CHESS GAME (AI)

P SAIKARTHIK

VU21CSEN0100398

Abstract

This report presents an overview of a chess game developed using Python and Pygame, featuring an AI opponent capable of playing against a human player. The project is designed to simulate a standard chess game with an interactive graphical user interface (GUI), providing options for either a two-player mode or playing against an AI. The game's core functionality is built on a custom chess engine that manages game rules, piece movements, and win conditions, while the AI leverages a basic algorithm to make strategic moves. The game aims to provide an engaging and educational experience for players, showcasing the implementation of AI in a traditional board game.

Introduction

The AI Chess Game is developed using Python, utilizing the Pygame library to create the graphical interface and manage user interactions. The game offers two modes: a human vs. human mode and a human vs. AI mode. The primary objective is to create a functional chess game that not only allows users to play against each other but also to experience playing against an AI opponent, which poses a challenge to the player's strategy and decision-making skills.

Game Structure and Implementation

1. Game Setup

Board Initialization: The chessboard is initialized using a custom board class, which creates and displays the 8x8 grid of the chessboard and places pieces in their initial positions.

Piece Classes: Individual chess pieces such as queen, rook, knight, and bishop are represented by their respective classes, encapsulating the movement logic for each piece.

Null Piece Handling: A nullpiece class is used to represent empty spaces on the board, simplifying the logic for move validation.

2. AI Implementation

The AI opponent is implemented using a basic decision-making algorithm, encapsulated in the AI class. The AI evaluates potential moves and selects the best possible action based on the current state of the board.

The move class is responsible for executing the AI's selected move, updating the board state accordingly.

```
import pygame
from board.chessboard import board
import math
from pieces.nullpiece import nullpiece
from pieces.queen import queen
from pieces.rook import rook
from pieces.knight import knight
from pieces.bishop import bishop
from player.AI import AI
import copy

from board.move import move



pygame.init()
gamedisplay= pygame.display.set_mode((800,800))
pygame.display.set_caption("pychess")
clock=pygame.time.Clock()
```

3. User Interface

The game interface is rendered using Pygame, where the board and pieces are displayed. The interface also includes text elements such as the game title, player mode selection, and endgame conditions (checkmate, stalemate).

The color scheme and font rendering are handled through Pygame's text rendering functionalities, creating a visually appealing display.

