C.H.E.S.S.
Chess **H**andling with **E**xtremity **S**urrogate **S**ystem

MAE C163A/263A Project Report

Group 2

Team Boomers:

Gabriel Alpuerto, Ethan Brandt, Hayato Kato, John Tabakian

Professor Dennis Hong, Junjie Shen

18 December 2020

Abstract

With the onset of the COVID Pandemic preventing chess players from safely playing comfortably face to face, we designed our robot C.H.E.S.S., to solve this issue. Normally chess playing robots are autonomous and are designed to play against each other or human opponents. Instead, our robot takes human input and translates it into chess moves to allow humans to either safely play apart and/or allow those without use of their arms to play chess. With a prismatic joint to slide along the length of the board and a unique rotating gripper that can hold two pieces at once, our robot can cover an entire chess board and comfortably move and capture any chess piece.

In accordance with the required project goals of this assignment, C.H.E.S.S. must have at least 4 degrees of freedom and utilize the Dynamixel MX-28AR servo motors as actuators. On top of these base requirements, we decided that our robot must utilize a prismatic joint for easy access to the entire chess board and have a rotating double gripper that can grab pieces that are captured along with placing its own piece to save movement time. Additionally, the workspace of the robot must cover the entire chess board for full functionality and its movement and gripper must not disturb any chess pieces as it grabs and moves pieces around the board.

To demonstrate the fulfilment of these goals, we have completed robust SolidWorks design and analysis along with MATLAB analysis to showcase how the robot would move and interact with the board. Both forward and inverse kinematics derivations for our manipulator with workspace analysis were completed to accurately describe the motion of the robot. Finite element analysis was also conducted to ensure any manufactured parts were capable of supporting the arm.

Since we are unable to manufacture and assemble this robot, the electronics and control systems are simply proposals that would need to be tested and refined.

Table of Contents

List of Figures

List of Tables

1 Preliminary Design

1.1 Concept Generation and Evaluation

Our design conceptualization began with our choice of degrees of freedom and joint types. We quickly decided upon using one prismatic joint followed by a series of revolute joints, the idea being that the prismatic joint moves to the correct file, and the revolute joints perform any actions necessary from there.

Next, we brainstormed several different configurations and joint lengths that could potentially work. Examples of these simple sketches are displayed below.

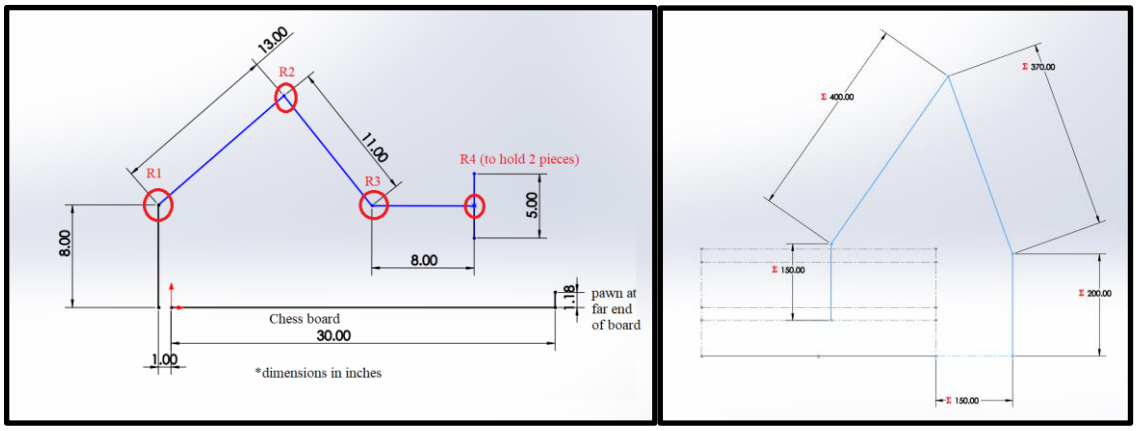


*Figure 1:***: *Preliminary link length configuration sketches***

We evaluated each potential configuration based on range of motion, potential torque issues, and piece carrying capability. It was important for us to maximize range of motion and the number of pieces the manipulator was able to move while also mitigating mechanical concerns.

1.2 Selected Design

Ultimately, the selected design was a 5 degree of freedom robotic arm system, with the 5th degree of freedom corresponding to a custom flipping gripper design (detailed in **Section 2.3**). The link lengths were iterated upon until a solution that optimized the system with respect to minimal servo torque and maximum lateral reach was found. The design is detailed more thoroughly in the following sections.

2 Detailed System Design

2.1 Prismatic

2.1.1 Design

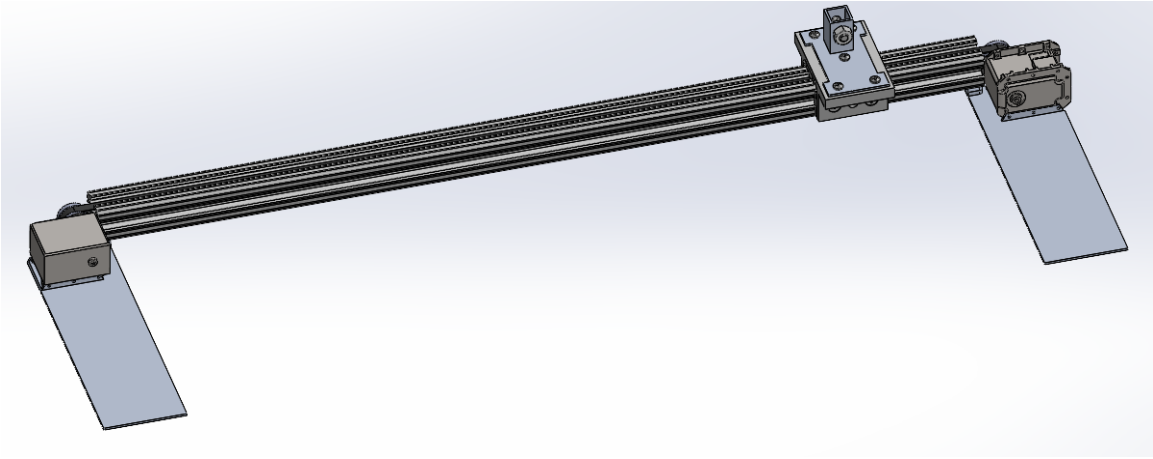
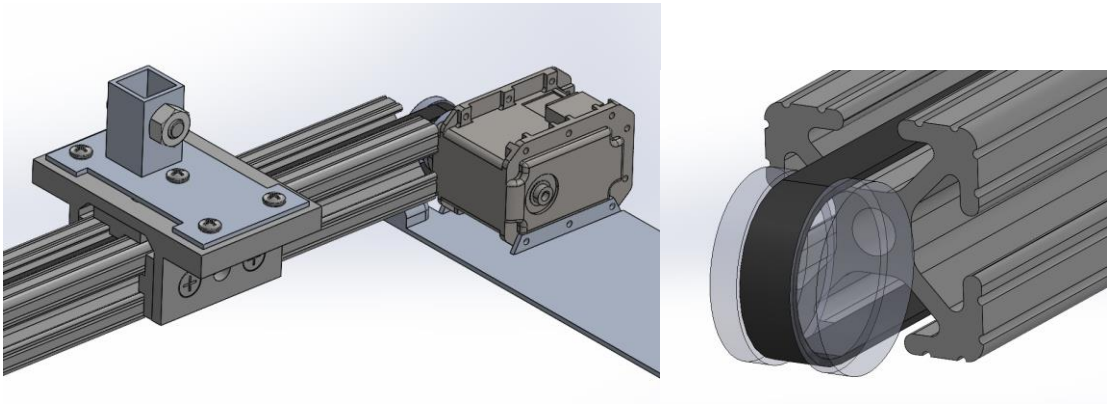


