The University of Hong Kong

2024 – 25

COMP4801

Final Year Project

Project Plan

Autonomous Chinese Checkers Playing Robot with Remote User Interface.

Name	**UID**
LEUNG Ho Ning	3035801453
SHIU Chun Nam Alex	3035800849

1. Introduction

1.1 Background

Recently, the focus of AI and robotics has shifted from just business uses to making entertainment and other interactive uses. One such area that has gained a lot of attention is the incorporation of AI in playing strategy games without any human intervention. Classic games like chess have already incorporated AI with successful models such as Stockfish systems. These AI-powered systems have proven that machines can be not just rivals but better than humans in complex strategic games and productive systems. On the other hand, we can see the increased use of robotics in our daily lives, such as some parcels moving arms and robotic customer services. These developments nowadays nurture the idea of creating a project that blends Artificial intelligence, robotics and gamification.

1.2 Motivation

The success of AI in mastering strategy games like chess has gained increasing popularity, on different digital platforms. Under the circumstances, we would like to explore how we could extend the idea further, from virtual to physical, from popular, attention-received games to classic traditional games. While games like chess have received significant attention from the AI and robotics community, Chinese Checkers—with its unique movement mechanics and multi-player dynamics—remains relatively underexplored. Therefore, the first aim of the project is to develop an autonomous system that can play Chinese Checkers without human intervention.

The COVID-19 pandemic has also contributed to the interest in creating technologies that allow engagement while adhering to social distancing protocols. Under social distancing, everyone has been craving for the old-fashioned, in-person ways of board games while in self-isolation with family or friends. This project addresses that by creating an interactive experience that combines AI and robotics, allowing users to play Chinese Checkers on their own, with the robotic arm executing moves autonomously.

1.3 Project Plan

Our project preliminarily would have 3 stages.

Firstly, we would procure a robotics arm with self-customization graips and software for our checkers game, it is expected that with a self-developed control program and parts, it could precisely move the pieces on the board on the checkerboard.

Secondly, we would design our application, which will recognise the checkerboard position and input to our AI decision model. It is expected that the app could recognise the player's new move and determine what is the best play to against it.

In the final stage, a Bluetooth connection, or any possible communication protocol, would be established between the app and the arm. It is expected that after the best move is decided, it will deliver the instructions to the robotics arm and perform the play.

2. Objective

The main goal of this project is to develop a Chinese Checkers playing machine with a robotic arm, capable of competing against a human player using AI. The specific objectives are as follows:

Initial Goal: Design and construct a Chinese Checkers playing machine, equipped with a robotic arm and an AI-driven control system that can autonomously play the game against a human opponent.

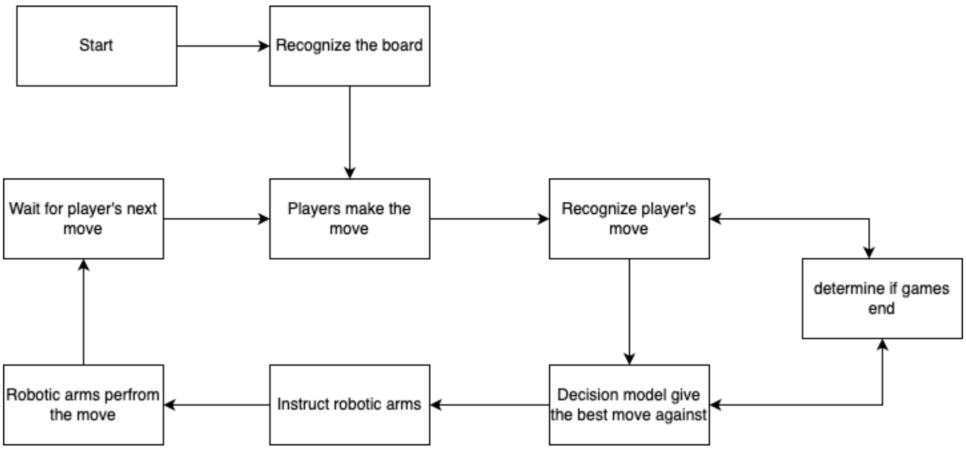
Intermediate Goal: Enhance the AI system to increase the number of participants in the game, starting from 2 and with a maximum of 6 players (including the machine). This brings a more challenging gaming experience for human players and simulates real-life family time.

