Improving Deep Reinforcement Learning Training Convergence using Fuzzy Logic for Autonomous Mobile Robot Navigation

Abdurrahman bin Kamarulariffin, Azhar bin Mohd Ibrahim*, Alala Bahamid
Department of Mechatronics Engineering, International Islamic University Malaysia, Kuala Lumpur, Malaysia

***Abstract*—Autonomous robotic navigation has become hotspot research, particularly in complex environments, where inefficient exploration can lead to inefficient navigation. Previous approaches often had a wide range of assumptions and prior knowledge. Adaptations of machine learning (ML) approaches, especially deep learning, play a vital role in the applications of navigation, detection, and prediction about robotic analysis. Further development is needed due to the fast growth of urban megacities. The main problem of training convergence time in deep reinforcement learning (DRL) for mobile robot navigation refers to the amount of time it takes for the agent to learn an optimal policy through trial and error and is caused by the need to collect a large amount of data and computational demands of training deep neural networks. Meanwhile, the assumption of reward in DRL for navigation is problematic as it can be difficult or impossible to define a clear reward function in real-world scenarios, making it challenging to train the agent to navigate effectively. This paper proposes a neuro-symbolic approach that combine the strengths of deep reinforcement learning and fuzzy logic to address the challenges of deep reinforcement learning for mobile robot navigation in terms of training time and the assumption of reward by incorporating symbolic representations to guide the learning process, and inferring the underlying objectives of the task which is expected to reduce the training convergence time.**

Keywords*—Autonomous navigation; deep reinforcement learning; mobile robots; neuro-symbolic; Fuzzy Logic

I. INTRODUCTION

Advancements in robot navigation have spurred the development of algorithms that leverage basic rules and environmental mapping to optimize path planning. Rule-based methods, such as Fuzzy logic and Neuro-fuzzy techniques, have been extensively explored to enhance navigation decisions and tracking performance under uncertain conditions [1], [2]. While these methods offer valuable insights, they often require extensive justification and may not fully meet the demands for efficient and accurate path planning.

To address this challenge, researchers have turned to bio-inspired approaches, such as genetic algorithms and swarm optimization, which draw inspiration from biological behavior and incorporate prior knowledge to simulate human cognitive processes [3], [4]. One particularly promising area in navigation research is reinforcement learning (RL), which enables autonomous agents to learn and make sequential decisions in complex environments. Machine learning models, including supervised, unsupervised, and reinforcement learning, have played a pivotal role in robotics research, enabling learning, adaptation, and effective detection and classification. Deep reinforcement learning (DRL), a fusion of RL and deep neural networks, has emerged as a powerful approach for decision-making tasks involving high-dimensional inputs [5], [6].This article aims to delve into the application of RL techniques, specifically Q-learning and deep Q-networks, for mobile robot path planning. By seamlessly integrating these techniques with widely used frameworks such as ROS, Gazebo, and OpenAI, a robust and autonomous navigation system can be developed, leading to improved performance, optimized routes, and efficient obstacle avoidance in complex environments. The evaluation of this system will undoubtedly contribute to the advancement of autonomous robotics. The trial-and-error learning process inherent in RL offers immense potential for building human-level agents and has been extensively explored in various domains [7] [8]. Deep learning (DL), characterized by its ability to extract meaningful patterns and classifications from raw sensory data through deep neural networks, has revolutionized the field of machine learning. When combined with RL, in the form of DRL, this integration has shown remarkable success in tackling challenges associated with sequential decision-making [9], [10]. Notably, DRL excels in scenarios involving a vast number of states, making it an ideal candidate for addressing navigation complexities. Nevertheless, achieving optimal navigation remains an ongoing challenge, necessitating further optimization and effective handling of high-dimensional data. Reinforcement learning methods offer valuable approaches for learning and planning navigation, empowering agents to interact with their environment and make autonomous decisions. Various studies have proposed agent-based DRL approaches for navigation, successfully simulating diverse scenarios without the need for intricate rule-based systems or laborious parameter tuning. However, there is still room for improvement in terms of achieving the shortest and fastest routes. To enhance navigation performance and optimize evacuation paths, researchers have explored techniques such as look-ahead crowded estimation and Q-learning, which have demonstrated superior results compared to other RL algorithms [6]. Additionally, CNN-based robot-assisted evacuation systems have been developed to maximize pedestrian outflow by extracting specific features from high-dimensional images. Furthermore, iterative, and incremental learning strategies,

like vector quantization with Q learning (VQQL), have been proposed to expedite the learning process and optimize navigation by gradually improving interactions among agents [11], [12]. These advancements in DRL continue to show great promise in addressing the speed of agent learning and optimizing navigation processes. In the realm of task planning, the ability to find a series of steps that transform initial conditions into desired states is crucial. Task planning becomes especially important when atomic actions alone cannot accomplish a task. Neuro-symbolic task planning has emerged as an effective approach, allowing for the incorporation of restrictions, guidelines, and requirements in each activity. However, traditional task planners often rely on detailed hand-coded explanations, limiting their scalability. To overcome this limitation, a combination of deep learning and symbolic planning, known as a neuro-symbolic approach, has shown potential by leveraging visual information instead of hand-coded explanations [3], [13], [14]. However, collecting image data for neuro-symbolic models in robotic applications is a labor-intensive process that involves steps such as creating problem instances, defining initial and goal states, operating robots, and capturing scene images. The challenges associated with data collection have hindered the widespread adoption of neuro-symbolic models in robot task planning. Neuro-symbolic models excel in reasoning, providing explanations and manipulating complex data structures. Conversely, numerical models, such as neuronal models, are preferred for pattern recognition due to their generalization and learning abilities. A unified strategy proposes that the characteristic properties of symbolic artificial intelligence can emerge from distributed local computations performed by neuronal models, spanning cognitive functions from the neuron level to the structural level of the nervous system. By integrating neuro-symbolic and numerical models, a comprehensive framework can be established to leverage the strengths of both approaches in robotics. This integrated approach holds the potential to enable efficient task planning, grounding symbols in perceptual information, and enhancing pattern recognition capabilities. Ultimately, this integration could advance cognitive functions and pave the way for the creation of more sophisticated robotic systems.

