Finding Robotic Wide Free Space Path in Dynamic Environment by Improving MAX–MIN Ant System Algorithm

https://doi.org/10.3991/ijim.v17i13.41513

Ali Hadi Hasan[1](✉), Hasanen S. Abdullah[2], Mohamad Ab. Saleh[3]
[1] College of Information Technology, University of Babylon, Babylon, Iraq
[2] Department of Computer Science, University of Technology, Baghdad, Iraq
[3] College of Economic and Administration, Al Iraqia University, Baghdad, Iraq
alihadi@itnet.uobabylon.edu.iq

Abstract—This paper proposes a new obstacle avoidance method for finding an optimal path passing through wide free space based on the optimized MAX-MIN Ant System (MMAS) algorithm in dynamic environments. The proposed improvement of the MMAS algorithm occurs in two stages. The first stage analyses the environment with two modifications by proposing a new relation of pheromone trail updating to construct the consequence modified (deposited) pheromone trail update in each iteration based on adding a new parameter (clean) to calculate the possible wide empty locations. Another alteration at this level involves checking and identifying the four horizontal and vertical states. Suppose the two nodes on either side of the four diagonal nodes are barriers in the next neighbor locations. In that case, the Tight Tunnel state is then attained, and it is managed by sealing this location to prevent passing through it before the robot movement stage. Another alteration is made to the robot-moving stage by recommending a new relation of tour building probabilities to determine the best option for moving the robot from the start node through a dynamic environment that incorporates dynamic obstacles moving through free space by locating and exhibiting the ideal path via broad free space. The main outcome of the experiment simulation of the new method involves that the robot avoids going through narrow tunnels and chooses the best course through open areas to reach its destination without running into any impediments in various dynamic environments conditions. In addition, the comparison of average time occupied to find the paths by the new method reduces to 54.32% as compared with that required with the method of hybrid the basic D* algorithm with the Lbest PSO algorithm, and to 11.95% in comparison to the time occupied recorded when applying the method of the improved MAX–MIN ACO algorithm.

Keywords—Max-Min ACO algorithm, mobile robot path planning, obstacle avoidance method, and dynamic environment

1 Introduction

One of the main attentions of the mobile robot is path planning nowadays. So must propose a path planning optimization method to calculate the best collision-free path from source to destination by avoiding static and dynamic obstacles. Off-line (global)

route planning is the process of determining a path when all of the environment's details, including the location of static obstacles and the robots' motion direction, are previously known. However, when the surroundings are unknown, the robot will move based on sensors that gather and transmit data as it saunters through the area. Online (local) route planning is the term used in this situation. Initially, online path planning operates in an offline form; however, after learning about fresh modifications to barrier circumstances, it switches to an online one [1, 2]. To improve path planning, the proper optimization method must be used [3-5]. Therefore, researchers have proposed many path planning methods concerning this problem, including the grid method [6], D* algorithm [7], and some intelligent methods, like particle swarm optimization (PSO) [8], Ant Colony Optimization (ACO) [9], artificial bee colony (ABC) [10], and Glowworm Swarm Optimization (GOS) [11]. To find safe ways with minimal energy consumption, Mansoor Davoodi et al. [12] proposed two multi-objective planning models. These models addressed the problem of two five- and three-objective optimization path plans with five goals and three goals. Each of them is useful because it allows you to determine the maximum error in your robot's sensors and actuators. In addition, they introduced a multi-target genetic algorithm for problem-solving and efficient exploration and development, and we suggested a group of engineering way refinery operators, which are very helpful in their demanding jobs. Finally, on a variety of workspaces, they evaluated the algorithms and models and contrasted them with the Pareto Evolutionary Algorithm 2 and multi-objective particle swarm optimization. According to [9, 13], a new robotic path planning in dynamic environments was present in a variety of complicated environment maps, it locates the closest and most effective path. Furthermore, the combination of optimization algorithms like hybrid MMAS with D* may obtain the best performance results. The improvement of MMSA algorithm to find the convergence solution by the equilibrium of the important characteristics represented in choosing the values α and β with the start (nest) and goal (food) locations and the number of static obstacles that have different shapes and sizes and are distributed randomly over the environment in order to determine a local robotic (ant) path. Jiang Zhao et al. [14] established a theoretical foundation for the enhanced ant colony algorithm by a precise mathematical derivation for the omnidirectional mobile vehicle's path planning. By enhancing the original pheromone's non-uniform distribution and the selection approach with direction, they significantly aid in the route search. In order to minimize repetitive searches and lessen the effect of ant numbers on algorithm performance, they implemented pheromone coverage and an updating technique. Furthermore, the convergence speed and searchability may be successfully balanced by splitting and changing the pheromone evaporation coefficient. Ali Hadi Hasan and Akmam Majid Mosa [15] Presented a technique to determine the best course for competitive and centralized multi-robots operating in the same dynamic environment. These robots are capable of starting at several places and going to the same destination. In order to create a trail of the modified (deposited) pheromone that is updated with each iteration, this approach combined the MAX-MIN ACO algorithm's pheromone trail updating with D* algorithm methods. The robots found and displayed the optimum route for each robot in the dynamic environment, which incorporates dynamic obstacles traveling cross open space. This was done using tour construction probabilities. The simulation of the experimental results showed that the robots compete with one another to reach their goals without running into obstacles or other robots, and they do it by finding the best path

with the fewest iterations and the lowest overall arc cost. In [16], addressing the issue of robot route planning in a static environment, a developed algorithm named (Tis) based on free parts and a turning point approach was presented. Their method targets two distinct objectives: the first is path integrity and length, and the second is to provide a reliable control code called Slip Mode Regulate to control the stability of a mobile, autonomous robot to trace the intended path. Researchers assert that the created technique is a good alternative to finding the right path and demonstrating the effectiveness of the suggested control code for reliable tracking of the mobile robot based on simulation results that have come to light. Employing the traits of the A algorithm and MAX-MIN Ant system. Xiaolin Dai et al. [17] suggested an improved ant colony algorithm. Furthermore, to increase search efficiency and path smoothness, the estimated function of the enhanced A algorithm is employed as a heuristic function. By presenting a new weighted adjacency matrix to identify the walking direction and rethinking the walking rules. Hong-Jun Wang et al. [18] proposed an improved ant colony algorithm for path planning. However, they claim that the present ant colony algorithms still have flaws, such as an inability to handle workspaces with narrow aisles and a significant amount of calculation. Zhen Yang et al. [19] suggested a bidirectional alternating jump point search to handle the path planning problem for dispersed multiple robots in dynamic situations. A* BAJPSA* algorithm works in two phases using an adaptive dynamic window approach. In the initial step, they used the BAJPSA* algorithm to plan each robot's globally optimum path, and simulation results showed how successful BAJPSA* is at planning global paths. Then, they executed the local path planning in the second step. The adaptive navigation method and route deviation assessment function are presented to improve the path-tracking skills of the standard DWA. They then split into groups, examined a number of unknown static and dynamic obstacle environments with motion conflict scenarios, and proposed dynamic obstacle avoidance rules for a single robot. After reviewing various motion conflict scenarios, combine the prioritized avoidance processes to accomplish cooperative multi-robot avoidance by expanding the single-robot to distributed multi-robot with decision rights. The simulation results demonstrate how effectively this algorithm plans multi-robot paths in unknown dynamic situations. The problem in this paper can be defined in terms of constructing a new obstacle avoidance method by finding an optimal path passing through wide free space (i.e. not to pass near and in contact with static obstacles) on dynamic environments based on developing tour construction probabilities equation in MAX-MIN Ant System (MMAS) algorithm by adding the new parameter. A method will also be created in the analysis environment stage to avoid the robot from going through narrow tunnel paths by closing entries of these paths in order to choose the best course through the open areas and reach its destination.

