UNIVERSITY OF GEORGIA
College of Engineering

Capstone Design Final Report

University of Georgia Motion Simulation Chair

Customer: Dr. Thomas

Advisor: Dr. Yao

5/6/2024

CSEE4911

Design Group:

Matthew Coffey

John Cook

Sean Schlief

Charles Zipperer

Table of Contents

Abstract

Our motion simulation chair was designed as a fully-functional, open-source, and modular product providing accurate motion simulation experiences. It is defined by its unique disassembly and portability, making it an ideal product for convenient simulation demos for driving, flying, roller coasters, video/arcade games, and more. Specifically, this motion simulation chair is designed to provide pitch, roll and yaw. To achieve these three degrees of motion the design is divided into three modular sections. The bottom section provides yaw with a caster and track design with a central stepper motor. The middle section provides pitch and roll using two stepper motors rotating a frame which is held by a universal joint. The top section includes the chair, safety harness and footrest for the user. Each axis is powered by 9NM stepper motors and each motor is driven by a capable high current driver that is in turn powered by a 350W DC power supply. These components are specified to ensure that the motion delivered to the user is always exciting. At the heart of our Motion Simulation Chair is an Arduino-based control system, tuned to translate user inputs into three- dimensional movements, along with a pre-programmed, full motion experience.

Introduction

Project Objective

The main goal of this Capstone Project was to design and fabricate a prototype for an affordable and capable modular motion simulation chair. This is because current simulation chairs that have fully-featured movement are expensive which his limits their appeal for education and hobbyist groups. Because we also wanted the design to be open-source, we wanted to create a product that could be easily recreated for a similar affordable price.

Design Approach

The design approach for this project involved several key stages, beginning with extensive research into technology involving motion simulation and existing products on the market. This stage was important for collecting data on what the technical requirements for the project should be and for getting ideas on what potential design directions that we could take.

Following this stage, we proceeded into the computational design phase, where we leveraged software tools to model and simulate different design concepts. This was complemented by calculation verifications, ensuring that the designs that we came up with met the necessary performance targets that we were trying to achieve while following safety specifications. With a verified design in mind, the components needed to manufacture and assemble the chair were purchased.

This moved us into the manufacturing and assembly stages, where we carefully constructed the modular sections of the chair, while also integrating both the electrical and software components. Finally, we conducted thorough testing and evaluation to assess the performance of our design and make small adjustments as needed. This design approach ultimately allowed us to develop a functional and modular motion simulation chair that exceeds the project's objectives.

Goals

The goals for this project were centered around creating a motion simulation chair that was highly functional and versatile. This involved designing a chair that would be modular, easily portable, have three degrees of freedom (pitch, roll and yaw), and include a demo that would showcase the chair's capabilities. The modularity of the design would lie in its ability to be easily assembled and disassembled, while the portability would come from its ability to be wheeled around and fit through the standard doorway space.

Team Introduction

Matthew Coffey is a 5th year mechanical engineering major. He is the Capstone Project team leader and the yaw platform mechanical designer.

Jack Cook is a 5th year mechanical engineering major. He is the pitch and roll platform mechanical designer.

Charles Zipperer is a 4th year electrical engineering major. He is the electrical system designer and he ensures parity between electrical, mechanical, and computerized systems.

Sean Schlief is a 5th year computer systems engineer. He is the software designer and collaborates with both the electrical and mechanical engineers to ensure seamless between the chair's mechanical, electrical, and control systems.

Accomplishments

Despite the challenges we endured during the course of this project, we were able to successfully overcome delays and managed to deliver a design that exceeded our client's expectations. Our design was fully functional for the Capstone showcase, and allowed us to present a working motion simulation chair to our audience. Furthermore, our efforts were verbally recognized by the judges and our design was assessed within the Communicative Software Design category at the Capstone showcase.

Design Constraints

Customers and Stakeholders

For this project, we identified both internal and external stakeholders who played crucial roles in shaping the design and ensuring the success of it all. The internal stakeholders included the project team, consisting of Sean Schlief, Charles Zipperer, Jack Cook, and Mathew Coffey. The team was also supervised by Dr. Peter Kner, managed by Dr. Ben Thomas, and mentored by both Dr. Ben Thomas and Dr. Kun Yao. All of the internal stakeholders listed were crucial in addressing technical challenges and ensuring that the design met the desired specifications.

As for the external stakeholders, this project encompassed a wide variety of potential end users who would benefit from the chair. These end users included, but are not limited to, the University of Georgia, pilots, gamers, driving instructors, arcade owners, and other motion simulation enthusiasts. These stakeholders provided valuable insights into what they would be looking for in a motion simulation chair, emphasizing the importance of aspects like affordability, modularity, and realistic motion simulation.

Requirements

The requirement of this senior design capstone project was to design and fabricate a Motion Simulation Chair that is relatively affordable and equally capable as existing market options. This requires coordination between mechanical, electrical, computer system engineering students for successful construction as the chair must have a working frame design, electromechanical movement design, and a computerized control system.

For specific requirements we turned to already existing products on the market. Current fully-featured motion simulation chairs that have two or more degrees of freedom cost over two thousand dollars and are all closed-source. The affordability of current designs on the market restricts who has access to be entertained by motion simulation something. Based on this we intended to design and build a prototype model on a restricted budget of $1600 or lower, while also having as many degrees of motion as possible.

Standards and Safety

Due to the nature of this project being a moving system with several integrated components, it falls under several different engineering standards. The IEEE publishes a series called the National Electrical Safety Codes (NESC) that are important for this project. These codes specifically cover the installation, operation, and maintenance of electrical equipment. To follow the NESC codes we planned on ensuring every component is properly grounded, that all the wires are insulated, and that no exposed connectors are easily reachable.

The Mechanical Safety Codes outlined by the ASME have the goal of helping to ensure safety and reduce the risk of failures that could cause personal injury or property damage. To this extent we took great care to ensure that our design was safe by following factor of safety recommendations and using a four-point safety harness. Additionally, all of the metal used to

make the frames came from reputable suppliers which provided ASTM grades.

The Permissible Exposure Limitations (PEL) Noise Standards written by OSHA are also important for this project because it integrates several electronic motors, which can be loud enough to damage a person's hearing. To ensure compliance with this we made sure to only chose motors that are known to generate limited noise output. Because hearing damage starts around 80dB which is well above what our selected motors could output, we still focused on lowering our device's noise floor to make sure it would not annoy any potential users.

Engineering Specifications

- What Degrees of Motion are we trying to achieve?
  - We are trying to achieve three degrees of freedom. Specifically, the rotational degrees of freedom, which are pitch, roll, and yaw.
- What range of motion are we trying to achieve?
  - We are trying to achieve 15° of motion minimum in all three rotational directions.
- What speed of motion are we trying to achieve?
  - We are trying to achieve a speed of 90 deg/s in all rotational directions.
- What weight capacity are we trying to achieve?
  - We are trying to achieve a weight capacity maximum of ~330lbs.
- What cost are we trying to achieve?
  - We are trying to achieve a cost of under ~$1600.
- What power consumption are we trying to hit?
  - We are trying to incorporate a total power consumption under 1.2kW.
- What computational interface are we trying to incorporate?
  - We are trying to incorporate a simple demo selection interface via Arduino IDE.
- How long of demo are we trying to provide?
  - We are trying to provide at least 60 sec. worth of high-quality motion simulation demos.

Background Research

Motion simulation chairs are often used for flight training, recreational/hobbyist simulation, and educational demonstrations by moving and rotating in correspondence to user input and audio/video output. The problem for our team is that affordable and capable modular motion simulators do not exist, which limits their mass-market appeal.

We began our research by looking into previous products and developing our design based on what they do right and what they do wrong.

Current Motion Simulation Chairs Models:

Yaw 2 Pro Edition 3DoF:

Figure 1: Yaw 2 Pro Product Image [7]

Yaw 2 Pro Edition 3DoF [7]

- Design Specs
  - DoF- 3 (roll, pitch, yaw)
  - Motion Range- roll: 40° pitch: 70°
  - Speed-160 °/sec (roll and pitch) 360 °/sec yaw
  - Size- L4.95 x H2.79 x W2.17 ft.
  - Operator Max Weight- 287 lbs.
  - Price- $4070
- Pros:
  - The design allows for a very high range of motion.
  - The speed is very fast compared to other around the same price.
- Cons:
  - The design is complex which brings high manufacturing costs
- Parts we would like to contribute from design and other takeaways
  - The design for the base platform that produces rotation in yaw is very effective yet conceptually simple

Qubic System- QS-CH1:

Figure 2: Qubic System Product Image [9]

Qubic System- QS-CH1 [9]

- Design Specs
  - DoF- 3 (pitch, roll, heave)
  - Motion Range- roll: 15.8° pitch: 10.2° heave: 100 mm.
  - Speed- max speed for actuators 300 mm/s
  - Maximum total weight- 661 lbs.
  - Acceleration- 0.4G up to 0.8G
  - Size- L56 x W33.7 x H30 in.
  - Price- $12,320
- Pros:
  - Simple design that is easy to manufacture yet quite capable.
  - The total weight capacity is significantly higher than most other products on the market.
- Cons:
  - Price is very high due to industrial linear actuators.
  - Low range of motion and can't rotate in yaw.

DOF Reality H3:

Figure 3: DOF Reality H3 Product Image [8]

DOF Reality H3 [8]

- Design Specs:
  - DoF- 3 (pitch, roll, yaw)
  - Motion Range- 20°
  - Speed- 50 cm/s (86°/sec)
  - Torque- 25 n/m
  - Operator max weight- 330lbs
  - Size- W3 × L5 × H2 ft. (without seat)
  - Acceleration- up to 0.7G
  - Price- $2527
- Pros:
  - Motor and gearbox assembly is a cheap mechanical actuation option that also provides very responsive and accurate motion.
  - Frame design is made of off-the-shelf parts that are easy to acquire.
  - Design is modular which achieves our project goal.
  - Provides three rotational degrees of motion which achieves our project goal.
  - Design is transportable and the dimensional footprint is sufficient which achieves our project goal.
- Cons:
  - The method through which motion is achieved limits the range of motion to low values
- Parts we would like to contribute from design and other takeaways
  - Method of achieving rotation in pitch and roll is simple and effective

By analyzing these current motion simulation chairs, we were able to sort them into categories based on how they create motion. The three chairs listed above are an example from each category. The Yaw 2 pro is an example of a chair that uses the rotational motion from electric

motors to rotate different parts of the chair using a system of gears and rollers. We named this category Yaw VR. Chairs in the linear actuator category use linear actuators to tilt a central frame in order to create motion. The Qubic System QS-CH1 seen above is an example of this type of design. The last category is the DOF Reality style. This design concept uses electric motors with perpendicular arms extending from the shaft. The arms are connected to a separate frame which is usually pinned so it can rotate in one degree of freedom or connected with a universal joint to allow rotation in two degrees of freedom. Almost all of the motion simulation chair that were analyzed fell into one of these groups. By finding the average values of important specifications in each category we were able to compare categories to determine which design concept would be best for our application.

Current Motor Solutions

As for the motors needed to move the various platforms, there are many options including standard DC motors and its many variants, servo motors, stepper motors, linear actuators, direct drives, and universal motors. We decided that we needed affordability and performance above almost all other factors if we wanted to simulate motion as accurately possible. The only two motor designs that were commonly used in motion simulation chairs are DC motors and linear actuators, but there were a select few that utilize stepper motors. Stepper motors and linear actuators were more expensive than DC motors, but offered a much higher level of performance. After some research we came across the NEMA line of stepper motors.

Figure 4: CL Stepper Motor & Driver Kit

These motors offered much faster impulse responses and more overall torque than any other motors did at their price point. Alongside this, these motors are closed-loop, meaning their efficiency is incredibly high and track absolute position. All of this made these motors incredibly appealing and so we chose the NEMA 34 9Nm stepper motors as our motors of choice for this project.

Current Coding Solutions

In regards to the research involving software, our initial software investigation focused on finding a robust library for controlling the stepper motors. The AccelStepper library was a strong

candidate, given its extensive functionality and ease of use. However, during extensive testing phases, we encountered a critical constraint outlined in its documentation: “speeds of more than 1000 steps per second are unreliable” [5]. Given the specific gearbox output ratio and stepping mode configurations we employed, this maximum speed threshold proved insufficient for achieving the desired motion.

Subsequent investigation led us to the FastAccelStepper library, developed for optimizing stepper motor control on platforms like the Arduino Mega 2560. This library leverages the capabilities of the ATMega 256 microcontroller, boasting significantly enhanced performance metrics. With reported speeds of “up to 50000 generated steps per second for single stepper operation, 37000 for dual stepper, and 20000 for three steppers” [6], it presented ample capacity to meet the demands of our motion simulation requirements. The FastAccelStepper library ultimately proved to be a more suitable solution for our project.

Design

Mechanical Design: Pitch and Roll Platform

The goal for the pitch and roll platform was to create a device that would rotate the rider in two degrees of freedom. From research a few different methods for obtaining movement in pitch and roll were determined. After identifying these different methods, the chairs were able to be sorted into categories corresponding to the method they use. Further analysis was done to determine pros and cons of each movement method as well as finding the typical values associated with specific metrics from products that fall into each category. Some of the metrics that were collected were range of motion, speed, weight limit and cost. This information allowed us to determine which method would be best for our application and what specifications we would need to reach in order to be competitive.

Pitch and Roll Platform Design Decision Matrix		
Criteria and Weight	Designs and Scoring (1-5)	
	Design 1: Yaw VR Style	Design 2: DOF Reality Style
Manufacturability (x4)	8	16
Range of Motion (x3)	15	9
Acceleration of Motion (x2)	8	4
Smoothness of Motion (x3)	12	12
Modularity and Assembly (x4)	12	16
Dimension Footprint (x2)	8	6
Weight (x1)	2	5
Cost (x5)	10	20
Total Score	75	88

Figure 5: Pitch and Roll Design Decision Matrix

The DOF Reality style was the method chosen based on a number of factors. The main benefits of this are its low cost and simplicity. When compared to the Yaw VR style which is included in the decision matrix in **Figure 5** the DOF Reality style typically has a lower range of motion. The way the design operates, achieving a high range of motion is difficult, however it was determined from research that a smaller range of motion is not always a negative. The idea is to have enough motion to trick the brain into thinking it is actually doing the simulated activity, and for many applications a range of motion of around 20 degrees in pitch and roll is enough. Some applications include: a racing simulator, roller coaster simulator and a basic flight simulator. On the other hand, devices capable of producing a large range of motion say, 360 degrees in pitch roll and yaw, are typically only utilized in advanced flight trainers. The high cost of these devices restricts their use to instances where it is economical. For the above example, training a pilot can be dangerous and is very expensive so the role of the simulator is to reduce the total time the pilot needs to actually fly therefore reducing the total cost and risk of training. The Yaw VR style falls in between these two. Providing a higher range of motion than most basic motion simulators and a cost that is typically higher but still competitive. To achieve the large range of motion the simulator relies on a complex mechanism. It was determined that creating a design that utilized the Yaw VR style and still meet the original goal of three degrees of freedom would most likely fall outside of the budget and time constraints.

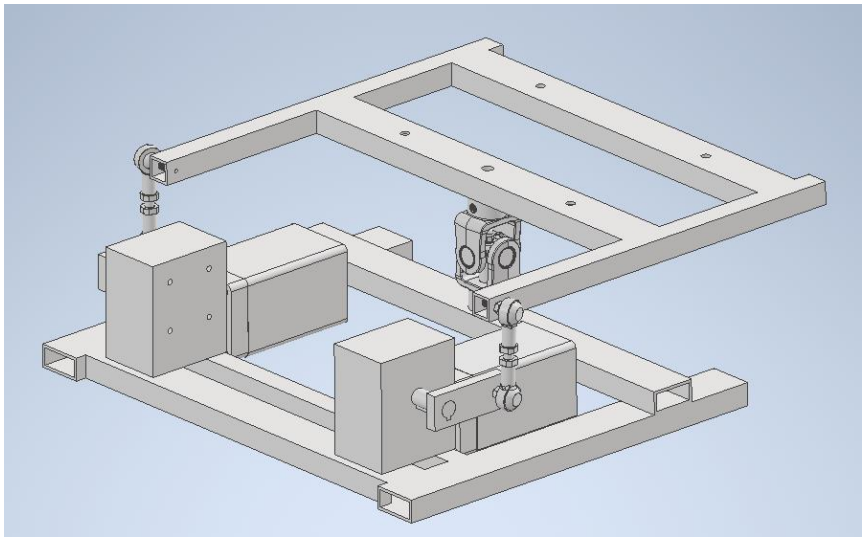


Figure 6: Pitch and Roll Platform Isometric View

The DOF Reality style of producing motion typically consists of two frames that are connected by a universal joint and two motors that move the top frame by each rotating a different arm that is perpendicular to the shaft. The arms are typically connected to the top frame via tie rods. Shown above in **Figure 6** is the final design of the pitch and roll platform which was based on designs that fall into the DOF Reality style category.

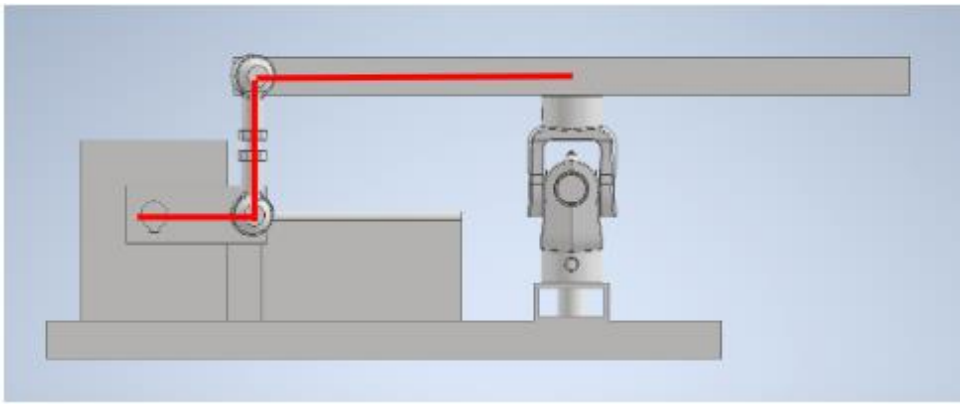


Figure 7: Pitch and Roll Platform Side View

A large part of the design process was dedicated to the top frame and shaft arm. These two components play a significant role in the torque needed from the motors, the speed of rotation, and the range of motion. By looking at the pitch and roll platform from the side it can be observed that the mechanism resembles a 3-bar linkage. Shown in **Figure 7** are the three bars used in the analysis. Since the positions and velocities of each bar are related, an analysis was done to determine this relation and how it changes with increasing or decreasing the lengths of the bars. Using desired values for speed and range of motion a rough idea of the geometry of the mechanism could be determined. Using this model, forces were applied to the top frame which simulated the weight of a passenger as well as torque values from the motors.

An analysis was performed to determine how much torque was needed from the motors and to get an idea of the speed. The analysis was performed with the device in its most extreme positions in pitch and roll and it was assumed that the motors would be operating at our determined max speed at this instance. Due to the geometry of the mechanism the forces transferred from the motor are the lowest at its extreme positions. Additionally, by assuming the motors are operating at max speed the lowest torque value from the motor is used due to the inverse relationship between torque and rpm. This situation is not realistic because the motors would never be at max speed at the most extreme position. The power of the motor is intentionally underestimated. The reasoning is that if the motors can provide enough torque in the worst-case scenario, geometry constraints are the greatest and torque from motor is the least, then the motors would work in normal operation. For this design having slightly more torque than needed is much better than not having enough. Calculations can be found in the appendix.

The risk of damage to the frame is very little as the forces that would oppose the motors would not change and since the motors can be limited to a certain speed the overall stress on the frame is not changed with the increased torque value. In the case of not enough torque the motors would not be able to support the top frame and it would not even work as a normal chair much less a motion simulation chair. From this analysis it was determined that two 9Nm stepper motors with a 5:1 speed reducer would be adequate for our application.

The next step in the design was to incorporate the correctly sized key components and motors into a full design. A top and bottom frame were created using the information found previously. The bottom frame was fairly simple. Its only functions are to provide a place for the motors and universal joint to mount as well as connecting to the pitch and roll platform. The top frame took the size requirements from the 3-bar linkage analysis and added cross members in order to mount the chair and universal joint. It was determined that the universal joint and rod ends were to be ordered from a vendor. Comparing the forces that were expected to act on the pitch and roll platform to the listed strengths of the parts allowed easy selection. Ultimately, the universal joint and rod ends were purchased from McMaster-Carr.

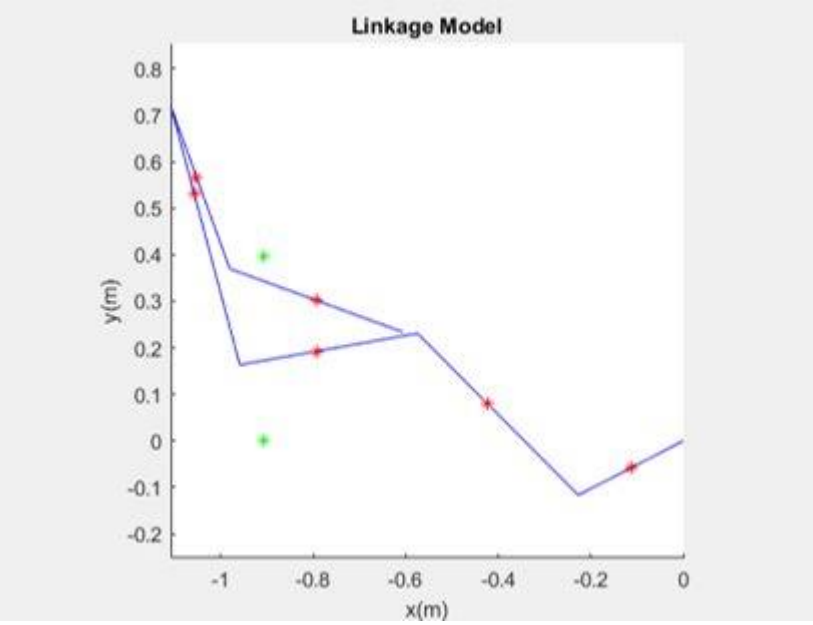


Figure 8: Body Linkage Model MATLAB Diagram

Much of the design was based on the assumption that the center of gravity of the rider would be located over the universal joint. This is key because it reduces the required torque of the motors and net force acting on the frame. Much time was devoted to locating the chair position to ensure the center of gravity of the person would rest over the universal joint. Using biometrics data on the average lengths and weights of body parts first hand calculations were done to try and get an accurate estimation of the location. Later on, a MATLAB code was developed in which the position of the person could be more easily adjusted.

During the manufacturing process some design changes were made. The most major modifications were changes in the location of crossmembers on the top frame, a new mounting method for the universal joint and the thickness of the shaft arm. The exact dimensions of the seat mounting bracket were not given from the manufacturer so once we received the part the locations of the crossmembers on the top frame were adjusted in order to align with the mounting holes on the seat.

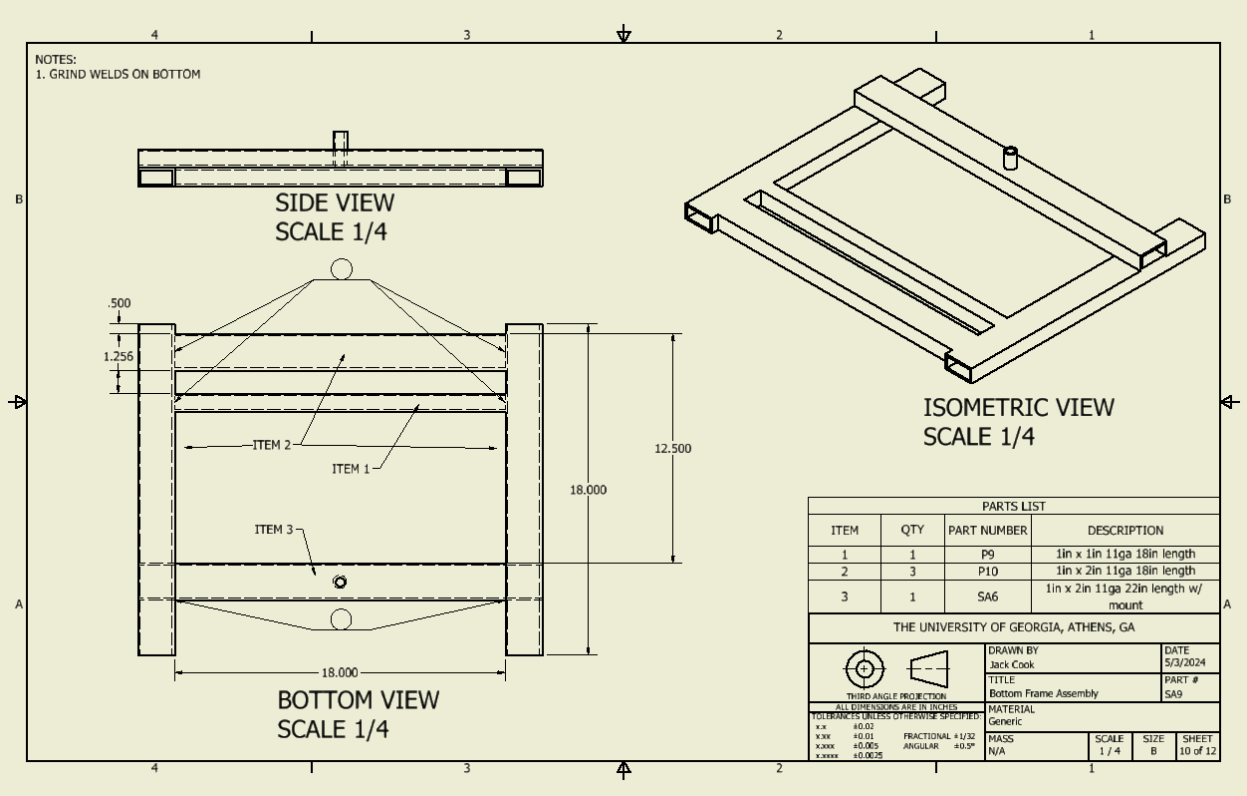


Figure 9: Pitch and Roll Bottom Frame

Initially the universal joint was going to be welded to the frames. This could provide a secure connection but then the frames would be inseparable without cutting the weld. After talking to George in the machine shop it was determined that welding would be challenging as the universal joint metal is a lot thicker than the tube steel. This could lead to warping or burning through the tubing making up the frames. Even if these problems were avoided it was questioned if adequate penetration would be achieved. This idea was scrapped and a new method was made. Using a steel rod that was hollow in the center two small shafts would be made. A hole placed in

the top and bottom frames would seat part of the shaft. The shafts would then be welded to the frames. A section of shaft 1.2 inches would stick out of the frames and would allow the u-joint to mount using the shaft connection that it was built with. This allowed the frames to be separated and solved the manufacturing issue. Shown above are updated manufacturing drawings reflecting the changes.

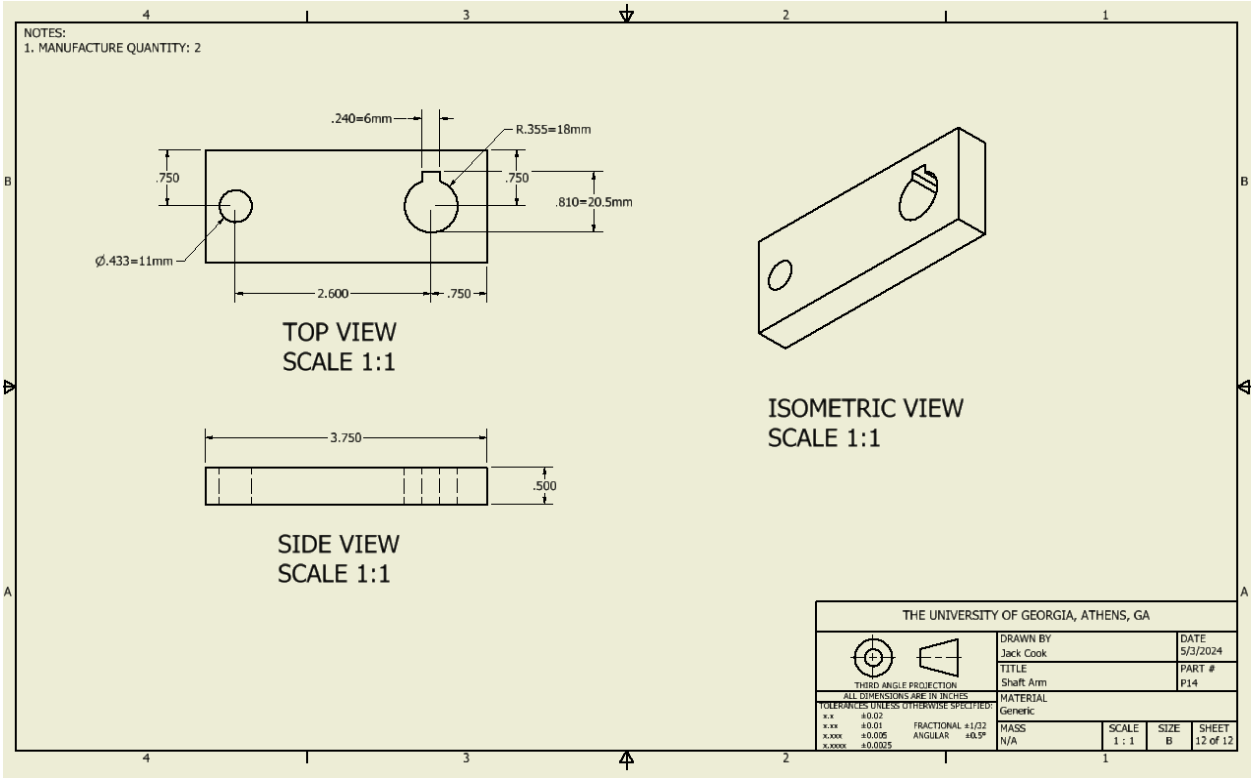


Figure 10: Shaft Arm

The original arm was made from leftover 11ga steel. Calculations were performed on the original design and it was determined it could withstand the forces required. Concerns were brought up about the part twisting. This was not considered in the original analysis. Enough ½ inch bar stock was left over from the manufacturing of a different part so it was used instead of the 11ga. The shape stayed the same aside from the removal of rounded edges as this would increase the amount of work needed for the parts. The increased thickness greatly improved the strength of the part and allowed the rod ends to more securely connect.

Conclusion

Overall, the pitch and roll platform was successfully able to move a rider in pitch and roll. It proved to be robust and provided the rider with an enjoyable and safe experience.

Mechanical Design: Yaw Platform

The third modular platform is the yaw platform located underneath the pitch and roll platform that rests on the floor. The main purpose of the yaw platform is to provide rotational motion in the form of yaw simulating the motion of flat spinning clockwise and counterclockwise about the vertical y-axis. The considerations that were taken into account in the platform's design include

maximizing rotational degree range, achieving structural safety factor, providing stability to prevent tipping, reducing components cost, creating an easily manufacturable product, reducing platform's weight, ability to replace and upgrade components, avoiding possible damages to the surface while in use, and limiting dimensional footprint. With all these considerations and research, we narrowed down to two design concepts and completed a decision matrix (**Figure 11**) to make our final decision on which to pursue. Both concepts scored very similarly, so the final decision was to develop the turn table version because of its range of motion and novelty to motion simulation chairs.

Yaw Platform Decision Matrix		
Criteria and Weight	Designs and Scoring (1-5)	
	Design 1: Turn Table Yaw Platform	Design 2: Rail Yaw Platform
Manufacturability (x4)	8	12
Range of Motion (x3)	15	6
Acceleration of Motion (x2)	6	8
Smoothness of Motion (x3)	12	12
Modularity and Assembly (x4)	12	16
Dimension Footprint (x2)	10	6
Weight (x1)	4	3
Cost (x5)	10	15
Total Score	77	78

Figure 11: Yaw Platform Design Decision Matrix

With the current motion simulation chairs on the market consumers are required to make the choice of sacrificing the range of yaw motion or saving money. In our prototype we wanted to incorporate both by achieving 360 degrees of yaw motion. After researching ideas to achieve this outside of current motion simulation chairs, inspiration came from the tea cup rides of Disney World and turntables to display cakes. The main design aspect was to split the platform into a top and bottom section where the bottom section would stay stationary while the top section would rotate about a central pivot or shaft. However, in order to avoid damaging the motor and gearbox that would be connecting the sections, there must not be any vertical load placed on the shaft. The design must distribute the vertical load of the user and other platform away from the shaft using supports.

The first iteration was to configure the top section as a lid that would rest on a ball bearing track built into the bottom section as seen in **Figure 12** and **Figure 13**. Problems with manufacturability of this design rose after speaking with Trevor Bowden and George Haynie. They informed me that Driftmier's machine shop was not capable to fabricate the parts required for the design.

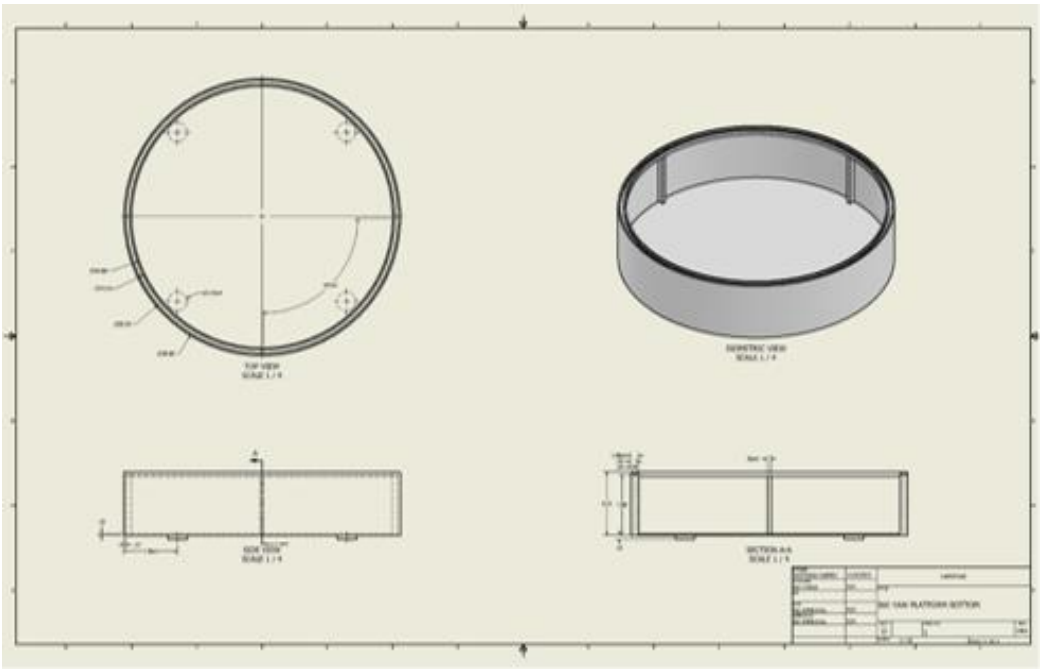


Figure 12: Yaw Iteration 1 Bottom

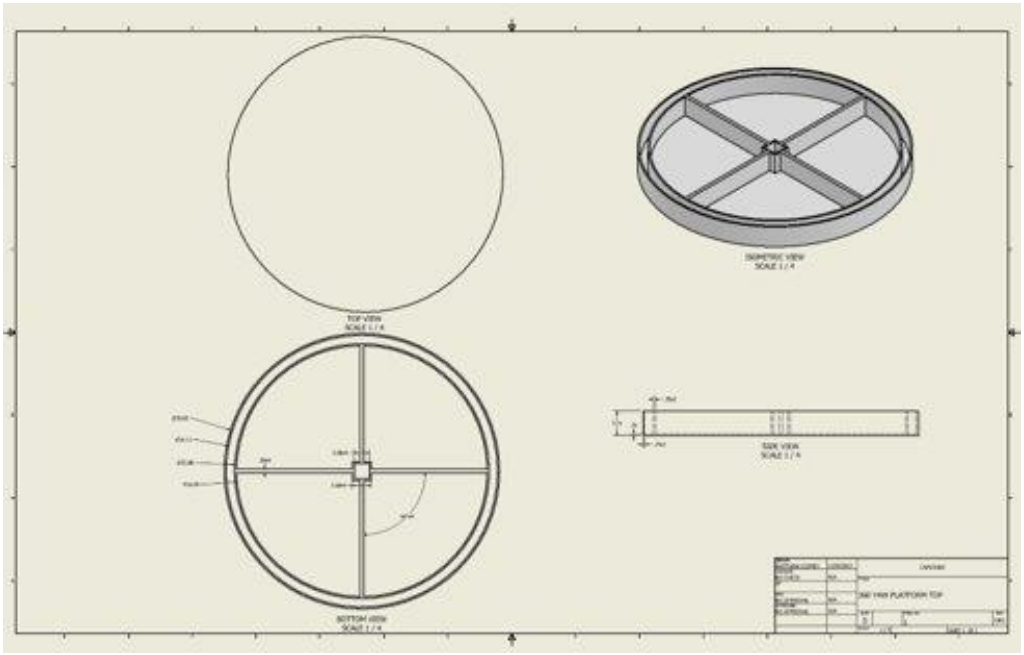


Figure 13: Yaw Iteration 1 Top

Iteration two switched from a ball bearing track design to a caster track design as seen in **Figure 14**. With this design, all the components were standard metal parts supplied by vendors which reduced the cost and increased the manufacturability.

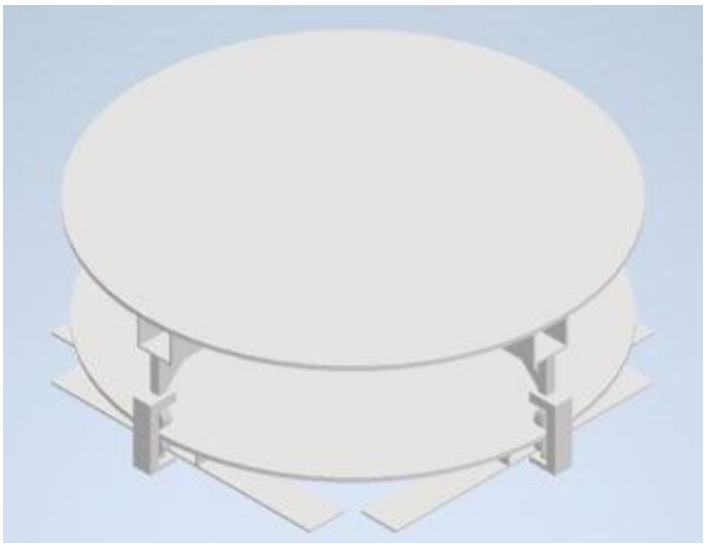


Figure 14: Yaw Platform Isometric Render

In order to ensure the design would structurally support the loads encountered while in use stress analysis were conducted through Autodesk Inventor. The stress analysis revealed iteration 2 was overdesigned with successfully passing the stress analysis with enormous safety factors. The overdesign was due to the thickness of the selected metal parts. Replacing the current parts with thinner alternatives would reduce weight, reduce cost, and increase manufacturability while still providing the structural strength needed.

Iteration three with the thinner parts was subjected to the stress analysis simulating the vertical load placed by a user and other platform as seen in **Figure 15**. The results showed a maximum displacement of 0.00327in located at the very edge of the top platform with a minimum safety factor of 10.11. After analyzing the results of the stress analysis test, we concluded that iteration 3 successfully passed with the preferred safety factors.

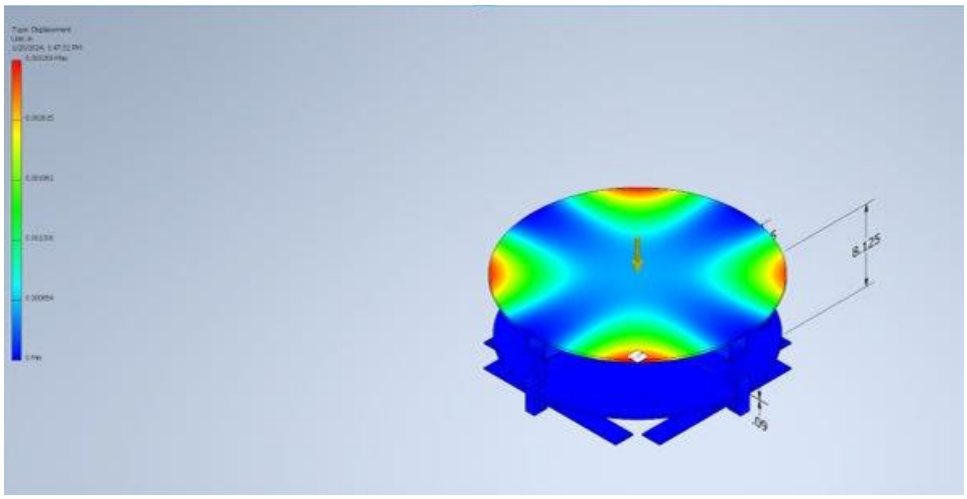


Figure 15: Yaw platform FEA-1

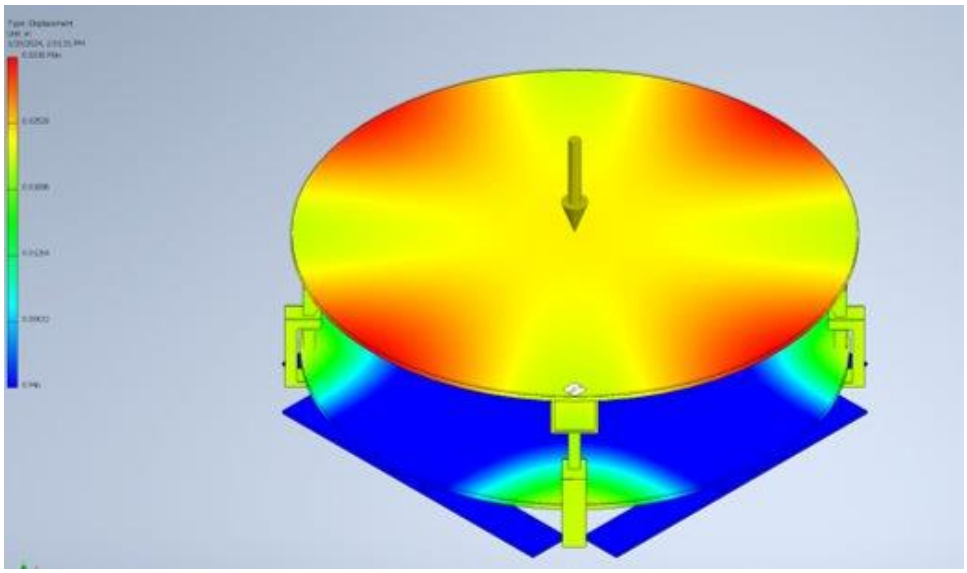


Figure 16: Yaw Platform FEA-2

In order to increase the stability of the chair, flat bar metal pieces were added to the bottom section's crossmember to act as stabilizers as seen in **Figure 17**. These stabilizers increase the bases surface area in order to distribute the loads that would cause tipping. This distribution decreases the likeliness of tipping. Concerns were raised about these metal stabilizers damaging the floor, so 2 rubber adhesive pads were added to each stabilizer.

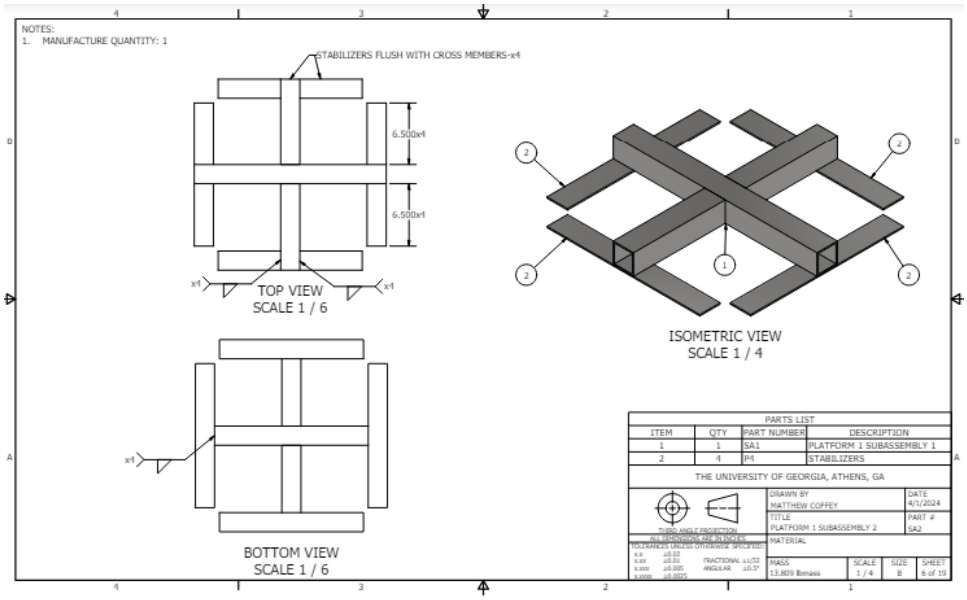


Figure 17: Bottom Crossmember Subassembly

During manufacturing, minor flaws were addressed causing modifications needing to be made along the way.

First modification was the caster assembly and bottom section design. Iteration 3 did not incorporate the casters purchased causing fitment issues. Changes can be seen in **Figures 18-21.** The larger caster radius required elevating the bottom section an additional 2 inches with rectangular tube steel. The larger caster radius required a redesign of the caster assembly too.

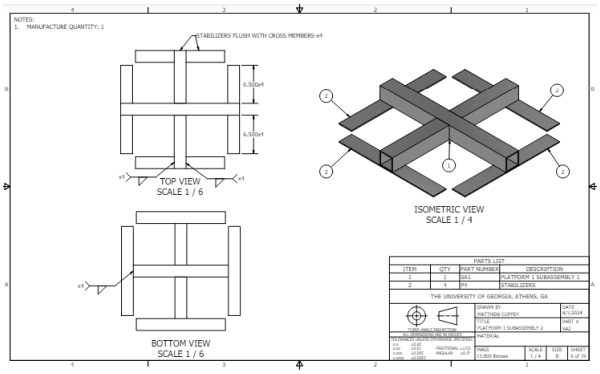


Figure 18: Iteration 3 Bottom Crossmember Subassembly

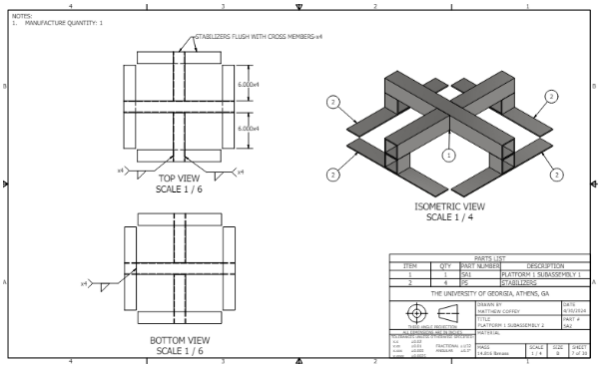


Figure 19: Modified Bottom Crossmember Subassembly

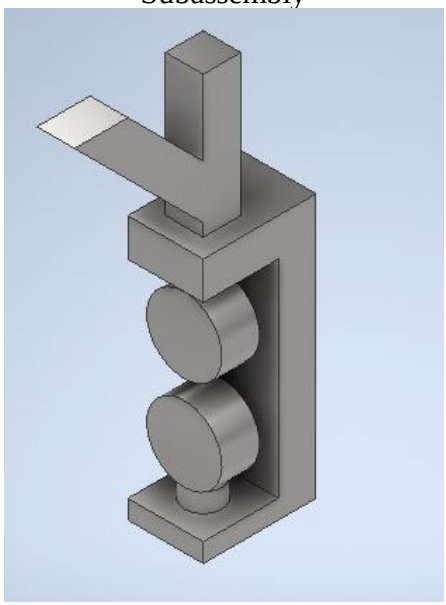


Figure 20: Caster Assembly Iteration 3

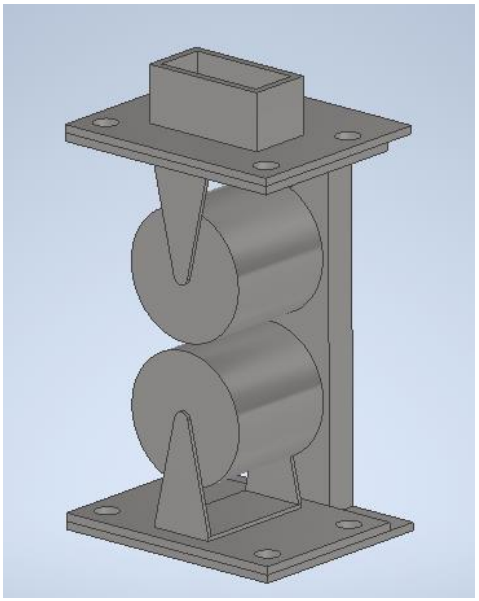


Figure 21: Modified Caster Assembly

Second modification was thickening the connection point between the top section and motor shaft. In order to increase the thickness of the junction a ½" bar was welded at the connection point as seen in **Figure 22** and **Figure 23.**

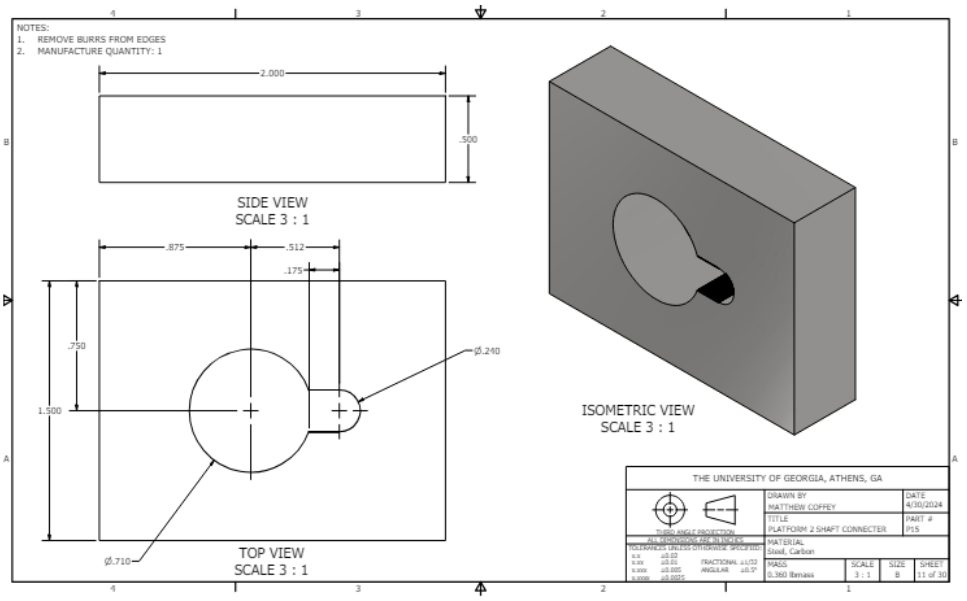


Figure 22: ½" Motor Connection Piece

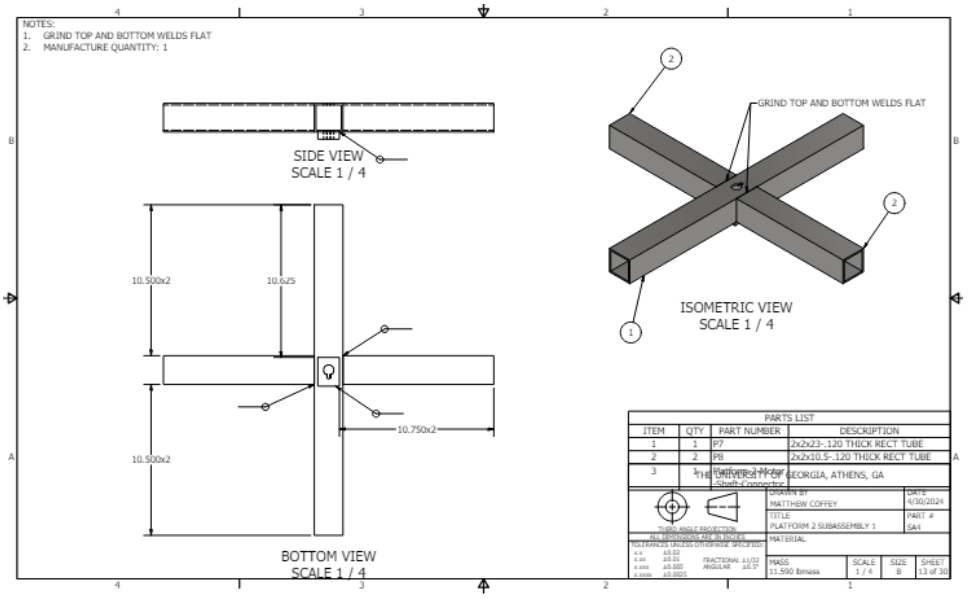


Figure 23: Modified Top Crossmember Subassembly

The final manufactured yaw platform is shown in **Figure 24.**

Figure 24: Yaw Platform Final Construction Side Profile

Electrical Design

The purpose of the electrical design for this project is to act as a bridge between the mechanical frame and the software aboard the Arduino MEGA. With this in mind the electrical design had two major requirements, meet the torque requirements to move the frame and have an easy interface with the MEGA. We found that of all the various motor types, linear actuators and stepper motors offered the greatest promise for this project. We created a decision matrix to make an informed decision on which of these two to choose.

Actuator Types		
Criteria and Weight	Designs and Scoring (1-5)	
	Actuator 1: Linear Actuator	Actuator 2: Motor and Gearbox
Range of Motion (x3)	9	12
Acceleration of Motion (x2)	6	6
Smoothness of Motion (x3)	12	12
Modularity and Assembly (x4)	20	12
Dimension Footprint (x3)	12	9
Power Consumption (x1)	4	5
Nominal Noise Level (x2)	4	8
Cost (x5)	10	25
Total Score	77	89

Figure 25: Motor Design Decision Matrix

As discussed earlier, this decision matrix led us to select NEMA 34 9Nm stepper motors. These motors also conveniently come in simple kits that only require power to get up and running, so the convenience and performance were exceptionally appealing.

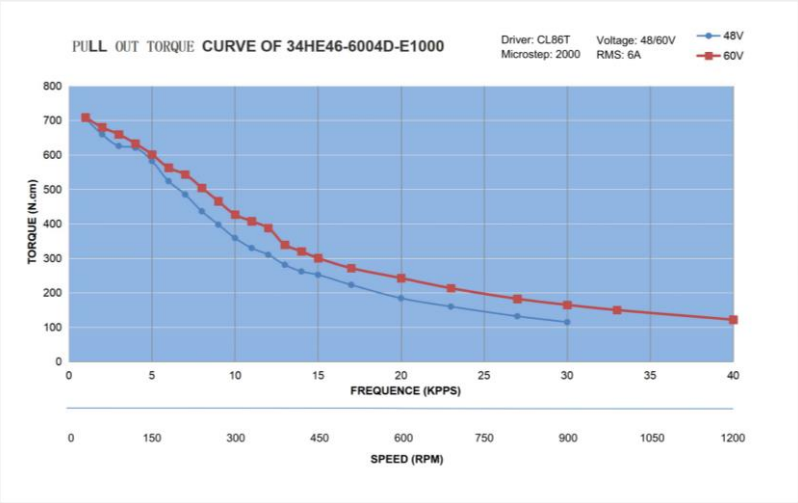


Figure 26: Stepper Motor Torque Turve

This graph shows us the characteristic torque curve for these motors. This allowed us to determine using mechanical simulations, that with a gearbox reduction of 5:1, any speed under approximately 400 RPM would output enough torque for the chair to move freely with a person's weight applied. So, we then calculated how fast we could theoretically move the motors given our MEGA.

$$360\left(\frac{(20000)(10)}{(5)(200)(3)(60)}\right) = 400\ RPM$$

Figure 27: Maximum Motor Speed Calculation

This calculation shows that if we step at 20000 steps per second, which is out hard output limitation, then the motor can be spun at 400 RPM. This means that even if we spun a single motor as fast as our hardware allows, it would still be able to move the chair theoretically. This then led us to finding an appropriate gearbox to attach to this motor to get the 5:1 reduction we were looking for to increase torque output. More information on this is found in the mechanical design section.

