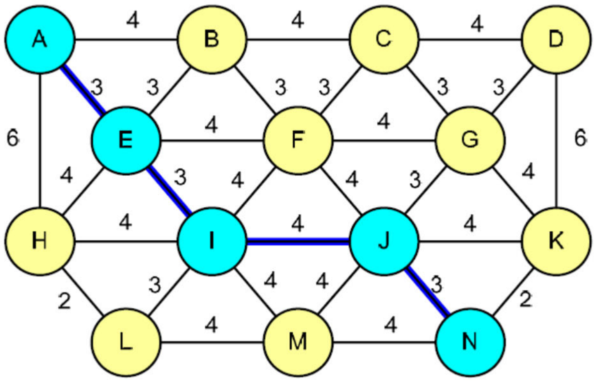

FIGURE 1. A* Search Algorithm [22].

The A* search algorithm was presented by [18] in the year 1968. It is one of the variants of Dijkstra's algorithm and commonly used to find the shortest path in a weighted directed graph (see Fig. 1). Unlike Dijkstra's algorithm which only can explore one possible path at a time, the algorithm can explore multiple possible paths at a time from the origin vertex, until one of the paths ends at the destination vertex [18]. The efficiency of the algorithm in pathfinding is higher than Dijkstra's algorithm. However, the algorithm is mainly used for global search in static environments. Therefore, a few variants of the algorithm, such as D* algorithm were developed.

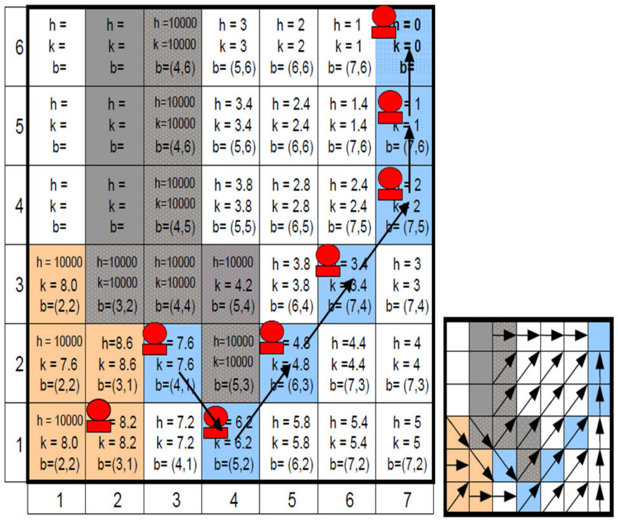

FIGURE 2. D* Search Algorithm [22].

The D* algorithm was presented by [19] in the year 1994. It is one of the variants of A* search algorithm and has a few variants such as field D* algorithm and Theta* algorithm (see Fig. 2.). As the mobile robot path planning problem is gradually aware of dynamic environmental information, Dijkstra's algorithm and A* search algorithm which find paths in static environments are not so useful anymore. Therefore, the algorithm was developed to find paths in dynamic environments [19].

The Particle Swarm Optimization (PSO) algorithm was presented by [13] in the year 1995. This algorithm begins with a random solution and searches for the optimal solution through numerous iterations (see Fig. 4). The solution quality will be evaluated based on its fitness value over time, and eventually, the global optimal will be discovered by comparing the currently searched optimal value. There are several advantages of PSO such as fast convergence, high precision, and most importantly, simple implementation [13]. An improved PSO for a mobile robot's path planning was proposed by [20]. The method is shown effective based on the simulation results. A global path planning method based on multi-objective PSO was proposed by [9]. The simulation has demonstrated the algorithm effectiveness. Reference [10] proposed some key technologies based on PSO for path planning in a radiation environment. The experiment has demonstrated the effectiveness and probability of the method. A predetermined waypoints method based on PSO was proposed by [11]. The approach has shown favorable results. Even though PSO is fast and efficient, it often suffers from local optimum [21]. PSOA, which is a hybrid algorithm of particle swarm optimization (PSO) algorithm and A* search algorithm was presented by [15] in the year 2018 for multiple robots to perform target searching in unknown environments.

PSOA starts with initializing a swarm of particles based on the robot's initial position. Next, the parameters for the PSO algorithm will be initialized, and the first position will be calculated randomly. Then, three phases based on the diversity measurement (Div) for attraction and repulsive of PSO with local search, namely attraction, repulsion and combination of attraction and repulsion. For each iteration, Div will be calculated to set the personal best for the particle. The fitness function refers to the detection of obstacles using the robot's camera; however, in this research, the fitness function is the detection of obstacles using the robot's sensor because TurtleBot3 Burger does not have a camera. Therefore, if the robot detects an obstacle during its navigation, the A* search algorithm will be used to continue planning the path for the robot. Besides, d_{high} and d_{low} refers to the upper and lower thresholds of Div, respectively. If Div is more than d_{high}, the robot will move towards the global best position – it is the attraction phase. If Div is in between d_{low} and d_{high}, the robot will move towards its own best position – it is the combination of attraction and repulsion phase. Similarly, if Div is less than d_{low}, the robot will move towards its own best position – it is the repulsion phase. The new position will be updated accordingly in each iteration after one of the above steps is executed. The pseudo-code for PSOA can be referred to Algorithm 1 (Fig. 3).