The main contributions made by this research project are listed below:

1. Creating a new class parameter called clean that is calculated by adding a clean weight constant to each neighbor location that has static obstacle status as one of the propagation information obtained in the robot environment analysis stage.
2. Creating a new method for checking the cases of tight tunnels for static obstacles in which the size of the entrance is the same as the size of the robot during the stage of analyzing the environment and closing it by converting the state of the tunnel entrance to a dynamic obstacle before the robot’s launching to find the optimal path.

3. Developing the equation of computing the tour construction probabilities by adding the new parameter clean $clean_{ij}(t)$ to the pheromone trail $\tau_{ij}(t)$ for exploring and finding a solution by the probabilistic decision for all the next states (nodes) of neighbor.

This paper is organized as follows: Section 2 addresses MAX–MIN Ant System, Section 3 includes designing a new obstacle avoidance method, while Section 4 shows the simulation results, and Section 5 concludes the paper.

2 MAX–MIN Ant system

ACO may perform better if the best solutions discovered during search and search space analysis are used more effectively. However, doing an interesting search might cause the search to stall prematurely. Therefore, by combining enhanced exploitation of the top solutions discovered during the search with an efficient mechanism, acquiring the highest performance of ACO algorithms is possible to avoid early search stagnation. As a result of these criteria, MAX-MIN Ant System differs from AS in the following ways:

1. During each iteration, just one ant adds a pheromone to exploit the best solution; this can be done either by the iteration-best ant or by the algorithm's running ant (global-best ant) [20-22].
2. Restricting the range of pheromone trails on each solution to a range [min; max] prevents stagnation.
3. The intentional setup of the pheromone to max at the start of the algorithm allows for a deeper degree of solution discovery [20].

2.1 Pheromone trail limits

It may not be possible to avoid search stagnation by selecting between the iteration-best and the global-best ant for the pheromone trail update. When the pheromone trail in one arc is higher than in the others, it could happen during optimum path design. An ant that prefers this solution will set it repeatedly in this scenario, considering the probabilistic decision made by Eq. (1) and further exploring the search space stops.

$$p_{ij}^{k}(t) = \frac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l\in N_i^k}[\tau_{il}(t)]^{\alpha}[\eta_{il}]^{\beta}} \quad if\ j \in N_i^k \tag{1}$$

Changing the probability options for the next solution is one technique to prevent stagnation. Heuristic data and pheromone trails can be used to do this. The latter is dependent on the issue at hand and is static during the algorithm's execution. To avoid the excessive buildup of pheromone throughout the algorithm's execution, the MMAS establishes limitations on the minimum and maximum amounts of pheromone (such that for all pheromone trails $\tau_{ij}(t)$, $\tau_{min} \leq \tau_{ij}(t) \leq \tau_{max}$). This will prevent the high discrepancy in pheromone trails. The limits imposed by the MMAS will be checked after each iteration. If we have $\tau_{ij}(t) > \tau_{max}$, we set $\tau_{ij}(t) = \tau_{max}$; analogously, $\tau_{ij}(t) < \tau_{min}$, we set $\tau_{ij}(t) = \tau_{min}$. Also, note that enforcing $\tau_{min} > 0$, and if $\eta_{ij} < \infty$

for all solution components, any solution component may probably be chosen. The MMAS convergence happens for every decision point when all solution components have pheromone trails min, with the exception of one solution having $\tau_{\max}$. In this scenario, the algorithm's best solution will be created by selecting max pheromone, and it will be created with a probability p best that is higher than 0. In order to build the optimal solution at each point, an ant should thus select the solution component using $\tau_{\max}$. Naturally, the values of max and min directly impact the likelihood of choosing a solution component at any given decision point [20].

2.2 Updated the pheromone trail

In MMAS, the pheromone trails are refreshed after each iteration using just one ant. The following rule provides the updated pheromone trail update:

$$\tau_{ij}(t+1) = \rho\, \tau_{ij}(t) + \Delta\tau_{ij}^{best} \tag{2}$$

Where $\Delta\tau_{ij}^{best} = 1/f(s^{best})$ and $f(s^{best})$ denotes the solution cost of either the iteration-best (s^{ib}) or the global-best solution (s^{gb}). Although sib has been employed in some comparative studies, the use of a single ant in a pheromone trail has also been discussed in ACS. MMAS focuses on the use of iteration-best solutions. The method of utilizing the search in MMAS will be determined by selecting just one solution (either sib or S^{gb}) to be employed in the pheromone update. If S^{gb} is the sole color used, it will be heavily strengthened. The search will be narrowed to this answer, decreasing the possibility of employing alternative, perhaps superior, solutions. This raises the prospect of abusing subpar solutions. When sib is used for trail updates, the scenario is different since the optimum options vary from iteration to iteration. As a result, a greater number of solution components might get occasional reinforcement. The most effective method, however, will be the combination of local search with MMAS to identify larger optimal pathways, which is illustrated by using sib as the default solution for updating the pheromones and sgb for every set number of repetitions. The most successful strategy appears to be using a mixed dynamical technique that increases the frequency of using sgb for the pheromone update during the search [20].

2.3 Initialized the pheromone trail

Initializing the pheromone trail involves assigning t(0) to an arbitrarily high number, which causes all trails to conform to $\tau_{min}(1)$. During the algorithm's initial iteration, this form of initialization increases the exploration of potential solutions. Due to trail evaporation, there will be a ratio-based difference in pheromone trails after the first repeat, a 2-factor difference after the second, etc. (specified by parameter). The ratio of τ_{min}, to the quantity of pheromone deposited on a solution element is (1)(avg p dec) /(1p dec). This ratio is much greater than the relative difference among the pheromone trail when it is initialized $\tau_{\max}$, depending on the empirical parameters that are selected. The selection probabilities of Eq. (1) will gradually increase when the pheromone trail is initiated at $\tau_{\max}$. Therefore, investigating potential solutions is preferred. The experimental findings support the hypothesis that a bigger search space exploration enhances MMAS's performance as a result of setting (1) = $\tau_{\max}$ [20].

3 Designing a new obstacle avoidance method

This section presents a design of new obstacle avoidance method and finding an optimal wide path. The design structure consists of the three main parts the first one is Determine and constructed the Robot Environment which out of MMAS algorithm and remain two in MMAS algorithm Analysis Robot Environment and Moving a Robot in dynamic environment as shown in Figure 1.

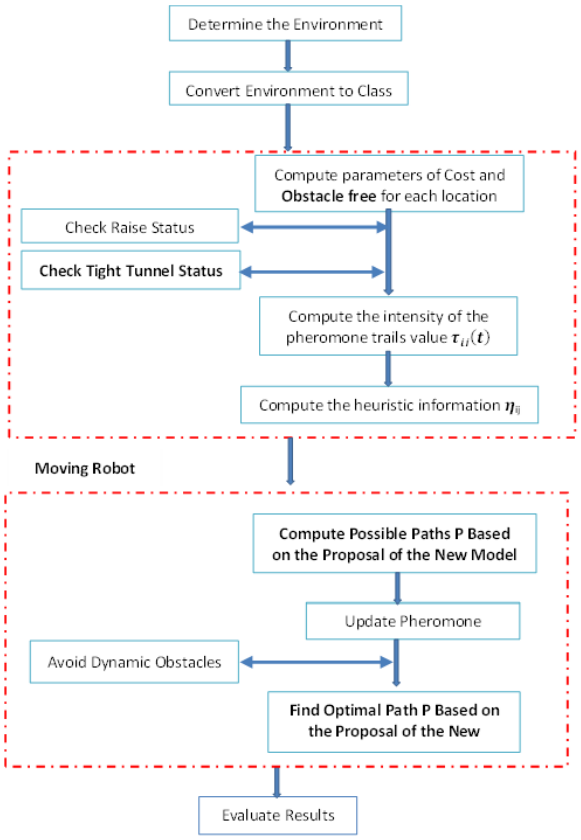

Fig. 1. The block diagram structure of the proposed new obstacle avoidance method to find an optimal wide path

3.1 Determine and construct robot environment

In the First section, choose a 2D workspace environment with fixed obstacles represented by regularity grids from a library store. The beginning component is divided into two stages. Before creating the class node for the arrays, decide the robot's starting location and one objective.

3.2 Robot environment analysis

The second section includes many combined phases for the robot environment analysis. The D* algorithm method, which estimates the route costs of the states in the space, is hybridized with the pheromone trail update based on iteration-best to retain the propagated information about the intensity of the pheromone trails values $\tau_{ij}(t)$ values. Pheromone deposition repeatedly occurs, which allows for proliferation. The path width is computed by adding the clean weight constant to each neighboring node that is not in the way after each iteration, which expands to modify the pheromone trails values $\tau_{ij}(t)$ to its neighbors and change the path width. To continue the propagation process and change the pheromone trail values $\tau_{ij}(t)$, these neighbors add ants one at a time until the start states for each robot are discovered. The robot (ant) goes across free space nodes with broad locations and a minimal number of obstacles in the third section, avoiding the dynamic obstacles until it reaches the objective state based

on tour construction probability.Pseudocode of algorithm Pheromone Trail Updating illustrated in algorithm 1 contains several steps, some in bold, that were added to satisfy the proposed improvement to compute the pheromone trail update Eq. (2).

```
  Algorithm 1: Pheromone Trail Updating
  Input: Start and Goal nodes, queue list OpenList, array
of class node State and array of class color NodeColor.
  Output: array of class node State, array of class color
NodeColor.
  1: Begin
  2: Goal_i = goal_pos_x
  3: Goal_j = goal_pos_y
  4: ρ = 0.98            // select the initial value of ρ
  5: CleanWeigh = 0.5 // select Clean Weight constant
  6: goal_pher=0.0
  7: Insert(OpenList, goal); // add ant to queue propaga-
tion
  8:  repeat
  9:   Begin
  10:   node X = DequeueOpenList (); //moving ant to
propagation
  11:   X_tag = "Close"
  12:// expanded by finding the eight neighbor locations
Y of current position X
  13:   FindNeighbor (Y,X)
  14:   for each Neighbor Y of X do
  15:    begin
  16:     If (Y.status == "OBSTACLE") then X.clean =
X.clean + CleanWeigh
  17:        PheromoneTrails (X ,Y, OpenList)
  18:        GateRaiseState (X)
  19:        CheckTightTunnelState(X)
  20:        SortQueue (OpenList)
  21:     end
```

```
22:  until (X_status = "Start")
23: end
```

By choosing an initial evaporation rate value of 0.98, the step in line 4 is completed. The new step in line 5 is to determine the clean weight constant equal to 0.5, which we found through tests is the best suitable value that can work with different environments. Then, in step line 7, the goal state (food location) is put in the queue as a current state (location) used (Insert procedure) to add ants propagation. Finally, by locating the eight neighboring states (locations), the existing state (step line 13) is increased (Find Neighbor procedure). The step in line 16 computed and saved a value of a new class parameter called clean by adding a clean weigh constant to each neighbor location Y has obstacle status to explore wide free spaces (empty-obstacles) of the current node. The step in line 17 procedure, Pheromone Trails computed the intensity of the pheromone trails value τij (t) and the heuristic information η_{ij} for each state on neighbors. When the status is an obstacle, the neighbors are verified; no pheromone trail values are supplied (unlike D*, which is given a high value of 100000). When the status is cleared, check the position if the object is moving on the diagonal line of the current status (position), multiply the value of τij (t) by 1.4, and multiply by 1 if the object is moving horizontally or vertically. After that, the evaporation rate doubles for each instance. Then, the evaporation rate is doubled with each instance. Additionally, the distance between any neighbor sites and the objective position is used to determine the value of the heuristic information η_{ij} (food). Line 18 contains a function called GateRaiseState used to examine and change gate raise states. If two diagonal vertex neighbor sites of the ant's present position are in an obstruction condition, the ant cannot travel to those places since its size has not passed through them. As a result, the present state is a gate-raising state, and to shut it down, its status type must be changed to obstacle. The new step in line 19, shown in algorithm 2 pseudocode performs the Checking of Tight Tunnel state, which consists of several steps that check and manipulate the four horizontal and vertical states in which the next-neighbor location is the Tight Tunnel state. The steps from line 2 to line 7 Test to select the positions of the four horizontal and vertical nodes of the current node in neighbor locations Y in two arrays Mi and Mj. Steps 8 and 9 Test to select the positions of four neighbor diagonal nodes from Y to the position of current node X to save the status of the four diagonal nodes of the current node in array A in a sequence dependent on the specified conditions as shown in line 11 and line 12 also saving the status of the four diagonal nodes in array B. After specifying and saving all the positions of the four horizontal and vertical nodes in both arrays Mi and Mj and specifying all the positions of all four adjacent diagonal nodes and saving their status in both arrays A and B.

