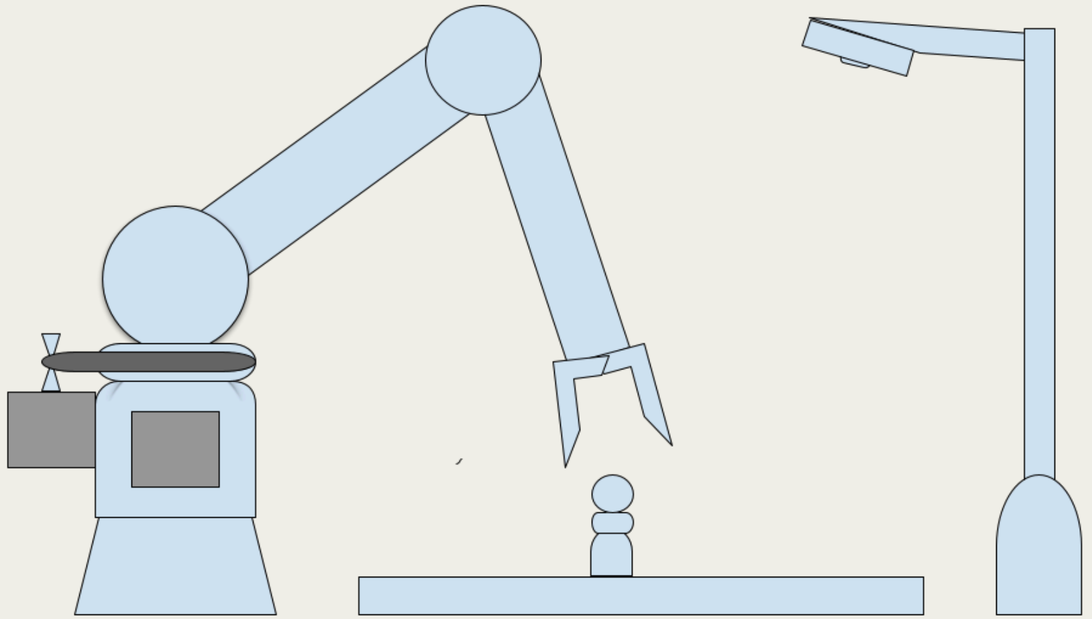
Robotic Arm

DARION KWASNITZA

MY GOAL

To create a robotic arm with a controller capable of fully autonomously playing chess against a human opponent

