INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATIONS AND ROBOTICS
ISSN 2320-7345

MULTI-ROBOT PATH PLANNING WITH FUZZY OBSTACLES

Asseel Jabbar Almahdi[1]

Computer Science, Thi-Qar university,Iraq, asseelalmahdi@gmail.com

Abstract: - Robot path planning is a key issue in mobile robot navigation problem which gained a lot of attention in recent years. In this paper, we have modeled the environment as fuzzy and have considered "being an obstacle" as a fuzzy feature to look for a path from the robot location to the goal location by using QPSO* algorithm, while the environment is unknown. In this problem, while we look for a path, we keep getting to know the environment, and the planning is online. We have two behaviors. First, when the obstacle is not in the view of the robot. Robots' motion is done based on three factors – tendency to goal, keeping distance from the rest of the robots, and random factor. Second, when the obstacle is in the view of the robot. In this case, bug2 algorithm is performed. In our tests, modeling the environment as fuzzy helps improve the route.

Keywords*:* Path planning, Multi-robot, Unknown environment, Fuzzy set

1. Introduction

In the past five decades, the use of robots has increased remarkably due to lowered prices and increased precision in pieces. The robots are used to do hard and repetitive works, or are used in places where human presence is dangerous or impossible for any reason such as aerospace research, the nuclear industry, and the mining industry. Human desire to reduce fuel costs of the robots, accelerate doing works, decrease maintenance costs, and to increase longevity of robots cause this question: What is the safe and short path? Which is a fundamental question in an important research area under the title "the robot path planning" and have been studied from various aspects. Path planning is to generate a collision free path in an environment with obstacles from a specified start location to a desired goal destination and optimize it with respect to some criterion [2, 3]. However, Depending on how much the robot knows about the entire information of the surrounding environment the path planning methods can be classified into the following two types:

(1) Robot path planning in a clearly known environment in which the robot already knows the location of the obstacles before it starts to move.

(2) Robot path planning in a partly known or uncertain environment [4] in which the robot path planning depends on the sensory information of the environment to acquire the local information of the location, shape and size of obstacles, and then uses the information to proceed local path planning.

Like other branches of science, robotics has made great progress under the influence of fuzzy set and fuzzy logic. But mainly, its focus has been on fuzzy problem solving and using fuzzy logic, and less attention has been paid to fuzzy environment. But with the advancement of science and manufacturing of various types of movement devices (robots with legs, robot bird, wheeled robot, etc.), the need for a more accurate model of environment becomes indispensable. For rescuing robot in a city after earthquake, very tall building is impassable then the building is totally in obstacles set, un damage street is passable then placed in the non-

obstacles set, but what about a pit of water, soil mass ? (They are member of which set? Obstacle or non-obstacle). Suppose that there is a hole of water between robot and person who need help, if person be in normal condition, robot will put hole of water in obstacle set. Robot prefer to be safe than reach the person fast. If person be in emergency condition, robot will put the hole of water in the non-obstacle set. Robot prefers to be damaged than reach person later. We consider being obstacle as fuzzy feature and passing over things (consider them as obstacle or not) depend on condition. We assign fuzzy number to each part of the environment. The bigger number mean, more membership in obstacle set, more danger for robot or more harmful environment. At first; in 1983 – 2000 [5, 6] environments is category in two set obstacle and non-obstacle set. Then in 2001[9] the environment contains both obstacles and so-called danger zones but in [7] and [10] the environment is divided into 4 categories desirezones, common zones, dangerzones and obstacle. Now we divided the environment to infinite set by assigning real number to each part of set and to some extent useful information is lost but in this approach because each part is a real number, all information is considered.

Generally, existing studies on robot path planning have two main groups [8], i.e. classical and heuristic approaches, classical approaches like cell decomposition, Roadmap and potential fields, etc. However, the classical approaches have several drawbacks, e.g. high time, large computation and trapping in local minima, these drawbacks cause the classic techniques to be inefficient in high dimensions spaces and the path obtained is often not globally optimal. So, heuristic methods have been developed in order to improve the efficiency of Classic methods. Heuristic methods for robot path planning include probabilistic roadmaps method (PRM), rapidly-exploring random tree (RRT), particle swarm optimization (PSO), genetic algorithms (GAs), Simulated Annealing (SA), neural networks (NNs), ant colony optimization (ACO), etc.

In this paper, we have modeled the environment as fuzzy and have considered “being an obstacle” as a fuzzy feature to look for a path from the robot location to the goal location, while the environment is unknown. We will use a special algorithm called Quantum Particle Swarm Optimization Star Algorithm (QPSO*) to solve this problem. Which is similar to the PSO algorithm, but here we did not use it as a solution of the Optimization problems.

