The Evolving Maze

Artificial Intelligence Project Report

Submitted By

Zakyas Ali
22K-4709

Mehtab Ahmed
22K-4771

Ahmed Hussain
22K-4778

Section: B(CY)-6B

Department of Cyber Security
National University of Computer & Emerging Sciences

➢ Project Title:

The Evolving Maze — An AI-Driven Adaptive Maze Game

➢ Objective:

To develop an unconventional board game where the maze evolves in real time based on the player's progress using artificial intelligence techniques such as Genetic Algorithms, Manhattan Heuristic, and Breadth-First Search (BFS).

➢ Problem Statement:

Traditional maze games are static and predictable. This project introduces a dynamic challenge where the maze adapts in complexity depending on how close the player is to the goal. The goal is to make the game more engaging using AI, while ensuring the maze remains solvable.

➢ AI Techniques Used:

Technique	Purpose
Genetic Algorithm	To evolve new maze layouts dynamically.
Mutation Operator	To increase difficulty when players progress.
Manhattan Distance	To detect proximity to the target and trigger evolution.
BFS Pathfinding	To ensure all generated mazes are solvable.

➢ Tools and Technologies:

- Programming Language: Python 3.x
- Libraries: random, copy, collections
- IDE: VS Code / PyCharm / Jupyter
- Execution: Terminal / Command line

➢ Game Rules and Mechanics:

Game Flow:

1. A maze is generated with walls and paths.
2. The player starts at a defined position.
3. The target (goal) is randomly placed.
4. The player uses W, A, S, D keys to move.
5. If a move brings the player closer to the goal:
    - The maze evolves using Genetic Algorithm.
    - The evolved maze is checked to ensure solvability.
6. When the player reaches the goal:
    - A success message is printed and the game ends.

Winning Condition:

Reach the goal before getting trapped by an evolved maze.

➢ AI Algorithms in Detail:

Genetic Algorithm

- **Chromosome**: A 2D matrix representing the maze.
- **Fitness Function**: Shortest valid path from player to target (lower is better).
- **Selection**: Top-performing mazes are selected.
- **Crossover**: Mixes parts of two parent mazes.
- **Mutation**: Adds walls near the player or path to make it harder.

Manhattan Heuristic

- Computes the player's closeness to the goal.
- Triggers maze evolution if player gets closer.

Breadth-First Search (BFS)

- Ensures a path exists in the generated maze.
- Calculates shortest distance for fitness evaluation.

➢ Sample Output Screenshots: