Department of Computer Science
The University of Hong Kong
COMP4801 Final Year Project

Final Report

[MakerLab Project] Pick and Place Game App for 3D Printed Robotic Arm

Team member

Wan Tsun Wai, Alan (3035569017)

Wong Ka Ngai, Benny (3035568881)

Supervisors

Dr. T. W. Chim

Mr. David Lee

Date of Submission: 19/4/2022

Abstract

Artificial Intelligence is becoming one of the most crucial fields of study in the world. One of the applications is building a game AI that can defeat humans like AlphaGo in 2017. In this project, a fully automated Connect Four AI will be built to play with humans physically. To accomplish this target, a mobile application and a fine-tuned robotic arm will be delivered. The mobile application will capture and translate images to computer-readable data. Then data will be processed and the next optimal move will be sent to the robotic arm. The robotic arm will receive the command wirelessly and perform the action. When this project is complete, a fully automated Connect Four robot will be delivered and a ready-to-use library of Connect Four AI will be published.

The experiments conducted throughout the project including the retrieval of board state, designs of disc stacker and enhancements on Connect Four AI will also be discussed. After a year of development, the board detection algorithm is accurate and the Connect Four AI algorithm is optimal. It is reasonable to say that the objective of this project has been fulfilled.

Acknowledgements

Our team would like to express my special thanks to my supervisors Dr T. W. Chim and Mr David Lee for guiding us and comments on this project. Without their help, this project would not be able to make progress. Our team would also like to thank the Department of Computer Science and HKU CS MakerLab for the hardware support.

Table of contents

List of Figures

List of Tables

Abbreviations

3D	Three-Dimensional
AI	Artificial Intelligence
CI/CD	Continuous Integration Continuous Deployment
CS	Computer Science
GO	Weiqi
GUI	Graphical User Interface
HKU	The University of Hong Kong
HSV	Hue, Saturation, Value
IDE	Integrated Development Environment
IOS	iPhone OS
LED	Light Emitting Diodes
RGB	Red, Green, Blue
STEM	Science, Technology, Engineering, and Mathematics
ToF	Time of Flight
UI	User Interface
UX	User Experience

1 Introduction

The project aims to deliver an application that gives brand new Connect Four experience using the combination of AI and a robotic arm.

This chapter will give an introduction to this final year project report. First, the background of AI and a robotic arm will be introduced, followed by the game rule of Connect Four. Then, the motivation of the project and objectives will be discussed. After that, the report outline of the project will be mentioned in the last section.

1.1 Background

In May 2017, an AI player-AlphaGo defeated the best GO(Weiqi) human player-Ke Jie in the world [1]. In the past, GO was considered as one of the games that a computer could not beat human players attributed to its complexity. After the victory, people started to notice the potential of AI in strategic games. Despite the fact that AI gives fast and accurate results, human support is often needed to translate the calculation result into action.

To automate the process of translating decisions into physical movement, a robotic arm can be used. Robotic arm was first invented for industrial use, mainly for replacing human work. The advantage of a robotic arm is that it could perform precise tasks repeatedly with less error compared to humans. However, the drawbacks of old robotic arms were that they can only follow programmed routines and cannot react responsively to real-time changes.

With the aid of computer vision and AI, the problems of old robotic arms can be solved. To capture the real-time changes, a computer vision library called OpenCV can be put into use. After training, it can translate images into computer recognisable objects. Then, an AI algorithm will be able to compute results from real-time status and output complex decisions. Therefore, it is possible to build a responsive robotic arm that plays decision-making games against human players.

1.2 Motivation

Connect Four [2] is a two-player connection board game published in 1974. To start a game, a vertical board with six-row and seven-column, 21 red discs and 21 yellow discs will need to be prepared. Each player will take turns and put a disc into a vertical slot. If four or more

discs can form a horizontal, vertical or diagonal line, that player will win the game (see Fig.1). However, if all the 42 discs have been placed but the victory criteria are not satisfied, then it is a draw situation (see Fig.2).

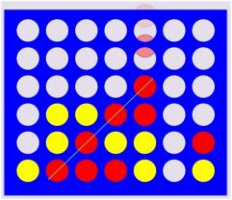

Fig.1 one of the winning situations, four red discs align diagonally

Fig.2 a draw condition, 42 discs have been placed but no player won the game

Connect Four is a very simple game but mastering it is not. Making a Connect Four AI that could precisely pick and place discs and perform optimal moves is also not easy. In this chapter, the motivations of this project will be explained.

Connect Four is a solved strategy game. The first player of the game has a winning strategy that is unbeatable. The algorithm was developed by James Dow Allen and Victor Allis [3] in 1988, which in theory an unbeatable AI can be built with their algorithm. Compared to unsolved games like chess, solved games are often less mentioned. Therefore, there is James Dow existing OpenCV or Pytorch libraries ready to be used. Object detection algorithm for Connect Four must be developed from scratch. It is hoped that this project could give some insights and discussions to the solved game community upon completion.

Capturing the checkerboard of Connect Four is a challenge in this project. For most of the board games that is placed horizontal on a table, the mobile phone is often kept at a fixed distance. However, in this project the distance between the board and the mobile phone may vary. A way to capture the board state in this situation must be found.

In addition, the column slot of the Connect Four board is very narrow, a very precise control of the robotic arm is required. Otherwise, the discs might not be able drop in the slot properly, and the game could not proceed, which adds difficulties to the project.

Lastly, the robot will be put in public places for people to play with when completed. It is hoped that the robot could serve educational and entertaining purposes. For instance, the project has an unbeatable AI, people might not be able to defeat the AI, but they can see their score and learn from the mistakes. Moreover, Alpha-Beta pruning algorithm can be introduced after people was defeated by the AI, even though they might not have computer science knowledge, it is hope that they could appreciate the power and beauty of the AI algorithm that contributes a lot in the world.

1.3 Objective and Deliverable

The purpose of this project is to build a mobile application for controlling a 3D-printed pick-and-place robotic arm that the AI can play Connect Four with human adversary.

To fulfil this objective, the project will deliver the following: A mobile application and a fine-tuned robotic arm. The mobile application will capture and translate images to computer-readable data. The data will then be calculated and sent the next optimal move to the robotic arm. After receiving the move, the robotic arm will perform the move accordingly. The application should consist of a user-friendly interface and an optimised structure. The fine-tuned robotic arm should be able to perform the auto-home function and receive commands through the Bluetooth module. If commands were received, the robotic arm should be able to pick and place the discs in the column slot fast and accurately. The whole playing process should be smooth and fast.

With the mobile application and robotic arm completed, the project will deliver a fully functional, automated Connect Four AI player. As well as a Connect Four OpenCV library that is ready to use in the future. Moreover, the project should be able to show the capabilities of AI algorithms in object detection and strategic decision, as well as giving insights in STEM education.

1.4 Contribution of members

During the development process, the development of board state retrieval including board detection, disc detection, the scoring system that involves setting up a cloud database, the design and building of the disc stacker is responsible by Wan Tsun Wai, Alan.

The UI and UX design, implementation of Connect Four AI Algorithm is responsible by Wong Ka Ngai, Benny.

It is worth mentioning that a lot of the tests and fine-tuning were carried out together. The application is the effort of both members.

1.5 Literature Review

Two Connect Four game applications namely “4 in a row King” and “4 in a row” were being used to study the performance of the existing applications (see Fig.3). “4 in a row King” has 10+ downloads [4], while “4 in a row” has 5M+ downloads [5]. They both support single player mode, online multiplayer mode and offline multiplayer mode. In the single player mode, different levels of difficulties were offered. After testing the moves of the AI in these two applications, it is found that their AI are not optimal. Maybe the algorithm in these applications is minimax algorithm with only a few layers, but it is an unknown as the source codes are not given.

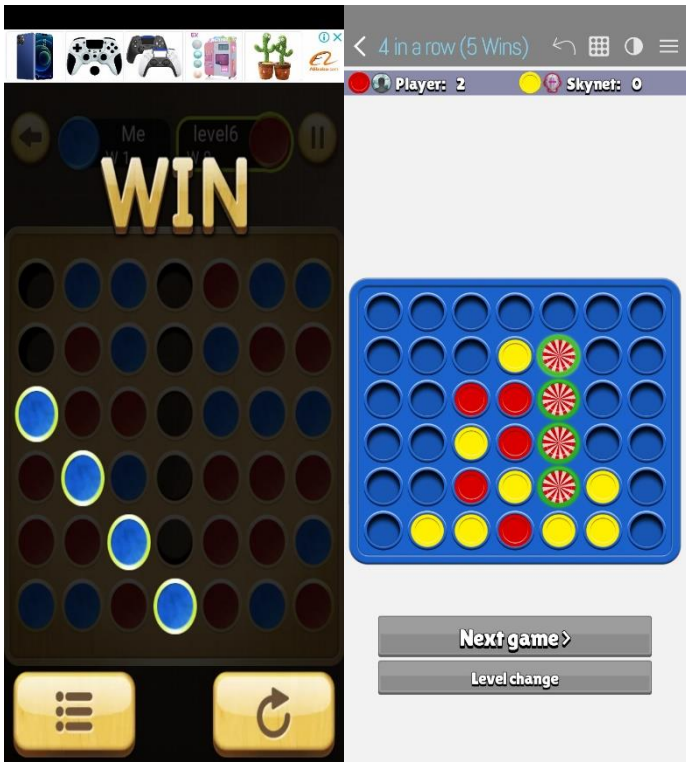

Fig.3 Two Connect Four applications that can be downloaded in Google Play Store

Another literature reviewed was the theory of Pascal Pon’s strategy. Through his algorithm, all possible states are put into consideration [6]. Hence, creating the perfect AI. A web based Connect Four game implemented his idea and the moves are all optimal [7]. A complete

guide to build the AI was also revealed in his blog. This gave insights to the project on building the optimal AI.

1.6 Report Outline

The remaining part of the progress report will be as follows. Chapter 2 is in relation to the methodology of the project, explaining the application design, software and hardware that will be used in this project. Chapter 3 will be the current status of the project, including the completed work. In addition, chapter 3 will include the experiments, results and discussions performed in the project. Chapter 4 will be limitation encountered in this project, and future works that can improve the application. Finally, a conclusion of the project will be given to wrap up the whole final year project.

2 Methodology

In the following chapter, the methodology of this project will be given. The project can be mainly divided into two major parts, the first part will be about the software implementation of the mobile application and the second part will be the hardware implementation. and it will mainly be divided into five sections.

The first section will be about the design of the project, including the overall UI design, the choices in software tools, for instance Android Studio, the OpenCV library, Alpha-Beta pruning algorithm and Firebase database, as well as hardware design softwares like SolidWorks and Shapr3D. Then, the overall workflow and the algorithm used in the the mobile application will be discussed. For example, the algorithm used in object detection, the alogorithm used in processing optimal moves and the score system will be introduced. The next section will be about the game modes we implemented in the application, including the arm controller for manually controlling the robotic arm and the Player VS AI mode which player can play with an AI opponent. The fourth section will be the modification made on the robotic arm to improve the overall controllability. The last section will be introducing the disc stacker we used in the project, and the rationale behind.

2.1 Design

In this section, the tools for developing the mobile application, the tools for constructing 3D model parts will be introduced. Moreover, the overall UI design, such as the colour scheme, design principle will also be introduced in this section.

2.1.1 Software

To finish the mobile application, a variety of software developing tools was used to facilitate the development.

Android Studio was selected for the integrated development environment in this project. Android Studio is the used to develop native android applications and it provides a lot of useful packages, libraries for developers. For instance, the Bluetooth packages, OpenCV packages and Firebase packages were utilised in this project. In comparison with other

mobile operating systems, Android occupied more than 70% of market shares. Therefore, it is believed that the application can cater most users.

Score system was implemented in this project. To make the data consistent across different machines, Firebase database was used to store data [8]. Firebase was developed by Google, providing a fast and easy cloud database options. The setting up and configuration is easier compared to traditional databases. It is believed that firebase will be a suitable tool for the project.

Collaboration is very important in this project. To facilitate better version control, Github along with Git extension were used. During the development of the application, a project management approach called CI/CD was used. Basically, branches were created for members to develop separately and do integration. After the codes were examined and tested in the testing environment, the application will be pushed to the main branch for further development or deployment.

By using Github, the change in codes and versions can easier been seen online. The merging of codes can also be made easier with Git extension, of which the versions in different branches will be visualised (see Fig.4). It helped a lot when collaboration was mostly conducted remotely due to the pandemics.

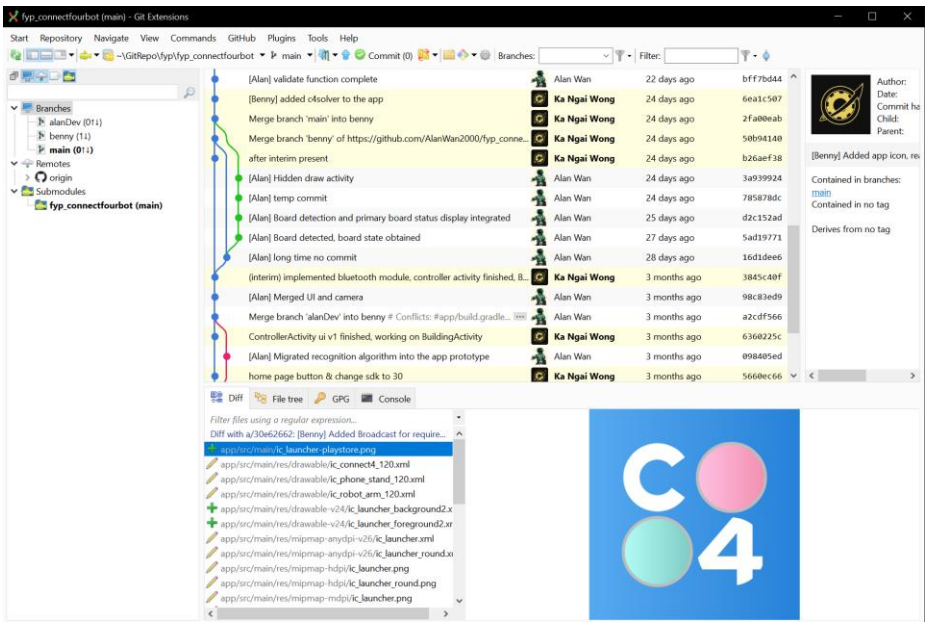

Fig.4 a screenshot of project version control using Git Extension

2.1.2 Hardware

To finish the whole project, hardware parts were printed to store the discs properly. In the construction of the disc stacker, various software was used.

3D printers were used to print out the first two versions of the disc stacker. Hence, 3D modelling software Shapr3D was selected for this project. Shapr3D is an intuitive CAD tool built to make 3D model and it is easy to use even for people don't have much experience for 3D modelling. Since the main objective of the project is not about 3D models, Shapr3D can help create prototype fast and efficiently.

