IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 10, 2018, accepted November 30, 2018, date of publication December 25, 2018,
date of current version January 7, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2886245

Multi-Robot Path Planning Based on Multi-Objective Particle Swarm Optimization

SAHIB THABIT AND ALI MOHADES

Laboratory of Algorithms and Computational Geometry, Mathematics and Computer Science Faculty, Amirkabir University of Technology (Tehran Polytechnic), Tehran 15875-4413, Iran

Corresponding author: Ali Mohades (mohades@aut.ac.ir)

ABSTRACT In this paper, a new method is proposed for the path planning of multi-robots in unknown environments. The method is inspired by multi-objective particle swarm optimization (MOPSO) and is named multi-robot MOPSO. It considers shortness, safety, and smoothness. Due to the obscurity of the environment, the robots should decide the moving direction based on the information gathered by sensors only such that the optimal path between the start and goal positions can be found at the end of the algorithm. Sharing knowledge among the navigating robots is necessary to achieve this aim. So, a new concept, named the probabilistic window, is introduced in this paper. It combines the current information obtained through the robot sensors and experiences of the previous robots to select the paths that seem more likely to achieve higher fitness in the mentioned objectives. The proposed method has an outstanding performance on different complex benchmarks, and the results have shown that it is more effective and efficient compared with the classic and the state-of-the-art methods.

INDEX TERMS Path planning, trajectory optimization, particle swarm optimization, multi-robot systems.

I. INTRODUCTION

Path planning (PP) is one of the principal procedures in Multi-Robot Systems (MRS) which have widespread application nowadays [1]. It can be described as seeking a valid and optimal path between two positions without colliding with the obstacles. Various approaches have been proposed to solve PP problem [2] but generating an optimal obstacle-free path in MRS remains challenging due to its NP-hard nature and complexity. These methods can generally be classified into two main groups: Offline (global) and online (local). In global algorithms, all the information about the environment is known in advance. But, there is no complete information about the environment in local algorithms. Hence, the algorithms depend on the information gathered by the sensors [3].

Offline approaches such as cell decomposition and roadmap [4] partition the workspace into obstacle free cells that are considered as the nodes of a graph. The existence of a direct route between adjacent cells is represented by the edges of the graph. Then, a graph searching algorithm is used to find an optimal path [5]. The offline approaches are not suitable for high-dimensional space and real-time applications [6]. Furthermore, these approaches might get trapped in a local minima in large environments with several solutions [7].

Online methods such as RRT [8] and potential fields [9] are proposed to solve the PP problem in unknown dynamic environments. These methods provide the ability to react to the changes during the robot movement. However, due to factors such as high dimensionality of the search space, the geometric nature of the obstacles, and the complexity of robot's kinematic and dynamic model, PP problem cannot be solved, given reasonable computational resources [10].

Utilizing heuristic approaches inspired from biological or sociological systems like Genetic Algorithm (GA) [11], particle swarm optimization (PSO) [12] and Ant Colony Optimization (ACO) [13], [14] is a very popular way to solve the PP problem. These methods often achieve higher performance in comparison with the classical approaches, especially in complex environments. But, there are two main defects in the heuristic methods proposed previously. Firstly, in most of the existing algorithms, it is assumed that the environment is totally or at least partially known. Secondly, these approaches are proposed for the single robot systems. To alleviate the mentioned shortcomings, a heuristic

approach is proposed for solving the path planning problem of multi-robot systems in an unknown environment in this paper. The proposed method finds the optimal path via sharing knowledge among navigating robots and exploiting the power of Multi-Objective PSO (MOPSO).

Shortness, safety, and smoothness are three considered objectives and the proposed algorithm tries to provide the best Pareto front paths in the mentioned term. One of our main contributions in this paper is introducing the concept of the probabilistic window. It quantifies the information obtained through sensors and experiences of the previous robots by assigning a probability to the reachable positions. Then, the robot moves according to these probabilities. Since unknowingness of the environment, there is no predefined map and this window was evolved over a PSO inspired procedure. The proposed method is named Multi-robot Multi-objective Particle Swarm Optimization (MMPSO). The simulation results show its superiority to other state of the art and classical PP methods.

The remainder of this paper is organized as follows. Some of the related works are reviewed in the next section. The third section provides the proposed method. Results and the conclusion are presented in fourth and fifth sections, respectively.

II. RELATED WORK

The idea of utilizing Multi-Objective Optimization (MOO) algorithms for PP is deeply exploited before [15], [16]. Some of the most relevant literature to this subject is reviewed in the following.

Masehian and Sedighizadeh [17] proposed a multi-objective path planning method. This hybrid algorithm employs PSO to generate Probabilistic Road Map (PRM). The main idea of the algorithm is using PRM for collision avoidance while PSO is responsible for searching the best path in the global space; the algorithm considers two objectives, the shortness, and the smoothness. Gong *et al*. [18] divided the robot workspace by a series of horizontal, and vertical parallel lines and employed MOPSO to tackle the PP problem. The considered objectives are the safety and the path length. This algorithm works based on a foreknowable map of the environment which is not always possible.

Bhattacharjee *et al.* [19] employed the Artificial Bee Colony (ABC) algorithm to tackle the PP problem for an MRS. The algorithm assigns fitness values to the paths of robots and tries to find the motion trajectories which minimize the moving distance of all robots. But, the main issue is that the algorithm may not work properly because the objective function could take several values. Wang *et al.* [20] used the improved multi-objective ACO. The objective function is proposed in such a way that it has a tendency to a short, secure and smooth path. This algorithm assumes that the environment is known in advance, too. Hidalgo-Paniagua *et al.* [21] presented a heuristic algorithm for solving PP problem in the static environments. The method tries to find the shortest, the safest, and the smoothest path based on the Firefly Algorithm (FA). Panda *et al.* [22] presented a hybrid algorithm for multi-robot PP problem in the static environments. The authors combined FA algorithm with Invasive Weed Optimization (IWO) to overcome the limitation of slow convergence of the FA algorithm in large space problems and provided a balance between exploration and exploitation in the workspace. The presented approach by Hajimirsadeghi and Lucas [23] solves the PP problem for multi-robot in a dynamic environment using Improved Gravitational Search Algorithm (IGSA). The algorithm only considers the length of the robot paths.