Our driver selection was simple as there are not many options when it comes to compatible motor drivers. This is because the NEMA 34 motor is closed-loop, which helps increase its reliability and efficiency but also limits its driver compatibility to ones with the proper feedback connections. The driver and motor were even sold as a kit, so it was decided to purchase the kit that included a CL86T closed-loop stepper motor driver.

Figure 28: CL86T Product Image

This motor driver also easily allowed us to interface the MEGA with the drivers. All that is required is for a pulse train to be send along the PUL+ pin to step the motor, with a simple single timed pulse to indicate the direction to step in. We cross-verified that this driver would be compatible with our motors and the MEGA and began to look into its power requirements in its datasheet.

2.1 Electrical Specifications

Parameters	Min	Typical	Max	Unit
Peak Current	5.6A (RMS 4A)	7A (RMS 5A)	8A (RMS 5.7A)	A
Operating Voltage	18 24	-	80 110	VAC VDC
Logic input signal current	7	10	20	mA
High speed pulse input frequency (5V)	0	-	500	kHz
Pulse input frequency (24V)	0	-	200	kHz
Input signal voltage	5	-	24	VDC
Logic current output	-	-	100	mA

Figure 29: CL86T Datasheet Specifications

Taking into account the max current on medium gain settings (7A) and using the voltage-torque curve we based our calculations off of (48V), we can assume that the maximum power one driver can consume, at max load, cannot exceed 336W. This informed us that we needed a PSU of at least 350W to power each of these drivers or one large PSU that is greater than 1008W.

$$P_{drivers-max} = (7)(48)(3) = 1008W$$

Figure 30: Maximum Wattage Calculations

In our research we found that buying three separate PSUs would be more budget friendly despite having a lower overall power density, and three 350W PSUs provides us with a reasonable 1050W in total.

Figure 31: 350W PSU Product Image

This power supply is from the same company as the stepper motors and drivers and advertised recommended compatibility with those two products. It also outputs the 48V we need at 7.3A, which is more power than the 7A we need at most. With the power supply selected all we needed was to pick rated cabled and connectors based off of each one's required ampacity, as discussed in the engineering specifications section.

Software Design

Objectives

The primary objective of the software design for our Motion Simulation Chair was to create an intuitive and responsive system that translates user input into smooth, realistic motion experiences. The software needed to accomplish the following:

1. Control: Accurately control three degrees of freedom (pitch, roll, and yaw) using stepper motors.
2. Usability: Provide a user-friendly interface for the user to select different motion experiences.
3. Flexibility: Allow for customization and multiple modes of operation.
4. Safety: Ensure that the chair operates within safe parameters.

Methodology:

To achieve these objectives, the following methodology was used:

1. Stepper Motor Control:
    - The "FastAccelStepper" library was used for control of the three stepper motors. This library allowed us to achieve precise control over the motors, including setting speed and acceleration.

- The software controls three separate motors. Two of which control pitch and roll, and the last controlling yaw.
- The following code snippets below shows how the engine and motors are set up.

```
28      // Create the engine and the steppers
29      FastAccelStepperEngine engine = FastAccelStepperEngine();
30      FastAccelStepper *stepper1 = nullptr;
31      FastAccelStepper *stepper2 = nullptr;
32      FastAccelStepper *stepper3 = nullptr;
33
```

Figure 32:

```
65      // Initialize steppers (stepper1 and stepper2 are used for pitch and roll and stepper3 is used for yaw)
66      stepper1 = engine.stepperConnectToPin(STEPPER1_STEP_PIN);
67      if (stepper1) {
68        stepper1 -> setDirectionPin(STEPPER1_DIR_PIN);
69        stepper1 -> setSpeedInHz(1000); // steps/sec
70        stepper1 -> setAcceleration(1000); // steps/sec^2
71      }
72
73      stepper2 = engine.stepperConnectToPin(STEPPER2_STEP_PIN);
74      if (stepper2) {
75        stepper2 -> setDirectionPin(STEPPER2_DIR_PIN);
76        stepper2 -> setSpeedInHz(1000); // steps/sec
77        stepper2 -> setAcceleration(1000); // steps/sec^2
78      }
79
80      stepper3 = engine.stepperConnectToPin(STEPPER3_STEP_PIN);
81      if (stepper3) {
82        stepper3 -> setDirectionPin(STEPPER3_DIR_PIN);
83        stepper3 -> setSpeedInHz(1000); // steps/sec
84        stepper3 -> setAcceleration(1000); // steps/sec^2
85      }
```

Figure 33:

2. State Machine:
    - The core logic of our software was based on a state machine. This approach enabled us to handle various stages of user input and motor control cleanly and systematically.

```
34    // Enumeration of states to structure the simulation process
35    enum State {
36      WAIT_FOR_INPUT,
37      WAIT_FOR_AXIS_CHOICE,
38      WAIT_FOR_SPEED_AND_ACCELERATION,
39      WAIT_FOR_POSITION,
40      PROCESSING
41    };
```

Figure 34:

```
45    // Start at the first state
46    State currentState = WAIT_FOR_INPUT;
```

Figure 35:

```
100   /**
101    * Function to continuously loop through the enumeration states
102   */
103   void loop() {
104     switch (currentState) {
105       case WAIT_FOR_INPUT:
106         getExperience();
107         break;
108       case WAIT_FOR_AXIS_CHOICE:
109         getAxis();
110         break;
111       case WAIT_FOR_SPEED_AND_ACCELERATION:
112         getSpeedAndAcceleration();
113         break;
114       case WAIT_FOR_POSITION:
115         getPosition();
116         break;
117       case PROCESSING:
118         if (choice == "1") {
119           moveMotor(positionInSteps);
120
121         } else if (choice == "2") {
122           performFullExperienceMotion();
123
124         } else if (choice == "3") {
125           performRollerCoasterSimulation();
126         }
127         break;
128     }
129   }
```

Figure 36:

3. User Interface:
    - The user interface was implemented through the serial monitor, where the user could select different motion experiences and input parameters such as speed and acceleration.
    - The following code snippet, showing the “getExperience()” function, illustrates how user input is handled.

```
/**
 * Function to obtain the kind of experience that the user would like
*/
void getExperience() {
  Serial.println("What kind of user experience would you like?");
  Serial.println("1: Move in one direction");
  Serial.println("2: Full experience of motion");
  Serial.println("3: Roller coaster simulation");
  Serial.println();

  // Get the desired movement
  while (!Serial.available()) {
  }
  choice = Serial.readStringUntil('\n');
  choice.trim();

  // If the user provided 1, change the state to WAIT_FOR_AXIS_CHOICE
  // If the user provided 2 or 3, change the state to PROCESSING
  // If the user didn't provide 1, 2 or 3, do not change the state so that the program can ask the user again
  if (choice.equals("1")) {
    currentState = WAIT_FOR_AXIS_CHOICE;

  } else if (choice.equals("2") || choice.equals("3")) {
    currentState = PROCESSING;

  } else {
    Serial.println("Invalid choice. Please enter 1 for single direction or 2 for full experience.");
    Serial.println();
  }
}
```

Figure 37:

4. Motion Profiles:
    - The software included several motion profiles, including single-direction movement, a full motion experience, and a roller coaster simulation. Each profile was designed to provide a different type of motion experiences.
    - The following code snippet shows the “performRollerCoasterSimulation()” function, which simulates a roller coaster experience for the user.

```
/**
 * Function that mimics what it would be like to be on a roller coaster
*/
void performRollerCoasterSimulation() {
  Serial.println("Starting Roller Coaster Simulation...");

  // Begin with a slow climb to the first peak
  Serial.println("Climbing to the first peak...");
  slowClimb();

  // Level out at the first peak
  Serial.println("Leveling out at the peak...");
  levelOut();

  // The first major drop
  Serial.println("Descending the first major drop...");
  fastFall();

  // Level out at the bottom
  Serial.println("Leveling out after the drop...");
  levelOut();

  // A series of smaller hills and dips
  Serial.println("Navigating smaller hills...");
  for (int i = 0; i < 4; i++) {
    smallHill();
    gentleDip();
  }

  // Level out after last dip
  Serial.println("Leveling out after hills and dips...");
  levelOut();

  // Introduce a sharp turn
  Serial.println("Executing a sharp turn...");
  sharpRightTurn();
  sharpLeftTurn();
  stepper3 -> moveTo(0, true);

  // Simulate a fast climb to another peak
  Serial.println("Fast climbing to another peak...");
  fastClimb();
```

Figure 38:

```
480       // Level out at the second peak
481       Serial.println("Leveling out at the second peak...");
482       levelOut();
483
484       // Final major fall
485       Serial.println("Descending the final major fall...");
486       fastFall();
487
488       // Level out at the bottom to end the ride
489       Serial.println("Leveling out to end the ride...");
490       levelOut();
491
492       // End of the simulation
493       Serial.println("Roller Coaster Ride Complete!");
494
495       // Go back to getting a new user desired movement
496       currentState = WAIT_FOR_INPUT;
497     }
```

Figure 39:

5. Safety Features:
    - The software included safety checks to ensure that the motors did not exceed specified limits, and that the user could not input invalid or unsafe parameters.

```
14    #define MAX_SPEED 10000
15    #define MAX_ACCELERATION 10000
16    #define MIN_SPEED 100
17    #define MIN_ACCELERATION 100
18
19    #define MAX_POS_PITCH_POSITION 83 // 30/(360/(5 * 200))
20    #define MAX_NEG_PITCH_POSITION -83
21
22    #define MAX_POS_ROLL_POSITION 83 // 30/(360/(5 * 200))
23    #define MAX_NEG_ROLL_POSITION -83
24
25    #define MAX_POS_YAW_POSITION 55 // 20/(360/(5 * 200))
26    #define MAX_NEG_YAW_POSITION -55
```

Figure 40:

Issues and Improvements

1. Rumble Function:

- The rumble feature, which simulated a shaking motion, was finicky and inconsistent when combined with other movement functions. This issue could be addressed in future iterations by optimizing the motor control during rapid direction changes.

2. Motion Smoothness:
    - While the chair's motion was generally smooth, further improvements could be made by optimizing the acceleration profiles and using feedback from the motors to adjust movements dynamically.
3. User Interface:
    - The serial-based interface, while functional, could be improved with a more user-friendly graphical interface or a dedicated control panel.

Code Flow

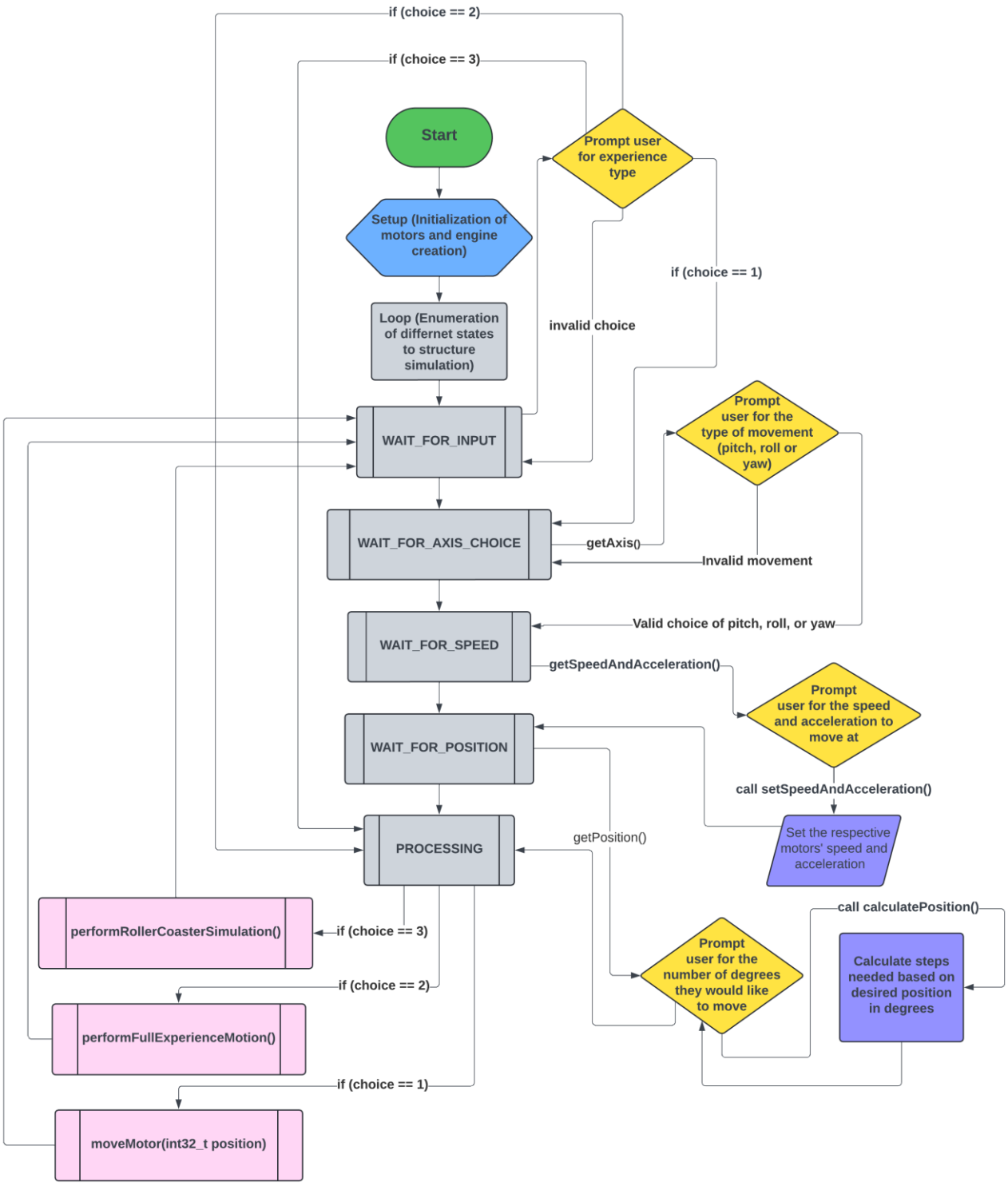


Figure 41: Main Program Code Flowchart

Github

https://github.com/SeanSchlief/MotionSimulationChair

Conclusion

Overall, the software successfully met the project's objectives. It provided robust and flexible control over the chair's motion, supported multiple motion experiences, and ensured safety throughout. The issues encountered provided valuable learning opportunities and pointed towards potential future improvements for an even better motion simulation experience.

Bill of Materials

The final bill of materials, at the time of this report, came out to a total cost of $1436.48 for all of the materials. This is under the budget we were given from Dr. Thomas and we were excited to see that our budget-oriented yet performance-uncompromising design philosophy paid off in this manner.

Item Number	Part Number	Manufacturer	Part Description	Supplier	Quantity	Unit Cost	Total Cost	Website Link	Purpose
1	34HS46-6004D-E1000	STEPPERONLINE	Motor and Driver Kit	STEPPERONLINE	3	$ 94.61	$ 283.83	Link	Provides power for the chair to move
2	NMRV40-G5-D14	STEPPERONLINE	Gearbox	STEPPERONLINE	3	$ 41.56	$ 124.68	Link	Lowers motor RPM for more torque
3	6452K1	McMaster Carr	Universal Joint	McMaster Carr	1	$ 115.00	$ 115.00	Link	Connects bottom and top frames
4	6275K55	McMaster Carr	Rod Ends	McMaster Carr	4	$ 16.97	$ 67.88	Link	Connects gearboxes to frame
5	T12111	Metals Depot	2 X 1 X 11 GA (.120 wall)*2ft	Metals Depot	6	$ 13.17	$ 79.02	Link	Frame construction material
6	T11111	Metals Depot	1 X 1 X 11 GA (.120 wall)*2ft	Metals Depot	3	$ 9.20	$ 27.60	Link	Frame construction material
7	TBD	TBD	Fasteners	Driftmier Machine Shop	~12	$ -	$ -	N/A	Fastners used to mount motor and gearbox
9	F212112	Metals Depot	.5"x.5"x4' Flat Bar HR Steel	Metals Depot	1	$ 24.64	$ 24.64	Link	Steel frame pieces to create casters
11	C0020120ZN-POL01(KK)B	Shepherd Caster	Swivel Plate Caster: 2 in Wheel Dia., 300 lb, 2 13/16 in Mounting Ht, Polyolefin	Grainger	8	$ 13.34	$ 106.72	Link	Wheels for yaw platform
10	S111	Metals Depot	1'x2'-11GA Thick High Carbon Steel Sheet	Metals Depot	1	$ 27.50	$ 27.50	Link	Steel frame pieces used as augers to prevent tipping
13	S111	Metals Depot	2'x4'-11GA Thick High Carbon Steel Sheet	Metals Depot	1	$ 95.00	$ 95.00	Link	Steel frame pieces used to create 24" and 23.25" diameter plates for the yaw platform
14	T12211	Metals Depot	2"x2"x8'-11 GA Thick High Carbon Steel Rectangular Tube	Metals Depot	1	$ 64.36	$ 64.36	Link	Steel frame pieces for the yaw platform
15	A000062	Arduino	Arduino DUE	Amazon	1	$ 52.00	$ 52.00	Link	Controlls the motor drivers
16	S-350-60	STEPPERONLINE	Power Supply	STEPPERONLINE	3	$ 25.72	$ 77.16	Link	Powers the motor system
17	4800J	CCCEI	Power Strip	Amazon	1	$ 34.39	$ 34.39	Link	Connects subsystems to wall outlet
18	RV40-AS	STEPPERONLINE	Gearbox Output Shaft	STEPPERONLINE	3	$ 4.10	$ 12.30	Link	Connects gearbox to rod ends
19	B08NP5YHX2	SalPhines	Blade Wall Plug	Amazon	3	$ 2.33	$ 6.99	Link	Connects power supplies to power strip
20	B089CZFDX2	Fermerry	26AWG Multicolor Kit	Amazon	1	$ 10.19	$ 10.19	Link	Wires DUE to motor drivers for control
21	B089CXJPSC	Fermerry	20AWG Multicolor Kit	Amazon	1	$ 23.49	$ 23.49	Link	Wires power supplies to motor drivers
22	B0B8CBFK8K	Matugajp	14AWG 3-Core Wire	Amazon	1	$ 24.99	$ 24.99	Link	Wires power supplies to blade plugs
23	N/A	Arduino	Arduino IDE	Arduino	1	$ -	$ -	Link	Coding environment for DUE
24	B0006HKK3Y	RCI	Bucket Seat	Summit Racing	1	$ 92.99	$ 92.99	Link	Seats the user on the platform
25	Y-G-FJKRED-S	Yakitoko	Safety Harness	Amazon	1	$ 39.79	$ 39.79	Link	Secures user in the seat
26	N/A	Autodesk	Autodesk Inventor	Autodesk	1	$ -	$ -	Link	CAD application used to design frame
27	AQ-17	OIKWAN	USB to RS232 Serial Adapter	Amazon	3	$ 11.99	$ 35.97	Link	RS232 to PC cable to read motor performance values
28	00920	LitStar	DUE Power Plug	Amazon	1	$ 9.99	$ 9.99	Link	Powers the DUE
					Total Price:		**$1,436.48**		

Figure 42: Finalized BOM for the Project

All of these components are off-the-shelf and require no specialty licenses or storefronts in order to obtain them. This, in conjunction with our GitHub, electrical schematic, and mechanical drawing, makes our design as open-source and replicable as we set it out to be.

