INTERNATIONAL ISLAMIC UNIVERSITY MALAYSIA
Garden of Knowledge and Virtue

TAWHIDIC EPISTEMOLOGY LEADING THE WAY UMMATIC EXCELLENCE LEADING THE WORLD
KHALĪFAH • AMĀNAH • IQRA' • RAḤMATAN LIL-'ĀLAMĪN

**MCTA 3371 - COMPUTATIONAL INTELLIGENCE**

# INTELLIGENT MOBILE ROBOT NAVIGATION USING COMPUTATIONAL INTELLIGENCE TECHNIQUES

**MINI PROJECT**

**SEMESTER 1 2025/2026**

**SECTION 2**

**LECTURER: DR. AZHAR BIN MOHD IBRAHIM**

**KULLIYAH OF ENGINEERING**

| **GROUP** | | **NAME** | **MATRIC NO.** |
|---|---|---|---|
| 5 | 1 | AHMAD NIZAR BIN AMZAH | 2312111 |
| | 2 | TAN YONG JIA | 2319155 |
| | 3 | MUHAMMAD IRSYAD HAZIM BIN ROZAINI | 2310303 |

**DATE OF SUBMISSION**

Wednesday, 21st January 2026

## Table of Contents

## 1. Introduction

Autonomous robot navigation in unknown or partially structured environments involves significant uncertainty arising from noisy sensor data, dynamic obstacles, and imprecise decision boundaries. Conventional deterministic control approaches often require precise models and hand-tuned parameters, which limits their adaptability across different environments. To address these challenges, soft computing techniques provide flexible and robust alternatives that can tolerate imprecision while producing effective navigation behaviour.

This project proposes a Genetic Algorithm (GA)–optimised fuzzy logic navigation system for a simulated mobile robot. The robot navigates in two-dimensional environments containing obstacles of varying complexity, including open spaces and maze-like layouts. Fuzzy logic is employed to perform real-time navigation decisions based on approximate distance sensor information and goal direction, while a Genetic Algorithm is used to evolve optimal controller parameters automatically.

Instead of evolving fuzzy rules, the system focuses on optimising behavioural parameters such as sensor sensitivity, turning strength, speed, goal attraction, and recovery thresholds. This approach preserves the interpretability of the fuzzy rule base while allowing the controller to adapt its behaviour to different environments through evolutionary optimisation.

The system supports multiple maps, with each environment maintaining its own evolved genome. Performance is evaluated through simulation by analysing navigation success, efficiency, and collision behaviour. The results demonstrate how combining fuzzy reasoning with evolutionary computation can improve autonomous navigation robustness and adaptability.

## 2. Project Methodology

The project methodology follows a clear flow from perception to decision-making and optimisation, implemented entirely within a simulation framework. The system operates through three main stages: environment sensing, fuzzy-based control, and genetic optimisation.

First, the robot perceives its surroundings using seven simulated distance sensors distributed across its front, left, and right directions. These sensors are implemented using ray casting, providing distance measurements up to a fixed sensing range. The raw sensor data captures obstacle proximity and forms the basis for subsequent control decisions.

Second, navigation decisions are generated using a fuzzy logic controller. At each simulation step, sensor readings are grouped into three aggregated inputs representing left, front, and right obstacle distances. In parallel, the angular difference between the robot's current heading and the goal direction is computed. These inputs are fed into the fuzzy controller to determine the robot's turning angle, which is combined with a constant forward motion.

To improve robustness, several behaviour-handling mechanisms are incorporated. A stuck detection mechanism monitors the robot's movement over time and triggers corrective turning when insufficient progress is detected. A goal override strategy allows direct steering toward the goal when the robot is close and the path ahead is sufficiently clear. Additionally, collision recovery behaviour is implemented using temporary reverse motion and cooldown periods.

The third stage involves Genetic Algorithm optimisation. A population of candidate genomes is generated, with each genome encoding a unique set of fuzzy controller parameters. Each genome is evaluated by deploying a robot in the selected environment and measuring its navigation performance over a fixed number of steps. Fitness is computed based on goal achievement, number of steps taken, distance to the goal, and collision penalties.

