COMPUTATIONAL INTELLIGENCE(MCTA 3371)

SEMESTER I, 2025/2026

SECTION 1

Intelligent Mobile Robot Navigation Using Computational Intelligence Techniques

NO.	NAME	MATRIC NO.
1	AHMAD SYAHMI WAJDI BIN AHMAD SHUKRI	2310663
2	MUHAMMAD IRFAN BIN MOHD NAZIRUDIN	2414855
3	NOR EFFENDI BIN ANUAR	2310519

1.0 Introduction

Mobile robot navigation is the ability of an autonomous robot to move safely and efficiently from a starting position to the goal that been set while avoiding obstacles in the map. This task is fundamentally challenging due to uncertainties in sensor data, environmental complexity, and the presence of narrow passage obstacles.

One major challenge in robot navigation is the local minima problem, where a robot becomes trapped in a region that appears safe locally but does not lead toward the goal. Traditional reactive controllers, such as pure Fuzzy Logic systems, rely only on local sensor information and therefore are highly susceptible to this issue.

Soft computing techniques provide an effective solution to this problem. Fuzzy Logic enables human-like decision making under uncertainty, while Genetic Algorithms (GA) provide global optimization capabilities through evolutionary search. By combining both methods, a Hybrid Fuzzy–GA system can exploit the strengths of each: reactive safety from fuzzy reasoning and global path optimization from GA.

2.0 Objectives

The primary goal of this mini-project is to develop an intelligent autonomous navigation system for a mobile robot using a hybrid computational intelligence approach. The specific objectives are:

- To design a Fuzzy Logic system for evaluating navigation risk based on goal distance and obstacle proximity.
- To implement a Genetic Algorithm for global path optimization.
- To integrate Fuzzy Logic into the GA fitness function to form a Hybrid Fuzzy–GA system.
- To compare the performance of Fuzzy Logic only and Hybrid Fuzzy–GA navigation in both simple and complex environments.
- To evaluate system performance based on success rate, path length, collisions, and smoothness.

3.0 Environment Setup and Robot Model

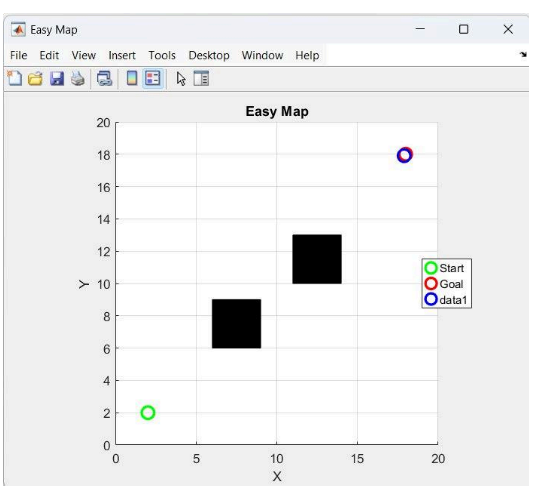
The simulation environment was developed using MATLAB. The workspace is modeled as a discrete 20 x 20 grid represented by a binary matrix, where 0 denotes free navigable space, and 1 denotes obstacles (walls or blocks).

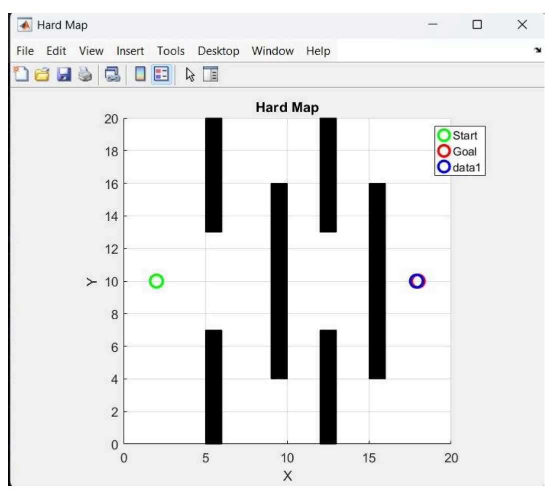
3.1 Map Generation

Two distinct map configurations were designed to test the robot's adaptability:

- **Map 1 (Easy Map):** A sparse environment containing two isolated rectangular obstacles positioned diagonally across the center. This map provides ample open space and is designed to test the robot's basic pathfinding and straight-line efficiency.
    - *Start Point:* Bottom-Left (2, 2)
    - *Goal Point:* Top-Right (18, 18)
- **Map 2 (Hard Map):** A complex "Slalom-like" environment featuring multiple vertical walls extending from the top and bottom borders. This layout divides the map into narrow vertical corridors, forcing the robot to maneuver carefully around barriers rather than moving in a straight line. It tests the controller's ability to handle narrow passages and significant deviations from the direct path.
    - *Start Point:* Middle-Left (2, 10)
    - *Goal Point:* Middle-Right (18, 10)

The maps were visualized using MATLAB's figure window, with the Start position marked in Green and the Goal position marked in Red.