*Figure 2:**: Completed CAD of prismatic joint.***

The prismatic joint uses a belt and pulley system to achieve translation. This was chosen over a screw prismatic joint due to faster movement speed. In the figure above, the right pulley is driven by a MX-28AR servo, and the left pulley is not driven. Instead, it is held in place by a 3D printed structure. Additional 3D printed material extends out perpendicular from the belt to prevent the arm from flipping over during operation.



(a) **(b)**

*Figure 3:**: Details of the prismatic joint.** *(a) shows rail and carriage. (b) shows belt running within the rail*

The belt is used for achieving motion, but the structure that supports the manipulator arm during this motion is the rail. The belt runs inside of the rail, and the carriage that the

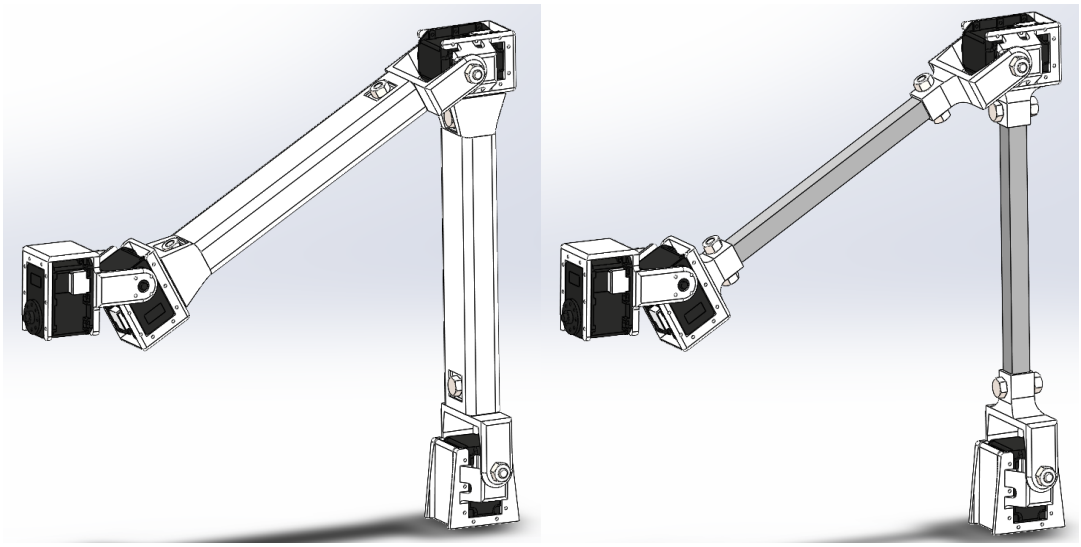
manipulator is mounted on is attached to both the belt and the rail. This allows the rail to support the weight of the arm while the belt moves it around the chess board.

2.1.2 Material Choice and Manufacturing

Every part that makes up the prismatic joint is either 3D printed or available for purchase off-the-shelf. The rail is a standard size T-slot rail. The carriage, by far the most complex geometry of any part in the prismatic joint, is an off-the-shelf part. This emphasis on using off-the-shelf parts allows for easy manufacturing and assembly, with no machining required (except for drilling holes in the belt to mount it to the carriage).

2.2 The Arm

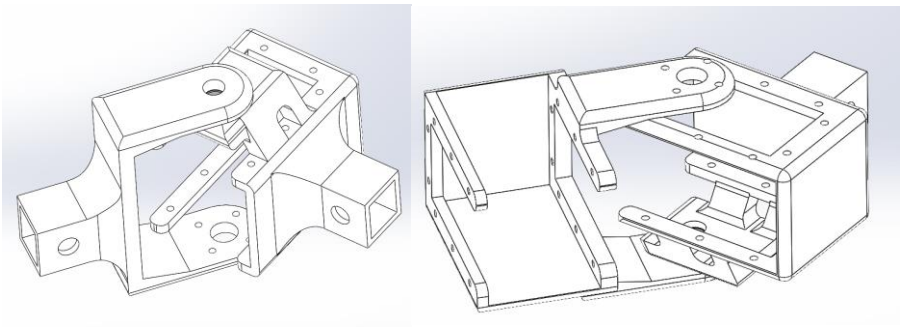
2.2.1 Design



(a) **(b)**

*Figure 4**: Completed CAD of arm assembly.** The link covers are hidden in **(b)**.*

The design of the arm is relatively simple: four MX-28AR servos, each mounted to a custom servo joint bracket, interconnected by metal square tubing. The first two revolute joints (counting from the base of the arm) share near identical mounting brackets. As seen in **Figure 4 (a)**, the j-1 bracket attaches to the servo directly while the j bracket attaches at one end to the servo horn and at the other to a ball bearing. Additionally, each bracket secures to square tubing via a 1/4-20 nut and bolt. The first j-1 bracket only differs in that it attaches to the prismatic joint instead of to square tubing, hence the lack of a square tube inserts at its base.



(a) **(b)**

*Figure 5:**: **Custom servo joint brackets.** The j-1 link is shown on the right of each picture with the corresponding j link shown on the left.*

The brackets for the third revolute joint are identical to those of the first two revolute joints, however the j bracket attaches directly to next servo instead of to square tubing. This forms the wrist since that next servo is the fourth and final revolute joint—connecting directly to the end effector.

Lastly, custom link covers were created to mount via the same 1/4-20 bolts that secure the servo brackets to the square tubing. These were created purely for aesthetic purposes and do not improve the performance of the robot whatsoever.

2.2.2 Material Choices and Manufacturing

It was decided that the 6061 aluminum would be used for the square metal tubing. The decision was made based on low cost, high availability, and ease of manufacturing. Additionally, 6061 aluminum is relatively lightweight while retaining the necessary material properties.

The custom servo brackets and link covers are to be made using 3D printed PLA (polylactic acid). This was done to reduce cost and time of manufacturing since a lot of subtractive manufacturing would need to be done for the brackets. PLA was chosen because of its excellent printing performance and availability.