Through repeated generations involving tournament selection, uniform crossover, mutation, and elitism, the Genetic Algorithm evolves increasingly effective parameter configurations. The best-performing genome for each environment is stored and can be deployed for navigation after the evolutionary process is complete.

## 3. Soft Computing Modelling

The navigation system is based on a hybrid soft computing model that integrates fuzzy logic control with Genetic Algorithm–based optimisation. Each technique addresses a different aspect of the navigation problem while complementing the other.

## I. Fuzzy Logic Controller

The fuzzy logic controller is responsible for real-time navigation decisions under uncertainty. It processes four input variables: left distance, front distance, right distance, and goal direction angle. Distance inputs are derived from the robot's simulated sensors and represent obstacle proximity, while the goal angle represents the relative orientation of the target location.

Distance inputs are described using three linguistic terms: near, medium, and far. The membership functions for these terms are parameterised using the genome's sensitivity value, allowing the perception of obstacle proximity to adapt through evolution. The goal direction input is represented using linguistic terms left, ahead, and right.

The output variable of the fuzzy system is the robot's turning angle, defined using linguistic terms such as hard left, left, straight, right, and hard right. The range of this output is controlled by the genome's turn power parameter. A fixed and interpretable rule base combines obstacle avoidance and goal-seeking behaviour, prioritising safety when obstacles are near and goal alignment when the path is clear.

## II. Genetic Algorithm Optimisation

The Genetic Algorithm optimises the fuzzy controller's parameters rather than its rules. Each genome encodes parameters including sensor sensitivity, turning power, movement speed, goal attraction force, goal override distance, goal clearance threshold, and stuck detection threshold.

During evolution, each genome is evaluated by simulating a robot navigating the environment using that parameter configuration. Fitness is calculated based on successful goal reaching, navigation efficiency, collision avoidance, and remaining distance to the goal if unsuccessful. Tournament selection is used to choose parent genomes, uniform crossover combines parental traits, and mutation introduces controlled randomness. Elitism ensures that the highest-performing genomes are preserved across generations.

## III. Hybrid System Characteristics

The hybrid design separates decision-making from learning. Fuzzy logic provides transparent and explainable control during navigation, while the Genetic Algorithm performs offline optimisation to adapt behaviour across environments. This combination allows the system to handle uncertainty effectively while maintaining stability and interpretability.

Overall, the soft computing model demonstrates how evolutionary optimisation can enhance fuzzy-based navigation without sacrificing clarity or control structure.

## 4. Implementation Details

### I. System Overview

The intelligent mobile robot navigation system was implemented in Python using a hybrid approach that combines fuzzy logic control, heuristic-based decision mechanisms, and genetic algorithm (GA) optimization. The simulation environment was developed using the Pygame library, enabling real-time visualization of robot motion, sensor sensing, obstacle interaction, and navigation performance.

### II. Sensor Modeling and Inputs

The robot is modeled as a circular mobile agent equipped with seven simulated range sensors distributed across angles of −90°, −50°, −25°, 0°, 25°, 50°, and 90°. These sensors perform ray-based distance sensing to detect nearby obstacles within a defined sensing range.

To reduce rule complexity while preserving directional awareness, sensor readings are grouped into three representative inputs: left, front, and right. Each input is computed by taking the minimum distance among its corresponding sensors. These three distance values serve as obstacle proximity inputs to the fuzzy logic controller.