Jun and Qingbao [24] employ the NSGA-II to solve multi-objective path planning in a known environment. The authors considered shortness, smoothness, and secureness as the objectives. They introduce a chaotic method that uses the acquired knowledge from the environment to increase the performance of the algorithm. Ahmed and Deb [25] used an NSGA-II algorithm with enhanced selection scheme to solve multi-objective PP with the same three objectives. Davoodi *et al.* [26] applied GA to solve the multi-objective PP problem in known discrete environments. They considered the shortness and safety in fitness function and introduced a new operator which is claimed to cause better exploration of the search space. Geetha *et al.* [27] proposed multi-objective path planning based on a hybrid algorithm. They combined ACO and GA to achieve three objectives including safety, smoothness, and shortness. In order to increase the efficiency of the proposed approach, the genetic operator was adapted with ACO. There are two main drawbacks in the proposed algorithm. First, the approach of computing the optimal solution has ambiguity and second, the algorithm can deal with known environments while in many situations, the robot must work in unknown environments that necessitate the real-time process.

In this paper, we employ the PSO algorithm due to the simplicity, efficiency and faster convergence. It is worth mentioning here that, all the mentioned approaches do not have any post-processing step. But, the found paths by the proposed algorithm are enhanced through an extra post-processing step.

III. METHODOLOGY

In this section, we briefly discuss the assumption and the problem formulation first. Then the PSO algorithm and the general procedure of the proposed method are mentioned.

A. ASSUMPTION AND PROBLEM DEFINITION

A group of robots traverses within an environment E, searching for a specified goal point. As the group reaches the goal, they send the path information to the next group. The set of robots is given as (1).

$$S = \left\{R_i^j\right\} i = 1, 2, \ldots, G_s \quad j = 1, 2, \ldots, N_g \tag{1}$$

where the subscript i denotes the robot number inside the group, the superscript j denotes the group number, G_s denotes the total size of the group, and N_g denotes the total number of groups. It is assumed that every robot can be modeled as a free

moving point in a two-dimensional space that is considered as a grid cell. Each robot has equipped with S_n sensors that can measure the distance to the surrounding obstacles and the distance to the goal point. The configuration of sensors provides information of a circular area with radius r that is called selection window SW (see Fig. 1). The selection window is divided into S_n sectors and every grid cell is assigned to the sector that covers its most area.

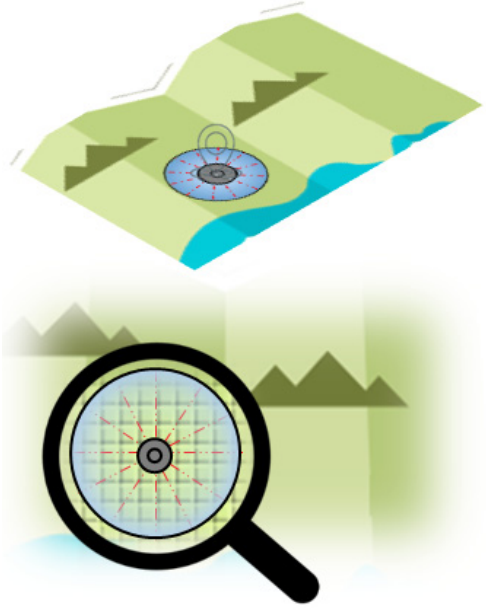

FIGURE 1. Schematic of the selection window.

A finite subset of the environment cells is occupied by the obstacles. Furthermore, different configurations of obstacles in terms of shape, density, occupancy level, and positions may exist. We assume that a cell is either entirely occupied by an obstacle or free of obstacles.

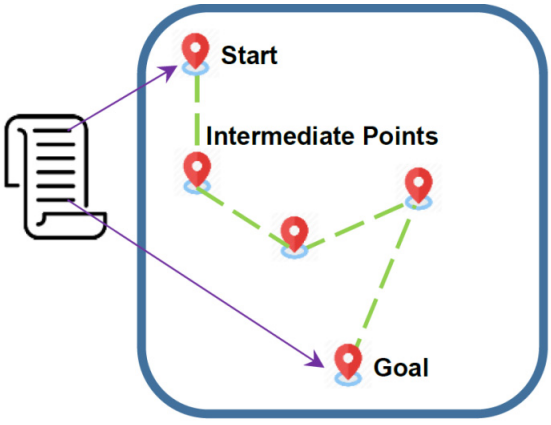

FIGURE 2. Schematic of a path.

As it is shown in Fig. 2, each path is a sequence of coordinate's doublets (row, column). The first cell of each path corresponds to the start point while the final cell of the list corresponds to the goal point. It may be different number of intermediate cells in each path.

The PP problem is finding the shortest, safest and smoothest available path due to the multi-objectiveness of the problem. There could be multiple non-dominating solutions instead of one unique solution. The quantitative measure of the mentioned objectives and the proposed method are described in the following subsections.

B. PARTICLE SWARM OPTIMIZATION (PSO)

Initially, PSO is introduced by Eberhart and Kennedy [28] in 1995 and got popular due to its simplicity. In PSO, the swarm of particles is initialized by random positions and velocities. The velocity parameter tunes the movement of particles. Each particle moves toward the best-found position inside the group that is called Pbest and the best-found position by all member of swarm Gbest. The position and velocity of the i_{th} particle in t_{th} iteration are denoted by $x_i(t)$ and $v_i(t)$, respectively and can be calculated through (2) and (3).

$$v_i(t+1) = wv_i(t) + r_1c_1\left(Pbest_i(t) - x_i(t)\right) + r_2c_2(Gbest(t) - x_i(t)) \quad (2)$$

$$x_{i+1}(t) = x_i(t) + v_i(t+1) \quad (3)$$

where c_1, c_2 are the acceleration coefficients. r_1, r_2 are the random numbers uniformly distributed in the range of [0, 1], and w is a real value [29].

C. GENERAL PROCEDURE

The pseudocode of the proposed algorithm is mentioned in Algorithm 1. At each iteration, a group of robots searches the goal. Inside each group, the robots move in turn. After each robot reaches the goal, it does a post-processing step to obtain the enhanced path. So, whenever all the j_{th} group robots attain the goal, a set of paths $PathG^j$ are at hand and the evaluation routine is triggered. These paths are evaluated using three objectives (fitness) functions FG^j.

$$PathG^j = \{EnahncedPathR^j_1, \ldots, EnhancedPathR^j_{G_s}\} \quad (4)$$

$$FG^j = \{FR^j_1, FR^j_2, \ldots, FR^j_{G_s}\} \quad (5)$$

$$FR^j_i = \{PLF^j_i, PSF^j_i, P\alpha F^j_i\} \quad (6)$$

where PLF^j_i denotes the path length, PSF^j_i denotes the path safety, and $P\alpha F^j_i$ denotes the path smoothness. If the best Pareto paths of the j_{th} group dominates the current Pareto front members, the Pareto front is modified. So in each iteration, the Pareto front contains the best found path, heretofore. Afterward, a new group of robots will roam through the environment for finding the goal, starting from the same initial position while carrying the solution generated by previous groups. The goal is to provide the Pareto front path from the start point to the goal point in three considered objectives.

