Fuzzy Logic Control for Mobile Robot Navigation in Automated Storage

Chadi F. Riman* and Pierre E. Abi-Char

College of Engineering and Technology, American University of the Middle East, Kuwait;
Email: pierre.abichar@aum.edu.kw (P.E.A.)
*Correspondence: chadi.riman@aum.edu.kw. (C.F.R.)

***Abstract*—Automated Storage (AS) are designed to store and retrieve products in specific locations within manufacturing, warehouses, institutions, and others. These AS involve the usage of robots to move the stored items in and out of the warehouse. However, a challenge for AS systems is to solve the path planning for finding shortest path in a minimum amount of time while avoiding collisions with other robots or static obstacles. Fuzzy Logic systems are widely used in several application areas requiring mimicking the human decision logic under uncertainty. In this paper, we proposed an Automated Storage (AS) robot navigation by using three Fuzzy subsystems combined together to ensure path planning with obstacle avoidance. These three fuzzy subsystems are: Reach Target, Avoid Obstacle, and Escape Cul-De-sac. Therefore, fuzzy rules are employed along with the corresponding defuzzication process to control left and right wheel movement steps of the robot. These systems achieve reaching the goal (using the first subsystem) while avoiding different obstacles on the way (using the second subsystem), even the ones that form a trap (using the third subsystem). These three systems will be used for path planning and following. The overall model was simulated using C# code. The initial results showed the effectiveness of the model in different scenarios: namely no obstacles, static ones, traps, and dynamic obstacles. The path length was comparable to that of traditional shortest path methods such as Dijkstra and A*. The results were also compared to a newer method called APSO. The system's response was quick due to the fewer needed instructions and reduced memory storage needs. All this was done assuming a constant speed for robots and dynamic obstacles.**

***Keywords*—automated storage, fuzzy logic, path planning, obstacle avoidance, robotics**

I. Introduction

Automated Storage (AS) retrieval systems are currently widely used worldwide for many companies, and are considered to be a part of the industry 4.0 standard. These AS involve usage of robots to move the stored items in and out of the warehouse. A challenge for AS is to solve the path planning for finding shortest path in a minimum amount of time while avoiding collisions with other robots or static obstacles. There are several Shortest-path algorithms which are used, such as Dijkstra and A*. Dijkstra's algorithm is a famous algorithm that is used for finding the shortest path, from starting node to target node in a weighted graph. This algorithm makes a tree of the shortest path from the starting node to all other nodes in the graph. It makes use of weights of the edges for finding the path that minimizes the total distance from the source node to all other nodes. The A* algorithm is just like Dijkstra's algorithm. The only difference is that A* algorithm tries to look for a better path by using a heuristic function H which gives priority to nodes that are supposed to be better than others while Dijkstra's algorithm just explores all possible paths [1].

Fuzzy logic is also used in robotics to solve the navigation problems without initial calculations of the whole path. Fuzzy logic is a simple way of thinking that mimics human being's decisions when uncertainty is involved. Fuzzy logic is widely used in a range of applications, such as control systems, image processing, natural language processing, medical diagnosis, and artificial intelligence.

In our previous works, an enhanced A* algorithm was used to handle the robot's navigation in AS system. The first work [2] suggested a simple and efficient algorithm to find the path planning with collision-free using an enhanced A* algorithm to find the shortest path from a source node to a destination node, with two heuristic functions H1 and H2. In our second work [3] the addition of priority path in a multiple robot navigation problem.

In this paper, we present a different approach to the navigation problem in AS by using Fuzzy Logic control. This new approach does not need exact locations of all obstacles in the map, but just the adjacent ones by using the available distance sensors. Our model includes three fuzzy systems combined together: Reach Target, Avoid Obstacle, and Escape Cul-De-Sac. These subsystems will be explained in details in a following section.

The rest of this paper is organized as follows. In the following section, we briefly survey the relevant related work. Following comes Fuzzy Logic definition along with system model requirements. The proposed fuzzy model with its subsystems is described next. Next, applications to AS systems with simulation discussions are presented. Finally, we conclude the paper and present possible future work.

Manuscript received March 27, 2023; revised May 6, 2023; accepted June 8, 2023.

II. Related Work

In robotics navigation, the goal is to reach the target following the shortest path, while avoiding obstacles on the way including concave ones (also called cul-de-sac). Obstacles can be either static or dynamic. There are many available solutions using fuzzy logic systems. These systems can be a single fuzzy inference system that includes all situations, or a separate dual system with one activated at a time: one for reaching target and one for avoiding obstacles. There is also a triple fuzzy inference system: one for reaching target, one for avoiding obstacles, and one for escaping concave obstacles by wall-following behavior. A fuzzy system can also be used as is or with reinforcement learning that helps optimizing fuzzy rules after a learning phase. In all cases, fuzzy membership sets for inputs/outputs and fuzzy rules should be well defined in order to make a successful robotics navigation.

The work done in [4] defines a single fuzzy system with reinforcement learning. The system is applied on a Khepera robot and shows its power and flexibility in solving robotic issues in detecting a target in a changing physical environment.

The system in [5] also defines a single fuzzy system with reinforcement learning. The system needs sufficient training and also fine training using the reinforcement learning to determine the membership functions for the input and output variables. After that, the mobile robot is able to perform collision-free navigation. The system is applied in a virtual environment.

The paper work in [6] presents a dual fuzzy controller with reinforcement learning. The first controller is to follow target and avoid obstacles, while the second controller is using a wall-following behavior to escape concave traps. A simulated robot is tested in a complicated unknown environment for training on obstacle-avoidance and target-approaching.

A single fuzzy system is used in [7] to apply both obstacle avoidance and target seeking behaviors. For demonstrations of the effectiveness of the proposed fuzzy logic based controller, simulations using a mobile robot simulator are performed, to move a robot from a given current position to a desired goal position in various environments.

The work in [8] presents a dual fuzzy controller: one for angular velocity and one for linear velocity of the robot. A multi agent model simulation for robot navigation in dynamic environments is done. The robot only used ultrasonic sensors in addition to the suggested fuzzy system in order to achieve its goal without collision.

A dual fuzzy system is used in [9] for robot navigation. The first fuzzy system is to move to the goal, and the second is to avoid obstacles. The work is implemented on both simulation and real-time. The real-time implementation used Khepera II mini robot. Both implementations gave good performances.

An elaborated fuzzy system in [10] is presented using four separate fuzzy controllers in a hierarchical manner: reach target, avoid obstacle, escape trap with right wall follow, and escape trap with left wall follow. The system is applied to a Khepera robot, and used in a re-active manner without an environment map. Because of the missing map, the system is not able to find an optimized path.

A dual system fuzzy logic controller is defined in [11]. The two systems are target reach and obstacle avoidance. It is using the wireless control approach to control a swarm of wheeled mobile robots Scout-II in a warehouse with static and dynamic obstacles. The environment is dynamic and unknown. Smooth paths were generated using this method.