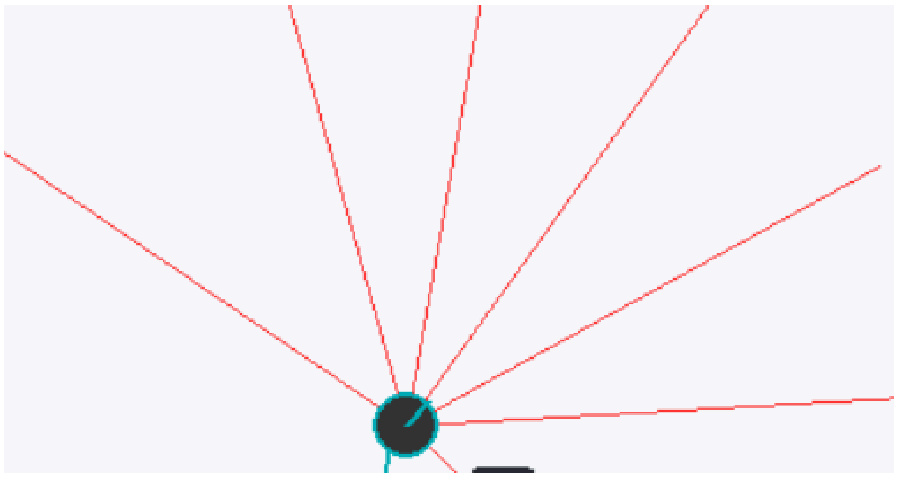


**Figure 4.1** Displays the 7 sensors in the Intelligent Mobile Robot

## III. Fuzzy Logic Controller

The fuzzy logic controller forms the primary navigation mechanism of the system. It processes the left, front, and right obstacle distances together with the angular difference between the robot's current heading and the target goal direction. Fuzzy membership functions are used to represent obstacle proximity levels and directional error in a smooth and interpretable manner.

The fuzzy inference system outputs a steering angle that determines the robot's turning behavior at each simulation step. This allows the robot to react smoothly to obstacle configurations while maintaining forward progress toward the goal.

## IV. Heuristic Overrides and Recovery Mechanisms

While fuzzy logic provides adaptive obstacle avoidance, additional heuristic mechanisms are incorporated to enhance robustness. When the robot is sufficiently aligned with the goal and no immediate frontal obstacle is detected, a goal-alignment heuristic biases the steering direction directly toward the target. This prevents unnecessary detours and improves convergence speed.

A recovery mechanism is also implemented to handle stuck or near-deadlock situations. When insufficient movement is detected over a period of time, the system forces corrective turning to help the robot escape local minima. These heuristics complement the fuzzy controller without replacing its core functionality.

## V. Genetic Algorithm Optimization

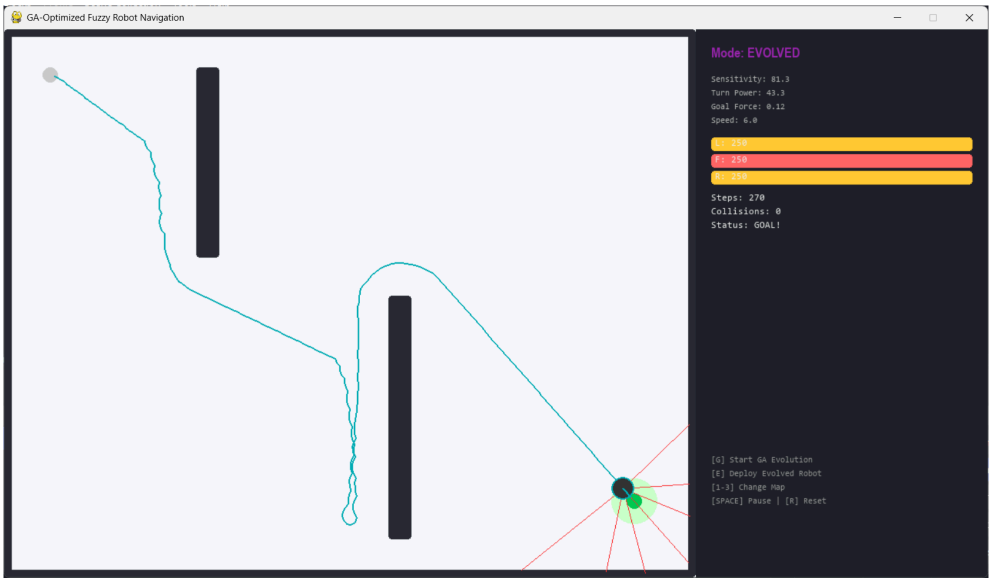
A genetic algorithm is employed to optimize the numerical parameters governing the fuzzy controller and heuristic behaviors. Each genome represents a candidate set of parameters, including sensor sensitivity, turning power, movement speed, goal attraction strength, and recovery thresholds.