PSOD, which is a hybrid algorithm of particle swarm optimization (PSO) algorithm and D* algorithm was presented by [25] in the year 2018 for mobile robot path planning in dynamic environments. PSOD starts with robot finding paths in the environment using the D* algorithm after it is launched.

```
Algorithm 1 PSOA
1:   create and initialize a swarm of particles based on the robot's initial position
2:   initialize parameters for PSO algorithm
3:   calculate the first position randomly
4:
5:   while stopping criteria is not satisfied do
6:       calculate diversity (Div)
7:
8:       if FitnessFunction(robot, position) > θ then
9:           continue planning the path for the robot using A* search algorithm
10:      else if Div > $d_{high}$ then
11:          $v_{id} = w \times v_{id} + c_1 \times r_1 \times (pbest_{id} - x_{id}) + c_2 \times r_2 \times (gbest_{id} - x_{id})$
12:      else if $d_{low}$ < Div < $d_{high}$ then
13:          $v_{id} = w \times v_{id} + c_1 \times r_1 \times (pbest_{id} - x_{id}) - c_2 \times r_2 \times (gbest_{id} - x_{id})$
14:      else if Div < $d_{low}$ then
15:          $v_{id} = -1 \times w \times v_{id} - c_1 \times r_1 \times (pbest_{id} - x_{id}) - c_2 \times r_2 \times (gbest_{id} - x_{id})$
16:      end if
17:      update new position accordingly
18:  end while
```

FIGURE 3. Pseudo code for PSOA.

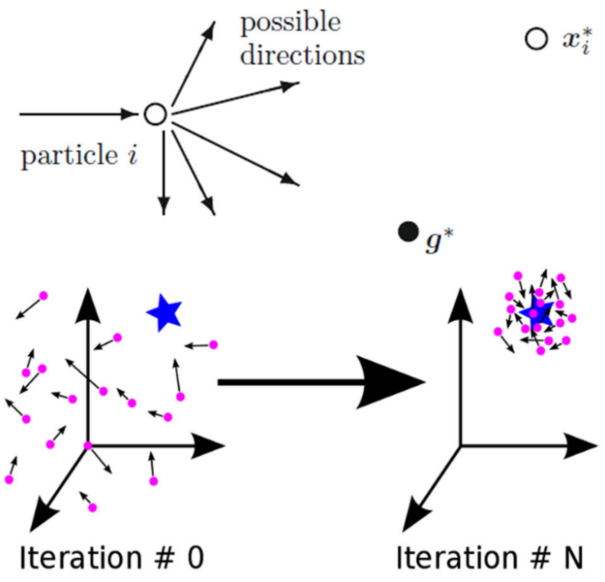

FIGURE 4. Particle Swarm Optimization [23], [24].

Once a path is obtained, the PSO algorithm is executed by first defining the number of via points and parameters for the algorithm, followed by randomly selecting a via point from the obtained path. Upon selection, the local and global values will be updated, and the via point will be interpolated by the spline equation. After that, the length of the respective path will be calculated.

At the end of the iteration, the algorithm will end with the best path with the best cost. Otherwise, it will return to the steps of defining the number of via points and parameters for the algorithm, randomly selecting a via point from the path obtained, updating local and global values, interpolating via point using spline equation, calculating the length of the respective path. The pseudo-code for PSOD can be referred to Algorithm 2 (Fig. 5).

```
Algorithm 2 PSOD
1:   pathfinding using D* algorithm
2:   define the number of via points and
3:   parameters for PSO algorithm
4:
5:   while stopping criteria is not satisfied do
6:       select a via point from the obtained
7:       path randomly
8:
9:       update the local and global values
10:      using PSO algorithm
11:
12:      interpolate the via point by spline
13:      equation
14:
15:      calculate path length
16:  end while
17:
18:  return the shortest path
```

FIGURE 5. Pseudo code for PSOD.

Lastly, a solution, which consists of values for path length, smoothness, and safety, is created at the end of each run for all algorithm. After the algorithm is run for 20 times, there will be 20 solutions created, and the best solution will be decided from the Pareto frontier, which is made up of some of these solutions. After the best solution is decided for each proposed and existing algorithm, there will be three best solutions to be compared and finalized.

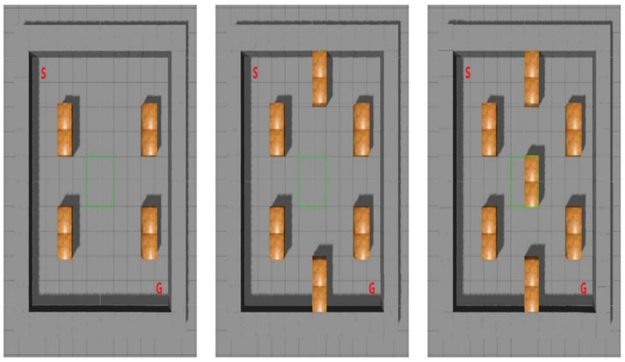

FIGURE 6. The Easy, Medium, and Hard level worlds in Gazebo where S and G are the start and goal points.

III. RESEARCH METHOD

The proposed methodology for this research is combination or hybridization of Particle Swarm Optimization (PSO) with Fringe Search Algorithm (FS) As shown in Algorithm 3 (Fig. 7), the robot's starting point, the list of static obstacles in the respective world, and the robot's destination are initialized. In this research, the starting point and destination are top left and bottom right coordinates, respectively in all worlds, while the list of obstacles is extracted from the respective_world file. Next, an empty list named path_points is created, and the robot's starting point is then added to the list as the first point of the path. After that, the robot's current position is initialized to the robot's starting point and used as a particle in the implementation of the Particle Swarm Optimization algorithm (PSO) and the parameter setting is set as in Table 1. The PSO is utilized using Equation (1) – (4):

$$v(t+1) = wv(t) + cognitive + social \tag{1}$$

$$cognitive = c_1 r_1 (p_{best} - x(t)) \tag{2}$$

$$social = c_2 r_2 (g_{best} - x(t) \tag{3}$$

$$x(t+1) = x(t) + v(t+1) \tag{4}$$

v is the velocity, w is the inertia weight, c_1 and c_2 are coefficient for cognitive and social respectively, r_1 and r_2 are a random number between 0 to 1 and x is the position of the particle. P_{best} is the best position found by the particle so far while G_{best} is the best position found by the swarm so far. During the iterations, the points generated by the PSO algorithm are evaluated by solving an objective function. The objective function is about minimizing the distance from the point to the destination, in which there is only one of the optimization criteria mentioned earlier, i.e. path length, is used for the objective function. Whichever point in the swarm can move the furthest from the current point, and it will be used as the next turning point for the robot to make a turn and continue moving towards the destination. If the point generated by the PSO algorithm is less than one meter away from the edge of any obstacles, which is not safe for the robot to move towards, the Fringe Search algorithm (FS) will be used to find a safer point for the robot's navigation.