3.2 Robot Model

The robot is modeled as a point robot operating in a discrete grid space. At each step, the robot can move in one of eight possible directions (up, down, left, right, and diagonals).

The robot is equipped with:

- A goal distance estimator (Euclidean distance to the target).
- An obstacle proximity sensor, which computes the minimum distance to surrounding obstacles within a local window.

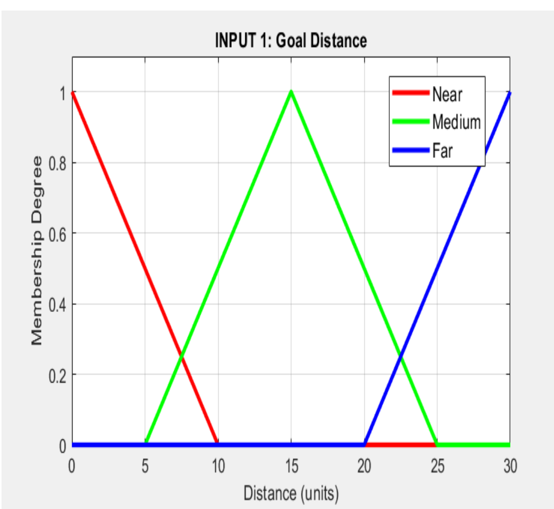
4.0 Design and Development Process

4.1 Fuzzy Logic Design

Fuzzy Input:

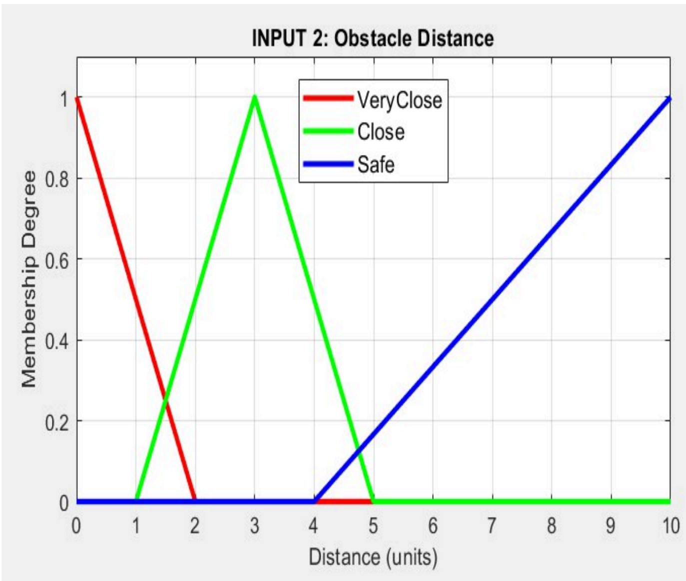
1. Goal Distance (Range: 0-30)

- Near: [0 ,10]
- Medium: [5,15,20]
- Far: [20,25,30]



2. Obstacle Distance (Range: 0-10)

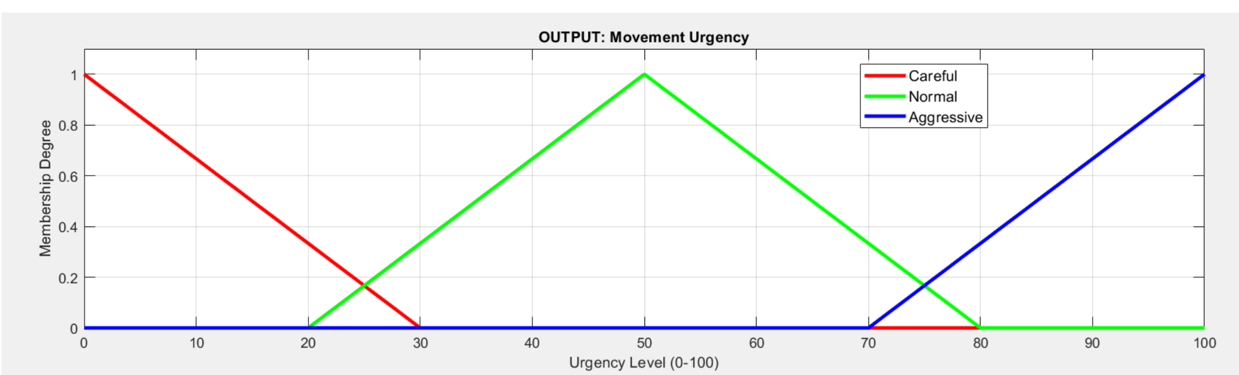
- Very Close: [0,1,2]
- Close: [1,3,5]
- Safe: [4,10]