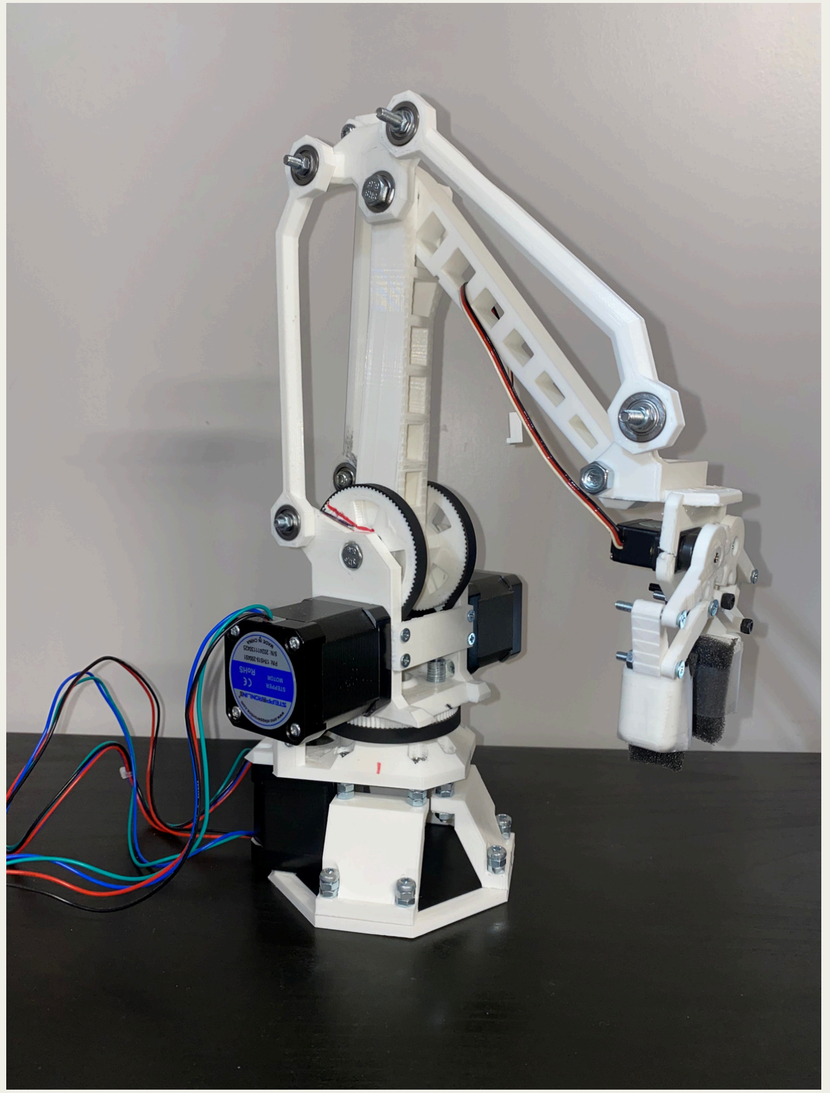


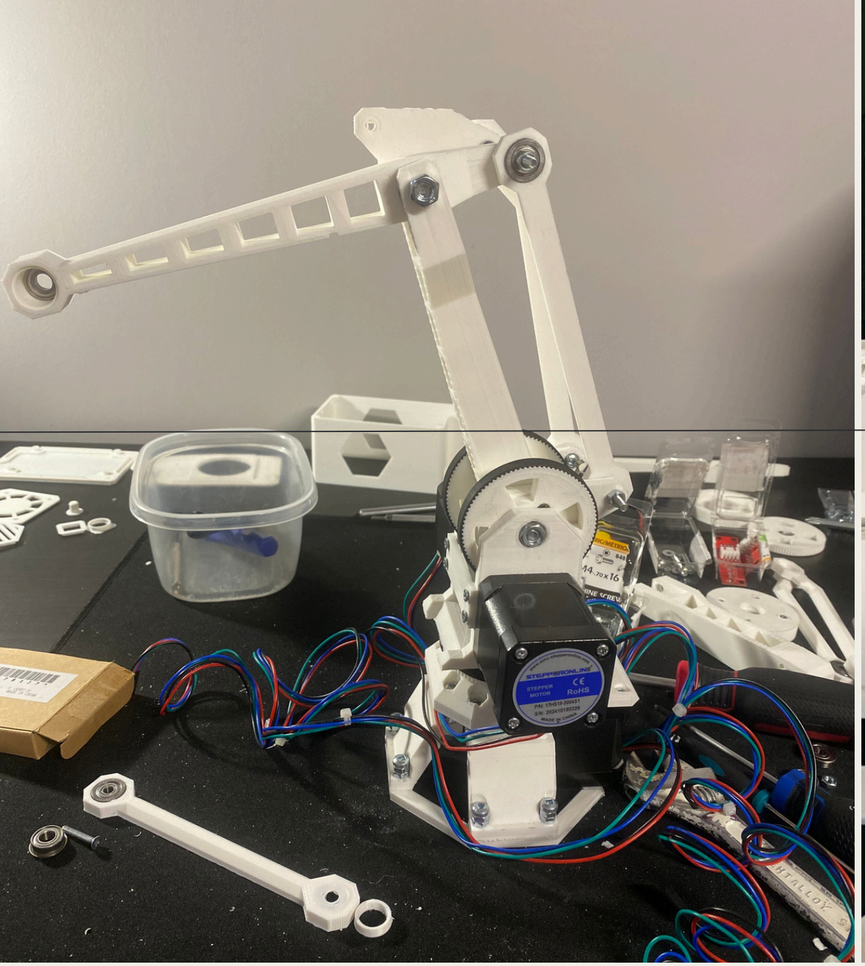
HARDWARE: ROBOTIC ARM

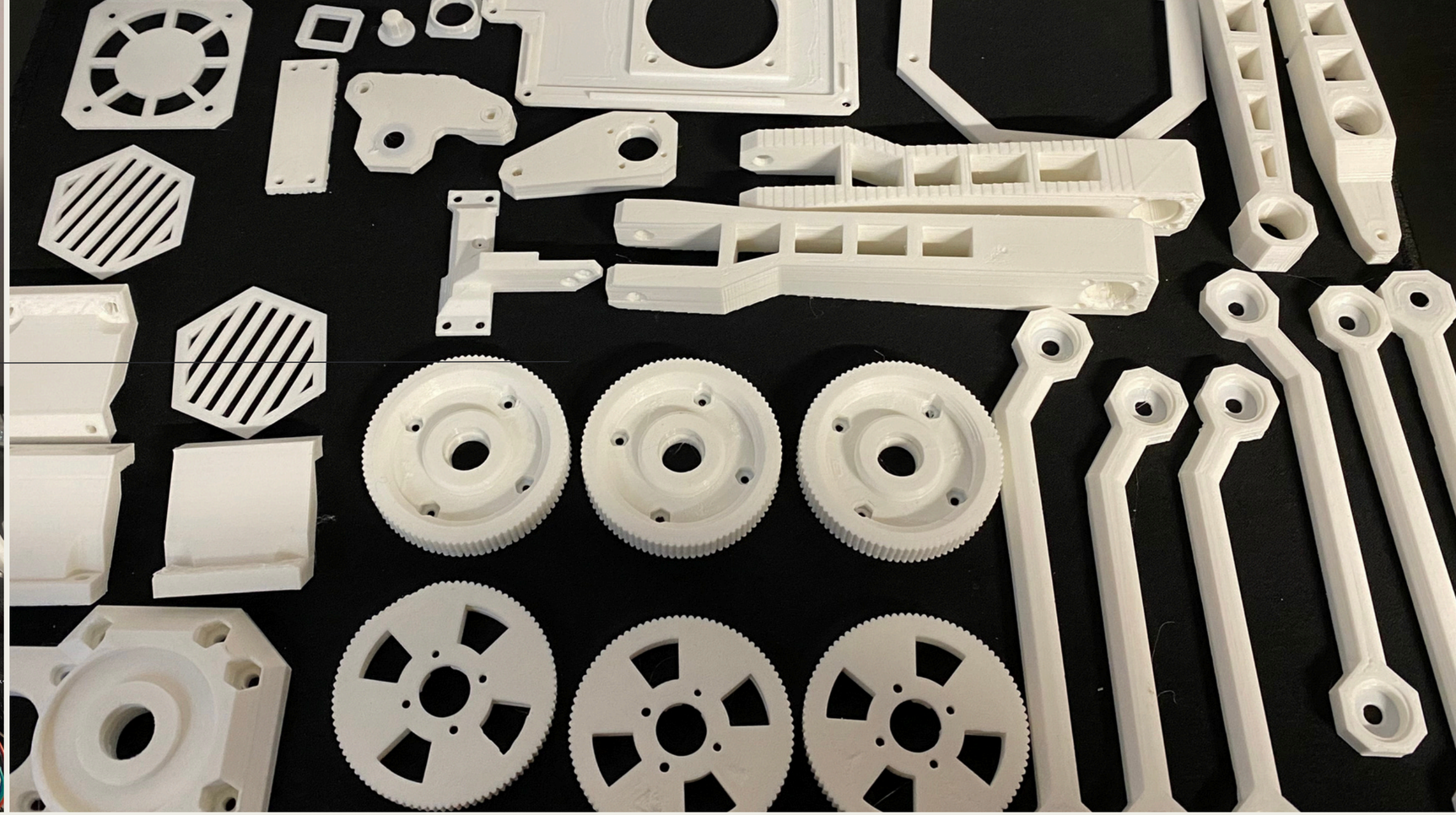
- Claw Style
  - Does not require frame
  - Versatility (can get to 3D point)