Algorithm 2: Check Tight Tunnel State

```
Input: array of class node State, array of class color
Node Color.
Output: array of class node State, array of class color
Node Color.
1: begin
2:  for each neighbor Y of X do
```

```
  3:   if ( Y horizontal nodes of X ) or ( Y vertical
nodes of X ) then
  4:    begin
  5:      Mi [] = Y_iold;
  6:      Mj [] = Y_jold;
  7:    end
  8:  for each neighbor Y of X do
  9:    if ( Y diagonal nodes of X ) then
  10:    begin
  11:      A [] = Y_status;
  12:      B [] = Y_status;
  13:     end
  14:  for  each horizontal or vertical nodes Y of X do
  15:   if (Y_row=Mi[] and Y_col=Mj[]) and (A[]="Obsta-
cle" and B[]=" Obstacle") then
  16:    begin
  17:      setpixel (Y, Orange);
  18:      Y_k = 10000; // it is represent a very large
value
  19:      Y_status = " Obstacle ";
  20:    end
  21: end
```

After specifying and saving all the coordinates of the positions of the four horizontal and vertical nodes in each of the single arrays Mi and Mj, and determining all the positions of all the four adjacent diagonal nodes and saving their states in both single arrays A and B. In steps 14 and 15, each of the four horizontal and vertical nodes is tested to determine if the statuses of the two nodes on both sides from the four diagonal nodes are obstacles. Once this condition is met, steps 17, 18, and 19 are modified by turning the color orange, increasing the cost value such that the pheromone value is too low, changing the node status to the obstacle, and closing this site.

3.3 Moving robot

The robot (ant) goes to the next place in this part by selecting the optimal option based on the likelihood of the tour's construction at each iteration. Until it achieves the desired condition, it chooses to go via a broad free cell (location) and avoid dynamic obstacles. The process of pseudocode, as in algorithm 3 **Move Robot Tour Construction** illustrates a process for discovering tour construction.

```
  Algorithm 3: Move Robot Tour Construction
  Input: array of class node State, an array of class color
Node Color.
  Output: Display Path color, Time occupies, No. of Iter-
ations and No. of cost.
  1: begin
  2:  X_curri = start_pos_x
```

```
  3: X_currj = start_pos_y
  4: α =3;  β =1;
  5: PathCost=0;
  6: FindNeighbor (Y,X)
  7: CrateDynObs(X1,temporal_k)
  8:  repeat
  9:   begin
  10:  UpdateDynObs(X1,temporal_k)
  11:     for each neighbor Y of X do
  12:       if (Y_status  != "Obstacle"  or  Y_status  !=
"Dobstacle" ) then
  13:         P[i] = ((state[Y].pherm+ state[Y]. clean)^α
* (state[Y].hur)^β)/ ∑((state[Y].pherm+ state[Y]. clean)^α
* (state[Y].hur) ^β)
  14:           for each item of (P[]) do
  15:               if (P[] < min_best_pphr) then
  16:           begin
  17:             min_ min_best_pphr = P[]
  18:             mini_ pphr = neighbori[]
  19:             minj_ pphr = neighborj[]
  20:         end
  21: X_corri = mini_ pphr;
  22: X_corrj = minj_ pphr;
  23: iter_no = iter_no + 1;
  24: EndTime = TimeNow();
  25: PathCost =PthCost +X. k;
  26: X_SetPixel (Green);
  27:    end
  28: TimeOcc = EndTime – StateTime,
  29:  until (current_position = goal_position or iter_no
≥ max_itr)
  30: end
```

The start state (nest) is chosen as the first current node (location) in lines 2 and 3 for moving the ants. Then, give the proper weight values in line 4 with the pheromone and heuristic values taken into consideration. Also give initial vale of cost in line 5. The process is in line 6 to travel from the current node (position) X and use **Find Neighbor** to locate the eight nearby nodes (locations). Then, using a process, a random position was chosen at line 7 in a free-space environment to create a dynamic obstacle. **CrateDynObs.** The robot (an ant) is shown moving from lines 8 to 27 according to a probabilistic judgment to the next position in the neighborhood that provides the optimal answer. Each time convergence occurs (iteration-best ant), the process is repeated until the robot achieves the desired location (food). The **Update Dynobs** procedure on line 10 displays dynamic obstacles moving to random positions in the adjacent free-space environment. After that, the steps from lines 11 to 12 are used for testing all free space locations (nodes) of neighbors. Then in line 13, computing the tour construction probabilities Eq. (1) is a development by adding a new parameter clean $clean_{ij}(t)$ which

computed in the previous analysis stage to the pheromone trail $\tau_{ij}(t)$ and on locally available heuristic information η_{ij} for all locations (nodes) of neighbor as shown in Eq. (3).

$$p_{ij}^{k}(t) = \frac{[\tau_{ij}(t)+clean_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l\in N_i^k}[\tau_{il}(t)+clean_{il}(t)]^{\alpha}[\eta_{il}]^{\beta}} \quad if\ j \in N_i^k \tag{3}$$

In comparison to other ants in nearby sites, the steps from line 14 to line 20 demonstrate the choice of the optimal option for the ant. The best ant moves to a new location to become a current node, and the convergence process is repeated until the best ant reaches the goal position (food). The last stages from lines 21 to 28 depict the outcome of the optimal path as the path appears on the chosen dynamic environment as a green color, the number of iterations (locations) of the path, the total cost of the path, and the amount of time required discovering a path.

4 Simulation results

To demonstrate the viability of using the new technique, various simulations in different situations are shown in this section. This variety differs in terms of the size of the dynamic environments, the number and types of static obstacles, the number of tight tunnel states closed off during analysis before finding an optimal path, the number of gates raised during analysis before finding an optimal path, the number of positions of the dynamic obstacles selected at random in the invalid (free) space of the environment, and the number of the dynamic obstacles moving continuously in random variant neighboring situations. The Pentium CORE i7 processor, Windows 10, and Microsoft Visual Studio 2019 were used to implement these simulation findings.

4.1 Implemented of a new obstacle avoidance method

The results of the new obstacle avoidance method for finding an optimal wide path based on the improved MMAS algorithm are compared with the results of the paper [7], which are both shown as a green color in Figure 2 and are tested and implemented. The results are found and studied with the same small environment of size (10*10) positions. Select the start state at position (0,9) and the target state at position (7,0), providing the value weigh of pheromone α equal to 2 and the value weigh of heuristic β equal to 1, in a random placement of the dynamic obstacle of blue color. The first test is the proposed new method for avoiding the obstacles presented in Figure.2-a, which leads to different results in the locations of the optimal path far from the static obstacles and the number of iterations is 14, the total path cost is 106.2 and the occupied time is 1.55 seconds. The second testing case illustrated in Figure 2-b is implemented to find the optimal path of the paper [9]. The findings were 13 iterations, 99 total costs along the path, and 1.40 seconds of time spent. One of the most crucial parameters in the MMAS algorithm for determining the best solution is the weight value of the pheromone α. To obtain the best outcomes, several values in various environment types are being tested. The first experiment's findings are illustrated in Figure 3-a, where changing weight value of the pheromone α to 3 yields the best outcomes and creates a broad

empty path that is smoother and more ideal, with a total path cost of 105.8 and a time duration of 1.53 seconds. When the dynamic obstacle intersects with the robot (ant) in the same location, the second test is carried out. In this case, the robot (ant) moves away from the dynamic obstacle to the ant's optimal neighbor location with the minimum probability of contact. Figure.3-b shows that the local way depicted in green is different from the path in Figure 3-a without the intersecting dynamic barrier, and that the overall path cost changes to 106.2, with a time occupy equal to 1.51 sec.

