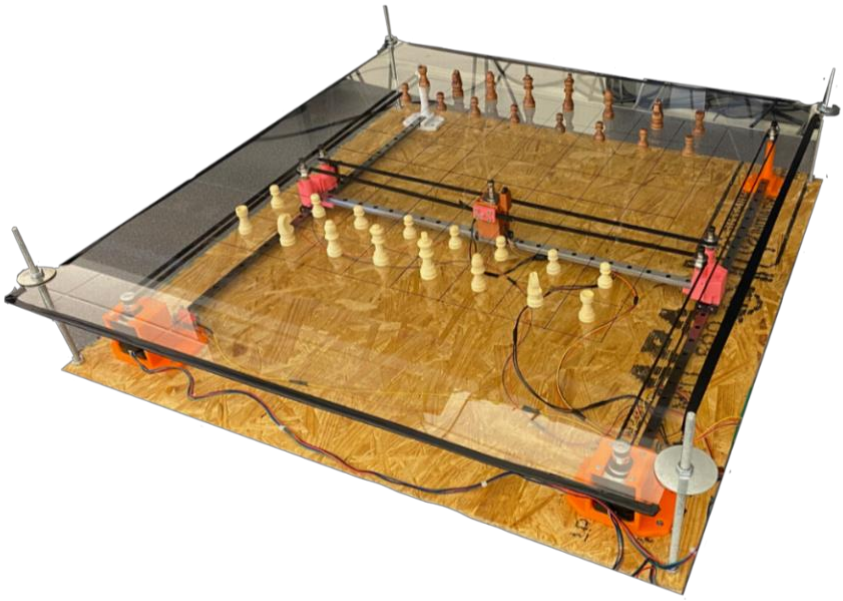


CHESSBOT

ME507 Fall 2022 Project Report

Abstract

The report documents the design and build of a robotically operated chessboard, that is controlled by an online website. The ChessBot successfully replicated early-game moves (such as simple moves and take moves), but not promotion moves. The novel aspect to this chess robot is the infrared sensing method which is enabled by the glass chessboard.

Scott Dunn, Sam Hudson, Dylan Ruiz

Table of Contents

1. List of Figures ***2***

2. List of Tables ***2***

3. Introduction ***3***

3.1. Background Research **3**

4. Specifications ***4***

5. Design Development ***5***

5.1. Hardware Design **5**

5.1.1. PCB Design 5

5.1.2. Robot Design 6

5.2. Software Design **8**

5.2.1. ChessBot Controller 8

5.2.2. ChessBot Server 15

6. Results ***16***

7. Appendices ***19***

7.1. Code Documentation and References **19**

7.2 Kinematics Calculations **19**

1. List of Figures

Figure 1: Board Design 5
Figure 2: Assembled board 6
Figure 3: H-Bot vertical and horizontal motion 7
Figure 4: Isometric view of the ChessBot 8
Figure 5: Diagram of objects that are instantiated in main 9
Figure 6: Dependency injection for the FetchMove task in main 9
Figure 7: FreeRTOS task definition for the controller, in main 9
Figure 8: Task Diagram 11
Figure 9: Controller Task FSM Diagram 12
Figure 10: Motor Task FSM Diagram 13
Figure 11: Limit Switch Task FSM Diagram 14
Figure 12: Fetch Move Task FSM Diagram 15
Figure 13: ChessBot server components and technologies 15
Figure 14: ChessBot server frontend interface 16
Figure 15: Scenario where game piece positions could not be determined by scanning algorithm. 17

2. List of Tables

Table 1: Technical Specifications 4
Table 2: Program object responsibilities 10
Table 3: External code library references 19

3. Introduction

The goal of this engineering project was to design and build a chess robot that could replicate an online chess game to moves on a physical board. The motivation for this project was to enable people with dexterity difficulties to play on a physical chessboard. The hope was that this will help players visualize chess moves better. The report outlines the design and development of the robot.

3.1. Background Research

Previous attempts to develop robotic chessboards can be split into two system architectures:

1. XY cartesian robots with an actuator for grabbing the pieces
2. Robotic arms with an end effector for grabbing the pieces

The first architecture is the easiest and cheapest to implement, given the project time constraints. Furthermore, in recent news a chess robot based on the latter architecture broke the finger of a child, as it mistook the child's finger for a chess piece[1]. The second architecture thus has a greater need for safety protocols, which increases its cost and complexity. Consequently, only the first architecture was researched further.

Commercial products and student research projects have achieved chess robots using the first architecture to various levels of success. Square Off[2] is the most successful chess robot commercial offering. It connects to a smartphone app and uses a linear actuator with a magnet to grab the pieces. However, it costs $400 and has had reported issues with its reliability. Ghost Chess[3] is a student project from the University of Glasgow which achieves a similar result, though uses an electromagnet for grabbing the pieces, and is controlled via a computer GUI. Their robot also uses a matrix of hall effect sensors to detect the placement of chess pieces. The matrix sensing method adds significant complexity to the construction and programming of the system. Josh Eckels[4] successfully created a 2D gantry chess robot with a claw-machine like grabber for the 3rd axis, instead of using linear actuators or electromagnets. The robot successfully moves pieces, but the speed at which it does so is very slow.

[1] https://www.theguardian.com/sport/2022/jul/24/chess-robot-grabs-and-breaks-finger-of-seven-year-old-opponent-moscow
[2] https://squareoffnow.com/product/gks
[3] https://magpi.raspberrypi.com/articles/ghost-chess-electromagnets-move-pieces
[4] https://www.rose-hulman.edu/news/2021/student-creates-robot-that-is-one-move-ahead-on-chess-board.html

4. Specifications

The technical specifications for the system were developed, based on the research done previously.

Table 1: Technical Specifications

Specification	Verification Method	Result
System shall move pieces from one square to another to within 5mm accuracy	Test: measure on final system with calipers	Pass
System shall detect pieces when under them	Test: check sensor calibration LED is on when under the piece	Pass
System shall be controlled through a website that enforces chess rules	Test: test various moves and if they are correctly rejected/accepted and sent to the robot	Pass
System shall cost under $300	Observation: calculate total cost	Fail
System shall have a user-friendly controller website interface	Demonstration: let others use the system and get their feedback	Pass
System shall have a controller website that can be accessed by two users simultaneously	Test: two users playing against each other	Pass
System shall be able to take replicate moves which involve the taking of a piece	Test: set up a piece take move on the controller website and see if it successfully completes the sequence	Pass
System shall use a custom designed and built PCB	Inspection: does the system use a custom designed and built PCB when functioning?	Pass
System shall be able to carry out a full chess game	Test: try complex moves that may occur during a full game	Fail: promotion not implemented

5. Design Development

5.1. Hardware Design

5.1.1. PCB Design

The electronics consisted of an ESP32 microcontroller board, a 12V-5V buck converter, two TMC2208 stepper motor drivers, two NEMA 17 stepper motors, two limit switches, a 12V linear actuator, and a TCRT5000 infrared reflective sensor. The linear actuator was controlled via a PMV16XNR MOSFET. The circuit was designed in EAGLE.

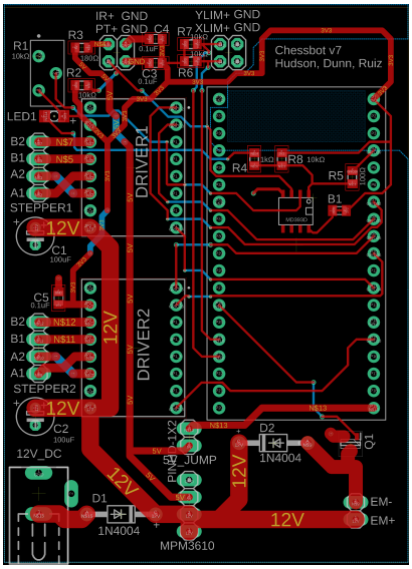


Figure 1: Board Design

After testing the circuit on a bread board, the circuit was sent to be fabricated by JLCPCB, and then assembled.