2.3 The Gripper

2.3.1 Design

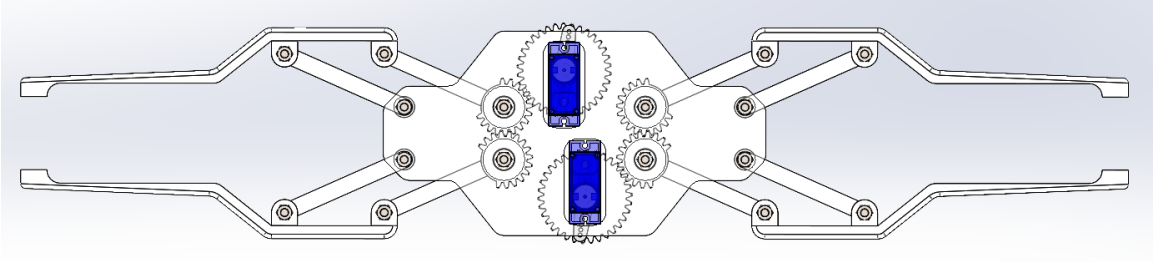


*Figure 6**: **Completed CAD of custom “X-Wing” gripper.***

The gripper consists of two sets of identical tweezers, oriented 180° from one another. Each tweezer is made of up a simple 4-bar-linkage, driven by a set of gears directly attached to a micro servo. The tips of the tweezers were made circular to match the base of a standard chess piece. Lastly, the backplate of the gripper mounts directly to the wrist servo via the provided circular servo horn.

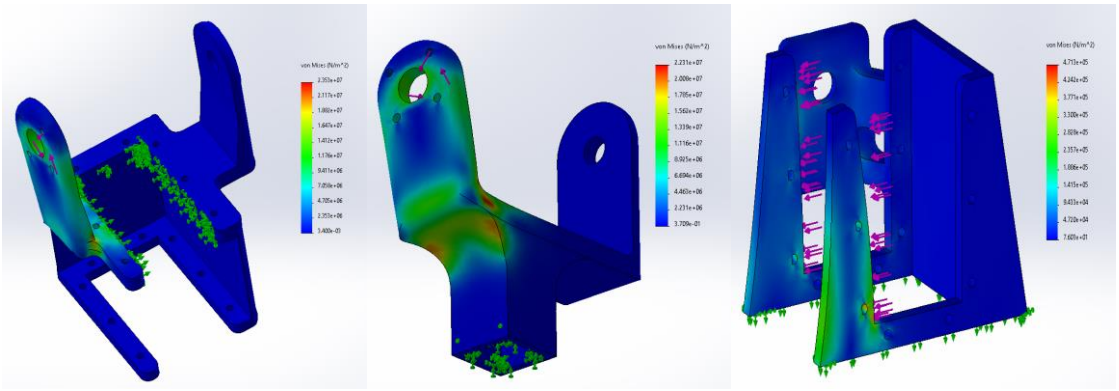
2.3.2 Material Choices and Manufacturing

The entirety of the gripper was designed to be 3D printable to reduce manufacturing time and cost. Although 3D printed gears are often troublesome, special care was given to ensure the involute curves of the spur gears were correct. Since, the force experienced by the gripper linkages will be relatively small, this material choice should not be a problem.

3 Engineering Analysis

3.1 Finite Element Analysis

Finite element analysis (FEA) was performed on all major 3D printed components of the arm to ensure validity of the design. Each component was subjected to an applied torque of ±2.5 N·m (corresponding to the stall torque of each servo) while fixing the interface between the bracket and the square tubing. In the case of the final servo joint bracket, the interface between the bracket and the servo itself was held fixed. The results indicated that the 3D printed components would not experience plastic deformation.



(a) **(b)** **(c)**

*Figure 7**: Selected FEA results.***

3.2 Kinematic Derivation

To begin deriving the kinematics of the robotic manipulator, frames are assigned to all the joints and the DH parameters are measured accordingly. The full derivation is shown in Appendix A1. Figure 3.2.a shows the kinematic scheme of C.H.E.S.S. with the appropriate frames attached to the joints.

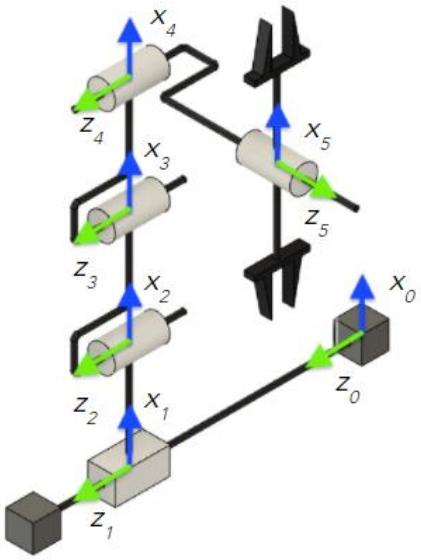


Figure 3.2.a

*Figure 8**: Kinematic Scheme of Robotic Manipulator***

Using SolidWorks and its evaluation tool, the DH parameters were measured within the SolidWorks assembly and are listed below in table 3.2.a.

i-1	i	α_{i-1}	a_{i-1}	d_i	θ_i
0	1	0	0	d_1	0
1	2	0	a_1	0	θ_2
2	3	0	a_2	0	θ_3
3	4	0	a_3	0	θ_4
4	5	$\pi/2$	a_4	d_5	θ_5
5	6	0	a_5	0	0

Table 1: DH Parameters of C.H.E.S.S

With the tabulated DH parameters, the equation for calculating the transformation matrix between joints (equation (1)) is used to calculate the transformation matrix for each joint. With these matrices, all the matrices are combined as seen in equation (2) and (3) to produce the transformation matrix from the zero frame to the 5th frame. This completes the forward kinematics of C.H.E.S.S.

$$ {}^{i-1}_{i}T = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \sin\theta_i \cos\alpha_{i-1} & \cos\theta_i \cos\alpha_{i-1} & -\sin\alpha_{i-1} & -d_i \sin\alpha_{i-1} \\ \sin\theta_i \sin\alpha_{i-1} & \cos\theta_i \sin\alpha_{i-1} & \cos\alpha_{i-1} & d_i \cos\alpha_{i-1} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1) $$

$$ {}^{0}_{5}T = {}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T\,{}^{3}_{4}T\,{}^{4}_{5}T \quad (2) $$

$$ {}^{0}_{5}T = \begin{bmatrix} C_{234}C_5 & -C_{234}C_5 & S_{234} & a_1 + a_2C_2 + a_3C_{23} + a_4C_{234} + d_5S_{234} \\ S_{234}S_5 & -S_{234}S_5 & -C_{234} & a_2S_2 + a_3S_{23} + a_4S_{234} - d_5C_{234} \\ S_5 & C_5 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3) $$

To derive the inverse kinematics, we start with equating the derived forward kinematics to the goal orientation and position which is represented symbolically as seen in equation (4). We need to derive equations for 5 variables: $d_1, \theta_2, \theta_3, \theta_4, \theta_5$.To start, variables from the goal matrix is equated with the respective equation from the transformation matrix from equation (3) which produces equations (5) – (9)

$$ \begin{bmatrix} r_{11} & r_{12} & r_{13} & P_x \\ r_{21} & r_{22} & r_{23} & P_y \\ r_{31} & r_{32} & r_{33} & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4) $$

$$ S_5 = r_{31}, C_5 = r_{31} \text{ (5), (6)} $$

$$ S_{234} = r_{13}, -C_{234} = r_{23} \text{ (7), (8)} $$

$$ d_1 = P_z \text{ (9)} $$

Two of the variables, d_1 and θ_5 , are easy to derive and are listed in equations (9) - (10).

$$ d_1 = P_z \text{ (9)} $$

$$ \theta_5 = a\tan 2\,(r_{31}, r_{32}) \text{ (10)} $$

Next, we derive θ_3 using a geometric solution. Using the sketch shown in figure 9, the Law of Cosines can be used to calculate 180 - θ_3.