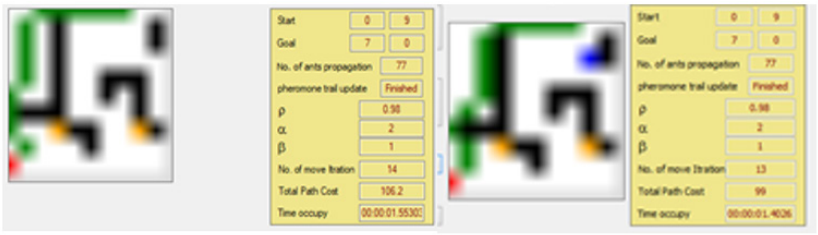

Fig. 2. Comparison of the results of the new obstacle avoidance method to find path planning with the paper [7] on the small environment

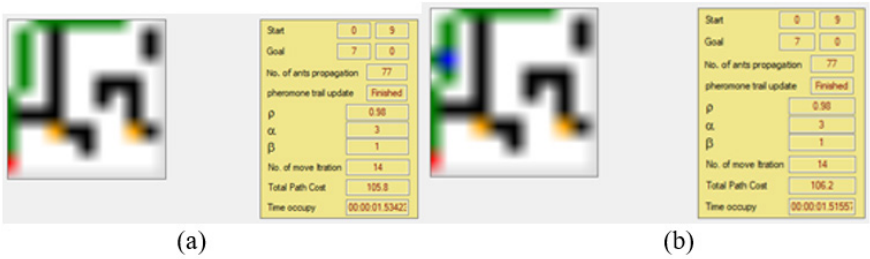

Fig. 3. Find the best results of new obstacle avoidance method when avoiding dynamic obstacle

4.2 Experiment results

In order to determine the outcomes of local route planning on six distinct complicated dynamic environments of size 50*50 robot locations, experimental simulation is finally implemented on the new obstacle avoidance approach for finding an appropriate broad path based on an enhanced MMAS algorithm. These simulation experiment results were obtained by selecting different start and target positions, the same evaporation rate ρ value of 0.98, different pheromone weigh values α ranging from 3 to 5, and the same weigh the value of heuristic β, which is equal to 1. Figure 4 displays some simulation experiments together with these six dynamic parameters. The experiment simulation result is illustrated in Figure 4-a. The bold color line at (Table 1) is implemented in environment 1, whose characteristics include large numbers of fixed obstacles of different forms and sizes located randomly. In the middle, there is a continuous horizontal series of obstacles, the number of gates raises state and a large number of Tight Tunnel state. The result of the proposed method is found in the optimal wide empty path with No. of gate raise state that equals to 3, No. of Tight Tunnel state that

equals 101, No. of iterations that equal 103, total path cost equals 6124.4 and time occupied is 5.92 seconds shown in green color based on giving the parameters selected the start location at (3,2) and goal location at (2,48), gives the value of evaporation rate ρ that equals 0.98, gives the weigh value of pheromone α equals to 5 and the weigh value of heuristic β that equals 1.

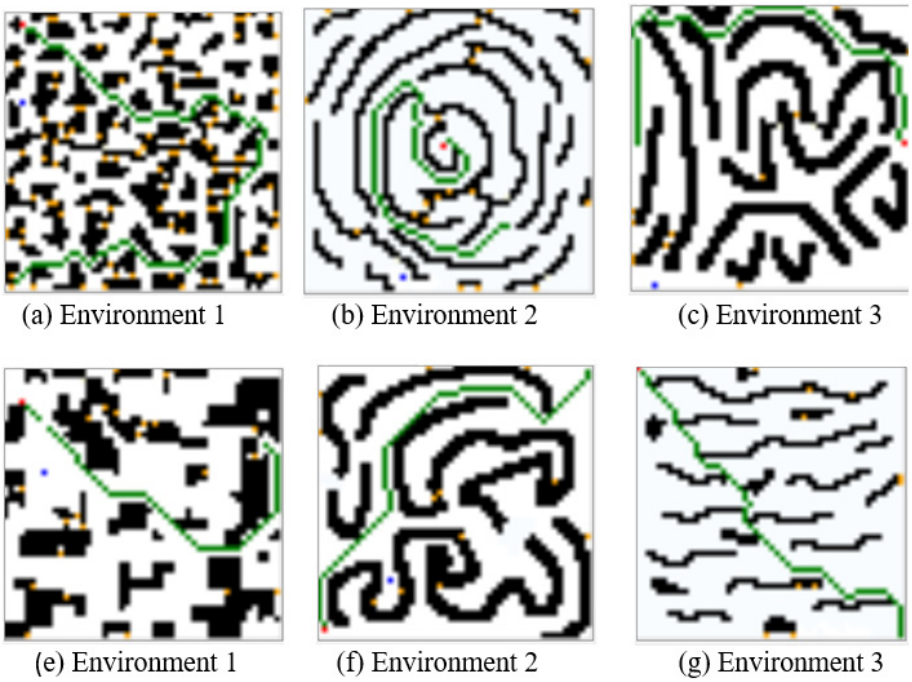

Fig. 4. Elected some experiment simulation results of the new obstacle avoidance method on six 50x50 different complex dynamic environments

A distinctive note when testing and implementing experiments simulation on this environment are that it gives the weigh value of the pheromone α more than 5 and gives weigh the value of heuristic β equals 1 to increase and maximize the value of the pheromone trail $\tau_{ij}(t)$ add with the new parameter clean $clean_{ij}(t)$ compared to the value of the local heuristic information η_{ij} for all locations (nodes) of neighbor required to compute the tour construction probabilities Eq. (3) to avoid the local solution loop. Figure 4-b shows the result of implemented experiment simulation on environment 2, whose characteristic specialty is that the static obstacles are established in circular shapes overlapping each other, and some circular block obstacles are open from different positions. Therefore, it contains several gates that lie on some circular block obstacle, and a number of tight tunnel state lies on variant positions. The result of the proposed method, illustrated in bold color line (Table 1), is found in the optimal wide empty path with 3 gate raise state, 21 Tight Tunnel state, No. of iterations that equal 73, total path cost equals 2988.2 and time occupied is 4.28 second shown in green color based on giving the parameters selected the start location at (24,24) and goal location at (35,38), gives the value of evaporation rate ρ that equals 0.98, gives the weigh value

of pheromone α equals to 3 and the weigh value of heuristic β that equals 0. A distinctive note when testing and implementing experiments simulation on this environment gives weigh the value of heuristic β equals 0 and disregard local heuristic information η_{ij} . Computing the tour construction probabilities Eq. (3) depended only on the pheromone trail $\tau_{ij}(t)$ add with the new parameter clean $clean_{ij}(t)$ all locations (nodes) of neighbors require avoiding the local solution loop. The special characteristic of environment 3 is that static obstacles are organized in curve shapes, and some of these curve block obstacles are open from different directions. Therefore, the experiment simulation is implemented on it, and the result of the proposed method is to find the optimal wide empty path without gate raise state, 11 Tight Tunnel state, No. of iterations that equals 80, total path cost equals 3452 and time occupied is 4.37 seconds, shown in green color illustrated in Figure 4-c and bold color line at (Table 1). Furthermore, these results are found based on giving the parameters selected the start location at (48, 24) and goal location at (1,24), gives the value of evaporation rateρ that equals 0.98, gives the weigh value of pheromone α equals to 3 and the weigh value of heuristic β that equals 1.