Combining experiences of previous robots and current sensor measurement is a straight way to this goal in such an unknown environment. In this way, the paths of robots are

Algorithm 1 Pseudocode of MMPSO

1: **MMPSO** ( )
2: **input**: E, G_s, N_g, *Start*, *Goal*
3: **output**: *ParetoFront*
4: $GroupIDX = 1$
5: $ParetoFront = []$
6: **while**(GroupIDX $< N_g + 1$) **do**
7: $RobotIDX = 1$
8: $BestPath^{\text{GroupIDX}} = []$
9: $PathG^{\text{GroupIDX}} = []$
10: **while**($RobotIDX < G_s + 1$) **do**
11: $EnhancedPathR_{RobotIDX}^{\text{GroupIDX}} =$ ***Roam_Robot***$(RobotIDX, GroupIDX, Start, Goal)$
12: Add $EnhancedPathR_{RobotIDX}^{\text{GroupIDX}}$ to $PathG^{\text{GroupIDX}}$
13: $RobotIDX = RobotIDX + 1$
14: **endwhile**
15: $ParetoBest^{GroupIDX} =$ ***Evaluation***$(PathG^{GroupIDX} N_g)$
16: **if**($ParetoBest^{GroupIDX}$ dominates *ParetoFront*)
17: Modify *ParetoFront* according to $ParetoBest^{GroupIDX}$
18: **end if**
19: GroupIDX = GroupIDX + 1
20: **end while**
21: **end MMPSO**

evolved and get enhanced whatever more robots traverse the environment. The combination is considered in each robot moving strategy and will be completely described in the following.

Algorithm 2 Pseudocode of Individual Robot PP

1: **Roam_Robot**( )
2: **input**: *RobotIDX*, *GroupIDX*, *Start*, *Goal*
3: **output**: $EnhancedPathR_{RobotIDX}^{GroupIDX}$
4: $PathR_{RobotIDX}^{\text{GroupIDX}} = []$
5: **while**($Robot^{RobotIDX}$ *does not reach the Goal*) **do**
6: Update Selection Window Trough Sensors Information
7: Update Probabilistic Window
8: select one position to move according to assigned probability by Probabilistic Window
9: Add selected Position to $PathR_{RobotIDX}^{\text{GroupIDX}}$
10: **end while**
11: $EnhancedPathR_{RobotIDX}^{GroupIDX} =$ ***PostProcess***$(PathR_{RobotIDX}^{\text{GroupIDX}})$
12: **end Roam_Robot**

The pseudocode of individual robot PP procedure is mentioned in Algorithm 2. During the movement, each robot updates the selection window by information gained from the surrounding environment through equipped sensors. Then, the probabilistic window is updated. It determines which position seems more promising according to the current information and previous experiences. The robot will move according to the assigned probability by the probabilistic window until it reaches the goal. After the robot arrival, the post-processing procedure starts.

The evaluating procedure of the probabilistic window will be described completely in the next section. It is designed in such a way that causes robots to move toward the global best solution of the former groups while reduces the probability of collision with discovered obstacles. It gives a reasonable weight to exploring the environment to finding shorter and smoother paths. After all robots reach the goal, a non-dominated sorting algorithm will be applied on the found path by different groups and the best paths are stored in the Pareto front.

It can be presumed that paths are encoded as the points of a new space and a multi-objective fitness value is assigned to them. Then, a PSO inspired algorithm is used to search the new space for the global min which is representative of the shortest, safest and smoothest path. Every single robot that roams around the environment can be taught as a particle which explores the search space. The robot has a tendency to the global best points while keep trying to explore the space individually. The set of non-dominated paths that are found by robots are considered as the Pareto optimal solutions and returned by the algorithm.

D. PROBABILISTIC WINDOW

To convert the sensor measurements into the decision variables, we introduce the concept of the probabilistic window. It normalizes the sensor measurements to a global scale. Let the i_{th} robot of the j_{th} group be in the grid cell (C), all the reachable cells are denoted by $N(C)$ and $PW_{\text{i}}^{\text{j}}(C)$ denotes the probabilistic window of the grid cell (C). $PW_{\text{i}}^{\text{j}}(C)$ is a probability measure and assigns a probability to all of the cells in the $N(C)$. The robot selects one of $N(C)$ members as the next cell to move according to a probability value assigned by $PW_{\text{i}}^{\text{j}}(C)$. Equations (7) to (9) present a formal definition of $PW_{\text{i}}^{\text{j}}(C)$.

$$PW_i^j(C) : N(C) \rightarrow \{0, 1\} \tag{7}$$

$$PW_i^j(C) = \left\{PW_i^j(c_n)\right\} \quad \forall n \in N(c) \tag{8}$$

$$PW_i^j(c_n) = \frac{PV_i^j(c_n)}{\sum_{n \in N(c)} PV_i^j(c_n)} \tag{9}$$

where $PV_{\text{i}}^{\text{j}}(C_n)$ is the probabilistic value of the n_{th} cell in $N(C)$. It is considered as the mean of probabilistic distance window $(PDW_{\text{i}}^{\text{j}}(C_n))$ and probabilistic sector window $(PSW_{\text{i}}^{\text{j}}(C))$.

$$PV_i^j(c_n) = (1 + GB(c_n)) \times mean\left(PDW_i^j(c_n), PSW_i^j(c_n)\right) \tag{10}$$

where $GB(c_n)$ returns 1 if c_n is located on one of the previous global best paths otherwise, it returns 0.

The first term in (10) doubles the probability of selecting points on the global best path of the previous groups.