Evaluation

Pitch and Roll Platform Testing

Static Test

Once the pitch and roll platform was assembled tests were conducted to ensure proper function. The first test was a stationary test in which a rider would sit in the chair with the motors powered on. The goal of this test was to determine if the systems were working properly and if the motors would keep their position when stationary. It was successful as the top frame did not move despite the added weight. The stepper motors were closely observed to ensure there was no slipping.

Movement Test

Figure 43: Pitch and Roll Weighted Testing Validation

During this test the pitch and roll platform, starting from the home position would move to its 4 maximum positions. Observations were made that with heavier riders returning to home position from pitch forward can cause motors to struggle. Sitting cross legged seemed to help this problem which pointed to a weight distribution issue. Observations made when the chair was moving in the roll direction noted that the motors are able to move the chair without issue. Since the center of gravity is naturally centered in the roll direction due to the nature of the design this led to the conclusion that the center of gravity was not centered in the pitch direction. The likely cause is that the center of gravity of the rider is too far forward as this would require more force to return to home position resulting in the motors struggling.

Reduced Range of Motion Movement Test

An additional test was conducted where the shaft arm range of motion was restricted to 30 degrees from home instead of 35 degrees. Due to the geometry of the device, the lowest available torque on the top frame is when the shaft arm is farthest from home. Reducing the range of motion decreased the frequency of the problem indicating an issue with weight distribution.

Yaw Platform Testing

Movement Test

With the full design assembled testing of the yaw platform was able to be performed. Similar to the pitch and roll testing a movement test was conducted in which the platform was rotated to its maximum position in both directions. It was observed that at some points the top platform would get caught on the base. This was due to an error during the manufacturing of a specific part. Grinding this down allowed the platform to spin easier. Additionally, observations were made regarding cable management.

Electrical System Testing

Validation Setup

With the electrical system completely connected the necessary checks and tests were conducted to ensure that all of the devices were functioning properly. Initially the debug LED on the PSUs were validated as working. Then the power indicators on the drivers were verified to be functioning appropriately. Lastly came the connection testing between the drivers and motors which first involved continuity testing every wire to ensure no shorts were present. Then, with the motor wires connected, the driver communication debug LEDs were read.

At first, we obtained a communication error which was indicative of a few possible problems. We quickly narrowed this down to two swapped connectors on a single motor driver by reading off the blink pattern and referencing the driver's datasheet. Then with all of the connections working as intended and all systems powered properly, we could move on to testing software on the system.

Software Evaluation

Experimental Setup

The evaluation of the software involved a series of tests to verify the functionality, performance, and user interaction aspects of the software that controlled the motion simulation chair. These tests were performed to measure the accuracy, responsiveness, and reliability of the software under different conditions.

Tests and Observations

User-Programmed Movement and Pre-Programmed Movements

- Procedure: The software offered three options to the user via a serial interface to select between customizable single-axis movement, a pre-programmed experience

for undergoing the maximum possible position limits for the three degrees of freedom, and a pre-programmed roller coaster experience.

- Observation
    i. Customizable Single-Axis Movement: Users could easily select and modify the extent and speed of the motion along any single axis (pitch, roll, or yaw). The software made sure the desired speeds, accelerations, and desired positions stayed within safe and capable parameters and accurately executed these custom commands.
    ii. Pre-Programmed Maximum Position Limits Experience: This mode was designed to demonstrate the chair's full mechanical range by sequentially moving to the maximum and minimum positions of each axis. The software performed this pre-programmed experience flawlessly, with transitions between axes being smooth and controlled. This mode was particularly useful for showcasing the chair's capabilities to new users.
    iii. Pre-Programmed Roller Coaster Experience: The roller coaster simulation was the most complex of the pre-programmed sequences, involving rapid, synchronized movements across multiple axes to simulate a roller coaster ride for the user. While it did deliver an exciting and immersive experience for the user, there were occasional discrepancies in motion synchronization when transitioning between complex maneuvers, especially when weights of approximately 190lbs and above were applied. Additionally, there were noticeable moments where the transitions between different movements felt abrupt or unexpectedly delayed. The existing delays between movement were sometimes too long or too short, ultimately disrupting the natural flow of a roller coaster ride.

Findings and Recommendations

- Findings:
    i. Interface: While the serial interface was completely functional, it was rather basic for users unfamiliar with serial communication. Ideally a graphical user interface (GUI) would have been more intuitive and could've provided an easier selection and adjustment and selection of the motion profiles.
- Recommendations for Future Improvement:
    i. Enhance User Interface: A GUI of some sort that would be able to run on a connected device or directly on a touchscreen panel attached to the chair would prove to be a great addition. Additionally, being able to directly interface with a video game, a racing simulator, flight simulator, etc. would further improve the immersive experience.
    ii. Fine-Tuning: Further reviewing the delay intervals within a controlled environment would allow for better determination of the optimal timings that better mimic natural movement dynamics.

Conclusions and Recommendations

The main objectives: simulating motion in three degrees of freedom, a sturdy design capable of moving a rider, quick movement, an adequate range of motion, a safe experience for the rider and preprogramed movement, have been accomplished.

Plans for future improvement would be focused on improving the mechanical design and the integration of user-controlled motion via interactions with a video game. In terms of the mechanical design a repositioning of the seat would allow for better weight distribution and prevent the motors from stalling. Additionally, an effort would be made to increase the stability of the chair to prevent it from rocking when in use. The user-controlled motion would provide a much more immersive experience for the rider and could potentially allow the chair to be used in training applications.

This project was a great opportunity to get experience working on a long-term project. All members of the group gained valuable teamwork and leadership skills. Throughout the duration of the project our team had to effectively organize and practice time management skills in order to stay on track. Overall, the capstone project was a great learning experience for all involved and we are proud of the things we were able to accomplish.

References

[1] "Wiring Diagram for Closed Loop Stepper Motor," Help Center, https://help.stepperonline.com/en/article/wiring-diagram-for-closed-loop-stepper-motor-4cddqy/.

[2] Helen, "DC Motor vs Stepper Motor vs Servo Motor – Which Motor Should you Choose for Your Project?," Latest Open Tech From Seeed, https://www.seeedstudio.com/blog/2019/04/01/choosing-the-right-motor-for-your-project-dc-vs-stepper-vs-servo-motors/.

[3] George, "Choosing a Stepper Motor Power Supply," Simply Smarter Circuitry Blog, https://www.circuitspecialists.com/blog/stepper-motor-power-supplies/.

[4] "Arduino Mega 2560 REV3," Arduino Online Shop, https://store-usa.arduino.cc/products/arduino-mega-2560-rev3?selectedStore=us.

[5] M. McCauley, "Accelstepper," AccelStepper - Arduino Reference, https://www.arduino.cc/reference/en/libraries/accelstepper/

[6] gin66, "FastAccelStepper," GitHub, https://github.com/gin66/FastAccelStepper

[7] "Yaw2 Pro Edition 3DoF," YAW Motion Simulator, https://shop.yawvr.com/product/yaw2-pro-edition-3dof-without-seat/.

[8] "DOF Reality H3 Consumer Motion Simulator Platform," DOF Reality, https://dofreality.com/motion-simulator-h3/#h3

[9] "QS-CH1," Qubic System, https://qubicsystem.com/product/qs-ch1/.

[10] McMaster-Carr, McMaster-Carr

Appendix

Full Code for MotionSimulationChair.cpp

```
#include "FastAccelStepper.h"

// Define pins for the motors
#define STEPPER1_STEP_PIN 6
#define STEPPER1_DIR_PIN 3

#define STEPPER2_STEP_PIN 7
#define STEPPER2_DIR_PIN 4

#define STEPPER3_STEP_PIN 8
#define STEPPER3_DIR_PIN 5

// These values can be tuned
#define MAX_SPEED 10000
#define MAX_ACCELERATION 10000
#define MIN_SPEED 100
#define MIN_ACCELERATION 100

#define MAX_POS_PITCH_POSITION 83 // 30/(360/(5 * 200))
#define MAX_NEG_PITCH_POSITION -83

#define MAX_POS_ROLL_POSITION 83 // 30/(360/(5 * 200))
#define MAX_NEG_ROLL_POSITION -83

#define MAX_POS_YAW_POSITION 55 // 20/(360/(5 * 200))
#define MAX_NEG_YAW_POSITION -55
```

```
// Create the engine and the steppers
FastAccelStepperEngine engine = FastAccelStepperEngine();
FastAccelStepper *stepper1 = nullptr;
FastAccelStepper *stepper2 = nullptr;
FastAccelStepper *stepper3 = nullptr;

// Enumeration of states to structure the simulation process
enum State {
  WAIT_FOR_INPUT,
  WAIT_FOR_AXIS_CHOICE,
  WAIT_FOR_SPEED_AND_ACCELERATION,
  WAIT_FOR_POSITION,
  PROCESSING
};

// Global variables //

// Start at the first state
State currentState = WAIT_FOR_INPUT;

// Pitch, roll, or yaw
String currentAxis = "";

// Desired movement
String choice = "";

// The # of steps to achieve a certain position
int32_t positionInSteps = 0;
```

```
/**
* Function to setup the Serial, steppers, and move the motors to the zero position if they are not already there
*/
void setup() {
  Serial.begin(4800);
  while (!Serial); // Wait for the Serial to start
  engine.init(); // Start the engine

  // Initialize steppers (stepper1 and stepper2 are used for pitch and roll and stepper3 is used for yaw)
  stepper1 = engine.stepperConnectToPin(STEPPER1_STEP_PIN);
  if (stepper1) {
    stepper1 -> setDirectionPin(STEPPER1_DIR_PIN);
    stepper1 -> setSpeedInHz(1000); // steps/sec
    stepper1 -> setAcceleration(1000); // steps/sec^2
  }

  stepper2 = engine.stepperConnectToPin(STEPPER2_STEP_PIN);
  if (stepper2) {
    stepper2 -> setDirectionPin(STEPPER2_DIR_PIN);
    stepper2 -> setSpeedInHz(1000); // steps/sec
    stepper2 -> setAcceleration(1000); // steps/sec^2
  }

  stepper3 = engine.stepperConnectToPin(STEPPER3_STEP_PIN);
  if (stepper3) {
    stepper3 -> setDirectionPin(STEPPER3_DIR_PIN);
    stepper3 -> setSpeedInHz(1000); // steps/sec
```

```
  stepper3 -> setAcceleration(1000); // steps/sec^2
 }

 // Move motors to the absolute 0 position
 stepper1 -> moveTo(0, true);
 stepper2 -> moveTo(0, true);
 stepper3 -> moveTo(0, true);
 stepper1 -> setCurrentPosition(0);
 stepper2 -> setCurrentPosition(0);
 stepper3 -> setCurrentPosition(0);

 // Begin the program
 Serial.println("Welcome  to our motion simulation  chair!");
 Serial.println();
}

/**
* Function to continuously loop through the enumeration states
*/
void loop() {
 switch (currentState) {
  case WAIT_FOR_INPUT:
   getExperience();
   break;
  case WAIT_FOR_AXIS_CHOICE:
   getAxis();
   break;
  case WAIT_FOR_SPEED_AND_ACCELERATION:
   getSpeedAndAcceleration();
```

```
      break;
    case WAIT_FOR_POSITION:
      getPosition();
      break;
    case PROCESSING:
      if (choice == "1") {
        moveMotor(positionInSteps);

      } else if (choice == "2") {
        performFullExperienceMotion();

      } else if (choice == "3") {
        performRollerCoasterSimulation();
      }
      break;
  }
}

/**
* Function to obtain the kind of experience that the user would like
*/
void getExperience() {
  Serial.println("What  kind of user experience would you like?");
  Serial.println("1: Move in one direction");
  Serial.println("2: Full experience of motion");
  Serial.println("3: Roller coaster simulation");
  Serial.println();

  // Get the desired movement
```

```
  while (!Serial.available())  {
  }
  choice = Serial.readStringUntil('\n');
  choice.trim();

  // If the user provided 1, change the state to WAIT_FOR_AXIS_CHOICE
  // If the user provided 2 or 3, change the state to PROCESSING
  // If the user didn't provide 1, 2 or 3, do not change the state so that the program can ask the user again
  if (choice.equals("1"))  {
    currentState = WAIT_FOR_AXIS_CHOICE;

  } else if (choice.equals("2") || choice.equals("3"))  {
    currentState = PROCESSING;

  } else {
    Serial.println("Invalid  choice. Please enter 1 for single direction or 2 for full experience.");
    Serial.println();
  }
}

/**
* Function to get the axis that user would like to move in
*/
void getAxis() {
  Serial.println("Enter  axis (Pitch/Roll/Yaw):");
  Serial.println();

  // Get the desired axis of movement
```

```
  while (!Serial.available())  {
  }
  String axisChoice = Serial.readStringUntil('\n');
  axisChoice.trim();

  // If the user provided a valid axis (pitch, roll, or yaw), set the desired axis
  if (axisChoice.equalsIgnoreCase("Pitch")  || axisChoice.equalsIgnoreCase("Roll")  ||
axisChoice.equalsIgnoreCase("Yaw"))  {
    currentAxis = axisChoice;

  // Otherwise the axis is invalid and the user is asked again
  } else {
    Serial.println("Invalid  axis. Please enter Pitch, Roll, or Yaw.");
    Serial.println();
    return;
  }

  // Move to getting the speed and acceleration
  currentAxis = axisChoice;
  currentState = WAIT_FOR_SPEED_AND_ACCELERATION;
}

/**
* Function to get the speed and acceleration that the user would like to move at
*/
void getSpeedAndAcceleration() {
  Serial.println("Enter  the speed you want to move at (steps/second):");
  Serial.println();
```

```
  // Get the desired speed
  while (!Serial.available())  {
  }
  String speedInput = Serial.readStringUntil('\n');
  float speed = speedInput.toFloat();

  Serial.println("Enter  the acceleration you want to move at (steps/second^2):");
  Serial.println();

  // Get the desired accleration
  while (!Serial.available())  {
  }
  String accelerationInput = Serial.readStringUntil('\n');
  float acceleration = accelerationInput.toFloat();

  // If the provided speed and acceleration are within the parameters, set the motors speed and
acceleration
  if (speed > 0 && speed <= MAX_SPEED && acceleration > 0 && acceleration <=
MAX_ACCELERATION) {
    if (currentAxis.equalsIgnoreCase("Pitch") || currentAxis.equalsIgnoreCase("Roll"))  {
      setMotorSpeedAndAcceleration(stepper1, speed, acceleration);
      setMotorSpeedAndAcceleration(stepper2, speed, acceleration);

    } else if (currentAxis.equalsIgnoreCase("Yaw")) {
      setMotorSpeedAndAcceleration(stepper3, speed, acceleration);
    }

  // Otherwise, there is or are values that are not within the parameters
  } else {
    if (speed <= 0 && acceleration <= 0) {
```

```
    Serial.println("Speed  and acceleration values are 0 or below. Setting to " +
String(MIN_SPEED) + " steps/s and " + String(MIN_ACCELERATION) + " steps/s^2.");
    Serial.println();

    speed = MIN_SPEED;
    acceleration = MIN_ACCELERATION;

  } else if (speed > MAX_SPEED && acceleration > MAX_ACCELERATION) {
    Serial.println("Speed  and acceleration values are above the specified limit. Setting to " +
String(MAX_SPEED) + " steps/s and " + String(MAX_ACCELERATION) + " steps/s^2.");
    Serial.println();

    speed = MAX_SPEED;
    acceleration = MAX_ACCELERATION;

  } else if (speed <= 0 && acceleration > 0 && acceleration <= MAX_ACCELERATION) {
    Serial.println("Speed  value is 0 or below. Setting to " + String(MIN_SPEED) + " steps/s.");
    Serial.println();

    speed = MIN_SPEED;

  } else if (speed > MAX_SPEED && acceleration > 0 && acceleration <=
MAX_ACCELERATION) {
    Serial.println("Speed  value is above the specified limit. Setting to " + String(MAX_SPEED)
+ " steps/s.");
    Serial.println();

    speed = MAX_SPEED;

  } else if (speed > 0 && speed <= MAX_SPEED && acceleration <= 0) {
```

```
    Serial.println("Acceleration  value is 0 or below. Setting to " +
String(MIN_ACCELERATION) + " steps/s^2.");
    Serial.println();

    acceleration = MIN_ACCELERATION;

  } else if (speed > 0 && speed <= MAX_SPEED && acceleration >
MAX_ACCELERATION) {
    Serial.println("Acceleration  value is above the specified limit. Setting to " +
String(MAX_ACCELERATION) + " steps/s^2.");
    Serial.println();

    acceleration = MAX_ACCELERATION;
  }

  // With the fixed speeds and accelerations, we can set the speed and acceleration of the
motor(s)
  if (currentAxis.equalsIgnoreCase("Pitch") || currentAxis.equalsIgnoreCase("Roll")) {
    setMotorSpeedAndAcceleration(stepper1, speed, acceleration);
    setMotorSpeedAndAcceleration(stepper2, speed, acceleration);

  } else if (currentAxis.equalsIgnoreCase("Yaw")) {
    setMotorSpeedAndAcceleration(stepper3, speed, acceleration);
  }
 }

 // Move to getting the desired position
 currentState = WAIT_FOR_POSITION;
}
```

```
/**
* Function called to set the speed and acceleration for a specified stepper
* @param stepper the stepper to set the speed and acceleration for
* @param speed the speed to set the passed in stepper to
* @param acceleration the acceleration to set the passed in stepper to
*/
void setMotorSpeedAndAcceleration(FastAccelStepper* stepper, float speed, float acceleration)
{
  stepper->setSpeedInHz(speed); // steps/sec
  stepper->setAcceleration(acceleration);  // steps/sec^2
}

/**
* Function to get the position the user would like to move to
*/
void getPosition() {
  String positionInput; // Variable to take in the angle input from user
  float positionInDegrees; // Variable to hold the positionInput to float conversion

  // Get wanted position in degrees if axis is pitch
  if (currentAxis.equalsIgnoreCase("Pitch"))  {
    Serial.println("Enter  a position in degrees (-30 to 30): ");
    Serial.println();
    while (!Serial.available())  {
    }

    positionInput = Serial.readStringUntil('\n');
    positionInDegrees = positionInput.toFloat();
```

```
  if (positionInDegrees < -30) {
    positionInDegrees = -30;

  } else if (positionInDegrees > 30) {
    positionInDegrees = 30;
  }
}

// Get wanted position in degrees if axis is roll
else if (currentAxis.equalsIgnoreCase("Roll"))  {
  Serial.println("Enter  a position in degrees (-30 to 30): ");
  Serial.println();
  while (!Serial.available())  {
  }

  positionInput = Serial.readStringUntil('\n');
  positionInDegrees = positionInput.toFloat();

  if (positionInDegrees < -30) {
    positionInDegrees = -30;

  } else if (positionInDegrees > 30) {
    positionInDegrees = 30;
  }
}

// Get wanted position in degrees if axis is yaw
else if (currentAxis.equalsIgnoreCase("Yaw"))  {
  Serial.println("Enter  a position in degrees (-20 to 20): ");
```

```
    Serial.println();
    while (!Serial.available())  {
    }

    positionInput = Serial.readStringUntil('\n');
    positionInDegrees = positionInput.toFloat();

    if (positionInDegrees < -20) {
      positionInDegrees = -20;

    } else if (positionInDegrees > 20) {
      positionInDegrees = 20;
    }
  }

  // Utilize the calculatePosition function to convert the degrees to # of steps
  positionInSteps = calculatePosition(positionInDegrees);

  // Move to executing move of motor(s)
  currentState = PROCESSING;
}

/**
* Function to calculate the number of steps needed to move to a certain number of degrees
* 360.0 is the degrees per revolution
* 5 is for the 1:5 ratio due to the gearbox
* 200 is the numbers of steps in a full rotation due to normal stepping
* @param positionInDegrees degrees to be converted into steps
*/
```

```
int32_t calculatePosition(float positionInDegrees) {
  return floor((positionInDegrees/(360.0/(5  * 200))));
}

/**
* Function to actually move the specified motor(s) to a specified position
* @param position the number of steps to move to get to a certain position
*/
void moveMotor(int32_t position) {
  // If the desired movement is pitch, move stepper1 and stepper2 in the same direction
  if (currentAxis.equalsIgnoreCase("Pitch"))  {
    stepper1 -> moveTo(position, false);
    stepper2 -> moveTo(position, true);

    stepper1 -> moveTo(position * -1, false);
    stepper2 -> moveTo(position * -1, true);

    stepper1 -> moveTo(0, false);
    stepper2 -> moveTo(0, true);

  // If the desired movement is roll, move stepper1 and stepper2 in opposite directions
  } else if (currentAxis.equalsIgnoreCase("Roll"))  {
    stepper1 -> moveTo(position, false);
    stepper2 -> moveTo(position * -1, true);

    stepper1 -> moveTo(position * -1, false);
    stepper2 -> moveTo(position, true);

    stepper1 -> moveTo(0, false);
```

```
    stepper2 -> moveTo(0, true);

  // If the desired movement is yaw, move stepper3
  } else if (currentAxis.equalsIgnoreCase("Yaw")) {
    stepper3 -> moveTo(position, true);

    stepper3 -> moveTo(position * -1, true);

    stepper3 -> moveTo(0, true);
  }
  // Go back to getting a new user desired movement
  currentState = WAIT_FOR_INPUT;
}

/**
* Function that moves the chair in a multi-axial experience
*/
void performFullExperienceMotion() {
  Serial.println("Moving  all three motors!");
  Serial.println();

  // 30 degress up with two motors
  // Then -60 degrees down with two motors
  // Then 30 degrees up with two motors to go back to original location
  currentAxis = "pitch";
  moveMotor(MAX_POS_PITCH_POSITION);

  // 30 degrees up with one motor and -30 degrees down with the other
  // Then 60 degrees up with one motor and -60 degrees down with the other
```

```
  // Then 30 degrees up with one motor and -30 degrees down with the other to go back to original location.
  currentAxis = "roll";
  moveMotor(MAX_POS_ROLL_POSITION);

  // 20 degrees yawing to the right
  // Then -40 degrees yawing to the left
  // Then 20 degrees yawing back to the original position
  currentAxis = "yaw";
  moveMotor(MAX_POS_YAW_POSITION);

  // Go back to getting a new user desired movement
  currentState = WAIT_FOR_INPUT;
}


/**
* Function that mimics what it would be like to be on a roller coaster
*/
void performRollerCoasterSimulation() {
  Serial.println("Starting Roller Coaster Simulation...");

  // Begin with a slow climb to the first peak
  Serial.println("Climbing to the first peak...");
  slowClimb();

  // Level out at the first peak
  Serial.println("Leveling out at the peak...");
  levelOut();
```

```
// The first major drop
Serial.println("Descending the first major drop...");
fastFall();

// Level out at the bottom
Serial.println("Leveling out after the drop...");
levelOut();

// A series of smaller hills and dips
Serial.println("Navigating smaller hills...");
for (int i = 0; i < 4; i++) {
  smallHill();
  gentleDip();
}

// Level out after last dip
Serial.println("Leveling out after hills and dips...");
levelOut();

// Introduce a sharp turn
Serial.println("Executing a sharp turn...");
sharpRightTurn();
sharpLeftTurn();
stepper3 -> moveTo(0, true);

// Simulate a fast climb to another peak
Serial.println("Fast climbing to another peak...");
fastClimb();
```

```
  // Level out at the second peak
  Serial.println("Leveling  out at the second peak...");
  levelOut();

  // Final major fall
  Serial.println("Descending the final major fall...");
  fastFall();

  // Level out at the bottom to end the ride
  Serial.println("Leveling  out to end the ride...");
  levelOut();

  // End of the simulation
  Serial.println("Roller  Coaster Ride Complete!");

  // Go back to getting a new user desired movement
  currentState = WAIT_FOR_INPUT;
}

/**
* Function to perform a slow climb up a hill
*/
void slowClimb() {
  setMotorSpeedAndAcceleration(stepper1, 600, 300);
  setMotorSpeedAndAcceleration(stepper2, 600, 300);
  int climbingHeight;

  // Inch up in 10 increments to simulate a climb
```

```
  for (int i = 1; i <= 10; i++) {
    climbingHeight = MAX_NEG_PITCH_POSITION * (0.1 * i);
    stepper1 -> moveTo(climbingHeight, false);
    stepper2 -> moveTo(climbingHeight, true);
    delay(100);
  }
  // Stay at the max climb angle for 1 second
  delay(1000);
}

/**
* Function to perform a fast climb
*/
void fastClimb() {
  setMotorSpeedAndAcceleration(stepper1, 2000, 1000);
  setMotorSpeedAndAcceleration(stepper2, 2000, 1000);

  int climbingHeight = MAX_NEG_PITCH_POSITION;

  stepper1 -> moveTo(climbingHeight, false);
  stepper2 -> moveTo(climbingHeight, true);

  // Stay at max climb angle for 2 seconds
  delay(2000);
}

/**
* Function to level out
*/
```

```
void levelOut() {
  int currentHeight = stepper1->getCurrentPosition();
  int levelingHeight;

  // Get the average speed to have a dynamic level out
  uint32_t speed1 = stepper1 -> getSpeedInMilliHz() * 1000;
  uint32_t speed2 = stepper2 -> getSpeedInMilliHz() * 1000;
  uint32_t averageSpeed = (speed1 + speed2) / 2;

  // The faster the speed, the smaller the leveling factor
  float levelingFactor = max(0.01, 1.0 - (averageSpeed / 1000.0));

  // The faster the speed, the longer the delay between each level out
  int dynamicDelay = min(500, max(100, int(averageSpeed / 10)));

  // Level out in 10 increments based on the levelingFactor and the dynamicDelay
  for (int i = 9; i >= 0; i--) {
    levelingHeight = int(currentHeight * levelingFactor * i / 9);
    stepper1->moveTo(levelingHeight, false);
    stepper2->moveTo(levelingHeight, true);
    delay(dynamicDelay);
  }
}

/**
* Function to perform a fast fall
*/
void fastFall() {
  setMotorSpeedAndAcceleration(stepper1, 3000, 1500);
```

```
  setMotorSpeedAndAcceleration(stepper2, 3000, 1500);

  int fallingHeight = MAX_POS_PITCH_POSITION;

  stepper1 -> moveTo(fallingHeight, false);
  stepper2 -> moveTo(fallingHeight, true);

  // Stay at the max fall angle for 2 seconds
  delay(2000);
}

/**
* Function to perform a gentle dip
*/
void gentleDip() {
  // Dip is half the max angle possible
  int midHeight = MAX_POS_PITCH_POSITION / 2;

  stepper1 -> moveTo(midHeight, false);
  stepper2 -> moveTo(midHeight, true);

  // Stay on the dip for only half a second
  delay(500);
}

/**
* Function to perform a small hill
*/
void smallHill() {
```