TABLE 1. Best parameters for PSOFS, PSOA, and PSOD which are found through trial and error.

Parameter	Value
Population size	15
Number of iterations	30
Inertia weight	0.5
Cognitive coefficient	1
Social coefficient	2

FS utilized is the single-agent notation based on equation 5.

$$f = g + h \tag{5}$$

g is the search path costs from the origin node to the current node and h is the heuristic estimation of the path cost from the current node to the target node. Once the robot reaches the destination, the robot's destination will be added to path_points as the last point of the path. Finally, a solution with values for path length, smoothness, and safety is created and returned at the end of each run of the proposed algorithm along with the path.

IV. EXPERIMENTAL SETUP

The experimental setup consists of three steps: environment modelling, pathfinding and optimization, and obtain the optimum path. Environment modelling is to model the actual environment where mobile robots perform their tasks based on known map information to understand environmental variables such as obstacles better and reduce excessive planning and most computations.

This step is usually taken to solve global path planning problems, and the methods which are commonly used for modelling a robot's working environment are cell decomposition approach and road map approach.

In this research, three indoor environments for mobile robot navigation are modelled; however, the environment maps remained unknown to the robot to solve the local path planning problem. Consequently, the robot had to explore the environments with its simultaneous localization and mapping (SLAM) capability and find its path to the known destination. Furthermore, inspired by video game difficulty levels, the simulated indoor environments are named Easy level world, Medium level world, and Hard level world, as shown in Fig. 6. A simulated environment is commonly known as a world in Gazebo. Each of the above-mentioned worlds is created from an empty world using 10×10 square cells and has the same start and goal points which are $(-4.5, -4.5)$ and $(4.5, 4.5)$ respectively, though contains a different number of wood color square boxes as the obstacles. The Easy, Medium, and Hard level worlds contain 8, 12, and 14 boxes, respectively. The worlds are shown along with the start and goal points in Fig. 6.

The second step of the proposed methodology – pathfinding and optimization – is about finding paths that fulfil optimization criteria. According to [12], there are three

Algorithm 3 PSOFS
```
1:  initialize robot_starting_point, world_obstacles, robot_destination
2:  create an empty list named path_points
3:  add robot_starting_point to path_points
4:  robot_current_position = robot_starting_point
5:
6:  while robot_current_position != robot_destination do
7:          turning_point = PSO(robot_current_position)
8:
9:          if the possible collision area is detected then
10:                 set robot_current_position to the position where it almost collides with obstacles
11:                 turning_point = FringeSearch(robot_current_position)
12:         end if
13:
14:         if the robot reaches the turning_point then
15:                 add turning_point to path_points
16:                 robot_current_position = turning_point
17:         end if
18: end while
19:
20: add robot_destination to path_points
21: create solution using path_points and world_obstacles
22: return solution, along with path_points
```

FIGURE 7. Pseudo code for PSOFS.

frequently used optimization criteria: path length, smoothness, and safety. A hybrid algorithm of particle swarm optimization (PSO) [13] and fringe search [14] algorithms are proposed to implement this step of the methodology. The robot's starting point, the list of obstacles in the respective world, and the robot's destination are initialized. In this research, the starting point and destination are top left and bottom right coordinates, respectively in all worlds, while the list of obstacles is extracted from the respective world description file. Next, an empty list named path_points is created, and the robot's starting point is then added to the list as the first point of the path. After that, the robot's current position is initialized to the robot's starting point and used as a particle in the implementation of the PSO algorithm (see Table 1 for parameter setting). During the iterations, the points generated by the PSO algorithm are evaluated by solving an objective function.

The objective function is about minimizing the distance from the point to the destination, in which there is only one of the optimization criteria mentioned earlier, i.e. path length, is used for the objective function. Whichever point in the swarm can move the furthest from the current point, it will be used as the next turning point for the robot to make a turn and continue moving towards the destination. If the point generated by the PSO algorithm is not safe for the robot to move towards, the fringe search algorithm will be used to find a safer point for the robot's navigation. Once the robot reaches the destination, the robot's destination will be added to path_points as the last point of the path. Finally, a solution with values for path length, smoothness, and safety is created and returned at the end of each run of the proposed algorithm.

In continuation of the previous paragraph, PSOFS is proposed to solve the local path planning problem for mobile robot navigation in unknown indoor environments. In order to evaluate the performance of the proposed algorithm, two existing algorithms named PSOA and PSOD are implemented for comparison (see Table 1 for parameter setting). PSOA, which is a hybrid algorithm of PSO and A* search algorithms, was presented by [15] for multiple robots to perform target searching in unknown environments. On the other hand, PSOD, which is another hybrid algorithm of PSO and D* algorithms, was presented by [16] for mobile robot path planning in dynamic environments.