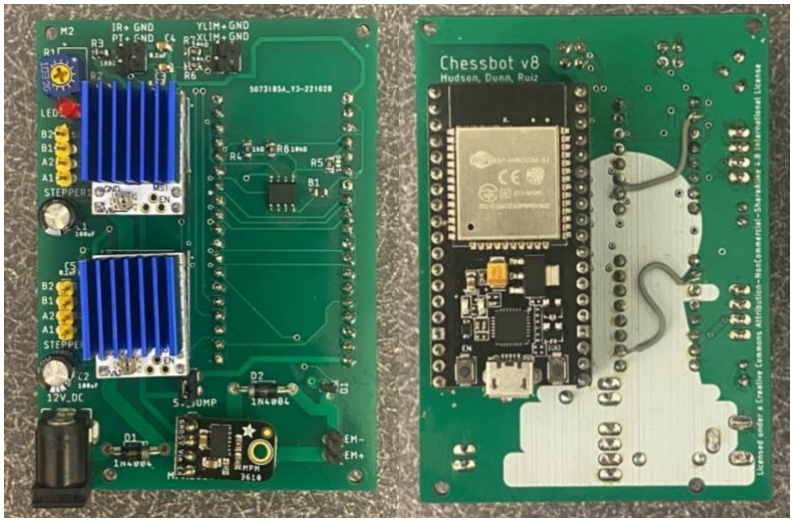


Figure 2: Assembled board

The infrared sensor circuit was taken from Open Impulse[5], who provided the schematic for the TCRT5000 breakout board. When a chess piece is put over the sensor and is detected, the circuit outputs low to a GPIO pin. The sensor can be calibrated using the variable resistor and the LED, which lights up when detecting an object. The external components (the limit switches, infrared sensor, and stepper motors) connect through header pins. A 2.5mm DC jack allows a standard 12V power supply to be used. The stepper motor boards and the ESP32 board was attached to the board with female header pins.

Some design adjustments were needed after testing the board. Since the stepping mode of the drivers was decided after the board had been sent to be fabricated, two jumper wires needed to be attached to set the drivers to the correct stepping mode (1/2 micro steps). The ESP32 board was also mistakenly designed to be placed on the back side of the board. Additionally, the SMD LED initially installed was damaged in the process. Other components had been added around the LED, and so it was difficult to replace. Therefore, a through-hole LED was soldered on top of the SMD pads instead, which worked as an adequate replacement. Lastly, the DC jack was wired for a center negative supply, whereas a center positive supply was purchased. This was fixed by changing the polarity of the supply. Other than these adjustments, the board worked as intended and no further changes were required.

5.1.2. Robot Design

Following the specification that the robot must be able to detect where pieces are placed on the board, the idea to use a glass board and an infrared reflective sensor was proposed. This would result in only one

[5] https://www.openimpulse.com/blog/products-page/product-category/tcrt5000-infrared-sensor-module/

sensor being required, rather than a matrix of sensors as past projects have attempted. The carriage would hold the infrared sensor and reflect off the bottom of pieces when it is under them. Since the board is glass, there will be a measurable difference in received infrared when the sensor is under a piece versus when it is not.

The chess bot manipulates game pieces about the game board from the bottom side using an H-Bot style gantry driven by two NEMA17 stepper motors and structured by three MGN12H linear rails. The H-Bot design was chosen as it reduces the complexity in the design significantly. This is due to the design only requiring a singular belt which is driven by two stepper motors placed on the two parallel linear rails. Thus, a motor does not need to be mounted on the carriage, reducing the load requirements on the rest of the system.

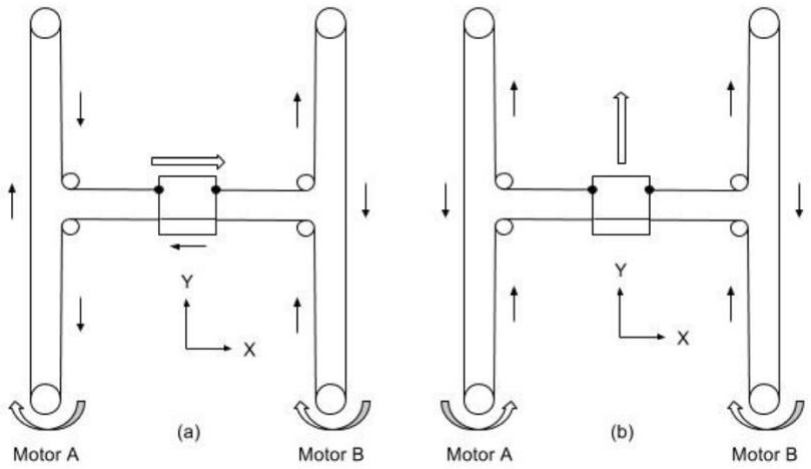


Figure 3: H-Bot vertical and horizontal motion[6]

A linear actuator with a magnet would selectively move semi-ferrous chess pieces when it was in the extended position and underneath a piece. The game pieces slide over the glass board to their commanded position by moving collinear with the grid lines, to maximize the distance between adjacent pieces and avoid the actuator's magnet from affecting idle chess pieces.

[6] https://www.galil.com/download/whitepapers/wp_h-bot.pdf

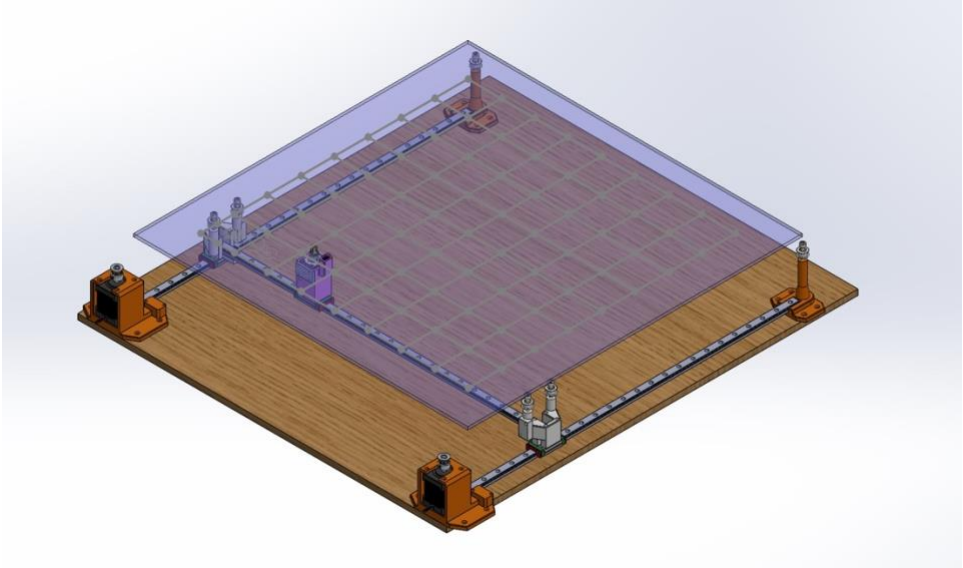


Figure 4: Isometric view of the ChessBot

The CAD software SOLIDWORKS was used to model the full assembly of the project which helped with ordering parts and fabrication. Creality and Prusa 3D printers were utilized through the Cal Poly Mustang '60 shop to quickly create custom mounting parts for linear rails, stepper motors, idler pulleys and the primary actuator.

5.2. Software Design

5.2.1. ChessBot Controller[7]

The complexity of the software necessitated the use of object-oriented programming principles. Dependency injection was used to enable loose coupling between each class. This allowed the team to split the programming work up without adding complexity when programs were integrated together.

[7] The ChessBot controller repository can be found here: https://github.com/Dyruiz131/me507_Term_Project

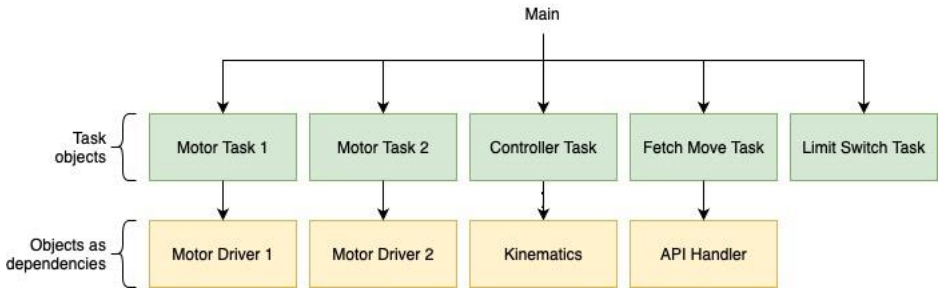


Figure 5: Diagram of objects that are instantiated in main

Figure 5 shows the objects that are instantiated in the main program. The objects on the bottom of the diagram must be instantiated first, as they are dependencies required to instantiate the task objects above. Figure 6 shows an example of this in the main program.

```
// Create fetch move task object using APIHandler object
APIHandler apiHandler(SSLCertificate);
FetchMove fetchMoveTask(apiHandler);
```

Figure 6: Dependency injection for the FetchMove task in main

Each task was made into an object, containing a run method which acted as the finite state machine for the task. In the main program, all dependent objects were instantiated (the motor drivers, kinematics, and API handler). Then, the task objects were instantiated using the dependent objects. Each task object was then defined as a FreeRTOS task with its run method and task period.

```
// Main controller task (FSM)
void defControllerTask(void *p_params)
{
  while (true)
  {
    mainController.run();
    vTaskDelay(100); // Task period
  }
}
```

Figure 7: FreeRTOS task definition for the controller, in main

A clear benefit to this approach can be seen with the motor control method. Since the motors are identical, but must both be operated individually, the use of object-oriented programming allowed two

motor driver objects, and subsequently two motor tasks to be instantiated. This reduced the amount of duplicate code.