This paper is organized as follows. In Section 2, the Problem definition is given. In Section 3, the new proposed algorithm (QPSO*) is proposed. In section 4 we present the experiment results and finally we conclude and give some of future works in Section 5.

2. Problem definition

In this paper, the problem of robot path planning in unknown environments with fuzzy obstacles is studied. Robots studied here are considered of type disc. The number of robots in the environment that constitute multi-robot systems is limited and specific. Each robot is displayed with R_i, i = 1, 2, 3. The starting point of all robots is the same and its name is *Start*, and the goal point is *Goal*. The robots always move with its maximum speed, V_i, and the ability to rotate is 180 degrees. Obstacles available in the environment are static and have different sizes and dimensions. Measurements of Obstacles are determined by fuzzy number. Whenever this number is larger, obstacle becomes more impassable (or more dangerous). These obstacles are considered in the form of concave and convex shapes. Other features of the system are determined as follows:

1. Robots’ contact with each other is low and hardly done with noise
2. Each robot is only aware about the location of other robots. Realization of other robots’ location may be associated with noise.
3. Each robot is capable of calculating its needs.
4. Organizing system takes the form of a decentralized system.
5. Each robot designs its path as it moves (because the environment is unknown).
6. Each robot is unaware of the path of other robots.
7. Path planner is in the online form and is performed with the help of robot’s sensor.

Example of this environment with the presence of some obstacle and 7 robots are shown in the figure (1).

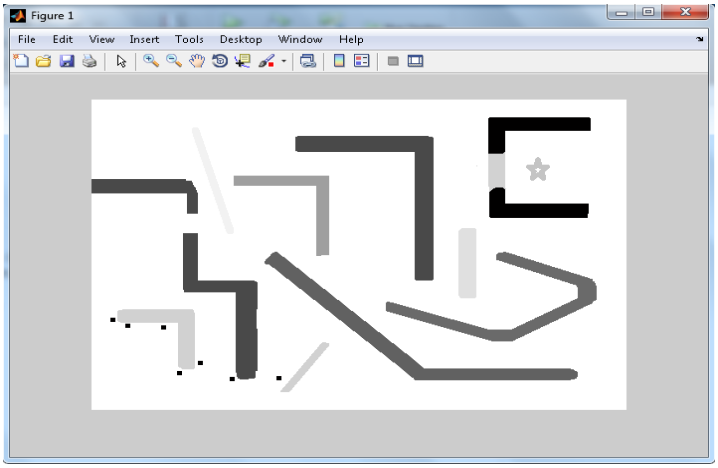


Figure1: example of the problem with the presence of some obstacle and 7 robot

3. Quantum Particle Swarm Optimization Star Algorithm (QPSO*)

PSO algorithm is an algorithm that is innovative. Because of its specific features such as high speed, ability to cover all the space, and requiring very little memory, it is very useful for optimization problems. In [1], QPSO algorithm, by adding a random vector to particles motion vector, space covering factor will be reinforced more than before and a more optimal solution with more speed can be achieved. In this paper we will introduce QPSO* which, unlike previous works, is not a solution for optimization problem; and the particles are not processes in the solution space but they are real robots in a real unknown environment.
Finding robot's path toward known goal, while the environment is unknown, is one of the most important issues in robots' motion planning. Here the robots show two types of behavior: the behavior when the obstacle is not in the view of the robot which is planned with the QPSO* algorithm, and the behavior when the obstacle is in the view of the robot which is planned by BUG2 algorithm.

3.1 The first behavior (the obstacle is not in the view of the robot)

When the Robots do not encounter obstacles, they move according to this algorithm. Movement of the robots in this case is based on three factors:
1. A robot's desire to reach the goal.
2. The factor that causes distribution and distance of robots from one another.
3. Random factor.
Motion vector is the result of the weighted sum of these three factors. The weight of each factor, based on simulations and previous experience, is regulated before start of robots' movement and does not change while they are moving.

3.1.1 Robot's desire to reach the goal

This vector is easily obtained by subtracting the target location from the particle location and then normalizing it (making vector length equal to one).

$$\boldsymbol{V_{goal}} = \frac{\boldsymbol{goal\ Coordinate - particle\ Coordinate}}{\boldsymbol{|goal\ Coordinate - particle\ Coordinate|}} \qquad (1)$$

3.1.2 Distribution factor:

Distribution factor objectives:

1. Covering all areas to achieve the less costly path.

2. Non-collision movement of robots (prevent robots from collision).

Distribution vector which has been used in the model of this paper can be named the behavior simulation of electrical charges which have the same sign.