This paper is organized as follows. Section II presents the proposed method which integrates the reinforcement learning (RL) and fuzzy logic for mobile robot path planning, aiming to create a robust autonomous navigation system that optimizes routes and efficiently avoids obstacles in complex environments. Section III illustrates the simulation set-up, while Section IV provides an evaluation of the training process of the policy optimization. Finally, Section V presents the evaluation and verification of the developed policy based on the proposed method, followed by the conclusion.

II. METHODS

The methodology for this project involves the utilization of simulation tools, namely Gazebo, ROS (Robot Operating System), and OpenAI Gym. Gazebo provides a realistic environment for simulating the mobile robot path planning system, while ROS serves as a comprehensive framework for controlling the robot and interfacing with its sensors and actuators. OpenAI Gym is used to train and evaluate the reinforcement learning algorithms. The main focus of this project is to apply reinforcement learning techniques to mobile robot path planning. Unlike traditional approaches that rely on SLAM or mapping techniques, the project aims to enable the robot to learn the optimal path through a reward and punishment system. By using reinforcement learning algorithms such as Q-learning, SARSA, and DQN, the robot can learn to navigate its environment efficiently and safely. To facilitate communication between the simulation and the robot, ROS integration is implemented. This integration allows the robot to receive sensor data, send control commands, and interact with the simulation environment seamlessly. By leveraging the capabilities of ROS, the reinforcement learning algorithms can effectively interface with the robot's actions and observations [15]–[17].The reinforcement learning algorithms receive feedback through a reward and punishment system based on the robot's performance in reaching the goal while avoiding collisions and obstacles. The training aims to optimize the robot's decision-making and path planning abilities. Performance analysis is conducted to assess the effectiveness of the trained reinforcement learning models. Metrics such as the time taken to reach the goal, collision occurrences with static and dynamic obstacles, and the number of pathing alterations are measured and analyzed. These metrics provide insights into the path planning efficiency, collision avoidance capabilities, and adaptability of the reinforcement learning approach. In conclusion, the methodology of this project involves using simulation tools (Gazebo, ROS, and OpenAI Gym) to evaluate the application of reinforcement learning algorithms (Q-learning, SARSA, and DQN) in mobile robot path planning. The integration of ROS ensures seamless communication between the simulation environment and the robot, while the OpenAI Gym environment provides a standardized framework for training and evaluating the algorithms. The methodology enables rigorous testing and analysis of the robot's performance in terms of path planning, collision avoidance, and adaptability to dynamic environments. This following subsection discusses the mathematical model of Q-learning with fuzzy logic approach theory towards navigation problems, and experimentation setup that is used in this work.

In the context of agents utilizing visual SLAM, traditional algorithms are still employed for final path planning on the map. However, RL offers numerous applications, and in mobile robot navigation, it can replace the path planning part. The RL model, after training, can effectively make decisions, enabling the agent to select its path from one location to another based on interactions with the environment [18], [19]. The environment is abstracted into a grid map representation, with each position on the map corresponding to an agent's state. Transitioning from one state to another reflects the actual movement of the entity, while the agent's behavioral decision-making is represented by its state choice at each step in the RL model. The reward value plays a pivotal role in guiding path selection. Early Q-learning recorded reward values between position states in a table, guiding the next state selection. As depth-enhanced learning emerges, the DL model is integrated, replacing the table with a neural network, which provides corresponding decision results by inputting the state [20], [21]. The weighting parameters in the neural network

influence the choice of the next state. On the other hand, when incorporating fuzzy logic into the RL model, the decision-making process becomes more nuanced and interpretable. Fuzzy logic allows for handling uncertainties and imprecise information, enabling the agent to reason with vague input and output values. By combining RL and fuzzy logic, the agent can make more human-like decisions, considering both the environment's precise measurements and the agent's subjective understanding of the surroundings. This fusion can enhance path planning in complex and dynamic environments by considering various factors and optimizing the decision-making process.

A. Q-Learning Algorithm

RL defines any decision maker as an agent and everything outside the agent as the environment. The agent aims to maximize the accumulated reward and obtains a reward value as a feedback signal for training through interaction with the environment. Beyond the agent (who perform actions) and the environment (which made of states), there are three major elements of a reinforcement learning system:

- Policy $\boldsymbol{\pi}$: It is to formalize an agent's decision and determine the agent's behaviour at a given time. A policy π is a function that maps between the perceived state and the action is taken from that state.
- Reward $\boldsymbol{r}$: The agent receives feedback known as rewards, r_{t+1} for each action at time step t, indicating the inherent desirability of that state. The main goal of the agent is to maximize the cumulative reward over time. The total sum of the rewards (return) is:

$$R_t = r_{t+1} + r_{t+2} + r_{t+3} + \ldots r_T,\ T\text{: final time step}$$

The agent-environment interaction breaks into episodes where each episode ends in a state called the terminal state, followed by a reset to a standard starting state. In some cases, the episodes continue where final time step would be $T = \infty$, and the return become infinite. So, a discount factor γ is introduced. The discounted return is defined as:

$$R_t = r_{t+1} + \gamma\, r_{t+2} + \gamma^2\, r_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

$$0 < \gamma < 1$$

Rewards can be sparse (after a long sequence of actions), every time step, or at the end of the episodes.

