SmartPac

Intelligent path-finding agent for Pac-Man using A* informed search algorithm

By:

Aya Osama Majar – sec 2

Basma Walid Elsayed – sec 2

Reem Mohammed Eldafrawy – sec 3

Dr. Sara Elsayed ELmetwally

1. Problem Formulation

1.1 Problem Definition

Classic arcade games like Pac-Man are ideal platforms for developing and testing intelligent agents. In this project, we aim to implement a one of algorithms used in developing intelligent agents; which is: A* informed search algorithm.

The agent's goal is to collect all dots (pellets) while avoiding ghosts and walls efficiently.

1.2 Previous Solutions

We have chosen a game that was implemented using all early AI algorithms, or even was developed with no AI, like random movement, after search we concluded that *A** is generally the **Best Choice** for Pac-Man. It provides the optimal balance of:

- Finding shortest paths to dots (efficiency)
- Avoiding ghosts (safety)
- Computational feasibility (performance)

2. SmartPac - Subsystems

Subsystem	Description
Maze Renderer	Builds and draws a grid-based maze layout with walls and dots.
Pac-Man Agent	Autonomous player using A* search to reach the nearest pellet.

Ghost Agent	Dynamic enemies with randomized movement logic.
Collision Detection	Detects when Pac-Man touches a ghost or eats a dot.
Game Loop	Integrates real-time updates, rendering, and agent movement.

3. SmartPac - Technologies

- **Python**: Core programming language.
- **Pygame**: For game simulation and rendering.
- *A Search**: Heuristic pathfinding used in the player agent.
- **OOP Design**: Maze, Player, and Ghosts as separate classes.

4. Future Work

Enhance the player algorithm to be more efficient and include a penalty cost when the ghost is near the player which will make the player survive longer and avoid the ghost