After two trials on 3D printing, limitations on 3D printing were discovered. After receiving advice from Technical Manager Mr. C.S. Seto in the Tam Wing Fan Innovation Wing, laser cutting parts was put to one of the candidate methods. To cut parts in a laser cutting machine, sketches must be first be prepared. Hence, basics of SolidWorks and CorelDraw were learnt for this purpose.

2.1.3 UI and UX design

UI is the face of an application and UX affects the feel of user. This application should be a simple, responsive and task-oriented application. It is hope that users can understand the functions without any effort. The design principle of the application is to make a modern and minimalistic outlook. The UI design of this application will be introduced first, following by the UX design.

To begin with, the colour theme of the whole application follows suggestions from Material Design [9]. The primary colour of the application is teal, and the secondary colour is pink (see Fig.5). They are chosen because the colours of the Connect Four discs are also green and pink. It is easy to create a harmonious atmosphere for the application.

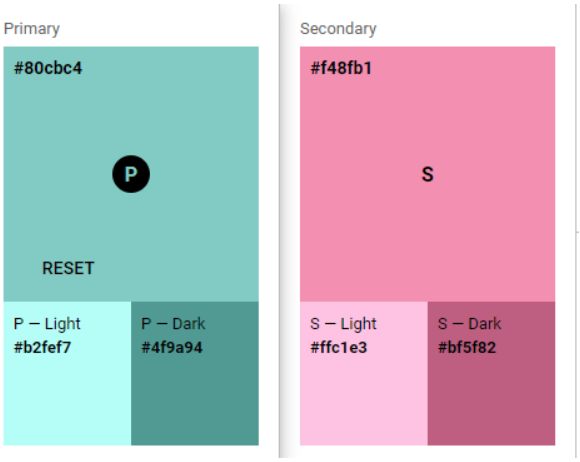

Fig.5 The color palette

Throughout the whole application, teal is used as the system bar colour and the dark variant is used as the colour of the top app bar, this creates an atmosphere that teal is a representation of the player. This design helps to create a contrast and draw user's attention into the application. The secondary colour of the application is pink, and it is used to represent the robot. In the select page (see Fig.6), teal is used in the player button and pink is used in the robot button. This ensure the player will not be confused when selecting the playing order.

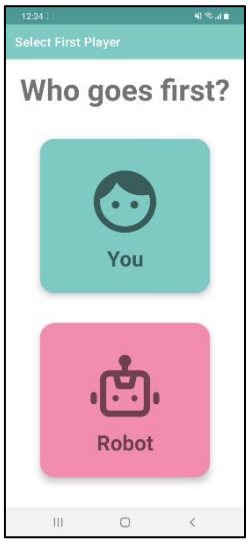

Fig.6 the select page, an example of color usage

Except teal and pink, blue is also used in the application for the tertiary colour as the Connect Four board is in blue. For the colour of background and text in the application, different levels of grey were used to keep the colour theme consistent throughout the application. To keep the aesthetic of the application in light tone, the base level of teal, pink and blue are at 200. Other darker variants will be in the range of 300 to 700.

To create a minimalistic feel, monotone colours and gradient colours are widely used in the application. In contrast to monotone colours, horizontal linear gradient colours create more impact to the user. It is utilised in two ways. The gradient of a colour from light to dark and gradient of primary colour to secondary colour.

In modern UI design, elements in an application often have a rounded corner. Psychological studies have showed that sharp corners appear to be harmful to users [10]. Therefore, all UI elements have rounded corners to create a comfy atmosphere. To make the elements more eye-catching, elevation is applied to the elements.

The font used in this application is Roboto font, which is commonly used font in iOS applications. It matches the overall minimalistic principle of the application very well. When choosing the colours of text, background colour is put into consideration. Different colours were selected based on the background, the figure (see Fig.7) below shown an example of different text colour usages.

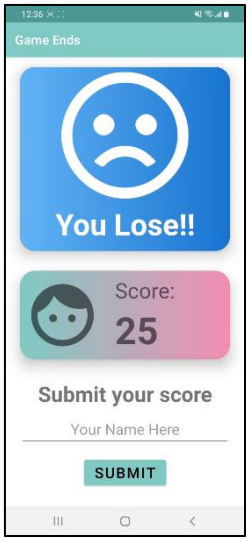

Fig.7 the game end page, an example of different text color usages

In this project, all graphics are in vector forms to preserve maximum clarity and cleanness. All the elements in the application are constructed with lines and circles to make them as minimalistic as possible. The only except in the application is the app icon which is created using Figma (see Fig.8).

Fig.8 the app icon

The user experience (UX) will be heavily affected by the user flow of the application (see Fig.9). An easy to use and learn user flow is very important if the UX wants to match usefulness, findability and usability. These criteria are proposed by Peter Morville in the UX honeycomb [11]. First, the home page can navigate user to any functions provided by the application. There will be instructions, toast messages or titles that give user enough information to understand the flow of the application. It is designed to let user be able to use the application smoothly even for the first time.

The application is also forgiving, if the user fails to identify in the first attempt, the refresh button allows the user to reposition the camera and perform the board detection process again. Since the application involves Bluetooth connection, there is chance that the Bluetooth module disconnect with the phone suddenly. It may cause catastrophic result if no measures were taken. In the application, a boardcast message will be sent to on-going activities and kill themselves when Bluetooth connection failed. It will return to home page and wait for user to connect to the Bluetooth module again. This preventive measure is very useful since all hardware components are uncertain factors. The details workflow of each game modes will be discussed in section 2.3.

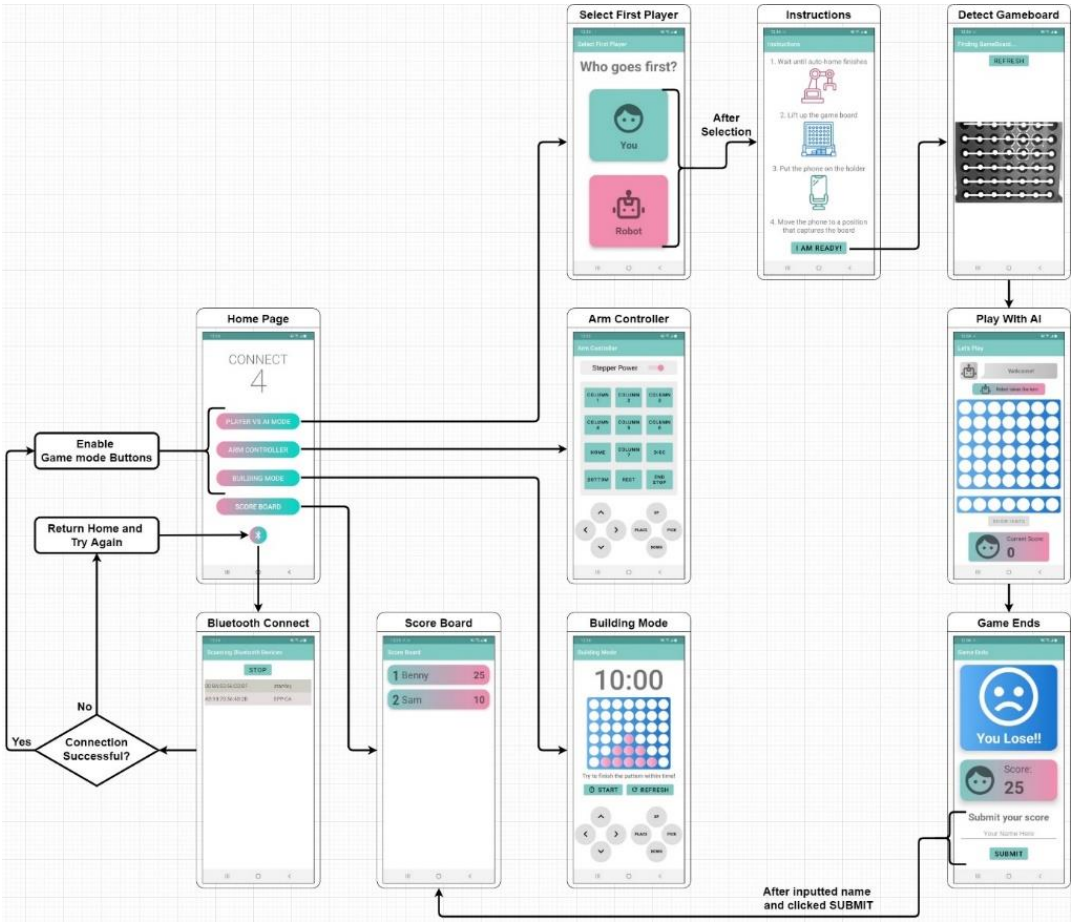

Fig.9 the UI workflow of the application

2.1.4 System Workflow

In the Connect Four application, different components written in different languages were put together. The figure (see Fig.10) below demonstrated the whole workflow of the application.

The application is developed in Android Studio using Java. In order to perform computer vision task, OpenCV library was imported in order to access and capture the board image. The board image will be stored in Mat format for processing and detection. After the detection process, the board state will be stored and sent to the next algorithm for computing the optimal moves.

The alpha-beta pruning algorithm is written in C++, and Android Studio cannot call C++ service directly. Therefore, Java Native Interface (JNI) was used to translate Java data types to C++ data types. In order to run a C++ program in Java based application, CMake is used to do the compilation. After transformation, the board state can be processed by the algorithm.

The algorithm make use of an external transposition table to reduce the consumption of resources and computation time. The results in C++ data types will then be translated into Java data types using JNI and return the optimal moves to the application.

Then, the optimal moves will be processed by the algorithm and translate it into a series of G-code commands. The commands will be sent to control the movements of the robotic arm.

This process will continue to repeat through the whole game, until one of the ending conditions is achieved.

After the game is set, the application will ask the player to enter his/her name. When the submit button is pressed, the score and player name will be sent to Firebase database using "POST" method. Then the player records will be retrieved from the Firebase database using "GET" method. All the player records will be displayed in the score board in descending order.

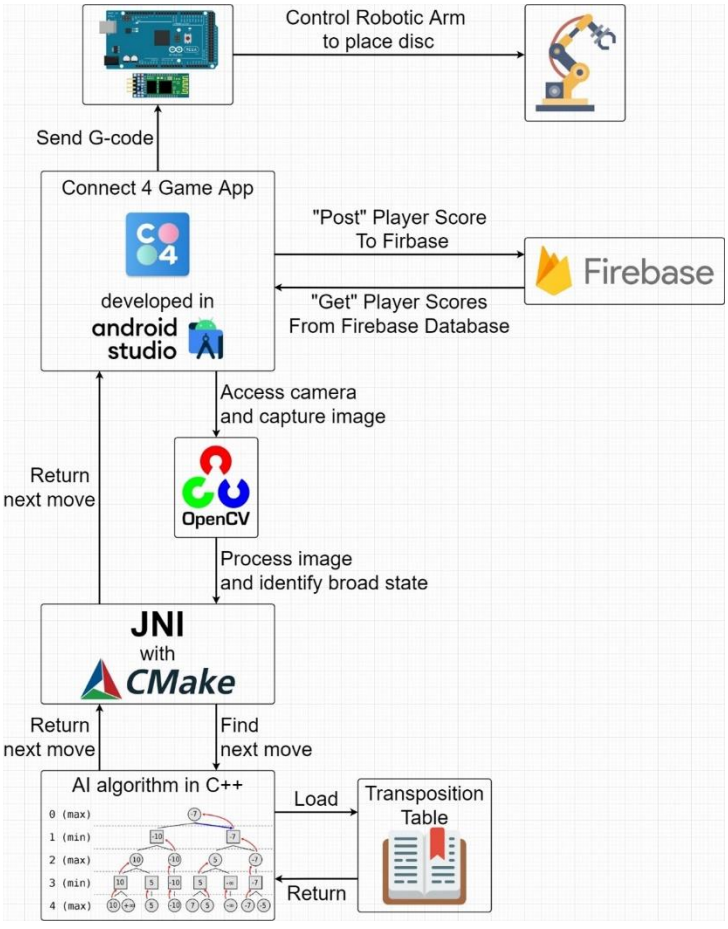

Fig.10 the system workflow of the application

2.2 Software – Mobile Application

In this section, the algorithm designs will be introduced. First, the overall workflow of the application including different approaches adapted in this project, the data passing through in each component will be mentioned. Then, the algorithm used to detect the board state will be explained in detail. After that, the Alpha-Beta Pruning Algorithm used to compute the optimal strategy will also be provided and explained. Fourthly, the score system implemented in this application for player's record will be illustrated.

2.2.1 Board Detection

In order to calculate the optimal move in the alpha-beta algorithm, the board situation must be obtained. To retrieve the changing board status efficiently, the process was divided into two parts, the first part was to get the circles coordinates in the frame, the second part was to identify the change of board state in the frame.

To solve the problem, the characteristics of the board and the constraints must be first clarified. The board is standing vertically and it consists of 42 round holes (see Fig.11). The distance and angle between the mobile phone and the board is not fixed. After knowing the problem details, a solution can be formulated. Since the distance is not fixed, the coordinates of the holes appearing on the camera frame varies each time. However, 7 columns by 6 rows in total of 42 round holes must be present to start a game. Therefore, the solution is to detect 42 round holes in a frame and examine if they aligned correctly.

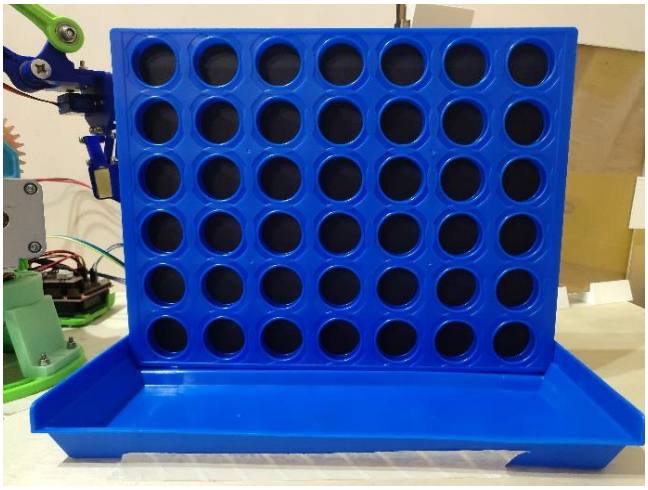

Fig.11 image of the actual game board

First, the image will be captured in Mat format supported by OpenCV. According to documentation [12], the class Mat represents an n-dimensional dense numerical single-channel or multi-channel array. Mat can be used to store grayscale or colour images in matrices form, which can be transformed and extract values easily.

The captured image will be first processed by trimming and blurring.

To begin with, the top, bottom, left side and right side of the image will be trimmed to reduce noise and interference. Since the application is designed for portrait mode only, it is reasonable to trim off unnecessary pixels for better performance.

Then, the image will be blurred by a 3x3 size kernel filter. Blurring, also known as smoothing, is a simple and commonly used technique in image pre-processing operation. Different projects might have different reasons for using blurring, but in this project the reason would be to reduce noise in the image. It is very important that the image is blurred because the speed of Hough Circle Transform algorithm that will be used later variates a lot with the noise in the image.