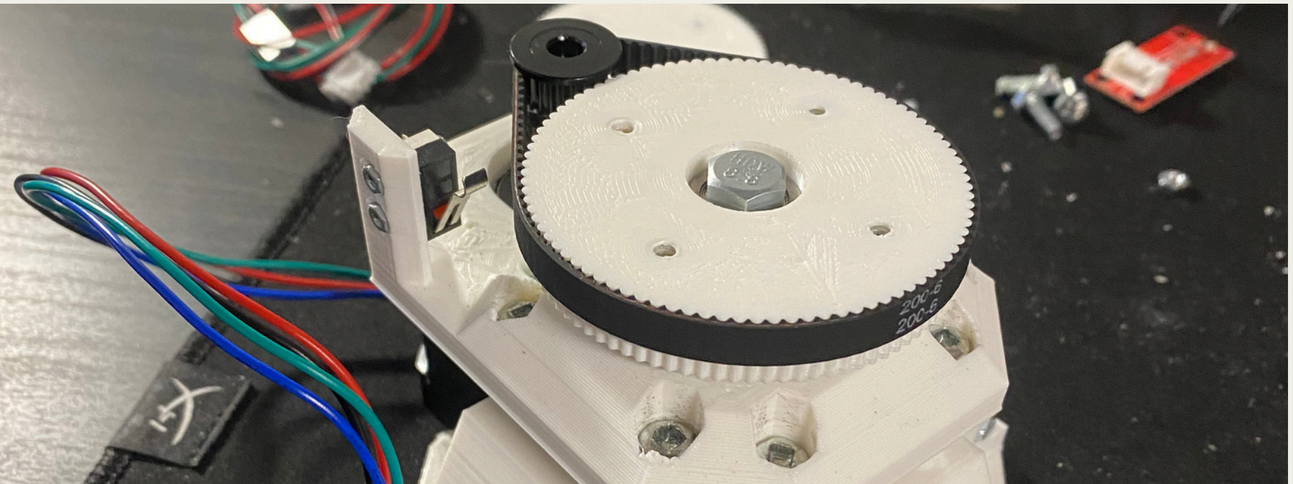
- 3 Motors
  - Simplicity (3 joints)
  - Compatible with chosen NEMA 17 Stepper Motors
  - Sufficient torque to pickup a piece at the farthest end of the board

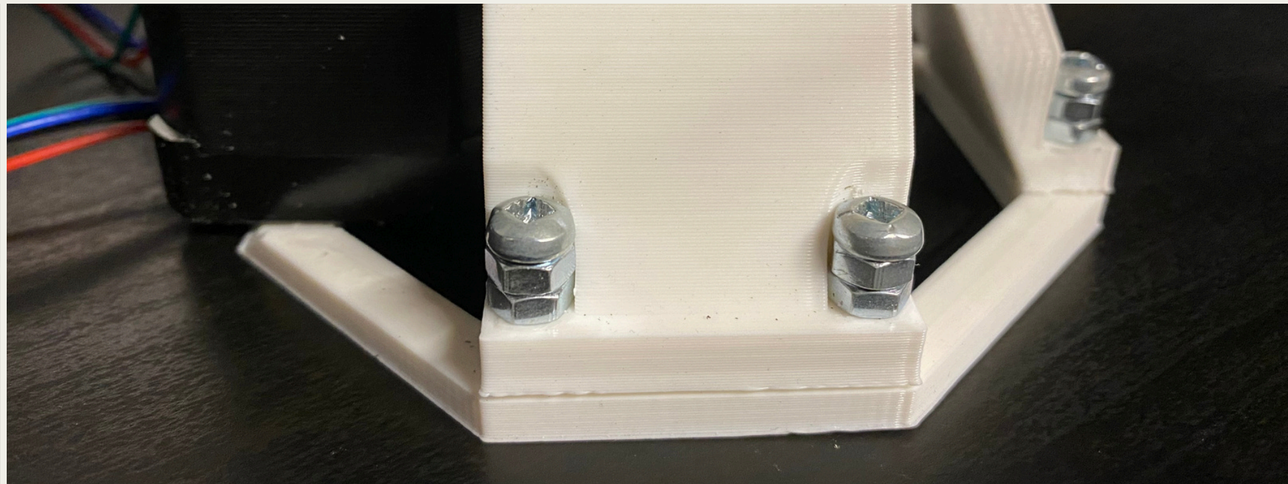
- Ftobler’s Design (extended the arms)





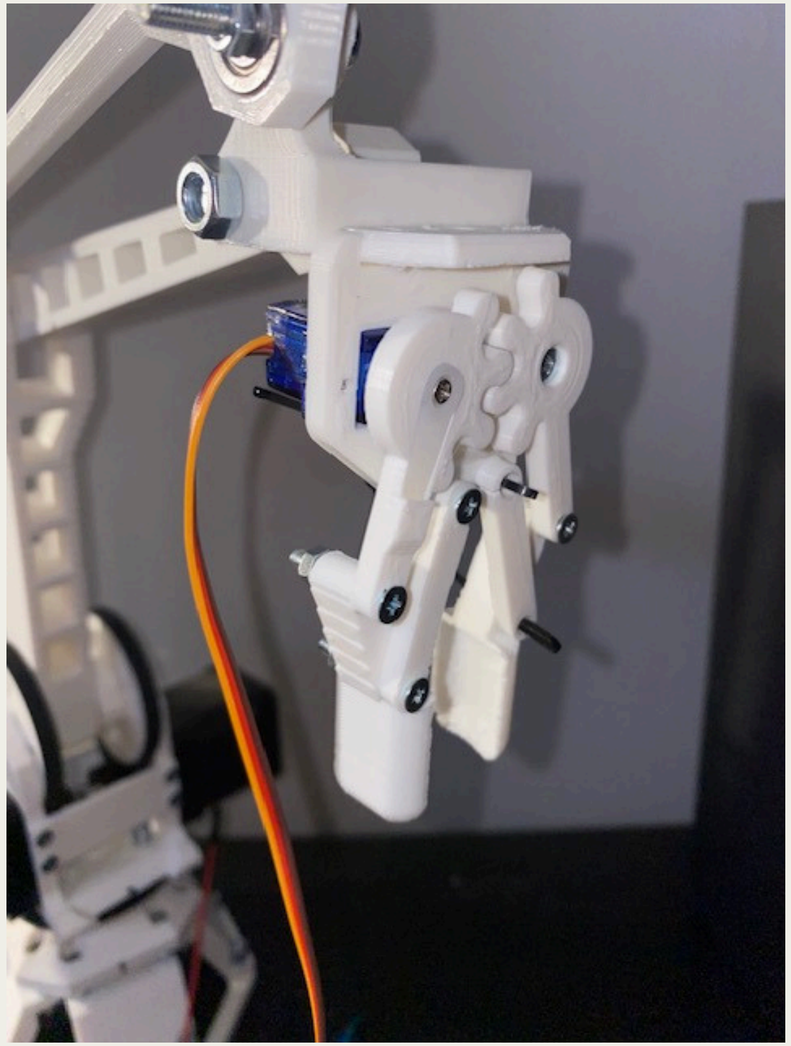






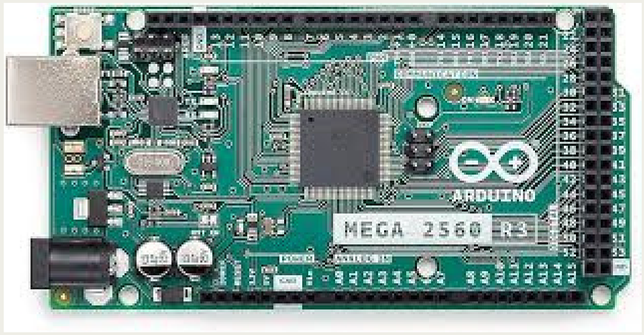
HARDWARE: GRIPPER

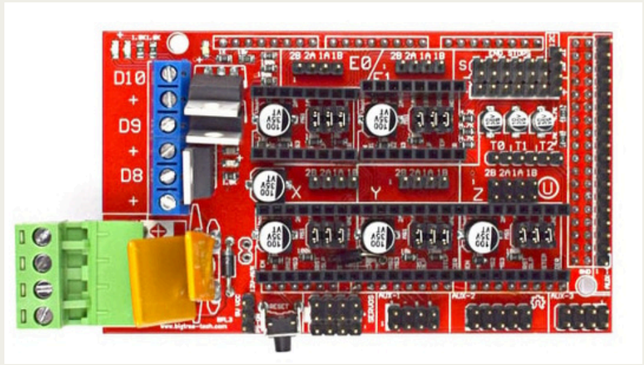
- Pincher Style
  - Both fingers close together in parallel
  - Less chance of tipping over a piece
- Size
  - Sufficient length to pick up the biggest piece from close to the bottom without interference on the top
- 1 Motor
  - Simple SG90 Servo Motor
- Hkucs Design

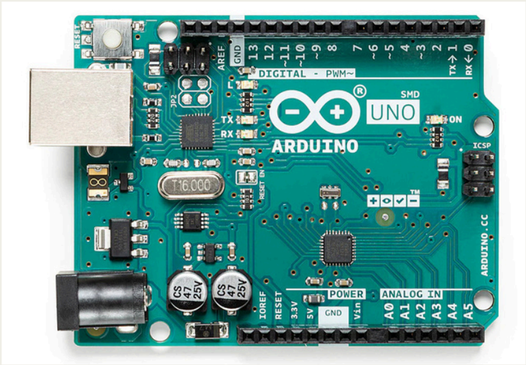


HARDWARE: CONTROLLERS

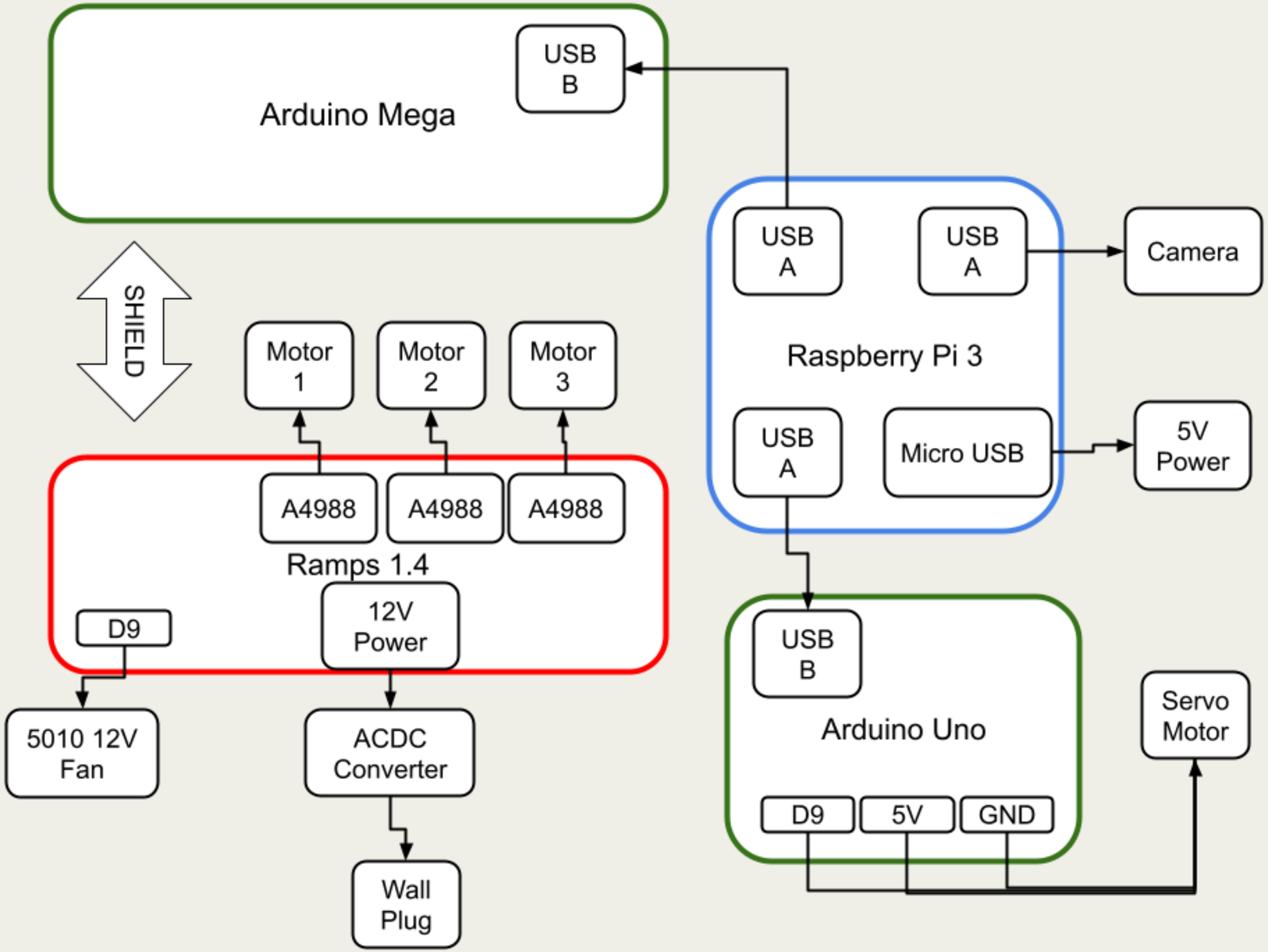
- Arduino MEGA 2560
  - Motor code
  - Other external components code (fan)
- Ramps 1.4 Shield
  - Space for 5 motor drivers
  - Space for other components (fan, endstops, external power)
  - Compatibility with microstepping
- Arduino UNO
  - Gripper code
- Raspberry Pi Model 3 B
  - Chess logic code
  - Object Detection code





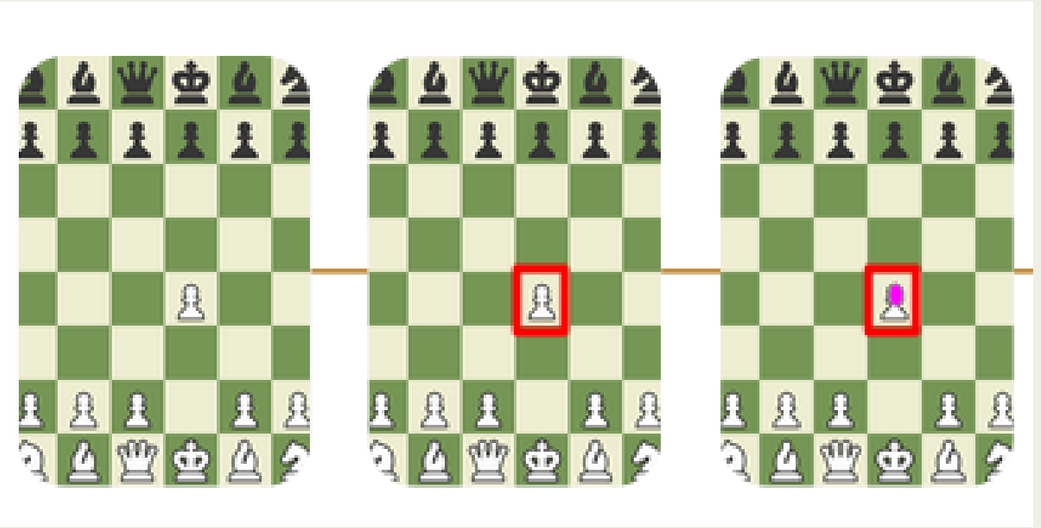


Circuit Diagram



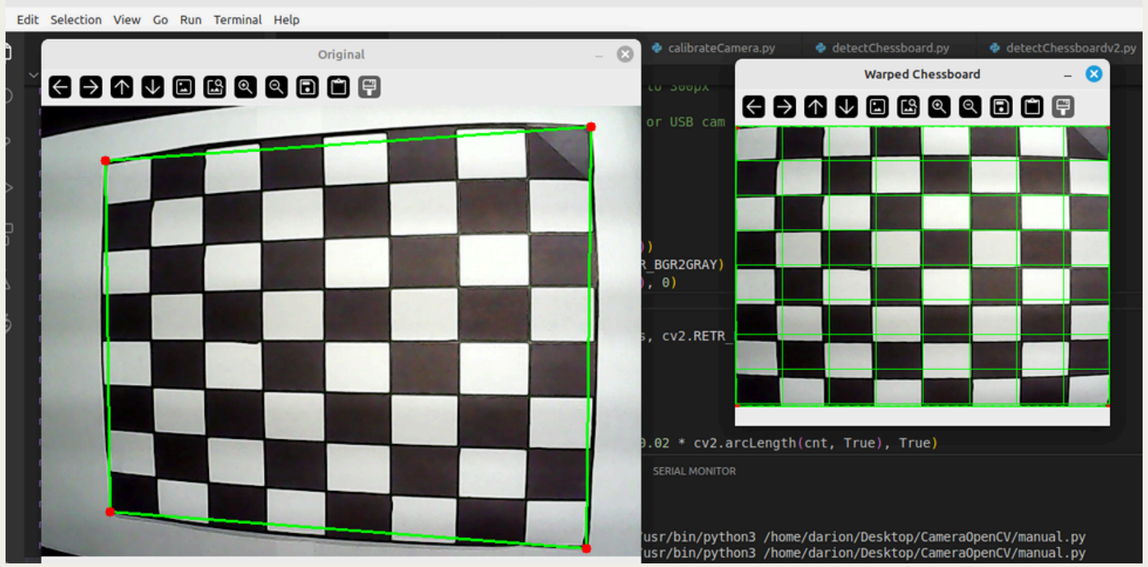
SOFTWARE: OBJECT DETECTION

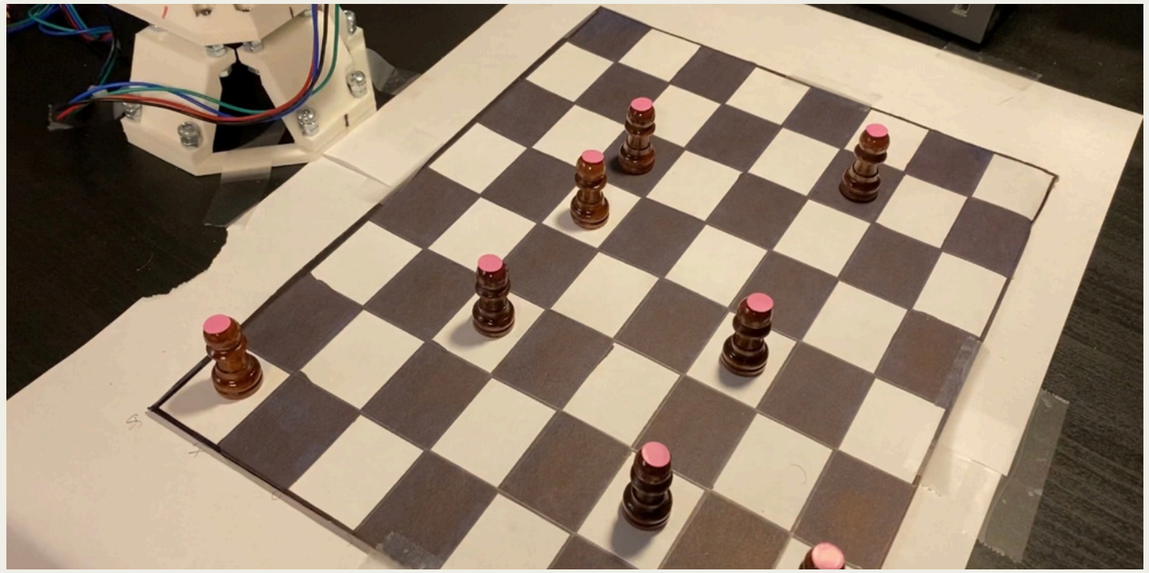
- OpenCV Python
  - Raspberry Pi’s connected camera
- Chessboard Detection
  - Find the largest contour in live video stream
  - Warp perspective and apply squares
  - Automatic mode and Manual mode
- Colour Detection
  - Label each piece type with a different colour
  - Apply HSV masks to board
  - If colour detected, determine which square and which piece
- Assemble FEN String

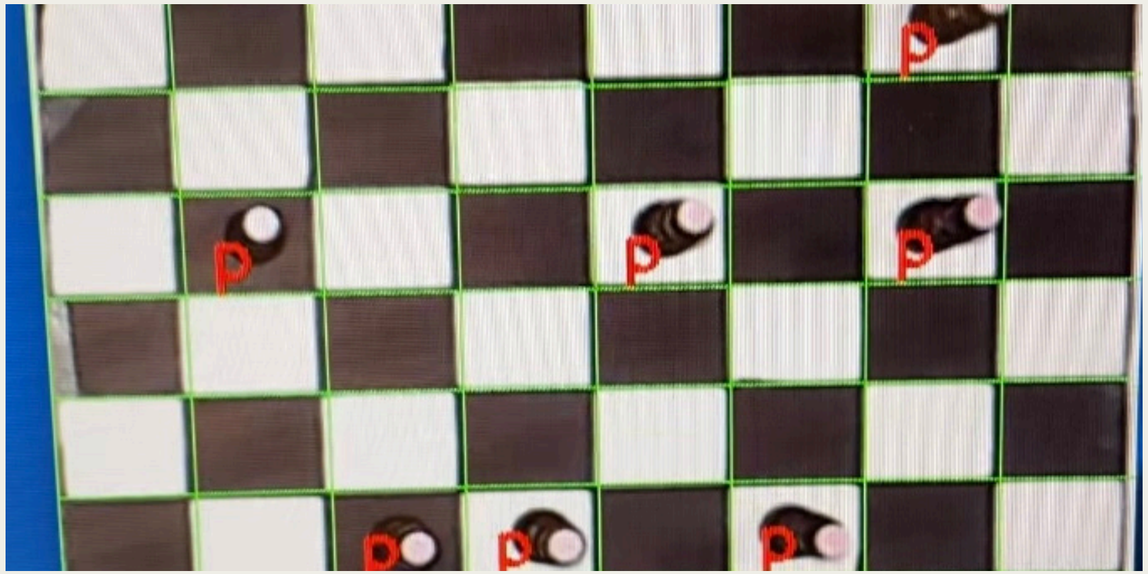


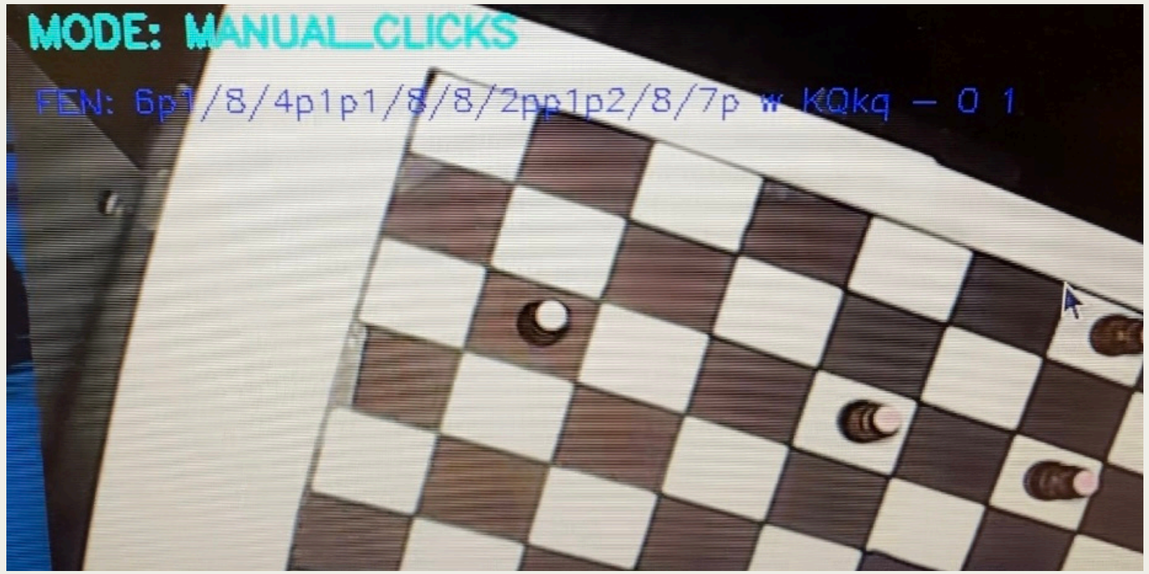
‘P’

SOFTWARE: OBJECT DETECTION









SOFTWARE: CHESS LOGIC

- Stockfish Chess Engine Python
  - Given an FEN string, returns the best move
  - Best move in 4 length string format (“e5e7”)
  - Best move determined by depth of search and difficulty level

“rnbqkbnr/pppp1ppp/8/4p3/6P1/5P2/PPPPP2P/RNBQKBNR b KQkq - 0 2"

