Progress Report 2

Playing Chess with Robotic Arm

By

Yunhai Wang
wyh5625@hku.hk

Supervisor: Professor K.P. Chan

Date: 3/12/2012

Acknowledgement

Throughout the process of researching, planning and implementing this project, I have received many assistance and guidance from various parties. Without these individuals who are willing to share their experiences and time to give me a helping hand, I may not have arrived the current milestone - build the runnable robotic arm and a simple chess game app. Thus, in this section, I would like to express my heartfelt gratitude to all of these individuals who had supported me.

Firstly, I would like to thank my supervisor, Professor K.P Chan who has guided me on how to search the paper of AI and prepare the project. He had inspired me with his ideas on how to implement AI technology by using different platforms. Thanks to his sharing of his precious time with me, I could find me to discuss on the project weekly.

Next, I would also like to thank the technical staff - David Lee who have given me lots of instructions on how to build robotic arm and develop chess app on Android platform. He patiently taught me hardware and software techniques of realise the ideas from theory to the project. Without his kindly help, I may not be able to make great progress by my little experience.

Last but not least, I would also like to thank Faculty of Computer Department for providing facilities such as 3D printer, tool kits and desktops which had facilitated my progress toward finishing the project.

Abstract

In the last decade, 3D printer has expanded to make it possible for people to build robots at home [3]. In this report, a chess playing robotic arm has been made and will be improved in the future phases.

This type of robotic system consists of the physical robotic arm and a chess AI. The robotic arm requires active control to be manipulated accurately in moving pieces on chessboard and the AI software developed on Android platform takes jobs of chessboard recognition and decision making strategy. Applying BLE technology, the robotic arm and the mobile app can be connected together to produce the final product.[4]

The robotic arm applies the fundamental control methods of kinematic forward and inverse theories.[5] Being a open loop controlling system, the arm need to be well designed in order to overcome the positional error during manipulation. What's more, the recognition of chessboard should be accurate since the manipulation depends on the it.

In the latest progress of this project, the basic prototype of robotic arm has been designed and put into experiment. The physical part of is project is almost done, while the software part has just been made as a simple chess console game on Android platform. The function of app controlling robotic arm will be implemented in the future.

Table of Contents

Table of Figures & Tables

1. Introduction

1.1 Background

Today AI has become more and more important in many areas of human activity. It can not only handle most of repetitive tasks in a variety of working area such HR, IT, marketing and so on, but it also takes part in entrainment in our daily life. Thus, ideas of integrating AI elements into human competition have been emerging in recent years. One of the most interesting and fantastic idea is making a chess playing AI engine that can compete with top human players.

1.2 Motivation

In recent events, powerful AI engine has been created which is able to beat grandmasters in chess games. For example, AlphaGo, a DeepMind's AI, has defeated some of world's top human Go players and has become the strongest Go player in history. Although AI becomes more intelligent, it is a mere computer program that can not accomplish physical tasks in the real world. Therefore, this project aims to make a dexterous, perceptual and thinking robotic arm as physical extension of AI. In the first stage, it will focus on making a dexterous robust robotic arm that can precisely complete tasks of moving pieces on chessboard. In the second stage, an AI app will be designed and built to handle perception of chessboard position and decision-making of best move. In the final stage, communication technology such as BlueTooth will be applied to make a connection between the robotic arm and the AI.

1.3 Previous work

There are plenty of projects realizing this idea. For example, Raspberry Turk[6] is such a project that uses Raspberry Pi together with Camera Module for handling computer vision. Certabo Chessboard is another case of using sensors to perceive the position of pieces instead of Camera Module. As a matter of fact, these kinds of machine are not widely used due to its bulky body which roughly combines robotic arm, Raspberry Pi and camera together. Thus, it may not be easy and convenient to put chess machine into use for general users. For the benefit of making relatively small and user-friendly one, this project will divide overall system into parts and move more functions of RPi and camera onto mobile phone.