The GA evaluates each genome by running the robot navigation simulation and computing a fitness score based on goal completion, distance to the target, collision count, and number of steps taken. Selection, crossover, mutation, and elitism are applied across generations to evolve increasingly effective navigation controllers. The optimized parameters are then deployed for real-time navigation.

## 5. Simulation Setup

### I. Environment Design

The simulation environment consists of a two-dimensional bounded map containing static obstacles, a predefined robot start position, and a target goal location. Multiple environments with varying levels of complexity are used, including narrow passages and open layouts with scattered obstacles. These environments are designed to evaluate navigation robustness under different spatial constraints.



```
Sensitivity: 81.3
Turn Power: 43.3
Goal Force: 0.12
Speed: 6.0
```

```
Steps: 270
Collisions: 0
Status: GOAL!
```

**Figure 5.1** The simulation environment in the GUI

### II. Evaluation Criteria

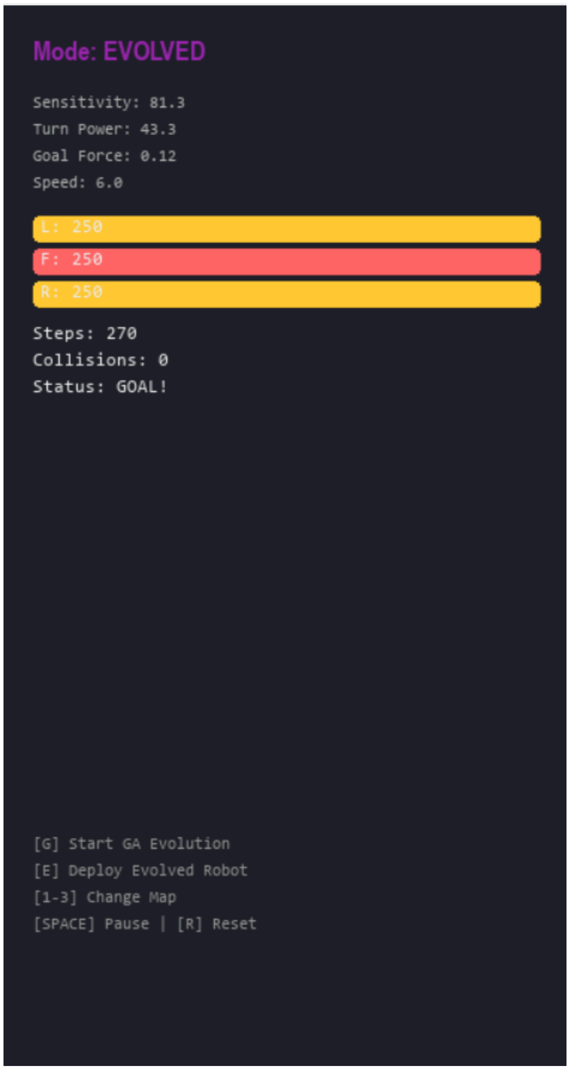
The robot’s objective is to reach the goal while avoiding collisions. A goal is considered reached when the robot enters a circular threshold region surrounding the target. Collisions are detected using geometric intersection between the robot and obstacles. Repeated collisions beyond a defined limit result in navigation failure.

For genetic algorithm evaluation, each genome is tested over a maximum number of simulation steps. Performance metrics include distance to the goal, number of steps taken, collision count, and successful goal completion. These metrics are combined into a fitness score that guides evolutionary optimization.

## III. User Interface and Controls

The simulation interface provides real-time visualization of the robot's trajectory, sensor rays, obstacle positions, and goal location. Performance statistics are displayed during execution to monitor navigation quality.