$PDW_i^j(C_n)$ is calculated by adding the normalized value of the distance matrix ($D(c_n)$) and the occupancy matrix ($O(c_n)$) according to (11).

$$PDW_i^j(c_n) = \frac{\alpha\,(Max\_D - D\,(c_n))}{Max\_D} + (1-\alpha)\,\frac{(Max\_O - O\,(c_n))}{Max\_O} \quad (11)$$

$$Max\_D = \max_{n\in N(C)} D(c_n) \quad (12)$$

$$Max\_O = \max_{n\in N(C)} O(c_n) \quad (13)$$

where O (c_n) is the occupancy value of cell (c_n) and belongs to [0, 1] where 0 shows the emptiness of the cell and 1 means that the cell is full. D (c_n) is the shortest distance from the cell (c_n) to the virtual line connecting the start and goal points. Max_O and Max_D are the maximum values of occupancy and cell distances within the probabilistic window. The virtual line (l) is defined mathematically in (14) and $D(c_n)$ is computed by (15).

$$l = \left\{ \begin{bmatrix} x \\ y \end{bmatrix} | y = mx + b;\ m = \frac{Y_{goal} - Y_{start}}{X_{goal} - X_{start}},\ b = Y_{start} - mX_{start} \right\} \quad (14)$$

$$D(c_n) = \frac{|b + mX_{c_n} - Y_{c_n}|}{\sqrt{1+m^2}} \quad (15)$$

where X_{c_n} and Y_{c_n} are the Euclidean coordinates of the cell (c_n), m and b are the slope and intercept of the virtual line, respectively. α is a coefficient in the range of [0, 1] that tunes the tradeoff between shortness and security of the path. Selecting near zero value for this parameter causes tending to short paths while near one values drive robots to the safer paths.

If the covered grid cells by k_{th} sector are denoted by $CN(S_k)$, the occupancy of the k_{th} sector for the i_{th} robot in the j_{th} group $SO_i^j(s_k)$ is defined as the sum of occupancy of its grid cell.

$$SO_i^j(s_k) = \sum_{c_n \in CN(S_k)} O(c_n) \quad (16)$$

The probabilistic sector window of the cell (c_n) is calculated according to (17).

$$PSW_i^j(c_n) = \sum_{k=1}^{12} \delta(c_n, s_k)\left(1 - \frac{SO(s_k)}{TO(C)}\right) \quad (17)$$

where $\delta(c_n s_i)$ function returns 1 if the cell (c_n) is covered by the sector s_k otherwise, it returns 0. $TO(c)$ denotes the total occupancy of all sectors and calculated through (18).

$$TO(C) = \sum_{k=1}^{12} SO(s_k) \quad (18)$$

E. PATH POST PROCESSING

This process enhances the path in terms of shortness and smoothness by omitting redundant points. It starts after the robot arrival and continues until the state in which adding new non-crossing segment is impossible or continuous for some constant iteration. An exemplary schematic is depicted in Fig. 3 and pseudocode of the procedure is mentioned in Algorithm 3.

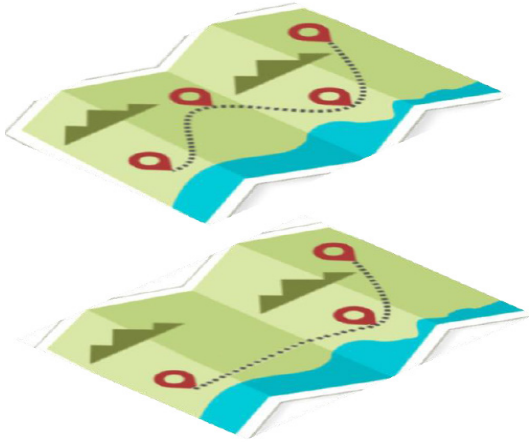

FIGURE 3. Schematic of path enhancement.

Algorithm 3 Pseudocode of Post-Processing

```
Post − Processing( )
input: PathR^GroupIDX_RobotIDX
output: EnhancedPathR^GroupIDX_RobotIDX
EnhancedPathR^GroupIDX_RobotIDX = PathR^GroupIDX_RobotIDX
while (MaxIteration or Enhancementisimpossible)
    for i = 1 : 1 : length(EnhancedPath^RobotIDX)
        for j = length(EnhancedPath^RobotIDX) : −1 : i + 2
            if(line(Path^RobotIDX(i) andPath^RobotIDX(j)) is safe)
                Modify EnhancedPathR^GroupIDX_RobotIDX
            end if
        end for
    end for
end while
: end Post − Processing
```

F. PATH EVALUATION

The evaluation process investigates the post-processed paths in terms of shortness, safety and smoothness. These objectives are discussed for a path containing n cells in the following and the pseudocode of this process is mentioned in Algorithm 4.

The path length is the total Euclidean distance roamed by the robot. It is obtained through summing out the lengths of

Algorithm 4 Pseudocode of Path Evaluation

1: **Evaluation**( )
2: **input**:$PathG^{GroupIDX}$, G_S
3: **output**: $ParetoBest^{GroupIDX}$
4: **for** $RobotIDX = 1 : 1 : G_S$
5: $\mathrm{FR}^{GroupIDX}_{RobotIDX}(1) = \mathrm{PLF}(EnhancedPathR^{GroupIDX}_{RobotIDX})$
6: $\mathrm{FR}^{GroupIDX}_{RobotIDX}(2) = \mathrm{PLS}\left(EnhancedPathR^{GroupIDX}_{RobotIDX}\right)$
7: $\mathrm{FR}^{GroupIDX}_{RobotIDX}(3) = \mathrm{P}\alpha\mathrm{F}(EnhancedPathR^{GroupIDX}_{RobotIDX})$
8: **end for**
9: $\mathrm{ParetoBest}^{\mathrm{GroupIDX}}$=$\mathrm{NondominantFR}^{GroupIDX}$ elements
10: **end Evaluation**

path segments using (19).

$$PL = \sum_{\mathrm{k}=1}^{\mathrm{n}-1} \mathrm{d}(\mathrm{L}[\mathrm{k}], \mathrm{L}[\mathrm{k}+1]) \tag{19}$$

where $d(L[k]L[k+1])$ is the segment length between the k_{th} and $(\mathrm{k}+1)_{\mathrm{th}}$ coordinates of the path list. The nearest path between two points, is a direct line. So, the shortness fitness is estimated by (20).

$$\mathrm{PLF} = \frac{\mathrm{d}(\mathrm{Start}, \mathrm{goal})}{\mathrm{PL}} \tag{20}$$

where $d(Start, goal)$ is the length of the direct line between start and goal positions.

