COMP4801

AI Chess
With Robotic Arm And Without Screen
Detailed Project Plan

Student: Wang Zehao

UID: 3035331642

Supervisor: Vincent Lau

Project Background

- **Historical work**

The robot was first introduced from a Czech play, which was sketched as a manufactured being to perform as labor. After this scientific conception, robots have been developed to be more and more anthropomorphized [1], which means not only performing logic or handling communications but attending actual human's life as well.

Chess is an ideal game for judging anthropomorphized intelligence. It has sophisticated game states, and practical for training. Under such circumstances, previous studies have gradually built up a system of computational logic powerful enough for self-studying and deep learning, such as tabula rasa reinforcement learning applied in alpha zero agents [2].

However, more concern was cast on the development of the game and search logic instead of anthropomorphization. While artificial intelligence has defeated human world champions, it still needs human effort for chess movement, and need human to read and input the game state of chess [3], which indicates that, based on the reality, the anthropomorphization of current chess robots contains a huge technical gap that can be improved.

- **Opinions**

AI chess can be played so conveniently now that relative applications are well developed on all devices. Basically, with the graphical user interface, the user can easily control the chess on the virtual chessboard and read the game state information. However, all user actions and resulted animations are only on a virtual basis. To progress further studies in a more anthropomorphized method, letting robots playing real chess with direct control of real chess and fetch information directly from the chessboard is a significant pace.

- **Current status**

  Recent studies on the chess robot control are focusing on fields of microelectronics and sensors control. These projects were designed with direct interfaces, low power supply, and simple logic [4]. From the results, we would appreciate these attempts with the robotic arm as a strategy to reflect the robot outcome and moreover, it follows the intention of anthropomorphizing. We noted that game step notifying and visual function activating still highly depends on human effort, thus hypothesizing that there is significant value in designing a robot with no human effort, that is, with a robotic arm to pick chess and without a screen for human control.

Project Objective

Our goal is to design a robot system with a control unit and a robotic arm. The control unit is able to retrieve current game data from the chessboard through the visual method, process and evaluate the next step, and inform the robotic arm to proceed.

Project Scope

- **Chess to use**

  In this project, we are using Chinese Xiangqi (象棋) for our concern. Xiangqi, a game of traditional chess popular in East and Southeast Asia, has a limited study with intelligence applications, thus having a huge value to explore and utilize. Also, as a sport especially popular among the elderly [5], Xiangqi will significantly benefit them once our study is put into use.

- **Computer Vision**

  For the current project, we are going to use Aruco visual library [8] for object and position retrieving. Aruco is a built-in library for OpenCV, which can detect the moving set of markers in the live feed. We will assign four markers to four corners of the chessboard, and calculate the perspective transformation. We apply the transformation to each chess with a particular id to identify their position and role, thus retrieve the game state.

- **Chess Logic**

For the backend logic, we will implement a suitable backend logic including the learning and search algorithm of the XiangQi. Previous studies have done well with state checking and pruning. We will check them [6] as a reference and implement our project. Also, error alerts should be implemented if the player gives out the wrong step.

Game	Space complexity	Tree complexity	Branching factor
Western Chess	50	123	35
Xiangqi	**48**	**150**	**38**
Go	160	400	250

Table 1 [7]: The properties of Xiangqi comparing with other chess games, we can notice that it has similar properties with Western Chess, which does not require high concern of algorithm study, thus allowing implementation easier

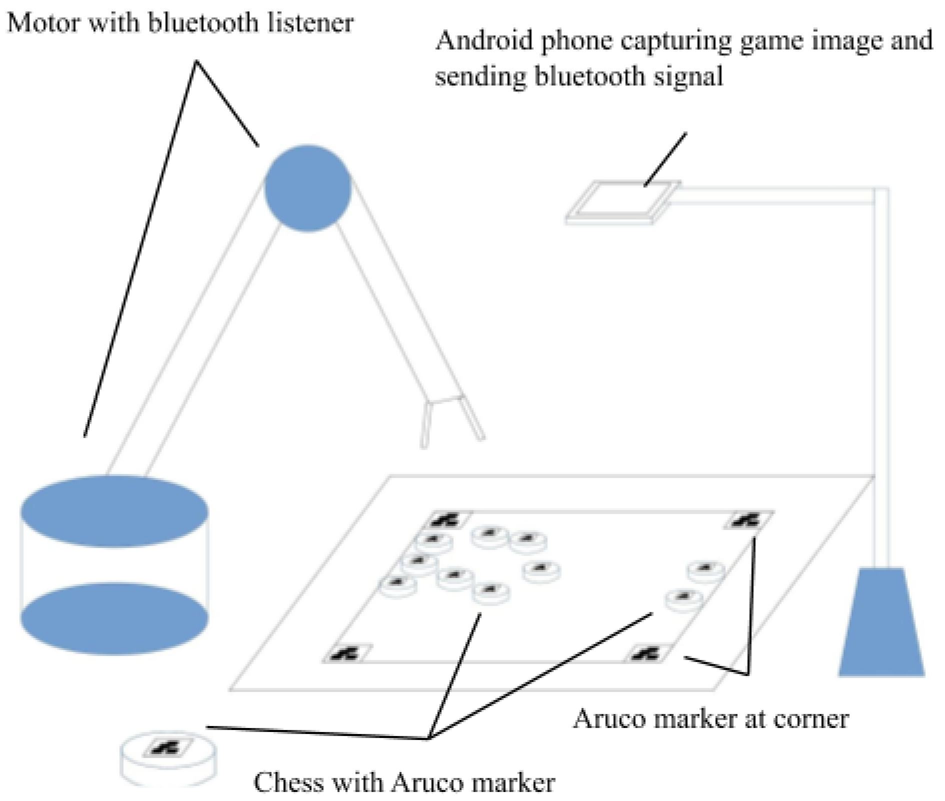
- **Robotic Arm**

A connection with Bluetooth will be an easy way to control and communicate. The robotic arm will be provided after the earlier stage is accomplished. An Aruco code will be attached on the Robotic Arm to adjust and control the picker position

- **Android Development**

And considering the complexity that microelectronics involved, utilize of Android phones will be an ideal integration of sensors and microcircuit as the central control unit, which is easy to control and implement via coding method and thus more convenient for development than the previous project [4]. Also, phones have a smaller scale and are easy to carry, it will arouse more interest in users, and make it possible to establish its usage and robotic concept into the public.

Project Methodology


Motor with bluetooth listener
Android phone capturing game image and sending bluetooth signal
Aruco marker at corner
Chess with Aruco marker

- **Environment Set-Up**

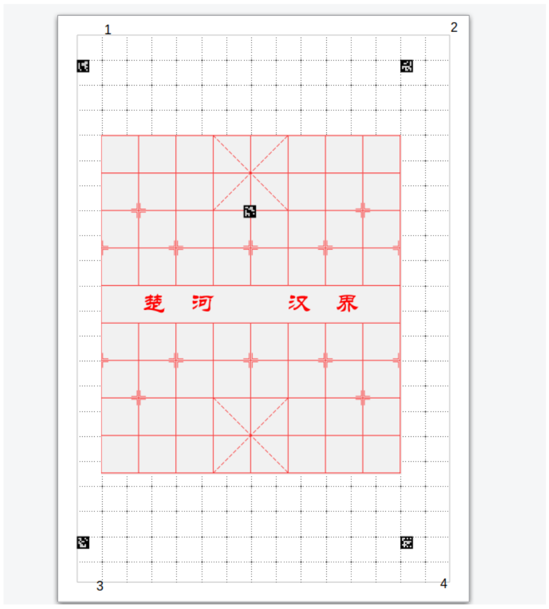
  Python 3.7.3
  OpenCV 4.1.1
  Android 9 (Pie) /10 (Q)
  Flask 1.0.2

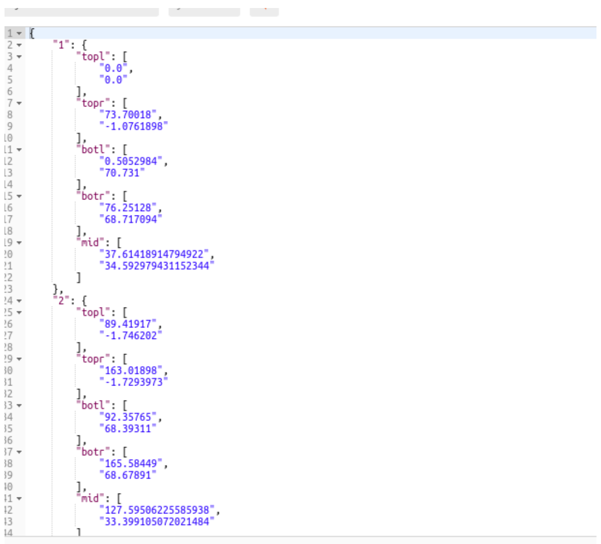
- **Variables**

  Chess number: 32
  Aruco marker number: 37

Chessboard ratio: 9:8
Game State Matrix: 10*9

- **Current preparation for the plan**




```
{
    "1": {
        "topl": [
            "0.0",
            "0.0"
        ],
        "topr": [
            "73.70018",
            "-1.0761898"
        ],
        "botl": [
            "0.5052984",
            "70.731"
        ],
        "botr": [
            "76.25128",
            "68.717094"
        ],
        "mid": [
            "37.61418914794922",
            "34.592979431152344"
        ]
    },
    "2": {
        "topl": [
            "89.41917",
            "-1.746202"
        ],
        "topr": [
            "163.01898",
            "-1.7293973"
        ],
        "botl": [
            "92.35765",
            "68.39311"
        ],
        "botr": [
            "165.58449",
            "68.67891"
        ],
        "mid": [
            "127.59506225585938",
            "33.399105072021484"
        ]
```

*Picture 1 & 2: Example chessboard picture for aruco functional test with Aruco 6*6 library. We put Aruco image processing on server and test for 1. id and position recognition; & 2. Time of server's respond (20ms for localhost)*

- **Stage 1: chessboard recognition**

The chessboard and chess will be prepared with Aruco code. The Android phone will be able to record the video and upload it to the server and corresponding API. Python server will be built up, and able to handle the video stream, then return the current information of the chessboard. Additionally, implementing the auto-detection of the chess movement will be expected. We will temporarily display the game state on phone for debugging use.

- **Stage 2: Connection to the robotic arm**

We will do the pre- and post-processing of game data to make it fit with the referenced project [6]. For the result, it will be added to the server and can indicate the next step for the robotic arm. As Android receives the instruction, it will detect the position of the robotic arm and send out Bluetooth signals accordingly. The robotic arm will be able to move accordingly.

Project Schedule and Milestones

Sep 1 to Sep 30	The first deliverable of the project
Oct 1 to Oct 30	The server will be built up and tested with the picture, which is taken from real chess and chessboard with corresponding Aruco marker. It should return the game state correctly. At last, it should accept the video stream.
Nov 1 to Nov 30	Android Application build-up, test on the real phone
Dec 1 to Jan 30	Test the robotic arm with Bluetooth signal, implement backend logic, we will use human arm instead of robotic arm for mock testing, the first presentation
Feb 1 to Feb 30	Robotic arm implementation, build server for arm position detection, the interim deliverable
Mar 1 to Mar 30	Adjust and debug
Apr 1 to Apr 30	Optimization, the final deliverable

References

[1] Ashok K. Hemal & Mani Menon. Robotics in Genitourinary Surgery. Springer. p. 4., 2018.

[2] M. Campbell, A. J. Hoane, and F. Hsu. Deep Blue. Artificial Intelligence, 134:57–83, 2002.

[3] AlphaGo: The story so far. Retrieved from https://deepmind.com/research/case-studies/alphago-the-story-so-far.

[4] Instructables. (2017, October 12). Chess Robot. Retrieved from https://www.instructables.com/id/Chess-Robot/.

[5] White Paper for Xiangqi. (2018 edition). p14. Retrieved from https://www.xqinenglish.com/Download01/象棋白皮书%202017版%20珠江文化.pdf.

[6] https://github.com/huygithub/hoxChess.

[7] AI Agent For Chinese Chess. D. Li. Retrieved from http://stanford.edu/~dengl11/resource/doc/221-Report.pdf.

[8] Aruco: a minimal library for Augmented Reality applications based on OpenCV. Retrieved from: https://www.uco.es/investiga/grupos/ava/node/26.