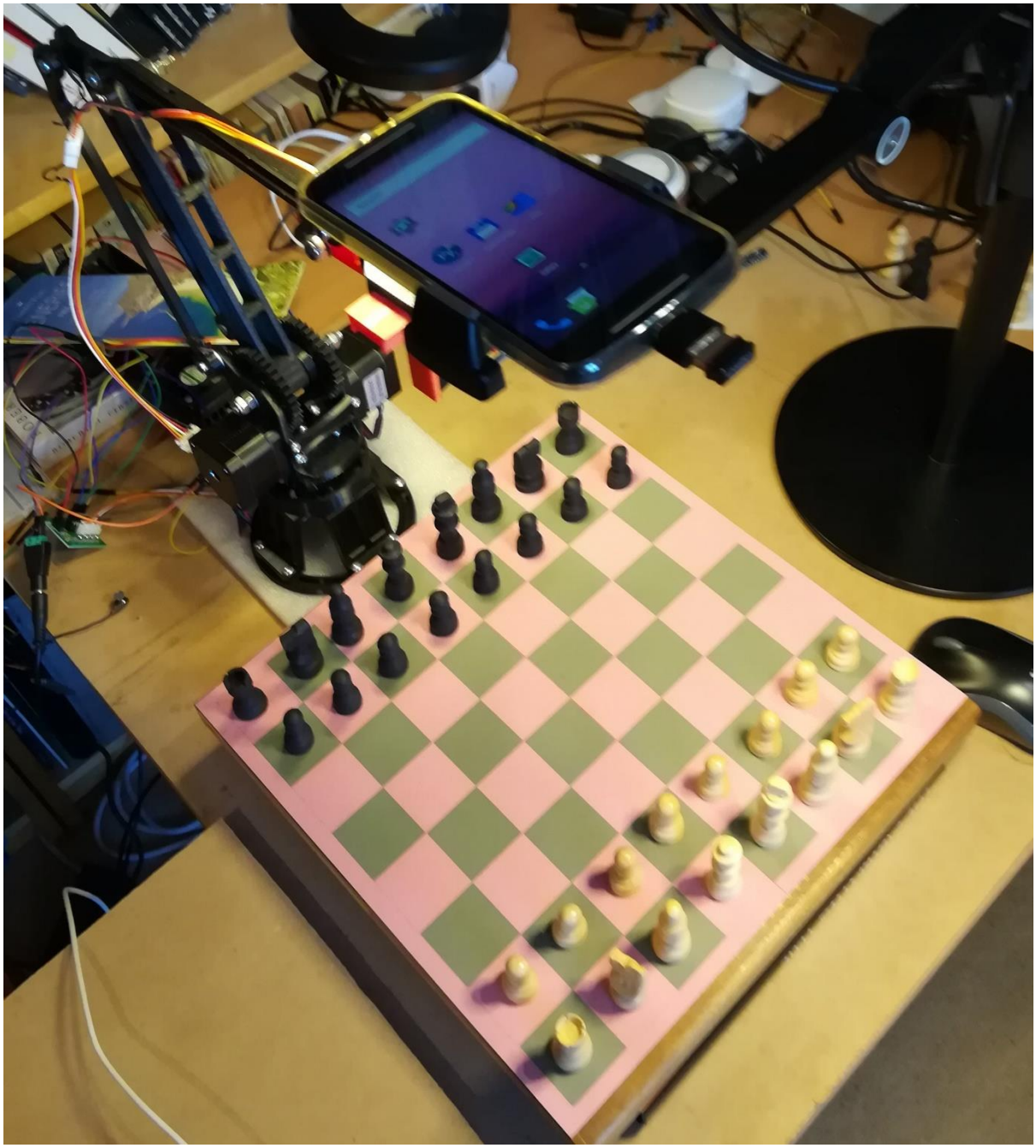

Introduction

This chess robot can beat any human!

I previously made a similar 3D-printed chess robot which makes use of a Raspberry Pi or Windows. That version needs a USB camera, a display screen and keyboard as well as an RPi or Windows processor. Whereas the version described here just needs a discarded Android phone. The phone's camera, microphone and touch screen are used, and the phone is connected to an Arduino via Bluetooth - so no wires come out of the phone.