The experiment simulation is implemented in environment 4, where both static obstacles and free space are organized in big blocks containing hole-free space opens down, and one gate lies at the above position. The result of the proposed method is found in the optimal wide empty path shown in green color illustrated in Figure 4-d, and the bold color line in Table 1 are 2 gate raise state, 22 Tight Tunnel state, No. of iterations that equal 59, total path cost equals 2073.2 and time occupied is 3.66 seconds based on giving the parameters selected the start location at (3,6) and goal location at (47,14), gives the value of evaporation rateρ that equals 0.98, gives the weigh value of pheromone α equals to 3 and the weigh value of heuristic β that equals 1. The special characteristic of environment 5 is that the static obstacles are organized in curve shapes, and some of these curve block obstacles are open from different directions. Therefore, the experiment simulation is implemented on it, and the result of the proposed method is to find the optimal wide empty path without gate raise state, 12 Tight Tunnel state, No. of iterations that equal 71, total path cost equals 3069.2 and time occupied is 4.10 seconds, shown in green color illustrated in Figure 4-e and bold color line at (Table 1). These results are found based on giving the parameters selected the start location at (1, 48) and goal location at (49, 1), gives the value of evaporation rate ρ that equals 0.98, gives the weigh value of pheromone α equals to 3 and the weigh value of heuristic β that equals 1. The experiment simulation is implemented on last environment 6, whose characteristics specialty is that the static obstacles are organized in horizontal sequence obstacles, and some of these obstacles are open at different positions. The experiment simulation is implemented on it, and the result of the proposed method is to find the optimal wide empty path without gate raise state, 10 Tight Tunnel state, No. of iterations that equals 64, total path cost equals 2442.8 and time occupied is 3.69 seconds, shown in green color illustrated in Figure 4-f and bold color line at (Table 1). These results are found based on giving the parameters selected the start location at (0, 0) and goal location at (49, 49), gives the value of evaporation rateρ that equals 0.98, gives the weigh value of pheromone α equals to 3 and the weigh value of heuristic β that equals 1.

Table 1 illustrates the comparison between the overall results of the proposed method and the results of the method of paper [7] hybrid of basic D* with Lbest PSO algorithms and the method of paper [6] improved MAX–MIN ACO algorithm on the same six dynamic environments shown in Figure 4 and on the same requirements of the start and goal positions, the value of evaporation rate ρ, the value of pheromone weigh α and the value of heuristic weigh β. Also, the table clarifies the results numbers of gate raise state, number of Tight Tunnel state of the proposed method, number of iterations, time occupied to find the path by second and the total path cost.

To analyze and discuss the overall results obtained from the implementation of the proposed method and other comparison methods given in (Table 1). The first results are to find the numbers of gate raise state observed that have the same numbers to each experiment simulation implemented by all comparison methods. The second results are the calculations of the tight tunnel case numbers found by the proposed method only, and the average numbers of Tight Tunnel states for each of the six dynamic environments are calculated respectively 83.6, 16.8, 11.8, 17.6, 8.4 and 7.4. We noticed that environment 1 have a high number of Tight Tunnel state, and the high number equal 101 states, as shown in Figure 4-a, while environment 6 has fewer numbers Tight Tunnel state because of the special characteristic shape of each environment. In the third comparison of the results of the average number of iterations, the method hybridizing the radix D* with the Lbest PSO algorithm and the improved MAX-MIN-ACO algorithm is to search broad and free paths, we find that the average total path cost results of the proposed method increase by 9.99% and 11.35%, respectively of obstacles. The last comparison of the results is the average time occupied to the found path by second found that the proposed method reduced the time by 54.32% and 11.95% with respect to the method of hybrid of basic D* with Lbest PSO algorithm and method of improved MAX–MIN ACO algorithm respectively.

Table 1. shows the comparison experiment simulation results of the proposed new method and the two methods on the same six dynamic environments shown in Figure 1

Envt. No.	Algorithm Name	Start State	Goal State	No. of Gate	No. of Tunnel	No. of Iteration	Total Arc Cost	Time Occupy
Environment 1	Hybrid Basic D* with Lbest PSO [8]	37, 8	5, 38	3	-	60	2198.2	7.87
		45, 6	19, 41	3	-	50	1590.4	6.71
		0, 42	6, 2	3	-	88	4809.4	13.45
		3,2	2, 48	3	-	89	4935	13.40
		21, 14	21, 37	3	-	53	1736.6	7.43
	Improved MAX–MIN ACO [9]	37, 8	5, 38	3	-	61	2139	4.43
		45, 6	19, 41	3	-	52	1634.8	3.92
		0, 42	6, 2	3	-	90	4891.6	5.64
		3,2	2, 48	3	-	91	4999.4	6.89
		21, 14	21, 37	3	-	54	1737	4.33
	proposed Method	37, 8	5, 38	3	66	73	3031.8	4.17
		45, 6	19, 41	3	67	60	2103	2.64
		0, 42	6, 2	3	100	103	6400.4	5.12
		3,2	2, 48	3	101	103	6124.4	5.92
		21, 14	21, 37	3	84	68	2660.6	3.34

Envt. No.	Algorithm Name	Start State	Goal State	No. of Gate	No. of Tunnel	No. of Iteration	Total Arc Cost	Time Occupy
Environment 2	Hybrid Basic D* with Lbest PSO [8]	24, 24	35, 38	3	-	56	1903	8.51
		34, 38	25, 13	2	-	48	1358.2	6.18
		18, 21	32, 42	2	-	47	1305.6	6.59
		33, 18	25, 35	1	-	53	1679.8	8.25
		42, 3	0, 37	0	-	58	2020.8	6.78
	Improved MAX–MIN ACO [9]	24, 24	35, 38	3	-	57	1902.6	4.70
		34, 38	25, 13	2	-	49	1357.4	2.28
		18, 21	32, 42	2	-	48	1310.4	3.99
		33, 18	25, 35	1	-	55	1767.2	4.10
		42, 3	0, 37	0	-	58	1962.2	4.26
	proposed method	24, 24	35, 38	3	21	73	2988.2	4.28
		34, 38	25, 13	2	14	49	1367.4	2.12
		18, 21	32, 42	2	16	48	1315	3.25
		33, 18	25, 35	1	13	54	1685.8	3.99
		42, 3	0, 37	0	20	80	3761.2	4.71
Environment 3	Hybrid Basic D* with Lbest PSO [8]	28, 7	26, 26	2	-	42	1038.4	7.01
		14, 48	33, 16	1	-	53	1669.4	8.03
		43, 47	23, 15	1	-	46	1273.2	7.78
		8, 0	48, 46	0	-	61	2303.4	9.90
		48, 24	1,24	0	-	71	2877.2	10.37
	Improved MAX–MIN ACO [9]	28, 7	26, 26	2	-	44	1106.8	3.65
		14, 48	33, 16	1	-	52	1505.6	3.40
		43, 47	23, 15	1	-	47	1254	3.96
		8, 0	48, 46	0	-	61	2264	4.26
		48, 24	1,24	0	-	71	2804.4	5.63
	proposed method	28, 7	26, 26	2	15	43	1047.6	3.36
		14, 48	33, 16	1	10	53	1628.2	3.74
		43, 47	23, 15	1	6	47	1296.8	3.10
		8, 0	48, 46	0	17	62	2295	5.56
		48, 24	1,24	0	11	80	3452	4.37
Environment 4	Hybrid Basic D* with Lbest PSO [8]	3, 6	47, 14	2	-	63	2414.8	9.82
		25, 22	47, 14	1	-	47	1255.2	5.40
		41, 1	2, 3	1	-	54	1701.2	7.32
		5,49	18,0	0	-	70	3050.4	10.76
		27, 1	16, 48	0	-	66	2480.8	10.23
	Improved MAX–MIN ACO [9]	3, 6	47, 14	2	-	61	2217	2.9
		25, 22	47, 14	1	-	37	803.8	2.4
		41, 1	2, 3	1	-	45	4282.6	3.54
		5,49	18,0	0	-	55	1796.8	3.31
		27, 1	16, 48	0	-	46	1155.8	3.20
	Proposed method	3, 6	47, 14	2	22	59	2073.2	3.66
		25, 22	47, 14	1	4	37	797.8	2.00
		41, 1	2, 3	1	18	51	1597.6	3.38