After processing the image, it will be passed to the HoughCircles() function provided by OpenCV [13]. It is a very powerful and efficient algorithm when it comes to circle detection. The rationale of the algorithm will be as follows.

Hough Circle Transform based on a simple circle equation:

$$r^2 = (x - h)^2 + (y - k)^2$$

In the equation, *r* represents the radius of a circle, *h* and *k* represent the x and y coordinates of the centre respectively, x and y are the variables in a rectangular coordinate system. With known r, h and k, a circle can be drawn on a rectangular coordinate plane.

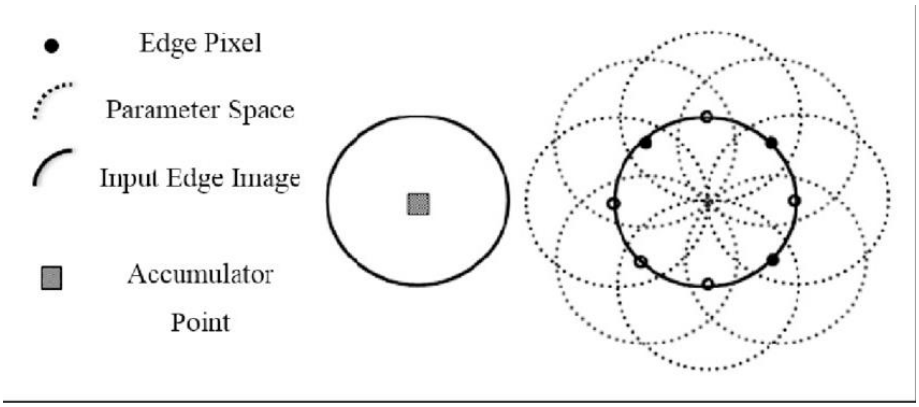

Fig.12 figure illustrating the circle detection

Let say there is only one circle in the image with known radius *r*. The circle's circumference was represented an array of x and y coordinates. For each pair of coordinates as centre, a circle will be plotted with the known *r*. After plotting all coordinates, a point with high frequency of overlapping will appear, which is called the *accumulator point* in the algorithm. From the figure above (see Fig.12), the accumulator point is coincided with the centre of the circle. Since the radius is the same for all circles, the intersection point is the same. Therefore, by identifying the point will highest overlapping frequency, it is possible to know the coordinates of centre, and the respective radius will be *r*. This is the basic principle of finding the circle centre with known radius *r* [14]. Yet, in this project, since the distance between the camera and the board is not fixed, the radii of circles is unknown.

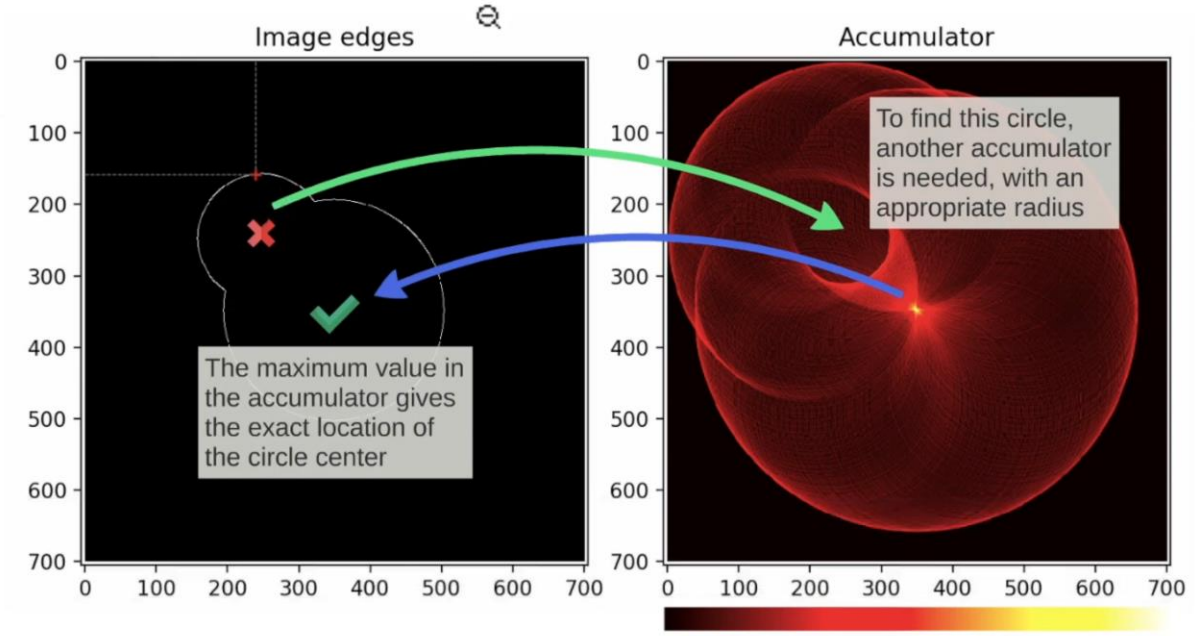

Fig.13 figure illustrating the circle detection process

Using this method, the radius must be specified and only the circles with that radius can be located (see Fig.13). To locate all circles, brute force can be used to solve the problem. Given a range of radii, for every radius in the range, it will go through the whole circle location process and pixels with high gradient will be selected as candidate pixels for voting. After voting, the voted pixel, which coincide with the true centre of the circle, will be stored in a centre array for output. This is the theory and workflow of Hough Circle Transform.

In the actual implementation, the algorithm consists of two main stages.

The first stage is edge detection and locate possible circles. The effectiveness of edge detection algorithm variate with the blurriness of the image. If the image is not blurred, there will be a lot of noise that produce unnecessary edge. If the image is blurred heavily, edge detection on the image would be too difficult. A slight blur can help increase the speed of circles location since the algorithm doesn't have to go through irrelevant edges. After this process, possible centres and radii will be stored in the next array.

The second stage would be finding the best radius for each centre. It is due to the insensitiveness of radius in circle location and it will be chosen by taking average.

The circle detection process will continue until the board is detected. Instead of detecting the rectangular board, the solution will be detecting 42 round circles. The candidate circles will be stored in a candidate array and a count will be applied to each circle. Whenever an incoming circle have similar centre and radius as any circle in the candidate array, the count will increase by 1 and the incoming circle will not be added as a new circle in the candidate array. If any circle in the candidate array has a count of more than 10, it will be considered as a stable circle. The algorithm will then try to link up stable circles to form a horizontal line. If the horizontal line consists of 7 circles, it will consider as a valid row (see Fig.14). When 6 valid rows appear, the board detection process ends, and the circle coordinates will be passed to the next activity for playing.

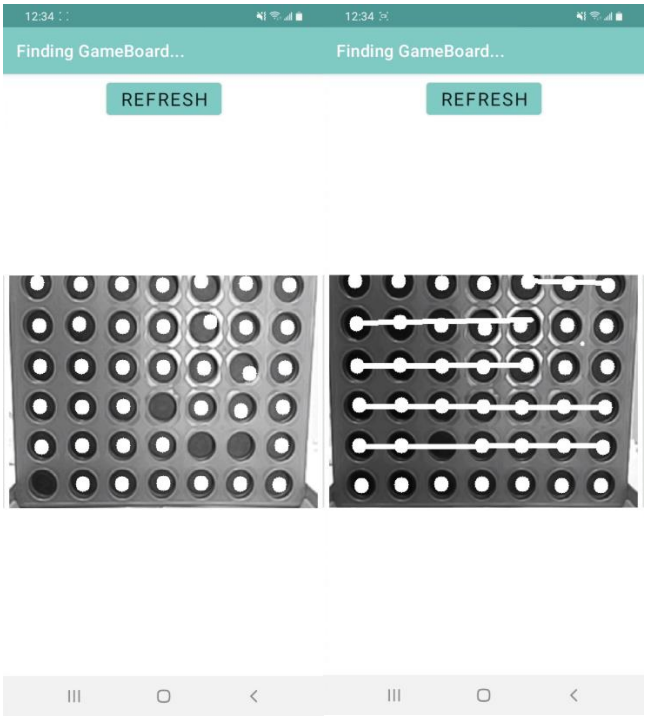

Fig.14 board detection process

After detecting the board, the remaining work for the algorithm will be detecting the colour change in the stored coordinates. After a few tests on it, HSV value will be used for identifying discs. The algorithm will convert the RGB image to HSV image, then for each pixel coordinates stored in the array, the colour of pixel will be read continuously. Hue, in colour theory can represent all colours in 360 degrees, which is very useful in colour detection since the lighting and environment might not be consistent. Through obtaining hue value from pixels and compare it to hue table, an accurate colour detection result can be achieved. This simple approach is very effective when it comes to colour detection.

2.2.2 Connect Four AI Algorithm

After the obtaining the current board state, optimal moves will be calculated from the state. In order to get the optimal moves from all possible outcomes, a minimax algorithm with Alpha-Beta pruning is used in this project.

Minimax algorithm is often used in building two player turn-based game AI, AlphaGO is one example. The two players in the game are called maximiser and minimiser. Maximiser tries to maximize the possible outcome and score is usually given to represent possible outcomes. In contrast, minimiser tries to minimise the opponent's score to make him/her lose the game [15]. When score of different states are correctly estimated, a minimax tree can be built. A minimax tree is similar to a decision tree, except layers in a minimax tree are arranged in maximiser layer and minimiser layer alternatively. The nodes in each layer are either maximised nodes or minimised nodes. The values of nodes in a layer are computed from the values of nodes in the lower layer. The algorithm will traverse the whole minimax tree to obtain the optimal choice (see Fig.15).

The gaining of a player equals to the losing of another player. Therefore, Connect Four is a zero-sum game and the algorithm can be further simplified to negamax algorithm. Only one recursive function is needed since the score in a position of a player is the negation of another player.

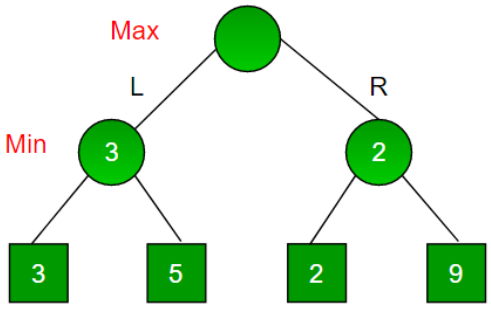

Fig.15 an example of minimax tree

Since the number of nodes are limited, it is possible to transverse the whole tree to find the optimal moves.Yet, the time taken might be very significant. To solve this problem, Alpha-Beta pruning is used to cut of branches that are impossible to become one of the optimal moves (see Fig.16) [16]. This way the computation time for the algorithm is significantly reduced.

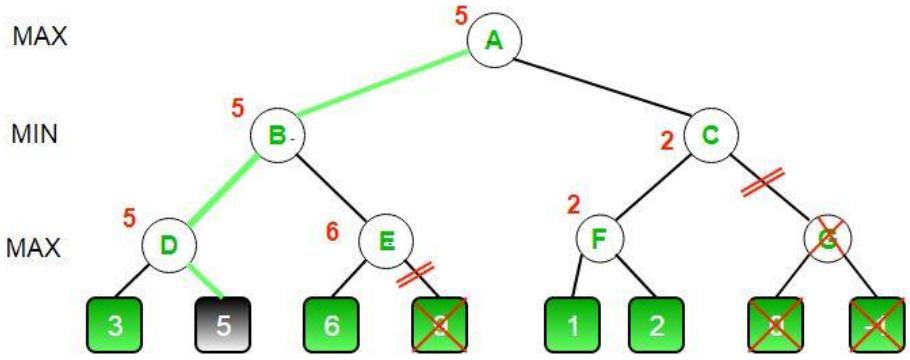

Fig.16 example of Alpha-Beta pruning on a minimax tree

When the game starts, there are in total 7 moves that a player can make. In fact, this holds true until one or more columns are filled up with discs. The whole board consists of 42 circles. That means the maximum moves in a game is exactly 42. From this information, the time complexity for Connect Four minimax algorithm is O(b^d), where b stands for the branching factor 7 as a node can have a maximum of 7 children for every move made, and d stands for the depth of tree 42 as the maximum number of moves in a game is 42. Luckily, since optimal moves are searched in prior, by using Alpha-Beta pruning the time complexity can be reduced to O($b^{d/2}$).

Although the tree size has been reduced, the problem is still too big to be solve in computation time. Studies on Connect Four shown that the number of reachable states from an empty board is 4,531,985,219,092. Meanwhile, there are 1,905,333,170,621 terminal

states and 2,626,652,048,471 non-terminal states in total [17]. In the table below (see Table.1), the number of possible states after each move is shown. To make the problem more manageable, optimisation will be carried out. The goal is to build a Connect Four AI that can output optimal strategy in a reasonable time.

Move	Possible States	Move	Possible States
0	1	22	22010823988
1	7	23	38263228189
2	49	24	60830813459
3	238	25	97266114959
4	1120	26	140728569039
5	4263	27	205289508055
6	16422	28	268057611944
7	54859	29	352626845666
8	184275	30	410378505447
9	558186	31	479206477733
10	1662623	32	488906447183
11	4568683	33	496636890702
12	12236101	34	433471730336
13	30929111	35	370947887723
14	75437595	36	266313901222
15	176541259	37	183615682381
16	394591391	38	104004465349
17	858218743	39	55156010773
18	1763883894	40	22695896495
19	3568259802	41	7811825938
20	6746155945	42	1459332899
21	12673345045	Total	4531985219092

Table.1 Table of moves and corresponding possible states

The optimisation process is performed following the tutorials by Pascal Pons [6]. It included several steps to enhance the performance of the algorithm. The steps are listed below.

1. Bitmaps are used to store the locations instead of an array to save running time.
2. The exploration of the best nodes is prioritised by using a move exploration order.
3. Calculated outcomes are saved in a transposition table to avoid repeated calculations.
4. A pre-generated book is imported in the transposition table to save running time even more.

The detailed enhancements procedures performed will be discussed in section 3.3.

2.2.3 Score System

Score system was implemented to enhance the learning experience of the player. Through providing scores to player, it is hoped that player will be able learn from mistakes and get better in playing Connect Four.

Before the human player makes his/her move, the state of the board will be read. From the board state, the next optimal moves will be able to obtain using Alpha-beta pruning algorithm. An array of possible optimal moves will be stored and to be used by the score system as well as other components in the application. When the human player makes his/her move, the change will be captured by the object detection algorithm. If the move of the human player matches one of the optimal moves proposed by the Alpha-beta pruning algorithm, 5 points will be given to the player. Whenever the human player saw an increase the score, that means he/her made an optimal move.

A hint button was also implemented in the game. When the human player pressed the “Show Hint” button, the possible optimal moves according to current board state will be shown. In this case, even if the human player made an optimal move according to the hints, points will not be given. It is to advocate player to think of the optimal strategy and try to defeat the Connect Four AI.

2.3 Software – Game Modes

In the application, there are different game modes for users to select. These game modes have their own functionalities and characteristics. The first game mode is the arm controller that can control the arm directly, mainly serving for testing purpose. Then, in the Player VS AI mode, the AI which play Connect Four with the human player, the human player will try to defeat the AI. The last mode is the building mode, pattern will be generated randomly and the player has to control the arm to reproduce the same pattern in the real board.

