Project Plan

Playing Chess with Robotic Arm

By

Yunhai Wang
wyh5625@hku. hk

Supervisor: Professor K.P. Chan

Date: 29/9/2019

Table of Contents

1. Introduction

1.1 Background

Nowadays AI has become more and more important in many areas of human activities. It can not only handle repetitive tasks in a variety of working area such HR, IT, marketing and so on, but it also helps to improve human's wellbeing through taking part in entrainment with them. 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 an 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 became the strongest Go player in history. Alhough 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 for it. In the first stage, it will focus on making a dexterous robust robotic arm that can precisely complete task of moving pieces on chessboard. In the second stage, 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 robotic arm and AI.

1.3 Previous work

There are plenty of projects realizing this idea. For example, Raspberry Turk is such a project that uses Raspberry Pi together with Camera Module handling computer vision. Other examples are 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 is not easy and convenient to put it into use for general users. For the benefit of making it small and user-friendly, this project will divide overall system into parts and move more functions of RPi and camera onto mobile phone.

1.4 Scope

Since this project is focusing 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 more and more complex games in the future.

2. Objective

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 contains 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 game and want to improve skills through practice.

3. Methodology

This part will first introduce the working logic of overall system. Then, it will give a detailed description of the implementation.

3.1 Working Logic

The working logic is just a loop workflow, which is sufficient to describe the system of machine(Figure 1). Chess game 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.

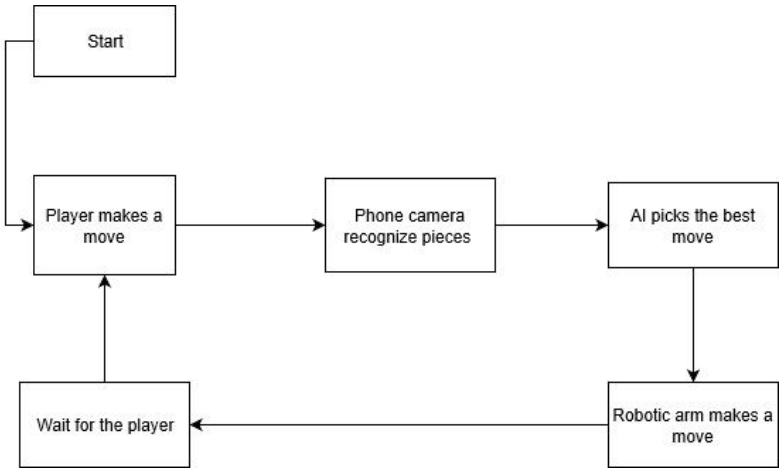

Figure 1. Workflow of System

3.2 Implementation

The overall system has two major parts: the robotic arm (hardware) and an AI app (software).
The first element to design is the robotic arm. The robotic arm is designed such that it can reach all 64 blocks on the chessboard. It is controlled by AI app via Bluetooth. The second part is an AI app which is the most challenging part throughout this project. It is installed on mobile phone such that it can use phone camera to recognize chessboard and recognize pieces on it. Then it makes the best move determined by powerful AI which is programmed and trained by Deep Learning.

3.2.1 Hardware -- Robotic Arm

In the first phase, a very smooth and flexible robotic arm will be designed which is controlled by a controller. The preview of completed robotic arm will be like the figure 2 below. However, there are a lot of robotic arm 3D structures online provided for downloading, which saves much time for designing. There are just a few modifications needed for making a proper one.

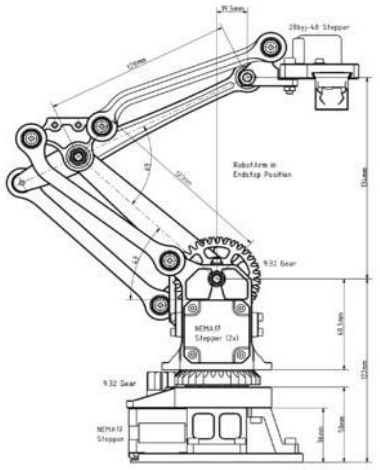

Figure 2. preview of robotic arm

3.2.2 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(Figure 3). It is intuitive to describe the end effector position, x_e and y_e by

$$x_e(\theta_1,\theta_2) = l_1\cos\theta_1 + l_2\cos(\theta_1+\theta_2)$$
$$y_e(\theta_1,\theta_2) = l_1\sin\theta_1 + l_2\sin(\theta_1+\theta_2)$$