1.4 Scope and Objectives

Since this project is focuses on application level, it will not go deeper into inventing new types of AI method. It will simply apply well-behaved AI model which can perform as well as grandmasters. For category of game, it will take the baby step of mastering chess before it can compete in more complex games in the future.

The basic goal of this project is to make a chess playing machine. Furthermore, it aims to extend its capabilities and functions. The overall objectives are as follows:

- The initial goal is to build a chess playing machine which features a robotic arm and has a controlling system based on AI.

- The intermediate goal is to further develop the capability of this machine so that it can play more kinds of chess game, such as Go, Othello and so on. This mainly involves the development of AI application.

- The ultimate goal is to develop a learning platform for users who have just started learning playing chess and want to improve skills through practice.

1.5 Outline

The remaining parts of this report proceeds as follows. First, it explains the theory applied in building the robotic arm controller and give a detailed description of the implementation. Next, it discusses on the current status which shows the intermediate product of this project and what will be implemented in the next phase. Then, it summarizes the deliverables in schedule and milestone chapter. Finally, it give a discussion on the challenges and limitation of the overall development procedure.

2. Theory

2.1 Control Theory - Forward Kinematics

The robotic arm requires three degrees of freedom at least to complete the movement task. To make things a bit simpler, a manipulator with two degrees of freedom is analysed here where the consideration is rotational effect of Joint 1 and Joint 2 (Fig 2.1). It is intuitive to describe the end effector position, x_e and y_e by

$$x_e(\theta_1,\theta_2) = l_1\cos\theta_1 + l_2\cos(\theta_1+\theta_2) \quad (2.1)$$

$$y_e(\theta_1,\theta_2) = l_1\sin\theta_1 + l_2\sin(\theta_1+\theta_2) \quad (2.2)$$

Furthermore, given the end effector position, the combination of joint angles can also be calculated by inverse kinematics which is obtained as input of step motor.

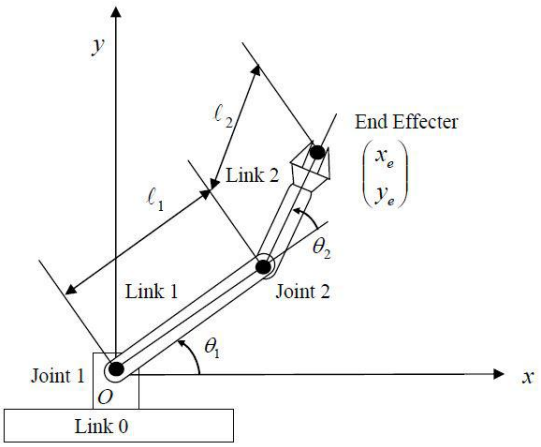

Fig 2.1. Manipulator with two degrees of freedom [7]

2.2 AI - Deep Learning + Search tree

It is worth noting that different positions result in different advantages/disadvantages by comparing with slightly different positions. For example, positions featuring more one's pieces than the opponent's increase the opportunities of winning the game without considering the level of player. To make a good AI, it would be better to bear two things in mind:

1. Extract features from position such as how many pieces left on the chessboard, whether castling is still available, etc.
2. Compare positions with respect to their features.

Thus, idea of mixing Neural Network and Alpha-Beta Search is borrowed from DeepChess. The approach is dividing into three parts:

2.2.1 Training Pos2Vec:

It trains a deep belief network (DBN), which will extract features vectors from position without incorporating the rules of the game by unsupervised training (Fig 2.2). Each position is converted to a 773-bit string. There are 2 sides(White and Black), 6 piece types(pawn, knight, bitshop, rook, queen and king) and 64 blocks. There are five additional bits representing side to move(1 for White and 0 for Black) and castling rights. Therefore, the input size is $2 \times 6 \times 64 + 5 = 773$. The DBN network consists of five fully connected layers of size: 773-600-400-200-100.