Supplies

3D printer

Stepper motor NEMA 17 (x3)

Stepper motor driver board A4988 (x3)

Stepper motor 28BYJ-48 (for grabber)

Stepper driver board ULN2002

Arduino Mega 2560

Ramps 1.4

USB lights (x2)

Spare Android phone (!)

USB speaker

Bluetooth module HC-05

Stand for lights and phone

Wireless mouse and dongle (for chessboard calibration)

Micro USB (or USB-C) OTG to USB Adapter (ditto)

Power supplies

Requirements

The chess-playing code is written in Python and runs on an Android phone. The code for inverse kinematics and driving the motors runs on an Arduino Mega 2560. A Ramps 1.4 board is attached to the Arduino

The 3D-printed robot is built using STL files.

You will also need a table, lighting and something to hold the phone above the board. And of course, chess pieces and a chessboard. I describe all these things in more detail in the subsequent steps

How It Works

It works like this:

The human, playing white, makes a move. This is detected by the visual recognition code on the phone. The robot then ponders and then makes its move.

And so on...

Perhaps the most novel thing in this robot is the code for move recognition.

Because the human's move is recognised by a vision system, no special chess board hardware (such as reed switches, or whatever) is needed.

The Hardware Build

The 3D printer files for the robot are freely available as specified by ftobler here. Modifications for a longer arm are here. This is required in order to reach the far corners of the chess board.

We built the cartesian robot, but the SCARA robot should also work.

In the previous builds we used a special mini-gripper, but this time we used a gripper based on the original ftobler gripper but we modified the files to have longer and thinner fingers, which makes it more suitable for picking up chess pieces. All the STL files for the gripper are here. The mini-gripper worked with a mini servo motor, but this time we followed ftobler in using a 28BYJ-48 stepper motor.

The stepper motors give very high precision and repeatability.

However, there is a problem with the gripper stepper motor in that when it grips it thinks it has moved to the target position when it hasn't. So when the gripper is opened again it opens to far, and then when closed it doesn't close enough. This is resolved by putting in two screws as endstops for the open position.

The code supports Bluetooth (not BLE), using an HC-05.

The RX output from the HC-05 is connected via a voltage divider to D16 on AUX-4.

The TX output is connected directly to D17 on AUX-4.

https://osoyoo.com/2016/07/03/reprap-3d-printer-circuit-connection-graph/

Voltage divider example (Ignore connections to Arduino):

https://electronics.stackexchange.com/questions/280500/why-do-you-have-to-use-a-voltage-divider-with-hc-05-bluetooth-module-arduino

The Software Which Moves the Robot

All the Android code is written in Python 3, and runs on Pydroid 3. The Python files, together with instructions for installing and running can be found here. (This includes instructions for installing Pydroid 3).

So, we then have code which will move pieces, take pieces, castle, support en passant, and so on.

There is a choice of chess engines. If your phone is running Android 7, or if it is rooted, then Stockfish can be used.

Stockfish can beat any human! It is one of the strongest chess engines in the world. It is also much stronger than the best human chess grandmasters.

If your phone is not rooted and it is running a version of Android higher than 7, then Sunfish can be used. It is a simple chess engine - however, it plays at ratings above 2000 at Lichess.

I use some code from chessfortherapy.co.uk to validate the human's move and interact with the chess engine. My code for recognising the human's move and moving the robot arm interfaces with that.

On the Arduino, inverse kinematics code is used in order to move the various motors correctly such that chess pieces can be moved. This code is based on code which ftobler wrote, and can be found via the link above.

The Software for Recognising the Human's Move

This is similar to the code that I wrote for my previous Cartesian chess robot so please see the description there.

Other considerations

The algorithms work best if the chessboard has a colour that is a long way from the colour of the pieces! In my robot, the pieces are off-white and matt black, and the chess board is hand-made using a colour printer onto thin card. It can be seen in the video.

The chessboard should well-lit and evenly lit with minimal shadows from the chess pieces. Light should not be reflected back into the camera from the board or pieces. A sturdy table is needed.

Instructions for calibrating the chessboard and the robot can be found via the first link in the previous section.