Game Mechanics

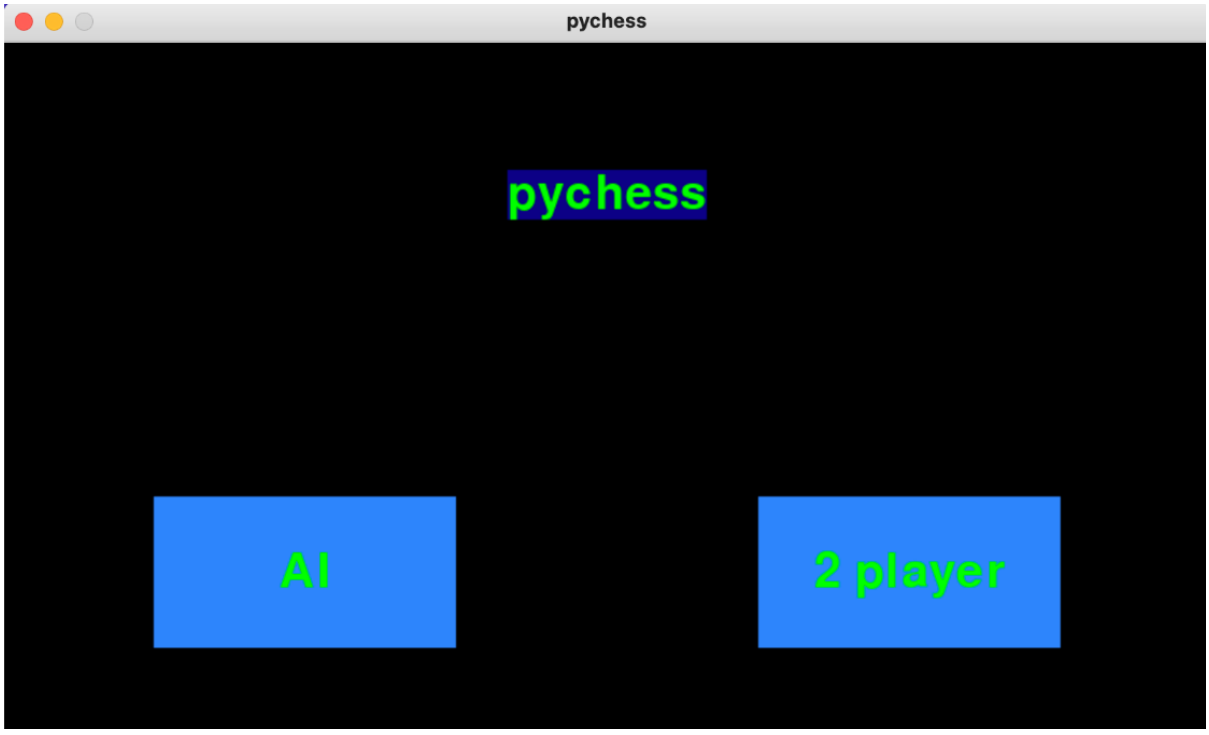
1. Player Modes

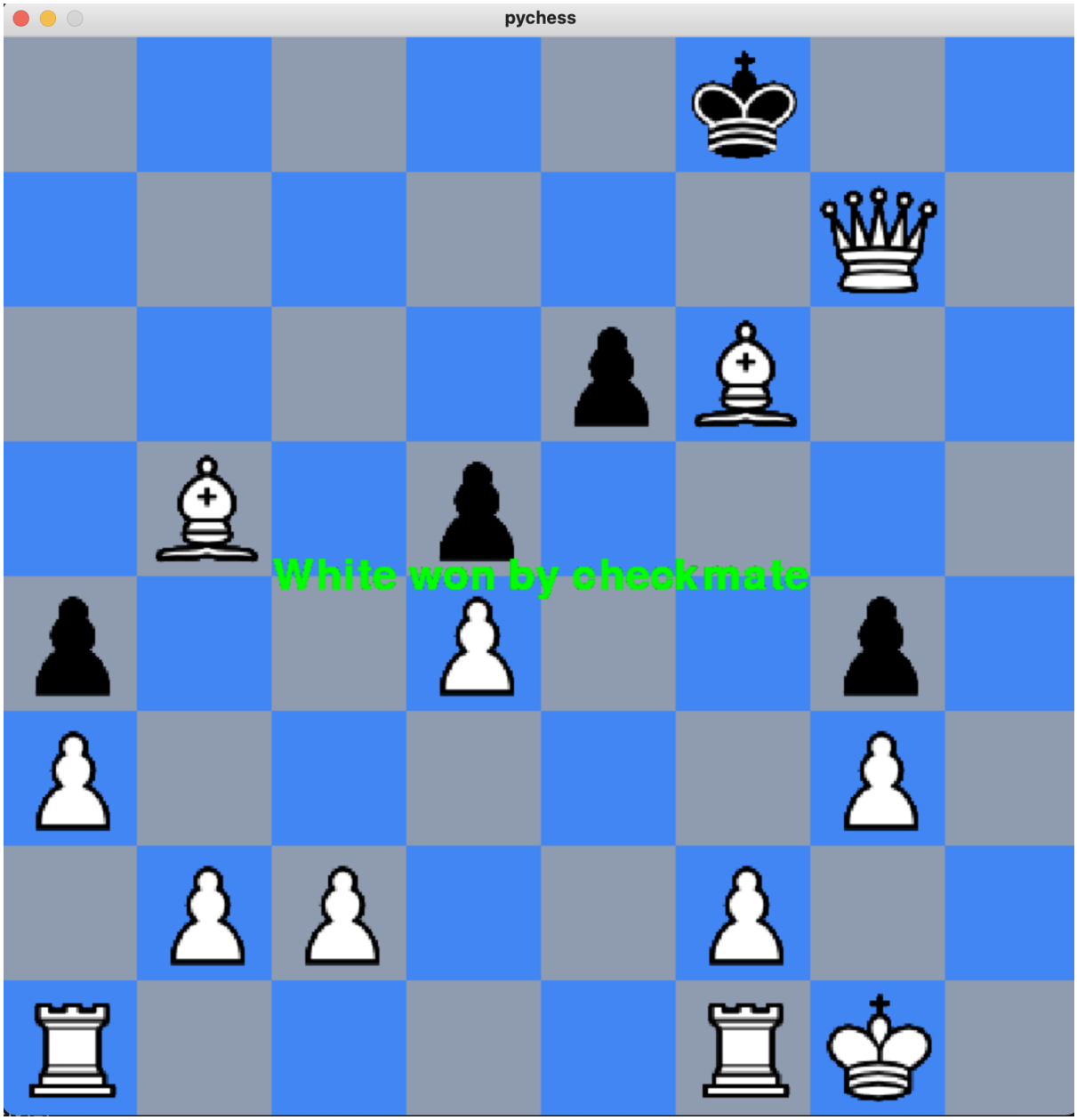
Two-Player Mode: Allows two human players to compete against each other on the same device. Players alternate turns, moving their pieces according to standard chess rules.

AI Mode: In this mode, a human player competes against the AI. The AI makes strategic moves after the human player's turn, challenging the player to outmaneuver the computer.

2. Endgame Conditions

The game detects checkmate, stalemate, and other win conditions, displaying appropriate messages to the player through the GUI. For example, "Black won by checkmate," "White won by checkmate," or "stalemate."





Conclusion

This AI Chess Game serves as an effective demonstration of integrating AI into traditional board games using Python. The game is designed with educational and entertainment purposes in mind, offering players an opportunity to engage with a classic game of chess while exploring the dynamics of AI gameplay. Future improvements could include enhancing the AI's decision-making capabilities, introducing different difficulty levels, and expanding the graphical interface to include more features such as move history and hints.

Future Work

Potential areas for enhancement include:

AI Improvement: Implementing more advanced AI algorithms, such as minimax with alpha-beta pruning or using machine learning techniques, to increase the AI's competitiveness.

Additional Features: Adding features like move history, game save/load options, and more detailed endgame analysis.

Multiplayer Online Mode: Expanding the game to allow online multiplayer matches.

References

Python Pygame Documentation

Chess Programming Resources

AI Algorithms in Game Development