2.3.1 Before playing

All the three game modes require the control of robotic arm. Therefore, the mobile phone must be connected to the robotic arm first via Bluetooth first. The procedures are as follows.

When the player first enters the application, all game mode buttons are disabled as the robotic arm is not yet connected (see Fig.17).

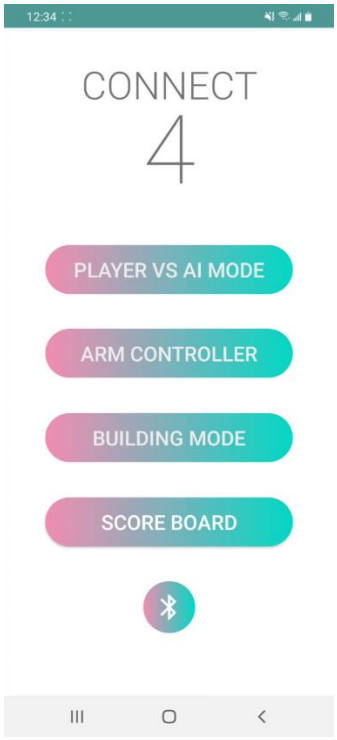

Fig.17 home page when Bluetooth is not connected

After the Bluetooth button is pressed, “Scanning Bluetooth Devices” activity will be entered. If the robotic arm had not been searched before, the list of devices will be empty (see Fig.18). Now press the scan button and wait until a list of devices appear. When the desire device appears, press that device to pair with it (see Fig.19).

Fig.18 empty Bluetooth device list

Fig.19 list of recognized Bluetooth devices

If the device was paired before, the process will be performed automatically. When the player enters the application, it will search for devices that had been paired before and pair with it automatically without the need of manual pairing (see Fig.20) like pairing normal Bluetooth devices.

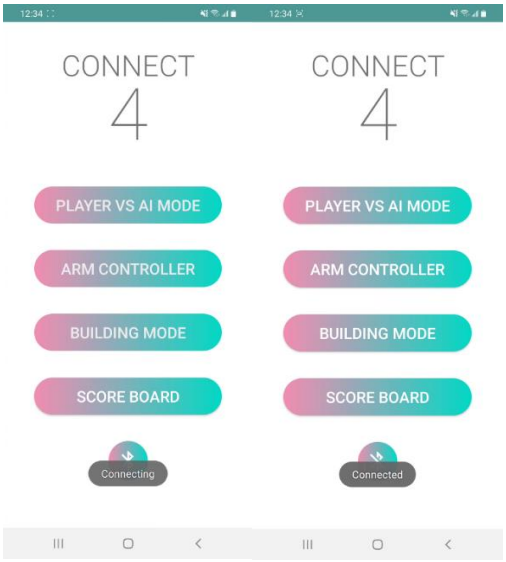

Fig.20 connecting to Bluetooth device that was paired before

After the pairing is done, the application will go back to the home page automatically. This time, all the game modes will be enabled (see Fig.21) and the application will be ready for playing.

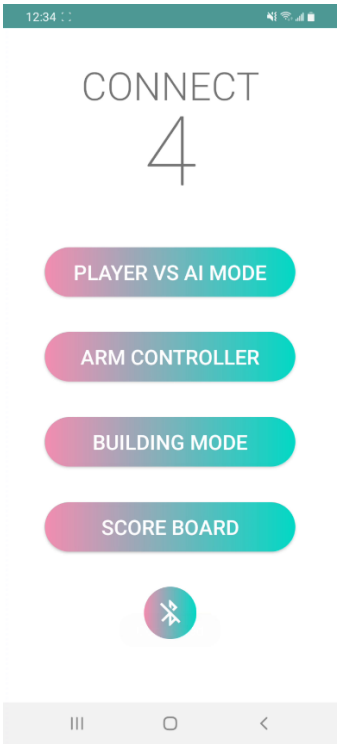

Fig.21 home page when Bluetooth is connected

2.3.2 Arm Controller

Arm Controller was the first mode developed in the application. The main function of the Arm Controller is to manually control the robotic arm to destined position. This mode was frequently used in the testing and enhancement stage.

Player can access this mode by pressing the “Arm Controller” button when the robotic arm is connected via Bluetooth. The figure below (see Fig.22) shows the GUI of the Arm Controller mode and it is divided into 2 parts.

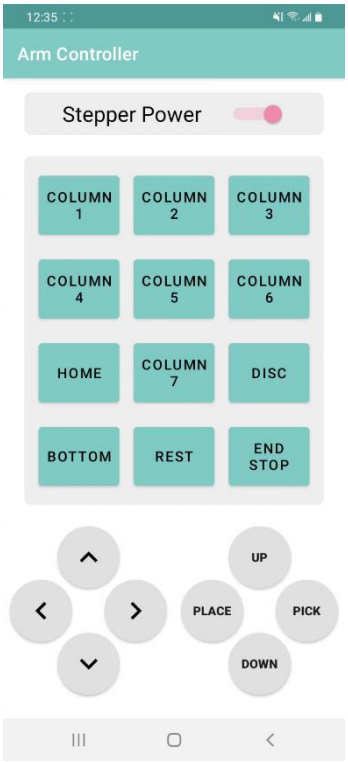

Fig.22 image of Arm Controller

The first half consists of one switch and 12 buttons. The switch controls the power of the stepper motors located in the robotic arm. The robotic arm can be switched on or off directly from the application.

There are 12 buttons arranged in a 3 x 4 manner which support predefined functions. The seven buttons labelled from “Column 1” to “Column 7” correspond to the seven columns of the Connect Four board columns from left to right. When the player press any of the seven columns, the robotic arm will move to the respective column and perform a drop action. Meanwhile, the “Home” button returns the robotic arm to its home position when the button is pressed. The “Disc” supports the auto pick-up disc function. When it is pressed, the robotic arm will pick up a disc from the disc stacker and return to the home position. The “Bottom” button will go the lowest position where the robotic arm can go, while the “Rest” and “End Stop” were used for testing and debugging only.

These 12 buttons serve as a calibration tool in the fine-tuning stage as well as a guideline for player to understand the whole working principle of the pick-and-place behaviour. The player can use it as a reference to further calibrate the disc stacker and game board position.

The second part of the Arm controller consists of a joystick-like game pad that allows player to control the movements of the robotic arm in 3-axes using the eight buttons. The function of

the directional button on the left is straight forward, it controls the movement of the robotic arm along the horizontal plane. The “Up” and “Down” button controls the movements of the arm along the vertical plane. The “Pick” button will close the grip and the “Place” button will open the grip. When the 6 buttons that involve the movement of the robotic arm are held, the robotic arm will continue to move along the respective direction. It is because G-code is sent repeatedly when the button is pressed. These are the main functions of the Arm Controller mode.

2.3.3 Player VS AI mode

Player VS AI mode is one of the most important modes in this application. Under this mode, the player will experience a Connect Four gameplay experience that one will not have been experienced before. In a human versus human game, the movement of opponent might be predictable and beatable. However, in this mode, human is less likely to win due to the unbeatable AI. Player could learn from the mistakes and get better in playing Connect Four, but most importantly appreciating the marvellous of computer algorithm.

To begin with, the player must first connect to the robotic arm via Bluetooth. The connection method was explained in section 2.3.1. Assume the robotic arm is now connected, the player can press the Player VS AI mode button to start the game.

The first page that the player will come across is the Select First Player page (see Fig.23). Since there is no limitation for which player to go first, both the player and the robot can be the first player. By press the “You” button or the “Robot” button, the playing order will be arranged.

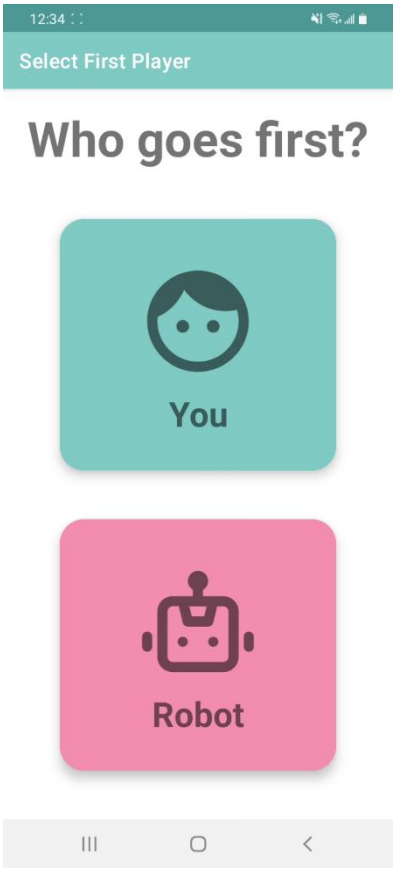

Fig.23 select first player page

After that, the robotic arm will do the auto-home position automatically. The player should follow the instructions on the screen to prepare the game (see Fig.24). When everything is ready, the player can press the “I am ready button!” to do the board recognition.

Fig.24 instruction page

The application will now show a rectangular frame, here is where the board will be identified. Put the board inside the frame to perform the board detection. To achieve the best result, try to fit the board into the frame as much as possible, but it is not a must, the algorithm will still do the work even if there are some distortions. (The details of board detection algorithm were discussed in section 2.2.1.) If the detection takes too long, the player can press the “Refresh” button to do the detection again. The detection process will be finished by the time when 6 rows of 7 dots were linked up (see Fig.25). Then the board will become an RGB board to indicate a successful detection. It will then proceed to the playground of the game and the game configuration is now completed.

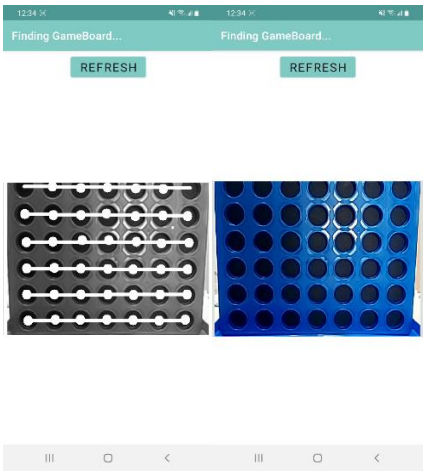

Fig.25 performing board detection and detection is successful

Now, a virtual board, two message dialogs, a hints button and a score box will appear on the screen (see Fig.26).

The virtual board will reflect the real time change in the board and show in on the virtual board for better experience.

The first message dialog displays the state estimation of the current board. According to different board state, it will display messages such as “Good Move” for an optimal move, “Bad Move” for a non-optimal move, “You win in n Move(s)!” if the player has advantage, “You lose in n Move(s)!” if the player is losing the game or “You can draw” if a draw game is possible according to the board state. These messages based on the calculation in the Alpha-Beta Pruning algorithm which was discussed in section 2.2.2.

The second message dialog is a simple reminder for the player to know who should take the next move to avoid confusion.

If the player cannot come up with a good move, he/her can press the “Show Hints” button. The bottom row will show the optimal moves according to current board state. The user can follow the suggestions and learn from it.

The last element in the activity is the score box. For every optimal that the player made, 5 marks will be given. The max step for each game is 21, therefore the maximum score of a game is 105 marks.

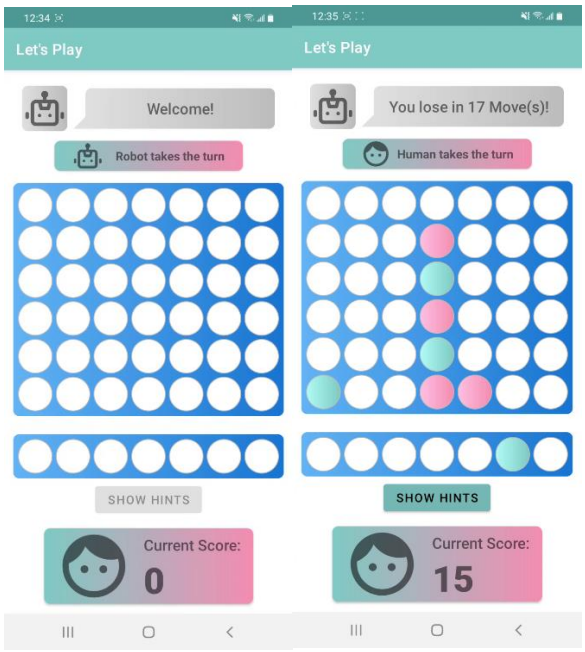

Fig.26 playing page at the begin and after a few moves

After the game is finished, the game result and the score will be shown. Then the player will be asked to enter his/her name and submit the record to the player database. When the “Submit” button is pressed, the application will automatically jump to the score board and show all the players record in descending order (see Fig.27). This marks the end of a complete gameplay.

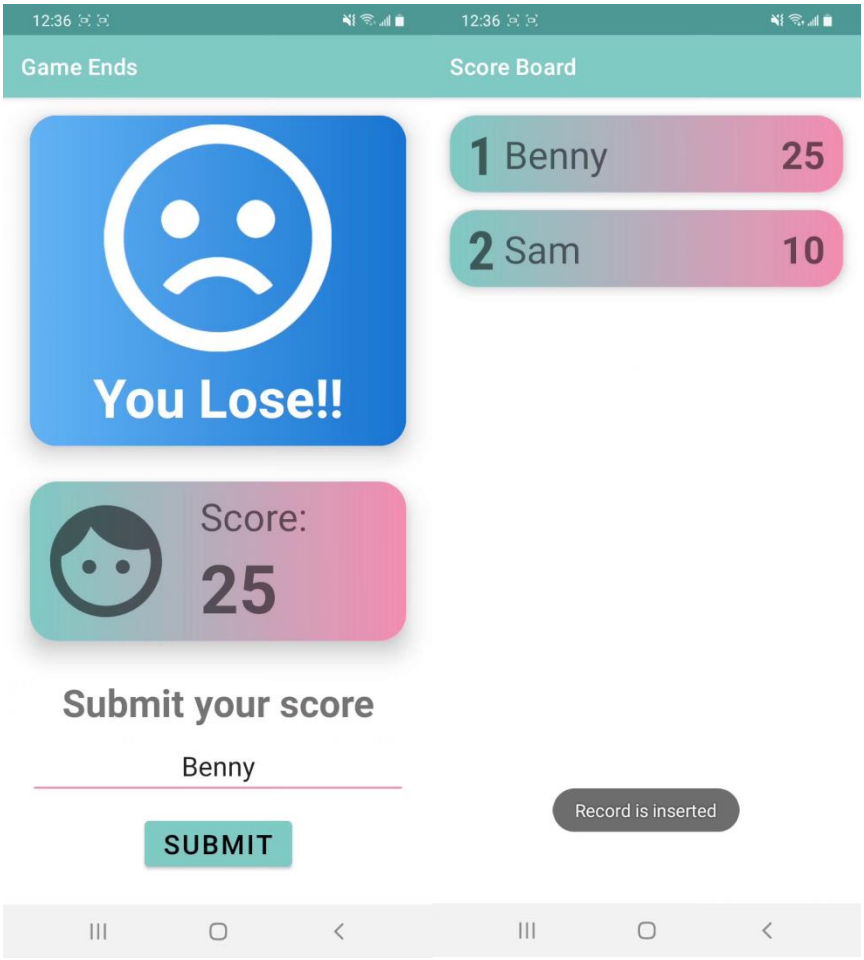

Fig.27 game end page and score board page

2.3.4 Building Mode

Building mode is another game mode in this application. In this mode, player's challenge will be to finish the given pattern before the given time runs out. The player must use the gamepad to pick and place the disc carefully and swiftly.