The work in [12] introduced dual system fuzzy logic controller with reinforcement learning (Q-learning). The two systems are goal seeking and wall following. The effectiveness of the Q-leaning optimization algorithm is verified by simulation.

Another work in [13] discusses a fuzzy controller with dual system: orientation and obstacle avoidance. The work is applied in simulation and actual robot implementation for unknown indoor environment with different obstacle course. Both tests show good results.

The paper in [14] shows a single fuzzy controller system for robot navigation in unknown environment. The fuzzy logic controller takes the input from the laser sensor of the robot and gives the change in the angular velocity as output to the robot to avoid the obstacle. Simulink is used to model the system in a simulation environment.

The research conducted in [15] explains a single fuzzy logic system to navigate mobile robot in an unknown dynamic environment. The fuzzy controller was used to assess the positions and select better steps that decrease the overall path length, together with avoiding collision with obstacles. It also keeps the robot away from local minimums. In order to analyze the performance of the proposed algorithm in the path planning's field, a simulation framework was designed using Matlab.

A paper in [16] described a single fuzzy logic system for robot navigation in real environments using landmark detection. A mobile robot was used to move along the hallways inside a building.

Another work in [17] explains a single fuzzy logic system to navigate in an unknown environment with-out hitting obstacles or other robots. The simulations and tests on actual robots have demonstrated that the system works correctly.

A dual system fuzzy logic controller is introduced in [18]. The dual system contains goal reaching and obstacle avoidance. High level global planning and Low-level reactive control were used. In high-level planning, the robot motion is determined and a minimum-cost path is followed to reach the target. In low-level reactive control, the robot uses current sensory information to change the motion direction reacting to unforeseen obstacles.

The authors in [19] presented three separate fuzzy controllers for mobile robot navigation in unknown environment using reinforcement learning. The three controllers are: goal seeking, wall following, and obstacle avoidance. The process of learning based on Q-learning consists to improve the mobile robot performance.

Simulation results on real robot show the system's effectiveness.

The work done in [20] shows a dual fuzzy controller system with reinforcement learning. The two systems are goal seeking and obstacle avoidance. The suggested algorithm is used for driving E-puck robot in an obstacle filled environment.

Another work done in [21] displays a single fuzzy controller system for mobile robot navigation to avoid dynamic and static obstacles. The navigation method is inspired from the Vector Field Histogram method. It is using a reactive approach for mobile robot control. Testing is done only in simulation environment.

Furthermore, the publication in [22] also explains a single fuzzy controller system for mobile robot navigation, but in static environments. The robot can reach its target in the shortest path with obstacle avoidance. The efficiency of this method is verified in simulation and experimental results. The MATLAB toolbox is used for software implementation.

Nadour *et al.* [23] presents a dual fuzzy controller system for mobile robot navigation based optical flow approaches. The two separate fuzzy systems are goal seeking and obstacle avoidance. The controller uses video acquisition and image processing algorithm. The proposed approach is simulated in 3D environment using VRML library.

The paper [24] presents a single fuzzy controller system for autonomous mobile robot navigation. The controller is designed on Matlab/Simulink environment. Simulations are presented to verify the performance of the fuzzy logic controller, which proves to be good.

Another single fuzzy controller system is shown in [25] for robot navigation with detecting and avoiding static and dynamic obstacles. Simulink is used to present simulation results.

Also, another single adaptive fuzzy controller system is shown in [26] for omnidirectional mobile robots with trajectory tracking. Simulation works demonstrating the validity of the proposed design are also presented.

On the other hand, the work suggested in [27] uses evolutionary programming method and reactive control for a non-holonomic mobile robot with a dual fuzzy controller system with reinforcement learning. The two systems are reach endpoint and obstacle avoidance. Simulation is done and shows that the designed fuzzy controller achieves effectively any movement control of the vehicle to its goal without any collision.

Another work in [28] suggests a single fuzzy system to control autonomous robot with obstacle deviation. A virtual simulation is done using MATLAB and the results are satisfactory. However, actual implementation shows weaknesses in obstacle detection for ultra-sonic sensors outside angles of actuation.

In the research [29], two fuzzy methods were implemented and compared together for optimum path planning of mobile robot in unknown static and dynamic environments. The proposed algorithms are successfully verified through both simulations and real-time experiments in the different environments. One of the fuzzy methods performed better than the other.

The authors in [30] explain fuzzy logic controllers design for omnidirectional mobile robot navigation (Robotino). The design includes three separate fuzzy controller systems: one for linear velocity, one for angular velocity, and one for obstacle avoidance. Simulation and experimental tests are performed for one, two and three obstacles, and show that the robot has good performance and efficiency to navigate in an unstructured and unknown obstacles environment.

The work done in [31] presents a single fuzzy controller system for navigation of humanoid robot in obstacle prone zone. The controller is tested on a NAO humanoid robot in V-REP simulation platform and also in a real experimental platform under laboratory conditions. The humanoid robot was successful in avoiding the obstacles and reach the target location in an optimized path.

Another work in [32] is based on four fuzzy controller systems: go to target, avoid obstacle, wall follow, and wander. The experimental results demonstrate that the proposed robot navigation system performs better than the conventional approach in terms of consistent motion and safe navigation.

An intelligent mobile manipulator navigation is proposed in [33] using dual fuzzy controller system: obstacle avoidance and join virtual target. A simulation is done and simulation results illustrate the efficiency of the proposed controller.

A work is done in [34] to address trajectory planning and navigation control problems for a mobile robot. An adaptive PSO (Particle Swarm Optimization) motion algorithm is developed using a penalty-based methodology. Simulation results are presented through the MATLAB environment in addition to real-time experiments conducted in a similar workspace. The results showed improvements over other techniques.

Another work in [35] designed an autonomous navigation robot platform. Based on the particle filtering mapping algorithm and using 2D LiDAR and an inertial measurement unit sensors, simultaneous localization, mapping, and autonomous navigation were realized. The experimental results showed that the platform effectively mapped the unknown environment and realized autonomous navigation through the proposed control algorithm.

III. Definitions and Model Requirements

A. *Fuzzy Logic Definition*

In mathematical logic, Boolean algebra is a branch of algebra and differs from elementary algebra in two ways. First, the values of variables are Booleans, usually denoted by 1 and 0, but in elementary algebra the values of variables are numbers. Second, Boolean algebra uses logical operators such as conjunction (and), disjunction (or), and negation (not). Elementary algebra, on the other hand, uses arithmetic operators such as addition, multiplication, subtraction, and division. Boolean algebra is thus a formal way of describing logical operations in the same way that elementary algebra describes numerical operations.

Fuzzy logic is a computational approach based on "degree of truth" rather than the usual "true or false" (1 or 0) Boolean logic on which modern computers are based. The idea of fuzzy logic was first developed in the 1960s by Lotfi Zadeh at the University of California, Berkeley. A fuzzy model or set is a mathematical means of representing fuzzy and imprecise information. These models are able to perceive, represent, manipulate, interpret and use ambiguous and uncertain data and information. Fuzzy logic can be thought of as the way thinking really works. Binary or Boolean logic is just a special case of that. Fuzzy logic uses a degree of truth ranging from 0 to 1 as a mathematical model of vagueness [36].