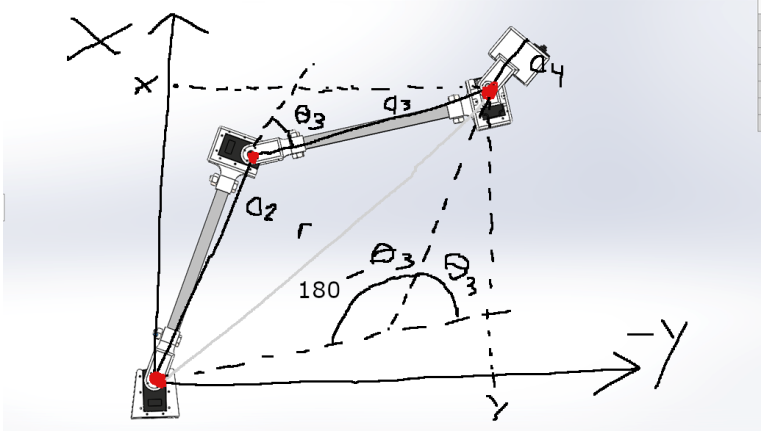


Figure 9: Sketch of Geometric Properties of robotic manipulator to determine theta 3

From equation (11), $\cos(\theta_3)$ can be isolated. With cosine, sin can be found using the trig identity: $C^2 + S^2 = 1$to produce an equation for θ_3.

$$r^2 = a_2^2 + a_3^2 - 2a_2a_2\cos(180 - \theta_3) \text{ (11)}$$

$$r^2 = x^2 + y^2 = a_2^2 + a_3^2 + 2a_2a_2\cos(\theta_3) \text{ (12)}$$

$$C_3 = \frac{x^2+y^2-a_2^2-a_3^2}{2a_2a_3}, S_3 = \pm\sqrt{1 - C_3^2} \text{ (13), (14)}$$

$$\theta_3 = a\tan 2\,(S_3, C_3) \text{ (15)}$$

Similarly, a geometric solution can be used to determine theta 2. As demonstrated in figure 10, theta 2 can be determined from angles β and ψ.

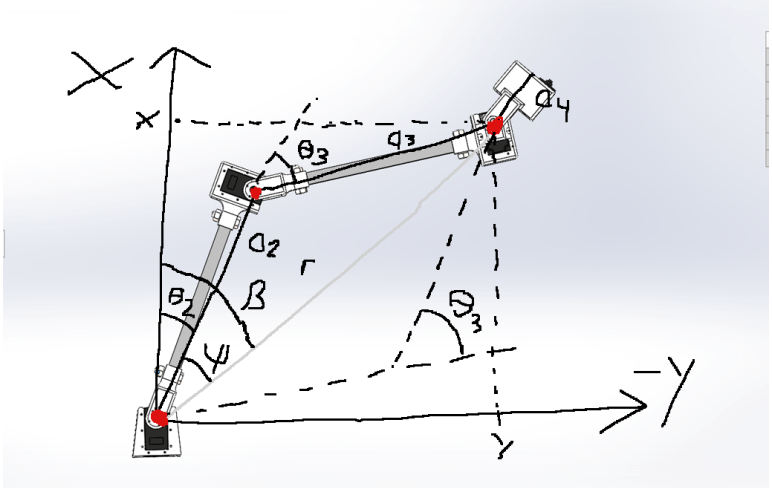


Figure 10: Sketch of Geometric Properties of robotic manipulator to determine theta 2

Angle β is simply $a\tan 2\,(y, x)$. Angle ψ can be determined using the law of cosines to produce equation (16). Isolating cosine leads to equation and sin can be found with the same trig identity as before. Thus, theta 2 equals equation (17).

$$\beta = a\tan 2\,(y, x)$$

$$0^\circ \le \psi \le 180^\circ$$

$$\psi = \cos^{-1} A$$

$$A = \frac{x^2 + y^2 + a_2^2 - a_3^2}{2a_2\sqrt{x^2 + y^2}} \quad (16)$$

$$\theta_2 = \beta + \psi \; if \; \theta_3 < 0 \quad (17.a)$$

$$\theta_2 = \beta - \psi \; if \; \theta_3 > 0 \quad (17.b)$$

Since the resulting goal orientation is a summer of theta 2, theta 3, and theta 4, theta 4 can be calculated with equation (18).

$$\theta_4 = a\tan 2\,(r_{13}, r_{23}) - \theta_2 - \theta_3 \quad (18)$$

Therefore, the solutions to the inverse kinematics can be summed up with the following equations.

$$d_1 = P_z$$

$$\theta_5 = a\tan 2\,(r_{31}, r_{32})$$

$$\theta_3 = a\tan 2\,(S_3, C_3)$$

$$\theta_2 = \beta + \psi\ (\theta_3 < 0)$$

$$\theta_4 = a\tan 2\,(r_{13}, r_{23}) - \theta_2 - \theta_3$$

3.3 Singularity Discussion

A singularity is the resulting loss of movement or loss of a degree of freedom under certain circumstances. There are two types of singularities robotic manipulators must look out for. First, there are boundary singularities which result from the robotic manipulator attempts to reach a point outside its workspace limits. Next there are joint alignment singularities that result from any of the joints physically aligning and cancelling out each other's respective rotations or translations.

As demonstrated in the subsequent workspace analysis, the workspace of the robotic manipulator covers the entirety of a chess board. The only case where a boundary singularity can occur is if the robot or the chess board is placed too far from the other.

As seen in the design discussion in the previous sections of the report, none of the joints physically align so a joint alignment singularity cannot occur either.

3.4 Workspace Analysis

Due to the specific application of the robotic arm, the workspace is restricted to the region above the chess board during operation. The entire workspace of the arm would be larger if there are no restrictions to the joint angles; however, that analysis is irrelevant here since the motion is restricted by software to have the end effector’s rotation axis parallel to the ground at all times. A visualization of the reachable workspace during a typical chess match is shown in **Figure 11**:

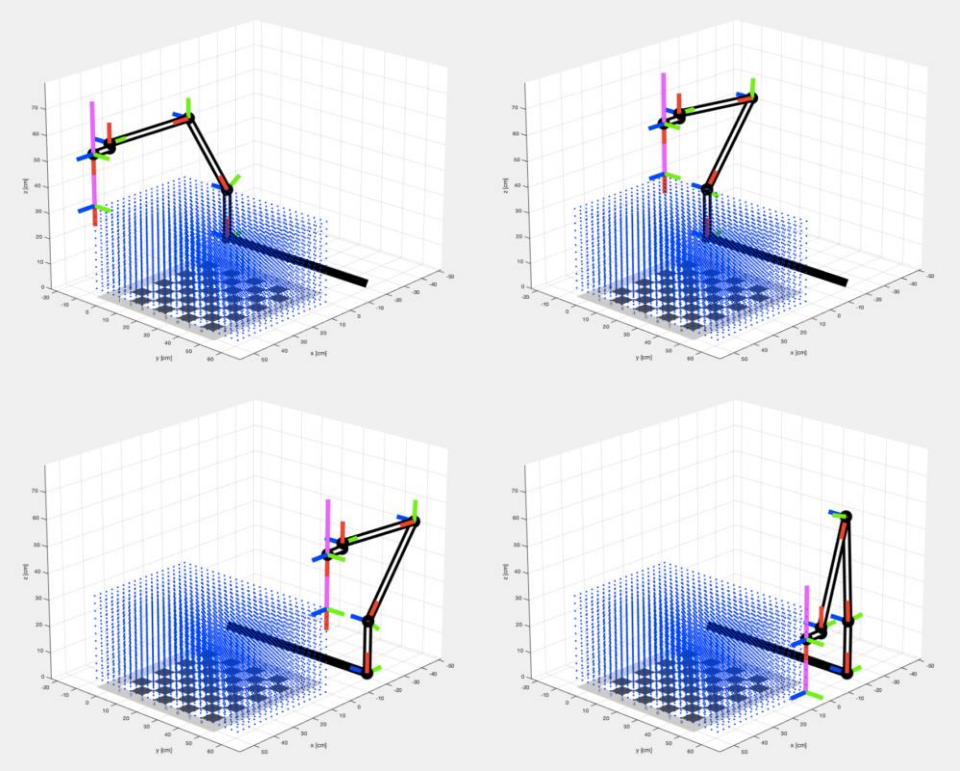


Figure 11: Reachable Workspace Visualization

The blue cuboid depicting the reachable workspace has a dimension of 18x27x7 inches, which fully encompasses the region above and around the chess board that has a dimension of 18x18 inches. The height of the reachable workspace is derived from the assumption that a height of at least 5 inches is required to avoid bumping into adjacent pieces when translating, in addition to the 2 inches that accounts for half of the length of the end effector. This accounts for half of the space needed for the end effector to rotate freely without colliding with the pieces on the board. Here, the workspace is drawn under the assumption that the robot arm is placed 9 inches away from the edge of the edge of a chess board.

3.5 MATLAB Simulation

3.5.1 Trajectory Creation

The trajectory of the robotic arm is calculated from a pair of board representation datasets that specify the chess move that is being made. The data is stored in a 2D array of strings for convenience and readability for a human, but future versions of this program should utilize a more efficient data structure that would drastically reduce the size of the data. The pair of datasets represent the board before the move is made and after the move is made, allowing the algorithm to extract important information such as the start and finish locations of a piece. A missing piece also lets the algorithm detect a capture, incorporating a flip motion when generating the motion path.

A typical chess move consists of three distinct motions: picking the piece up, translating over to the new location, and lowering to put the piece down. All three of these actions were modeled as simple linear translations that connected each waypoint using a straight line. Frames in between were calculated by using the linspace function from MATLAB that creates an evenly spaced vector between two values. The distance covered between successive frames was kept mostly constant by normalizing the translation distance per frame in the XY plane in respect to the distance covered during the raising and lowering of the piece in the Z direction. After creating the array of position vectors, the values were passed through a moving average filter that allowed it to smooth out the corners and make each motion look more seamless.

An exception is made when a capture is detected, in which case a 180 degree rotation is added during the translation animation to allow the end effector to point upwards. Due to the constraint of the end effector's rotating axis being parallel to the ground at all times, only one angle value is needed to specify the orientation of the end effector. To avoid the other side of the end effector colliding with the ground, the Z value of the destination is offset upwards by the length of the entire end effector, allowing for the other end effector to come down and grab the piece being captured. Here, the angle values were also passed through a moving average filter that enabled the same smooth transitions in between motions.

There are several pausing frames intentionally added to each starting and finishing location so that when the moving average is taken, the point still computes to be exactly that point due to the relatively small averaging window. This ensures that the trajectory actually reaches that point and without this compensation, the starting and finishing positions are shifted above the board by an arbitrary amount, which is not good.

The difference in the trajectory of the two types of moves are shown in **Figure 12**:

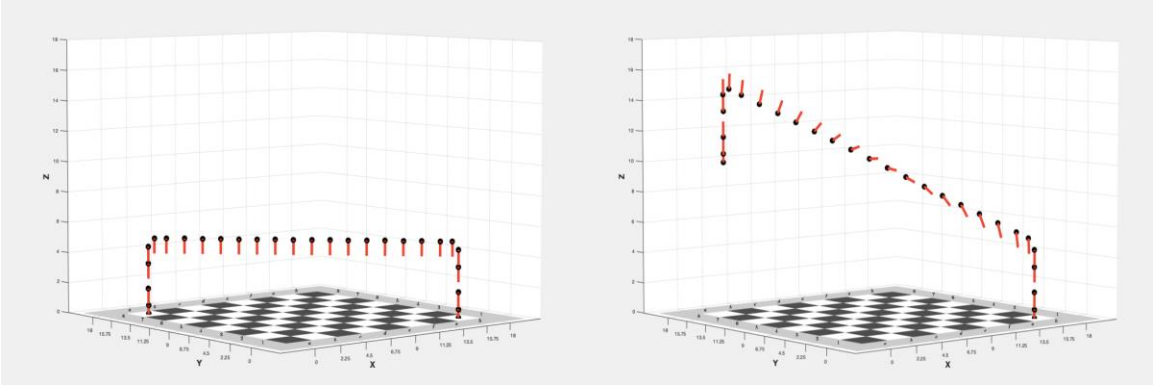


Figure 12: Trajectory of Moving a Piece (Left) versus Capturing a Piece (Right)

In addition to the first flip that is done while translating over to the new location, the capture move is made by adding another flipping motion that lets the robot arm put down the

originally picked up piece. Thus, another few frames are added afterwards in order to create a 180 degree motion in a similar fashion.

Overall, the three dimensions of the position vector and one angle value were calculated for each animation frame, which were later converted to a transformation matrix so that it can be passed onto the inverse kinematics function. For the final product, two more dimensions are required to synchronize the opening and closing of each end effector, which would most likely contain a set of angular values for the servos to rotate at.

3.5.2 Kinematic Simulation

Once a trajectory has been created, the MATLAB simulation uses the derived forward and inverse kinematics from the previous section to determine the motion of the robotic manipulator. The simulation itself showcases both the robotic manipulator with its frames (as shown in figure 13). The black bar near the chess board is the prismatic joint, the black bars with white are the links between revolute joints, and the pink bars represent the end effectors. The simulation also has a yellow line that shows the trajectory of a grabbed piece over time which is shown later.