- Value function: Most of the RL algorithms are based on estimating value functions (states or state action). Value function is used to estimate how good a certain state is for the agent to be in (state value function), or how good a certain action is to perform in a specific state (state-action value function). The state value functions under the policy π, denoted $V\pi(s)$, is the expected return,

$$V^{\pi}(s) = E_{\pi}\{R_t | s_t = s\} = E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+1+k} | s_t = s\right\}$$

The state-action value function under policy π, denoted $Q^{\pi}(s, a)$, as the expected accumulated return from state s and action a. Q^{π} is also known as action value function or Q-Learning algorithm.

$$Q^{\pi}(s,a) = E_{\pi}\{ R_t | s_t = s, a_t = a\} = E_{\pi}\{\sum_{k=0}^{\infty} \gamma^k r_{t+1+k} | s_t = s, a_t = a\}$$

$$Q^{\pi}(s,a) = E\,[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots | s, a\,]$$

Reinforcement learning is about finding an optimal policy that achieves a lot of reward over the long-term. A policy π is defined to be better than or equal to a policy π' if its expected return is greater than or equal to that of π' for all states.

$$\pi \geq \pi' \text{ if and only if } v^{\pi}(s) \geq V^{\pi'}(s), \text{for all states}$$

Optimal Value Functions must satisfy the below conditions:

$$V^*(s) = max\ V.(s), \text{ for all states}$$

$$Q^*(s, a) = maxQ.(s, a), \text{ for all states and actions}$$

We get the optimal policy by solving $Q*(s, a)$ to find the action that gives the most optimal state-action value function,

$$\pi^*(s) = arg \max_a Q^*(s,a)$$

Q-Learning algorithm is an off-policy value-based RL algorithm and very effective under unknown environment [6], [21], [22]. The value of a state-action can be decomposed into immediate reward plus the value of successor state-action $Q^{\pi}(s', a')$ with a discount factor (γ).

$$Q^{\pi}(s,a) = E_{s'a'}[r + \gamma Q^{\pi}(s', a') | s, a]$$

And according to the Bellman optimality, the optimal value function can be expressed as:

$$Q^*(s,a) = E_{s'a'}[r + \gamma \max_a Q^*(s', a') | s, a]$$

Update the value function iteratively to obtain optimal value function,

$$Q(s,a) \leftarrow Q(s,a) + \alpha.\,[r + \gamma \max a'\ Q(s', a') - Q(s,a)],$$

α: learning rate

$Q(s, a)$ converges to $Q^*(s, a)$ as $t \rightarrow \infty$.

Algorithm 1 illustrates the overall framework of the proposed Q-learning to generate the shortest route for navigation mapping.

Algorithm 1. Overall framework of the Q-Learning

```
Initialize Q (s, a) arbitrarily
repeat for each episode.
Initialize s.
  for each step of the episode do
    Choose a from s using € greedy policy.
    Do action a and observer r and s'
    Q(s, a) ← Q(s, a) + α. [r + γ max a'  Q (s', a') - Q (s, a)]
    s ← s'
  until s is terminal
```

B. Fuzzified Reward Function

The fuzzy logic-based approach has been adopted to enhance the decision-making [23], [24] process of the autonomous agent navigating through a maze. This work incorporated symbolic representation by adopting fuzzy logic into the reward function to guide the learning process and address the challenge of the computational demands of training. The proposed fuzzy reward function has three input variables and one output. The input variables are distance to obstacle (near, medium, far), distance to target (near, medium, far), and visual range (off target, medium, on target). The output variable is the reward points (see Fig. 4).

By employing three membership levels for each input variable (See Fig. 1, 2 and 3), a comprehensive set of 27 fuzzy rules has been devised (see Table I), effectively covering all possible combinations of the environment states. These rules dictate the agent's rewards, which are categorized as punishment (least), medium, and reward (most). By leveraging the flexibility and adaptability of fuzzy logic, the agent is guided through its learning process with a more nuanced and context-aware reward system, allowing it to make more informed decisions in a variety of maze scenarios and significantly improving its learning efficiency.

Table I presents a comprehensive and systematic overview of the fuzzy logic rules governing the agent's decision-making process in the maze navigation task. The table showcases the various combinations of input possibilities, encompassing distance with obstacles, distance with target location, and visual range, each categorized into appropriate linguistic variables (e.g., near, medium, far; off target, medium, on target). For every unique combination, the corresponding fuzzy logic "If/Then" rules are defined, determining the agent's rewards as punish, medium, or reward. Table I highlights the agent's adaptability and versatility through the vast array of rules, capturing the intricacies of different maze scenarios. With 27 distinct rules, the fuzzy logic system can precisely respond to the agent's real-time observations, guiding it towards optimal actions that lead to successful navigation. This rich and nuanced reward system empowers the agent to effectively learn from its experiences, enabling it to avoid obstacles, approach the target, and dynamically adjust its behavior based on varying visual cues. Consequently, the "If/Then Analysis Fuzzy Logic Rules Possibilities" table serves as a powerful tool in understanding and implementing the complex decision-making process of the agent, fostering efficient learning and successful maze navigation.

