CSE472s: Artificial Intelligence Course

Fall 2025

Term Project: Quoridor Game Implementation

Project Overview

In this project, you will develop a complete implementation of the abstract strategy board game Quoridor. This project will test your ability to implement game mechanics, develop a graphical user interface, create AI opponents of varying difficulty, and document your work professionally.

Game Description: Quoridor

Quoridor is a strategy board game that was invented by Mirko Marchesi and first released in 1997. The game has won several awards including the Mensa Mind Game award.

Game Rules

1. **Game Board**: The game is played on a 9×9 square board.
2. **Players**: 2-4 players can play, but you'll implement the 2-player version.
3. **Game Pieces**:
    - Each player has a pawn that begins at the center of their respective base line
    - Each player has 10 walls (5 walls in a 4-player game)
4. **Objective**: Be the first player to move your pawn to any square on the opposite side of the board.
5. **Movement**:
    - On each turn, a player must either move their pawn or place a wall
    - Pawns move one square orthogonally (not diagonally)
    - Players cannot move through walls or opponent pawns
    - If a player's pawn is adjacent to an opponent's pawn, the player can jump over the opponent's pawn (if there's no wall blocking)
    - If a jump is blocked by a wall, the player can move diagonally around the opponent's pawn
6. **Wall Placement**:
    - Walls are two squares long and are placed on the edges between squares
    - Walls cannot overlap or cross other walls
    - Walls cannot be placed to completely block a player's path to the goal (there must always be a valid path to the goal for each player)
    - Once placed, walls cannot be moved

Project Requirements

Core Requirements

1. **Game Implementation**:
    - Implement the complete ruleset of Quoridor for 2 players
    - Create a graphical user interface (GUI) for the game
    - Include game state visualization (board, pawns, walls, whose turn it is)
    - Implement valid move highlighting and error prevention for illegal moves
    - Include a path-finding algorithm to ensure wall placements don't completely block a player
2. **Game Modes**:
    - Human vs. Human (local play on the same computer)
    - Human vs. Computer (with AI opponent)
3. **User Interface**:
    - Clear visualization of the game board, pawns, and walls
    - Intuitive controls for moving pawns and placing walls
    - Turn indicator
    - Wall count display for each player
    - Game state messages (whose turn, invalid moves, winner announcement)
    - Game reset functionality
4. **Documentation**:
    - GitHub repository with complete source code
    - README file with:
        - Game description
        - Screenshots of the game in action
        - Installation and running instructions
        - Controls explanation
        - Link to demo video
    - Project report (PDF) including:
        - Design decisions and architecture
        - Implementation challenges and solutions
        - AI algorithm explanation
        - Assumptions made during development
        - References and resources used
5. **Demo Video**:
    - 3-5 minute video demonstrating:
        - Game setup and UI overview
        - Human vs. Human gameplay
        - Human vs. Computer gameplay

Bonus Features (for additional marks)

1. **AI Difficulty Levels**:
    - Implement multiple AI difficulty levels (e.g., Easy, Medium, Hard)
    - Document the algorithms used for each difficulty level
2. **Additional Features "Adds +10% of the total project total"** (choose one to implement):
    - Game state saving/loading
    - Undo/redo functionality
    - 4-player mode
    - Custom board sizes

Technical Requirements

1. **Programming Language**:
    - You may use any programming language and framework that supports GUI development
    - Popular choices include Python (with Pygame, Tkinter, or PyQt), Java (with JavaFX or Swing), JavaScript (with HTML5 Canvas for Web), or C++ (with SFML or Qt)
2. **Version Control**:
    - Create a public GitHub repository for your project
    - Make regular, meaningful commits showing your development process
    - Include a .gitignore file appropriate for your language/framework
3. **Code Quality**:
    - Well-organized, readable code with appropriate comments
    - Separation of concerns (game logic, UI, AI)
    - Avoid code duplication and use appropriate design patterns

Deliverables and Deadline

1. **GitHub Repository Link**:
    - Submit the URL to your public GitHub repository
    - The repository must include all source code, assets, and documentation
2. **Project Report**:
    - Submit a PDF report (6-10 pages) covering the topics outlined above
    - Include the link to your demo video in the report
3. **Demo Video**:
    - Upload to YouTube, Google Drive, or any similar platform
    - Include the link in both your report and README.md

Final Deadline: 21 December 2025 (11:59 PM 21-12-2025)

Grading Criteria

Component	Percentage	Description
Game Implementation	40%	Correctness, completeness, and quality of game rules implementation
User Interface	20%	Usability, intuitiveness, and visual appeal of the GUI
AI Implementation	20%	Quality and effectiveness of the computer opponent
Documentation	10%	Completeness and clarity of README and project report
Code Quality	10%	Organization, readability, and maintainability
Bonus Features	+10%	Implementation of bonus features (proportional to quantity and quality)

Submission Instructions

1. Ensure your GitHub repository is public and contains all required files
2. Submit the Project Report to the course LMS containing the link to the repo as well as the link to the demo video.
3. This is a team project. Each team should submit only ONE report. This submission is to be done by only one student (maybe the project leader can submit it).
4. The team should not contain more than 3 members.

Helpful Resources

Official Resources

- Official Quoridor Rules
- Quoridor on BoardGameGeek

Video Tutorials and Demonstrations

- Quoridor - Game Rules & Strategies Explained - Clear explanation of rules with animated examples
- How to Play Quoridor - Visual tutorial showing gameplay in action
- Quoridor World Championship Final Match - Expert-level gameplay to study advanced strategies
- Quoridor Gameplay and Strategy Tips - Analysis of effective tactics

Articles and Guides with Visualizations

- Quoridor Strategy Guide with Animated Examples
- Interactive Quoridor Tutorial - Online playable demonstration
- Mathematical Analysis of Quoridor - Blog with interactive visualizations of AI strategy
- Quoridor AI Development Guide - Article series on implementing Quoridor AI with diagrams

Technical Resources

- Path-finding Algorithm Tutorial - Interactive explanations of pathfinding algorithms
- Implementing Game Trees for Board Games - Guide for developing game AI
- Minimax Algorithm with Alpha-Beta Pruning Tutorial - Visual guide to AI decision making
- GUI Development Resources for Python/Pygame
- Git and GitHub Tutorial