The $MDO(c)$ for any arbitrary grid cell (c) is defined as the distance to the nearest obstacle. Considering the sensor ranges, if the distance to the nearest obstacle is greater than *Sensor Threshold*, we set the $MDO(c)$ value to *Sensor Threshold*. The safety fitness is estimated by mean of path cells *MDO*, according to (22). We set the *Sensor Threshold* to 5 in our experiments.

$$\mathrm{PS} = \sum_{\mathrm{i}=1}^{\mathrm{n}} \min(\text{Sensor Threshold}, \mathrm{MDO}(\mathrm{C_i})) \tag{21}$$

$$\mathrm{PSF} = \frac{\mathrm{PS}}{\mathrm{n} \times \text{Sensor Threshold}} \tag{22}$$

The smoothness fitness measure how snaky is a specific path. As it is depicted in Fig. 4, the smoothness depends on the angels between consecutive segments. The best angle for being smooth is the closest one to 180° which means that the path segment is a straight line. So, the smoothness objective function can be estimated as the division of angels mean in the consecutive path segments by 180 as (24).

$$P\alpha = \frac{1}{n-2} \sum_{k=1}^{n-2} \alpha\left(L(k), (k+2)\right) \tag{23}$$

$$P\alpha F = \frac{P\alpha}{180} \tag{24}$$

where $\alpha(k, k+2)$ is the angle between the consecutive path segments $\overrightarrow{L(k)L(k+1)}$ and $\overrightarrow{L(k+1)L(k+2)}$, $0 \leq \alpha(.) \leq \pi$.

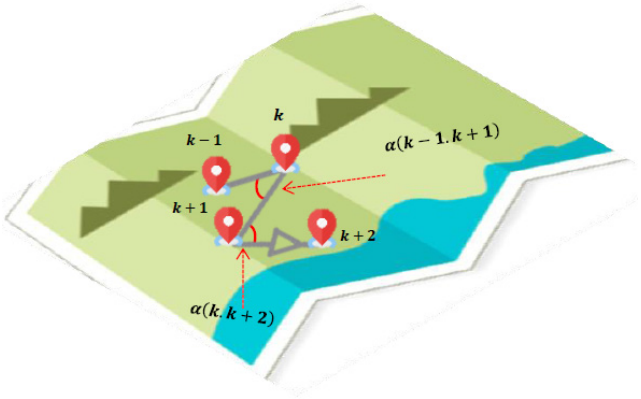

FIGURE 4. Schematic of path angle.

IV. RESULTS

This section contains two parts. In the first part, we investigate our proposed algorithm on different simple environments containing circular, oval and polygonal obstacles. The algorithm is compared with two state of the art methods in publicly available complex benchmarks for PP.

A scenario is formed by a map, a starting point, and a goal point. To compute the objective values of founded path in this section, we use the mentioned functions in the last subsection (Section III. F). formulations are completely discussed in this subsection; we just substitute the values in the formulas. The values of parameters that are used in the simulations are mentioned in Table 1.

TABLE 1. Values of parameters.

Parameter	Value
N_g	5
G_s	4
S_n	12
α	0.5
Sensor Threshold	5

A. SIMPLE ENVIRONMENTS

Four sparse and overcrowded environments are considered in this section. All the simple environments with sparse obstacles are 100 × 100 grids. The simple environments with overcrowded obstacles are 100 × 200 grids. The starting and the goal points in all scenarios are determined. But, the environments are completely unknown. The algorithm is compared with one heuristic and two classical approaches. The selected classical approaches are Potential Field (PF) and Probabilistic Roadmap (PRM), and Firefly Algorithm (FA) is chosen among the heuristic approach. The features of the selected algorithm are mentioned in TABLE 2. All of the methods except MMPSO assume all information about the environment is available. As it is declared, PF considers the shortness and safety objectives through attractive and

TABLE 2. Applied methods.

Algorithm	Using Sensor	Shortness	Safety	Smoothness
PRM	×	×	✓	×
PF	✓	×	✓	×
FA	×	✓	✓	✓
MMPSO	✓	✓	✓	✓

repulsive forces. Likewise, the applied FA algorithm considers the shortness and smoothness objectives.

To provide a reference path that makes comparison and evaluation of the optimality degree easier and due to the simplicity of the environments, we use a Brute Forth approach and produce a huge but limited number of paths. Then, the objective function of these paths is calculated and the most optimum path is considered as an approximation of Pareto optimal path in three mentioned objectives. The found path is depicted in violet color in all figures in this section. Whatever a path is closer to the ideal reference point, it means, it is the better approximation of Pareto optimal solution.

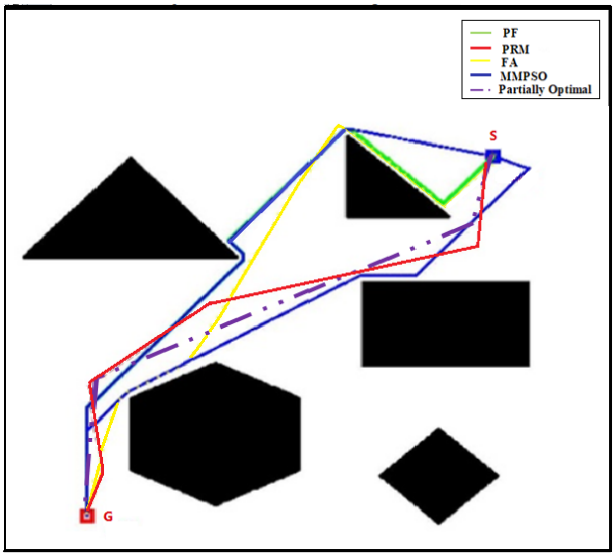

FIGURE 5. Sparse polygonal environment experiment – 1.

The found paths by the mentioned algorithm in a simple environment containing a few polygonal obstacles are depicted in Fig. 5 and Fig. 6. As it can be seen, the algorithm found the near-optimal path and performs better, although other methods are offline. The quantitative results are stated in TABLE 4 and TABLE 5. In the first scenario, MMPSO found two paths as Pareto optimal and both of them are drawn. One of these paths is similar to the PF path but shorter and smoother. This path is denoted by MMPSO/1 in TABLE 3. It is worth mentioning that these methods consider fewer objectives, the results showed that our proposed method achieves better.

The values of the objective functions in a crowded environment with the polygonal obstacles are mentioned in TABLE 5

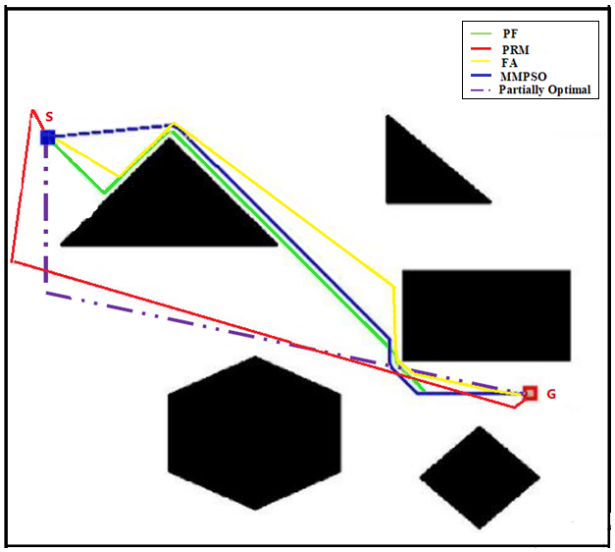

FIGURE 6. Sparse polygonal environment experiment - 2.