As we know, if the electrical charges have the same sign, the electrostatic force between them is repulsive, and movement of a charge is based on the sum of all these forces. The size of incoming force on one charge by another one is:

$$f = c\frac{q1 * q2}{r2} \qquad (2)$$

The important point of the behavior of these charges is prevention of conflict with each other and appropriate distribution in the environment which is a result of $1/r^2$ factor. When two robots are close to each other ($r \to 0$), distribution factor gives the result that the resulting force from the distribution tends to infinity and neutralizes the other factors (random, moving toward the goal, and the force that the other robots also exert to each other). In other words, in certain circumstances, the only aim of a robot movement is to get away from the other robot which is too close to it.

$$f_{ij} = \frac{1}{r_{ij}{}^2} \qquad (3)$$

f_{ij}: is the force exerted on the robot *i-th* by robot *j-th*

To achieve the best distribution, the reverse power of distance for force (as a reverse power function of distance) is variable given to the environment, and could be the first or the third power (Equation 4 and 5). As a result, depending on the environment one of the formulas below can be used:

$$f_{ij} = \frac{1}{r_{ij}{}^1} \qquad (4)$$

$$f_{ij} = \frac{1}{r_{ij}{}^3} \qquad (5)$$

$$f_i = \sum_{j=0}^{n} f_{ij} \quad i<>j \qquad (6)$$

Wheref_i: the force entered on robot *i-th*, n: the number of robots

This model requires two corrections:

The first correction: when robots are close to the goal, the distribution factor prevents concentration of the robots on the goal. That is why we define this force proportional to the distance from the goal.

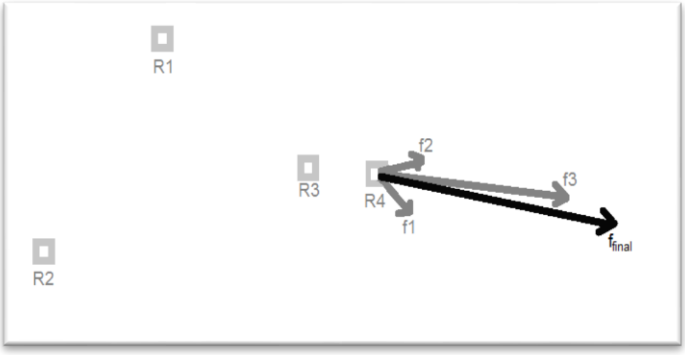


Figure2: an example shows thatf_i: is the force entered on R4 by Ri andf_final: Outcome of all forces, distribution factor

$$f_{ij} = \frac{1}{r_{ij}^2} \quad (3)$$

$$f_i = r_{i\,goal} \sum_{j=0}^{n} f_{ij} \quad i<>j \quad (8)$$

$r_{i\,goal}$: Distance between the robot i-th and the goal

However, this movement formula is not efficient enough, and reaching the goal by robots is not quite fast. The following formulas have better results:

$$f_{ij} = r_{goal\,j} \frac{1}{r_{ij}^2} \quad (9)$$

$$f_i = \sum_{j=0}^{n} f_{ij} \quad i<>j \quad (10)$$

Second correction: the distribution at a direction parallel to the vector <goal, start> is not desirable for us. It just slows down the particles motion. In fact, the optimal distribution is the direction perpendicular to this vector (fig. 3 and 4).

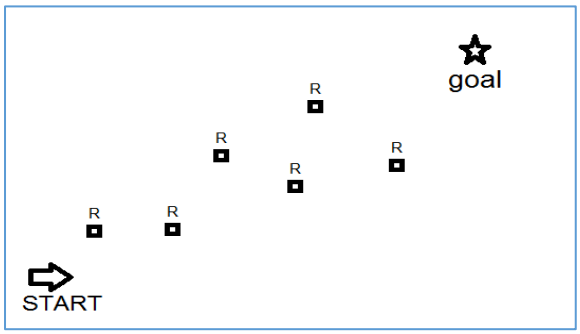


Figure 3: Non-optimal distribution

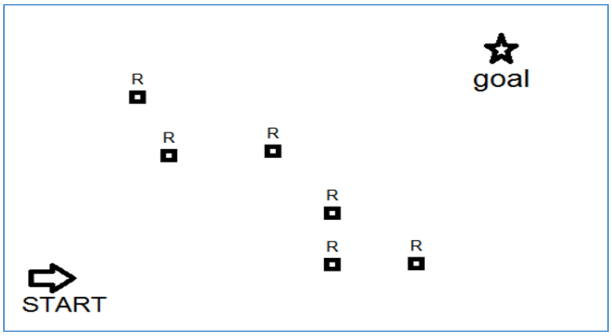


Figure 4: Optimal distribution

Of course, when encountered obstacles with a side parallel with vector <goal, start>, distribution factor in a direction parallel with the vector <goal, start> is useful. So instead of eliminating this factor, we will weaken it.