Fuzzy Output:

Risk Level (Range: 0–100)

- Careful: [0,30]
- Normal:[20,80]
- Aggressive: [70,100]



Fuzzy Rules:

The fuzzy rule consists of 9 rules:

1. IF Goal is Near AND Obstacle is Safe then Risk is Low
2. IF Goal is Near AND Obstacle is Close then Risk is Moderate
3. IF Goal is Near AND Obstacle is Very Close, then Risk is High
4. IF Goal is Medium AND Obstacle is Safe then Risk is Low
5. IF Goal is Medium AND Obstacle is Close then Risk is Moderate
6. IF Goal is Medium AND Obstacle is Very Close then Risk is High
7. IF Goal is Far AND Obstacle is Safe then Risk is Moderate
8. IF Goal is Far AND Obstacle is Close then Risk is Moderate
9. IF Goal is Far AND Obstacle is Very Close then Risk is High

4.2 Genetic Algorithm Design

The Genetic Algorithm parameters are:

- Population size: **150**
- Chromosome length: **100 steps**
- Selection method: **Tournament Selection**
- Crossover rate: **0.85**
- Mutation rate: **0.15**
- Elite count: **10**
- Maximum generations: **300**

Each chromosome encodes a sequence of robot movements.

4.3 Hybrid Fuzzy-GA Mechanism

The Hybrid system integrates both approaches by embedding **Fuzzy Urgency** into the GA fitness function.

Key Concept: Fuzzy Urgency

- Fuzzy Logic computes an **urgency score** based on proximity to obstacles
- Paths that pass near obstacles receive a **higher penalty**
- Safe, smooth paths are rewarded

Hybrid Fitness Function Behavior

This ensures:

- GA avoids risky regions identified by fuzzy logic
- The robot escapes local minima such as U-shaped traps
- Navigation is both **globally optimal** and **locally safe**

Obstacle Distance Evaluation

In the **Hard Map**, the Fuzzy-Only controller fails due to the **Local Minima problem**. The robot repeatedly reacts to nearby obstacles and oscillates inside the U-shaped trap without global awareness of an exit path.

The Hybrid system successfully avoids this issue because:

- The GA provides a **global escape strategy**
- Fuzzy Urgency discourages unsafe paths near obstacles
- The robot can temporarily move away from the goal to reach a better global solution

This combination allows the robot to exit the U-trap and reach the goal reliably.

5.0 Simulation and Results

The simulation interface was developed entirely within **MATLAB** to provide real-time visual feedback of the robot's decision-making process.

Simple Map:

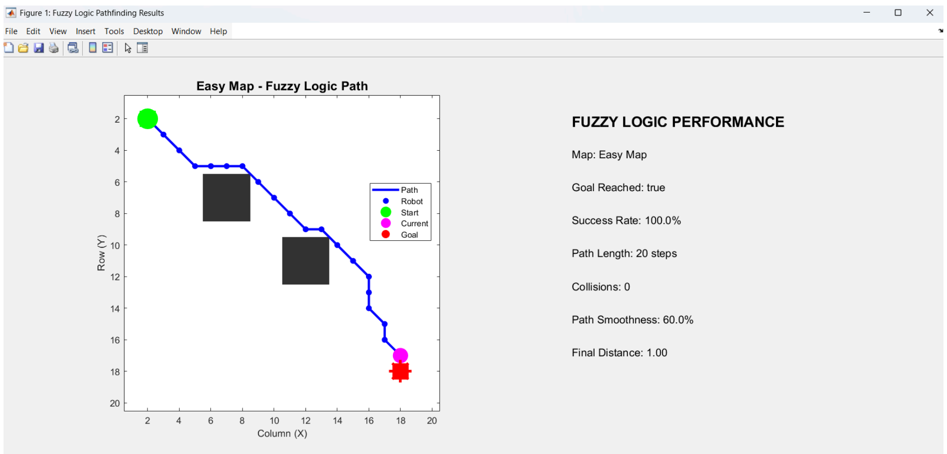


Figure 1: Fuzzy logic on simple map

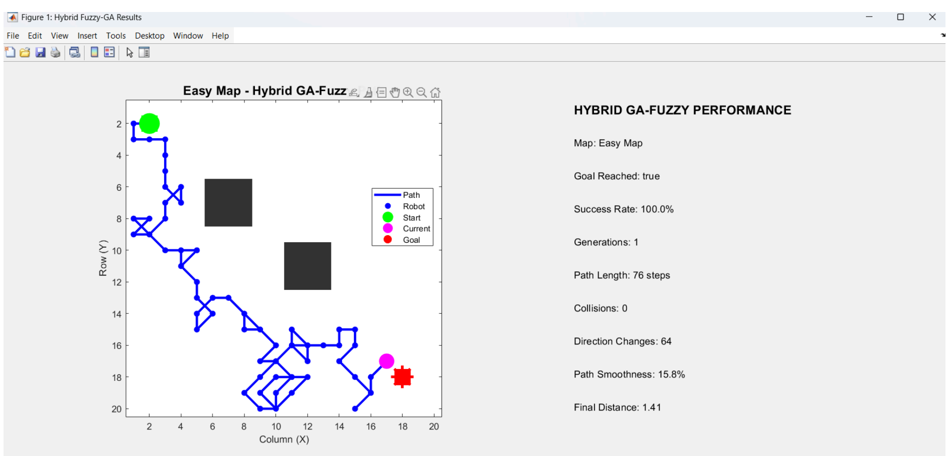


Figure 2: Hybrid GA-Fuzzy on simple map

Complex Map:

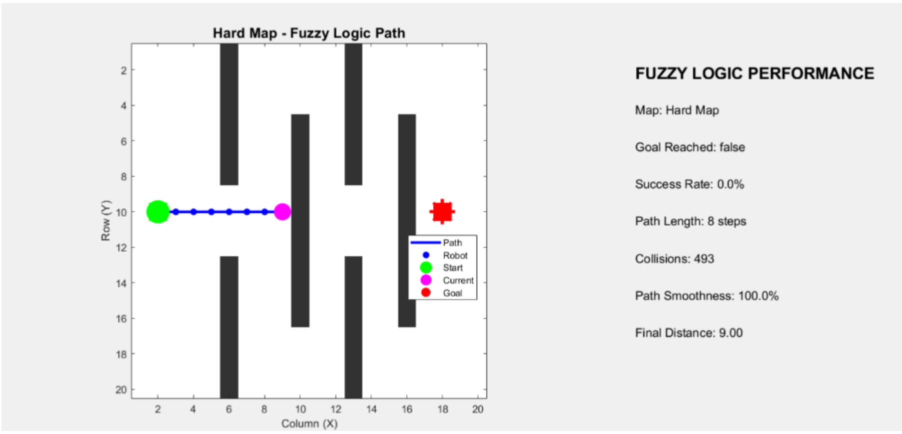


Figure 3: Fuzzy logic on a complex map

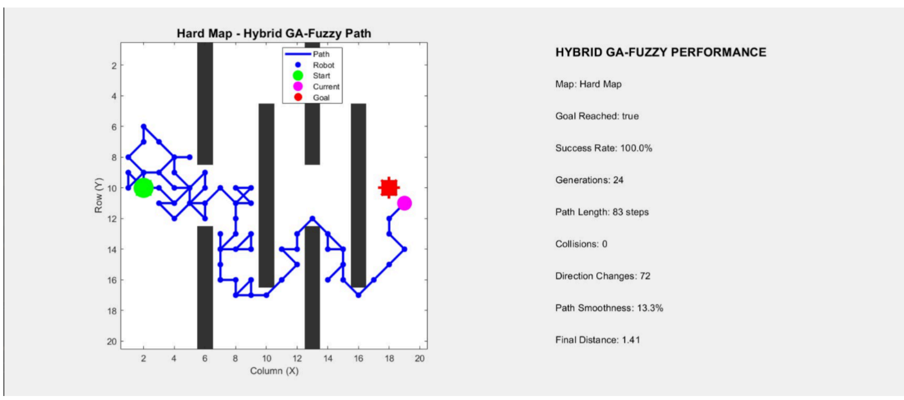


Figure 4: Hybrid GA-Fuzzy on a complex map

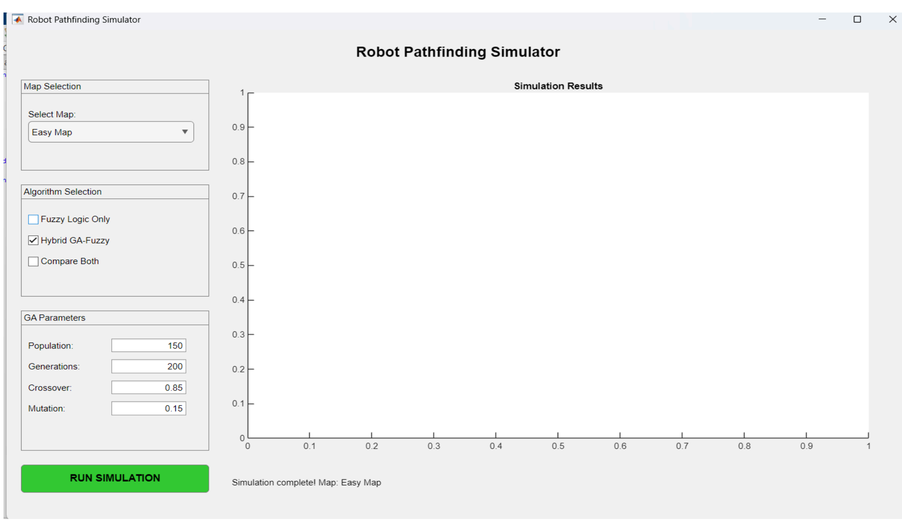


Figure 5: GUI simulation for our group mini project

Method	Goal Reached	Success Rate (%)	Path Lengths (steps)	Collisions	Path Smoothnes s	Final Distance
Fuzzy logic	Yes	100	20	0	60.0	1.00
Hybrid GA-Fuzzy	Yes	100	76	0	15.8	1.41

Table 5.1: Results on Simple Map

Method	Goal Reached	Success Rate (%)	Path Lengths (steps)	Collisions	Path Smoothness	Final Distance
Fuzzy logic	No	0	8	493	100.0	9.00
Hybrid GA-Fuzzy	Yes	100	83	0	13.3	1.41

Table 5.2: Results on Complex Map

6.0 Evaluation and Discussion

Simple Map:

Based on the simulation results, both the Fuzzy Logic and Hybrid GA–Fuzzy approaches successfully reached the goal in the Easy Map with zero collisions and a 100% success rate. However, clear differences in their path characteristics can be observed.

The Fuzzy Logic controller produced a much shorter and smoother path, completing the navigation in 20 steps with a path smoothness of 60%. This indicates efficient, goal-directed motion in a simple environment where local decision-making is sufficient. In comparison, the Hybrid GA–Fuzzy system generated a longer path of 76 steps with lower smoothness (15.8%), despite still avoiding all obstacles. This behavior is due to the genetic algorithm's exploratory nature, which prioritizes global feasibility rather than local optimality.

Overall, for simple environments, Fuzzy Logic outperforms the hybrid approach in efficiency and smoothness, while the Hybrid GA–Fuzzy method remains robust but unnecessary for low-complexity maps.

Hard Map:

In the Hard Map environment, a clear performance difference is observed between the Fuzzy Logic and Hybrid GA–Fuzzy approaches. The Fuzzy Logic controller failed to reach the goal, achieving a 0% success rate with a very high number of collisions (493). The short path length of 8 steps indicates that the robot became trapped and repeatedly collided with obstacles, demonstrating the local minima problem.

In contrast, the Hybrid GA–Fuzzy system successfully reached the goal with a 100% success rate and zero collisions. Although the generated path was longer (83 steps) and less smooth (13.3%), the robot was able to navigate safely through the narrow and complex obstacle layout.

Overall, the results show that Fuzzy Logic alone is not suitable for complex environments, while the Hybrid GA–Fuzzy approach effectively overcomes local minima by incorporating global path planning, making it significantly more robust in difficult maps.

7.0 Limitations and Future Improvements

While the proposed Hybrid Fuzzy-GA controller successfully achieved the project objectives, there are inherent limitations in the current simulation that provide opportunities for future research and development.

7.1 System Limitations

- **Kinematic Constraints:** The current simulation models the robot as a "point agent" with zero mass and instantaneous turning. Real-world robots have kinematic constraints (such as turning radius) and dynamic constraints (inertia, friction) that were not accounted for in this mathematical model.
- **Static Environments:** The maps used in this project were static, meaning obstacles did not move. In a real-world scenario, a robot must react to dynamic obstacles (e.g., walking humans or other machines), which requires a real-time re-planning capability that a standard Genetic Algorithm may struggle to provide due to computation time.
- **Discrete Movement:** The grid-based movement restricts the robot to 90-degree turns and fixed step sizes. This results in "jagged" paths rather than smooth, continuous trajectories.