The fuzzy set as is a class of objects with a continuous of grade of membership. The transition from membership to non-membership in a subset of reference set is gradual rather than abrupt. An example of a fuzzy set membership function is shown in Fig. 1. The definition of "fit" membership function depends on the "weight" input. As the weight increases, fit gradually goes up from zero to one then goes back to zero.

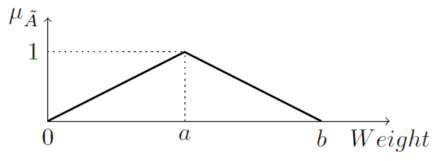

Figure 1. Sample fuzzy membership function.

To generalize, using the same axis "weight" input, and with three membership functions: underweight, fit, and overweight. We keep the same definition for "fit" function. We define "underweight" function starting from 1 for zero weight, going down to zero as weight increases. We define "overweight" function starting from zero for zero weight, going up to 1 as weight increases. The overall fuzzy set with its 3 membership functions is shown in Fig. 2.

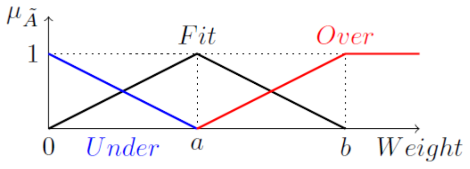

Figure 2. Sample fuzzy membership set with 3 functions.

Fuzzy Systems are composed of three parts [37]:

- Fuzzification: Fuzzify all input values into fuzzy membership functions. The fuzzification procedure maps the crisp input values to the fuzzy terms with membership values between 0 and 1.
- Fuzzy Rules Evaluation or inference mechanism: Execute all applicable rules in the rule base to compute the fuzzy output functions. This mechanism is responsible for decision-making in the fuzzy controller using approximate reasoning. The rule base is essential for the controller, which stores the rules directing the input and output relationship of the proposed controller.
- -Defuzzification: The defuzzification procedure maps the fuzzy output from the fuzzy rule's evaluation to a crisp output. For this, several methods are available such as Center of Gravity and Mean of Maxima. A basic fuzzy system is shown in Fig. 3.

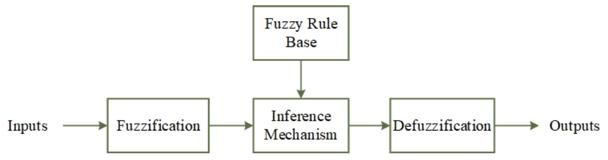

Figure 3. Basic fuzzy system.

B. *Model Requirements and Assumptions*

The model environment for solving a path planning can be reduced to an undirected connected graph G = V, E, L, where V is the set of nodes, E is set of edges, and L represents the effective length between two nodes. An example of the work environment is shown in Fig. 4. The requirements and assumptions are as follows:

- The path width between two nodes can only accommodate one mobile robot at a time.
- The running speed is constant and same for all mobile robots under normal operation condition. It is assumed to be 1m/s.
- The block's distance is 1 meter. The robot needs 1 second to move for one block.
- The robot needs 1 second to rotate by 90 degrees to any direction.
- Mobile robots' direction is considered at initial starting point.
- The path taken into consideration can be single way from start to target.
- The main robot only knows the location of other robots when it arrives to any neighboring cell.

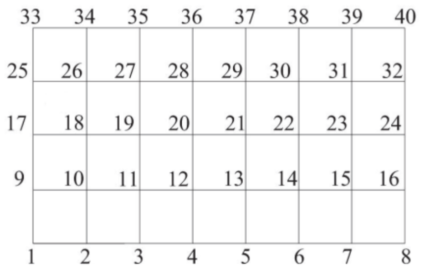

Figure 4. Environment grid used.

IV. Proposed Fuzzy Model

The proposed fuzzy model consists of three main parts, namely the fuzzy system 1: Reach Goal, the fuzzy system 2: Avoid Obstacle, and the fuzzy system 3: Escape Cul-

De-Sac. At the beginning, the Reach Goal fuzzy system is activated. It tries to guide the robot towards its target until success or reaching an obstacle on the way. In the latter situation, the control is given to the Avoid Obstacle fuzzy system which tries to change the path in order to deviate from the faced obstacle. This system stays active until the path is clear, and there-fore returning to the initial system. In case the path is not clear, and there is no way to deviate from the obstacle without hitting another one, then the Escape Cul-De-Sac fuzzy system is activated. This system tries to get around all the obstacles until the path is clear, returning to the previous fuzzy system. The following flowchart in Fig. 5 shows the overall workflow with the different fuzzy system parts interacting together.

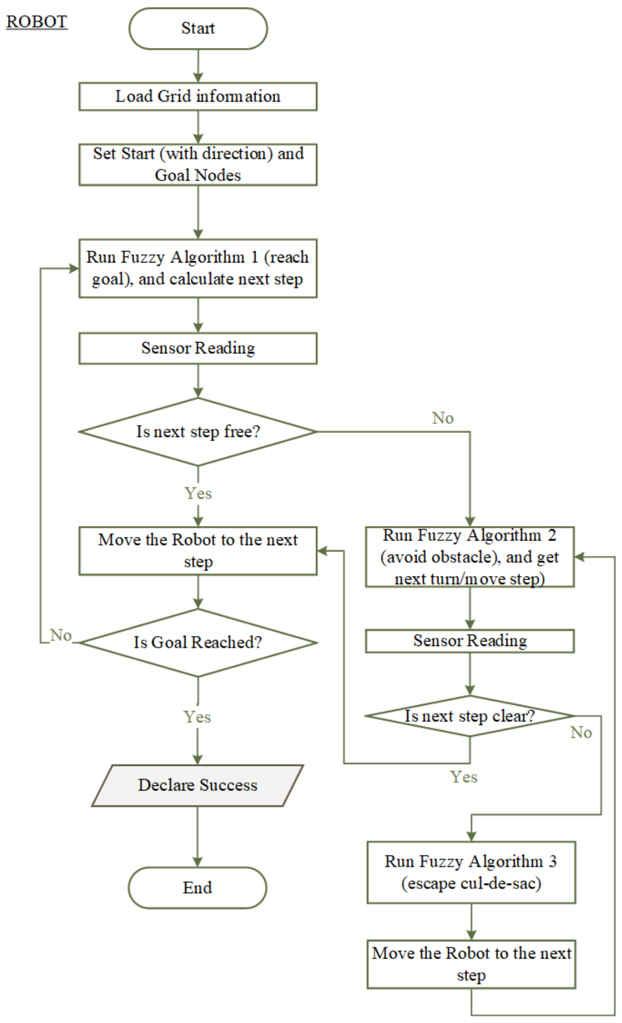
ROBOT
Start
Load Grid information
Set Start (with direction) and Goal Nodes
Run Fuzzy Algorithm 1 (reach goal), and calculate next step
Sensor Reading
Is next step free?
No
Yes
Move the Robot to the next step
Run Fuzzy Algorithm 2 (avoid obstacle), and get next turn/move step)
No
Is Goal Reached?
Sensor Reading
Yes
Is next step clear?
No
Yes
Declare Success
Run Fuzzy Algorithm 3 (escape cul-de-sac)
End
Move the Robot to the next step

Figure 5. Proposed fuzzy model.