Ultimate Goal: Develop the machine into a platform that offers personalized gameplay, where users can adjust the difficulty and receive real-time feedback on their moves, creating an engaging experience for players aiming to improve their skills.

3. Methodology

This part will discuss the implementation details of our project

3.1 Logic



The Chinese Checkers-playing robot operates in a structured loop that begins with setting up the board and continues through the game until a win condition is reached.
The workflow is as follows:

1. **Start**: The game begins with an initial setup, where the checkers are positioned in their default and correct starting positions on the Chinese Checkers board.
2. **Board Recognition**: The robot uses visual recognition (i.e. phone camera) to detect the positions of all pieces on the board. This provides the AI with the necessary data to understand the current state of the game.
3. **Player’s Move Detection**: Once the player makes a move, the system captures the updated board state. The robot recognizes the player's move by detecting any changes in piece positions.

4. **Decision Model Processing**: After recognizing the player's move, the AI decision model calculates the best possible counter-move.
5. **Instruct Robotic Arm**: Once the move is determined, the system instructs the robotic arm to perform the selected move.
6. **Robotic Arm Executes Move**: The robotic arm carries out the instructed move precisely, physically grabs and moves the piece on the board to its new position.
7. **Wait for Next Move**: After the robotic arm finishes the move, the system enters a waiting phase
8. **Game End Determination**: The system checks whether the game has reached a win condition whenever a move has been made, such as when a player successfully moves all their pieces to the opposite side. If the game is not over, the loop continues from step 3. If the game ends, the system terminates the current session.

3.2 Implementation

The implementation of the machine can be broken down into two parts: The Robotic Arm and the AI-powered Android Mobile App.

3.1 Robotic Arm

The arm is designed to reach all 121 squares on the checkboard. The Mobile App controls all the moves of the arm via Bluetooth connection with the mobile phone. We will procure the robotic arm from the internet and modify the hook to grab the pieces and hold them steadily. Attached in appendix is a sample of the arm that we would like to use.

3.2 Robotic Arm - Kinematic Logic

To effectively grab and hold pieces in the Chinese Checkers game, the robotic arm requires at least three degrees of freedom (x, y, z). However, for simplicity, we will focus on a manipulator with two degrees of freedom, specifically considering the rotational movements of two joints.

The position of the arm's end-effector (the part of the arm responsible for picking up and placing pieces) can be described using the angles of the two joints (θ_1 and θ_2). The horizontal (x_e) and vertical (y_e) positions of the end-effector are determined by the following equations:

$$x = l_1 \cdot \cos(\theta_1) + l_2 \cdot \cos(\theta_1 + \theta_2)$$

$$y = l_1 \cdot \sin(\theta_1) + l_2 \cdot \sin(\theta_1 + \theta_2)$$

3.3 Calibration and Control:

To ensure the arm performs accurate moves:

1. Joint Control: The angles of each joint are calculated based on the desired position of the end-effector. These angles are then translated into commands sent to the motors controlling the joints.
2. End-Effector Precision: The end-effector (the grip that picks up the checkers) must be aligned with the board positions to avoid misplacing the checkers.
3. Real-Time Adjustments: Cameras may be used to fine-tune the arm's position in real-time, ensuring that any slight misalignments are corrected before making a move.

3.4 Mobile App

In this project, we are going to write a mobile application on the Android platform, by Android Studio and Java. The app is responsible for recognizing the checkboard and pieces. After the recognition process, the AI programmed in the app will give the best move possible and instruct the Robotic Arm to execute the move via Bluetooth connection.