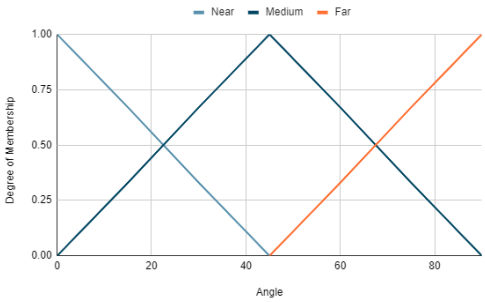


Fig. 1. Visual range.

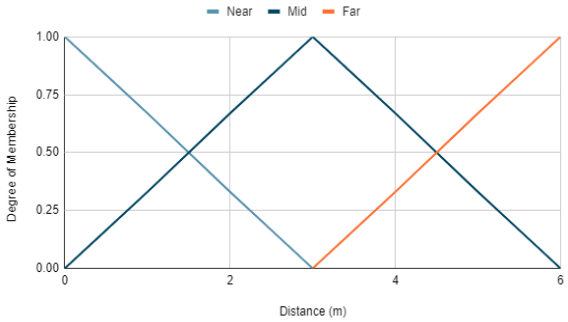


Fig. 2. Distance to target.

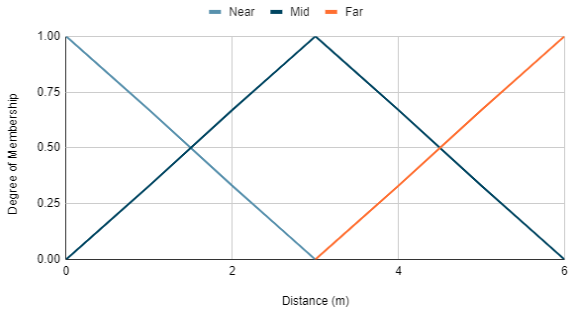


Fig. 3. Distance to obstacle.

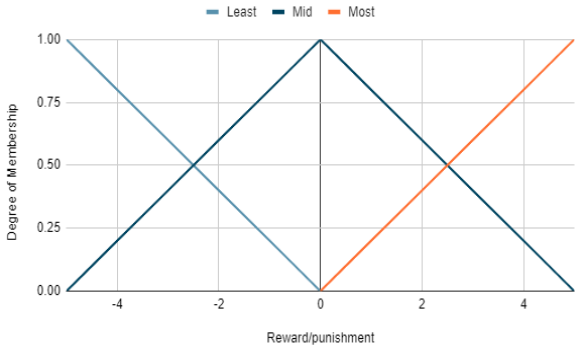


Fig. 4. Output fuzzy: reward points.

TABLE I. IF/THEN FUZZY LOGIC RULES FOR REWARDS

Distance	Visual Range	Obstacle (Near)	Obstacle (Medium)	Obstacle (Far)
Target (Near)	Near	High Positive	High Positive	High Positive
	Medium	Mid Positive	Mid Positive	Mid Positive
	Far	Low Positive	Low Positive	Low Positive
Target (Medium)	Near	High Middle	High Middle	High Middle
	Medium	Low Middle	Middle	Low Middle
	Far	Low Middle	Low Middle	Low Middle
Target (Far)	Near	Low Negative	Low Negative	Low Negative
	Medium	Mid Negative	Mid Negative	Mid Negative
	Far	High Negative	High Negative	High Negative

III. SIMULATION SETUP

To thoroughly evaluate the enhanced performance of the proposed method, this study embarked on constructing a detailed 3-D raster map model and crafting two distinct simulation maps, meticulously illustrated in Fig. 5(a) and Fig. 5(b). In this simulated environment, dynamic obstacles, symbolized by white cylinders, were strategically placed, posing challenges to a TurtleBot simulation machine car, visually presented in black. The machine car's laser range, portrayed in blue, scanned the surroundings as it navigated through the intricate maze. The red square pinpointed a target training point, emphasizing the complexity of the assigned tasks.

Fig. 5(a) specifically delves into a scenario where the TurtleBot is tasked with locating a singular target location represented by the red square amid a set of four cylindrical obstacles. This intricate setting simulates real-world challenges where the robot must efficiently identify and navigate towards a specific point among various hindrances.

Fig. 5(b) presents a more intricate scenario where the TurtleBot is assigned the mission of identifying two specific cylinders as target locations within a maze of block obstacles. This heightened complexity mirrors scenarios where the robot must discern and navigate through a maze-like environment to pinpoint multiple objectives. This detailed simulation environment allows for a comprehensive assessment of the proposed method's effectiveness in handling diverse and intricate navigation tasks.