The three fuzzy systems share the same inputs/outputs, with the second and third system having extra inputs. All take into consideration the starting or current point, and the goal point. By working on two-dimensional axes x and y, we use the difference delta(x) and delta(y) as inputs.

DisX = delta(x) = current(x) - goal(x)

DisY = delta(y) = current(y) - goal(y)

The inputs DisX/DisY are defined on a fuzzy set with three membership values: Zero, Positive, Negative. Zero means it is on the goal. It is a singleton. Positive means that the goal is far but in the same robot's direction. Negative means that the goal is far but in the opposite of robot's direction. Fig. 6 shows the corresponding fuzzy set with the three membership values, including one singleton.

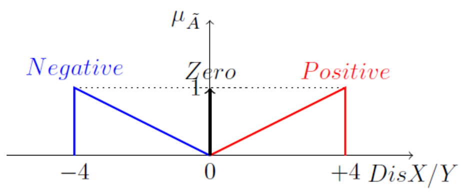

Figure 6. Fuzzy set for inputs DisX/DisY (distance to goal in X/Y directions).

Another input is the Robot's Direction (Dir). It defines one of the four possible current directions of the robot: 0-Left, 1-Up, 2-Right, 3-Down. Fig. 7 shows the corresponding fuzzy set and the four singletons. As seen in the figure, the membership values are singletons.

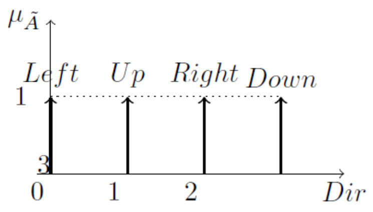

Figure 7. Fuzzy set for input Dir (current robot's direction).

The outputs of the system are MovX and MovY. MovX is defined as the movement of the robot in the X direction, while MovY is defined as the movement of the robot in the Y direction. The outputs MovX/MovY are defined on a fuzzy set with three membership values: Zero, Forward, backward. Zero means that the there is no movement on the corresponding axis. Forward means there is movement on the same direction of the robot. Backward means there is movement on the opposite direction of the robot. Fig. 8 shows the corresponding fuzzy set for the output movement in both x/y directions.

Another system's output is Rot. Rot is defined as the rotation of the robot. Its fuzzy set contains three membership values: Zero, Left, Right. Zero means that there is no rotation. Right means that the robot will rotate to its right. Left means that the robot will rotate to its left. Fig. 9 shows the corresponding fuzzy set for the rotation with three membership values.

The three fuzzy system parts are described in detail as follows.

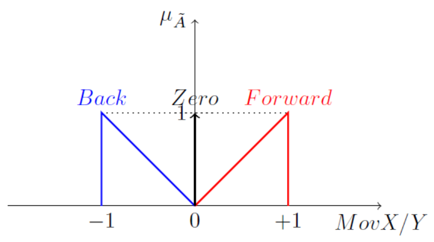

Figure 8. Fuzzy set for outputs MovX/MovY (robot's movement in X/Y directions).

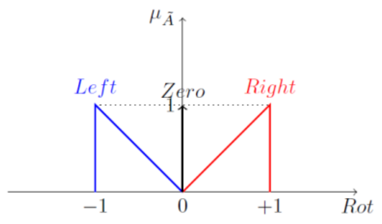

Figure 9. Fuzzy set for output Rot (robot's rotation).

A. *Fuzzy System 1: Reach Goal*

The reach goal system is simply trying to approach from target until it is reached.

For each iteration, the system checks the above in-puts (namely DisX/DisY/Dir), then applies the fuzzy model three parts: fuzzification, inference rules, then defuzzification. The outcome will be the new robot's move or rotation (MovX/MovY/Rot). This move is checked for validity. If it is valid, that is there is no obstacle is on the way, then it is carried out, otherwise the system switches to Fuzzy System 2: Avoid Obstacle. The system will stop when the robot reaches the target.

A few sample inference rules for the fuzzy system 1 are shown next.

If DisX>0 and Dir=Right then MovX=Forward
If DisX>0 and Dir=(Left/Up) then Rot=Right
If DisX>0 and Dir=Down then Rot=Left

The system's algorithm is depicted in Algorithm 1.

Algorithm 1 Fuzzy System 1: Reach Goal

```
1. Select the Start node V and the Goal node G.
2. Make current node i=V, Dist(i)=0, Dist is the traveled
   distance so far from V.
   Direction=initial direction of robot located in V
   (right/left/up/down).
3. Loop from step 4 to 9 to reach goal G.
4. Check if i=G, then exit loop and declare success.
5. Get distance from i to G in both directions x & y:
   DisX = x[G] – x[i]
   DisY = y[G] – y[i]
6. Call Fuzzy Model according to the chosen method:
   Center of Gravity or Mean of Maxima.
7. Check new position for obstacles.
   a. If there is no obstacle, advance to new location or
      direction. i = new location.
   b. If there is an obstacle, then exit algorithm and
      activate Fuzzy System 2: Avoid Obstacle.
8. End of Algorithm.
```

B. *Fuzzy System 2: Avoid Obstacle*

The avoid obstacle system is trying to avoid the detected obstacle on the robot's next move. Similarly, to the fuzzy system 1, it also takes into consideration the starting or current point, and the goal point. But it adds the difference between the current robot's location and the impeding obstacle.

$$ObX = Obstacle(x) - current(x)$$
$$ObY = Obstacle(y) - current(y)$$

The inputs ObX/ObY are defined on a fuzzy set with three membership values: Zero, Positive, Negative. The membership values are singletons: 0, +1, −1. Zero means that the obstacle is on the same level of X/Y axis as the robot. Positive means that the obstacle is on the X/Y axis as the robot plus one. Positive means that the obstacle is on the X/Y axis as the robot minus one. The following Fig. 10 shows the corresponding fuzzy set.

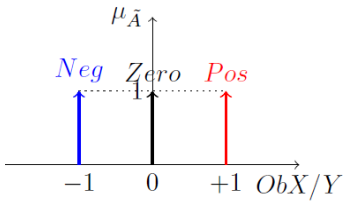

Figure 10. Fuzzy set for inputs ObX/ObY (distance to obstacle in X/Y directions).

For each iteration, the system checks the above distances, then applies the fuzzy model three parts: fuzzification, inference rules, then defuzzification. The outcome will be the new robot's move. This move is checked for validity. If it is valid, that is no obstacle is on the way, then it is carried out, otherwise the system switches to Fuzzy System 3: Escape Cul-De-Sac. The system will switch back to Fuzzy System 1 (Reach Target) when the obstacle is no longer a threat.

A few sample inference rules for the fuzzy system 2 are shown next.