When the player enters the building mode, the robotic arm will perform auto home position automatically to setup the game. Then a pattern will be selected from a list of patterns randomly and display it on the screen. The player will need to follow the pattern and complete it in the real board.

The layout of the game mode is as follows (see Fig.28). There is a timer on the top to indicate the time remaining for the player. The pattern will be shown below the timer for the player to follow. After that, a start and a refresh button are placed under the pattern. The start button

will start the game and the timer will start counting down. Meanwhile, the refresh button is used to pick another pattern if the player has played or not satisfied with the current pattern. The control panel lies below the two buttons. This is the same control panel with the Arm controller. Eight buttons that allow player to move the robotic arm along 3 axes, pick and place disc are placed in the control panel respectively.

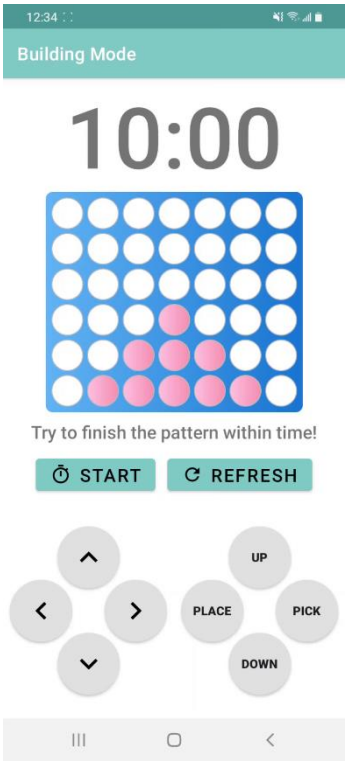

Fig.28 building mode page

When the player pressed start, the timer will start to count down (see Fig.29). Player needs to use the eight buttons in the control panel to pick and place disc continuously to finish the pattern.

If the player successfully finishes the pattern within the time limit, a toast message indicates the time used to finish the game will appear on the screen (see Fig.30). The page will reset itself.

If the player failed to finish the pattern within the time limit, a toast message “Time’s up!!” will appear on the screen. The whole page will reset automatically. The player can play with the same pattern again to get sophisticated in it.

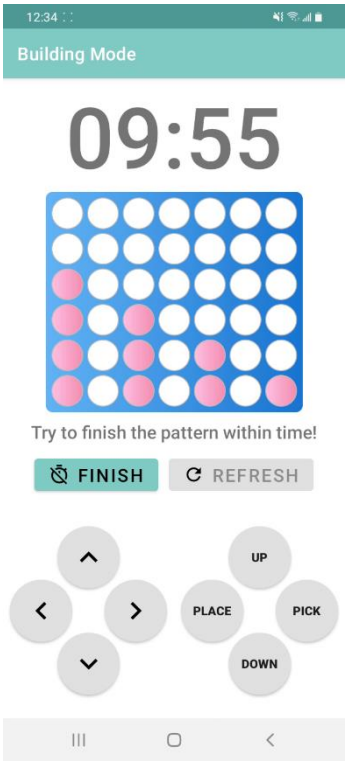

Fig.29 building mode page after countdown started

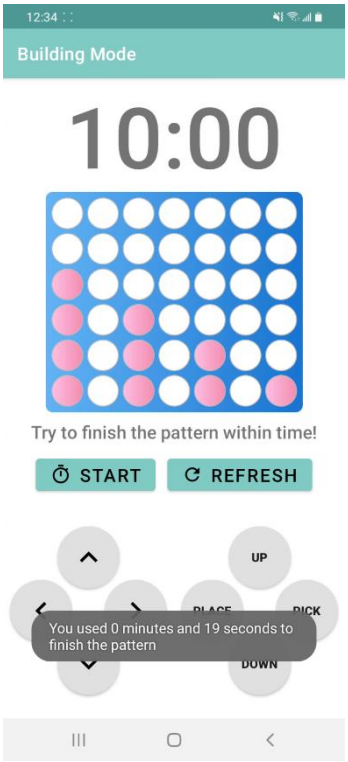

Fig.30 building mode page if the player can finish the pattern before the time limit

2.4 Hardware – Robotic Arm

The robotic arm is the bridge to turn digital signals into physical movements when playing Connect Four. In section 2.4.1, the background of the robotic arm including the specifications and the design will be given. In section 2.4.2, the improvements made to the robotic arm will be introduced.

2.4.1 Background of the robotic arm

First, the robotic arm is made and provided by HKU CS MakerLab. This robotic arm is a modified version of the original robotic arm designed by Florin Tobler [18]. The main difference between the original arm and the modified arm is the length of the arm. The MakerLab version has longer arm then the original one. The servo motors used in the arm are Tower pro SG90s servo (see Fig.31). Since there are 3 stepper motors controlling 3 axes, the robotic arm has in total 3 degrees of freedom (see Fig.32).

Fig.31 original arm (left) and modified arm (right)

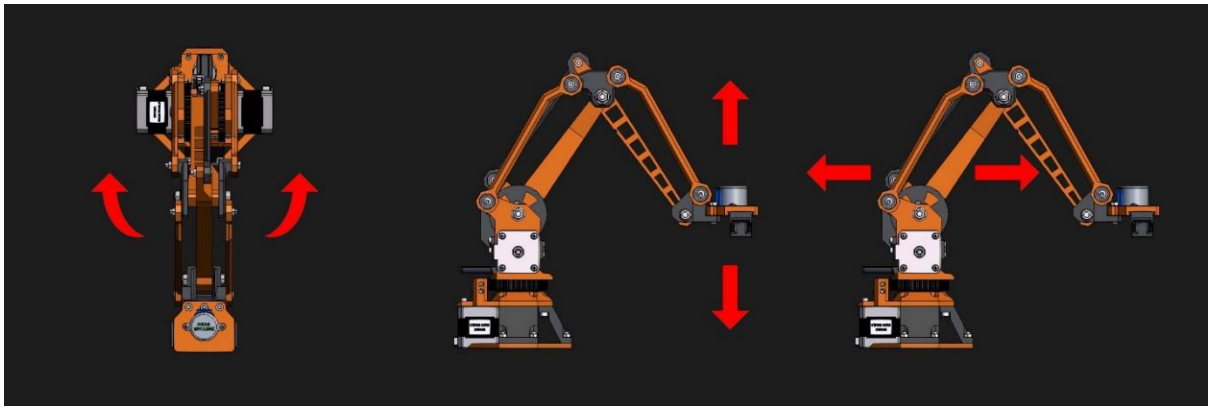

Fig.32 figure illustrating the degrees of freedom

The figure below (see Fig.33) shows the geometric structure and the kinematic algorithm of the robotic arm. It can be seen that the design is based on Florin Tobler's version but it still applies to the modified version.

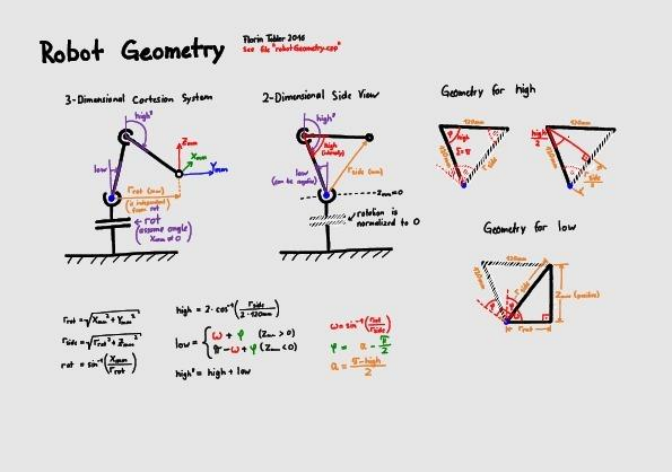

Fig.33 geometric structure and the kinematic algorithm of the robotic arm

Based on this design, an open-source Arduino firmware was developed by the robotic arm community. This firmware allows modifications to better fit in different versions of robotic arm. Meanwhile, MakerLab also developed a firmware to control the modified robotic arm.

The Arduino board used in this project is the Arduino MEGA 2560 with RAMP 1.4 expansion board installed. This board will receive commands from the Bluetooth module and instruct the stepper motors to move. The commands are in the form of G-code and G-code stands for "Geometric Code". Traditionally, the number of steps is given to the stepper motors to move the robotic arm to a specific location. However, G-code does it in an opposite way. The location is given to the program and the program will calculate the steps required to move to the destination. In a human sense, X, Y and Z coordinates are much easier to

understand then number of steps required. Not only it allows developers to predict the behaviour easily, but also debugging and fine-tuning.

Wireless control of the robotic arm is accomplished by installing a Bluetooth module on the Arduino board. The Bluetooth module used in this project is SPP-CA with baud rate 9600. After pairing is finished the Bluetooth module, the small LED on the Bluetooth module will keep on. Then, the application will be able to send G-code to the Arduino board via Bluetooth module and controls the robotic arm's movement.

2.4.2 Enhancements

Since the stepper motors used in this project do not support absolute encoder, The starting position varies each time when the stepper motors are turned on. It is needed for reference points to be found for home positioning. Otherwise, it is impossible to initiate the game. To solve this problem, 3 end-stop switches were installed on each axes to provide reference points in home positioning (see Fig.34).

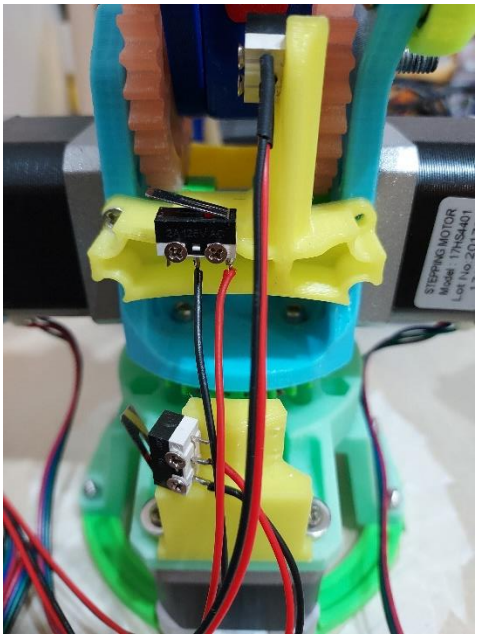

Fig.34 2 end-stop switches installed on the robotic arm

After installing 3 end-stop switches on the robotic arm, auto-home positioning can be performed using algorithm. From the figure below (see Fig.35), the default configuration of

the robotic arm indicated 3 variables that are important to the auto-home positioning function, which are INITIAL_X, INITIAL_Y and INITIAL_Z. They are composed by adding shank length and end effector offset. Since the modified arm has longer shank length than the original version, it must be changed to fit the real scenario. After experiment, the value of INITIAL_Z is 180, INITIAL_Y is 225 and INITIAL_X is 0.

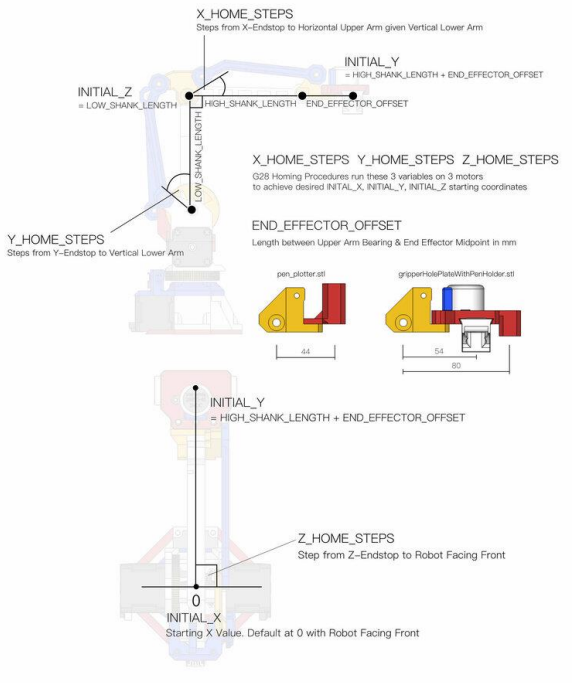

Fig.35 default configuration of the robotic arm

G28 is set for the home-positioning command. The steps for home-positioning are illustrated in the figure below (see Fig.36). Basically, they will first move towards the end stop switch respectively. If the end stop switch is hit, a signal will be sent and move the shank in opposite direction by their home step value. The home step value for different axes is different and they are manually tested for the best result. At the end, the upper and lower shank form a right angle when the home positioning is finished.

Fig.36 home-positioning steps

Except G28, G-code commands related to servo gripper are modified and listed below:

M3 T90/M5 T0: open servo gripper completely

M3 T0/M5 T90: close servo gripper completely

If the robotic arm moves beyond the limit, the stepper motors and the arm itself might be damaged. Therefore, limits were set to avoid the arm from moving beyond the operation range. The figure below (see Fig.37) shows the operation range of the robotic arm. R_MAX, R_MIN, Z_MAX and Z_MIN are calculated using the formula provided by the community firmware. Moreover, a function was implemented to check if the G-code to send is within the operation range as a safety measure.

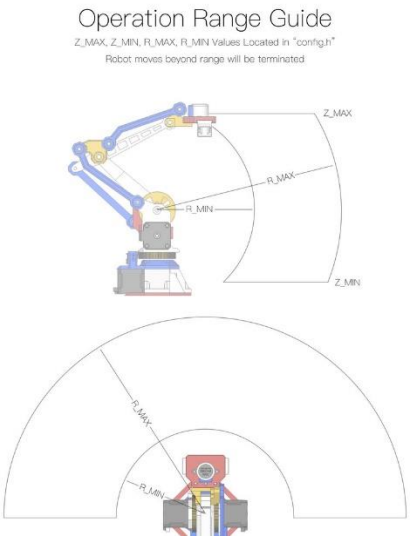

Fig.37 operation range of the robotic arm

2.5 Hardware – Disc Stacker

In a Connect Four game, discs must be insert vertically into the column. To pick up a disc using the robot arm, the disc will better be aligned vertically as well. To serve this purpose, a disc stacker was designed. It mainly serves two functions, the first function is to store the discs properly, the second function is to roll out disc automatically for the robotic arm to pick up. The design of the disc stacker is simple, it consists of three inclined tracks that are very smooth (see Fig.38). Due to gravitational force, the discs will roll down the track when the first disc is picked up immediately. The robotic arm will be able to pick up the next disc without delay. The detail specifications are as follows, the inclination of the tracks are 10 degrees, the length of the disc stacker is 31cm, the height of the disc stacker is 22.3cm and the width of the disc stacker is 1.5cm.