Envt. No.	Algorithm Name	Start State	Goal State	No. of Gate	No. of Tunnel	No. of Iteration	Total Arc Cost	Time Occupy
		5,49	18,0	0	23	64	2523	5.48
		27, 1	16, 48	0	21	48	1453.4	3.54
Environment 5	Hybrid Basic D* with Lbest PSO [8]	22, 13	27, 17	1	-	63	2414.8	9.82
		33, 36	22, 12	1	-	47	1255.2	5.40
		11, 42	31, 8	1	-	54	1701.2	7.32
		1, 48	49, 1	0	-	70	3050.4	10.76
		26, 4	23, 48	0	-	66	2480.8	10.23
	Improved MAX–MIN ACO [9]	22, 13	27, 17	1	-	68	2354	5.07
		33, 36	22, 12	1	-	48	1255.2	3.42
		11, 42	31, 8	1	-	55	1700.8	4.86
		1, 48	49, 1	0	-	71	3050.4	5.17
		26, 4	23, 48	0	-	67	2480.4	7.34
	Proposed method	22, 13	27, 17	1	11	67	2507.6	4.34
		33, 36	22, 12	1	3	51	1405	2.84
		11, 42	31, 8	1	7	54	1650	3.52
		1, 48	49, 1	0	12	71	3069.2	4.10
		26, 4	23, 48	0	9	67	2495.6	4.37
Environment 6	Hybrid Basic D* with Lbest PSO [8]	48, 49	1, 49	0	-	46	1128	4.62
		0, 49	0, 3	0	-	45	1081	4.98
		0, 0	49, 49	0	-	63	2436.8	8.67
		0, 49	49, 0	0	-	63	2421.2	8.92
		39, 2	15, 37	0	-	53	1762.6	8.59
	Improved MAX–MIN ACO [9]	48, 49	1, 49	0	-	47	1128	4.18
		0, 49	0, 3	0	-	46	1081	4.21
		0, 0	49, 49	0	-	64	2435.6	4.67
		0, 49	49, 0	0	-	64	2419.6	5.41
		39, 2	15, 37	0	-	54	1762.6	5.15
	Proposed method	48, 49	1, 49	0	4	47	1182	4.40
		0, 49	0, 3	0	4	46	1081	2.76
		0, 0	49, 49	0	10	64	2442.8	3.69
		0, 49	49, 0	0	10	64	2431.2	3.38
		39, 2	15, 37	0	9	54	1796	3.57

5 Conclusions

This paper presents a new obstacle avoidance method for finding an efficient path passing through wide free space based on developing the MMAS algorithm in variant complex environment maps. Through the completion of the work, it was concluded that the improvement in the MMSA algorithm occurred in two parts of the algorithm construction. In the first part, which is analyzing the environment of the robot, the new parameter clean weight constant value equal to 0.5 is determined, and through tests, we have found this is the best suitable value that can work with different environments. Then the value of a new class parameter called clean $clean_{ij}(t)$ for the current node,

X is calculated, and Y adds a clean weighting constant to each neighboring position with obstacle status to avoid wide open spaces is stored by allowing (empty obstacles) to explore the current node. Also, in this part, we create a new narrow tunnel state checking algorithm for any of the four horizontal and vertical nodes, and check whether the state of the two nodes on either side of the four diagonal nodes is an obstacle to judge Then four states, horizontal and vertical, are manipulated to change the color of the closest location to orange, which is very expensive. This means that the pheromone value gets too low and the node's status is changed to Obstacle and this location is closed. While in the second part, which is the movement of the robot, we developed a new equation to compute the tour construction probabilities by adding the new parameter clean $clean_{ij}(t)$ which computed in the previous analysis stage to the pheromone trail $\tau_{ij}(t)$ as shown in Eq. (3). From the implementation and testing of simulation experiments of the proposed new method on different dynamic environments, there are several issues that can be concluded, such as in environment 1, which includes large numbers of fixed obstacles of different shapes and sizes located randomly are must give the weigh value of pheromone α more than 5. Whereas in environments like environment 2 that include static obstacles are established in circular shapes overlapping each other, and some circular block obstacles are open from different positions. In this situation, it must give weigh the value of the heuristic β equals 0 and disregard local heuristic information η_{ij}. Also, it is seen that environment 1 have a high number of Tight Tunnel state and has a high number equal to 101 states while the environment 6 has less numbers Tight Tunnel state because of the special characteristic shape of each environment. Compared with the results of the method of the hybrid of basic D* algorithm with Lbest PSO algorithm and the method of improved MAX–MIN ACO algorithm, the new method can find an efficient path passing through wide free space in the shortest average time occupied in second and reduced the time by 54.32% and 11.95% among the compared methods respectively. While the average number of the total path costs of the proposed method is increased by 9.99% and 11.35%, respectively and also the average number of iterations of the proposed method increased by 5.44% and 7.10%, respectively.