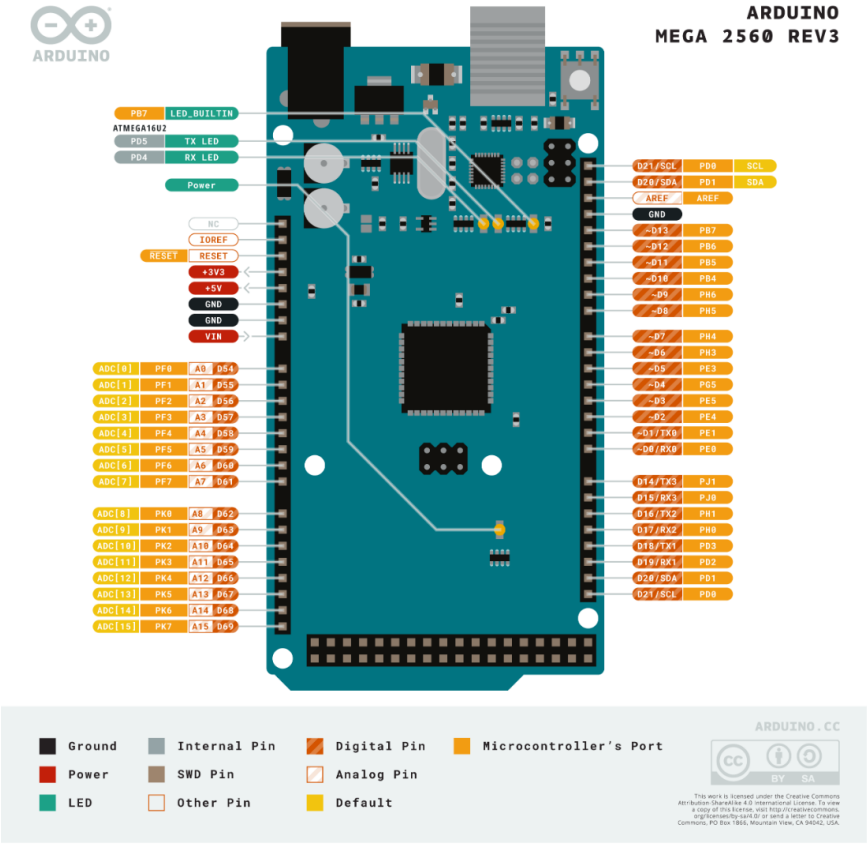
```
  // Hill is half the max angle possible
  int hillHeight = MAX_NEG_PITCH_POSITION / 2;

  stepper1 -> moveTo(hillHeight, false);
  stepper2 -> moveTo(hillHeight, true);

  // Stay on the hill for only half a second
  delay(500);
}

void sharpRightTurn() {
  setMotorSpeedAndAcceleration(stepper3, 2000, 1000);

  stepper3 -> moveTo(MAX_POS_YAW_POSITION, true);

  // Let the sharp turn last 300ms
  delay(300);
}

void sharpLeftTurn() {
  setMotorSpeedAndAcceleration(stepper3, 2000, 1000);

  stepper3 -> moveTo(MAX_NEG_YAW_POSITION, true);

  // Let the sharp turn last 300ms
  delay(300);
}

/**
```

```
* Function to perform a twist
*/
void suddenTwist() {
  stepper1 -> setSpeedInHz(2000);

  // Roll in one direction and stay there for 200ms
  stepper1 -> moveTo(MAX_POS_ROLL_POSITION, false);
  stepper2 -> moveTo(MAX_NEG_ROLL_POSITION, true);
  delay(200);

  // Roll in the other direction and stay there for 200ms
  stepper1 -> moveTo(-MAX_POS_ROLL_POSITION, false);
  stepper2 -> moveTo(-MAX_NEG_ROLL_POSITION, true);
  delay(200);
}

/**
* Function to perform a rumble of the chair
* This function seems to be rather finicky when implemented with the other movement functions
* @param numberOfRumbles how many rumbles should be performed (Ex: 40 rumbles is equal to 2 seconds of rumbles (50ms * 40))
*/
void rumble(int numberOfRumbles) {
  int pitchIntensity = 20;
  int rollIntensity = 30;
  int yawIntensity = 30;

  int32_t pitchCurrentPosition = stepper1 -> getCurrentPosition();
  int32_t rollCurrentPosition = stepper2 -> getCurrentPosition();
```

```
  int32_t yawCurrentPosition = stepper3 -> getCurrentPosition();

  stepper1 -> setSpeedInHz(1000);
  stepper2 -> setSpeedInHz(1000);
  stepper3 -> setSpeedInHz(1000);

  stepper1 -> setAcceleration(1000);
  stepper2 -> setAcceleration(1000);
  stepper3 -> setAcceleration(1000);

  // Randomly vary the intensity and direction of the steps
  for (int i = 0; i < numberOfRumbles; i++) {
    stepper1 -> move(random(-pitchIntensity, pitchIntensity), false);
    stepper2 -> move(random(-rollIntensity, rollIntensity), false);
    stepper3 -> move(random(-yawIntensity, yawIntensity), true);
    delay(50); // Short delay between the shakes
  }

  // Make sure the motors are done moving before continuing
  while (stepper1 -> isRunning() || stepper2 -> isRunning() || stepper3 -> isRunning()) {
    delay(10);
  }

  // Move the steppers back to their original position before the rumble happened
  stepper1 -> moveTo(pitchCurrentPosition, true);
  stepper2 -> moveTo(rollCurrentPosition, true);
  stepper3 -> moveTo(yawCurrentPosition, true);
}
```

Arduino MEGA 2560 Rev3 Datasheet

5 Connector Pinouts

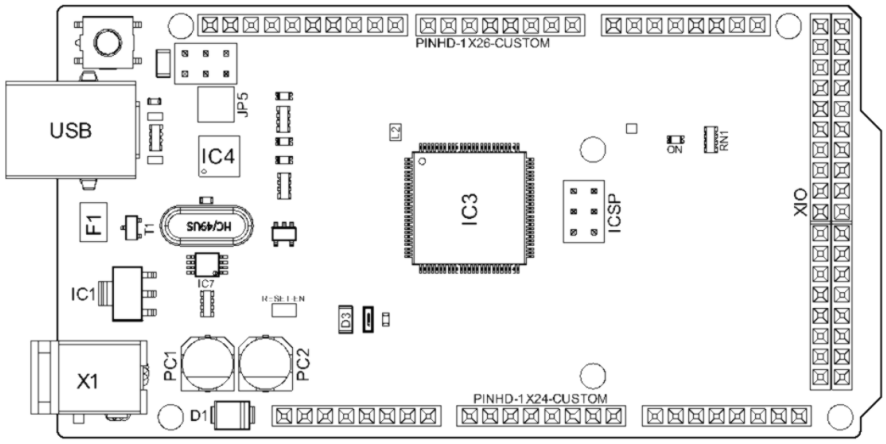


2.1 Recommended Operating Conditions

Symbol	Description	Min	Typ	Max	Unit
V_{IN}	Input voltage from VIN pad / DC Jack	7	7.0	12	V
V_{USB}	Input voltage from USB connector	4.8	5.0	5.5	V
T_{OP}	Operating Temperature	-40	25	85	°C

3.2 Board Topology

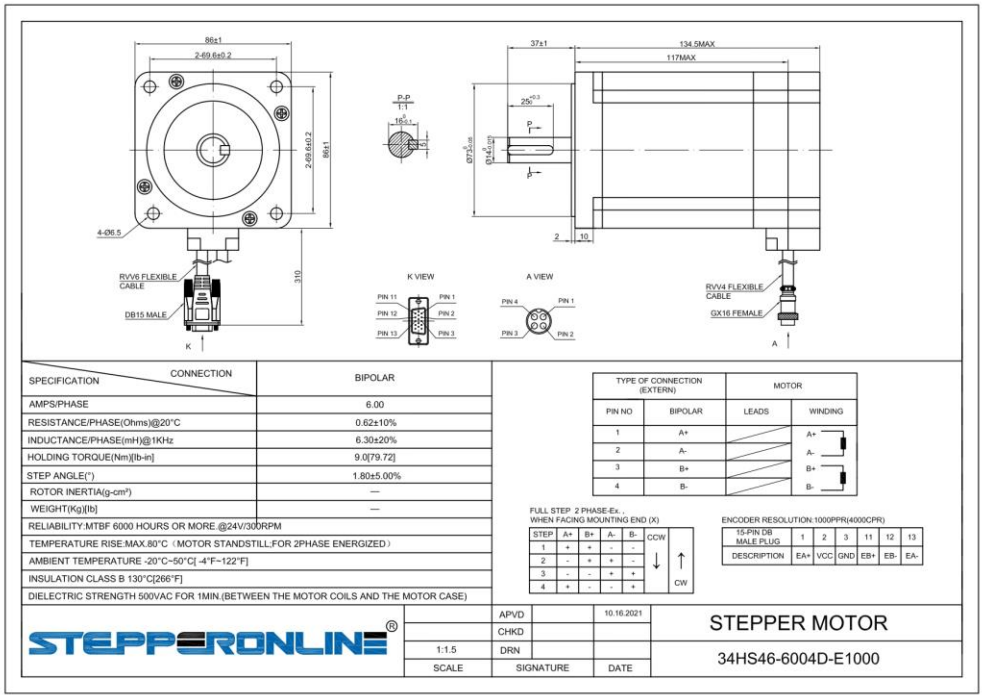
Front View



Arduino MEGA Top View

Ref.	Description	Ref.	Description
USB	USB B Connector	F1	Chip Capacitor
IC1	5V Linear Regulator	X1	Power Jack Connector
JP5	Plated Holes	IC4	ATmega16U2 chip
PC1	Electrolytic Alumninum Capacitor	PC2	Electrolytic Alumninum Capacitor
D1	General Purpose Rectifier	D3	General Purpose Diode
L2	Fixed Inductor	IC3	ATmega2560 chip
ICSP	Connector Header	ON	Green LED
RN1	Resistor Array	XIO	Connector

NEMA 34 9Nm Closed-Loop Stepper Motor Datasheet



SPECIFICATION / CONNECTION	BIPOLAR
AMPS/PHASE	6.00
RESISTANCE/PHASE(Ohms)@20°C	0.62±10%
INDUCTANCE/PHASE(mH)@1KHz	6.30±20%
HOLDING TORQUE(Nm)[lb-in]	9.0[79.72]
STEP ANGLE(°)	1.80±5.00%
ROTOR INERTIA(g-cm²)	—
WEIGHT(Kg)[lb]	—
RELIABILITY:MTBF 6000 HOURS OR MORE.@24V/300RPM	
TEMPERATURE RISE:MAX.80°C（MOTOR STANDSTILL;FOR 2PHASE ENERGIZED）	
AMBIENT TEMPERATURE -20°C~50°C[-4°F~122°F]	
INSULATION CLASS B 130°C[266°F]	
DIELECTRIC STRENGTH 500VAC FOR 1MIN.(BETWEEN THE MOTOR COILS AND THE MOTOR CASE)	