Each object has a specific function, intended to abstract away complexity for their parent tasks. The following is a table of each object and task responsibility:

Table 2: Program object responsibilities

Object	Responsibility
Controller Task	Main FSM, control actuator, read IR sensor, coordinate both motor movements
Kinematics	Helper functions for calculating necessary coordinated motor movements
Motor Task	Motor FSM, check for motor commands, move motor.
Motor Driver	Abstraction for parent task. Motor control specific to the TMC2208 stepper driver.
Fetch Move Task	Move Fetching FSM, convert received Chessboard moves to board coordinates, API communication.
API Handler	Manage specific implementation of REST API communication with server
Limit Switch Task	FSM for limit switch checking

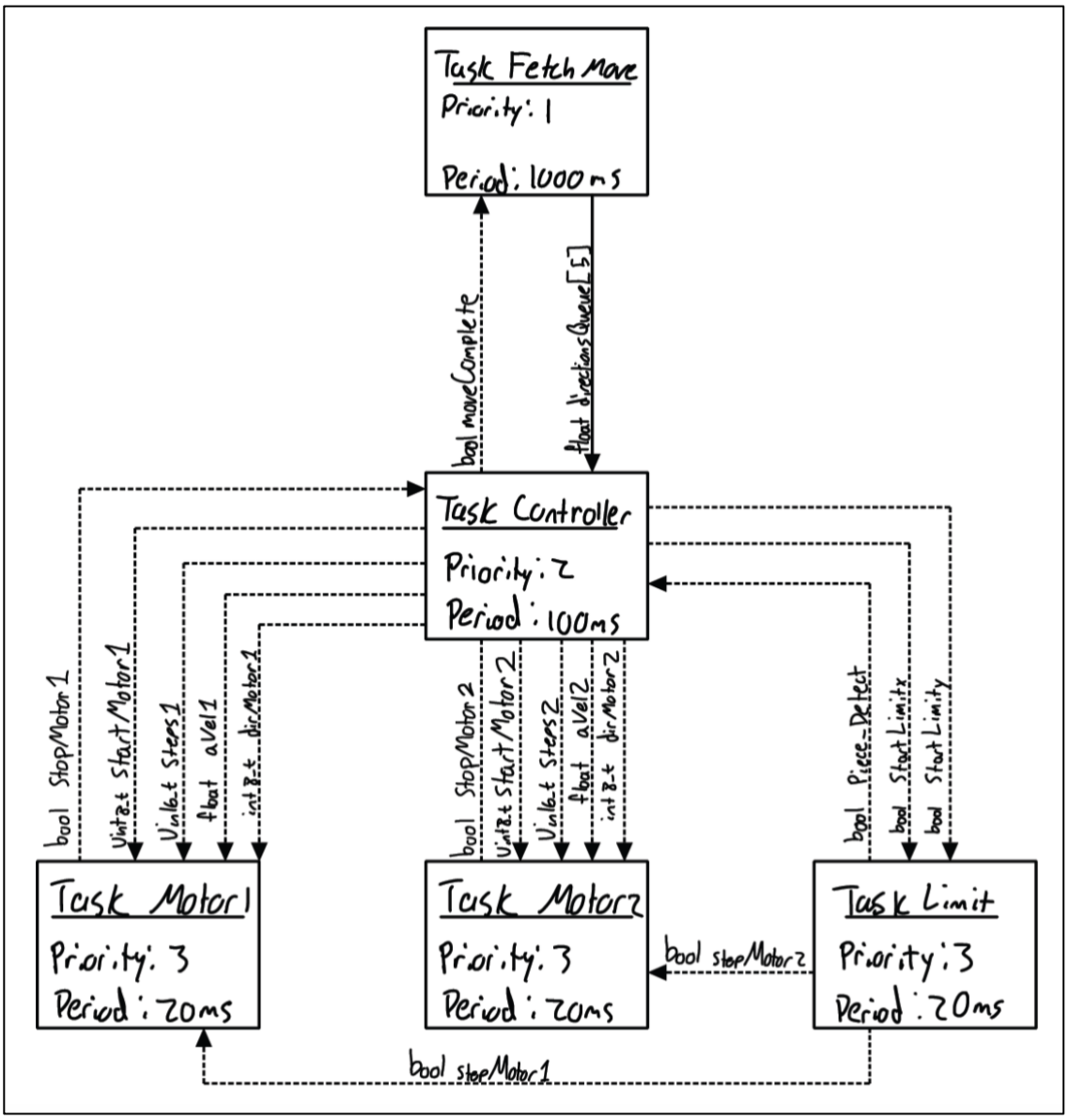


Figure 8: Task Diagram

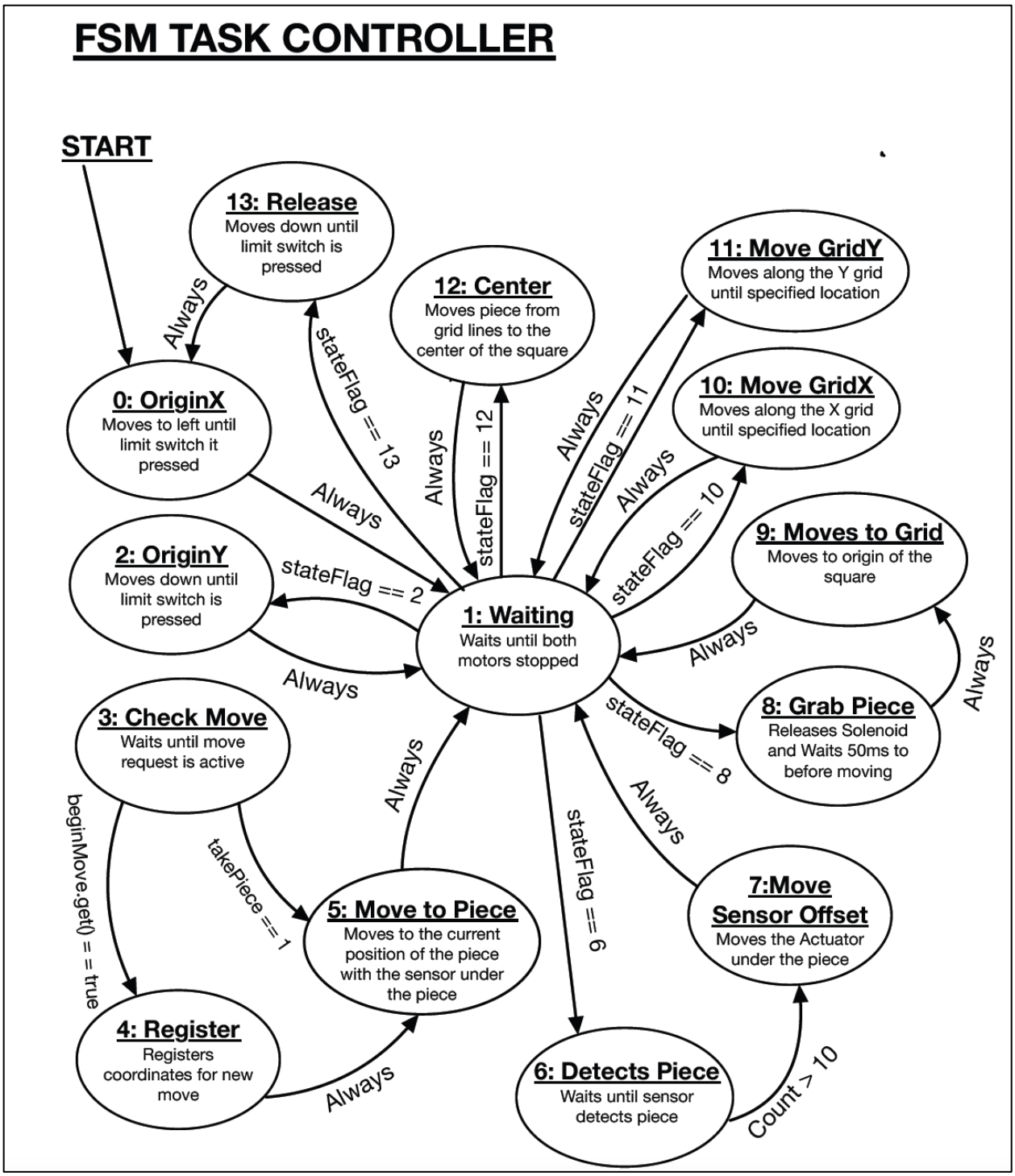


Figure 9: Controller Task FSM Diagram

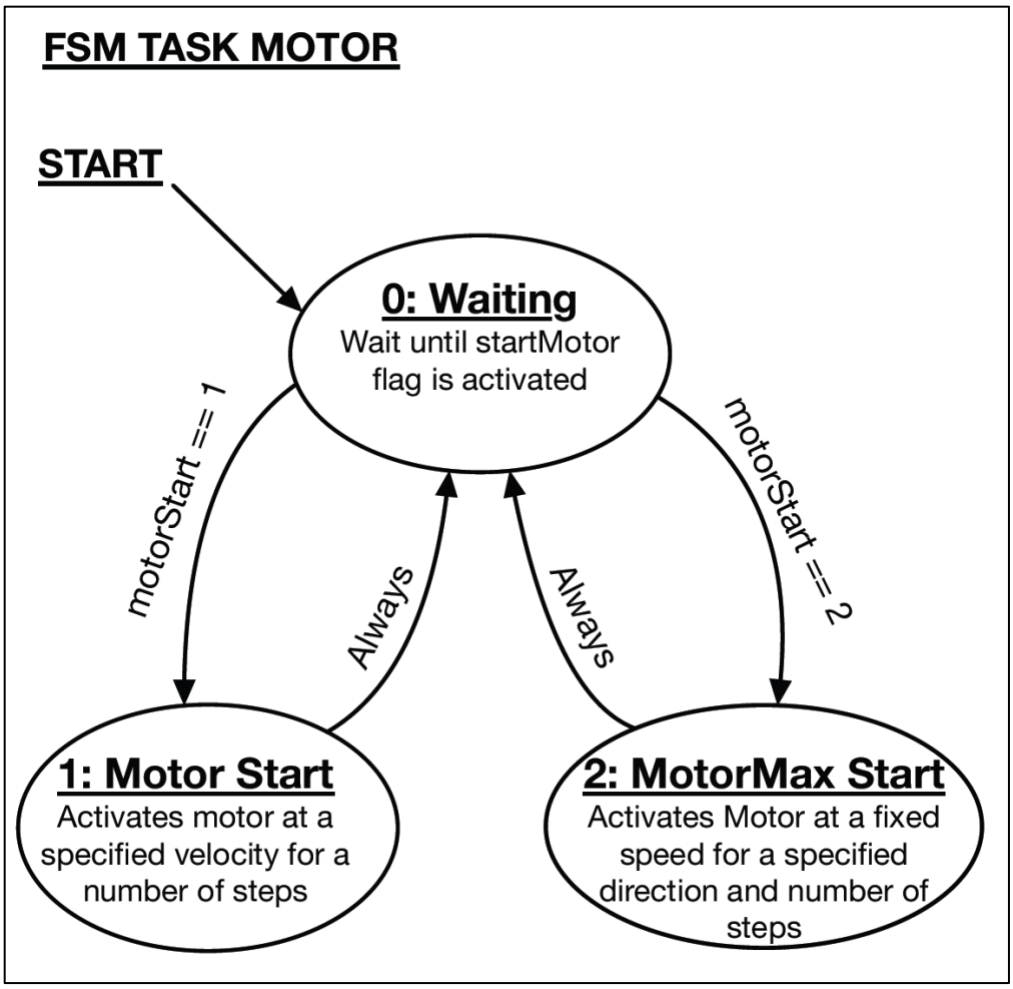


Figure 10: Motor Task FSM Diagram

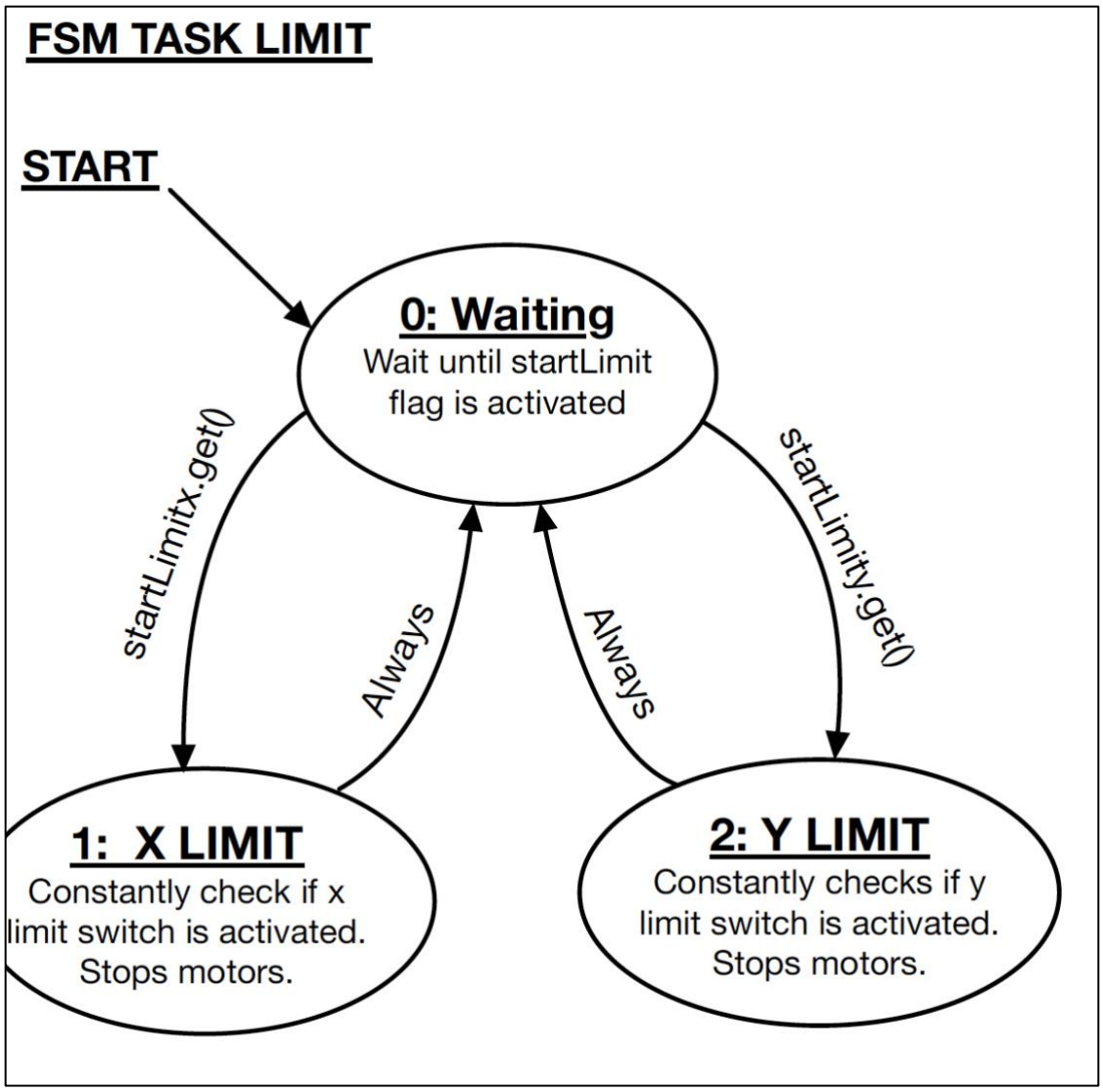


Figure 11: Limit Switch Task FSM Diagram

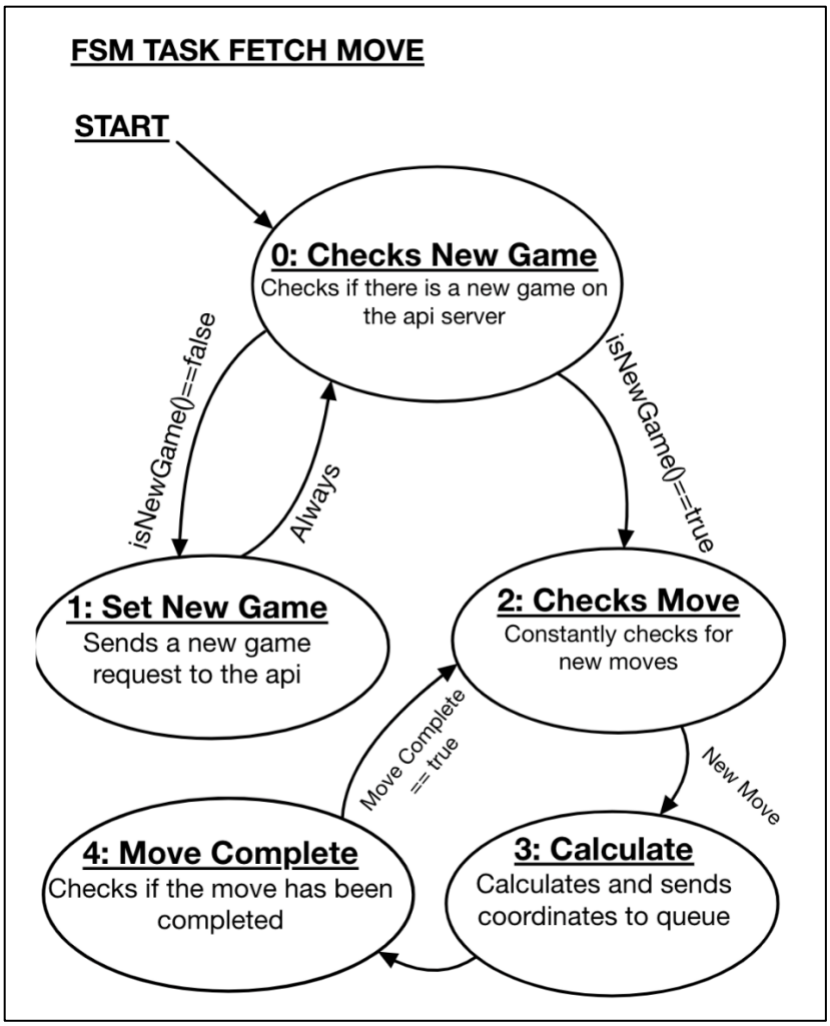


Figure 12: Fetch Move Task FSM Diagram

5.2.2. ChessBot Server[8]

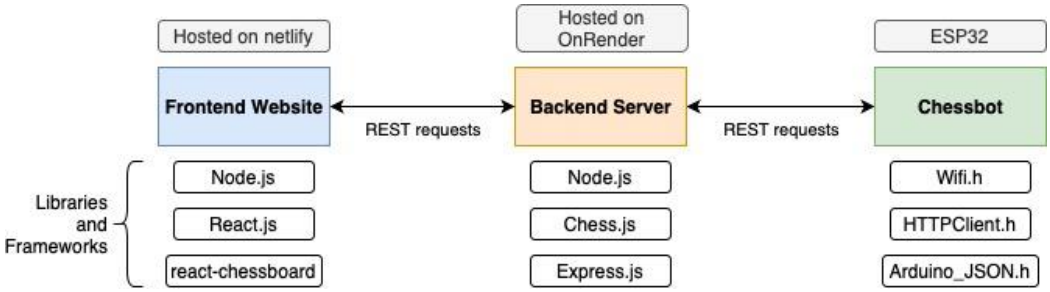


Figure 13: ChessBot server components and technologies

[8] The ChessBot server repository can be found here: https://github.com/sam-hud/ME507-ChessBotServer

The ChessBot is controlled via a REST API based backend server, which runs the Chess.js chess engine. The chess engine runs a chess game and enforces the game rules. User input to the backend server is gathered via the frontend website, which uses the react-chessboard component to display the current chess game and gather the new moves from the user. The chessboard on the frontend website is updated by making frequent API requests to the backend server. This allows multiple devices to access the same website and see and control the same game. The