If DisX>0,ObY=0,ObX=1,Dir=R,DisY>=0 then Rot=L
If DisX>0,ObY=0,ObX=1,Dir=R,DisY<0 then Rot=R
If DisX>0,ObY=0,ObX=1,Dir=U then MovY=F

The system's algorithm is depicted in Algorithm 2.

Algorithm 2 Fuzzy System 2: Avoid Obstacle

```
1. For current node i, goal node G, and the adjacent
   Obstacle on the path.
2. Get distance from i to G in both directions x & y:
   DisX = x[G] – x[i]
   DisY = y[G] – y[i]
3. Get distance from i to Obstacle in both directions x &
   y:
   ObX = x[Obstacle] - x[i]
   ObY = y[Obstacle] - y[i]
4. Call Fuzzy Model according to the chosen method:
   Center of Gravity or Mean of Maxima.
5. Check new position for obstacles.
   a. If there is no obstacle, advance to new location or
      direction. i = new location.
   b. If there is an obstacle, then exit algorithm and
      activate Fuzzy System 3: Escape Cul-De-Sac.
6. Repeat steps 2 to 5 until Obstacle is not visible.
7. Return to Fuzzy System 1: Reach Goal.
8. End of Algorithm.
```

C. *Fuzzy System 3: Escape Cul-De-Sac*

The escape cul-de-sac is trying to get out of the trap obstacles in the next few moves, then get back to the reach target system. It takes into consideration the starting or current point, and the goal point. It also takes into consideration the difference between the current robot's location and the obstacle detected previously in Avoid Obstacle system.

ObPX = PreviousObstacle(x) - current(x)

ObPY = PreviousObstacle(y) - current(y)

Furthermore, it checks the difference between the current robot's location and the new obstacle detected while trying to avoid the first obstacle.

ObX = Obstacle(x) - current(x)

ObY = Obstacle(y) - current(y)

For each iteration, the system checks the above distances, then applies the fuzzy model three parts: fuzzification, inference rules, then defuzzification. The outcome will be the new robot's move. This move is carried out and repeated until the robot is no longer trapped. The system will switch back to Fuzzy System 1 (Reach Target) in that case.

The system's algorithm is depicted in Algorithm 3.

Algorithm 3 Fuzzy System 3: Escape Cul-De-Sac

```
1. For current node i, goal node G, and the adjacent
   Obstacle and previous obstacle PrevObs on the path.
2. Get distance from i to G in both directions x & y:
   DisX = x[G] – x[i]
   DisY = y[G] – y[i]
3. Get distance from i to Obstacle in both directions x &
   y:
   ObX = x[Obstacle] - x[i]
   ObY = y[Obstacle] - y[i]
4. Get distance from i to PrevObs in both directions x
   & y:
   ObPX = x[PrevObs] - x[i]
   ObPY = y[PrevObs] - y[i]
5. Call Fuzzy Model according to the chosen method:
   Center of Gravity or Mean of Maxima.
6. Advance to new position or direction. i = new
   location.
7. Repeat steps 2 to 8 until both Obstacle and PrevObs
   are not visible.
8. Return to Fuzzy System 1: Reach Goal.
9. End of Algorithm.
```

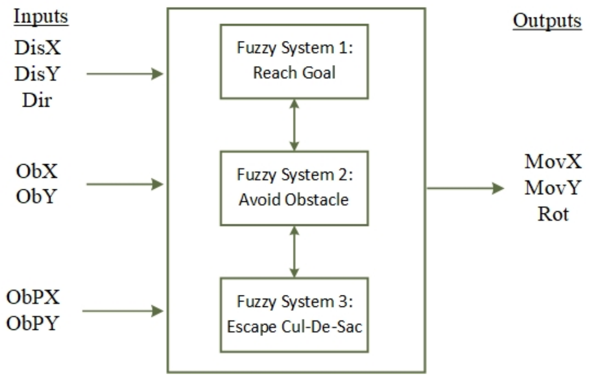
Inputs
DisX
DisY
Dir
ObX
ObY
ObPX
ObPY
Fuzzy System 1: Reach Goal
Fuzzy System 2: Avoid Obstacle
Fuzzy System 3: Escape Cul-De-Sac
Outputs
MovX
MovY
Rot

Figure 11. Block diagram for fuzzy input/output module.

A block digram for the input/output module is shown in the next Fig. 11. As seen in the figure, all inputs and outputs for the three sub systems are listed. The first set of inputs (DisX/DisY/Dir) is used in all sub systems. ObX/ObX are used in the last two only. ObPX/ObPY are only used in Escape Cul-De-Sac system only.

V. Experiments and Discussion

Our proposed system was implemented as a simulation and applied to Automated Storage and Retrieval Systems (ASRS). To verify the effectiveness of the proposed algorithm, extensive simulations for a range of scenarios for automated storage are carried out. The developed program was written in C# language and run on Window 10 Operating System.

A. Application to AS: Scenario 1 - No Obstacles

The first experiment was conducted by having no obstacles. Two simulations were conducted. In both simulations, the starting point is node 9 and the goal is node 31. The simulation results are shown in Fig. 12, where circles indicate source and goal, squares indicate obstacles, and the arrows indicate the followed path. For the simulations, Center of Gravity defuzzification and Mean of Maxima defuzzification were implemented with the initial robot's direction to the right. The main robot changed direction at positions 14 and 30 to reach target. Only the Fuzzy System 1 (Reach Goal) was active during all the path. Table I summarizes the position with respect to time for the robot with ten steps time and only one active fuzzy system.

B. Application to AS: Scenario 2 -Static Obstacles

The second experiment was conducted by having three static obstacles at locations 5, 13, and 21. Two simulations were conducted. In both simulations, the starting point is node 9 and the goal is node 31. The simulation results are shown in Fig. 13. For the simulations, Center of Gravity defuzzification and Mean of Maxima defuzzification were implemented with the initial robot's direction to the right. In the Fuzzy model, the main robot changed direction at positions 12 to avoid obstacles, then at position 28 to reach target. The fuzzy system 2 (Avoid Obstacles) was activated on nodes 12, 20, 28. The Fuzzy System 1 (Reach Goal) was active during all the remainder of the path. Table II summarizes the position with respect to time for the robot, showing the active fuzzy system at that point. As seen in the table, two fuzzy systems were active at different times.

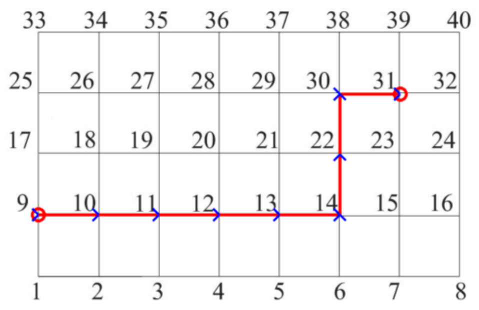

Figure 12. Scenario 1- no obstacles.

