Ghost Maze: The Haunted Board Game - Project Report

1. Introduction

The Ghost Maze project combines traditional maze navigation with dynamic AI-driven challenges. Players must escape a haunted maze while being chased by intelligent ghosts using advanced pathfinding algorithms. The primary AI technique utilized is the A* algorithm, which helps ghosts optimize their movement and predict the player's potential escape routes.

2. Problem Description

The game is set in a grid-based maze where the player must navigate to collect tokens while avoiding ghosts. The main challenge was programming the AI to allow ghosts to navigate the maze and chase the player efficiently, using A* search and heuristic evaluation.

3. Features of the Game

1. Dynamic Maze: Certain walls shift after a few turns, changing the paths available. This dynamic maze structure forces the player to constantly adjust their strategy.
2. AI-Controlled Ghosts: Ghosts are not random; instead, they use the A* pathfinding algorithm to chase the player intelligently. The ghosts will evaluate different paths and move towards the player's predicted location.
3. Power-ups: The player can collect power-ups that provide temporary benefits like speed boosts or ghost immunity. These power-ups add a strategic layer to the game.
4. Traps: Certain areas in the maze contain traps that slow the player down, increasing the challenge as they try to evade ghosts.
5. Score System: The player scores points by collecting items (like dots or coins) while avoiding the ghosts. Each collection increases the score and adds to the player's progress.

6. Winning Conditions: The player wins by reaching the maze's exit before being caught by the ghosts. If the ghosts catch the player, they win the game.

4. Literature Review

Pathfinding in Games: Pathfinding algorithms are crucial for navigating through complex environments filled with obstacles. Classic algorithms like A* have been extensively used for game AI, enabling characters (like ghosts) to find the shortest paths efficiently.

5. Methodology

Maze Structure: The maze is designed as a grid with traversable and non-traversable cells (walls). A* pathfinding is used to compute optimal paths for each ghost towards the player's position.

Ghost AI:
- Red Ghost uses A* with a heuristic based on Manhattan distance.
- Green Ghost uses a similar A* implementation.
- Orange Ghost uses Context Dependent Subgoaling A*, an optimization that limits node expansion for better performance.

6. Implementation

The game was implemented using Python and the Pygame library. The ghost AI calculates the next best move based on the player's position and the state of the maze. Power-ups and traps add strategic elements to the game.

7. Experiments and Results

Several experiments were conducted to test the efficiency of each pathfinding algorithm in real-time scenarios.
The Context Dependent Subgoaling A* algorithm was found to be the most efficient, reducing computation time by approximately 3 times compared to A*
and 18 times faster than Breadth-First Search.

Algorithm	Nodes Generated	Nodes Expanded	Computation Time (s)
A*	189	64	0.0015
CDSA*	114	39	0.0005
BFS	531	188	0.0090

8. Conclusion

The Ghost Maze game successfully demonstrates AI pathfinding in a dynamic maze environment. The Context Dependent Subgoaling A* algorithm outperforms other classical methods in both time and space complexity. Future work could include using Reinforcement Learning to make ghosts adapt to player strategies over time.