ESP32 gathers new move information by sending requests to the backend server. The response is then parsed using the Arduino_JSON library (since the responses from the backend server are given as JSON objects). Links to the libraries and frameworks used for the server can be found in the GitHub repository.

Figure 14: ChessBot server frontend interface

6. Results

The ChessBot performed well considering the scope of the project had to be reduced several times due to financial and time requirements. During the in-class demonstration day the functionality was limited as more debugging of the program was necessary. The web page and mechanical system were shown individually at that time. Several hours after the in-class demonstration, the minor bugs were ironed out of the tasks and a video was recorded highlighting the full integration and functionality of the systems within the robot.

The video can be seen here: https://youtu.be/pZI4D91iWpc

Horizontal and vertical movement patterns are displayed in the video, but the H-Bot motor and pulleys setup was also capable of diagonal translation for the main actuator. To speed up game piece

manipulation, diagonal movements could be utilized to reduce driving time of the stepper motors. Additionally, after more testing it could be unnecessary to perform the homing sequence for the robot after every move if steps are not skipped between position commands. Accumulation of seemingly minute time optimization processes would certainly make game play more efficient and fluid.

Stepper motors proved to be viable motors paired with the H-Bot configuration as precision and little force was necessary to move the main actuator around the board. The solenoid linear actuator struggled to stay retracted or extend at times as it appeared to experience overheating even though it was powered by its specified values. Before the solenoid experienced overheating it functioned adequately with the magnet on its end to pull chess pieces across the glass board.