The computer used for the simulations was equipped with a 4-core Intel i5 7400 CPU running at 3.00 GHz, 8 GB of RAM, running on the Ubuntu 16.04 operating system, and utilizing the ROS kinetic system. The article leveraged certain parameters for the 3-D environment model, which were sourced from the ROS open-source community. The corresponding parameter settings are as follows:

$$r_{goal} = 100,\ r_{obstacle} = -100,\ \varepsilon = -100,\ \sigma_x = \sigma_y = 1$$

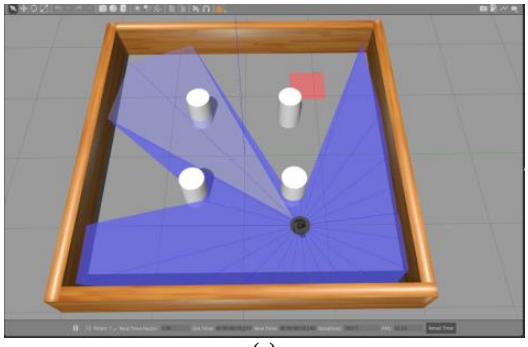
$$\gamma_{goal} = 0.9,\ r_{obstacle} = 0.9,\ r_{critical} = 0.8,\ r_{otherwise} = 0.75$$

In the context of this work:

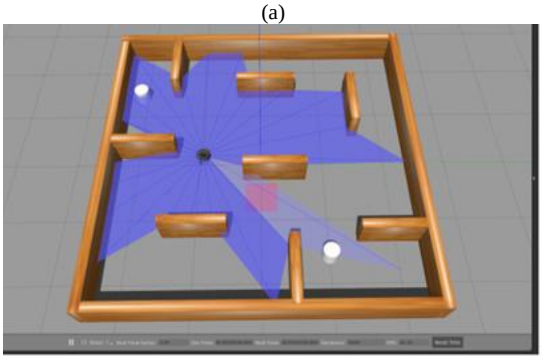
- " *r* " signifies a single reward.
- σ_x and σ_y represent the obstacle center coordinates.
- γ (gamma) serves as the discount factor, influencing the importance of future rewards.

In this context, *r* represents a reward, with r_{goal} and $r_{obstacle}$ being specific awards assigned to reaching the goal and encountering obstacles, respectively. The term γ serves as the discount factor, influencing the importance of future rewards in the context of reinforcement learning. The parameters ε, σ_x, and *and* σ_y represent the grid center coordinates, contributing to the spatial representation of the environment and the localization of obstacles.

Fig. 6 depicts The Turtlebot2 which is a popular mobile robot platform widely used in robotics research and applications.



(a)



(b)

Fig. 5. (a) and (b) Showcase maze circuits in the Gazebo simulation environment to test and evaluate the robot's path planning capabilities.

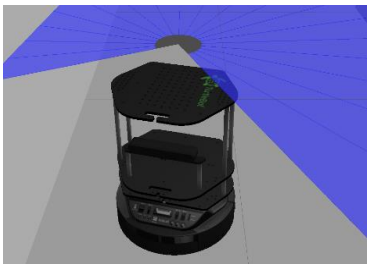


Fig. 6. The Turtlebot2 robot equipped with a lidar sensor.

IV. TRAINING PERFORMANCE OF THE PROPOSED METHOD

The visual representations in Fig. 5(a) and Fig. 5(b) aimed to illustrate the improved method's performance across different simulation maps. While these figures may not directly demonstrate capabilities, they serve as visual aids to showcase the distinct scenarios and complexities encountered by the agent in each environment. It's essential to acknowledge that the term "validation" in the context of comparing results from Reinforcement Learning (RL) and Fuzzy Logic (FL) approaches refers to a qualitative assessment rather than a formal validation process.

In Fig. 5(a), the experiment showcased the performance of the improved method on the first simulation map, providing insights into how the agent navigates a specific environment. On the other hand, Fig. 5(b) illustrated the capabilities of the improved method on the second simulation map, highlighting its adaptability to different scenarios. By comparing the results from RL and FL approaches, the article qualitatively validated the effectiveness of the enhanced technique in the complex 3-D environment. These visual representations offered a valuable qualitative assessment, helping to understand the

nuanced behaviors of the agent, its path planning strategies, and obstacle avoidance mechanisms in diverse settings. The improved method consistently demonstrated superior performance, efficiently finding optimal routes to reach the target point while navigating around obstacles effectively. The simulations offered valuable insights into the agent's behavior, path planning, and obstacle avoidance, elucidating fundamental aspects of autonomous robot navigation. The superior performance of the enhanced method was evident in its ability to navigate efficiently, choosing optimal routes while circumventing obstacles effectively.

Furthermore, the deliberate choice of Fig. 5(b) as a test run was made to rigorously assess the proposed method's robustness in scenarios with increased complexity and multiple target points. This strategic selection adds an additional layer of validation, demonstrating the algorithm's efficacy in handling intricate navigation tasks.

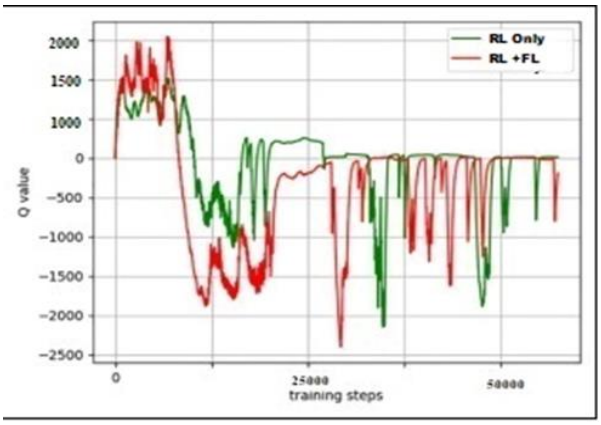


Fig. 7. Q-value comparison.