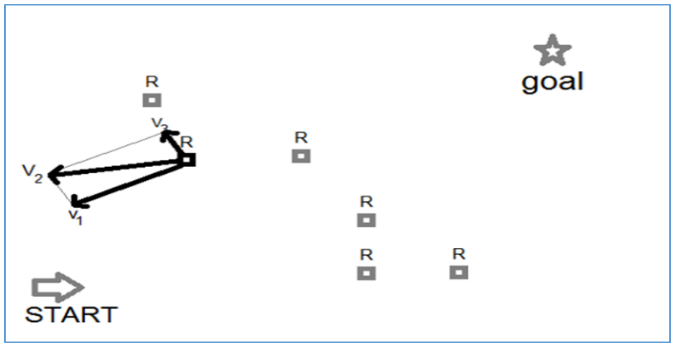


Figure 5

Figure 5: an example shows that v2: is the same fi or the sum of forces from other particles, v_1: Parallel vector with the vector <goal, start> and its magnitude resulting from the dot product (cosine) of unit vector of <goal, start> in v_2 andv_3: Distribution vector which <goal, start> vector completely removed from it.

Algorithm to obtain distribution vector:

1. Obtaining f_i with these formulas:

$$f_{ij} = r_{goal\ j} \frac{1}{r_{ij}^2} \quad (9)$$

$$f_i = \sum_{j=0}^{n} f_{ij\ \ i<>j} \quad (10)$$

2. Calculating the unit vector in the direction <goal, start>

$$v_{unit} = \frac{< goal, start >}{| < goal, start > |} \quad (11)$$

3. Calculating the result of dot product of the unit vector in the total force, this magnitude is the size of v_1 vector, fig. (5).

$$|v_1| = v_{unit} . f_i = x_{v_{unit}} * x_{f_i} + y_{v_{unit}} * y_{f_i} \quad (12)$$

4. By having the size and the direction of v_1, we get the vector v_1. This vector is the distribution factor in the direction parallel to the vector <goal, start>. As explained above, we do not want to neutralize this factor but only to weaken it.

$$v_{distribution} = f_i - c * v_1 \quad (13)$$

In performed tests, the best value for c is equal to 0.7

3.1.3 Random factor

The differences between PSO algorithm and QPSO is random factor. The aim of this factor is to search space better. Random factor as its name shows is a vector which is the result of two random numbers.

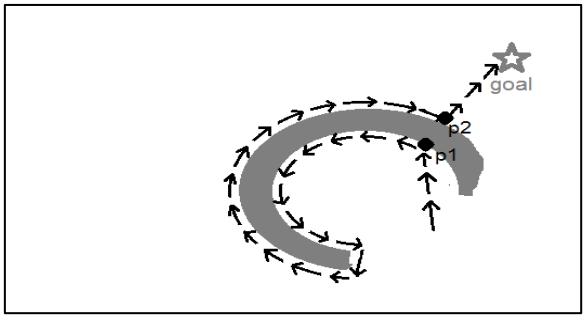
3.2 The second behavior (the obstacle is in the view of the robot)

After the obstacle was placed in the robot vision and robot found out that it cannot continue its normal QPSO* movement, the robot's movement mode changes and the decision of the movement will be made on the basis of bug2 algorithm. Bug1 algorithm can be more efficient for figures that have more complexity. Therefore, the movement algorithm can be selected between bug1 and bug2.

In bug2 algorithm, robots move in a way that the obstacle always be on the left (right) side of them. This movement will continue until the robot cuts the segment that connects the start point (first-point obstacle seen by the robot) to the goal. If the robot has not already passed that point and a possibility to follow the path toward the goal exists, the robot moves toward the goal and the movement mode becomes normal. However, based on the needs of the problem this algorithm is changed.