User controls allow starting and stopping the genetic algorithm, deploying the best evolved controller, resetting the robot's position, pausing the simulation, and switching between different environments. Accelerated simulation is used during GA evaluation to reduce computation time, while standard frame rates are applied during demonstrations.



```
Mode: EVOLVED

Sensitivity: 81.3
Turn Power: 43.3
Goal Force: 0.12
Speed: 6.0

L: 250
F: 250
R: 250

Steps: 270
Collisions: 0
Status: GOAL!

[G] Start GA Evolution
[E] Deploy Evolved Robot
[1-3] Change Map
[SPACE] Pause | [R] Reset
```

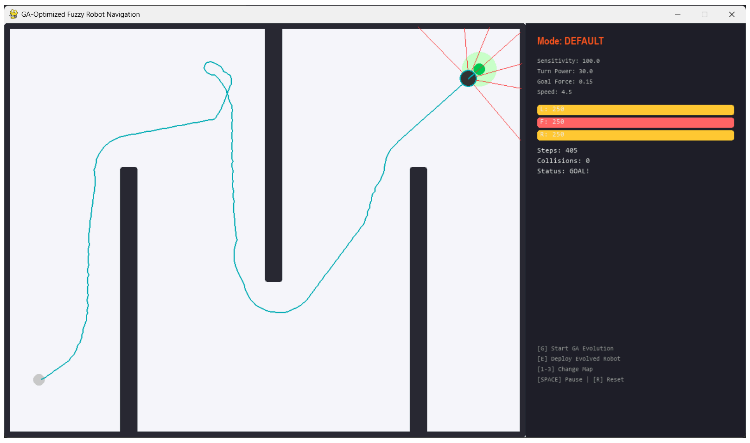
**Figure 5.2** The user interface & controls in the GUI

## 6. Results and Analysis

### I. Navigation Performance

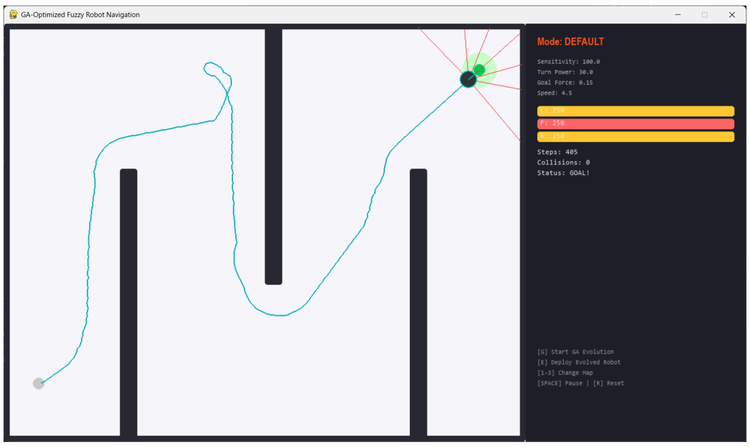
The navigation performance of the proposed system was evaluated by comparing the baseline fuzzy logic controller with the hybrid Genetic Algorithm–Fuzzy Logic (GA–Fuzzy) controller across multiple maps. Performance metrics included the number of steps required to reach the goal, collision count, and successful goal completion.

For the baseline fuzzy controller, the navigation behavior was fully deterministic because all control parameters were fixed. Repeated executions on the same map produced identical trajectories, step counts, and collision numbers. For example, in the Training Room map, the fuzzy controller consistently required approximately 405 steps to reach the target while maintaining the same parameter values for sensitivity, turn power, goal force, and speed.



```
Mode: DEFAULT

Sensitivity: 100.0
Turn Power: 30.0
Goal Force: 0.15
Speed: 4.5

L: 250
F: 250
R: 250

Steps: 405
Collisions: 0
Status: GOAL!
```