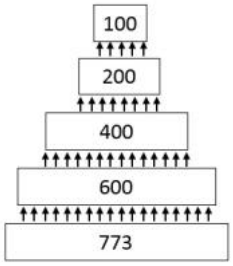

Fig 2.2. DBN [2]

2.2.2 Training DeepChess:

With DBN functions as feature extractor, a comparison method called DeepChess is employed which compares every feature results extracted from two positions by adding four fully connected layers of size: 400-200-100-2 on top of them with a 2-value softmax output layer (Fig 2.3). It is trained to predict which of the two positions are more likely to result in a win.

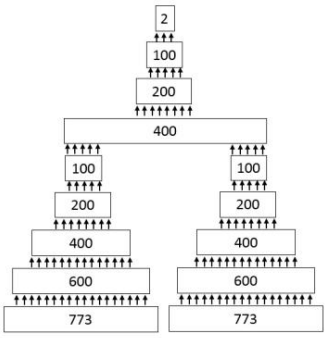

Fig 2.3. Two DBN plus DeepChess network [2]

2.2.3 Comparison-Based Alpha-Beta Search

In search of best possible move, alpha-beta search algorithm is used to prune unpromising branches of the search tree earlier to reduce the search time. It store two values, α and β, which represent the minimum score that maximizing player is assured of and maximum score that minimizing player is assured of. It stops when α is larger than β, which means minimizing player will never consider further descent nodes as it has better options. Most of time, it can not reach the leaf nodes as the search tree is very large. Thus, it adopts a evaluation function to value the non-terminal nodes. The concern is that it requires a rather reasonable and exact evaluation function to value the nodes. In this project, it store α_{pos} and β_{pos} instead of α and β. By using DeepChess, it compares the current position with its next possible positions. If the comparison shows that the new position is better than α_{pos}, it would become new α_{pos}, and if the new position is better than β_{pos}, it would be pruned. The benefit of this approach is it doesn't require any position score for performing the search.

3. Implementation

3.1 Working Logic

The working logic is just a looped workflow, which is sufficient to describe the system of machine (Fig 3.1). Chess is a turn-based game where two players take turns when playing. Therefore, analysing one turn of AI player is sufficient to obtain the overall functions of system. Firstly, human player makes a move, after which the position of chessboard has been slightly changed which is captured by phone camera for recognizing position. Secondly, the new position information is used as input of pre-trained powerful AI model to obtain the next best move for itself. Finally, it controls robotic arm to make a move.

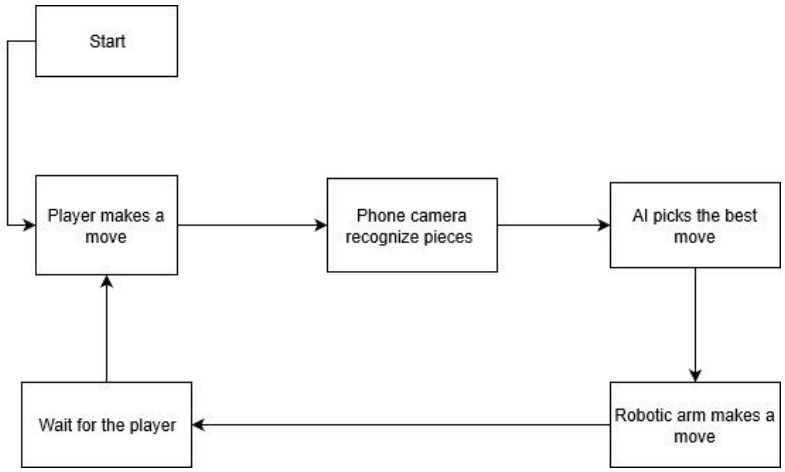

Fig 3.1. Workflow of System

3.2 Hardware

To begin with, a robotic arm was being designed and made in phase one. The robotic arm controlled by AI app via Bluetooth was designed such that it can reach all 64 blocks on the chessboard.