TYPE OF CONNECTION (EXTERN)		MOTOR	
PIN NO	BIPOLAR	LEADS	WINDING
1	A+		A+
2	A-		A-
3	B+		B+
4	B-		B-

FULL STEP 2 PHASE-Ex. ,
WHEN FACING MOUNTING END (X)

STEP	A+	B+	A-	B-
1	+	+	-	-
2	-	+	+	-
3	-	-	+	+
4	+	-	-	+

CCW ↓ ↑ CW

ENCODER RESOLUTION:1000PPR(4000CPR)

15-PIN DB MALE PLUG	1	2	3	11	12	13
DESCRIPTION	EA+	VCC	GND	EB+	EB-	EA-

STEPPERONLINE®		APVD	10.16.2021	STEPPER MOTOR
		CHKD		
	1:1.5	DRN		34HS46-6004D-E1000
	SCALE	SIGNATURE	DATE	

CL86T V4.1 Closed-Loop Stepper Driver Datasheet

1. Features

- Input voltage 18-80VAC or 24-110VDC
- No loss of step, No tuning
- 500 KHz max pulse input frequency
- Resolutions of 200-51,200 via DIP switches SW1 - SW4
- 2 out current settings and gain tuning via S1 rotating switch
- Optically isolated inputs with 5V or 24V
- Motor rotating direction setting by SW5
- Closed loop or open loop control setting by SW6
- Step&Direction or CW&CCW pulse type setting by SW7
- Configure position reach output or brake output by SW8
- Over-voltage, over-current protections, position following error, etc

2. Specifications

2.1 Electrical Specifications

Parameters	Min	Typical	Max	Unit
Peak Current	5.6A (RMS 4A)	7A (RMS 5A)	8A (RMS 5.7A)	A
Operating Voltage	18 24	-	80 110	VAC VDC
Logic input signal current	7	10	20	mA
High speed pulse input frequency (5V)	0	-	500	kHz
Pulse input frequency (24V)	0	-	200	kHz
Input signal voltage	5	-	24	VDC
Logic current output	-	-	100	mA

2.2 Environment

Cooling	Natural Cooling or Forced Cooling	
Operating Environment	Environment	Avoid dust, oil fog and corrosive gases
	Humidity	40%RH—90%RH
	Operating Temperature	0°C — 40°C (32°F - 102°F)
	Vibration	10-50Hz / 0.15mm
Storage Temperature	-20°C — 65°C (-4°F - 149°F)	
Weight	Approx. 600g	

3. Connections and LED Indication

3.1 Control and Digital Output Connections

PIN	I/O	Details
PUL+ (CW+)	I	Pulse and Direction Connection: **(1)** Optically isolated, high level 3.5-5V or 24V, low voltage 0-0.5V **(2)** Maximum 500 KHz input frequency **(3)** The width of PUL signal is at least 1.0μs, duty cycle is recommended 50% **(4)** Single pulse (step & direction) or double pulse (CW/CCW) is set by DIP Switch SW7 **(5)** DIR signal requires advance PUL signal minimum 2 μs in single pulse mode **(6)** The factory setting of control signal voltage is 24V, must need to set 5V/24V rotating switch if it is 5V
PUL- (CW-)	I	
DIR+ (CCW+)	I	
DIR- (CCW-)	I	
ENA+	I	Enable Signals: Optional. **(1)** Effective high level is 3.5-24V; Effective low level is 0-0.5V connection **(2)** ENA signal requires advance DIR signal minimum 200ms in single pulse mode, (default no connection)
ENA-	I	
BRK+ (PEND+)	O	Select brake output or pend output by switch 8, default as brake output Max 30VDC/100mA
BRK- (PEND-)	O	
ALM+	O	Max 30VDC/100mA
ALM-	O	

Notes:

(1) Shielding control signal wires is suggested;

(2) To avoid/reduce interference, don't tie control signal cables and power wires together;

(3) Brake output need to connect a relay and diode

5. Switch Configurations

5.1 Rotating Switch Configurations

This rotating switch is used to set the peak current of the drive and motion gain, from the motor phase current and application requirements.

Peak Current	Code	Velocity loop Ki	Position loop Kp	Velocity loop Kp
5.6A	0 (factory)	0	25	25
	1	0	50	15
	2	16	25	25
	3	16	50	15
7.0A	4	0	25	25
	5	0	50	15
	6	0	100	5
	7	16	25	25
	8	16	50	15
	9	16	100	5
8.0A	A	0	25	25
	B	0	50	15
	C	0	100	5
	D	16	25	25
	E	16	50	15
	F	16	100	5

350W Power Supply S-350-60

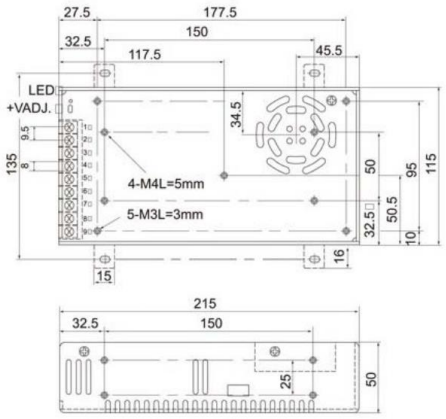
Features:

- 60V DC 5.9A output
- AC input voltage range: 93~132V/176~264VAC
- 115V/230V AC selected by switch
- High efficiency low cost
- Forced air cooling by built-in DC fan
- Low output ripple and yawp
- Over current, over voltage, short circuit and overheat protections
- 215*115*50mm（L*W*H）

General Specification:

Model	S-350-60
DC Output	60V 5.9A
Wave and Noise	300mVp-p
Inlet Stability	±0.1%
Load Stability	±0.3%
Efficiency	86%
Adjustable range for DC Voltage	54.5V~67V
AC Input Voltage	93~132V/176~264VAC Slected by Switch
AC Input Current	3.6A/115VAC 1.8A/230VAC
Working Temperature	-10~50°C
Safety Standards	GB4943, UL60950, EN60950
EMC Standards	GB9254, 55022, ClassB
Weight	1.1kg

Dimensions:

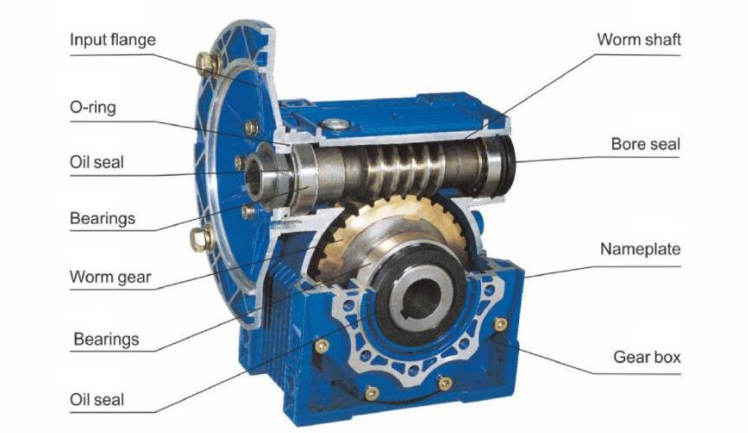


5:1 Worm Gearbox NMRV40 Datasheet

Product summary

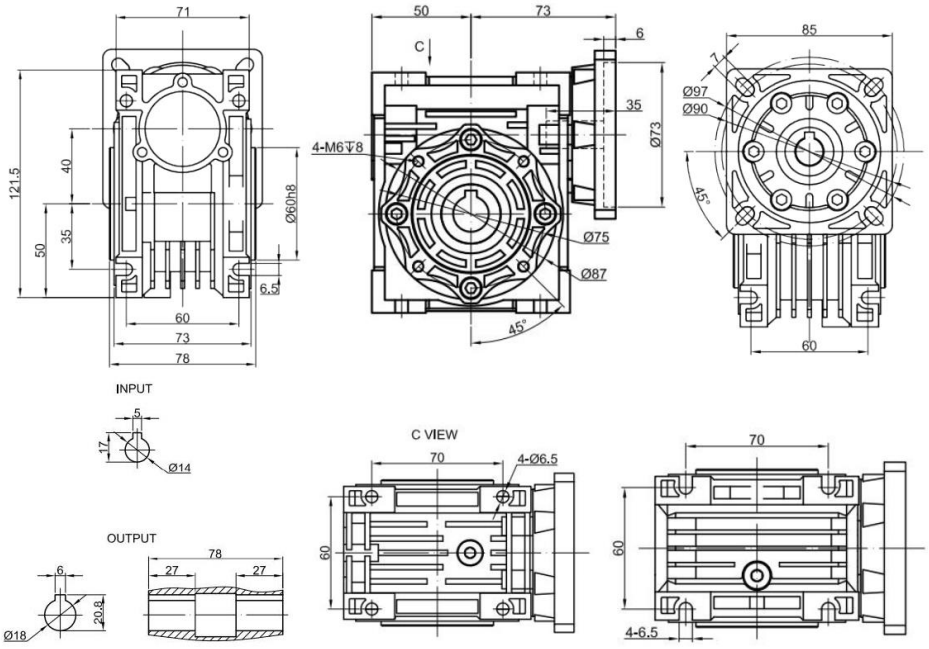
- Made of Aluminum alloy die-casting box, good looking in appearance, compact in structure rust proofing on Surface and small volume to save mounting space.
- Good radiating characteristic leads safe and high efficiency for using.
- The strong capacity of loading and overload ensure stable transmission, make less vibration and noise.
- Varies of connecting structure for power input and torque output meet different requirements; the design of box outline and the set of foot hole is apt to with many kinds of mounting.
- Besides big cases, no gap structure of box means a maintenance- free that is hermetically sealed. It prevents the lubricant from easily losing and going bad, and exchanging.

Products Structure View

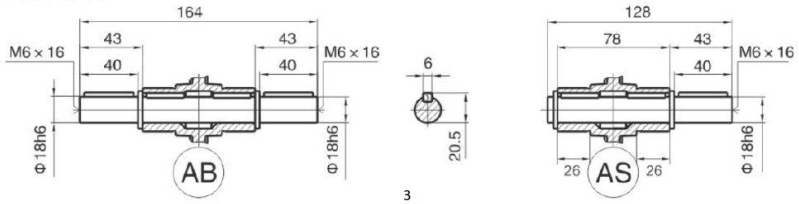


Model	Gear Ratio								
NMRV40	5	7.5	10	15	20	25	30	40	50
	Moment Permissible Torque(Nm)[lb-in]								
	40.00 [354.03]	40.00 [354.03]	40.00 [354.03]	40.00 [354.03]	40.00 [354.03]	40.00 [354.03]	45.00 [398.28]	40.00 [354.03]	38.00 [336.33]
	Efficiency								
	87%	85%	83%	82%	78%	75%	70%	65%	62%
	Shaft Maximum Radial Load(kN)								
	1.18	1.31	1.44	1.65	1.82	1.96	2.08	2.29	2.47
	Backlash@No-Load(°)								
	1								

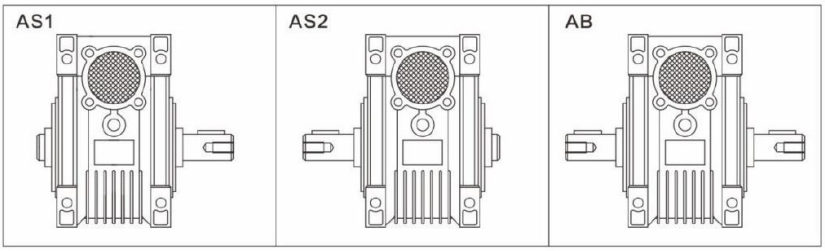
Dimension



Output Shaft



Position of output shaft



Operating Instructions

1 Single-stage worm gear reducer

1.1 It adopts high quality aluminum alloy die-casting box. It has a beautiful appearance, compact structure, small volume, light weight, saving installation space, and not easy to rust and corrosion.

1.2 Good heat dissipation performance, safe and reliable, high efficiency.

1.3 High load capacity, smooth transmission, low vibration and low noise.

1.4 A variety connecting structures with power input and torque output to meet the needs of a variety of connections; box exterior design and the setting layout of the foot hole can adapt to a variety of installation methods, with strong versatility.

2 Installation precautions

2.1 The reducer must be installed on a flat and solid base, the foot bolt must be tight and anti-vibration.

2.2 The coupling shaft extension of the prime mover-reducer-working machine must be accurately aligned with each other's axis after installation.

2.3 The outer diameter tolerance of the reducer input end and output end shaft extension is made according to h6, and the matching couplings, pulleys, sprockets and other transmission parts need to be configured according to the appropriate tolerance size. In order to avoid assembly too tight to damage the bearings, avoid assembly too loose to affect the normal power transmission.

2.4 When shaft extension is installed on sprockets, gears and other transmission parts, it should be as close to the bearing as possible to reduce the bending stress of shaft extension.

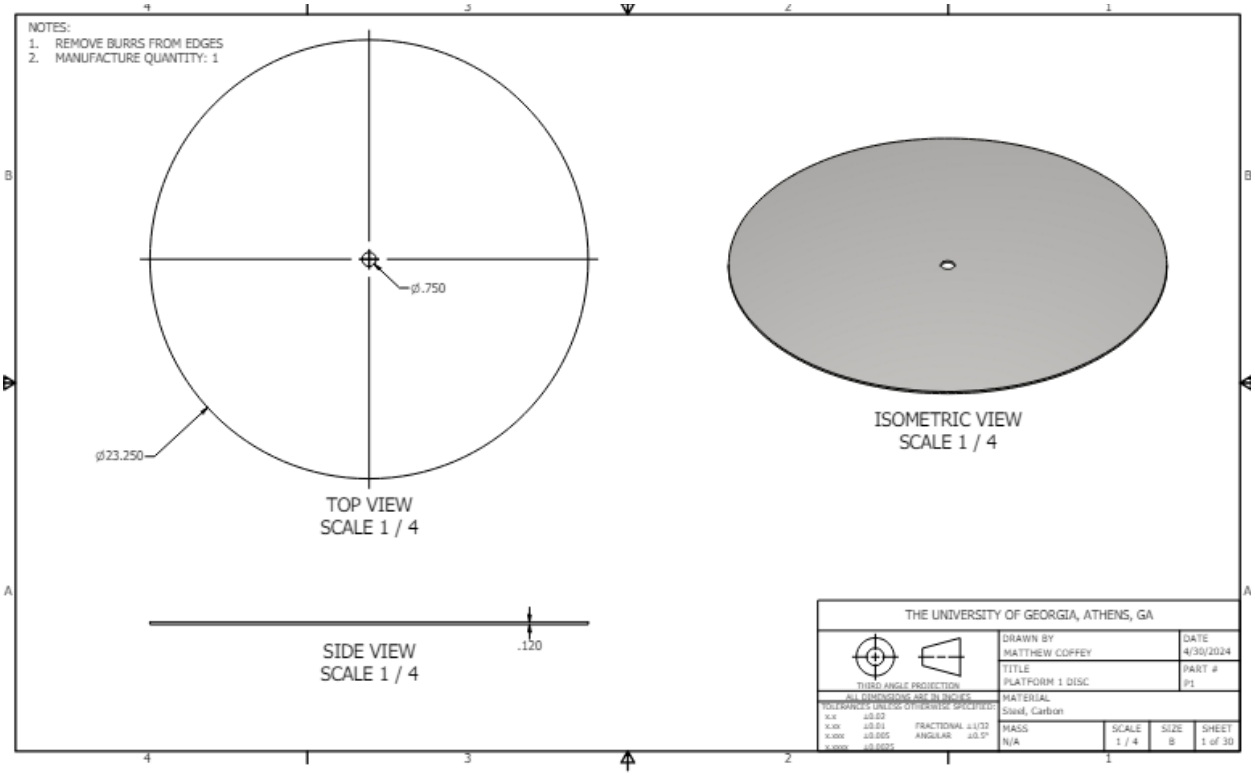
3 Precautions for use

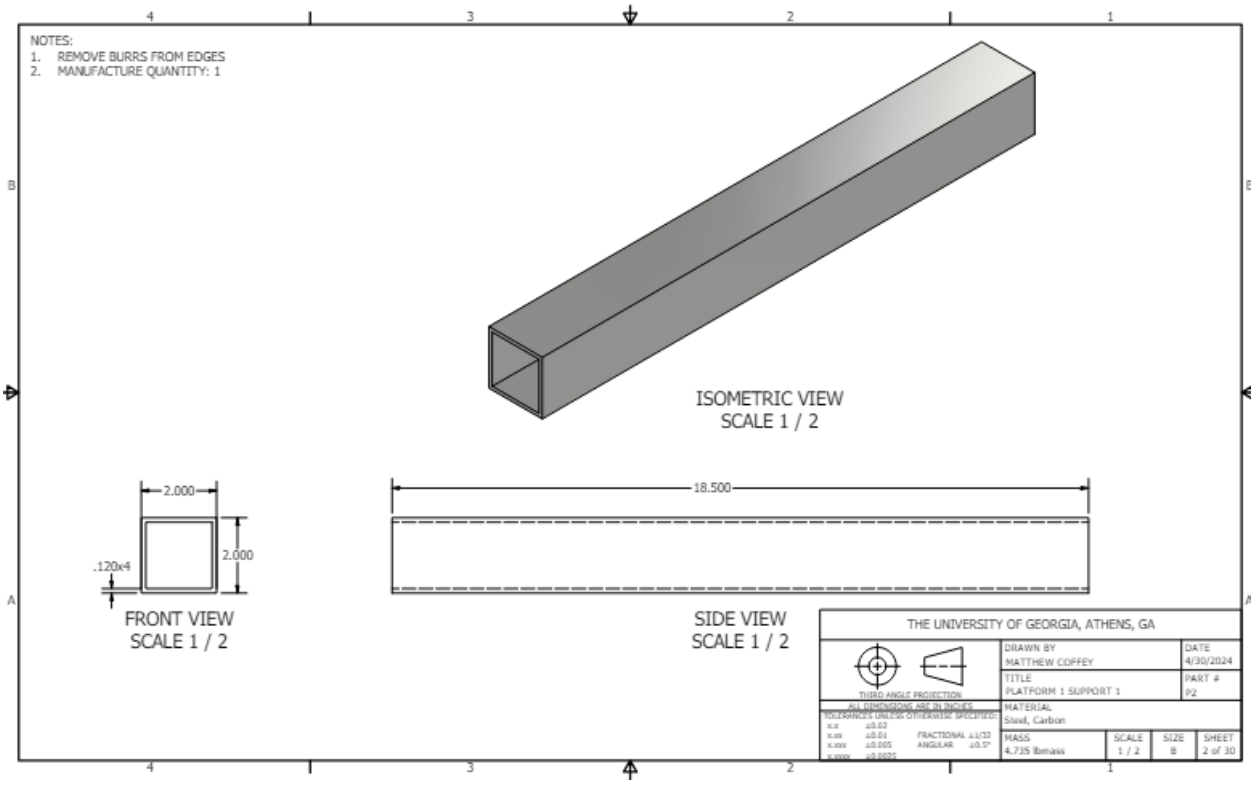
3.1 Before use, attention should be paid to check whether the reducer type structure, center distance specification, transmission ratio, input shaft connection, output shaft structure, axis pointing of input shaft output shaft and rotation direction are in line with the use requirements, and the worm input speed should not exceed 1500r/min.