The goal of having a person move a game piece physically and have an artificial intelligence robot be the opponent had to be changed as deadlines approached. For a human vs. robot interface, it is critical that the position of all game pieces be always known by the program. To satisfy this requirement a task for scanning the board using the infrared light sensor was drafted until the realization that it was error prone was discovered. Scanning the board squares could discern where pieces had moved to but in the event a take move was performed, it would not always be clear which piece belonged to which player. This scenario is depicted in Figure 15.

Figure 15: Scenario where game piece positions could not be determined by scanning algorithm.

The white pawn could take either pieces at B5 or D5 and after the take was executed the scanning process would not know whether the white pawn was located at B5 or D5. This problem led to the solution of just having two players make their moves in the web server.

Changes to the project would include switching the solenoid actuator to a servomotor, using a vision system set up to scan the board while recognizing the color of each piece, and scaling down the board to a more compact size. Using a servo instead of a solenoid would eliminate the problem of overheating, with the magnet being in one of two positions. A vision system would have allowed us to scan the board quickly and accurately. It would have solved the scenario presented in Figure 14 as well. After testing the range of the magnet's effect on the chess piece, as well as the accuracy of the H-Bot, the game board and the squares could have been sized down to an optimal size. Testing did not occur until the last week, so oversizing the board in our preliminary designs was the best option for this project.' Overall, the project was lots of fun for the team, despite the many sleepiness nights that occurred.