The pseudo-code of movement in the case that we are in contact with the obstacle:

1. *Meat obstacle*
2. *Save this point as "first obstacle point"*
3. *While (two segment "location before move location after move" and "first obstacle point goal location " have no intersect) do*
4. *Move*



5. Figure 6: an example shows that P1: is first Obstacle Point and P2: the intersection of two segments "location Before Move, location After Move" and "first Obstacle Point, goal Location"

In the figure (6) the robot moves in the form in which the obstacle always be toward his right. The robot does not have information about the form of the road map and the obstacle so it is possible to choose a longer path

3.2.1 Implementation of the movement

After obstacle be in the view of robot, the robot begins to rotate around the obstacle. The direction that the robot selects to move is the robot feature and its value is fixed in the project for each of them. For half of robots, while moving, the obstacles are on the left side, and for the other half, on the right side. The reasons why not all of the robots choose one same direction are raising distribution, and as a result, reaching the goal more quickly.

Movement algorithm:

We choose two points on the obstacle border from which the current location of the robot equals to the size of the step length (fig. 7).

We choose a point inside the obstacle. By calculating the sine multiplication of motion vector in "robot's location, a point inside the obstacle" vector, the fact that whether the obstacle is located on the right hand or on

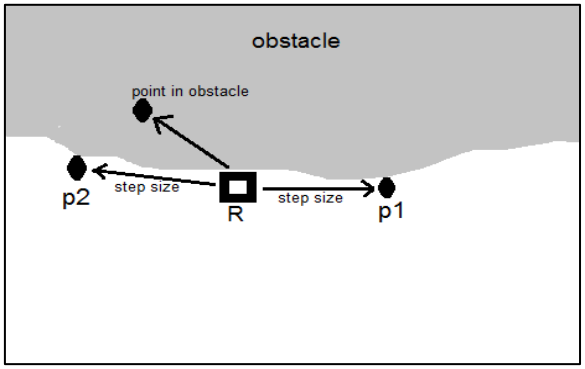


Figure7: Movement of the robot around the obstacle

the left of the moving robot is determined.

The pseudo-code of movement algorithm around the obstacle:

$V_1 = P_1\_Robot\ location$

$V_2 = obstacle\ point - Robot\ location$

$V_1 * V_2 = X_{V_1}.Y_{V_2} - Y_{V_1}.X_{V_2}$

$if\ (V_1 * V_2 > 0)$

If move toward P1, obstacle is in left hand of robot

else If move toward P1, obstacle is in right hand of robot

The environment in this issue is assumed to be limited, and the robot cannot get out of the environment. If the robot collides with the boundaries of the environment, the direction of the motion changes (if the obstacle is on the left side of robot, the obstacle locates on the right side, and vice versa) until in case there is any path, the robot finds that path.

3.2.2 Implementing change the mode from rotation around the obstacle to normal mode:

In order to detect whether the robot has crossed the line" FirstObstaclePoint goalLocation" or not, we store locations before and after every move, and then we check whether these two segments cut each other or not. By Solving the following two equation we obtain line2 Coefficients

$$a_{line2}.x_3 + b_{line2}.y_3 + c_{line2} = 0$$

$$a_{line2}.x_4 + b_{line2}.y_4 + c_{line2} = 0$$

After gaining $a_{line2}, b_{line2}, c_{line2}$, we compensate in the equation of Line 2 two points(x_1, y_1) and(x_2, y_2). If $(a_{line2}.x_1 + b_{line2}.y_1 + c_{line2}).(a_{line2}.x_2 + b_{line2}.y_2 + c_{line2}) < 0$ or different Signs shows that the line 2 cuts segment 1.

3.3 Modeling with Fuzzy obstacles:

As explained in chapter two, the obstacles are fuzzy. The decision over whether to pass or turn around the obstacle is made according to the following factors:

1. Fuzzy number of obstacle: represents an obstacle's being dangerous and damaging. Zero number means safe environment. One number means insurmountable (impassable) obstacle.

2. Constant factor: which includes the price of the robot, importance of the goal, modeling the environment, and other environmental conditions which is given in the form of a fixed number to the robots.

3. Time: the most important factor to decide the passage from one obstacle in our model is time. A time is given as safe time to the robot. For example, we have 12 days for providing the astronauts with food supplements. After this fixed time, the more time passes, due to the worsening situation, the more risk-taking the robot, and subsequently the stronger the robot's desire to pass difficult obstacles becomes.

4- Security of progress: The robot which after a certain time is located near a target, in comparison to the robot which at the same time has a distance from the target, has more security and is exposed to less risk.

The shape and structure of obstacles: because the environment is unknown, and we are unaware of the shape and the structure of the obstacles, this factor has no impact in the decision-making. When an obstacle is placed in the robot's vision, robot makes the decision to whether to cross the obstacle or to turn around it according to how much impassable the obstacle is, time, distance from the target, and the fixed number that has been given to it before the move.