As shown in Fig. 7, in the Q-value map, the Fuzzy Logic (FL) example has a faster convergence speed, especially in 50 K training sessions, and after approximating 25 K trainings, the Q value of the FL algorithm is still richly transformed, showing the FL is less likely to fall into local optimum. This segment of our analysis offers a glimpse into the noteworthy performance attributes of the FL local search approach. As depicted in Fig. 8, which illustrates the bonus map, the FL example stands out due to its utilization of a multiple reward mechanism and a loop memory network. This distinction is most evident in the greater reward values attributed to the FL path, which correspondingly signify a reduced occurrence of repeated errors. In essence, a higher reward value in this context indicates a superior capacity to identify and follow an optimal path with fewer deviations. Turning our attention to Fig. 9, we delve into the loss diagram. Here, we observe a compelling trend: the loss associated with the FL example is consistently lower compared to that of the RL example. This finding is particularly significant, as it underscores the model's proficiency in minimizing error during the learning process. A lower loss value reflects a more accurate prediction and action selection by the model, emphasizing the effectiveness of the FL approach in optimizing path planning. To provide a closer examination of this phenomenon, Fig. 10 offers a magnified view of the loss diagram from Fig. 9. This detailed perspective reaffirms the rationality and effectiveness of the loss function employed in our model. The stability in parameter learning, particularly evident in the FL example, facilitates faster convergence to the optimal values. This not only enhances the efficiency of path planning but also showcases the model's robustness in navigating complex environments.

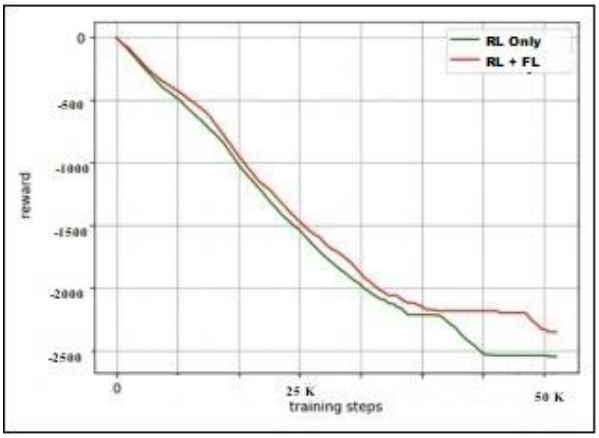


Fig. 8. Cumulative Rewards based on the proposed method vs. pure RL algorithm.

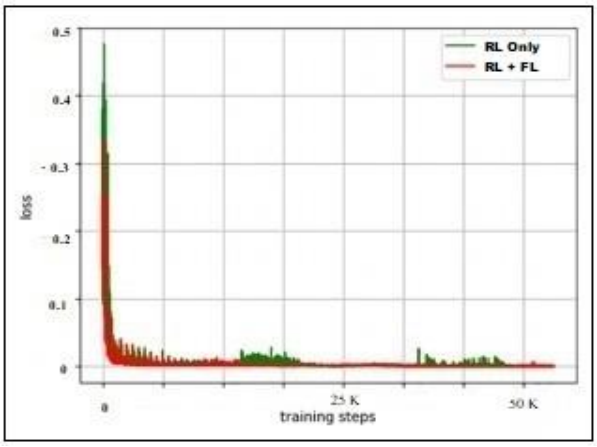


Fig. 9. Loss comparison.

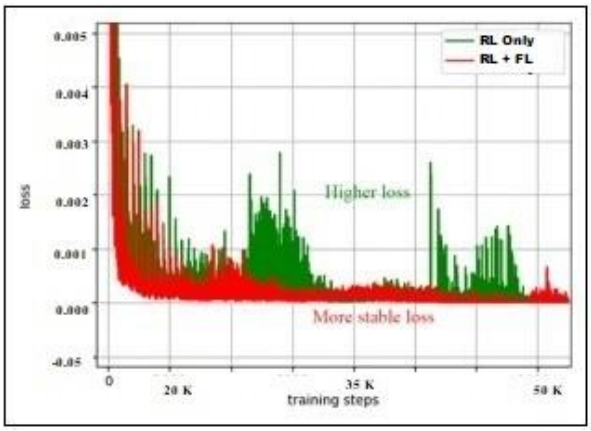


Fig. 10. Loss comparative enlarged view.

V. Evaluation and Verification of the Developed Policy

This work conducted three comprehensive tests to rigorously evaluate the performance of the proposed method. Each simulation aimed to assess the effectiveness of the respective algorithm in enabling the mobile robot to learn and navigate its environment autonomously.

To verify the practical performance of the model, physical tests were conducted on the robotic machine based on the robot operating system (ROS). The TurtleBot machine car was employed for these experiments to ensure consistency and reliability. The test environment comprised an obstacle zone constructed in the laboratory terrain, with the ideal distance from the starting point to the target point set at 8.3 meters. Fig. 5(a) and Fig. 5(b) depict the laser environment after its construction. It's important to note that the use of TurtleBot in these experiments is not meant to directly reduce errors in the algorithm. Instead, TurtleBot provides a standardized platform for testing, ensuring consistency and reliability across multiple trials. The choice of TurtleBot contributes to the creation of a controlled and reproducible testing environment, minimizing potential errors arising from variations in hardware and environmental conditions. This emphasis on error reduction pertains to the establishment of a robust and reliable basis for evaluating the proposed method's performance in real-world scenarios rather than directly mitigating errors in the algorithm or system. Following the integration of the trained model into the navigation function package, a meticulous series of verification tests was carried out to assess its performance. Each testing round consisted of five restarts, with three experiments conducted within each round to ensure the robustness of the evaluation process. For instance, in the first round of experiments, the robot's performance was tested through three individual trials: the first trial covered a distance of 8.8 meters in 77 seconds; the second trial covered 9.0 meters in 78 seconds, and the third trial spanned 8.6 meters in 73 seconds. By calculating the mean of these results, we obtained an average performance of 8.8 meters covered in 76 seconds.