SOFTWARE: ARDUINO CODE

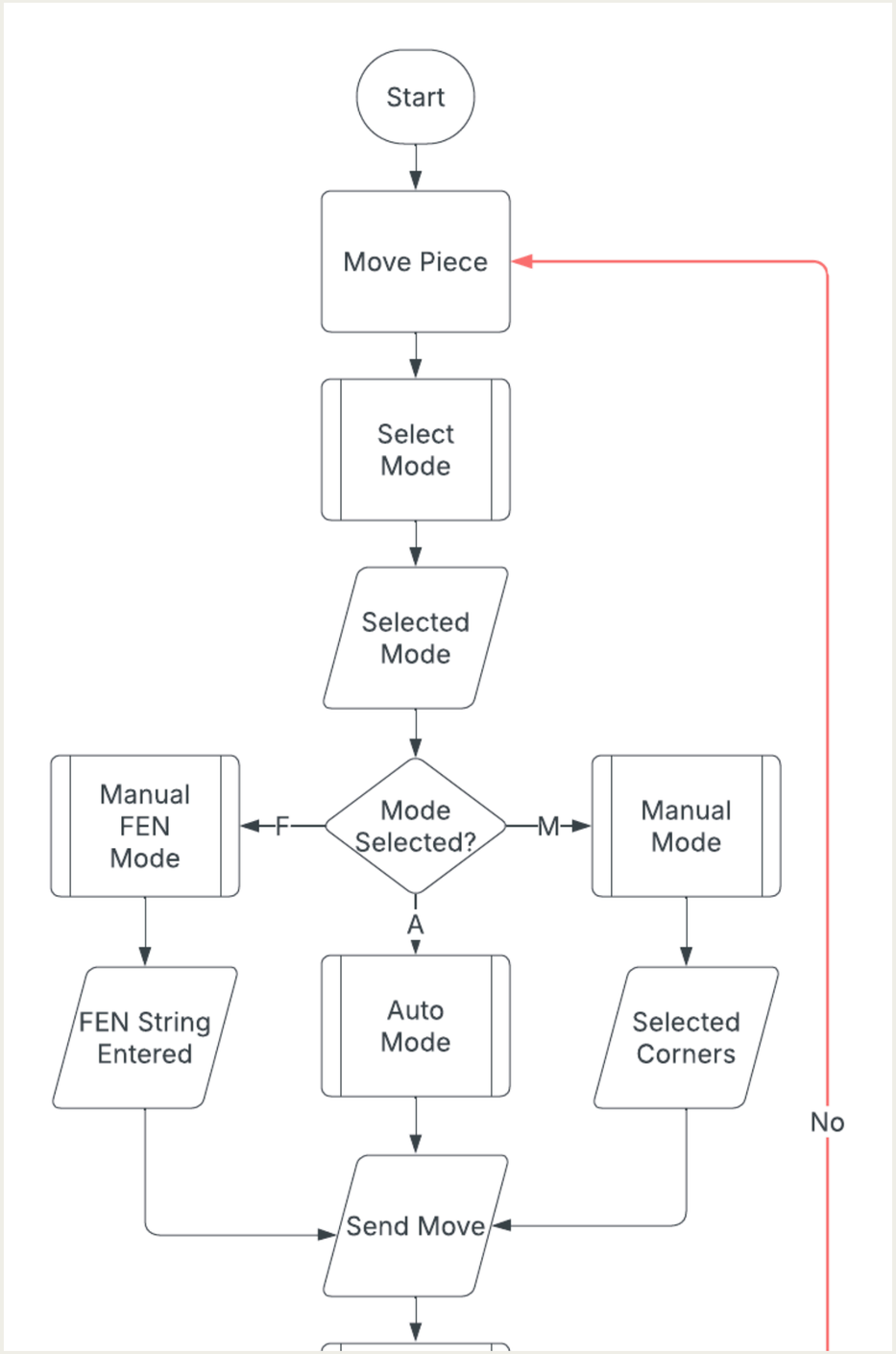
- Arm Movement
  - AccelStepper Library C++
  - Receives best move from Stockfish via serial communication
  - Breaks down move string into from and to squares
  - Move the arm based on angles calculated using Newton's Method (numerical way to solve the arm's inverse kinematics equations by starting with an initial guess and improving it step by step)
- Gripper Movement
  - Serial communication from MEGA to UNO ("READY_TO_GRIP", "GRIP")

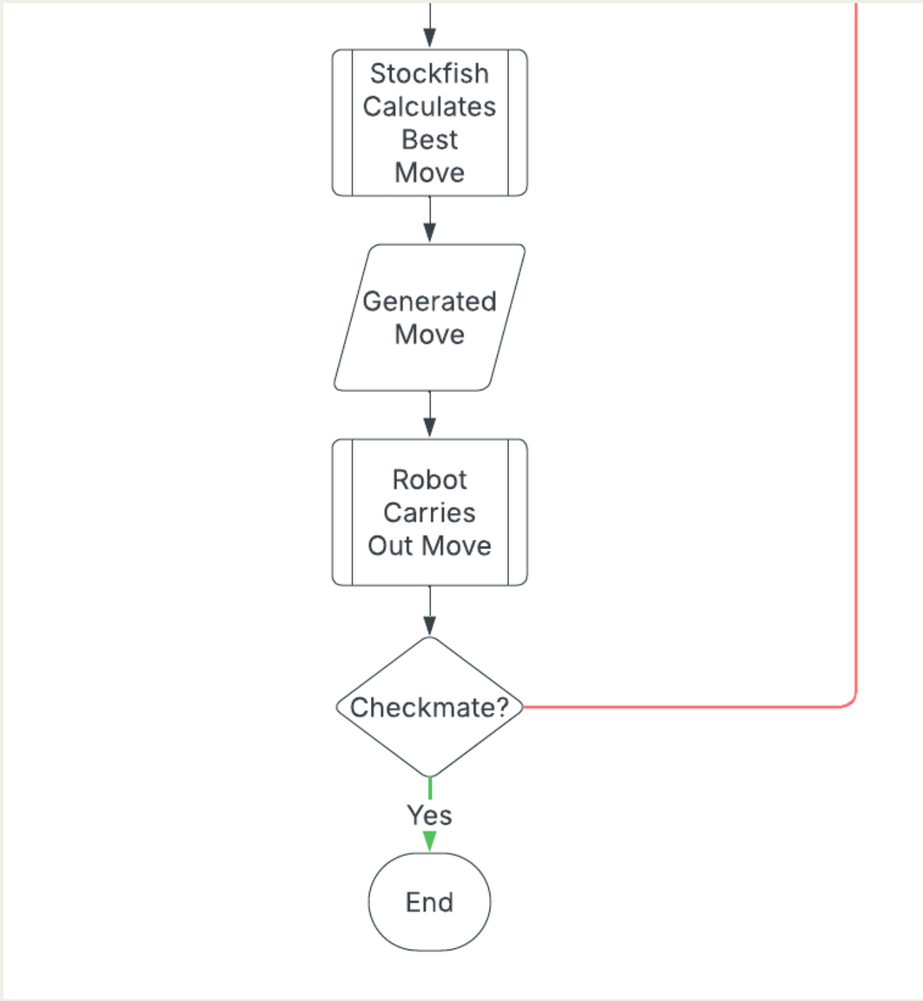
"e5e7" = from E5 to E7

E5 {thetaNot, theta1, theta2}
E5D {thetaNot, theta1, theta2}
E7 {thetaNot, theta1, theta2}
E7D {thetaNot, theta1, theta2}