TABLE I. ROBOT'S MOVEMENTS - SCENARIO 1

Time	**Robot's Location**	**Robot's Direction**
0	9	Right
1	10	Right
2	11	Right
3	12	Right
4	13	Right
5	14	Right
6	14	Turn Up
7	22	Up
8	30	Up
9	30	Turn Right
10	31	Right

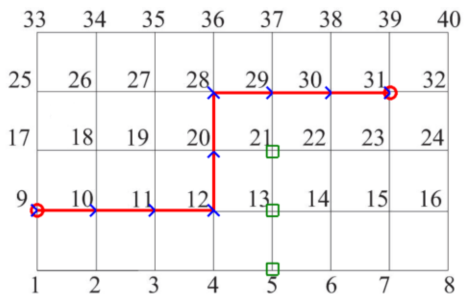

Figure 13. Scenario 2- three static obstacles.

TABLE II. ROBOT'S MOVEMENTS - SCENARIO 2

Time	**Robot's Location**	**Robot's Direction**	**Fuzzy System**
0	9	Right	1
1	10	Right	1
2	11	Right	1
3	12	Right	1
4	12	Turn Up	2
5	20	Up	2
6	28	Up	2
7	28	Turn Right	1
8	29	Right	1
9	30	Right	1
10	31	Right	1

C. *Application to AS: Scenario 3 –Cul-De-Sac*

The third experiment was conducted by having five static obstacles forming a Cul-De-Sac at locations 4, 5, 14, 20, and 21. Two simulations were conducted. In both simulations, the starting point is node 9 and the goal is node 31. The simulation results are shown in Fig. 14. For the simulations, Center of Gravity de-fuzzification and Mean of Maxima defuzzification were implemented with the initial robot's direction to the right. In Fuzzy model, the main robot changed direction 180 degrees at position 13 to escape Cul-De-Sac, went back to position 12, changed position up then back to position 11 then changed position at 11 and 27 to reach target. Fuzzy Systems 2 (Avoid Obstacle) and 3 (Escape Cul-De-Sac) were activated on nodes 13, 12, 11, 19, and 27. The Fuzzy System 1 (Reach Goal) was active during all the remainder of the path. Table III summarizes the position with respect to time for the robot, showing the active fuzzy system at that point. All three fuzzy systems were active at different times.

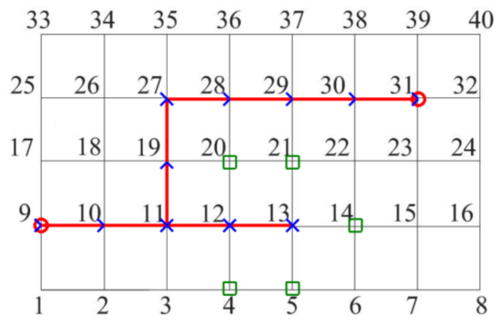

Figure 14. Scenario 3- cul-de-sac.

TABLE III. ROBOT'S MOVEMENTS - SCENARIO 3

Time	**Robot's Location**	**Robot's Direction**	**Fuzzy System**
0	9	Right	1
1	10	Right	1
2	11	Right	1
3	12	Right	1
4	13	Right	2
5	13	Turn Up	3
6	13	Turn Left	3
7	12	Left	3
8	12	Turn Up	3
9	12	Turn Left	3
10	11	Left	3
11	11	Turn Up	3
12	19	Up	3
13	27	Up	2
14	27	Turn Right	1
15	28	Right	1
16	29	Right	1
17	30	Right	1
18	31	Right	1

D. *Application to AS: Scenario 4 – Dynamic Obstacles*

The fourth experiment was conducted by having two mobile obstacles. One mobile obstacle started on point 6, then moved to 5 then stopped at 13. Another mobile obstacle started on point 36, then moved to 28 then stopped at 20. Two simulations were conducted. In both simulations, the starting point is node 9 and the goal is node 31. The simulation results are shown in Fig. 15. For the simulations, Center of Gravity defuzzification and Mean of Maxima defuzzification were implemented with the initial robot's direction to the right. In both Fuzzy models, the main robot changed direction at positions 12 then 11 to avoid obstacles, then on position 27 to reach target. Fuzzy System 3 (Escape Cul-De-Sac) was activated on nodes 12, 11, 19, 27, 28, 29. The Fuzzy System 1 (Reach Goal) was active during all the remainder of the path. Table IV summarizes the position with respect to time for the robot, showing the active fuzzy system at that point. Only fuzzy systems 1 and 3 were active at different times.

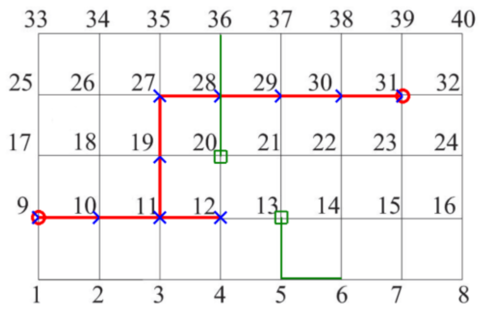

Figure 15. Scenario 4- dynamic obstacles.

TABLE IV. ROBOT'S MOVEMENTS - SCENARIO 4

Time	Robot's Location	Robot's Direction	Fuzzy System
0	9	Right	1
1	10	Right	1
2	11	Right	1
3	12	Right	1
4	12	Turn Up	3
5	12	Turn Left	3
6	11	Left	3
7	11	Turn Up	3
8	19	Up	3
9	27	Up	3
10	27	Turn Right	3
11	28	Right	3
12	29	Right	3
13	30	Right	1
14	31	Right	1

TABLE V. COMPARISON OF THE 4 SCENARIOS

Scenario	Obstacles Number	Dynamic/ Static Obstacles	Ideal Time	Actual Time	Activated Fuzzy Systems
Scenario 1: No Obstacles	0	None	10s	10s	1
Scenario 2: Static Obstacles	3	Static	10s	10s	1, 2
Scenario 3: Cul-De-Sac	5	Static	10s	18s	1, 2, 3
Scenario 4: Dynamic Obstacles	2	Dynamic	10s	14s	1, 3

The simulated 4 scenarios are compared in the Table V. The used criteria include the number of obstacles, their type, the needed ideal time to reach the target, the actual spent time to reach the target using fuzzy module, and the activated fuzzy subsystems. All the experiments were successful. The first scenario was the only one to have an actual output equivalent to the ideal one. The reason is that there were no obstacles involved. The other scenarios lagged behind the ideal case, mainly due to the fact that the robot has no prior knowledge of the locations for the other robots. This makes the other techniques like A* or Dijkstra not feasible for solving the navigation problem.