Table III presents the detailed results of the first round, where the robot covered distances of 8.8 meters in 65 seconds, 8.6 meters in 53 seconds, and 8.7 meters in 56 seconds during the three tests. The calculated mean for Table II was 8.7 meters covered in 58 seconds. Notably, Table II exhibited a higher learning rate compared to Table II, indicating improved efficiency in path planning and execution.

TABLE II. The Example of RL Algorithm

Examples	**length/time**			
	Test 1	Test 2	Test 3	Mean
First Round	8.8 m/77 s	9.0 m/78 s	8.6 m/73 s	8.8 m/76 s
Second Round	9.3 m/86 s	9.1 m/83 s	8.9 m/74 s	9.1 m/81 s
Third Round	8.9 m/68 s	8.6 m/63 s	9.2 m/70 s	8.9 m/67 s
Fourth Round	9.1 m/78 s	8.9 m/73 s	8.7 m/71 s	8.9 m/74 s
Fifth Round	9.2 m/80 s	9.2 m/77 s	8.6 m/77 s	9.0 m/78 s

TABLE III. The Example of Reinforcement Learning with Fuzzy Logic Algorithm

Examples	**Length/Time**			
	Test 1	**Test 2**	**Test 3**	**Mean**
First Round	8.8 m/65 s	8.6 m/53 s	8.7 m/56 s	8.7 m/58 s
Second Round	8.8 m/63 s	8.9 m/69 s	8.7 m/60 s	8.8 m/64 s
Third Round	8.7 m/66 s	8.7 m/70 s	8.5 m/68 s	8.6 m/68 s
Fourth Round	8.7 m/73 s	8.8 m/65 s	8.6 m/66 s	8.7 m/68 s
Fifth Round	8.4 m/71 s	8.7 m/73 s	8.4 m/69 s	8.5 m/69 s

The overarching analysis of these comprehensive tests reveals that the Fuzzy Logic approach consistently outperforms other methods in terms of both time consumption and path length, particularly in the scenario represented in Fig. 5(b). It consistently finds shorter paths in less time, highlighting its superior efficiency. Additionally, the Fuzzy Logic method demonstrates remarkable stability in locating multiple paths, underscoring its prowess in complex environment path-finding.

However, it's important to acknowledge certain limitations associated with the Fuzzy Logic-based approach. While it excels in various aspects of path planning, it may face challenges when confronted with highly dynamic and rapidly changing environments. Fuzzy Logic, being rule-based and reliant on predetermined membership functions, might struggle to adapt swiftly to unpredictable obstacles or situations. Additionally, its performance could be impacted by the complexity and size of the environment, as processing a vast amount of data can introduce computational overhead.

Therefore, while the Fuzzy Logic approach proves highly effective in many scenarios, it may not be the optimal choice for applications demanding real-time adaptability in extremely dynamic settings. Exploring its boundaries and considering alternative approaches for such specific scenarios remains a valuable avenue for future research and development.

VI. Conclusion

This research introduced a novel navigation method based on Q- learning and fuzzy logic for efficient path planning of agents in diverse environments. The proposed approach combines the strengths of deep learning with symbolic reasoning, specifically Fuzzy Logic, to overcome the challenges faced by traditional DRL methods in mobile robot navigation, reducing the global path search time by 6-9% and shortening the average path search length by 4-10% compared to pure Q-learning. The incorporation of symbolic representations in the learning process leads to reduced training convergence time and more practical path planning results. The experimental results demonstrate its efficiency and effectiveness in complex environments, making it a promising solution for autonomous robotic navigation in urban megacities. As future work, the effectiveness of new RL algorithms will be explored in even more challenging environments, further advancing the field of autonomous robotic navigation.

REFERENCES

[1] U. Rakhman, J. Ahn, and C. Nam, "Fully automatic data collection for neuro-symbolic task planning for mobile robot navigation," in Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics, Institute of Electrical and Electronics Engineers Inc., 2021, pp. 450–455. doi: 10.1109/SMC52423.2021.9658822.

[2] A. Zhu and S. X. Yang, "Neurofuzzy-based approach to mobile robot navigation in unknown environments," IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews, vol. 37, no. 4, pp. 610–621, Jul. 2007, doi: 10.1109/TSMCC.2007.897499.

[3] P. Coraggio and M. De Gregorio, "A Neurosymbolic Hybrid Approach for Landmark Recognition and Robot Localization."

[4] O. Castillo, R. Martínez-Marroquín, P. Melin, F. Valdez, and J. Soria, "Comparative study of bio-inspired algorithms applied to the optimization of type-1 and type-2 fuzzy controllers for an autonomous mobile robot," Inf Sci (N Y), vol. 192, pp. 19–38, Jun. 2012, doi: 10.1016/j.ins.2010.02.022.