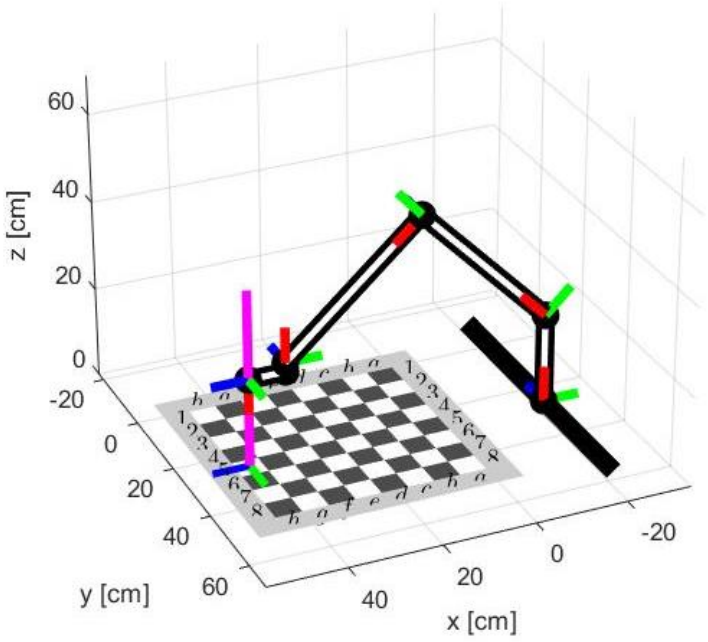
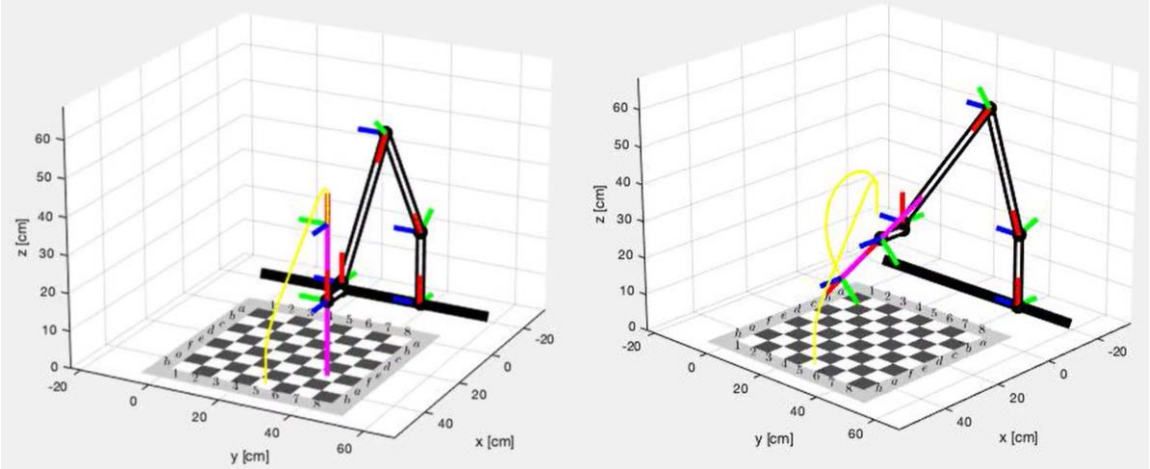


Figure 13: Starting position of robotic manipulator in the MATLAB simulation

As the simulation runs, a for loop calculates the related forward and inverse kinematics for every trajectory inputted into the loop. As mentioned before, the yellow line demonstrates the motion of a chess as it is grabbed by the first end effector. As shown in figure 14 the robot can grab a chess piece, rotate, grab the second chess piece, and place the first chess piece in the location of the 2nd chess piece.



(a) **(b)**

Figure 14:Starting position of robotic manipulator in the MATLAB simulation

4 Electronics System Design

4.1 The Arm

In terms of the electronics required for this robotic arm to properly function in the real world, a microcontroller with wireless connectivity is preferable. The controller itself does not have to be powerful, since all it needs to do is interpret the incoming chess moves into physical arm motion. Separate encoders are not required due to the Dynamixel servos having capability of reading and monitoring the actuator status. A separate power supply for the motors is needed since they take 12V to operate, which normal microcontrollers do not take. One bottleneck could be the image detection of the chess board, but this can also be offloaded onto the computer to handle by simply returning an image of the board instead of the move in algebraic notation.

4.2 Control System Design

Appendix A2 shows the entire flow chart of C.H.E.S.S. if it were to be manufactured as a chess playing robot.

The main features include taking in algebraic notation as inputs, which is convenient since many chess playing programs already output this notation, allowing for any chess program to become the backend of the robotic arm. This is then converted into board representation form such that it is easier for the program to interpret the starting and finishing locations of a move.

The camera will most likely use a form of object detection that would be able to examine the board to determine where each piece is located at. This functionality is needed when playing against a human player, since the robot would otherwise have no way of detecting

what move the human made. The image taken will be analyzed and will output a chess move in algebraic notation such that it can be directly interpreted by the chess playing AI.

5 Summary and Discussion

5.1 Summary of Goals and Objectives

With C.H.E.S.S., our team set out to create a robot capable of playing a game of chess on behalf of human players, to ensure their ability to socially distance while but also play chess in-person. A brief summary of our technical goals and progress towards achieving them is detailed below.

Planned Deliverables	Success?	Comments
Robot can reach any position on the chess board	Yes	This capability is demonstrated in our workspace analysis.
Ability to pick up and manipulate two pieces simultaneously	Yes	Our innovative "X-Wing" gripper design allows for independent manipulation of two chess pieces.
Solve the forward and inverse kinematics for our manipulator design	Yes	Thanks to Professor Hong and Junjie for teaching us how to do this.
Generate and simulate trajectories for moving and capturing pieces	Yes	Trajectories were successfully simulated using MATLAB.
Build and test our design	No :(	COVID-19 pandemic prevented our team from meeting in person.

Table 2: Table of Goal Completion

Aside from our unfortunate inability to physically create our robot due to world events outside of our control, our project achieved all of its major goals!

5.2 Competition Analysis

Most chess-playing robots in existence today have two major differences from C.H.E.S.S.

1. They play autonomously against a human or another autonomous robot
2. They are capable of carrying only one piece at a time

Figure 15: Typical chess-playing robots

Our product outperforms the competition, with the ability to carry multiple pieces for seamless, human-like capturing. And unlike the autonomous chess robots, our robot can respond to human commands and make moves on behalf of a human player. While our robot does not currently incorporate autonomous play as one of its features, this could be easily implemented in future editions with some electronics and software additions. In summary, C.H.E.S.S. is a unique chess-playing robot with enhanced manipulation capabilities and the ability to receive user input, making it unlike any other chess robot on the market.

6 Conclusions and Recommendations

With C.H.E.S.S., our team set out to create the definitive chess-playing robot. Thanks to our simple yet elegant designs, our product is easy to manufacture, assemble, and use. Through this project, we have verified the quality of our design through engineering analysis, successfully derived the kinematics of our link configuration, and generated workspace and trajectory simulations to visualize the operation of the robot. We learned a great deal about robotics through this project and created a product we are proud of.

If this project were to continue, some steps to take could include:

- Optimizing design for speed and precision during piece handling
- Adding autonomous functionality through incorporation of a chess engine and board vision
- Implementing voice control for the user input mode for greater ease of use

With these additional features, C.H.E.S.S. would be one of the most capable and user-friendly chess-playing robots in existence.