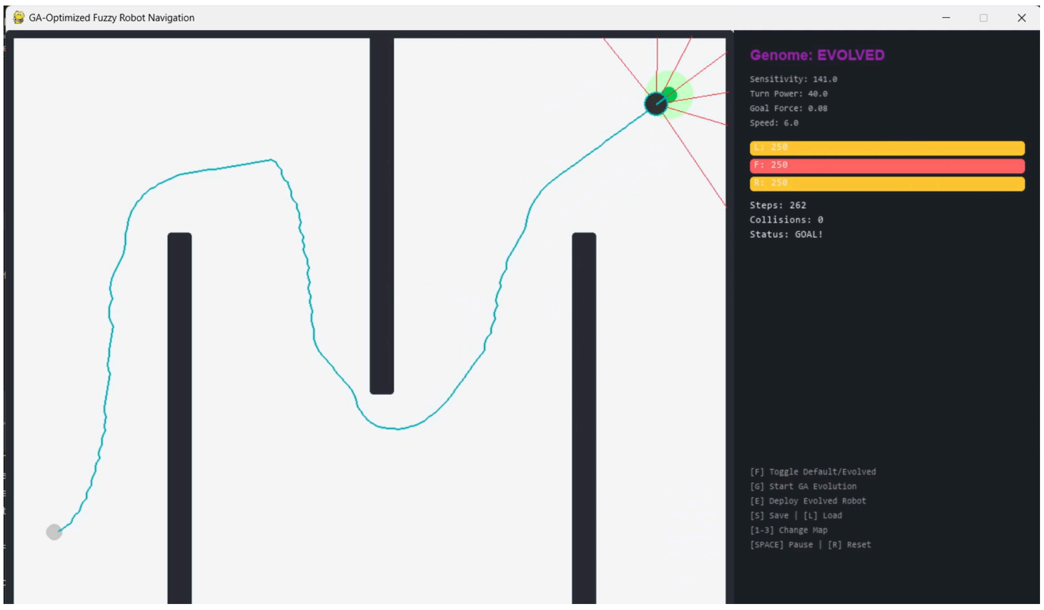
**Figure 6.1** First run using Fuzzy logic controller



```
Mode: DEFAULT

Sensitivity: 100.0
Turn Power: 30.0
Goal Force: 0.15
Speed: 4.5

L: 250
F: 250
R: 250

Steps: 405
Collisions: 0
Status: GOAL!
```

**Figure 6.2** Second run using Fuzzy logic controller

After applying the hybrid GA–Fuzzy optimization, a significant improvement in navigation efficiency was observed. The evolved controller reduced the number of steps in the same Training Room environment to 262 steps, representing a reduction of more than 35% in path length. At the same time, collision occurrences were reduced and smoother trajectories were produced. Similar improvements were observed in the Maze Complex and Open Field environments, where the optimized controllers demonstrated shorter paths and faster convergence toward the goal.



```
Genome: EVOLVED
Sensitivity: 141.0
Turn Power: 40.0
Goal Force: 0.08
Speed: 6.0
L: 250
F: 250
R: 250
Steps: 262
Collisions: 0
Status: GOAL!

[F] Toggle Default/Evolved
[G] Start GA Evolution
[E] Deploy Evolved Robot
[S] Save | [L] Load
[1-3] Change Map
[SPACE] Pause | [R] Reset
```

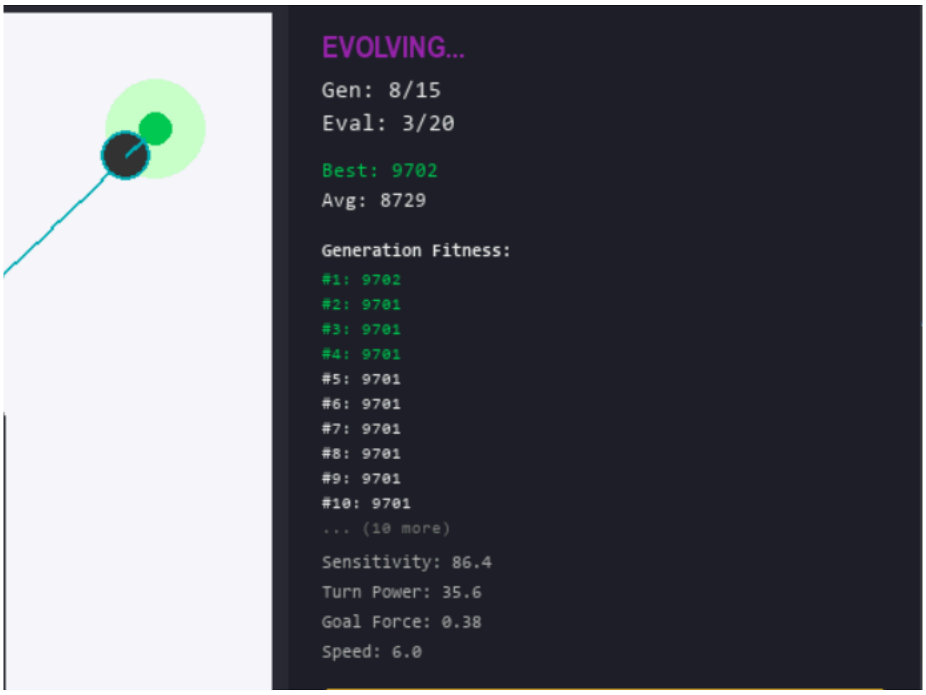
**Figure 6.3** Robot path using GA-Fuzzy hybrid approach

These results confirm that genetic optimization substantially enhances navigation efficiency by tuning the fuzzy controller parameters according to the environment.