TABLE 3. Sparse polygonal environment experiment – 1.

Algorithm	Shortness	Safety	Smoothness
PRM	0.73	0.96	0.80
PF	0.78	0.82	0.80
FA	0.82	0.74	0.80
MMPSO/1	0.81	0.83	0.83
MMPSO/2	0.79	0.81	0.87
Partially Optimal	0.85	0.84	0.93

TABLE 4. Sparse polygonal environment experiment - 2.

Algorithm	Shortness	Safety	Smoothness
PRM	0.75	0.89	0.81
PF	0.55	0.38	0.56
FA	0.77	0.62	0.71
MMPSO/1	0.80	0.91	0.83
MMPSO/2	0.88	0.82	0.86
Partially Optimal	0.90	0.83	0.88

TABLE 5. Crowded polygonal environment experiment.

Algorithm	Shortness	Safety	Smoothness
PRM	0.65	1.000	0.85
PF	0.75	0.53	0.77
FA	0.76	0.63	0.82
MMPSO	0.78	0.75	0.90
Partially Optimal	0.92	0.98	0.94

and the found paths are depicted in Fig. 7. MMPSO found two paths. The found path that is similar to the partially-optimal path is denoted by MMPSO/2 in TABLE 5, and the objective values are rounded up to two decimal places.

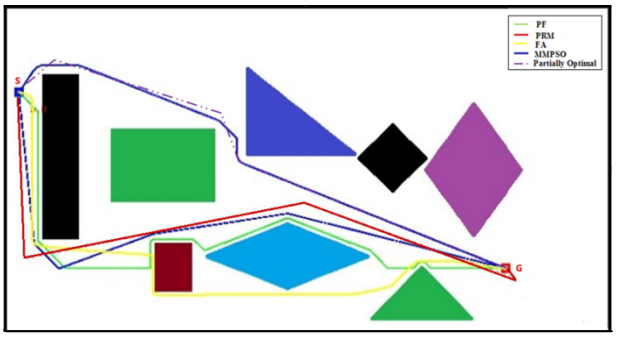

FIGURE 7. Crowded polygonal environment experiment.

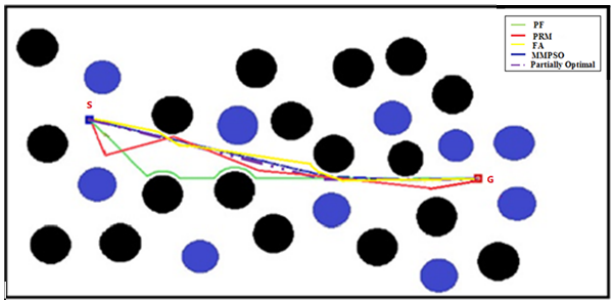

FIGURE 8. Circular environment experiment - 1.

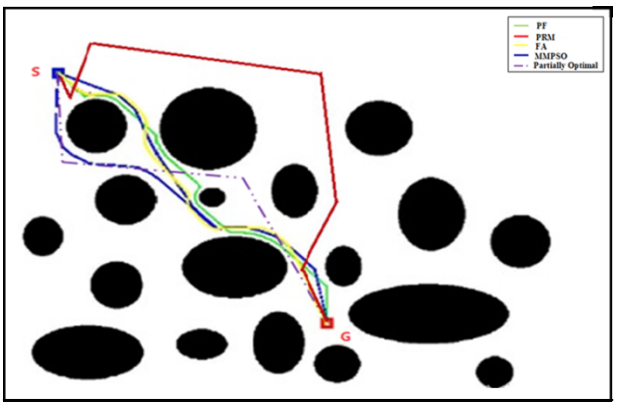

FIGURE 9. Circular environment experiment – 2.

Output paths of four above mentioned algorithms are drawn in Fig 8 and Fig.9 in two sparse and overcrowded environments with oval and circular obstacles, and the objective values are reported in TABLE 6 and TABLE 7. As the numerical results verify, MMPSO performs better in comparison with the other methods in both scenarios.

The superiority of the algorithm to the other methods is obvious in all of the above-mentioned scenarios. Although the output path of MMPSO maybe not the shortest available path, the safety is perfect and the smoothness is better in comparison with the other methods. It should be noted that the MMPSO is an online algorithm which only gets information from the surrounding environment and previous experiences.

TABLE 6. Circular environment experiment - 1.

Algorithm	Shortness	Safety	Smoothness
PRM	0.71	1.000	0.85
PF	0.65	0.53	0.77
FA	0.81	0.63	0.82
MMPSO	0.86	0.75	0.90
Partially Optimal	0.92	0.98	0.94

TABLE 7. Circular environment experiment - 2.

Algorithm	Shortness	Safety	Smoothness
PRM	0.52	1	0.92
PF	0.79	0.72	0.61
FA	0.86	0.82	0.85
MMPSO/1	0.85	0.88	0.89
MMPSO/2	0.85	0.86	0.89
Partially Optimal	0.83	0.92	0.95

B. COMPLEX ENVIRONMENTS

In this section, the evaluation metrics is discussed first. Then, the performance of the proposed method is analyzed and compared with two state of the art methods for PP based on the Non-Dominated Sorting Genetic Algorithm (NGSA) and Multi-Objective Firefly Algorithm (MOFA).

Evaluating the quality of non–dominated set of MOO solutions is an open problem and numerous approaches have been proposed to evaluate this quality. The most common method is quality indicators (QIs). A QI is a function that associates with a real number to the input non-dominated set(s). The most popular QIs are the unary QIs (UQIs) and the binary QIs (BQIs). UQIs take a single non-dominated set as input and return its quality evaluation. A BQI takes two non-dominated sets and outputs a real number that is the evaluation of the relative quality of one set with respect to the other. Majority of the proposed quality measures consider either one or two convergence and diversity metrics. The convergence measures indicate how far the found solutions are from the true Pareto optimal solutions while the diversity metrics specify the scatter of solutions on the true Pareto front. We consider the Hyper-Volume (HV) evaluation metric that is described in the following.