3.5 Minimax with Alpha-Beta Pruning

In this project, the decision-making process of the AI will rely on the **Minimax algorithm** combined with **Alpha-Beta Pruning**. As Chinese checkers is a perfect-information game, deterministic, it is well suited to this approach. The Minimax algorithm allows the AI to simulate both its own moves and the opponent's responses, ensuring that the AI makes the optimal decision at every turn.

Minimax Algorithm

The Minimax algorithm is designed to simulate all possible game states by exploring the game tree. The AI, acting as the maximizing player, attempts to choose moves that maximize its chances of winning while assuming the opponent (the minimizing player) is making optimal moves that minimize the AI's chances.

1. Game Tree Exploration: The Minimax algorithm builds a game tree where every AI move is calculated in advance and every possible feint that the opponent may take. This carries on until the terminal state (i.e. either win or lose) or a maximum search depth is attained, which is adjustable.
2. Score Evaluation: A terminal node of the game's search tree (or the desired search depth) triggered a score evaluation for that game state. For the particular game state, there is a score: positives are assigned to game states which are favourable outcomes (e.g. a winning move), and negatives are for game states that are not favourable. The AI then propagates these scores back through the game tree to determine the best move.

Alpha Beta Pruning

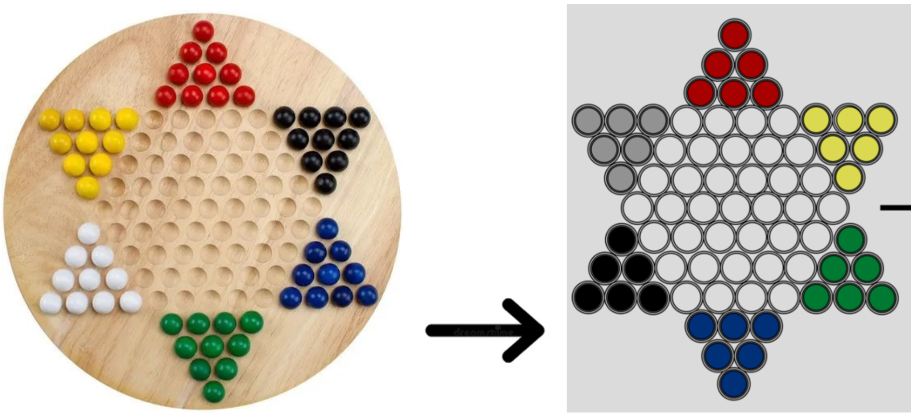
It is an optimization technique that reduces the number of game states the Minimax algorithm needs to evaluate, allowing the AI to make decisions faster, without any loss of accuracy. It works by eliminating branches in the game tree that cannot possibly influence the final decision.

- **Alpha (α)**: The best score that the maximizing player (the AI) can guarantee at any point in the game tree.
- **Beta (β)**: The best score that the minimizing player (the opponent) can guarantee at any point in the game tree.

During the Minimax search, if the algorithm finds a move that results in a score worse than the current alpha or beta values, it stops evaluating further moves along that branch (pruning). This reduces the computational cost of exploring all possible moves.