## II. Impact of Genetic Optimization

The impact of genetic optimization was analyzed by observing the evolution of fitness values across generations and comparing the optimized controller with the baseline fuzzy controller.

The genetic algorithm employed a population size of 20 individuals over 15 generations, with elitism applied by carrying forward the top **four best-performing genomes** into each subsequent generation. In each generation, only 16 new individuals were generated through crossover and mutation, while the four elite genomes preserved the best solutions discovered so far. This strategy ensured steady convergence while preventing the loss of high-quality controllers.



```
EVOLVING...
Gen: 8/15
Eval: 3/20

Best: 9702
Avg: 8729

Generation Fitness:
#1: 9702
#2: 9701
#3: 9701
#4: 9701
#5: 9701
#6: 9701
#7: 9701
#8: 9701
#9: 9701
#10: 9701
... (10 more)
Sensitivity: 86.4
Turn Power: 35.6
Goal Force: 0.38
Speed: 6.0
```

**Figure 6.4** Shows the carry forward of the top four best-performing genomes

Fitness was defined as a function of goal achievement, number of steps, distance to the goal, and collision penalties. Successful navigation was strongly rewarded, while longer paths and collisions were penalized. During training, the genetic algorithm evaluated multiple candidate controllers across generations and retained the best-performing individuals through elitism. This ensured that high-quality solutions were preserved and refined over successive generations.

The optimized controllers exhibited adjusted sensitivity, turning power, goal attraction force, and speed parameters that were tuned to the obstacle configurations of each map. Unlike the baseline fuzzy controller, which produced identical behavior in every run due to fixed parameters, the GA–Fuzzy controller adapted its parameters during training and converged to environment-specific optimized solutions.

| Parameter | Fuzzy | Fuzzy-GA |
|---|---|---|
| Sensitivity | 100.0 | 141.0 |
| Turn Power | 30.0 | 40.0 |
| Goal Force | 0.15 | 0.08 |
| Speed | 4.5 | 6.0 |
| | | |
| Steps | 405 | 262 |
| Collisions | 0 | 0 |

**Table 6.1** Fuzzy & GA-Fuzzy comparison

The reduction in step count from 405 steps (baseline fuzzy) to 262 steps (optimized hybrid) demonstrates the strong positive impact of genetic optimization. This improvement confirms that the genetic algorithm successfully refined the fuzzy controller parameters to produce shorter, safer, and more efficient navigation paths.

## III. Robustness and Limitations

The proposed hybrid system demonstrated strong robustness in handling different navigation environments. Separate optimized genomes were evolved and stored for each map, allowing the robot to maintain high performance across multiple obstacle configurations. The optimized controllers consistently achieved goal completion with fewer collisions and reduced navigation time compared to the baseline fuzzy controller.

However, several limitations remain. First, the optimization process is computationally intensive, as each genome evaluation requires a full navigation simulation. Although elitism reduces unnecessary reevaluation of high-quality solutions, training still requires significant computation time.

Second, the evolved controllers are environment-specific. A genome optimized for one map may not generalize well to a different environment without retraining. This limits direct transferability between maps and requires separate training sessions for each new environment.

Finally, the system operates in a simulated two-dimensional environment with idealized sensors and motion models. Real-world uncertainties such as sensor noise, wheel slippage, and dynamic obstacles are not considered. Future work may include extending the system to real robotic platforms, incorporating adaptive online learning, and testing generalization across unseen environments.

Despite these limitations, the hybrid GA–Fuzzy approach demonstrates clear advantages over conventional fuzzy control by achieving significantly improved navigation efficiency and adaptive behavior.