The reason to combine between PSO and FS which create the proposed method, PSOFS, is based on the performance of FS against A* search algorithm (a popular and highly optimized pathfinding algorithm in grid pathfinding approach) [14], [29]. PSOD is considered as part of comparison methods for PSOFS because it shows quite good promising results in a recent study. It is also quite decent in finding path within dynamic environments [30]. The source code and instructions on how to run the code is available here [31].

The third step of the proposed methodology – obtain the optimum path – is about obtaining the optimum path for a mobile robot to navigate smoothly from the starting point to the destination in its working environment. All obtained paths are evaluated based on the above-mentioned optimization criteria, which are path length, smoothness, and safety, to identify the optimum path. The criteria are described as follows.

Path length refers to the total distance travelled by the robot from the starting point to the destination. Mathematically, this criterion is about finding distance between two points, and the distance is calculated using the Euclidean distance formula, as shown in Equation (6). The steps to compute a path length are described as follows based on an example. Given an environment which is made up of five obstacles OB1, OB2, OB3, OB4, and OB5, and a path that contains five points A, B, C, D, and E, there are four path segments, named AB, BC, CD, and DE. The first step is to calculate the lengths of these path segments, such as the distance from point A to point B and the distance from point B to point C. Secondly, sum all the lengths. Lastly, obtain the total path length. The shorter the path, the less time the robot takes to navigate from starting point to a destination point.

$$d(\boldsymbol{p},\boldsymbol{q}) = \sqrt{(q_1-p_1)^2+(q_2-p_2)^2} \tag{6}$$

Path smoothness refers to the number of turns with turning angles that the robot needs to make while navigating from the starting point to the destination. Mathematically, this criterion is about finding the angle between two lines using the formula shown in Equation (7). Also, before finding the angle between any two lines, the slopes of the lines are calculated using the formula shown in Equation (8). The steps to compute a path smoothness are described as follows based on the above-mentioned example. The first step is to calculate the angles between these path segments, such as the angle between segments AB and BC and the angle between segments BC and CD. Secondly, average all the angles using the formula shown in Equation (9). Lastly, obtain the average angle for evaluating the path smoothness. A smaller average angle means that the path is rather smooth for the robot to navigate from the starting point to the destination without making too many turns, especially sharp turns.

$$\tan\theta = \left|\frac{m_2-m_1}{1+(m_2*m_1)}\right| \tag{7}$$

$$\mathrm{m} = \frac{y_2-y_1}{x_2-x_1} \tag{8}$$

$$\bar{\alpha} = a\,tan2\left(\frac{1}{n}\sum_{j=1}^{n}\sin\alpha_j, \frac{1}{n}\sum_{j=1}^{n}\cos\alpha_j\right) \tag{9}$$

Path safety refers to the space between a path and the obstacles in a robot's working environment. Mathematically, this criterion is about finding the distance from a point to a line. The steps to compute a path safety are described as follows based on the above-mentioned example. The first step is to calculate the midpoint for each path segment using the formula shown in Equation (10), such as midpoint Z for segment BC and midpoint Y for segment CD. Secondly, find the nearest obstacle to each midpoint using the k-dimensional tree data structure, which was presented by [17].

For example, among all obstacles, the obstacle OB3 is the nearest obstacle to midpoint Y. Thirdly, calculate the shortest distance from each path segment to their respective nearest obstacles using the formula shown in Equation (11), such as the shortest distance from segment CD to obstacle OB3. Lastly, identify the shortest distance from the path to any obstacles in the environment. A distance between the path created and the obstacle needs to be as far as possible to make it safe for the robot to navigate through the path without any collision against any obstacle within the environment.

$$(x_m, y_m) = \left(\frac{x_1+x_2}{2}, \frac{y_1+y_2}{2}\right) \tag{10}$$

$$\mathrm{d} = \frac{|\alpha\,(x_0)+b\,(y_0)+c|}{\sqrt{\alpha^2+b^2}} \tag{11}$$

V. RESULTS AND ANALYSIS

In this chapter, the results of the experiments in all worlds using the proposed algorithm, which is PSOFS, are compared with the results of the experiment in all worlds using another two existing algorithms, which are PSOA and PSOD. Each algorithm is run for 20 times to obtain the optimum path for the robot to navigate in all worlds. After any of the algorithms are run for 20 times, there will be 20 solutions created and to be evaluated based on path length, smoothness, and safety. Next, the best solution is decided from the Pareto frontier, which is made up of some of these solutions. The best solution is a solution which consists of the largest value for path safety. After the best solution is decided for each proposed and existing algorithm, there will be three best solutions to be compared and finalized for the robot's navigation in the respective world. Besides, the statistical analysis on PSOFS against PSOA and PSOD used the same sets of experiment results to run paired t-tests in Excel. A paired t-test is usually performed to determine whether the difference between the means of the two sets of scores is statistically significant. Before running paired t-tests on any two hybrids algorithms, the significance level is set to 0.05 by default and the hypothesized mean difference is set to zero, which is the null hypothesis value and represents no effect. In this case, a mean difference of zero represents no difference between the two algorithms, which is no effect.

The results of the local path planning experiments in Easy, Medium, and Hard level worlds using all algorithms will be discussed next. The results of each experiment will be organized into two parts. The first part of the results will contain a table containing the best paths and solutions obtained for the respective world using all algorithms, while another part of the results will contain the visualization of the best paths to find the best path for the robot to navigate in the respective world. Next, the results of the paired t-tests on PSOFS against PSOA in Easy, Medium, and Hard level worlds is discussed and followed by the results of the paired t-tests on PSOFS against PSOD in Easy, Medium, and Hard level worlds.

Based on Table 2 and Fig. 8, PSOFS returned the optimum path for the Easy level world, where the path length,

TABLE 2. Results of local path planning experiments in easy level world using PSOFS, PSOA, and PSOD.