4. Experiment results

In this section, we model diverse environments as both fuzzy and non-fuzzy environments. We attribute value to the answers of fuzzy and non-fuzzy models, and then compare them with each other. The most important criteria of value attributing are arrival time of the robot to the goal as well as the damage or the danger which the robot faces due to its cross over the obstacles. The second criterion of value attributing which is less important is computer's memory and time consumption for path planning. In order to compare the paths, we convert the amount of the risk of crossing the obstacles to path length linearly. In this study, we tried to take a variety of environments. The method presented in the previous section (QPSO*) is tested for single-robot and multi-robot systems.

Each environment is modeled as both fuzzy and non-fuzzy, modeling an environment as non-fuzzy means that the environments are divided into two categories of with obstacle and without obstacle. Then 20 test will be done on each of the models. Both starting and target points are selected randomly. Only in the third

environment, in order to face the challenge of local minimum, there are some limitations to choose starting and target points. To compare fuzzy and non-fuzzy models, it is necessary that start and end points are the same in both models. Finally, cost of the path is calculated based on cost of crossing the obstacle in addition to the path length. Costs of crossing the obstacle and traveling through the path are calculated with below formulas.

Path cost= Cost of crossing the obstacle+ path length

*Cost of crossing the obstacle= the membership in obstacle*the distance traveled over the obstacle*Conversion Factor*

Conversion factor: a fixed number which is considered 30 in the experiments.

After comparing fuzzy and non-fuzzy environments, we will compare speed of reaching the goal in single-robot and multi-robot systems.

4.1 Environment Types

In the experiments, we have used three types of environments.

1. Regular environment: regular environments are such as home, hotel, and office environments for the servant robot, and city for the taxi driver robot. Fuzzy obstacles in these environments can be noted as the possibility of traffic on the streets of the city for taxi driver robot, and disturbing for people for the servant robot (Fig. 8).

2. Irregular environment: irregular environments are such as city after the earthquake, or as fuzzy obstacles in which there are the hills of soil, water holes, and crumbling walls (Fig. 9)

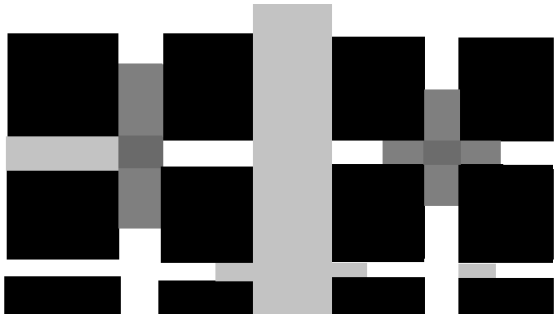


Figure 8: Regular environment

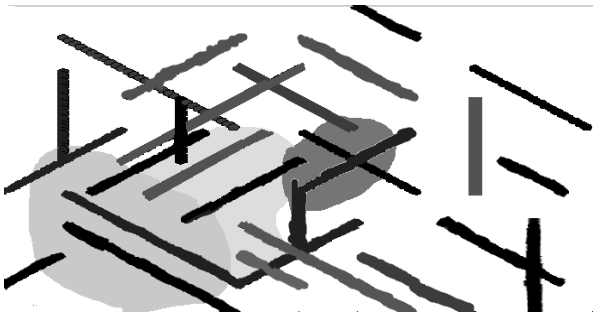


Figure 9: Irregular environment

3. Environments with high local minimum: this environment is designed for assessment of the ability of the algorithm to escape from local minima, and the impact of fuzzy look in the algorithm's escape from local minimum (Fig 10).

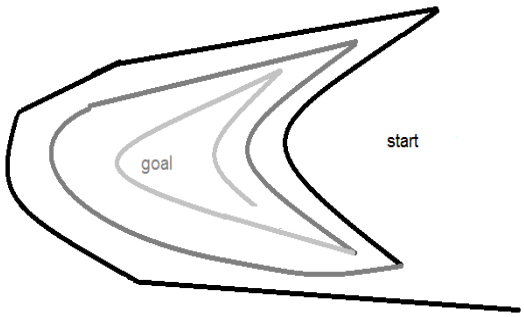


Figure 10: Environment with high local minimum

4.2 A comparison between fuzzy and non-fuzzy model in QPSO* algorithm for single-robot system

The purpose of this experiment is to evaluate the performance of QPSO* in fuzzy and non-fuzzy environments while there is only one robot. The data below is the results of 60 tests (20 tests in each of the three environments). In each experiment, one target point and one starting points is selected randomly, then the environment is modeled once in fuzzy and once again in non-fuzzy environment, and then the cost of the path will be calculated. In other words, each test includes two sub-experiments, one in fuzzy and the other in non-fuzzy model.

