National University
Of Computer and Emerging Sciences

Project Title: Mind-Maze

Submitted By: Ahmed Abdullah 22K-4449

Daniyal Ahmed 22k-4601

Course: AI

Instructor: Miss. Alishba Subhani

Submission Date: May 11, 2025

1. Executive Summary

Project Overview:

This project presents a turn-based tile strategy game developed using Python and Tkinter. It simulates a player-vs-AI competition, where the AI leverages the Minimax/Alpha-beta algorithm to make optimal decisions. The game features a grid-based board, booster tiles, and alternating player turns, with victory determined by board control and tile advantage.

2. Introduction

Background:

Inspired by classic grid-based strategy games, this project introduces a simplified tile conquest game aimed at evaluating AI strategic performance. The choice of Tkinter as the GUI framework and the implementation of Minimax with depth control allows for simulating real-time strategy decisions within a custom environment.

Objectives of the Project:

- Implement a tile-based strategy game in Python using Tkinter.
- Design and integrate an AI using the Minimax algorithm.
- Evaluate AI decisions based on game outcomes and response times.

3. Game description

Original Game Rules:

The original concept involves player alternately selecting tiles on a grid in minesweeper. Player compete to accumulate a higher total score by choosing optimal positions and not choosing tile with a mine. So in this code we combined minesweeper with bluff game to make it more interesting and challenging where players discover tiles and select weather to move to that tile or not.

Innovations and Modifications:

- Introduction of booster tiles that provide additional points.
- Implementation of a GUI using Tkinter.
- AI opponent using the Minimax algorithm for turn-based decisions

4. AI Approach and Methodology

AI Techniques Used:

The AI uses the Minimax algorithm to simulate all possible future moves up to a certain depth and selects the one that maximizes its chances of winning while minimizing the opponent's advantage.

Algorithm and Heuristic Design:

- Heuristics evaluate score difference, available boosters, and tile control.
- Depth-limited Minimax with alternating maximizing and minimizing levels.
- Alpha-beta pruning is implemented for performance improvements.

AI Performance Evaluation:

- Performance measured based on win ratio against human players and move decision time.
- At depth 3, average AI response time is under 2 seconds.
- Win rate in internal testing: ~60% against novice players.

5. Game mechanics and rules

Modified Game Rules:

- Grid-based board 6*6 board.
- Each turn, a player selects one unclaimed tile.
- Some tiles offer boosters like extra turns while some are traps which -1 points.
- Game ends when all tiles are claimed or player has reached upto some threshold set like we set up 20 points.

Turn-based Mechanics:

- Alternating turns between human and AI.
- Players can clain tiles and decide whether to move their or not if they don’t mov that tile is hidden thus confusing opponent what was there In that tile.

Winning Conditions:

- Player with the highest score at the end wins.

6. Implementation and development

Development Process:

- Initial board logic and Tkinter interface created.
- Booster tiles randomly assigned.
- Minimax logic developed and integrated with UI.
- Extensive testing done to tune heuristic evaluation.

Programming Languages and Tools:

- Language: Python
- Libraries: Tkinter (UI), random (tile assignment)
- Tools: like tkinter and also github.

Challenges Encountered:

- Balancing AI depth with performance as it was laging at higher depths.
- Managing state updates in Tkinter without freezing the interface.
- Designing an effective heuristic that scales with board size.
- Implementing the logic for the combined game of minesweeper and bluf.

7. Team contributions

Ahmed Abdullah: Developed Minimax algorithm and AI evaluation logic. Integrated AI with UI, managed game state logic.

Daniyal Ahmed : Designed game board, booster logic, and Tkinter interface. Conducted gameplay testing, heuristic tuning, and performance profiling.

8. Result and Discussions

AI Performance:

- Win Rate: ~90% against average human players
- Decision Time: ~1.5–2 seconds at depth 2
- Effectiveness: The AI demonstrates consistent strategy and avoids easily exploitable traps, showing promising behavior for a simple Minimax implementation as AI can easily calculate tiles of its benefit and act accordingly as most of time in tesing AI won.

9. References

- Russell, S., & Norvig, P. (2009). *Artificial Intelligence: A Modern Approach*.
- Tkinter Documentation: https://docs.python.org/3/library/tkinter.html
- Minimax Algorithm - GeeksforGeeks: https://www.geeksforgeeks.org/minimax-algorithm-in-game-theory/
- Python Official Documentation: https://docs.python.org/3/