3.6 Open CV

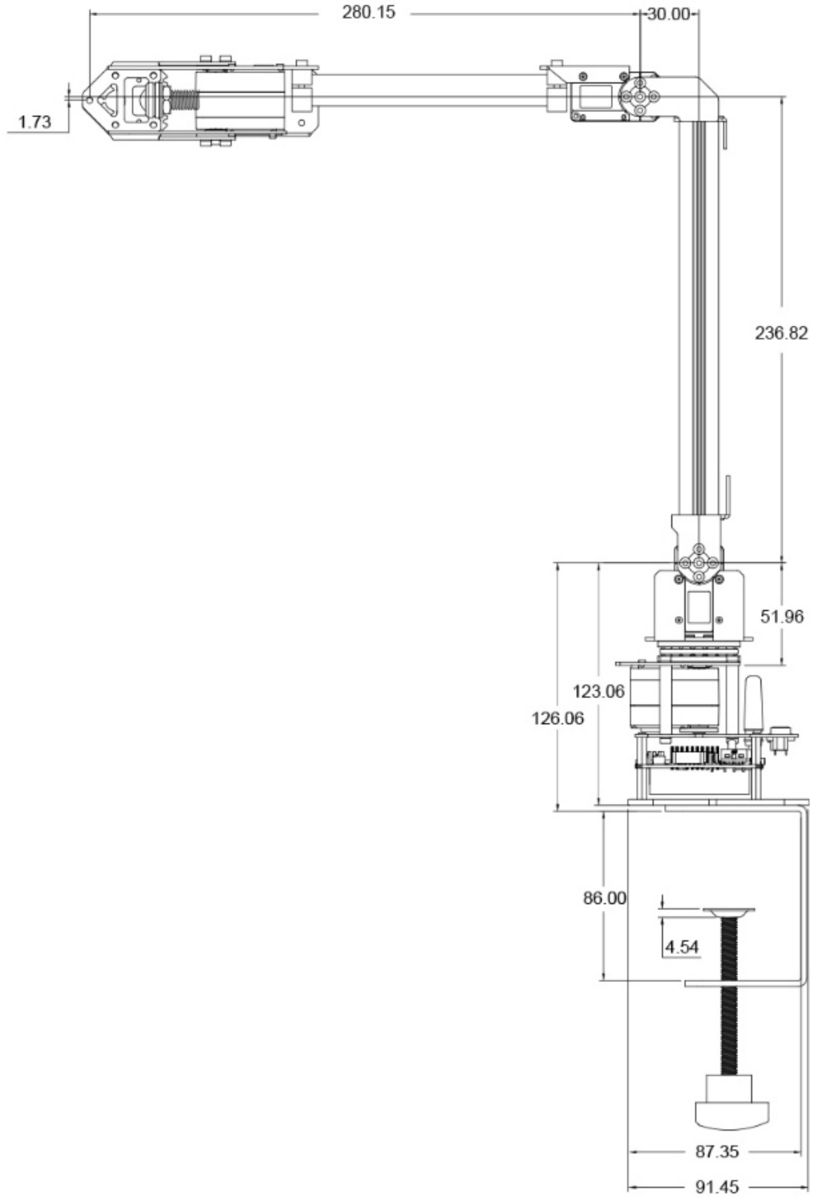
We will be making use of a library in Android Studio to help us with the checkboard and piece recognition. By implementing the library, the machine can scan the checkboard using the phone's camera and convert the image into position data that the machine can understand.

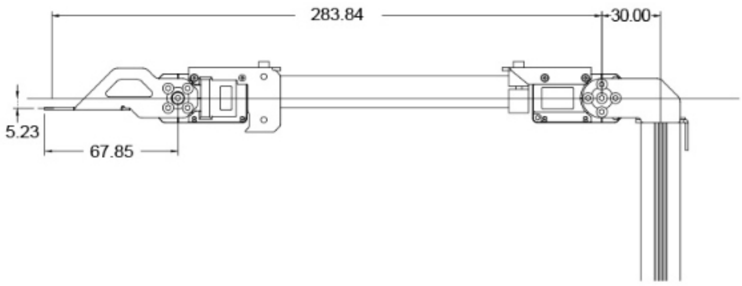


4. Project Schedule and Milestones

Date	Milestone
Sep 2024	Deliverables of Phase 1 • Detailed project plan • Project web page
Oct-Nov 2024	• Modify the robotic arm • Make a Demo version of mobile app
Dec 2024	Deliverables of Phase 2 • Further develop app so that it can control the robotic arm to execute the instructions • Implement checkboard and pieces recognition function into the app
Jan-Feb 2025	Deliverables of Phase 3 • Apply Minimax algorithm and Alpha Beta Pruning to the AI application
Apr 2025	Final presentation

Appendix





Unit: mm