Thanks to Professor Hong and Junjie for teaching this engaging and exciting course and giving us the opportunity to learn from them throughout the quarter.

Reference

1. “PLA Technical Data Sheet.” SD3D, 2017. https://www.sd3d.com/wp-content/uploads/2017/06/MaterialTDS-PLA_01.pdf

Bill of Materials

Table A: Bill of Materials

No.	Item (Part Number)	Supplier	Unit Cost	Qty	Total Cost
1	MX-28AR Servo (902-0095-000)	Robotis	$249.90	5	$1249.5
2	6061 Square Tubing (6546K49)	McMaster-Carr	$6.30	1	$6.30
3	Micro Servo (2307)	Adafruit	$11.95	2	$23.90
4	T-Slot Framing (47065T101)	McMaster-Carr	$10.57	1	$10.57
5	Raspberry Pi 4 Model B - 1GB RAM	Adafruit	$30.00	1	$30.00
6	USB-C Power Supply 5.1V 3A	Adafruit	$7.95	1	$7.95
7	Raspberry Pi Dynamixel Servo Controller board	Tribotix	$139.00	1	$139.00
8	Raspberry Pi Camera Board v2 8MP	Adafruit	$29.95	1	$29.95
9	Carriage for T-Slot Framing (60585K35)	McMaster-Carr	$46.16	1	$46.16
10	Prismatic Joint Belt (6484K501)	McMaster-Carr	$3.72	1	$3.72
11	Assorted Screws	McMaster-Carr	$10.00	1	$10.00
				Total:	**$**

Appendix

A1: Complete Kinematic Derivation

Forward Kinematics

$$ {}^{i-1}_{i}T = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \sin\theta_i \cos\alpha_{i-1} & \cos\theta_i \cos\alpha_{i-1} & -\sin\alpha_{i-1} & -d_i \sin\alpha_{i-1} \\ \sin\theta_i \sin\alpha_{i-1} & \cos\theta_i \sin\alpha_{i-1} & \cos\alpha_{i-1} & d_i \cos\alpha_{i-1} \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Only Non zero variables:

$$ a_1 = 5.5 \text{ cm} $$
$$ a_2 = 26 \text{ cm} $$
$$ a_3 = 40 \text{ cm} $$
$$ a_4 = \text{-2 cm} $$
$$ \alpha_4 = \frac{\pi}{2} $$
$$ a_5 = 8 \text{ cm} $$
$$ \mathrm{d}_5 = 20 \text{ cm} $$

Resulting Transformation Matrices

$$ {}^{0}_{1}T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{1}_{2}T = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & a_1 \\ \sin\theta_2 & \cos\theta_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{2}_{3}T = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & a_2 \\ \sin\theta_3 & \cos\theta_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{4}_{5}T = \begin{bmatrix} \cos\theta_4 & -\sin\theta_4 & 0 & a_3 \\ \sin\theta_4 & \cos\theta_4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{4}_{5}T = \begin{bmatrix} \cos\theta_5 & -\sin\theta_5 & 0 & a_4 \\ 0 & 0 & -1 & -d_5 \\ \sin\theta_5 & \cos\theta_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{0}_{5}T = {}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T\,{}^{3}_{4}T\,{}^{4}_{5}T $$

$$ {}^{0}_{5}T = \begin{bmatrix} C_{234}C_5 & -C_{234}C_5 & S_{234} & a_1 + a_2C_2 + a_3C_{23} + a_4C_{234} + d_5S_{234} \\ S_{234}S_5 & -S_{234}S_5 & -C_{234} & a_2S_2 + a_3S_{23} + a_4S_{234} - d_5C_{234} \\ S_5 & C_5 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Inverse Kinematics

$$ \begin{bmatrix} r_{11} & r_{12} & r_{13} & P_x \\ r_{21} & r_{22} & r_{23} & P_y \\ r_{31} & r_{32} & r_{33} & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ = \begin{bmatrix} C_{234}C_5 & -C_{234}C_5 & S_{234} & a_1 + a_2C_2 + a_3C_{23} + a_4C_{234} + d_5S_{234} \\ S_{234}S_5 & -S_{234}S_5 & -C_{234} & a_2S_2 + a_3S_{23} + a_4S_{234} - d_5C_{234} \\ S_5 & C_5 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