## 7. Discussion

The results of the simulation demonstrate that the proposed GA-optimized fuzzy logic controller is effective in solving the mobile robot navigation problem in environments with obstacles. The robot was able to navigate from the start position to the goal while avoiding collisions in both simple and complex maps, indicating that the fuzzy logic controller successfully handled uncertainty in sensor readings and decision-making.

The use of Genetic Algorithm (GA) significantly improved the performance of the fuzzy controller. Instead of relying on manually tuned parameters, the GA automatically optimized key fuzzy parameters such as sensitivity, turning power, robot speed, and goal influence. This optimization resulted in smoother navigation paths, reduced collisions, and fewer time steps required to reach the goal when compared to the default fuzzy controller.

Simulation results also showed that each environment required different optimal parameters, particularly in maps with narrow passages and dense obstacles. By evolving and storing a separate genome for each map, the system demonstrated adaptability to different navigation challenges. This highlights the advantage of hybrid computational intelligence approaches, where learning and optimization enhance rule-based systems.

Despite its effectiveness, some limitations were observed. In highly constrained areas, the robot occasionally exhibited oscillatory behavior before escaping tight spaces. This behavior is mainly due to the local nature of fuzzy rules and the absence of global path planning. Additionally, the optimization process increases computational time during evolution, making real-time learning less suitable for hardware implementation without further optimization.

Overall, the discussion confirms that the hybrid GA–Fuzzy approach provides a robust and flexible solution for intelligent mobile robot navigation in uncertain environments.

## 8. Conclusion

This project successfully developed an intelligent mobile robot navigation system using a hybrid Genetic Algorithm–Fuzzy Logic approach. The fuzzy logic controller enabled the robot to make navigation decisions under uncertainty, while the genetic algorithm optimized the controller parameters to enhance performance.

Simulation results showed that the robot could reliably reach the goal while avoiding obstacles across different environments. The optimized controller achieved better navigation efficiency, smoother paths, and reduced collision occurrences compared to a non-optimized fuzzy controller.

The project demonstrates that hybrid computational intelligence techniques are well-suited for mobile robot navigation tasks where precise mathematical models are difficult to obtain. Future work could include incorporating dynamic obstacles, integrating global path planning methods, or implementing the system on a real mobile robot platform.

In conclusion, the objectives of the project were successfully met, and the results validate the effectiveness of GA-optimized fuzzy logic for intelligent robot navigation.

## 9. AI Usage Appendix

### A. Application of Artificial Intelligence in the Project

This project applies artificial intelligence techniques in the form of fuzzy logic control and genetic algorithms to achieve intelligent mobile robot navigation. Fuzzy logic is used to handle uncertainty and imprecision in sensor data, enabling smooth and adaptive steering decisions. Genetic algorithms are employed to optimize the numerical parameters governing the fuzzy controller and heuristic behaviors through evolutionary processes.

### B. AI-Assisted Tools Usage

AI-assisted tools (Claude AI) were used during the development phase to support understanding of fuzzy logic concepts, genetic algorithm structures, and Python programming techniques. These tools assisted in learning, debugging, and refining the implementation.

### C. Scope and Limitations of AI Usage

All system design decisions, simulations, evaluations, and performance analyses were conducted by the project members using the generated code. No AI tools were used to generate experimental results, simulation outputs, or manipulate data.

### D. Academic Integrity and Responsibility

The use of AI-assisted tools adhered to academic integrity guidelines and served solely as a supplementary aid to enhance comprehension and development efficiency. The project members take full responsibility for the accuracy, originality, and authenticity of the work presented in this report.

## 10. Individual Contribution

| No. | Name | Parts Distribution |
|---|---|---|
| 1. | Ahmad Nizar Bin Amzah | Introduction<br>Project Methodology<br>Soft Computing Modelling |
| 2. | Tan Yong Jia | Soft Computing Modelling<br>Simulation Setup<br>Discussions<br>Conclusion |
| 3. | Muhammad Irsyad Hazim Bin Rozaini | Implementation Details<br>Simulation Setup<br>Results and Analysis<br>AI Usage Appendix |