7. Appendices

7.1. Code Documentation and References

The documentation for the code can be found here: https://dyruiz131.github.io/me507_Term_Project/

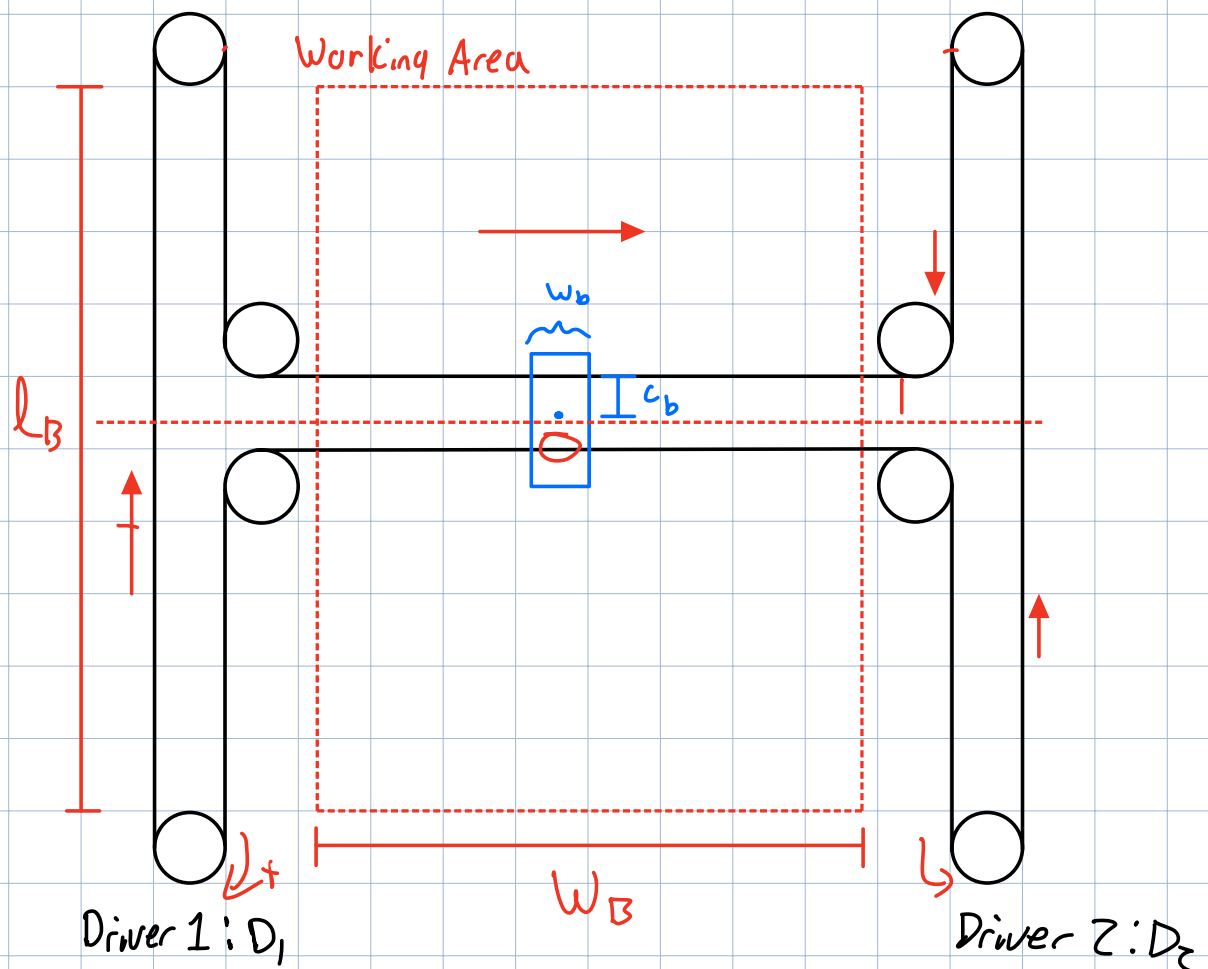
Table 3: External code library references

Library	Author	Link
Arduino_JSON	arduino-libraries	https://github.com/arduino-libraries/Arduino_JSON
ME-507 Support	spluttflob	https://github.com/spluttflob/ME507-Support
Arduino-PrintStream	fork: spluttflob, original: tttapa	https://github.com/spluttflob/Arduino-PrintStream

All other libraries used are standard ESP32 Arduino libraries.

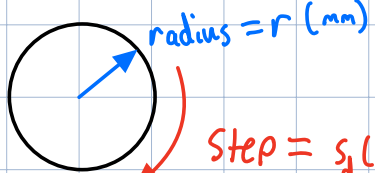
Referenced for ESP32 API setup: https://randomnerdtutorials.com/esp32-http-get-post-arduino/

7.2 Kinematics Calculations



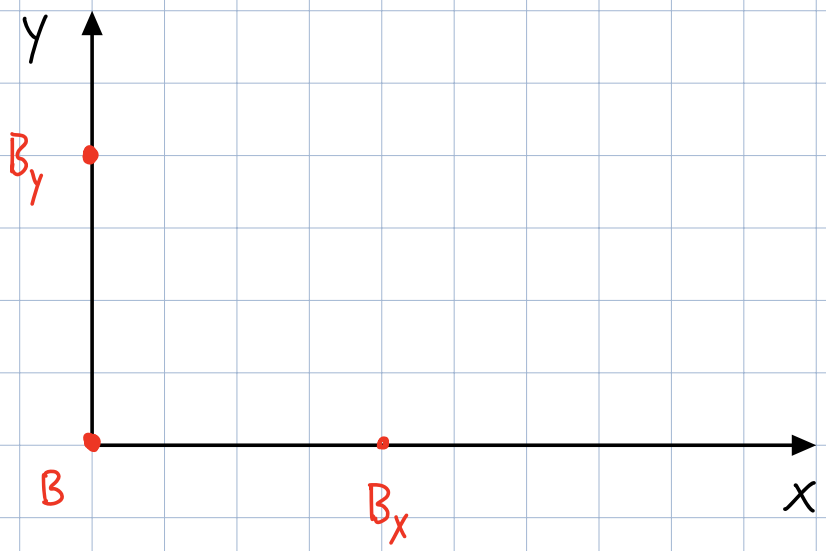
Kinematics

Driver Pulley



s_L = Length per step = $\frac{s_d}{360\left(\frac{deg}{rad}\right)} \cdot 2\pi r$

2 Scenarios - Lateral movement, Diagonal movement



$B \rightarrow B_x$: Horizontal movement

$$\theta_{D_1} = \theta_{D_2}$$
$$\dot{\theta}_{D_1} = \dot{\theta}_{D_2}$$

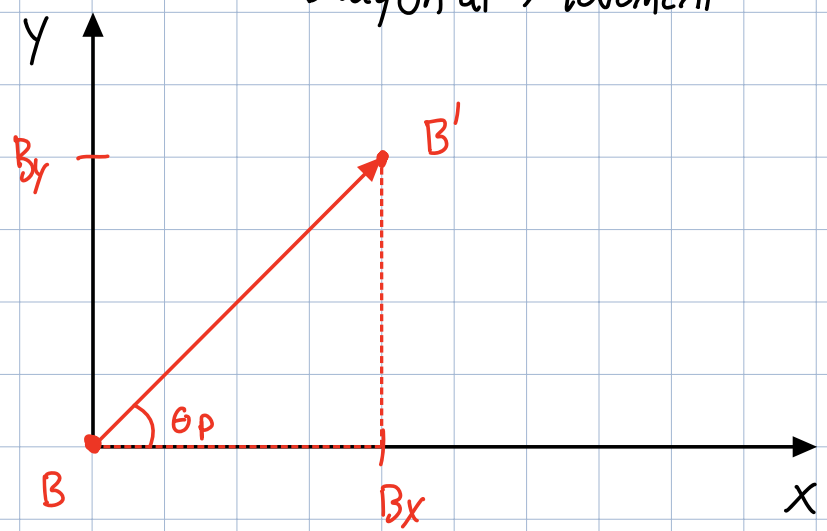
$$x = \#\text{ of steps} = \frac{B_x - B}{SL}$$