In the third environment, we want to face the challenge of local minimum, so we have some limits in selecting the start and end points (fig. 11, 12 and 13)

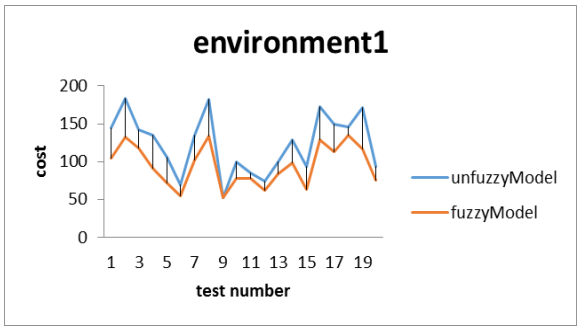


Figure 11: Diagram comparison between fuzzy and non-fuzzy model for single-robot system of QPSO* algorithm in first environment

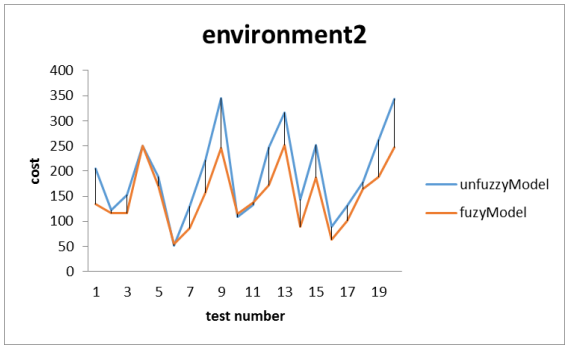


Figure 12: Diagram comparison between fuzzy and non-fuzzy model for single-robot system of QPSO* algorithm in second environment

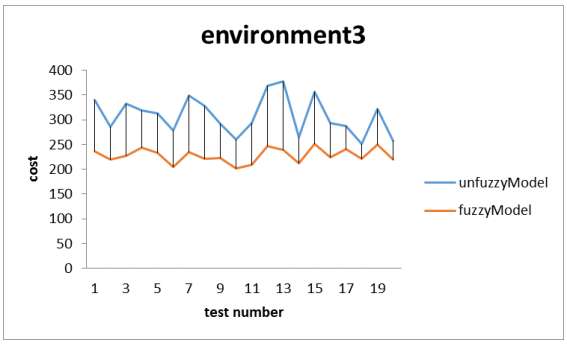


Figure 13: Diagram comparison between fuzzy and non-fuzzy model for single-robot system of QPSO* algorithm in third environment

The distribution of costs is seen in 1 and 2 environments due to the random selection of the starting and the target points. As explained above, to make the robot face the challenge of local minimum, in the third environment, start and end points were chosen in limited areas. As can be seen in the graphs, fuzzy modeling of environment leads to lower costs. This reduction is 18%, 33%, and 29% in the first, second, and third environment, respectively.

4.3 A comparison between fuzzy and non-fuzzy models in QPSO* algorithm for multi-robot systems

Like the previous series, tests were carried out in all three environments and 20 tests in each of them. After selecting a starting point and a goal point at random, every test which includes two sub-tests (one in fuzzy and the other in non-fuzzy model) is done. Tests were executed by 5 robots (fig. 14, 15 and 16).

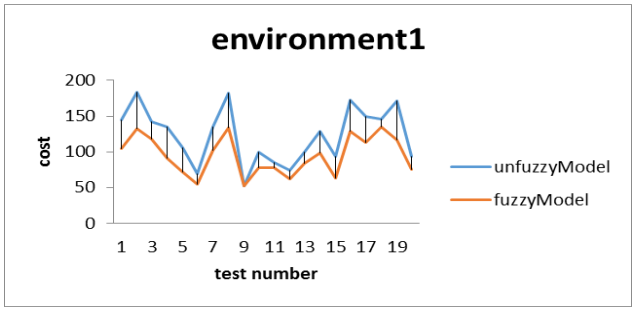


Figure14: Diagram comparison between fuzzy and non-fuzzy model for Multi-robot system of QPSO* algorithm in first environment