Algorithm	Path	Solution
PSOFS	[(-4.5, -4.5), (-1.2, -3.2), (2.2, 2.7), (3.7, 4.2), (3.6, 4.3), (4.3, 4.4), (4.5, 4.4), (4.5, 4.5)]	[**13.63**, **0.01**, **1.41**]
PSOA	[(-4.5, -4.5), (-1.8, -3.3), (-0.0, 3.5), (3.3, 3.7), (3.8, 4.0), (4.0, 4.3), (4.3, 4.6), (4.4, 4.5), (4.5, 4.5)]	[14.9, 0.05, 0.86]
PSOD	[(-4.5, -4.5), (-1.2, -3.3), (1.6, 4.0), (2.3, 4.4), (2.6, 4.5), (4.5, 4.5), (4.5, 4.5)]	[14.35, 6.19, 0.58]

TABLE 3. Results of local path planning experiments in medium level world using PSOFS, PSOA, and PSOD.

Algorithm	Path	Solution
PSOFS	[(-4.5, -4.5), (1.2, -1.5), (2.1, 4.3), (2.8, 4.3), (2.9, 4.3), (4.0, 4.5), (4.4, 4.4), (4.5, 4.5)]	[14.78, 0.11, 0.91]
PSOA	[(-4.5, -4.5), (-3.5, -0.5), (1.8, 3.0), (3.8, 4.2), (4.2, 4.2), (4.3, 4.2), (4.4, 4.3), (4.5, 4.5)]	[13.67, 6.24, **1.52**]
PSOD	[(-4.5, -4.5), (0.2, -2.0), (3.2, 3.9), (3.3, 4.2), (3.7, 4.3), (4.2, 4.3), (4.4, 4.4), (4.5, 4.5)]	[**13.54**, **0.08**, 0.85]

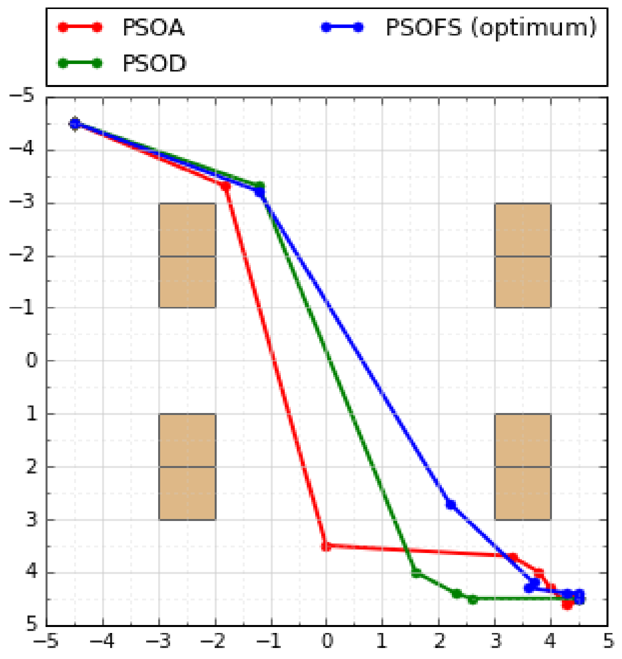

FIGURE 8. Visualization of the Best Paths for Easy Level World using PSOFS, PSOA, and PSOD.

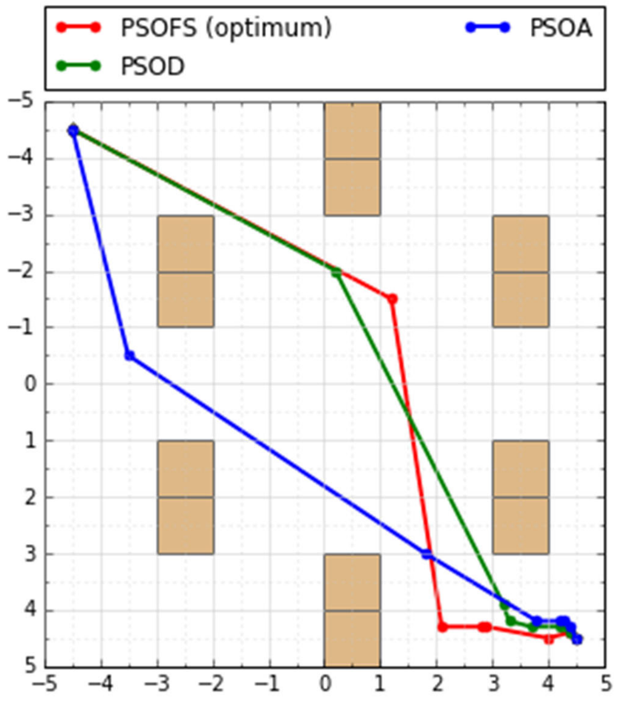

FIGURE 9. Visualization of the Best Paths for Medium Level World using PSOFS, PSOA, and PSOD.

smoothness, and safety are 13.63, 0.01, and 1.41, respectively. The path length of 13.63 means that the total distance travelled of the robot from the starting point to the destination is 13.63 meter. The path smoothness of 0.01 means that the path is rather smooth for the robot to navigate from the starting point to the destination without making too many turns, especially sharp turns. The path safety of 1.41 means that the shortest distance from the path to any obstacles in the respective world is 1.41 meter.

Based on Table 3 and Fig. 9, PSOFS returned the optimum path for Medium level world, where the path length, smoothness, and safety are 14.78, 0.11, and 0.91, respectively. The path length of 14.78 means that the total distance travelled of the robot from the starting point to the destination is 14.78 meter. The path smoothness of 0.11 means that the path is rather smooth for the robot to navigate from the starting point to the destination without making too many turns, especially sharp turns. The path safety of 0.91 means that the shortest distance from the path to any obstacles in the respective world is 0.91 meter which makes it safer compared to PSOD which is 0.85 meter although the path length of PSOD is slightly shorter compared to PSOFS. The path generated by PSOA is not considered as an optimal path because the smoothness value given is too high, which is 6.24.