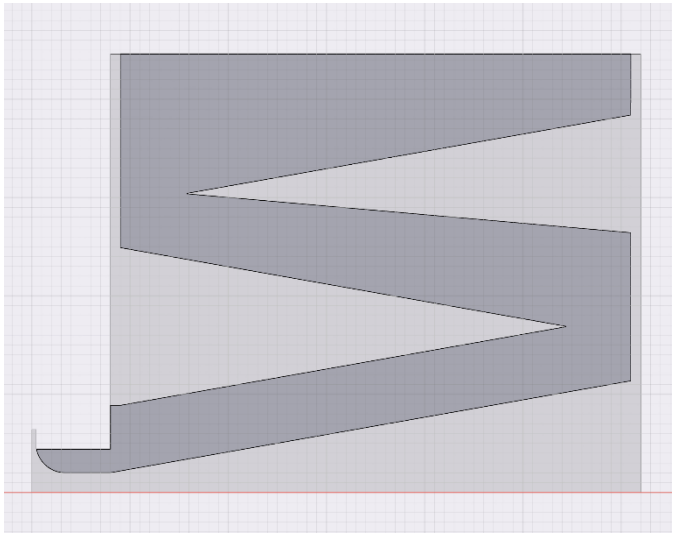

Fig.38 side view of the disc stacker

The black plates and the tracks were cut separately. As a result, a binding medium is required to bring them together. After consideration, a binding wedge was designed and printed using 3D printers. The shape of the wedge is a trapezium, the base of it is 3cm and the opening is 1.5cm. (see Fig.39) The wedge can bind the back plates and the trackers perfectly while providing base support and flexibility in repair. After a few versions of disc stacker, the performance of the current is the best among all. Hence, it was selected as the final version of disc stacker in this project.

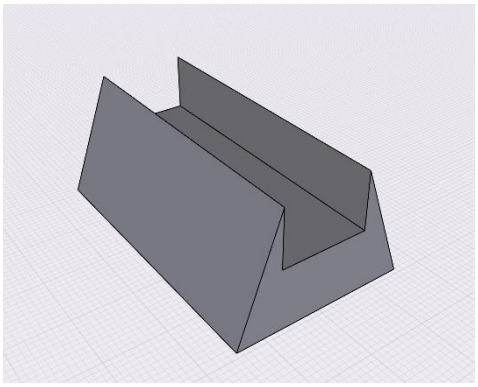

Fig.39 3D model of the binding wedge

3 Experiments, Results and Discussions

In this project, the testing of various tools and methods contributed a significant amount of time to achieve the best performance. Different versions of application were released to testing the usability, while some of them didn't perform as expected. As a result, more experiments were performed to understand the problem and to enhance the performance. For instance, the object detection method used in the interim presentation was completely different from the final version, it is due to the stability of algorithm. There are more experiments performed and their respective result will be further discussed in this section.

3.1 Experiment 1: Retrieving Board state

In this project, getting an accurate board state is very important. Without an accurate board state, subsequent processing and calculation would not be possible at most. Therefore, different object detection approaches were put into consideration in the early stage of the project. For example, neural networking training was considered initially. It is also one of the most widely used approaches in the field of object detection. There are many advantages in using neural network training approach, but the result was not very satisfactory. Therefore, other methods were being tested to find out the best solution. Except neural network training, lines detection is also very commonly used in the field of object detection. After various experiments. Circle Detection was used as the final approach.

3.1.1 Approach 1: Neural network training using TensorFlow Lite

The first approach adopted in this project was neural network training method using TensorFlow Lite. In particular, YOLOv5 was selected as the candidate model for neural network training. YOLOv5 is an open-source object detection model which is widely used in various project. For example, multi-objects detection, mask detection and face detection. It can be re-trained by using custom datasets for different purposes, which is the method adopted initially.

Since there is no existing Connect Four dataset available online, a dataset must be prepared. Dozens of photos of the Connect Four board were captured from different angle and environment. Then, the discs on the image were labelled manually using a software called "labellmg". The following figure illustrated the process of manual labelling (see Fig.40).

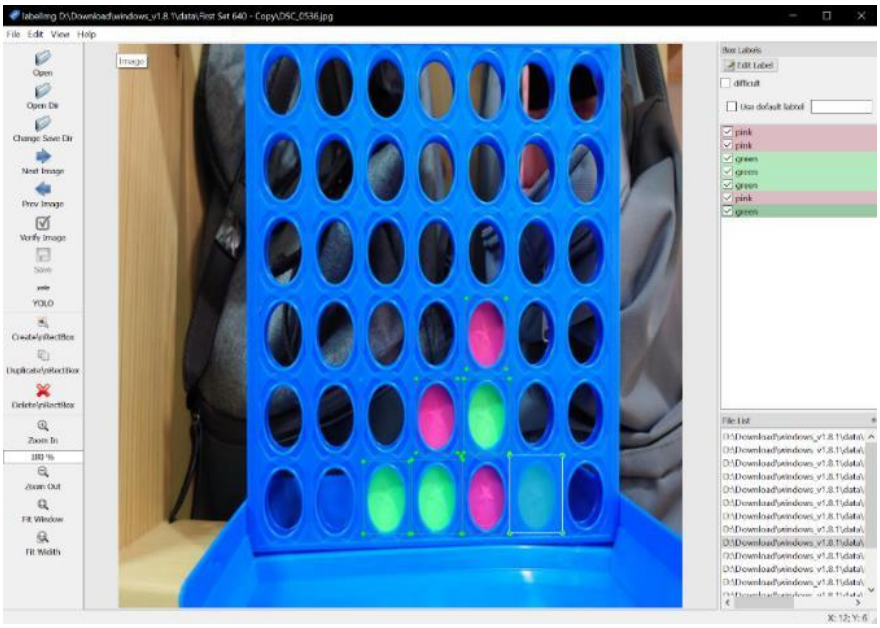

Fig.40 manual labelling process

After preparing the dataset, the YOLOv5s model, one of the YOLOv5 models, was used to perform supervised learning. Supervised learning indicates that the model will learn from the example input-output pairs given, which is also a common approach in the field of object detection.

When the training has completed, tests were performed to examine the performance of the model. From the figure below (see Fig.41), the framing of disc and their respective label were

correct. However, one pink disc was detected at the bottom of the frame, and it was not desirable.

Fig.41 real-time object detection using YoloV5

Another detection result (see Fig.42) could show that even though most of the discs has been recognised. 1 to 2 discs were still missed by the model.

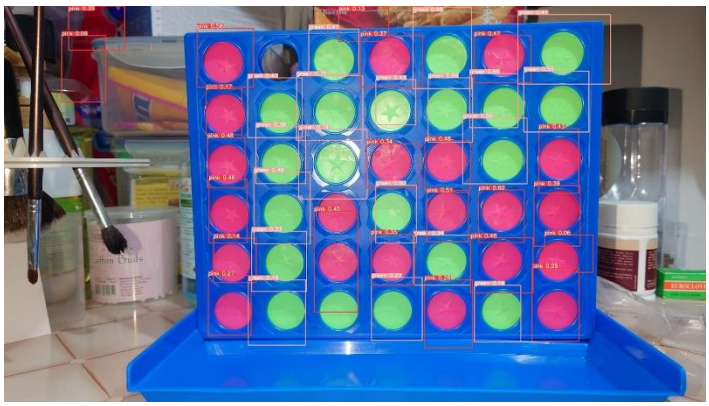

Fig.42 object detection using YoloV5

Various attempts were made to improve the model, including increase the size of the dataset with pictures taken from different environments, parameter tuning and increase training epochs. However, the result did not improve much.

The results were inspected, and conclusion was made. Since the model identify disc using its pattern and colour, it makes sense that under different lightings, the colour varies and the judgement of the model will be affected. It is an unsolvable problem since it is impossible to fix the environment variable. After careful consideration, this approach will not be used as the object detection approach in the project.

3.1.2 Approach 3: Circle Detection using OpenCV

From the first experiment, constraints were learnt. A method that could neglect the change in light condition must be found. After research, it is found that Hough Line detection is very common when it comes to chess board detection. The board will be first detected, and the algorithm will response according to board change.

From this perspective, the board itself should be detected and indexed before the game start. After looking through the libraries of OpenCV, an algorithm that is used to detect circles was found, it is called Hough Circle Transform.

Another version of board detection algorithm based on Hough Circle Transform was built to test the result. In the first trial, the algorithm already detected most of the circles successfully and quickly. However, if image with a complicated background was put into the algorithm, the computation time might be affected, a certain degree of lagging was also observed.

3.1.3 Result of Experiment: Circle Detection using OpenCV with fine-tuned parameters

Although the first trial of using Hough Circle Transform algorithm did not gave a perfect result, it is believed that the direction is correct.

To inspect the reason, debug logs were printed to gain some insights of the problem. It is found that over a hundred circles were located by the algorithm each time and the rapid update in each frame caused the lag. Moreover, the image was too sharp and too detailed, when passing the contour construction process, noisy contour was constructed. Hence it increases the time for the circle detection process (see section 2.2.1 for detailed explanation

of Hough Circle Transform). Therefore, unnecessary pixels were removed, and the image was blurred to reduce the noise.

After this process, the algorithm performed better, but sometimes large circles that did not fit into the board was found and many circles were detected very closely. The former problem was related to the range of circles radius set. If the range is too large, not only undesirable big circles will appear, but the processing time will also be increased. To solve this problem, range of circles that might appear on the image was tested and set. Meanwhile, the latter problem was due to the "distance between circles" parameter. This parameter controls the output of closely packed circles. If this parameter was set to 1, two centres that are 1 pixel apart will also be output, which in this project it is not possible for two centres to separate by 1 pixel only. After the parameter was set to 50, this problem was solved.

The algorithm then went through tests in different environments and it performed very well. Therefore, it was selected as the final solution of object detection.

3.2 Experiment 2: Disc Detection

A two-stage object detection approach is used in this project. The first stage is the board detection, the second stage is the disc detection. Identifying the board change fast and accurately is very important to the whole game experience. Therefore, experiments were conducted to find the best solution, which is using HSV colour space for disc detection.

3.2.1 Approach 1: RGB colour space

At first, different approaches were under consideration. For example, disc detection using YoloV5 was considered as a candidate method. However due to the low accuracy in the previous experiments of object recognition, it is believed that disc detection using YoloV5 is not an ideal solution.

After doing several research, many coloured object detection algorithms were using RBG colour space. In RGB colour space (see Fig.43), all colours can be mixed using Red, Blue and Green. The usual number of Red, Blue and Green range from 0 to 255. If Red has value of 0,

that means red is now black. If Red has value of 255, that means red is now white. The lightness of red increases with the value. This principle also applies to Blue and Green.

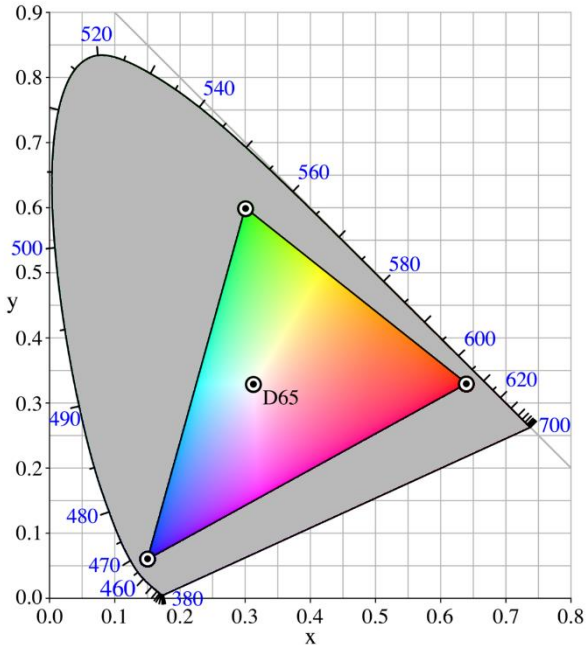

Fig.43 RGB color space diagram

Many algorithms set a range of RGB value for colour detection. The advantage of using this method is that it is very straight forward for colour estimation. Yet it also has some disadvantages. For example, when light intensity changed, the result might not be accurate. If the range is set to wide, the accuracy decreases even more.

Since the environment of playing is not fixed. Disc detection using RGB colour space might not be a good idea.

3.2.2 Approach 2: HSV colour space

It is hoped that another method of disc detection which is independent to environmental change can be found. Luckily, a video about image colour grading gave this problem some insight.

In image colour grading, HSV colour space was used instead of RGB colour space. HSV (see Fig.44) stands for Hue, Saturation and Value. The colour was also represented by 3 numbers, but the saturation and brightness were isolated. All colours can be represented from 0 degree

to 360 degree. If the image is converted from RGB colour space to HSV colour space, the problem of changing environment should be able to be solved.

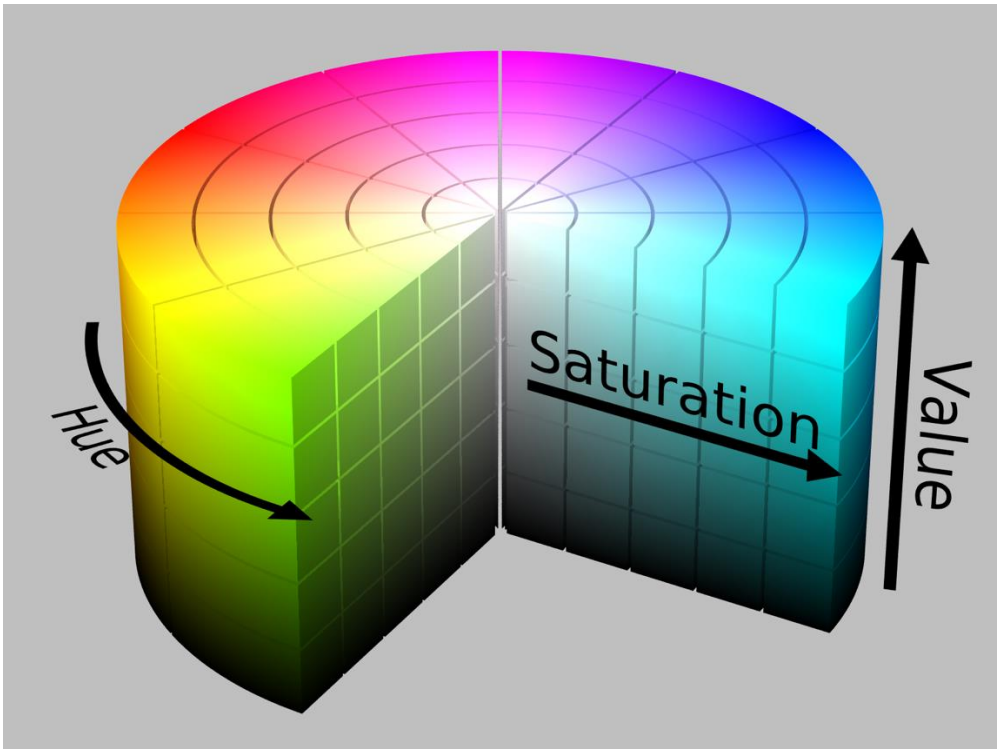

Fig.44 HSV color space diagram

Tests were conducted to know if the hypothesis was right. A range of green and pink was set. In various testing under different environments, the results were very satisfactory. No matter how the light intensity changed, the output was very consistent.