E. Comparison of our Fuzzy Model with APSO Model

In order to compare our model with an existing work, a fifth experiment was conducted by having one dynamic obstacle and four static obstacles. The mobile obstacle started on point 10, then moved to 11 then stopped at 12. The four static obstacles were at locations 6, 14, 22, and 30. The starting point is node 1 and the goal is at node 24. We compared our model with another model based on a technique called APSO [34]. We also assume that the distance between any 2 consecutive nodes is one meter, the robot's speed is 1m/s, and turning one right angle (90 degrees) will take 1 second. We got the following results:

- Our model performed well. It finished the track in 16 steps (seconds): 13 movements and 3 right angle turns. Fig. 16 shows the Fuzzy Model's used path.
- The other APSO model had a slightly better performance: 15 seconds instead of 16 seconds as in our model, so 6.67% better performance. In the APSO technique, the robot turned seven times 45 degrees, moved straight 3 moves, and 6 diagonals. Fig. 17 shows the APSO model's used path.
- The reason behind the difference of performance is that the other model relied on moving in diagonals and near obstacles. Our model had a restriction of moving in empty nodes and turning at right angles only.
- Table VI shows the comparison between the two models.

TABLE VI. COMPARISON OF THE 2 MODELS

Scenario	Obstacles Number	Dynamic/ Static Obstacles	Fuzzy Model Time	APSO Time	Diff. (%)
Scenario 5: Dynamic and Static Obstacles	5	1 Dynamic, 4 Static	16s	15s	6.67%

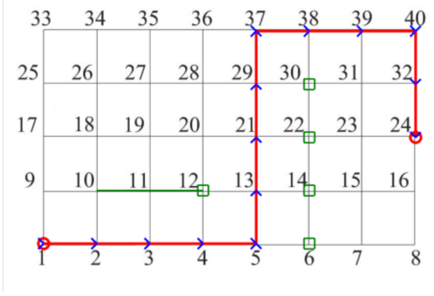

Figure 16. Scenario 5- Using Our Fuzzy Model.

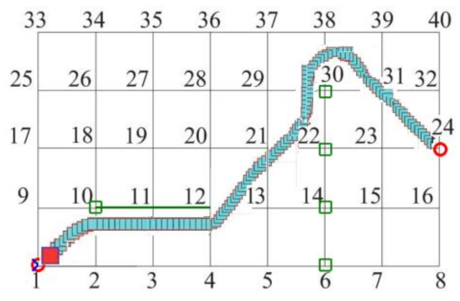

Figure 17. Scenario 5- Using APSO Technique.

VI. Conclusion

As a conclusion, a fuzzy model to handle robotic navigation in an automated storage retrieval system was suggested and simulated. The model contains three fuzzy subsystems: Reach Target, Avoid Obstacle, and Escape Cul-De-sac. The simulations were done using both Center of Gravity and Mean of Maxima for de-fuzzification. Both methods showed exactly the same results.

Four scenarios were simulated and compared. One without obstacles, one with static obstacles, one with static obstacles forming a Cul-De-Sac, and one with dynamic obstacles. The used criteria included the number of obstacles, their type, the needed ideal time to reach the target, the actual spent time to reach the target using fuzzy module, the activated fuzzy sub-systems, and the experiment's outcome. The results showed that the Fuzzy Logic System was capable of handling all the different scenarios with success as shown in a table. The only problem was that complicated scenarios caused some delay in order to reach the target.

An additional scenario was created and simulated using our Fuzzy Model and APSO technique. The comparison showed almost similar results in terms of performance and efficiency.

The results are promising, especially facing dynamic obstacles which are hard to detect using conventional shortest path methods with static memorized map . We intend to implement this system on a real robot in a real environment. Furthermore, reinforcement learning can be added in a future work to fine tune the fuzzy model sets and have a better adjustment to the specificities of the used warehouse map.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Chadi F. Riman and Pierre E. Abi-Char developed the theory by contributing to the design of this research. Both authors wrote the manuscript. Both authors discussed the results and commented on the manuscript. Chadi Riman performed the software simulation; Pierre AbiChar revised and refined the article. Both authors approved the final version.

References

[1] S. Permana, *et al.*, "Comparative analysis of pathfinding algorithms A *, Dijkstra, and Bfs on maze runner game," *International Journal Of Information System & Technology*, vol. 1, no.2, 2018.

[2] P. E. Abi-Char, and C. Riman, "A collision-free path planning algorithm for non-complex Asrs using heuristic functions," in *2021 44th International Conference on Telecommunications and Signal Processing (TSP)*, Brno, Czech Republic, 2021, pp. 52–57, doi: 10.1109/TSP52935.2021.9522682.

[3] C. Riman, and P. E. Abi-Char, "A Priority-based Modified A∗ Path Planning Algorithm for Multi-Mobile Robot Navigation," *2022 19th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)*, Mexico City, Mexico, 2022, pp. 1–6, doi: 10.1109/CCE56709.2022.9976025.

[4] 1- K. C. Tan, K. K. Tan, T. H. Lee, S. Zhao, and Y. J. Chen, "Autonomous robot navigation based on fuzzy sensor fusion and reinforcement learning," *Proceedings of the IEEE Internatinal Symposium on Intelligent Control*, Vancouver, BC, Canada, 2002, pp. 182–187, doi: 10.1109/ISIC.2002.1157759.

[5] Y. Cang, Y. Nelson, and W. Danwei, "A fuzzy controller with supervised learning assisted reinforcement learning algorithm for obstacle avoidance," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 33, no. 1, pp. 17–27, Jan 2003. 10.1109/TSMCB.2003.808179.

[6] Y. Duan, and X. Hexu, "Fuzzy reinforcement learning and its application in robot navigation," *2005 International Conference on Machine Learning and Cybernetics*, vol. 2, pp. 899–904, Aug 2005, doi: 10.1109/ICMLC.2005.1527071.

[7] S. M. Raguraman, D. Tamilselvi, and N. Shivakumar, "Mobile robot navigation using Fuzzy logic controller," *2009 International Conference on Control, Automation, Communication and Energy Conservation*, Perundurai, India, 2009, pp. 1–5.

[8] E. Ayari, S. Hadouaj, and K. Ghedira, "A fuzzy logic method for autonomous robot navigation in dynamic and uncertain environment composed with complex traps," *2010 Fifth International Multi-conference on Computing in the Global Information Technology*, Valencia, Spain, 2010, pp. 18–23, doi: 10.1109/ICCGI.2010.47.

[9] N. Yousfi, C. Rekik, M. Jallouli, and N. Derbel, "Optimized fuzzy controller for mobile robot navigation in a cluttered environment," *2010 7th International Multi- Conference on Systems, Signals and Devices*, Amman, Jordan, 2010, 10.1109/SSD.2010.5585508.

[10] S. A. -L. El-Teleity, Z. B. Nossair, H. M. Abdel-Kader Mansour, and A. TagElDein, "Fuzzy logic control of an autonomous mobile robot," *2011 16th International Conference on Methods & Models in Automation & Robotics*, Miedzyzdroje, Poland, 2011, pp. 188–193, doi: 10.1109/MMAR.2011.6031342.

[11] M. Faisal, R. Hedjar, M. A. Sulaiman, K. Al-Mutib, "Fuzzy logic navigation and obstacle avoidance by a mobile robot in an unknown dynamic environment," *International Journal of Advanced Robotic Systems*, vol. 10, no. 1, Jan 2013, doi:10.5772/54427.