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

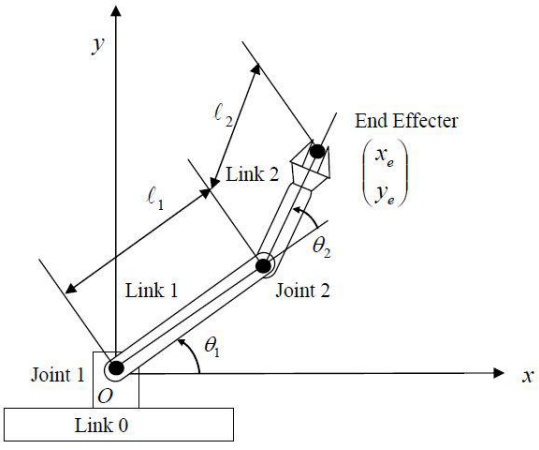

Figure 3. Manipulator with two degrees of freedom

3.2.3 Software -- AI app

This project will build an app on android platform which is widely used nowadays. This app behaves as eye and brain of robotic arm. It controls robotic arm through BlueTooth. The most challenging task of this app is to make an AI that can play chess as well as human or even better. On the other hand, it will also employ its camera to recognize the position of chessboard by using OpenCV library.

3.2.4 Deep Learning + Search tree

It is significant that different positions result in different advantages/disadvantages by comparing with slightly different positions. To make a good AI, it is better to bear two things in mind: 1. Extract features from position. 2. Compare positions with respect to their features [1]. Thus, idea of mixing Neural Network and Alpha-Beta Search is borrowed from [2]. The approach is dividing into three parts:

3.2.4.1 Training Pos2Vec:

It trains a deep belief network (DBN), which would extract features vectors from position without incorporating the rules of the game by unsupervised training (Figure 4). 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: 773-600-400-100.

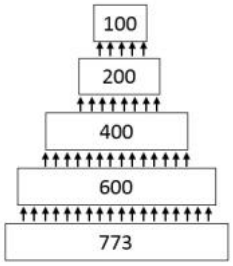

Figure 4. DBN

3.2.4.2 Training DeepChess:

With the above functions as feature extractor, a comparison method called DeepChess is employed which compared every feature results extracted from two positions. Then, four fully connected layers of size: 400-200-100-2 are added on top to them with a 2-value softmax output layer(Figure 5). It is trained to predict which of the two positions results in a win.

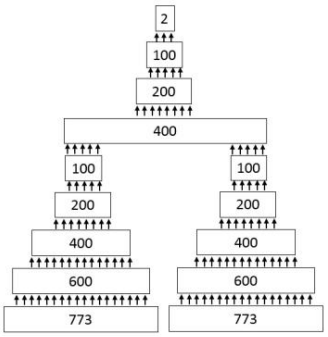

Figure 5. Two DBN plus DeepChess network

3.2.4.3 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.

3.2.4 Open CV

There is a library in android studio that can do recognition. It scans chessboard via phone camera, and present the position in the way that computer can understand.

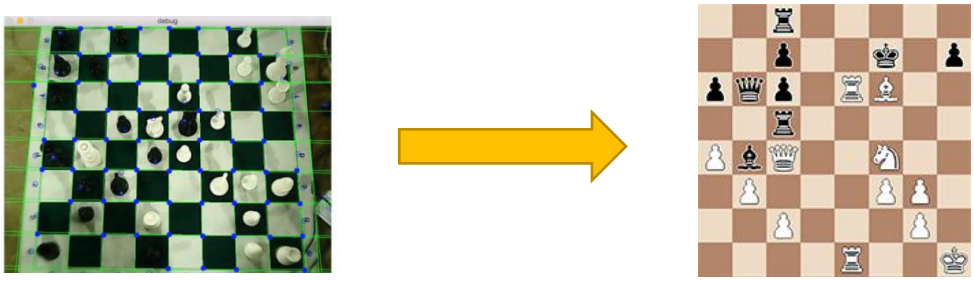

Figure 6. Transformation of position

4. 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 AI part. The schedule and milestones are as follows:

Date	Milestone
Sep 2019	Deliverables of Phase 1 ● Detailed project plan ● Project web page
Oct-Nov 2019	● Build robotic arm ● Make a simplifid version of chess app
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
Jan-Feb 2020	Deliverables of Phase 3 ● Apply DeepChess model for training. ● Import trained AI model onto app
Apr 2020	● Final presentation

5. 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