$B \rightarrow B_y$: vertical movement

$$\theta_{D_1} = -\theta_{D_2}$$
$$\dot{\theta}_{D_1} = -\dot{\theta}_{D_2}$$

$$y = \#\text{ of steps} = \frac{B_y - B}{SL}$$

Diagonal Movement



i) If $B_x > 0$ $B_y > 0$, $\dot{\theta}_{D_1} < 0$ $\dot{\theta}_{D_2} > 0$

$|\dot{\theta}_{D_1}|$ $|\dot{\theta}_{D_2}|$

If $B_x > 0$ $B_y < 0$, $\dot{\theta}_{D_1} > 0$ $\dot{\theta}_{D_2} < 0$

$|\dot{\theta}_{D_1}| > |\dot{\theta}_{D_2}|$

If $B_x < 0$ $B_y > 0$, $\dot{\theta}_{D_1} < 0$ $\dot{\theta}_{D_2} > 0$

$|\dot{\theta}_{D_1}| > |\dot{\theta}_{D_2}|$

If $Bx < 0$ $By < 0$, $\dot{\theta}_{D_1} < 0$ $\dot{\theta}_{D_2} > 0$

$$|\dot{\theta}_{D_1}| < |\dot{\theta}_{D_2}|$$

Code outline

Bx_1 = X coordinate - initial position

By_1 = y coordinate - initial position

Bx_2 = Designated x-coordinate

By_2 = Designated y-coordinate

$Dx = Bx_2 - Bx_1$

$Dy = By_2 - By_1$

Lateral Movement

If $Dx = 0$ & $Dy > 0$

$\dot{\theta}_1 < 0$ $\dot{\theta}_2 > 0$

If $Dx = 0$ & $Dy < 0$

$\dot{\theta}_1 > 0$ $\dot{\theta}_2 < 0$

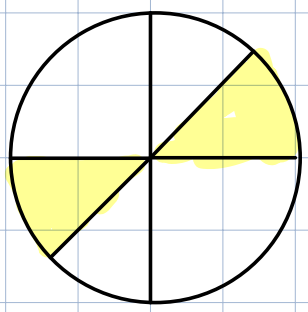
If $Dx > 0$ & $Dy = 0$

$\dot{\theta}_1 > 0$ & $\dot{\theta}_2 > 0$

If $Dx < 0$ & $Dy = 0$

$\dot{\theta}_1 < 0$ & $\dot{\theta}_2 < 0$

Diagonal Movement



Kinematics when $|\dot{\theta}_1| = \max$

Each movement w/ fixed $|\dot{\theta}_1|$

$\dot{\theta}_1 = \text{constant}$ $\quad \dot{\theta}_1 < 0 \quad \dot{\theta}_2 < 0$ $\quad \dot{\theta}_1 > 0 \quad \dot{\theta}_2 > 0 \quad |\dot{\theta}_1| > |\dot{\theta}_2|$

$$B_x = -\dot{\theta}_2 \cdot S_L \cdot t - \tfrac{1}{2}(\dot{\theta}_1 - \dot{\theta}_2) S_L \cdot t$$

$$B_x = -\tfrac{1}{2} S_L t (\dot{\theta}_2 + \dot{\theta}_1)$$

$$\dot{\theta}_2 = -\frac{2B_x}{S_L t} - \dot{\theta}_1$$

$$B_y = -\tfrac{1}{2}(\dot{\theta}_1 - \dot{\theta}_2) S_L \cdot t$$

$$t = -\frac{2B_y}{(\dot{\theta}_1 - \dot{\theta}_2) S_L}$$

$$\dot{\theta}_2 = \frac{2B_x (\dot{\theta}_1 - \dot{\theta}_2)}{2B_y} - \dot{\theta}_1$$

$$\dot{\theta}_2 = \dot{\theta}_1\left(\frac{\beta_x}{\beta_y} - 1\right) - \frac{S_x}{\beta_y}\dot{\theta}_2$$

$$\left(1 + \frac{\beta_x}{\beta_y}\right)\dot{\theta}_2 = \dot{\theta}_1\left(\frac{\beta_x}{\beta_y} - 1\right)$$

$$\boxed{\dot{\theta}_2 = \dot{\theta}_1 \frac{\left(\frac{\beta_x}{\beta_y} - 1\right)}{\left(1 + \beta_x/\beta_y\right)}}$$

Quad 2/8

Steps

$$\int_0^t \dot{\theta}_2 dt = \int_0^t \dot{\theta}_1 \frac{\left(\frac{\beta_x}{\beta_y} - 1\right)}{\left(1 + \beta_x/\beta_y\right)} dt$$

$$\theta_2 = \theta_1 \frac{\left(\frac{\beta_x}{\beta_y} - 1\right)}{\left(1 + \beta_x/\beta_y\right)}$$

$$\beta_x = -\tfrac{1}{2} S_L (\theta_2 + \theta_1)$$

$$-\frac{2\beta_x}{S_L} - \theta_2 = \theta_1$$

$$\theta_2 = \left(-\frac{2\beta_x}{S_L} - \theta_2\right)\left(\frac{\frac{\beta_x}{\beta_y} - 1}{1 + \beta_x/\beta_y}\right)$$

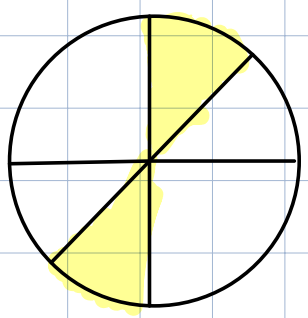
$$\theta_2 = -\frac{2\beta_x}{S_L} \cdot \left(\frac{\frac{\beta_x}{\beta_y} - 1}{1 + \beta_x/\beta_y}\right) - \theta_2\left(\frac{\frac{\beta_x}{\beta_y} - 1}{1 + \beta_x/\beta_y}\right)$$

$$\theta_2\left(1+\left(\frac{B_x/B_y-1}{1+B_x/B_y}\right)\right)=\frac{-2B_x}{S_L}\cdot\left(\frac{\frac{B_x}{B_y}-1}{1+B_x/B_y}\right)$$

$$\theta_2=\frac{\frac{-2B_x}{S_L}\left(\frac{B_x/B_y-1}{1+B_x/B_y}\right)}{\left(1+\frac{\left(B_x/B_y-1\right)}{1+B_x/B_y}\right)}$$

$$\theta_1=\frac{\frac{-2B_x}{S_L}}{\left(1+\frac{\left(B_x/B_y-1\right)}{1+B_x/B_y}\right)}$$

Kinematics when $|\dot{\theta}_1|=\max$



Each movement w/ fixed $|\dot{\theta}_1|$

$\dot{\theta}_1 = \text{constant}$

$\dot{\theta}_1<0 \quad \dot{\theta}_2>0$

$\dot{\theta}_1>0 \quad \dot{\theta}_2<0 \quad |\dot{\theta}|>|\dot{\theta}_2|$

$$B_y=\ddot{\theta}_2\cdot S_L\cdot t-\frac{1}{2}\left(\dot{\theta}_1+\dot{\theta}_2\right)\cdot S_L\cdot t$$

$$B_y=\tfrac{1}{2}S_L t\left(\ddot{\theta}_2-\dot{\theta}_1\right)$$

$$\dot{\theta}_2 = \frac{2\beta_y}{s_L t} + \dot{\theta}_1$$

$$\beta_x = -\frac{1}{2}(\dot{\theta}_1 - \dot{\theta}_2)\, s_L \cdot t$$

$$t = -\frac{2\beta_x}{(\dot{\theta}_1 - \dot{\theta}_2)\, s_L}$$

$$\dot{\theta}_2 = -\frac{2\beta_y\,(\dot{\theta}_1 - \dot{\theta}_2)}{2\beta_x} + \dot{\theta}_1$$

$$\dot{\theta}_2 = \dot{\theta}_1\left(1 - \frac{\beta_y}{\beta_x}\right) - \frac{\beta_y}{\beta_x}\dot{\theta}_2$$

$$\left(1 + \frac{\beta_y}{\beta_x}\right)\dot{\theta}_2 = \dot{\theta}_1\left(1 - \beta_y/\beta_x\right)$$

$$\boxed{\dot{\theta}_2 = \dot{\theta}_1 \frac{(1 - \beta_y/\beta_x)}{(1 + \beta_y/\beta_x)}}$$

From 90° or 270°

$$\int_0^t \dot{\theta}_2 dt = \frac{(1-\beta_Y/\beta_X)}{(1+\beta_Y/\beta_X)} \int_0^t \dot{\theta}_1 dt$$

$$\theta_2 = \theta_1 \frac{(1-\beta_Y/\beta_X)}{(1+\beta_Y/\beta_X)}$$

$$\beta_Y = \tfrac{1}{2} S_L (\theta_2 - \theta_1)$$

$$\theta_1 = \left(\frac{-2\beta_Y}{S_L} + \theta_2 \right)$$

$$\theta_2 = \left(\frac{-2\beta_Y}{S_L} + \theta_2 \right) \frac{(1-\beta_Y/\beta_X)}{(1+\beta_Y/\beta_X)}$$

$$\theta_2 = \frac{-2\beta_Y}{S_L} \frac{(1-\beta_Y/\beta_X)}{(1+\beta_Y/\beta_X)} + \theta_2 \frac{(1-\beta_Y/\beta_X)}{(1+\beta_Y/\beta_X)}$$