3.2.3 Result of Experiment: Disc detection using HSV colour space

From the test result, disc detection using HSV colour space was the most stable and accuracy among different approaches. The reason might be the isolation in saturation and brightness factor. Since the change in environment usually involves light changes, the colour itself is consistent. Disc detection using HSV colour space can yield a very satisfying result and it is the final approach adopted in the project.

3.3 Experiment 3: Connect Four AI

In many strategic games, depth limit and machine learning are used to build a sophisticated AI. Even brute force is one of the ways can be considered. However, thanks to Pascal Pons's theorem, an optimal Connect Four AI can be built [6]. The methods of enhancements will be given in this section.

3.3.1 Approach 1: Minimax with Alpha-Beta pruning

Minimax algorithm is being used in this project. To make use of minimax, an effective estimation of game state must be designed.

In the proposed algorithm, the estimation of a game state is represented by a score, which can be positive, negative or null. The score is calculated as follows.

If the current player is winning in a specific board state (see Fig.45), the score is

22 – the total number of discs needed to win.

If the current player is losing in a specific board state, the score is

the total number of discs needed to win for the opponent – 22.

If the game will be a draw game, the score is *null*.

This estimation not only gives score to board states for minimax algorithm, but also predicting the moves needed to end the game.

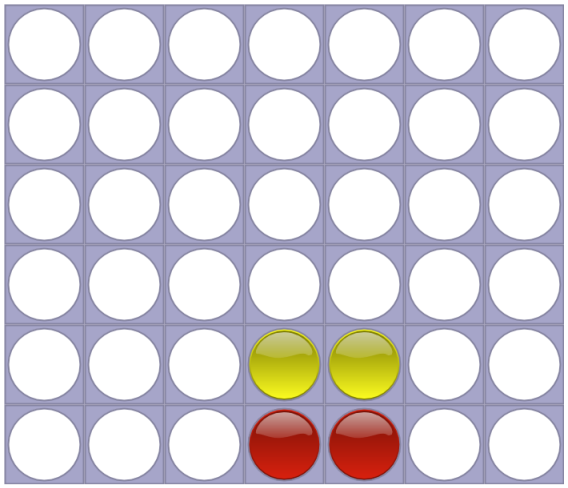

Fig.45 a board state with 18 score

As mentioned, negamax algorithm is used in this project to save computational time. If the score of one player is calculated based on a board state, then the score for another player can be calculated by simple maths. Through this implementation, the computation time can be significantly reduced along with the use of Alpha-Beta pruning. The actual implementation of negamax with Alpha-Beta pruning is shown in the figure (see Fig.46). In theory, this algorithm can compute the whole tree to obtain the result, but in reality, the problem size is still too big. A maximum of 14 moves can be computed within a minute. It takes too long of a reasonable response.

```
int negamax(const Position &P, int alpha, int beta) {
  if(P.nbMoves() == Position::WIDTH*Position::HEIGHT) // check for draw game
    return 0;

  for(int x = 0; x < Position::WIDTH; x++) // check if current player can win next move
    if(P.canPlay(x) && P.isWinningMove(x))
      return (Position::WIDTH*Position::HEIGHT+1 - P.nbMoves())/2;

  int max = (Position::WIDTH*Position::HEIGHT-1 - P.nbMoves())/2;       // upper bound of our score as we cannot win immediately
  if(beta > max) {
    beta = max;                     // there is no need to keep beta above our max possible score.
    if(alpha >= beta) return beta;  // prune the exploration if the [alpha;beta] window is empty.
  }

  for(int x = 0; x < Position::WIDTH; x++) // compute the score of all possible next move and keep the best one
    if(P.canPlay(x)) {
      Position P2(P);
      P2.play(x);               // It's opponent turn in P2 position after current player plays x column.
      int score = -negamax(P2, -beta, -alpha); // explore opponent's score within [-beta;-alpha] windows:
      // no need to have good precision for score better than beta (opponent's score worse than -beta)
      // no need to check for score worse than alpha (opponent's score worse better than -alpha)

      if(score >= beta) return score;  // prune the exploration if we find a possible move better than what we were looking for.
      if(score > alpha) alpha = score; // reduce the [alpha;beta] window for next exploration, as we only
      // need to search for a position that is better than the best so far.
    }
  return alpha;
}
```

Fig.46 negamax algorithm with Alpha-Beta pruning

3.3.2 Approach 2: Bitboard

To save computation time, bit map is used to store the board state instead of traditional array. Since there are 6 rows and 7 columns in a Connect Four board, a minimum of 42 bits are required to represent the board state, but for convenience 7 x 7 = 49 bits are used for storing the board state. The game board in bit order is shown in the figure below (see Fig.47).

```
.  .  .  .  .  .  .
5 12 19 26 33 40 47
4 11 18 25 32 39 46
3 10 17 24 31 38 45
2  9 16 23 30 37 44
1  8 15 22 29 36 43
0  7 14 21 28 35 42
```

Fig.47 game board in bit order

The encoding method make use of 1 and 0 to represent 3 situations. The discs for current player are encoded as 1, and the discs for the opponent are encoded as 0. To indicate empty cells, the 1 at the top of each column can tell that slots below that cell is full. The extra row on top will be useful if the column has 6 discs already. The figure below (see Fig.48) shows an example of board state represented by bit map.

```
          0000000
.......   0001000
...o...   0010000
..xx...   0011000
..ox...   0001100
..oox..   0000110
..oxxo.   1101101
```

Fig.48 an example of board state represented by bit map

Through bitwise operation, the checking of winning condition becomes very easy and efficient. In the figure below (see Fig.49), the winning checking algorithm when a new disc inserted is shown. The horizontal line will be checked first, then the diagonal lines and the vertical line. This method only checks the surrounding discs related to the new disc. This significantly save the computation time along with the use of bitwise operation.

```
static bool alignment(uint64_t pos) {
  // horizontal
  uint64_t m = pos & (pos >> (HEIGHT+1));
  if(m & (m >> (2*(HEIGHT+1)))) return true;

  // diagonal 1
  m = pos & (pos >> HEIGHT);
  if(m & (m >> (2*HEIGHT))) return true;

  // diagonal 2
  m = pos & (pos >> (HEIGHT+2));
  if(m & (m >> (2*(HEIGHT+2)))) return true;

  // vertical;
  m = pos & (pos >> 1);
  if(m & (m >> 2)) return true;

  return false;
}
```

Fig.49 alignment checking algorithm

3.3.3 Approach 3: Move Ordering

The exploration order of children nodes significantly affect the performance of Alpha-Beta pruning. If a better node is explored first, the pruning can be performed earlier. Hence the computation time is saved.

In the case of Connect Four, the middle column have advantages in comparison to other columns as alignments can be formed more easily on average. The moves in the middle column is often considered as a better move. Therefore, an order can be formulated to enhance the pruning performance, which is start with column 4, then 3, 5, 2, 6, 1 and 7. Following this order, no rearrangement of nodes is required and the performance of pruning is increased.

Moreover, “bad moves” that leads to the winning of opponent in the coming turn should be avoided. There are 3 rules related this principle.

1. If the opponent has an obvious winning move in the next move, the disc should be placed there to prevent opponent from winning.

2. Disc should not be placed if that move facilitate the winning of opponent in the next round.
3. If there are two or more winning slots for the opponent, no moves need to be explored. The winning of the opponent is unstoppable.

3.3.4 Approach 4: Transposition Table

In a Connect Four game, many game states will be analysed repeatedly. These results can be cached to boost the performance of the algorithm as recalculation takes a lot longer then retrieving an existing result. It is reasonable that if results are cached, the memory needed to store cache will increase significantly. However, the trade-off between the computation speed and memory size is worth it.

The table below (see Table.2) shows the benchmark result in different cases. In the early stage of the game where at least 28 moves can be made, the average computation time takes around 5 seconds.

Test Set name	nb moves	nb remaining moves	Mean time	mean nb of pos	K pos/s
End-Easy	28 < moves	remaining < 14	4.722 µs	54.93	11,630
Middle-Easy	14 < moves <= 28	remaining < 14	39.90 µs	517.4	12,960
Middle-Medium	14 < moves <= 28	14 <= remaining < 28	3.736 ms	48,450	12,970
Begin-Easy	moves <= 14	remaining < 14	275.5 µs	3,693	13,400
Begin-Medium	moves <= 14	14 <= remaining < 28	113.4 ms	1,459,000	12,870
Begin-Hard	moves <= 14	28 <= remaining	5.667 s	72,490,000	12,790

Table.2 benchmark results of different number of moves

To save time, extra book is added to help the calculation at early stage. The algorithm can read the book directly to avoid massive calculation. The size of the book is 32MB, and it is a byte sequence file that stores key value pairs [19]. Through this method, the computation time at early stage can be reduced significantly.

3.3.5 Result of Experiment: Enhanced Connect Four AI

After a series of enhancement, the Connect Four solving algorithm can compute any game state in a very short amount of time. Moreover, the algorithm is able to output optimal strategy that aligns with the objective of this project, which is building an unbeatable AI. It is believed that this optimisation of algorithm is much better than using brute force algorithm.

3.4 Experiment 4: Disc Stacker

In this project, disc stacker was used to store discs for the robotic arm. However, it is not as simple as imagined. Several versions were designed to fix problems encountered. Different techniques were also learned and used to solve those problems. At the end, a hybrid approach of 3D-printing and laser cutting were used to makes the final product.

3.4.1 Approach 1: 3D printing

Unlike many chesses game, the disc in a Connect Four game must be placed in a very narrow slot and it require very precise control of the position where the disc is being picked. To solve this problem as well as automate the disc refill process, a disc stacker was designed. The disc stacker must be able to serve a few requirements, including:

1. Every time the disc can be picked at the same place.
2. The disc must be able to contain a minimum of 21 discs.
3. The disc stacker must be able to roll out another discs automatically when one disc is picked by the robotic arm.
4. The disc must be able to stand alone without human interference.

After gathering the requirements, the first version was designed (see Fig.50). The overall design is simple, the disc stacker consists of 3 inclined tracks where the height is 3cm to fit the diameter of the discs. At the end of the track, a holder will stop the disc from rolling and hold the position. Soon after the design was made, the first trial was conducted.

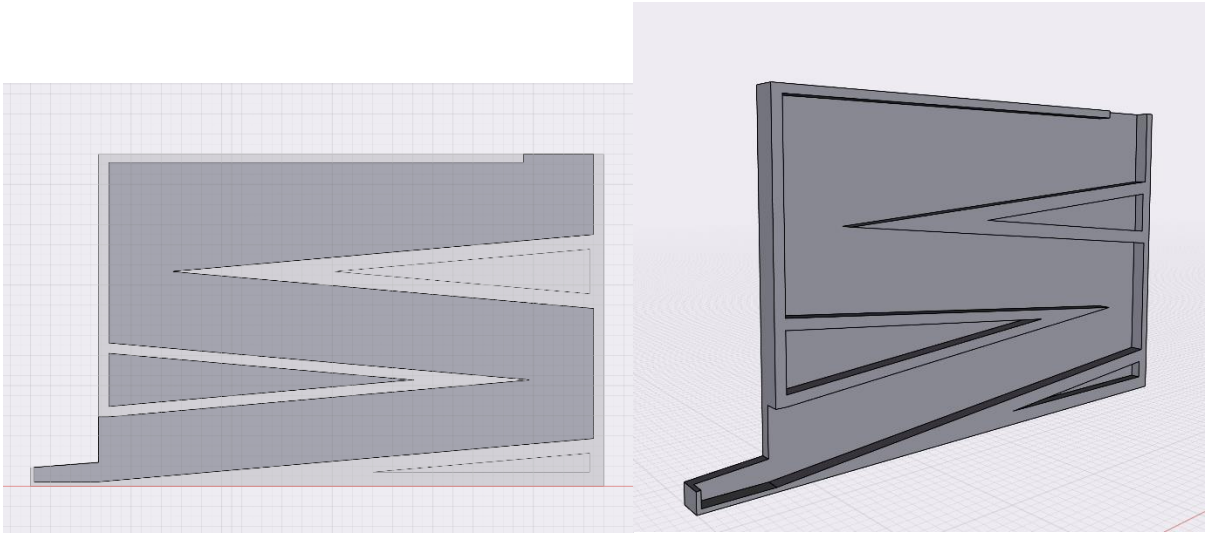

Fig.50 side view and 3D model of the first disc stacker design

The resulting disc stacker is shown as follows (see Fig.51). The disc stacker was printed using the Ultimaker 3 extended. There are a lot of debris along the tracks, discs cannot be rolled out smoothly. The height of tracks was also too short for the discs. This attempt was not successful.

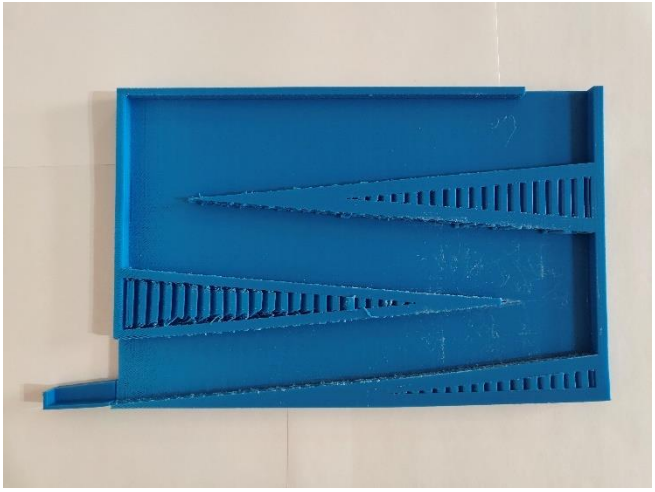

Fig.51 3D-print result of the first disc stacker

From the previous version, the next disc stacker cannot be printed vertically since it will produce a lot of debris and the height of the tracks must also be adjusted.

Therefore, another version of disc Stacker was designed (see Fig.52). This version of disc stacker is similar to the previous version except the height of the tracks were taller. Ultimaker 5 was used to print the disc stacker horizontally and no debris appeared this time. However, problem still exist.