6 References

[1] P. Raja and S. Pugazhenthi, "Optimal path planning of mobile robots: A review," *International journal of physical sciences,* vol. 7, no. 9, pp. 1314-1320, 2012. https://doi.org/10.5897/IJPS11.1745

[2] M. Kadhim, "Secured Transfer and Storage Image Data for Cloud Communications," *international Journal of Online and Biomedical Engineering,* vol. 19, no. 06, 2023. https://doi.org/10.3991/ijoe.v19i06.37587

[3] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research,* vol. 30, no. 7, pp. 846-894, 2011. https://doi.org/10.1177/0278364911406761

[4] C. Goerzen, Z. Kong, and B. Mettler, "A survey of motion planning algorithms from the perspective of autonomous UAV guidance," *Journal of Intelligent and Robotic Systems,* vol. 57, pp. 65-100, 2010. https://doi.org/10.1007/s10846-009-9383-1

[5] H. T. Hazim, "Secure Chaos of 5G Wireless Communication System Based on IOT Applications," *International Journal of Online & Biomedical Engineering,* vol. 18, no. 12, 2022. https://doi.org/10.3991/ijoe.v18i12.33817

[6] G. Tanzmeister, M. Friedl, D. Wollherr, and M. Buss, "Efficient evaluation of collisions and costs on grid maps for autonomous vehicle motion planning," *IEEE Transactions on Intelligent Transportation Systems,* vol. 15, no. 5, pp. 2249-2260, 2014. https://doi.org/10.1109/TITS.2014.2313562

[7] A. Stentz, "Optimal and efficient path planning for partially-known environments," in *Proceedings of the 1994 IEEE international conference on robotics and automation*, 1994: IEEE, pp. 3310-3317.

[8] A. T. Sadiq and A. H. Hasan, "Robot path planning based on PSO and D algorithmsin dynamic environment," in *2017 International Conference on Current Research in Computer Science and Information Technology (ICCIT)*, 2017: IEEE, pp. 145-150. https://doi.org/10.1109/CRCSIT.2017.7965550

[9] A. H. Hasan, "Robot path planning based on improved max–min ant colony optimization algorithm in dynamic environment," *Research Journal of Applied Sciences,* vol. 11, no. 10, pp. 1060-1068, 2016.

[10] M. A. Contreras-Cruz, V. Ayala-Ramirez, and U. H. Hernandez-Belmonte, "Mobile robot path planning using artificial bee colony and evolutionary programming," *Applied Soft Computing,* vol. 30, pp. 319-328, 2015. https://doi.org/10.1016/j.asoc.2015.01.067

[11] Q. R. Mahmood, A. H. Hasan, and H. K. Khafaji, "Robot Path Planning Based on Hybrid Adaptive Dimensionality Representation with Glowworm Swarm Optimization," in *2021 4th International Iraqi Conference on Engineering Technology and Their Applications (IICETA)*, 2021: IEEE, pp. 241-246. https://doi.org/10.1016/j.asoc.2015.01.067

[12] M. Davoodi, F. Panahi, A. Mohades, and S. N. Hashemi, "Clear and smooth path planning," *Applied Soft Computing,* vol. 32, pp. 568-579, 2015. https://doi.org/10.1016/j.asoc.2015.04.017

[13] A. Alaidi, I. Aljazaery, I. Mahmood, and F. Abed, "Design and Implementation of a Smart Traffic Light Management System Controlled Wirelessly by Arduino," *international Journal of Interactive Mobile Technologies,* vol. 14, no. 7, pp. 32-40, 2020. https://doi.org/10.3991/ijim.v14i07.12823

[14] J. Zhao, D. Cheng, and C. Hao, "An improved ant colony algorithm for solving the path planning problem of the omnidirectional mobile vehicle," *Mathematical Problems in Engineering,* vol. 2016, 2016. https://doi.org/10.1155/2016/7672839

[15] A. H. Hasan and A. M. Mosa, "Multi-robot path planning based on max–min ant colony optimization and D algorithms in a dynamic environment," in *2018 International Conference on Advanced Science and Engineering (ICOASE)*, 2018: IEEE, pp. 110-115. https://doi.org/10.1109/ICOASE.2018.8548805

[16] I. Hassani, I. Maalej, and C. Rekik, "Robot path planning with avoiding obstacles in known environment using free segments and turning points algorithm," *Mathematical Problems in Engineering,* vol. 2018, 2018. https://doi.org/10.1155/2018/2163278

[17] X. Dai, S. Long, Z. Zhang, and D. Gong, "Mobile robot path planning based on ant colony algorithm with A* heuristic method," *Frontiers in neurorobotics,* vol. 13, p. 15, 2019. https://doi.org/10.3389/fnbot.2019.00015

[18] H.-J. Wang, Y. Fu, Z.-Q. Zhao, and Y.-J. Yue, "An improved ant colony algorithm of robot path planning for obstacle avoidance," *Journal of robotics,* vol. 2019, pp. 1-8, 2019. https://doi.org/10.1155/2019/6097591

[19] Z. Yang, J. Li, L. Yang, Q. Wang, P. Li, and G. Xia, "Path planning and collision avoidance methods for distributed multi-robot systems in complex dynamic environments," *Mathematical Biosciences and Engineering,* vol. 20, no. 1, pp. 145-178, 2023. https://doi.org/10.3934/mbe.2023008

[20] T. Stützle and H. H. Hoos, "MAX–MIN ant system," *Future generation computer systems,* vol. 16, no. 8, pp. 889-914, 2000. https://doi.org/10.1016/S0167-739X(00)00043-1

[21] M. Brand, M. Masuda, N. Wehner, and X.-H. Yu, "Ant colony optimization algorithm for robot path planning," in *2010 international conference on computer design and applications*, 2010, vol. 3: IEEE, pp. V3-436-V3-440. https://doi.org/10.1109/ICCDA.2010.5541300

[22] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE computational intelligence magazine,* vol. 1, no. 4, pp. 28-39, 2006. https://doi.org/10.1109/CI-M.2006.248054

7 Authors

Dr. Ali Hadi Hasan, Assist. Professor qualified to direct research at University of Babylon, Iraq, and other Iraqi Institutions. He got his B.Sc. in Applied Mathematics from University of Technology, Iraq in 1989, M.Sc. and Ph.D. in Computer Science from University of Babylon, Iraq in 1999, 2013 respectively. He held a position for two years as the Associate Dean for Scientific Affairs and Graduate Studies at the College of Information Technology, University of Babylon, from 2020 to 2022. His area of interests is Artificial Intelligence, Robotic Path Planning Algorithms, Swarm Intelligence, Machine Learning, Pattern Recognition and Data Mining & warehouse. He can be contacted at email: alihadi@itnet.uobabylon.edu.iq.

Dr. Hasanen S. Abdullah, Assist Professor qualified to direct research at University of Technology, Iraq, and other Iraqi Institutions. He got his B.Sc., M.Sc. and Ph.D. in computer science from University of Technology, Iraq in 2000, 2004 and 2008 respectively. His area of interests is Artificial Intelligence, Machine Learning, Swarm Intelligence, Pattern Recognition, Data Mining & warehouse, and Business Intelligence. He can be contacted at email: Hasanen.S.Abdullah@uotechnology.edu.iq.

Dr. Mohamad AB. Saleh holds a Doctor of Computer Science degree from Lebanese University - EDST, Lebanon in 2018. He also received his B.Sc., H. Diploma and M.Sc. (Computer Science) from University of Technology and Almustansrya University, Iraq in 1993, 1998 and 2006, respectively. He is currently an associate professor at Al-Iraqia University, Baghdad, Iraq since July 2008. His research includes metaheuristics, machine learning, data mining, logic and computer security. He has published many papers in international journals and conferences. He can be contacted at email: mohamad.ab.saleh1@gmail.com.

Article submitted 2023-04-19. Resubmitted 2023-05-23. Final acceptance 2023-05-24. Final version published as submitted by the authors.