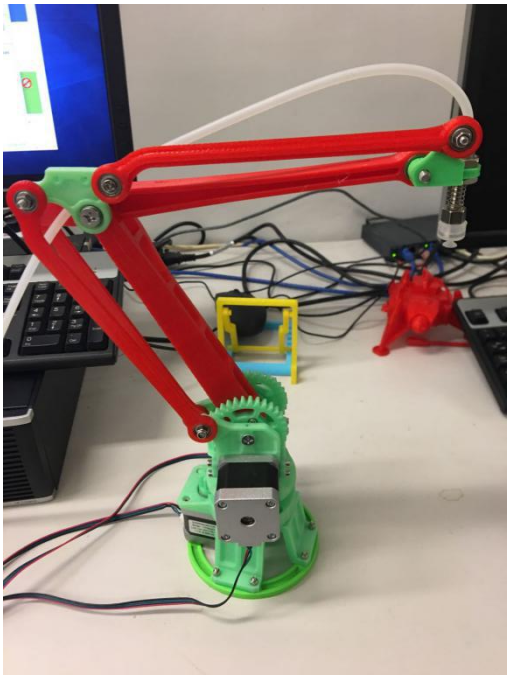

Fig 3.2 Alpha version of robotic arm

3.2.1 Controller

The micro controller board used in this project was the Arduino Mega 2560 together with a RAMPS 1.4 Board and A4988 motor Drivers installed on it (Fig 3.2). The Arduino Mega board was used as an integrated controller unit for converting high-level command into low-level machine code which was fed as input to motor driver. The Arduino Mega board consists of up to 54 digital input/output pins with benefits of multiple control compared with basic Arduino UNO board, which is enough for controlling step motors as well as communications of BLE. The Mega is compatible with most shield designed by Genuino Uno.

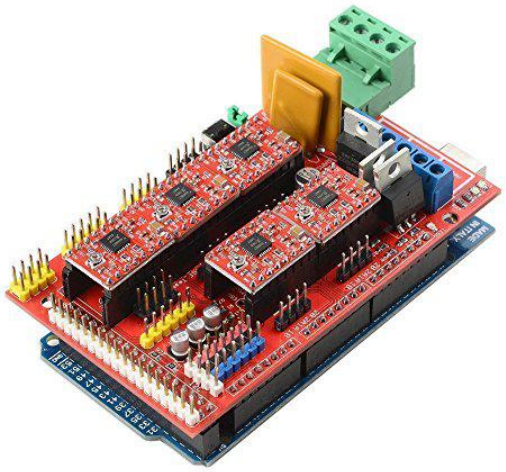

Fig 3.3. Controller kit

3.2.2 Stepper Motor

Three stepper motors (Fig 3.4) of type NEMA17 were assigned for this project, which provide enough power for fast movement and high precision. They take the responsibility of rotating robotic arm in its 3 degrees of freedom respectively. Table 3.1 shows each motor's specifications from the datasheet.

Table 3.1. Stepper motor specifications

Size	NEMA 17
Step angle	1.8° full step 0.9° half step
Phase/Windings	4/2
Voltage & Current	12V at 400 mA
Holding Torque	2000 g-cm
Detent Torque	220 g-cm max
Max continuous power	5 W

Fig 3.4 Stepper motor

3.2.3 Robotic Arm

This project applied SOLIDWORK for build robotic arm model since it provides sufficient features for components designing and assembly emulation. Moreover, it can save 3D model in STL format which is suitable for 3D printing (Fig 3.5). The model was initially downloaded from Thingiverse 3D model website, and was modified a little bit to meet the requirement of minimum arm length. The size of chessboard should be considered in order to obtain minimum arm length since it determines the area of its work zone. In other words, the working zone should cover entire area of chessboard. To achieve this, proper arm length had to be determined. A reasonable length obtained after plenty of experiments was 180 mm.

Fig 3.5. Assembly model of robotic arm built by SOLIDWORK

3.2.4 Vacuum Gripper

Vacuum gripper (Fig 3.6) was installed on the endpoint of robotic arm. It is ideal for picking up even and uneven pieces made of different materials including plastic. It has benefits of small size comparing with other types of gripper such as clamp gripper and so on.