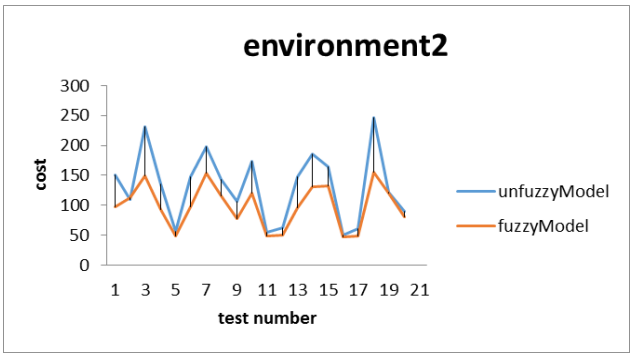


Figure 15: Diagram comparison between fuzzy and non-fuzzy model for Multi-robot system of QPSO* algorithm in second environment

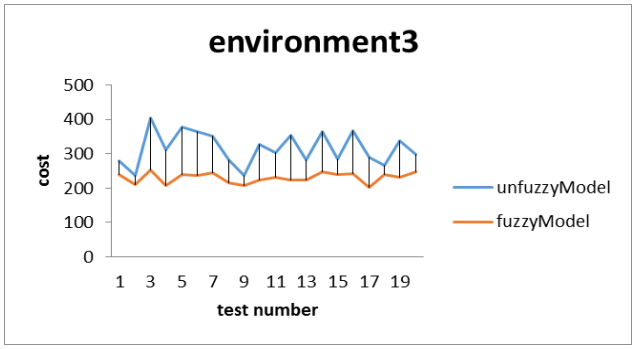


Figure 16: Diagram comparison between fuzzy and non-fuzzy models for Multi-robot system of QPSO* algorithm in third environment

In the third environment, we will face a local minimum challenge, therefore there are some limitations regarding selecting the start and end points.

Due to being multi-robot, speed increased in all environments. As it is seen, the average of the path cost in these diagrams is lower in comparison to that in single-robot diagrams.

Cost reduction with a fuzzy perspective is 23%, 34%, and 36%, in the first, second, and third environment, respectively.

5. Conclusion and future work

In this paper, we tried to have accurate look at the environment. In previous works, space was divided into two categories of with obstacle and without obstacle. Some of the works divided the environment space into the following categories: normal, desirable, undesirable, and obstacle. In our work, to each part of the space environment a fuzzy number was attributed. One algorithm for the unknown environment was written, and was tested in all the three environments. In all environments, fuzzy vision improved the path and reduced costs. Of course, fuzzy vision can slightly increase the computation time on the computer. In our work the cost of passing over the obstacles regardless of the condition of the robot calculated the same. While we know that risk-taking robot when fully healthy and when damaged must be different. In addition, modeling environment with moving obstacles and target are not considered in this study. Therefore nonlinear consider the cost of passing over the obstacle or moving obstacles exist in an environment is our future research topic.

REFERENCES

[1] Jun Sun, W. Xu and 'B. Feng, a Global Search Strategy of Quantum-Behaved Particle Swarm Optimization. Proceedings of the 2004 IEEE, Conference on Cybernetics and Intelligent Systems Singapore, 1-3 December, 2004.
[2] S. Florczyk, “Robot Vision Video-based Indoor Exploration with Autonomous and Mobile Robots “, WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim, 2005.
[3] Pradipta kumar Das, A. Konar and R. Laishram ,Path Planning of Mobile Robot in Unknown Environment, Special Issue of IJCCT Vol.1 Issue 2, 3, 4; 2010 for International Conference [ACCTA-2010], 3-5 August 2010.
[4] Ting-Kai, Wang, Q. Dang and P. Pan, Path Planning Approach in Unknown Environment. International Journal of Automation and Computing, 7(3), August 2010, 310-316.
[5] T. Lozano-Pérez. Spatial planning: A configuration space approach. IEEE Transactions on Computers, C-32(2):108–120, 1983.
[6] O. B. Bayazit, G. Song, and N. M. Amato. Ligand binding with OBPRM and user input. In IEEE International Conference on Robotics and Automation, volume 1, pages 954–959, Seuol, Korea, May 2001.
[7] K. Belghith, F. Kabanza, Anytime Dynamic Path-Planning with Flexible Probabilistic Roadmaps. IEEE International Conference on Robotics and Automation, pages 2372- 2377, 2006.
[8] D. Gong, J. Zhang, Y. Zhang. Multi-objective Particle Swarm Optimization for Robot Path Planning in Environment with Danger Sources, JOURNAL OF COMPUTERS, pages 1554- 1561, 2011.
[9] D. Sent & M. H. Overmars, Motion planning in environments with dangerzones, Proceedings of the 2001 IEEE International Conference on Robotics and Automation, Seoul, Korea, 2001, 1488-1493.
[10] Z. M. Ma1, Froduald, Towards a Robot Path Planner for Dangerzones and Desirezones, 2002.