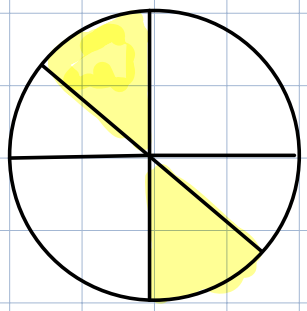
$$\theta_2 \left[1 - \frac{(1-\beta_Y/\beta_X)}{(1+\beta_Y/\beta_X)} \right] = \frac{-2\beta_Y}{S_L} \frac{(1-\beta_Y/\beta_X)}{(1+\beta_Y/\beta_X)}$$

$$\boxed{\theta_2 = \frac{\dfrac{-2\beta_Y}{S_L} \dfrac{(1-\beta_Y/\beta_X)}{(1+\beta_Y/\beta_X)}}{\left[1 - \dfrac{(1-\beta_Y/\beta_X)}{(1+\beta_Y/\beta_X)} \right]}}$$

$$\theta_1 = \frac{\frac{-2\beta_y}{S_L}}{\left(1 - \frac{(1-\beta_y/\beta_x)}{(1+\beta_y/\beta_x)}\right)}$$

Kinematics when $|\dot{\theta}_2| = \max$

Each movement w/ fixed $|\dot{\theta}_2|$



$\dot{\theta}_2 = \text{constant}$ $\quad \dot{\theta}_2 > 0 \;\; \dot{\theta}_1 < 0 \quad \dot{\theta}_2 < 0 \;\; \dot{\theta}_1 > 0$ $\;\; |\dot{\theta}_2| > |\dot{\theta}_1|$

$$\beta_y = -\dot{\theta}_1 \cdot S_L \cdot t + \frac{1}{2}(\dot{\theta}_2 + \dot{\theta}_1) \cdot S_L \cdot t$$

$$\beta_y = \frac{1}{2} S_L t (\dot{\theta}_2 - \dot{\theta}_1)$$

$$\dot{\theta}_1 = \dot{\theta}_2 - \frac{2\beta_y}{S_L t}$$

$$\beta_x = -\frac{1}{2}(\dot{\theta}_1 + \dot{\theta}_2) S_L \cdot t$$

$$t = -\frac{2\beta_x}{(\dot{\theta}_1 + \dot{\theta}_2) s_L}$$

$$\dot{\theta}_1 = \dot{\theta}_2 + \frac{\beta_y}{\beta_x}(\dot{\theta}_1 + \dot{\theta}_2)$$

$$\dot{\theta}_1 = \dot{\theta}_1 \frac{\beta_y}{\beta_x} + \dot{\theta}_2(1 + \beta_y/\beta_x)$$

$$\dot{\theta}_1(1 - \beta_y/\beta_x) = \dot{\theta}_2(1 + \beta_y/\beta_x)$$

$$\boxed{\dot{\theta}_1 = \dot{\theta}_2 \frac{(1 + \beta_y/\beta_x)}{(1 - \beta_y/\beta_x)}}$$

$$\int_0^t \dot{\theta}_1 = \int_0^t \dot{\theta}_2 \frac{(1 + \beta_y/\beta_x)}{(1 - \beta_y/\beta_x)}$$

$$\theta_1 = \theta_2 \frac{(1 + \beta_y/\beta_x)}{(1 - \beta_y/\beta_x)}$$

$$\beta_x = -\tfrac{1}{2}(\theta_1 + \theta_2) s_L$$

$$\theta_2 = \frac{-2\beta_x}{s_L} - \theta_1$$

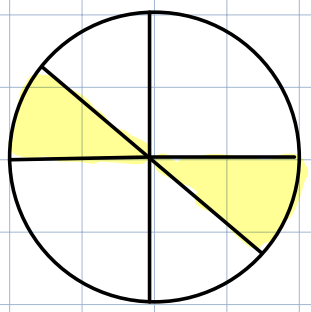
$$\theta_1 = \left(-\frac{2\beta_x}{SL} - \theta_1\right)\frac{\left(1+\beta_y/\beta_x\right)}{\left(1-\beta_y/\beta_x\right)}$$

$$\theta_1\left(1+\frac{\left(1+\beta_y/\beta_x\right)}{\left(1-\beta_y/\beta_x\right)}\right) = \frac{-2\beta_x}{SL}\frac{\left(1+\beta_y/\beta_x\right)}{\left(1-\beta_y/\beta_x\right)}$$

$$\theta_1 = \frac{\dfrac{-2\beta_x}{SL}\dfrac{\left(1+\beta_y/\beta_x\right)}{\left(1-\beta_y/\beta_x\right)}}{\left(1+\dfrac{\left(1+\beta_y/\beta_x\right)}{\left(1-\beta_y/\beta_x\right)}\right)}$$

$$\theta_2 = \frac{\dfrac{-2\beta_x}{SL}}{\left(1+\dfrac{\left(1+\beta_y/\beta_x\right)}{\left(1-\beta_y/\beta_x\right)}\right)}$$

Kinematics when $|\dot{\theta}_2| = \max$



Each movement w/ fixed $|\dot{\theta}_2|$

$\dot{\theta}_1 > 0 \quad \dot{\theta}_2 > 0$

$\dot{\theta}_2 = \text{constant} \quad \dot{\theta}_1 < 0 \quad \dot{\theta}_2 < 0 \quad |\dot{\theta}_1| < |\dot{\theta}_2|$

$$B_x = -\dot{\theta}_1 \cdot S_L \cdot t - \tfrac{1}{2}(\dot{\theta}_2 - \dot{\theta}_1) \cdot S_L \cdot t$$

$$B_x = \tfrac{1}{2} S_L t (\dot{\theta}_2 + \dot{\theta}_1)$$

$$\dot{\theta}_1 = -\frac{2B_x}{S_L t} - \dot{\theta}_2$$

$$B_y = \tfrac{1}{2}(\dot{\theta}_2 - \dot{\theta}_1) S_L \cdot t$$

$$t = \frac{2B_y}{(\dot{\theta}_2 - \dot{\theta}_1) S_L}$$

$$\dot{\theta}_1 = -\frac{2B_x (\dot{\theta}_2 - \dot{\theta}_1)}{2B_y} - \dot{\theta}_2$$

$$\dot{\theta}_1 = -\dot{\theta}_2\left(\frac{\beta_x}{\beta_y} + 1\right) + \frac{\beta_x}{\beta_y}\dot{\theta}_1$$

$$\left(1 - \frac{\beta_x}{\beta_y}\right)\dot{\theta}_1 = -\dot{\theta}_2\left(\frac{\beta_x}{\beta_y} + 1\right)$$

$$\boxed{\dot{\theta}_1 = -\dot{\theta}_2 \frac{\left(\frac{\beta_x}{\beta_y} + 1\right)}{\left(1 - \frac{\beta_x}{\beta_y}\right)}}$$

$$\int_0^t \dot{\theta}_1 dt = \int_0^t -\dot{\theta}_2 \frac{\left(\beta_x/\beta_y + 1\right)}{\left(1 - \beta_x/\beta_y\right)}$$

$$\theta_1 = -\theta_2 \frac{\left(\beta_x/\beta_y + 1\right)}{\left(1 - \beta_x/\beta_y\right)}$$

$$\beta_y = \tfrac{1}{2}\left(\theta_2 - \theta_1\right) S_L$$

$$\theta_2 = \frac{2\beta_y}{S_L} + \theta_1$$

$$\theta_1 = -\frac{2\beta_y}{S_L}\frac{\left(\beta_x/\beta_y + 1\right)}{\left(1 - \beta_x/\beta_y\right)} + \theta_1 \frac{\left(\beta_x/\beta_y + 1\right)}{\left(1 - \beta_x/\beta_y\right)}$$

$$\theta_1 \left(1 - \frac{(\beta_x/\beta_y + 1)}{(1 - \beta_x/\beta_y)} \right) = \frac{-2\beta_y}{S_L} \frac{(\beta_x/\beta_y + 1)}{(1 - \beta_x/\beta_y)}$$

$$\theta_1 = \frac{\dfrac{-2\beta_y}{S_L} \dfrac{(\beta_x/\beta_y + 1)}{(1 - \beta_x/\beta_y)}}{\left(1 - \dfrac{(\beta_x/\beta_y + 1)}{(1 - \beta_x/\beta_y)} \right)}$$

$$\theta_2 = \frac{\dfrac{2\beta_y}{S_L}}{\left(1 - \dfrac{(\beta_x/\beta_y + 1)}{(1 - \beta_x/\beta_y)} \right)}$$