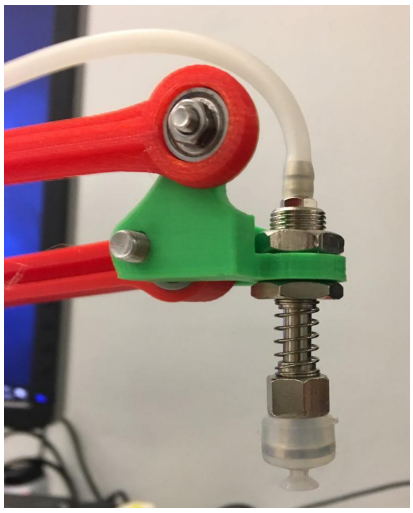

Fig 3.6 Vaccum gripper installed on the end point of robotic arm

3.2.5 Power Source

At the beginning stage, AC adapter (Fig 3.7) was used to provide power to robotic arm and the controller, which can provide voltage at 12V. In experiment, an AC adapter can provide stable power for robotic arm without considering about charging. In the future, battery will take the place of AC adapter to reduce the weight the robotic arm.

Fig 3.7 AC adapter

3.3 Software

An app will be built on Android platform. It gives user an interface to interact with robotic arm as well as plays roles as eye and brain of robotic arm which can captures chessboard and makes best movement. For the eye part, it will employ phone camera and OpenCV library to recognize the position of chessboard. For the brain part, it will apply powerful AI model of deepchess.

3.3.1 App Features

In order to build the full feature app that can handle all tasks mentioned above. The total app is divided into feature apps in order to acquire the target functions one by one.

Table 3.2 App features

Category	Target	Progress
Robotic Arm Controller	Robotic arm controller v1.0.(i)	Done

	Robotic arm controller v2.0.(ii)	Pending
Chess Board Scanner	Chess Board Scanner (iii)	Pending
Game Logic Design	Single Player Chess Game (iv)	Done

Robotic Arm Controller:

i. An robotic arm controlling interface that can control robotic arm endpoint movement in x,y and z axes.(See Fig 3.8)

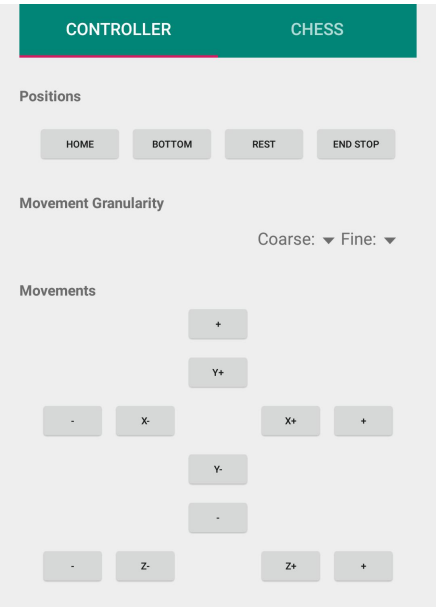

Fig 3.8 Controller UI

ii. An robotic arm controlling interface that can move a piece to a target position as a coordinate in 3D space.

Chessboard Recognition:

iii. A scanner that can scan chessboard, obtain position of each piece on the board and present the chessboard in a matrix.

Game Logic Design:

iv. A single player chess game app with a beginner level chess engine.(See Fig 3.9)

Fig 3.9 Chess UI

3.3.2 AI with TensorFlow

This project will develop the AI model using TensorFlow. As an open-source project, TensorFlow is a software library designed to do numerical computation. It can support multiple platform such as desktop, server and mobile platform, and can run on both CPU and GPU. By using TensorFlow, developers only need to provide the architecture of their Neural Network model, choose the adaptive function they want to use, and then feed the training data into model. TensorFlow handles the underlying layer computation, which makes model training easier and clearer.