HV is a UQI that enumerates the quality of the input NS with regard to the convergence and diversity on a single scale. It measures the covered volume by solutions in the objective function space. If $S = \{s_1, s_2, \ldots, s_n\}$ be a NS. The HV is calculated as the union of all elements of S hyper-cubes, according to (25). Where $HP(s_i)$ denotes the corresponding hypercube of the i_{th} element of S, and L is the Lebesgue measure [30].

$$\mathrm{HV}\,(\mathrm{S}) = \mathrm{L}\bigcup_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{HP}\,(\mathrm{s_i}) \tag{25}$$

TABLE 8. Complex environment hyper-volume results.

Scenario	NSGA-II /1	NSGA-II/2	MOFA	MMPSO
MAP A	0.599	0.564	0.713	0.718
MAP B	0.544	0.554	0.716	0.741

We used two datasets which are made publicly available by Hidalgo-Paniagua *et al.* [21]. Both datasets have 100 × 100 environments. To compare fairly, we use the same start and goal points. The results of NSGA and MOFA are fetched from the mentioned paper [21] and compared with MMPSO results in the following. Several runs to tune the parameter were run first. In the test time, we repeat each scenario 10 times to be sure about the statistical significance of the obtained results. The path with the median quality metric is depicted as the final result. TABLE 8 shows the median hyper-volume, of path found by MMPSO, NSGA-II/1 [24], NSGA-II/2 [25], and MOFA [21]. Fig. 10 and Fig. 11 show the best paths corresponding to the approximated Pareto front by MMPSO, MOFA versus NSGA-II when they are applied to the PP problem. MMPSO found two paths as the Pareto front in each scenario.

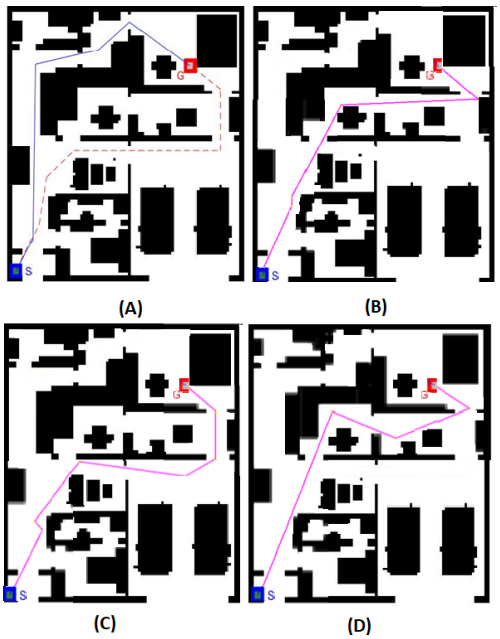

FIGURE 10. Complex environment experiment (Map A) results: (A) MMPSO (B) MOFA (C) NSGA-II/1 (D) NGSA-II/2.

The shortness, safety, and smoothness objective values for first and second complex environments are reported in TABLE 9 and TABLE 10, respectively. The blue line in MMPSO part is referred to MMPSO/1 and the red dashed line is referred to MMPSO/2 in TABLE 9 and TABLE 10. The same reference values are used to evaluate the objectives and the numbers are rounded up to two decimal places. As we can see, for all scenarios, the results obtained through MMPSO dominate solutions of the other algorithms.

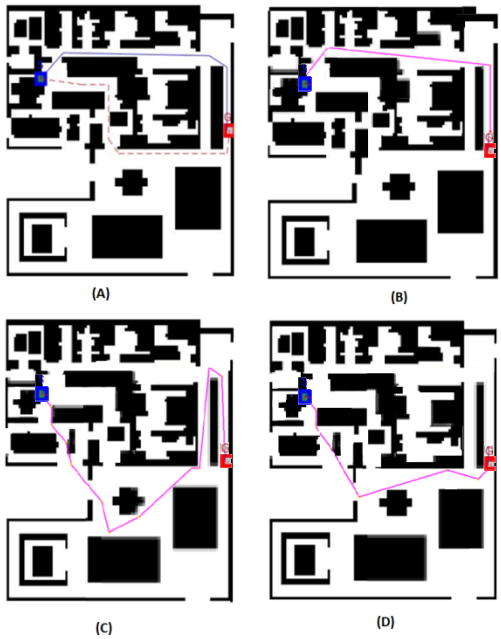

FIGURE 11. Complex environment experiment (Map B) results: (A) MMPSO (B) MOFA (C) NSGA-II/1 (D) NGSA-II/2.

TABLE 9. Complex environment experiment (Map A).

Algorithm	Shortness	Safety	Smoothness
MMPSO/1	0.8623	0.8251	0.8832
MMPSO/2	0.8323	0.8073	0.8161
MOFA	0.8723	0.8035	0.8509
NSGA-II /1	0.7605	0.8510	0.7909
NSGA-II /2	0.8182	0.8902	0.8429

TABLE 10. Complex environment experiment (Map B).

Algorithm	Shortness	Safety	Smoothness
MMPSO/1	0.8623	0.8251	0.8832
MMPSO/2	0.8323	0.8073	0.8161
MOFA	0.8723	0.8035	0.8509
NSGA-II /1	0.7605	0.8510	0.7909
NSGA-II /2	0.81823	0.8902	0.8429

V. CONCLUSION AND FUTURE WORK

We proposed a multi-objective path planning approach in this paper. Our main novelty is the introduction of the probabilistic window concept. It unifies the current sensor information and previous robot experience. It assigns a probability to each reachable position. The probability assignment is designed inspiring from PSO and makes the selection of the shorter, safer and smoother paths more probable. In addition, to achieve effective and accurate solutions (paths), a post-processing method, which enhances the paths in terms of

shortness and smoothness, is presented. The obtained results show the superiority of the proposed approach in simple and complex benchmark in comparison with the classic and state of the art PP methods. In our future work, we shall validate the approach on more complex obstacles configurations and its implementation it on real-world scenarios.