Based on Table 4 and Fig. 10, PSOFS returned the optimum path for the Hard level world, where the path length, smoothness, and safety are 14.15, 0.53, and 5.1, respectively. The path length of 14.15 means that the total distance travelled of the robot from the starting point to the destination is 14.15 meter. The path smoothness of 0.53 means that the path is rather smooth for the robot to navigate from the starting point to the destination without making too many turns, especially sharp turns. The path safety of 5.1 means

TABLE 4. Results of local path planning experiments in hard level world using PSOFS, PSOA, and PSOD.

Algorithm	Path	Solution
PSOFS	[(-4.5, -4.5), (-0.4, -2.7), (4.1, 0.2), (4.5, 4.5)]	[**14.15**, 0.53, **5.1**]
PSOA	[(-4.5, -4.5), (-2.4, -4.2), (3.7, -0.5), (4.3, 0.5), (4.3, 1.9), (4.3, 2.2), (4.4, 3.7), (4.4, 4.2), (4.4, 4.4), (4.4, 4.5), (4.5, 4.5)]	[14.53, **0.0**, 0.2]
PSOD	[(-4.5, -4.5), (1.8, -0.9), (4.2, 0.0), (4.2, 2.6), (4.2, 3.0), (4.3, 3.5), (4.4, 3.6), (4.4, 3.7), (4.4, 4.3), (4.5, 4.4), (4.5, 4.5)]	[14.44, 0.27, 1.37]

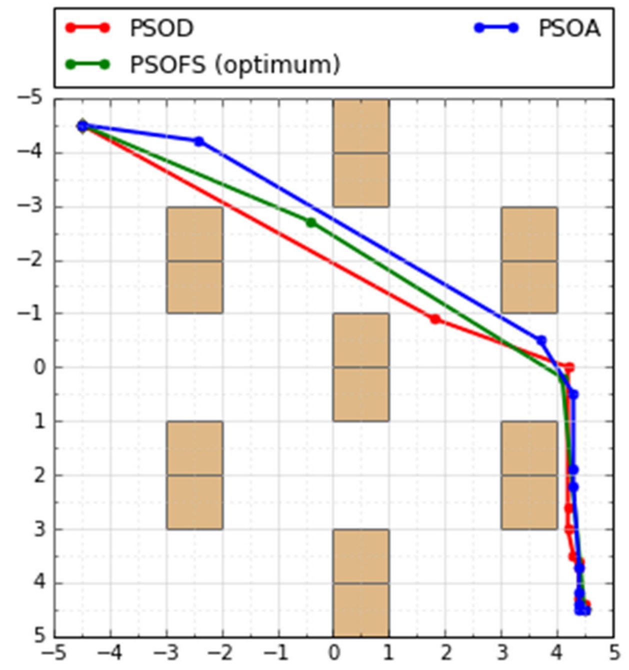

FIGURE 10. Visualization of the Best Paths for Hard Level World using PSOFS, PSOA, and PSOD.

that the shortest distance from the path to any obstacles in the respective world is 5.1 meter.

The p-values of the unpaired t-tests on PSOFS against PSOA in Easy, Medium, and Hard level worlds are 0.966051195, 0.001083842, and 0.010912967, respectively. Most of the p-values are less than the standard significance level of 0.05. Thus, the mean difference is statistically significant, and the null hypothesis can be rejected. The p-values of the unpaired t-tests on PSOFS against PSOD in Easy, Medium, and Hard level worlds are 1.52503E-11, 7.39472E-11, and 0.702707325, respectively. Most of the p-values are less than the standard significance level of 0.05, thus the mean difference is statistically significant, and the null hypothesis can be rejected.

In order to plan the most optimal path for a mobile robot to maneuver in a completely new environment, a metaheuristic algorithm with local search ability is required. To improve the PSO algorithm that does not have local searchability. Generally, there are four ways to improve the algorithm. First, the increase in population size will lead to faster and more precise convergence. Second, the balance between exploration and exploitation will unveil the most accurate solution that is close to global optima. The third approach would be the sub-swarm approach which will increase the algorithm efficiency. Finally, dynamic velocity adjustment will direct particles in different directions towards the global optimum. Ultimately, one of the possible solutions for this research would be implementing one of the approaches stated above. Another possible solution would be combining the PSO algorithm with a local search algorithm such as hill climbing and simulated annealing.

VI. CONCLUSION

This paper is aimed to solve the problem of planning the optimum path for a mobile robot to navigate in unknown indoor environments. Various existing path planning methods for mobile robot navigation are studied. Heuristic and metaheuristic algorithms are selected to develop the hybrid metaheuristic algorithm of this research. Next, the indoor environments for mobile robot navigation are modeled in the next stage of the experiment without the robot has a prior knowledge on the environment maps. Hence, the robot needs to explore the environment and solves the local path planning problem based on the proposed algorithms. The robot departed from the starting point, explore the environments with its simultaneous localization and mapping (SLAM) capability, and find its path to the known destination using the proposed algorithm and two existing algorithms. All the paths created by these algorithms are evaluated in terms of path length, smoothness, and safety during runtime. Finally, the optimum path obtained from the paths created by the proposed algorithm is compared with the respective optimum paths obtained from the paths created by another two existing algorithms.