[5] Y. Li, "Deep Reinforcement Learning: An Overview," Jan. 2017, [Online]. Available: http://arxiv.org/abs/1701.07274.

[6] H. Van Hasselt, A. Guez, and D. Silver, "Deep Reinforcement Learning with Double Q-Learning." [Online]. Available: www.aaai.org.

[7] K. Zhang, F. Niroui, M. Ficocelli, and G. Nejat, "Robot Navigation of Environments with Unknown Rough Terrain Using deep Reinforcement Learning," in 2018 IEEE International Symposium on Safety, Security, and Rescue Robotics, SSRR 2018, Institute of Electrical and Electronics Engineers Inc., Sep. 2018. doi: 10.1109/SSRR.2018.8468643.

[8] N. Altuntas, E. Imal, N. Emanet, and C. N. Öztürk, "Reinforcement learning-based mobile robot navigation," Turkish Journal of Electrical Engineering and Computer Sciences, vol. 24, no. 3, pp. 1747–1767, 2016, doi: 10.3906/elk-1311-129.

[9] C. Pérez-D'Arpino, C. Liu, P. Goebel, R. Martín-Martín, and S. Savarese, "Robot Navigation in Constrained Pedestrian Environments using Reinforcement Learning," in Proceedings - IEEE International Conference on Robotics and Automation, Institute of Electrical and Electronics Engineers Inc., 2021, pp. 1140–1146. doi: 10.1109/ICRA48506.2021.9560893.

[10] V. Zambaldi et al., "Relational Deep Reinforcement Learning," Jun. 2018, [Online]. Available: http://arxiv.org/abs/1806.01830.

[11] D. Dong, C. Chen, J. Chu, and T. J. Tarn, "Robust quantum-inspired reinforcement learning for robot navigation," IEEE/ASME Transactions on Mechatronics, vol. 17, no. 1, pp. 86–97, Feb. 2012, doi: 10.1109/TMECH.2010.2090896.

[12] Y. Zhu, Z. Wang, C. Chen, and D. Dong, "Rule-Based Reinforcement Learning for Efficient Robot Navigation With Space Reduction," IEEE/ASME Transactions on Mechatronics, vol. 27, no. 2, pp. 846–857, Apr. 2022, doi: 10.1109/TMECH.2021.3072675.

[13] J. Priya Inala et al., "Neurosymbolic Transformers for Multi-Agent Communication." [Online]. Available: https://github.com/jinala/.

[14] P. Coraggio, M. De Gregorio, and M. Forastiere, "ROBOT NAVIGATION BASED ON NEUROSYMBOLIC REASONING OVER LANDMARKS," 2008. [Online]. Available: www.worldscientific.com.

[15] M. Sokolov, R. Lavrenov, A. Gabdullin, I. Afanasyev, and E. Magid, "3D modelling and simulation of a crawler robot in ROS/Gazebo," in ACM International Conference Proceeding Series, Association for Computing Machinery, Dec. 2016, pp. 61–65. doi: 10.1145/3029610.3029641.

[16] K. Takaya, T. Asai, V. Kroumov, and F. Smarandache, Simulation Environment for Mobile Robots Testing Using ROS and Gazebo. 2016. doi: 10.0/Linux-x86_64.

[17] K. Sukvichai, K. Wongsuwan, N. Kaewnark, and P. Wisanuvej, "Implementation of Visual Odometry Estimation for Underwater Robot on ROS by using RaspberryPi 2."

[18] N. Botteghi, B. Sirmacek, K. A. A. Mustafa, M. Poel, and S. Stramigioli, "On Reward Shaping for Mobile Robot Navigation: A Reinforcement Learning and SLAM Based Approach," Feb. 2020, [Online]. Available: http://arxiv.org/abs/2002.04109.

[19] A. V. Bernstein, E. V. Burnaev, and O. N. Kachan, "Reinforcement learning for computer vision and robot navigation," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Springer Verlag, 2018, pp. 258–272. doi: 10.1007/978-3-319-96133-0_20.

[20] A. Newman, G. Yang, B. Wang, D. Arnold, and J. Saniie, "Embedded Mobile ROS Platform for SLAM Application with RGB-D Cameras," in IEEE International Conference on Electro Information Technology, IEEE Computer Society, Jul. 2020, pp. 449–453. doi: 10.1109/EIT48999.2020.9208310.

[21] Q. Jiang, "Path Planning Method of Mobile Robot Based on Q-learning," in Journal of Physics: Conference Series, IOP Publishing Ltd, Feb. 2022. doi: 10.1088/1742-6596/2181/1/012030.

[22] K.-H. Park, Y.-J. Kim, and J.-H. Kim, "Modular Q-learning based multi-agent cooperation for robot soccer," 2001. [Online]. Available: www.fira.net.

[23] G. Antonelli, S. Chiaverini, and G. Fusco, "A fuzzy-logic-based approach for mobile robot path tracking," IEEE Transactions on Fuzzy Systems, vol. 15, no. 2, pp. 211–221, Apr. 2007, doi: 10.1109/TFUZZ.2006.879998.

[24] E. Ayari, S. Hadouaj, and K. Ghedira, "A fuzzy logic method for autonomous robot navigation in dynamic and uncertain environment composed with complex traps," in Proceedings - 5th International Multi-Conference on Computing in the Global Information Technology, ICCGI 2010, 2010, pp. 18–23. doi: 10.1109/ICCGI.2010.47.