[12] L. Cherroun, and M. Boumehraz, "Intelligent systems based on reinforcement learning and fuzzy logic approaches, "Application to mobile robotic"," *2012 International Conference on Information Technology and e-Services*, Sousse, Tunisia, 2012, 10.1109/ICITeS.2012.6216661.

[13] J. Johnson, and D. J. Godwin, "Indoor navigation of mobile robot using fuzzy logic controller," *2015 3rd International Conference on Signal Processing, Communication and Networking (ICSCN)*, Chennai, India, 2015, pp. 1–7, doi: 10.1109/ICSCN.2015.7219873.

[14] N. Kumar, M. Takács, and Z. Vámossy, "Robot navigation in unknown environment using fuzzy logic," *2017 IEEE 15th International Symposium on Applied Machine Intelligence and Informatics (SAMI),* Herl'any, Slovakia, 2017, doi: 10.1109/SAMI.2017.7880317.

[15] F. Kamil, and M. Y. Moghrabiah, "Multilayer decision-based fuzzy logic model to navigate mobile robot in unknown dynamic environments," *Fuzzy Information and Engineering*, vol. 14, no. 1, pp. 51–73, 2022, doi: 10.1080/16168658.2021.2019432.

[16] E. T. Lee, “Applying fuzzy logic to robot navigation,” *Kybernetes*, vol. 24, no. 6, pp. 38–43, 1995.

[17] D. R. Parhi, “Navigation of mobile robots using a fuzzy logic controller,” *Journal of Intelligent and Robotic Systems*, vol. 42, pp. 253–273, 2005. https://doi.org/10.1007/s10846-004-7195-x.

[18] T. Hong, N. Danial, and B. Karasfi, “Application of fuzzy logic in mobile robot navigation,” Mar 2012, doi:10.5772/36358.

[19] L. Cherroun, and B. Mohamed, “Fuzzy logic and reinforcement learning based approaches for mobile robot navigation in unknown environment,” *Mediterranean Journal of Measurement and Control*, vol. 9, pp. 109–117, 2013.

[20] F. Fathinezhad, V. Derhami, and M. Rezaeian, “Supervised fuzzy reinforcement learning for robot navigation,” *Applied Soft Computing*, vol. 40, pp. 33–41, 2016.

[21] M. Boujelben, D. Ayedi, C. Rekik, and N. Derbel, “Fuzzy logic controller for mobile robot navigation to avoid dynamic and static obstacles,” *2017 14th International Multi-Conference on Systems, Signals & Devices (SSD)*, Marrakech, Morocco, 2017, pp. 293–298, doi: 10.1109/SSD.2017.8166963.

[22] N. H. Singh, K. Thongam, “Mobile robot navigation using fuzzy logic in static environments,” *Procedia Computer Science*, vol. 125, pp. 11–17, 2018, https://doi.org/10.1016/j.procs.2017.12.004.

[23] M. Nadour, M. Boumehraz, L. Cherroun, *et al.*, “Mobile robot visual navigation based on fuzzy logic and optical flow approaches,” *Int J Syst Assur Eng Manag*, vol. 10, pp. 1654–1667, 2019. https://doi.org/10.1007/s13198-019-00918-2

[24] H. Batti, C. B. Jabeur, and H. Seddik, “Fuzzy logic controller for autonomous mobile robot navigation,” *2019 International Conference on Control, Automation and Diagnosis (ICCAD)*, Grenoble, France, 2019, pp. 1–6.

[25] D. Babunski, J. Berisha, E. Zaev, and X. Bajrami, “Application of fuzzy logic and pid controller for mobile robot navigation,” *2020 9th Mediterranean Conference on Embedded Computing (MECO)*, 2020, pp. 1-4, doi: 10.1109/MECO49872.2020.9134317.

[26] H. J.-T., C. C.-K, “Adaptive fuzzy sliding mode control of omnidirectional mobile robots with prescribed performance,” *Processes*, vol. 9, no. 12, Dec 2021.

[27] F. Abdessemed, K. Benmahammed, and E. Monacelli, “A fuzzy-based reactive controller for a non-holonomic mobile robot”, *Robotics and Autonomous Systems*, vol. 47, no. 1, pp. 31–46, 2004, DOI: 10.1016/j.robot.2004.02.006.

[28] L. A. Dias, R. W. de Oliveira Silva, P. C. da Silva Emanuel, A. F. Filho, and R. T. Bento, “Application of the Fuzzy logic for the development of autonomous robot with obstacles deviation,” *International Journal of Control, Automation and Systems*, vol. 16, no. 2, pp. 823–833, 2018, DOI: 10.1007/s12555-017-0055-9.

[29] A. Pandey, and D. R. Parhi, “Optimum path planning of mobile robot in unknown static and dynamic environments using fuzzy-wind driven optimization algorithm,” *Defence Technology*, vol. 13, no. 1, pp. 47–58, 2017, DOI: 10.1016/j.dt.2017.01.001.

[30] M. S. Masmoudi, N. Krichen, M. Masmoudi, and N. Derbel, “Fuzzy logic controllers design for omnidirectional mobile robot navigation,” *Applied Soft Computing*, vol. 49, pp. 901–919, 2016, DOI: 10.1016/j.asoc.2016.08.057.

[31] A. K. Rath, D. R. Parhi, H. C. Das, M. K. Muni, and P. B. Kumar, “Analysis and use of fuzzy intelligent technique for navigation of humanoid robot in obstacle prone zone”, *Defence Technology*, vol. 14, no. 6, pp. 677–682, 2018, DOI: 10.1016/j.dt.2018.03.008.

[32] P. Nattharith, and M. S. Güzel, “Machine vision and fuzzy logic-based navigation control of a goal-oriented mobile robot”, *Adaptive Behavior*, vol. 24, no. 3, pp. 168–180, 2016, DOI: 10.1177/1059712316645845.

[33] A. Karray, M. Njah, M. Feki, and M. Jallouli, “Intelligent mobile manipulator navigation using hybrid adaptive-fuzzy controller”, *Computers & Electrical Engineering*, vol. 56, pp. 773–783, 2016, DOI: 10.1016/j.compeleceng.2016.09.007.

[34] K. K. Pandey, C. Kumbhar, D. R. Parhi, S. K. Mathivanan, P. Jayagopal, and A. Haque, “Trajectory planning and collision control of a mobile robot: a penalty-based PSO approach,” *Mathematical Problems in Engineering*, vol. 2023, pp. 10, 2023, DOI: 10.1155/2023/1040461.

[35] S. Zhao, and S.-H Hwang, “ROS-based autonomous navigation robot platform with stepping motor,” *Sensors*, vol. 23, no. 7, pp. 3648, Mar 2023, DOI: 10.3390/s23073648.

[36] S. K. Tyagi, “Reliability analysis of a powerloom plant using interval valued intuitionistic fuzzy sets,” *Information Control*, vol. 8, pp. 338–353, 1965.

[37] F. Dernoncourt, *Introduction to Fuzzy Logic*, Massachusetts Institute of Technology: Cambridge, MA, USA, 2013.