REFERENCES

[1] R. Lin and W. Guo, "Novel design of a family of legged mobile Lander," in *Proc. Int. Conf. Intell. Robot. Appl.*, 2017, pp. 261–272.
[2] J. Barraquand and J.-C. Latombe, "Robot motion planning: A distributed representation approach," *Int. J. Robot. Res.*, vol. 10, no. 6, pp. 628–649, 1991.
[3] S. Liu, D. Sun, and C. Zhu, "Coordinated motion planning for multiple mobile robots along designed paths with formation requirement," *IEEE/ASME Trans. Mechatronics*, vol. 16, no. 6, pp. 1021–1031, Dec. 2011.
[4] A. Swingler, "A cell decomposition approach to robotic trajectory planning via disjunctive programming," Duke Univ., âĂŐDurham, NC, USA, Tech. Rep., 2012.
[5] L. Dai, "Fast shortest path algorithm for road network and implementation," School Comput. Sci., Carleton Univ., Ottawa, ON, Canada, Tech. Rep. 4905, 2005.
[6] P. Raja and S. Pugazhenthi, "Optimal path planning of mobile robots: A review," *Int. J. Phys. Sci.*, vol. 7, pp. 1314–1320, Feb. 2012.
[7] J. Lee and D.-W. Kim, "An effective initialization method for genetic algorithm-based robot path planning using a directed acyclic graph," *Inf. Sci.*, vol. 332, pp. 1–18, Mar. 2016.
[8] L. Jaillet and J. M. Porta, "Path planning with loop closure constraints using an atlas-based RRT," in *Robotics Research*. Cham, Switzerland: Springer, 2017, pp. 345–362.
[9] Y. Rasekhipour, A. Khajepour, S.-K. Chen, and B. Litkouhi, "A potential field-based model predictive path-planning controller for autonomous road vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 5, pp. 1255–1267, May 2017.
[10] Z. Shiller, "Off-line and on-line trajectory planning," in *Motion and Operation Planning of Robotic Systems*. Cham, Switzerland: Springer, 2015, pp. 29–62.
[11] R. M. C. Santiago, A. L. De Ocampo, A. T. Ubando, A. A. Bandala, and E. P. Dadios, "Path planning for mobile robots using genetic algorithm and probabilistic roadmap," in *Proc. IEEE 9th Int. Conf. Humanoid, Nanotechnol., Inf. Technol., Commun. Control, Environ. Manage. (HNICEM)*, Dec. 2017, pp. 1–5.
[12] H. Mo and L. Xu, "Research of biogeography particle swarm optimization for robot path planning," *Neurocomputing*, vol. 148, pp. 91–99, Jan. 2015.
[13] Y. Z. Cong and S. G. Ponnambalam, "Mobile robot path planning using ant colony optimization," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatron. (AIM)*, Jul. 2009, pp. 851–856.
[14] Y. Zhang, D.-W. Gong, and J.-H. Zhang, "Robot path planning in uncertain environment using multi-objective particle swarm optimization," *Neurocomputing*, vol. 103, pp. 172–185, Mar. 2013.
[15] W. Zhang, X. Gong, G. Han, and Y. Zhao, "An improved ant colony algorithm for path planning in one scenic area with many spots," *IEEE Access*, vol. 5, pp. 13260–13269, 2017.
[16] Q. Yang and S.-J. Yoo, "Optimal UAV path planning: Sensing data acquisition over IoT sensor networks using multi-objective bio-inspired algorithms," *IEEE Access*, vol. 6, pp. 13671–13684, 2018.
[17] E. Masehian and D. Sedighizadeh, "A multi-objective PSO-based algorithm for robot path planning," in *Proc. IEEE Int. Conf. Ind. Technol. (ICIT)*, Mar. 2010, pp. 465–470.
[18] 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, 2011.
[19] P. Bhattacharjee, P. Rakshit, I. Goswami, A. Konar, and A. K. Nagar, "Multi-robot path-planning using artificial bee colony optimization algorithm," in *Proc. 3rd World Congr. Nature Biologically Inspired Comput. (NaBIC)*, Oct. 2011, pp. 219–224.
[20] Z. Wang, M. Li, L. Dou, Y. Li, Q. Zhao, and J. Li, "A novel multi-objective artificial bee colony algorithm for multi-robot path planning," in *Proc. IEEE Int. Conf. Inf. Automat.*, Aug. 2015, pp. 481–486.
[21] A. Hidalgo-Paniagua, M. A. Vega-Rodríguez, J. Ferruz, and N. Pavón, "Solving the multi-objective path planning problem in mobile robotics with a firefly-based approach," *Soft Comput.*, vol. 21, pp. 949–964, Feb. 2017.
[22] M. R. Panda, P. K. Das, S. K. Pradhan, and H. S. Behera, "An improved gravitational search algorithm and its performance analysis for multi-robot path planning," in *Proc. Int. Conf. Man Mach. Interfacing (MAMI)*, Dec. 2015, pp. 1–8.
[23] H. Hajimirsadeghi and C. Lucas, "A hybrid IWO/PSO algorithm for fast and global optimization," in *Proc. IEEE EUROCON*, May 2009, pp. 1964–1971.
[24] H. Jun and Z. Qingbao, "Multi-objective mobile robot path planning based on improved genetic algorithm," in *Proc. Int. Conf. Intell. Comput. Technol. Automat. (ICICTA)*, May 2010, pp. 752–756.
[25] F. Ahmed and K. Deb, "Multi-objective optimal path planning using elitist non-dominated sorting genetic algorithms," *Soft Comput.*, vol. 17, pp. 1283–1299, Jul. 2013.
[26] M. Davoodi, F. Panahi, A. Mohades, and S. N. Hashemi, "Multi-objective path planning in discrete space," *Appl. Soft Comput.*, vol. 13, pp. 709–720, Jan. 2013.
[27] S. Geetha, G. M. Chitra, and V. Jayalakshmi, "Multi objective mobile robot path planning based on hybrid algorithm," in *Proc. 3rd Int. Conf. Electron. Comput. Technol. (ICECT)*, Apr. 2011, pp. 251–255.
[28] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. 6th Int. Symp. Micro Mach. Hum. Sci. (MHS)*, Oct. 1995, pp. 39–43.
[29] Q. Bai, "Analysis of particle swarm optimization algorithm," *Comput. Inf. Sci.*, vol. 3, no. 1, p. 180, 2010.
[30] R. G. Bartle, *The Elements of Integration and Lebesgue Measure*. Hoboken, NJ, USA: Wiley, 2014.

SAHIB THABIT received the B.E. degree from the University of Mustansiriyah, Iraq, in 1994, and the M.S. degree from UTeM University Malaysia, in 2014. He is currently pursuing the Ph.D. degree with the Department of Mathematics and Computer Science, Amirkabir University of Technology. His research interests include multi-robot systems, computational geometry, motion planning, and heuristic algorithms.

ALI MOHADES is currently the Manager of the Computer Science Group, Faculty of Mathematics and Computer Science, Amirkabir University of Technology, Tehran, Iran. He is also the Dean of the Laboratory of Algorithms and Computational Geometry, Faculty of Mathematics and Computer Science, Amirkabir University of Technology. His main fields of interest are computational geometry, motion planning, and facility location.