Home → E5 → E5D → GRIP → E5 → E7 → E7D → RELEASE → E7 → Home

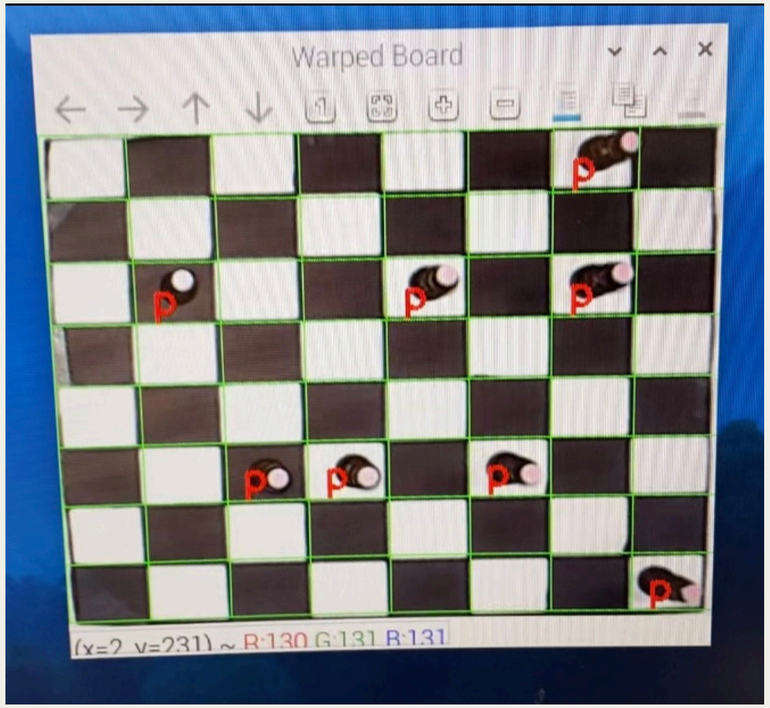
FlowChart

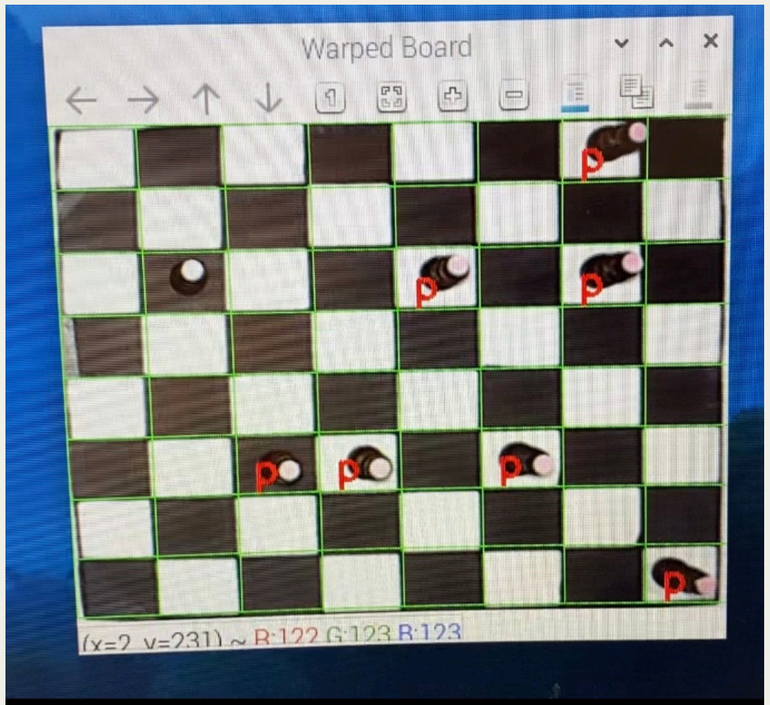

Start
Move Piece
Select Mode
Selected Mode
Mode Selected?
F
M
A
Manual FEN Mode
Manual Mode
FEN String Entered
Auto Mode
Selected Corners
Send Move
No


Stockfish Calculates Best Move
Generated Move
Robot Carries Out Move
Checkmate?
Yes
End

RESULTS: TESTING

Description	Pass/ Fail	Notes
Object Detection	Fail	Instances of sucess, but cannot detect autonomously
Chess Logic	Pass	Stockfish generates best move
Movement	Pass	1 Method good, human error bad! 2 Limited in far corners of board



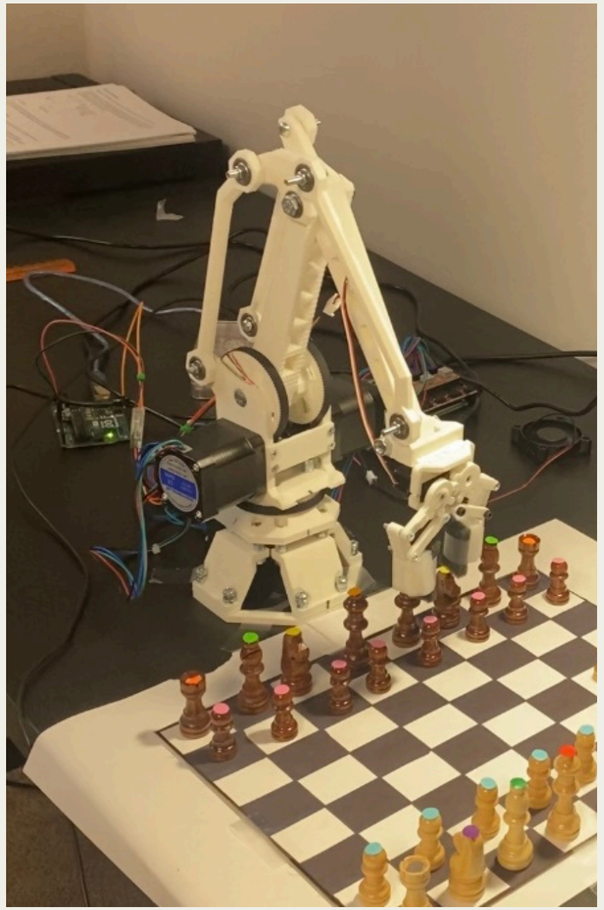


Auto Mode

CONCLUSION AND FUTURE WORK

- Conclusion:
  - The goal of this project was to create a robotic arm and controller capable of fully autonomously playing chess
  - Although there are areas of success, this goal was not fully met

- Future Work:
  - Lots to do!
  - Make gripper smaller
  - 3D print attached chessboard
  - Make a better homing sequence
  - Improve object detection



ACKNOWLEDGEMENTS

Thank you to Dr. Jeffrey Davis for supervising my project!

Thank you!

ANY QUESTIONS?