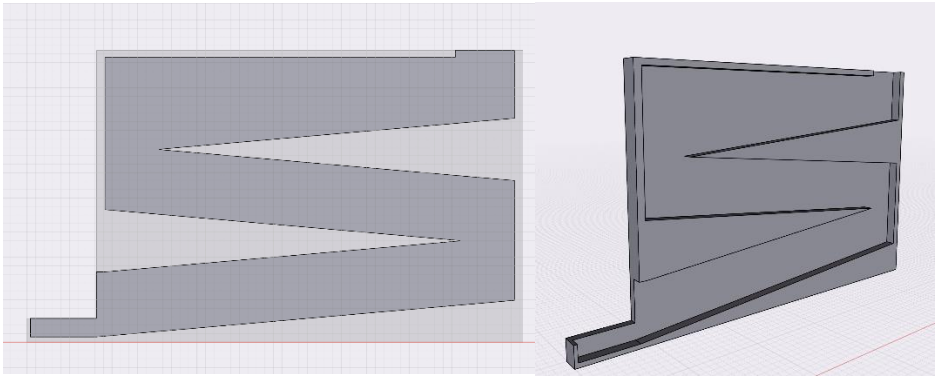

Fig.52 side view and 3D model of the second disc stacker design

From the figure (see Fig.53), it is clear that both the main track and the black plate is curved due to the cooling inconsistency in 3D printing. Not only the plate is curved, but the holder was also too shallow, which makes the discs roll out of the disc stacker easily. Most importantly, the inclined angle, which was 5 degrees, was not enough for disc to roll out very smoothly.

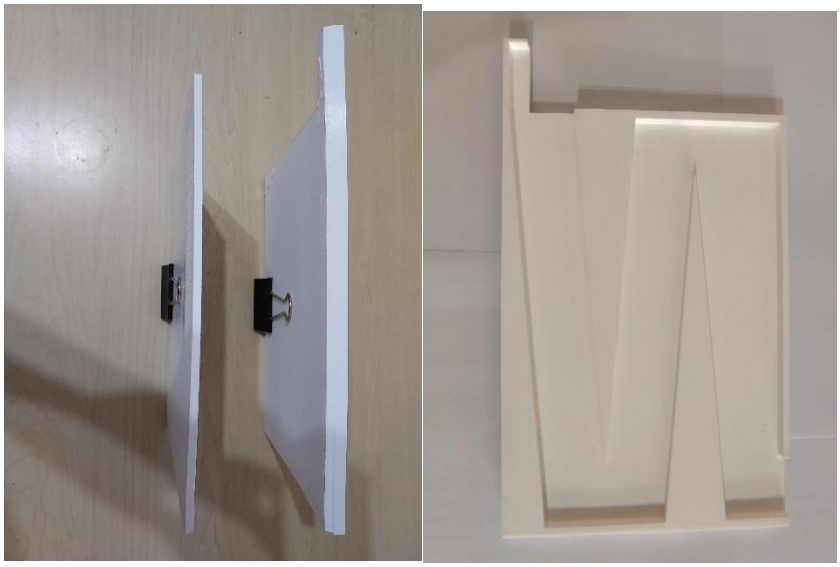

Fig.53 3D-print result of the second disc stacker

3.4.2 Approach 2: Laser Cutting

The result of only using 3D printing is not very satisfactory. Therefore, it is necessary to think of a new approach. After illustrating our requirements and problems with the Technical Manager of the Tam Wing Fan Innovation Wing – Mr. C.S. Seto, his suggestion was using laser cutting for the black plates and the tracks and combine them using glues or nails.

After receiving advice from Mr. C.S. Seto, a new model was designed (see Fig.54). Improvements were made in the angle of tracks as well as the holder. The angle was increased to 10 degrees. The holder was deeper and shaped to fit the disc better.

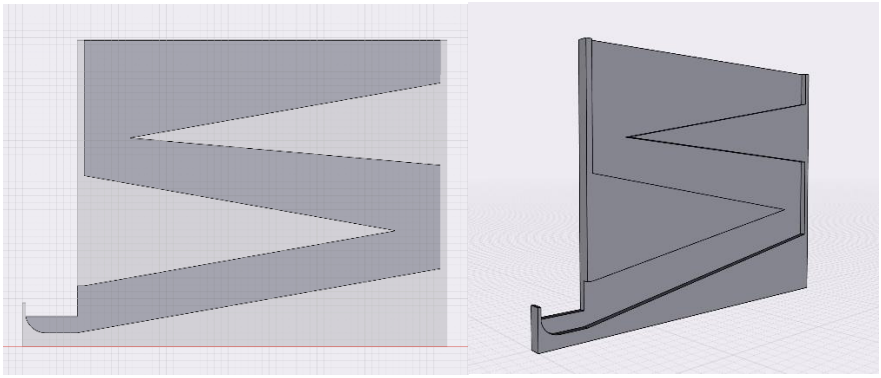

Fig.54 side view and 3D model of the third disc stacker design

3.4.3 Result of Experiment: A hybrid approach

The 3D model was then translated to dxf format, which is the format required for laser cutting (see Fig.55). With the help of Mr. C.S. Seto, parts were cut out using the GCC LaserPro Spirit in Tam Wing Fan Innovation Wing. The result was satisfactory, the tracks were smooth and the plates were straight.

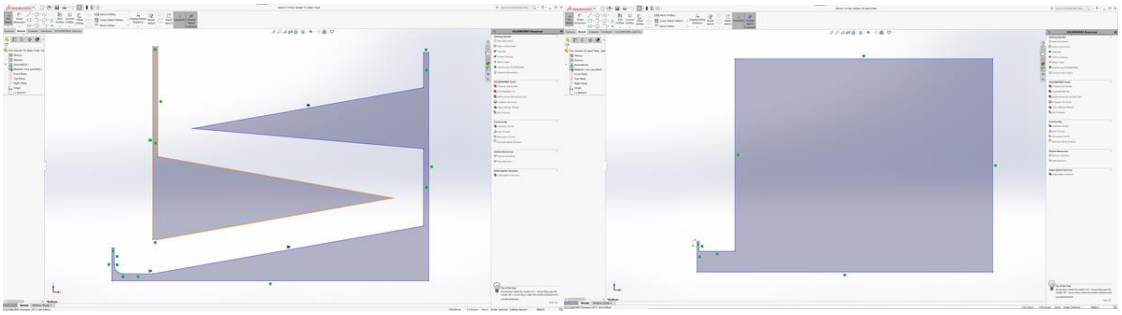

Fig.55 design of the third disc stacker drawn in SolidWorks

Yet, the parts must be combined, and the stacker must be able to stand alone. To solve this problem, a wedge was designed to holder everything together (see Fig.56).

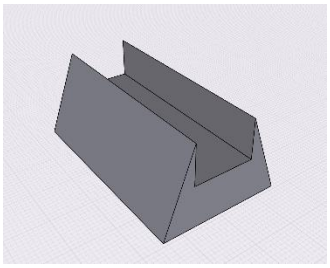

Fig.56 design of the wedge

The wedge can holder parts together while provide a solid base support that prevent the disc stacker from toppling. The final version of disc stacker is as follows (see Fig.57).

Fig.57 final version of disc stacker

In the figure, it can be seen that everything is hold together without applying too much pressure, while preserving the flexibility in disassembling the disc stacker. After over 20 times of testing, it is proved that the disc stacker works very well, none of the discs was stuck and disc can be gripped by the robotic arm easily.

4 Limitation and Future Works

In this section, the limitations of the project will be discussed. It will be divided into two topics, the hardware limitations and software limitations. Moreover, the enhancements that can be made in the project will be mentioned.

4.1 Hardware Limitation

First, distance between the robotic arm, the game board and the disc stacker are fixed to save time configuring the setup. If their relative positions are changed, the programmed coordinates will be wrong. Hence, the robotic arm will not be able to pick and place discs to desired location. The setup must be the same in order to reproduce the expected result every time.

Another limitation is the precision of the robotic arm. In theory, the robotic arm should be able to move to exact the same location if the same command is given. However, in reality there are always some slight differences in the position. After careful examination, it is found that the connection of plastic gears created a slight degree of freedom for the arm. Even though the stepper motors operate correctly, the robotic arm may still deviate from the expected result. Moreover, plastic gears and parts can be worn out or damaged easily, which both cases happened during the development process. The above problem caused the placing of discs sometimes inconsistent as the slot is very narrow.

In an ideal situation, the robotic arm should be able to pick up discs without the assistance of a disc stacker, but in real life it will take to much effort to build such a system. If the discs are placed horizontally in an ordered or un-ordered way, additional camera must be placed above the discs and use object detection to identify the positions of the discs. Moreover, the design of the gripper must be modified significantly. Since the discs are placed horizontally and the discs must be inserted vertically, it is necessary for the disc to be able to rotate the disc by 90 degrees. Due to the time constraints, the use of disc stacker solves the problem in an more efficient manner.

4.2 Software Limitation

Due to time limitations, and Android is selected as the target platform. The application only support Android phones but no other platforms such as iOS. However, it is understandable that cross-platform support is not given as it is not a main objective of the project. This is a very minor limitation.

To increase the accuracy in board detection, sacrifices was made such as sticking a black sheet on the back of the board and framing the board. Through varies testing, it is shown that with a complex environments and backgrounds, the accuracy of the detection algorithm decreased significantly. To ensure the game performance, modifications was made to minimize the possibilities of undesirable result. This is a worthy trade off as the installation of black sheet increased the overall performance of the object detection algorithm significantly.

4.3 Future Development

From the hardware perspective, a manual calibration function can be implemented so that the distance between the robotic arm, the game board and the disc stacker can be set in a larger degree of freedom. For example, the user can use the Arm controller to pick a disc and record it as the new position of the disc stacker. Then, the user can put the discs to the seven columns respectively and record the new positions. So that the application knows where the new positions of the board and the disc stacker are. It will be convenient when the layout of the setup is changed.

Moreover, different levels of AI can be built to simulate different difficulties. For instance, the game AI is this project always perform the optimal moves, but in reality, human will make mistakes. Another AI can be built to simulate human behaviours by making random mistakes. The lower the difficulty, the more mistakes the AI will be. On the other hand, a self-learning AI can be implemented to learn from player's movement. Given sufficient game data, an AI that replicate the player's behaviour can be built. Then the player will be able to play game with "himself/herself". Player can learn from himself/herself to improve the skills.

Object detection can also be enhanced. Currently, a black sheet was placed behind the board to reduce the noise. Yet, making use of ToF camera in some of the high-end phone might be

able to solve the problem. Through reading data from the ToF camera, the depth of the environment can be read. Hence, the board can be isolated from the background, eliminating any possible noise and improve the accuracy.

5 Conclusion

To sum up, this objective of this project is to build a Connect Four AI that can play with human opponents automatically with the assist of a robotic arm. After varies development stages and experiments, the application can now recognise the board accurately, calculate the optimal moves in a very short amount of time. The pick and place process have also been optimised, the robotic arm can perform the move smoothly and consistently.

This project involves many AI-related technologies such as object detection and strategic algorithms. It is hope that in the process of playing, users will be able to understand some of the principles behind these technologies. As well as admiring the beauty of Computer Science and develop interest in Computer Science field.

Reference

[1] P. Mozur, “Google's AlphaGo defeats Chinese go master in win for A.I.,” *The New York Times,* 23-May-2017. [Online]. Available: https://www.nytimes.com/2017/05/23/business/google-deepmind-alphago-go-champion-defeat.html. [Accessed: 27-Oct-2021].

[2] G. Team, “The rules of Connect 4 (according to M. Bradley & Hasbro),” *Gamesver,* 02-Jul-2021. [Online]. Available: https://www.gamesver.com/the-rules-of-connect-4-according-to-m-bradley-hasbro/. [Accessed: 27-Oct-2021].

[3] P. Pons, “Part 1 – introduction,” *Solving Connect 4: how to build a perfect AI*, 01-May-2016. [Online]. Available: http://blog.gamesolver.org/solving-connect-four/01-introduction/. [Accessed: 27-Oct-2021].

[4] “4 in a row king,” *Mobirix,* 20-Dec-2020. [Online]. Available: https://play.google.com/store/apps/details?id=com.mobirix.connectfour. [Accessed: 18-Apr-2022].

[5] “4 in a row,” *Quarzo Apps,* 4-Jan-2022. [Online]. Available: https://play.google.com/store/apps/details?id=com.quarzo.fourinarow. [Accessed: 18-Apr-2022].

[6] P. Pons, “Solving connect 4: How to build a perfect AI,” *Solving Connect 4: how to build a perfect AI*. [Online]. Available: http://blog.gamesolver.org/. [Accessed: 18-Apr-2022].

[7] P. Pons, “Connect 4 solver,” *Game Solver*. [Online]. Available: https://connect4.gamesolver.org/. [Accessed: 18-Apr-2022].

[8] “Firebase Realtime Database,” *Google,* 13-Apr-2022. [Online]. Available: https://firebase.google.com/docs/database. [Accessed: 18-Apr-2022].

[9] “The color system,” *Material Design*. [Online]. Available: https://material.io/design/color/
the-color-system.html. [Accessed: 22-Jan-2022].

[10] S. Subramaniyan, “The rounded user experience,” *Medium,* 30-Jul-2020. [Online]. Available: https://uxplanet.org/the-rounded-user-experience-ff7a1898ab33. [Accessed: 18-Apr-2022].

[11] E. Macpherson, “The UX honeycomb: Seven essential considerations for developers,” *Medium,* 08-Oct-2019. [Online]. Available: https://medium.com/mytake/the-ux-honeycomb-seven-essential-considerations-for-developers-accc372a398c. [Accessed: 18-Apr-2022].

[12] “Mat Class Reference,” *OpenCV*. [Online]. Available: https://docs.opencv.org/4.x/d3/d63/classcv_1_1Mat.html. [Accessed: 18-Oct-2021].

[13] “Hough Circle Transform,” *OpenCV*. [Online]. Available: https://docs.opencv.org/4.x/da/d53/tutorial_py_houghcircles.html. [Accessed: 18-Oct-2021].

[14] T. Körting, “How Circle Hough Transform works,” *YouTube*, 01-Dec-2013. [Online]. Available: https://www.youtube.com/watch?v=Ltqt24SQQoI. [Accessed: 17-Apr-2022].

[15] “Minimax Algorithm in Game Theory | Set 1 (Introduction),” *GeeksforGeeks*, 31-Mar-2021. [Online]. Available: https://www.geeksforgeeks.org/minimax-algorithm-in-game-theory-set-1-introduction/. [Accessed: 27-Oct-2021].

[16] “Minimax algorithm in Game theory: Set 4 (alpha-beta pruning),” *GeeksforGeeks*, 18-Aug-2021. [Online]. Available: https://www.geeksforgeeks.org/minimax-algorithm-in-game-theory-set-4-alpha-beta-pruning/. [Accessed: 27-Oct-2021].

[17] B. Haran, “Connect four - numberphile,” *YouTube*, 01-Dec-2013. [Online]. Available: https://www.youtube.com/watch?v=yDWPi1pZ0Po. [Accessed: 17-Apr-2022].

[18] F. Tobler, “Robotarm by ftobler,” *Thingiverse*, 14-Aug-2016. [Online]. Available: https://www.thingiverse.com/thing:1718984. [Accessed: 17-Apr-2022].

[19] P. Pons, “Release opening book · Pascalpons/Connect4,” *GitHub*, 25-Jan-2019. [Online]. Available: https://github.com/PascalPons/connect4/releases/tag/book. [Accessed: 17-Apr-2022].