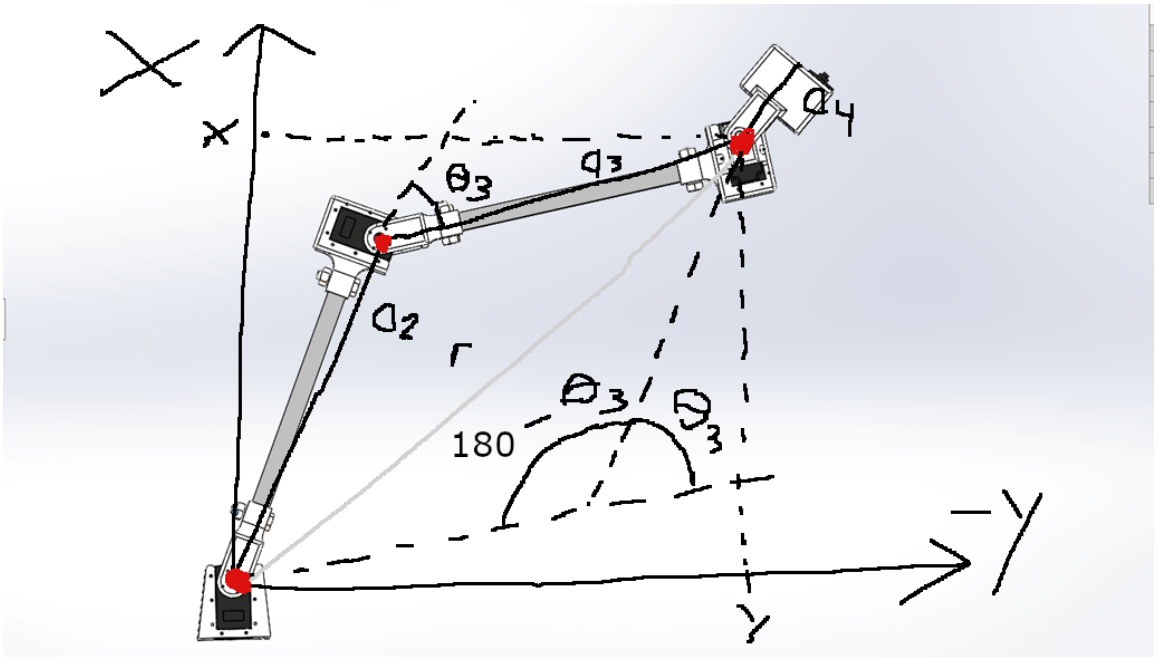
$$ S_5 = r_{31}, C_5 = r_{31} $$

$$ S_{234} = r_{13}, -C_{234} = r_{23} $$

$$ d_1 = P_z $$

$$ \theta_5 = a\tan 2\,(r_{31}, r_{32}) $$

$$ \theta_4 = a\tan 2\,(r_{13}, r_{23}) - \theta_2 - \theta_3 $$

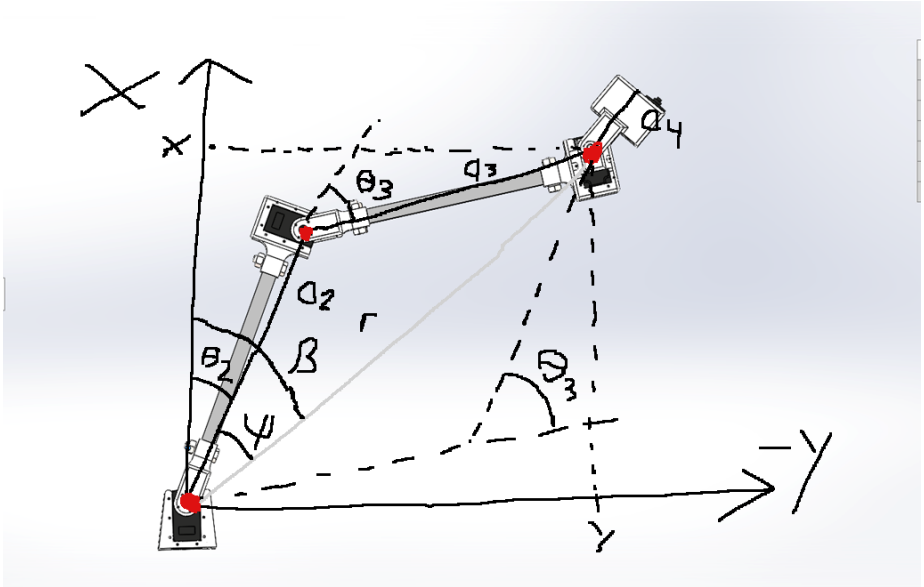


$$r^2 = a_2^2 + a_3^2 - 2a_2a_2\cos(180 - \theta_3)$$

$$r^2 = x^2 + y^2 = a_2^2 + a_3^2 + 2a_2a_2\cos(\theta_3)$$

$$C_3 = \frac{x^2 + y^2 - a_2^2 - a_3^2}{2a_2a_3}, S_3 = \pm\sqrt{1 - C_3^2}$$

$$\theta_3 = a\tan 2\,(S_3, C_3)$$



$$\beta = a\tan 2\,(y, x)$$

$$0^\circ \le \psi \le 180^\circ$$

$$\psi = \cos^{-1} A$$

$$A = \frac{x^2 + y^2 + a_2^2 - a_3^2}{2a_2\sqrt{x^2 + y^2}}$$

$$\theta_2 = \beta + \psi \; if \; \theta_3 < 0$$

$$\theta_2 = \beta - \psi \; if \; \theta_3 > 0$$

Summary of Solutions:

$$d_1 = P_z$$

$$\theta_5 = a\tan 2\,(r_{31}, r_{32})$$

$$\theta_3 = a\tan 2\,(S_3, C_3)$$

$$\theta_2 = \beta + \psi\,(\theta_3 < 0)$$

$$\theta_4 = a\tan 2\,(r_{13}, r_{23}) - \theta_2 - \theta_3$$

A2: Control Flow Diagram

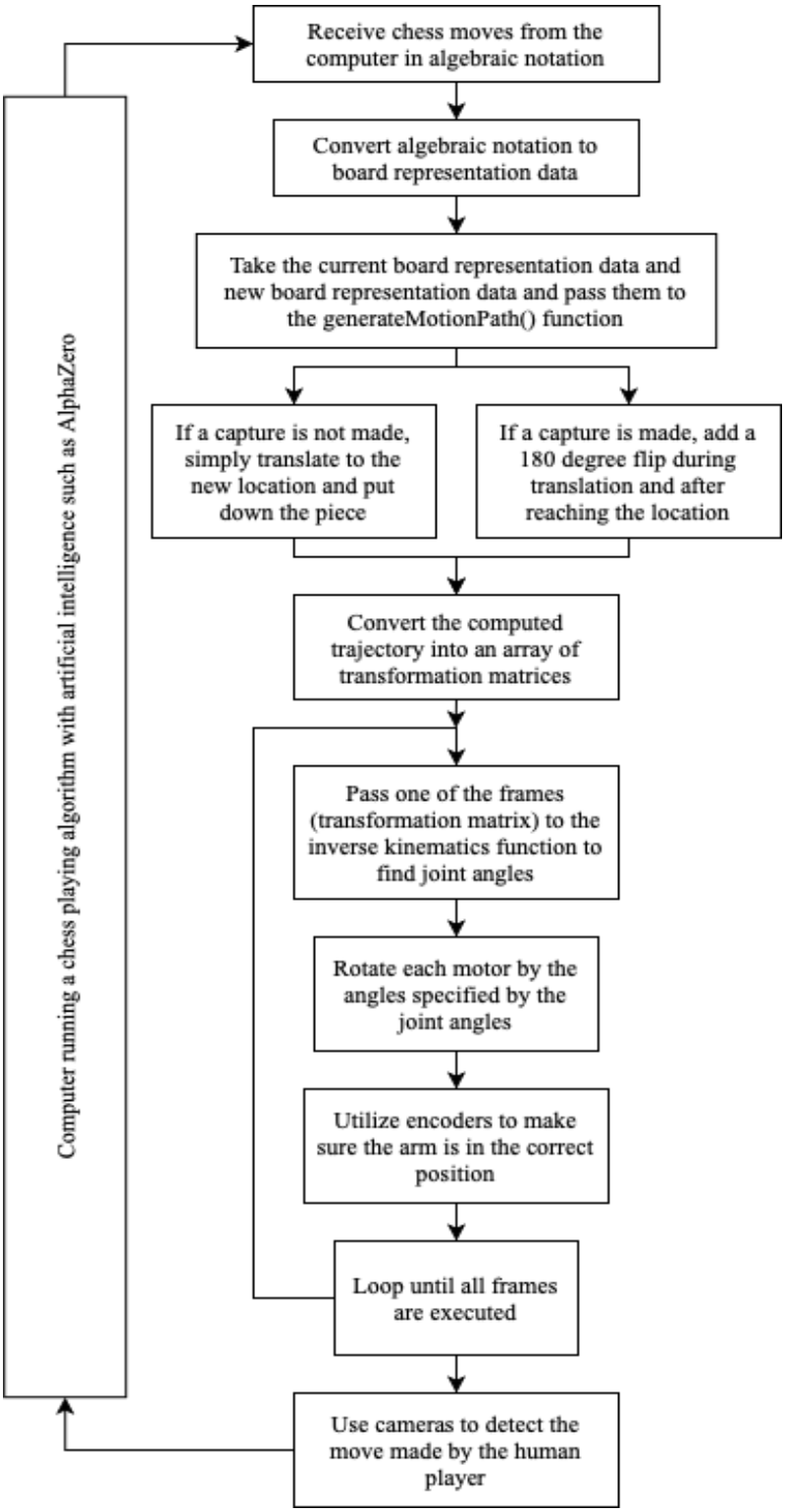


Figure #: C.H.E.S.S. Flow Chart