7.2 Recommendations for Future Work

- **Dynamic Obstacle Avoidance:** Future iterations could implement a "Rolling Horizon" GA or a fully reactive Fuzzy Controller to handle moving obstacles in real-time.
- **Hardware Implementation:** The code could be ported to a physical robot (such as a TurtleBot or Arduino-based rover) equipped with ultrasonic sensors to validate the simulation results against real-world sensor noise and motor slip.
- **Smoother Trajectories:** Replacing the discrete grid system with a continuous coordinate system would allow for smoother, more natural curves and more efficient pathfinding.

8.0 Conclusion

This mini-project successfully designed, implemented, and evaluated a Hybrid Fuzzy-Genetic Algorithm (Fuzzy-GA) navigation system for a mobile robot operating in grid-based environments. The primary objective was to address the limitations of traditional reactive navigation methods, particularly the local minima problem, by combining fuzzy logic reasoning with global optimization through genetic algorithms.

The simulation results demonstrate that Fuzzy Logic alone is effective in simple environments, as observed in the Easy Map, where the robot achieved a 100% success rate with a short and smooth path and zero collisions. This confirms that fuzzy logic is well-suited for real-time decision-making when obstacle layouts are sparse and do not create complex navigation traps. Its rule-based nature allows the robot to react efficiently to local sensor information with minimal computational overhead.

However, the limitations of pure fuzzy logic become visible in complex environments. In the Hard Map scenario, the Fuzzy Logic controller failed to reach the goal and exhibited excessive collisions due to its reliance on local information only. The robot became trapped in narrow corridors and U-shaped obstacle configurations, clearly demonstrating the local minima problem commonly encountered in reactive navigation systems.

In contrast, the Hybrid Fuzzy–GA system consistently outperformed the Fuzzy Logic approach in complex environments. By integrating fuzzy risk evaluation into the genetic algorithm’s fitness function, the hybrid system was able to balance safety and goal progression effectively. The GA provided global exploration capabilities, enabling the robot to escape local traps, while the fuzzy logic component ensured that unsafe paths were penalized. As a result, the hybrid system successfully reached the goal in the Hard Map with zero collisions, albeit with a longer and less smooth path.

Overall, this project confirms that the Hybrid Fuzzy-Genetic Algorithm approach provides a powerful and flexible framework for mobile robot navigation. The findings of this study highlight the potential of soft computing techniques in autonomous robotics and provide a strong foundation for future enhancements toward more realistic and dynamic environment.

9.0 Contribution

Name	Matric No.	Contribution	Estimated Contribution
AHMAD SYAHMI WAJDI BIN AHMAD SHUKRI	2310663	-Designed and implemented a complete Hybrid Fuzzy-GA. -Develop the fuzzy logic controller. -Conducted all the simulations, debugging, and performance evaluations for simple and complex maps. -Design the graphical user interface (GUI). -Responsible for technical documentation (report).	70%
MUHAMMAD IRFAN BIN MOHD NAZIRUDIN	2414855	-Responsible for technical documentation (report). -Responsible for slide presentation.	15%
NOR EFFENDI BIN ANUAR	2310519	-Assisted in the design in fuzzy logic controller. -Contributed to the development of a fuzzy rule base.	15%

APPENDIX

Appendix A: AI Usage Log

AI Tool Used: ChatGPT (OpenAI) and Gemini (Google)

Purpose of Use: To assist in generating the initial MATLAB simulation framework, debugging the Genetic Algorithm fitness function, and structuring the final report.

Examples of Key Prompts Used:

1. *"Generate a MATLAB script for a 20x20 grid map with slalom obstacles."*
2. *“Improve the code to include simulation and performance evaluation.”*
3. *“Generate a structured academic report based on MATLAB code.”*

Verification Strategy: All AI-generated code snippets were manually reviewed, tested, and modified by the team. Specifically, the map coordinates were manually adjusted to create the "Slalom" effect, and the fitness function weights were tuned to prioritize collision avoidance over speed. We verify that we understand the code implemented and take full responsibility for the final submission.

Appendix B: GitHub Repository Link:
https://github.com/syahmi1410/Computational-Intelligence-Mini-Project.git