In continuation of the previous paragraph, the results of the experiment in all worlds using the proposed algorithm, which is PSOFS, are compared with the results of the experiment in all worlds using another two existing algorithms, which are PSOA and PSOD. Based on the experiment results and the results of the statistical analysis on PSOFS against PSOA and PSOD, there are two comparison results for each level world (Easy, Medium, and Hard) with the consideration factors of Path Length, Smoothness, and Safety. Hence the total tests for PSOFS in statistical analysis are six tests and based on the observation the paths returned by PSOFS are indeed better optimized than the paths returned by PSOA and PSOD in 4 tests out of 6 tests which are 66.67% better performance. Hence, it is noteworthy that PSOFS can perform better compared to PSOA and PSOD in general.

Since the proposed method PSOFS is performing well compared to PSOA and PSOD then perhaps it can be further investigate the performance of PSOFS against more recent improved hybrid method between heuristic and metaheuristic algorithms for path planning application such as Genetic Algorithm (GA), Simulated Annealing (GA), Ant Colony Optimization (ACO), Artificial Neural Network (ANN), etc.

This research has good potential to be applied in autonomous vehicles such as Smart Car, Unmanned Aerial Vehicle (UAV), Unmanned Ground Vehicle (UGV), and Autonomous Underwater Vehicle (AUV). It also can be applied in navigation systems such as Google Maps and Waze.

ACKNOWLEDGEMENT

This research is a collaboration between School of Computer Sciences, USM with School of Electrical and Electronic, USM and affiliated with Robotics, Computer Vision & Image Processing (RCVIP) Research Group Lab at School of Computer Sciences, Universiti Sains Malaysia.

REFERENCES

[1] Y. D. V. Yasuda, L. E. G. Martins, and F. A. M. Cappabianco, ''Autonomous visual navigation for mobile robots: A systematic literature review,'' *ACM Comput. Surv.*, vol. 53, no. 1, pp. 1–34, May 2020.

[2] T. Lozano-Perez, ''Spatial planning: A configuration space approach,'' in *Autonomous Robot Vehicles*. New York, NY, USA: Springer, 1990, pp. 259–271.

[3] C. Li, J. Zhang, and Y. Li, ''Application of artificial neural network based on Q-learning for mobile robot path planning,'' in *Proc. IEEE Int. Conf. Inf. Acquisition*, Aug. 2006, pp. 978–982.

[4] S. Raza and S. Haider, ''Path planning in robocup soccer simulation 3D using evolutionary artificial neural network,'' in *Proc. Int. Conf. Swarm Intell.*, 2013, pp. 342–350.

[5] A. F. Contreras-González, J. I. Hernández-Vega, C. Hernández-Santos, and D. G. Palomares-Gorham, ''A method to verify a path planning by a back-propagation Articial neural network,'' in *Proc. LANMR*, 2016, pp. 98–105.

[6] C.-C. Tsai, H.-C. Huang, and C.-K. Chan, ''Parallel elite genetic algorithm and its application to global path planning for autonomous robot navigation,'' *IEEE Trans. Ind. Electron.*, vol. 58, no. 10, pp. 4813–4821, Oct. 2011.

[7] A. Tuncer and M. Yildirim, ''Dynamic path planning of mobile robots with improved genetic algorithm,'' *Comput. Electr. Eng.*, vol. 38, no. 6, pp. 1564–1572, Nov. 2012.

[8] H. Qu, K. Xing, and T. Alexander, ''An improved genetic algorithm with co-evolutionary strategy for global path planning of multiple mobile robots,'' *Neurocomputing*, vol. 120, pp. 509–517, Nov. 2013.

[9] D.-W. Gong, J.-H. Zhang, and Y. Zhang, ''Multi-objective particle swarm optimization for robot path planning in environment with danger sources,'' *J. Comput.*, vol. 6, no. 8, pp. 1554–1561, Aug. 2011.

[10] F. Liu, S. Liang, and D. X. Xian, ''Optimal path planning for mobile robot using tailored genetic algorithm,'' *Telkomnika Indonesian J. Electr. Eng.*, vol. 12, no. 1, pp. 1–9, Jan. 2014.

[11] T. S. T. Yusof, S. F. Toha, and H. M. Yusof, ''Path planning for visually impaired people in an unfamiliar environment using particle swarm optimization,'' *Procedia Comput. Sci.*, vol. 76, pp. 80–86, Jan. 2015.

[12] H.-Y. Zhang, W.-M. Lin, and A.-X. Chen, ''Path planning for the mobile robot: A review,'' *Symmetry*, vol. 10, no. 10, p. 450, Oct. 2018.

[13] J. Kennedy and R. Eberhart, ''Particle swarm optimization,'' in *Proc. IEEE ICNN*, vol. 4, Nov./Dec. 1995, pp. 1942–1948.

[14] Y. Björnsson, M. Enzenberger, R. C. Holte, and J. Schaeffer, ''Fringe search: Beating A* at pathfinding on game maps,'' in *Proc. CIG*, 2015, pp. 125–132.

[15] B. Nakisa, M. N. Rastgoo, and M. Z. Nazri, ''Target searching in unknown environment of multi-robot system using a hybrid particle swarm optimization,'' *J. Theor. Appl. Inf. Technol.*, vol. 96, no. 13, pp. 4055–4065, 2018.

[16] F. A. Raheem and U. I. Hameed, ''Path planning algorithm using D* heuristic method based on PSO in dynamic environment,'' in *Amer. Sci. Res. J. Eng., Technol., Sci. (ASRJETS)*, vol. 49, no. 1, pp. 257–271, 2018.

[17] S. Maneewongvatana and D. M. Mount, ''On the efficiency of nearest neighbor searching with data clustered in lower dimensions,'' in *Computational Science—ICCS*. Berlin, Germany: Springer, 2001, pp. 842–851, 2001.