4. Current status

4.1 Hardware

The alpha version of robotic arm was made and can be controlled by Robotic Arm Controller v1.0. However, since the robotic arm doesn't limits its movement in x, y and z axes, the arm always oversteps the limit of physical body, which may cause damage to gears. Therefore, it is required to find a method to limit the movement of robot arm within a optimally safe range.

4.2 Software

For app design, two function features have been implemented on Android platform which are a robotic arm controller (App Feature i) and a single player chess game (App Feature iv). The controller was made to control the position of robot arm end point, while the app feature a poor chess engine for human player to compete with. Next, App Feature ii and iii will be implemented to obtained its abilities of recognition and moving piece.
As for implementation of AI, no progress has been made in this part. During next phase, deepchess model will be built and trained using TensorFlow.

4.3 Schedule and Milestones

This project involves both of hardware and software, which requires deep understanding and application of theoretical knowledge. Therefore, it is better to build hardware first as it is simple with respect to complex software part. The schedule and milestones with status are as table 4.1:

Table 4.1. Schedule and milestones

Date	Milestone	Status
Sep 2019	Deliverables of Phase 1	

	● Detailed project plan ● Project web page	● Done ● Done
Oct-Nov 2019	● Build robotic arm ● Make a simplifid version of chess app	● Done ● Done
Dec 2019	Deliverables of Phase 2 ● Further develop app so that it can control robotic arm for playing chess ● Develop android app that can recognize position of chessboard	● Pending ● Pending
Jan-Feb 2020	Deliverables of Phase 3 ● Apply DeepChess model for training. ● Import trained AI model onto app	● Pending ● Pending
Apr 2020	● Final product	● Pending

4.4 Challenges and Limitations

This is a very complex project which contains hardware design and software design. The developer should have good knowledge in these two area. The challenges are as follows:

4.4.1 Challenges

- Find the best length of arm that can make sure the work zone cover the area of chessboard.
- Update controller parameter when modifying arm length of 3D model.

- Make a suitable gripper for robotic arm.

4.4.2 Limitations

Since the robotic arm is a open loop control system which contains no feedback of endpoint position, positional error exists when executing rotational command. This error may be caused by gear clearance, friction, material distortion and so on. The indelible positional error can restrict the feasibility of using specific type of gripper. For example, a vacuum gripper can grip piece with low risk of accidentally knocking down neighbour piece but requiring high precision of movement, while a clamp gripper can allow much more positional error but with high risk of accidentally knocking down neighbour piece. However, it is hard to find positional error range. In early phase of this project, the error is assumed to be small and can be ignored.

5. Conclusion

Robotic arm plays a role as physical agent for virtual AI mind when playing chess. This project aims to implements realize this idea and go further steps into making mobile applications for it. Currently, a workable robotic arm has been designed and made which can be controlled by an open-source app based on Window platform. This is the essential part of this project on which other parts depend. Next, the robotic arm will be involved in chess game and a simple chess game application will be developed for connecting the real chess game with virtual one.

Reference

[1] D. Hassabis. Artificial Intelligence: Chess match of the century. Nature 554,413-414, 2017

[2] O. David, N. Netanyahu and L. Wolf. DeepChess: End-to-End Deep Neural Network for Automatic Learning in Chess. International Conference on Artificial Neural Networks (ICANN), Vol. 9887, pp. 88-96, 2016

[3] C.X.F Lam, X.M Mo, S.H Teoh, and D.W Hutmacher, "Scaffold development using 3D printing with a starch-based polymer," Materials Science and Engineering: C, Volume 20, Issues 1–2, 31 May 2002, Pages 49-56

[4] O. Michel, A. Elie, B. Jad, B. Mark, and H. Rabih, "Design and Development of a Mobile Application to Control a Chess Playing Robot using NI-EVS and LabVIEW," AUST., America, May. 2014.

[5] O. Bottema, B. Roth, Theoretical kinematics[M]. Courier Corporation, 1990.

[6] J. Meyer. (2017). Raspberry Turk [Online]. Available: http://www.raspberryturk.com/ [Nov. 11, 2019].

[7] H. H. Asada, "Introduction to Robotics"