[18] P. Hart, N. Nilsson, and B. Raphael, ''A formal basis for the heuristic determination of minimum cost paths,'' *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100–107, Jul. 1968, doi: 10.1109/TSSC.1968.300136.

[19] A. Stentz, ''The focussed D* algorithm for real-time replanning,'' in *Proc. Int. Joint Conf. Artif. Intell.*, Aug. 1995, pp. 1652–1659.

[20] Y. Tang, Q. Li, L. Wang, C. Zhang, and Y. Yin, ''An improved PSO for path planning of mobile robots and its parameters discussion,'' in *Proc. Int. Conf. Intell. Control Inf. Process.*, Aug. 2010, pp. 34–38, doi: 10.1109/ICICIP.2010.5565285.

[21] M. M. Alateeq and W. Pedrycz, ''Analysis of optimization algorithms in automated test pattern generation for sequential circuits,'' in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2017, pp. 1834–1839, doi: 10.1109/SMC.2017.8122883.

[22] H. Choset, *Robotic Motion Planning: A* and D* Search*. Boston, MA, USA: Robotics Institute, 2007.

[23] T. Clark. (2017). *Search Algorithm Series: PSO*. Accessed: Apr. 6, 2020. [Online]. Available: https://medium.com/@iamterryclark/swarm-intelli-eb5e46eda0c3

[24] X.-S. Yang, ''Nature-inspired optimization algorithms: Challenges and open problems,'' *J. Comput. Sci.*, vol. 2020, Mar. 2020, Art. no. 101104, doi: 10.1016/j.jocs.2020.101104.

[25] F. A. Raheem and M. I. Abdulkareem, ''Development of path planning algorithm using probabilistic roadmap based on modified ant colony optimization,'' *World J. Eng. Technol.*, vol. 7, no. 4, pp. 583–597, 2019, doi: 10.4236/wjet.2019.74042.

[26] T. T. Mac, C. Copot, D. T. Tran, and R. De Keyser, ''Heuristic approaches in robot path planning: A survey,'' *Robot. Auto. Syst.*, vol. 86, pp. 13–28, Dec. 2016, doi: 10.1016/j.robot.2016.08.001.

[27] S. Thabit and A. Mohades, ''Multi-robot path planning based on multi-objective particle swarm optimization,'' *IEEE Access*, vol. 7, pp. 2138–2147, 2019, doi: 10.1109/ACCESS.2018.2886245.

[28] H. Zhang, Y. Wang, J. Zheng, and J. Yu, ''Path planning of industrial robot based on improved RRT algorithm in complex environments,'' *IEEE Access*, vol. 6, pp. 53296–53306, 2018, doi: 10.1109/ACCESS.2018.2871222.

[29] X. Sun, and S. Koenig, S, ''The fringe-saving A* search algorithm—A feasibility study,'' in *Proc. IJCAI Int. Joint Conf. Artif. Intell.*, 2007, pp. 2391–2397.

[30] A. T. Sadiq and A. H. Hasan, ''Robot path planning based on PSO and D*—Algorithmsin dynamic environment,'' in *Proc. Int. Conf. Current Res. Comput. Sci. Inf. Technol. (ICCIT)*, Apr. 2017, pp. 145–150, doi: 10.1109/CRCSIT.2017.7965550.

[31] C. M. Lee and M. N. Ab Wahab. (Aug. 2020). *Path Planning for Mobile Robot Navigation in Unknown Indoor Environments Using Hybrid Metaheuristic Algorithms Source Code*. Accessed: Aug. 26, 2020. [Online]. Available: https://github.com/chingmay131/robot-path-planning-hybrid-pso

MOHD NADHIR AB WAHAB (Member, IEEE) received the B.Eng. degree (Hons.) in mechatronics engineering and the M.Sc. degree in mechatronics engineering from Universiti Malaysia Perlis, in 2010 and 2012, respectively, and the Ph.D. degree in robotics and automation system from the University of Salford, U.K., in 2017. He is currently a Lecturer with the School of Computer Sciences, Universiti Sains Malaysia. His main research interests include mobile robotics, computer vision, optimization, navigation, and path planning.

CHING MAY LEE received the bachelor's degree (Hons.) in computing studies from Northumbria University, U.K., in 2017. She is currently pursuing the master's degree with Universiti Sains Malaysia. From 2017 to 2019, she was working as a Research and Development Software Engineer with Keysight Technologies Malaysia. Her current research interests include artificial intelligence and data science.

MUHAMMAD FIRDAUS AKBAR (Member, IEEE) received the B.Sc. degree in communication engineering from the International Islamic University Malaysia (IIUM), Malaysia, in 2010, and the M.Sc. and Ph.D. degrees from The University of Manchester, Manchester, U.K, in 2012 and 2018, respectively. From 2010 to 2011, he was with Motorola Solutions, Malaysia, as a Research and Development Engineer. From 2012 to 2014, he was an Electrical Engineer with Usains Infotech Sdn Bhd, Malaysia. He is currently a Senior Lecturer with Universiti Sains Malaysia (USM). His current research interests include electromagnetics, microwave nondestructive testing, and microwave sensor and imaging.

FADRATUL HAFINAZ HASSAN received the Ph.D. degree in computer science (CS) from the School of Information Systems, Computing and Mathematics, Brunel University, London, in 2013. She is currently a Senior Lecturer with the School of Computer Sciences, Universiti Sains Malaysia. Her research interest includes artificial intelligence (AI) for pedestrian simulation and spatial layout optimization. She has coauthored over 30 publications and secured 11 research grants; six as principal investigators and five grants as co-investigators. She currently involved in research studying pedestrian simulation models in the urban planning domain at the School of Architecture, Design and Planning, The University of Sydney.