3.2 When starting the machine, the load should be applied gradually and not start at full load.

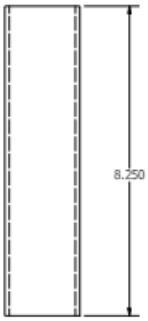
3.3 The reducer is equipped with oil filling hole and oil drain hole, ISOVg320 lubricating oil has been added in the reducer at the factory, users do not need to refuel again. After 500 hours of continuous operation, the lubricant oil should be changed. After that, the oil change cycle is 6000 hours.

Yaw Platform Manufacturing Drawings

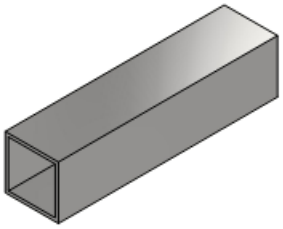




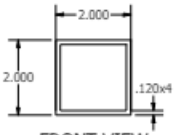
NOTES:
1. REMOVE BURRS FROM EDGES
2. MANUFACTURE QUANTITY: 2



TOP VIEW
SCALE 1 / 2



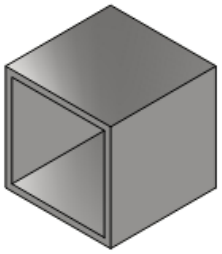
ISOMETRIC VIEW
SCALE 1 / 2



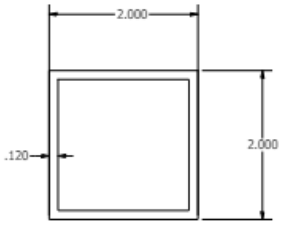
FRONT VIEW
SCALE 1 / 2

THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.03 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY MATTHEW COFFEY		DATE 4/30/2024
	TITLE PLATFORM 1 SUPPORT 2		PART # P3
	MATERIAL Steel, Carbon		
	MASS 2.111 lbmass	SCALE 1 / 2	SIZE B
			SHEET 3 of 30

NOTES:
1. REMOVE BURRS FROM EDGES
2. MANUFACTURE QUANTITY: 4



ISOMETRIC VIEW
SCALE 1 : 1



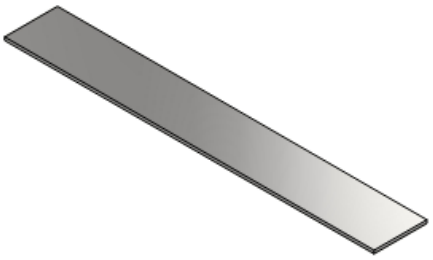
FRONT VIEW
SCALE 1 : 1

SIDE VIEW
SCALE 1 : 1

THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.03 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY MATTHEW COFFEY		DATE 4/30/2024
	TITLE PLATFORM 1 SUPPORT 3		PART # P4
	MATERIAL Steel, Carbon		
	MASS 0.512 lbmass	SCALE 1 : 1	SIZE B
			SHEET 4 of 30

NOTES:
1. REMOVE BURRS FROM EDGES
2. MANUFACTURE QUANTITY: 4

TOP VIEW
SCALE 1 / 2

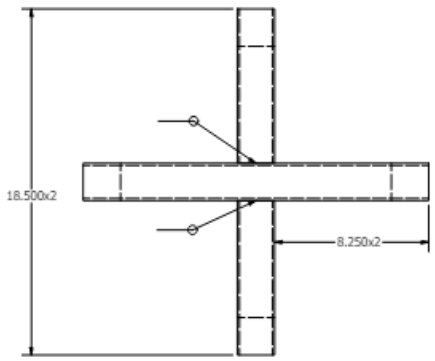


ISOMETRIC VIEW
SCALE 1 / 2

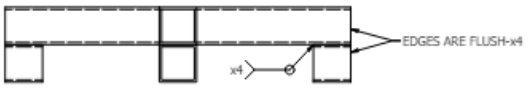
SIDE VIEW
SCALE 1 / 2

THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.03 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY MATTHEW COFFEY		DATE 4/30/2024
	TITLE PLATFORM 1 STABILIZER		PART # PS
	MATERIAL Steel, Carbon		
	MASS 0.953 lbmass	SCALE 1 / 2	SIZE B
			SHEET 5 of 30

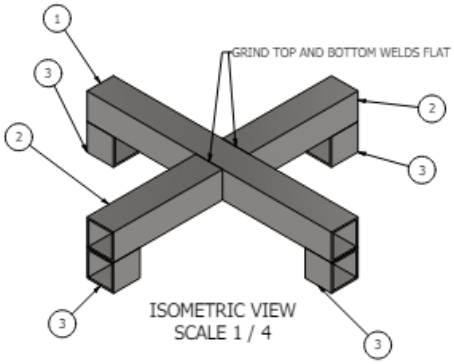
NOTES:
1. GRIND TOP AND BOTTOM WELDS FLAT
2. MANUFACTURE QUANTITY: 1



TOP VIEW
SCALE 1 / 4



SIDE VIEW
SCALE 1 / 4

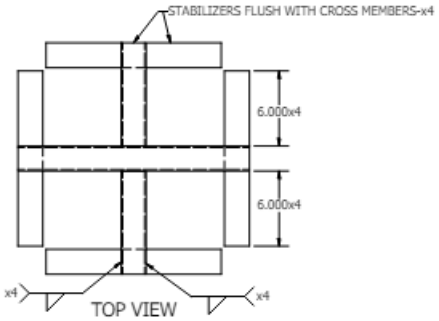


ISOMETRIC VIEW
SCALE 1 / 4

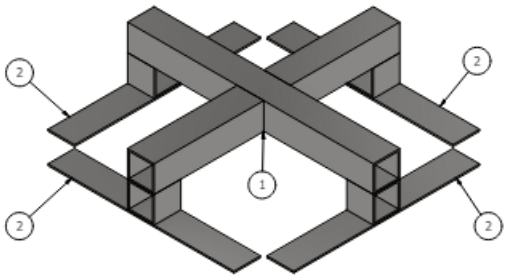
PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	P2	2x2x20-.120 THICK RECT TUBE
2	2	P3	2x2x9-.120 THICK RECT TUBE
3	4	P4	2x2x2-.120 THICK RECT TUBE

THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.03 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY MATTHEW COFFEY		DATE 4/30/2024
	TITLE PLATFORM 1 SUPPORT 1		PART # SA1
	MATERIAL		
	MASS 11.005 lbmass	SCALE 1 / 4	SIZE B
			SHEET 6 of 30

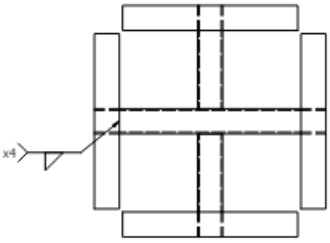
NOTES:
1. MANUFACTURE QUANTITY: 1



TOP VIEW
SCALE 1 / 6



ISOMETRIC VIEW
SCALE 1 / 4

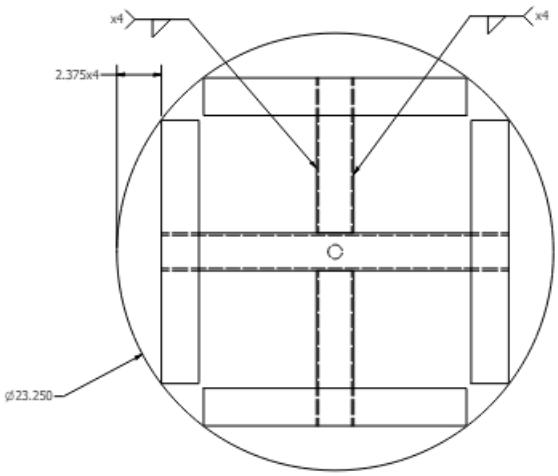


BOTTOM VIEW
SCALE 1 / 6

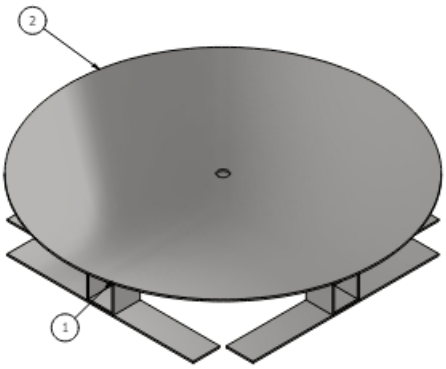
PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	SA1	PLATFORM 1 SUBASSEMBLY 1
2	4	PS	STABILIZERS

THE UNIVERSITY OF GEORGIA, ATHENS, GA		
THIRD ANGLE PROJECTION	DRAWN BY: MATTHEW COFFEY	DATE: 4/30/2024
	TITLE: PLATFORM 1 SUBASSEMBLY 2	PART #: SA2
ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.03 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	MATERIAL	
	MASS: 14.816 lbmass	SCALE: 1 / 4 | SIZE: B | SHEET: 7 of 30

NOTES:
1. MANUFACTURE QUANTITY: 1
2. CENTER THE DISC WITH THE CENTER OF THE CROSS MEMBERS IF DISC DIMENSION IS OFF



BOTTOM VIEW
SCALE 1 / 4



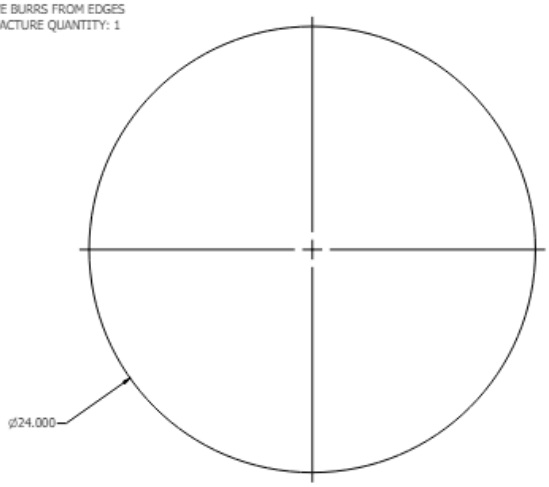
ISOMETRIC VIEW
SCALE 1 / 4

SIDE VIEW
SCALE 1 / 4

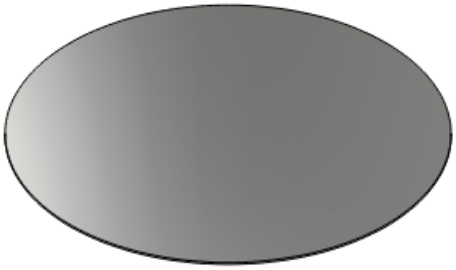
PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	SA2	PLATFORM 1 SUBASSEMBLY 2
2	1	P1	23.25" DIAMETER DISC

THE UNIVERSITY OF GEORGIA, ATHENS, GA		
THIRD ANGLE PROJECTION	DRAWN BY: MATTHEW COFFEY	DATE: 4/30/2024
	TITLE: PLATFORM 1 SUBASSEMBLY 3	PART #: SA3
ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.03 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	MATERIAL	
	MASS: N/A	SCALE: 1 / 4 | SIZE: B | SHEET: 8 of 30

NOTES:
1. REMOVE BURRS FROM EDGES
2. MANUFACTURE QUANTITY: 1



TOP VIEW
SCALE 1 / 4

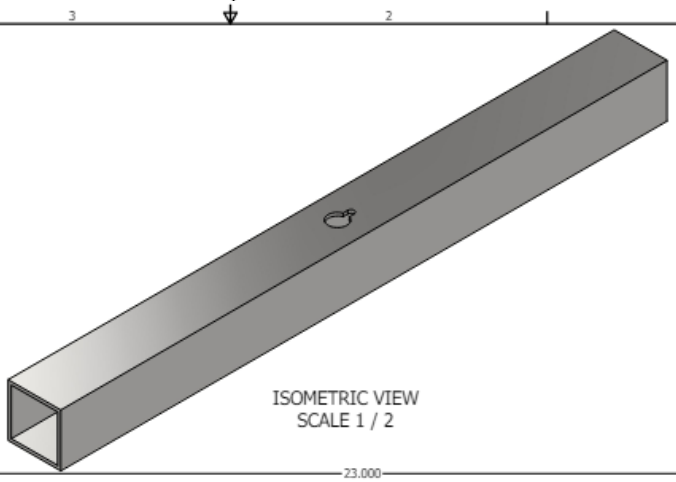


ISOMETRIC VIEW
SCALE 1 / 4

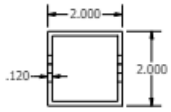
SIDE VIEW
SCALE 1 / 4

THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY MATTHEW COFFEY	DATE 4/30/2024	
	TITLE PLATFORM 2 DISC	PART # P6	
	MATERIAL Steel, Carbon		
	MASS 15.396 lbmass	SCALE 1 / 4	SIZE B
		SHEET 9 of 30	

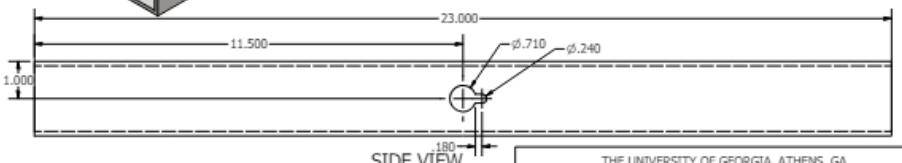
NOTES:
1. REMOVE BURRS FROM EDGES
2. MANUFACTURE QUANTITY: 1



ISOMETRIC VIEW
SCALE 1 / 2



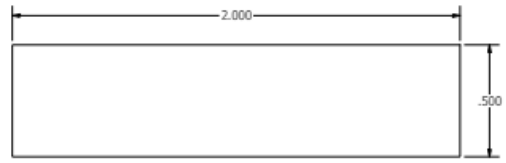
FRONT VIEW
SCALE 1 / 2



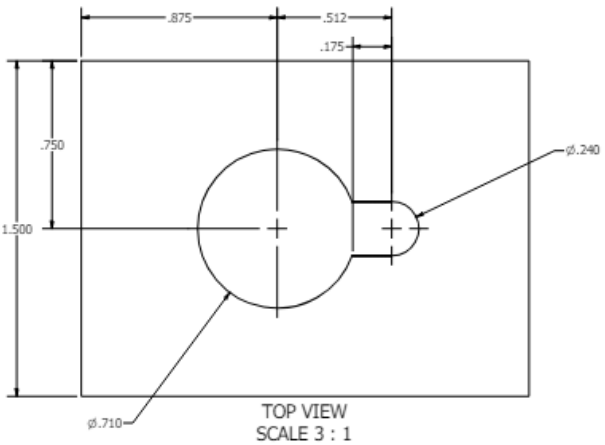
SIDE VIEW
SCALE 1 / 2

THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY MATTHEW COFFEY	DATE 4/30/2024	
	TITLE PLATFORM 2 SUPPORT 1	PART # P7	
	MATERIAL Steel, Carbon		
	MASS 5.855 lbmass	SCALE 1 / 2	SIZE B
		SHEET 10 of 30	

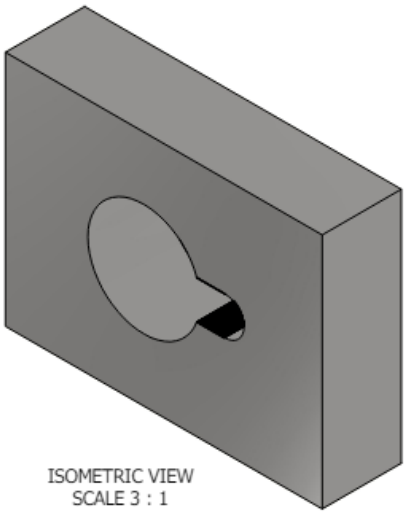
NOTES:
1. REMOVE BURRS FROM EDGES
2. MANUFACTURE QUANTITY: 1



SIDE VIEW
SCALE 3 : 1



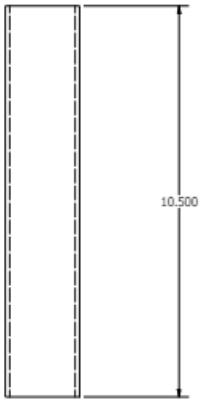
TOP VIEW
SCALE 3 : 1



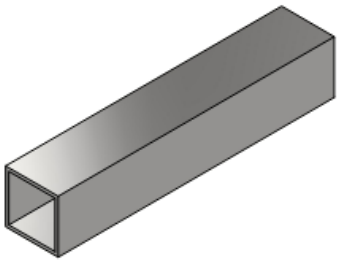
ISOMETRIC VIEW
SCALE 3 : 1

THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.03 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY MATTHEW COFFEY	DATE 4/30/2024	
	TITLE PLATFORM 2 SHAFT CONNECTER	PART # P15	
	MATERIAL Steel, Carbon		
	MASS 0.360 lbmass	SCALE 3 : 1	SIZE B
		SHEET 11 of 30	

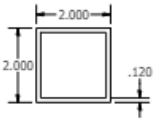
NOTES:
1. REMOVE BURRS FROM EDGES
2. MANUFACTURE QUANTITY: 2



TOP VIEW
SCALE 1 / 2



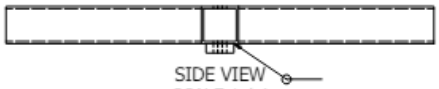
ISOMETRIC VIEW
SCALE 1 / 2



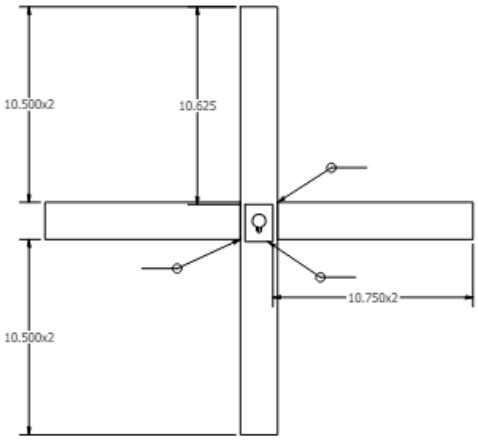
FRONT VIEW
SCALE 1 / 2

THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.03 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY MATTHEW COFFEY	DATE 4/30/2024	
	TITLE PLATFORM 2 SUPPORT 2	PART # P8	
	MATERIAL Steel, Carbon		
	MASS 2.687 lbmass	SCALE 1 / 2	SIZE B
		SHEET 12 of 30	

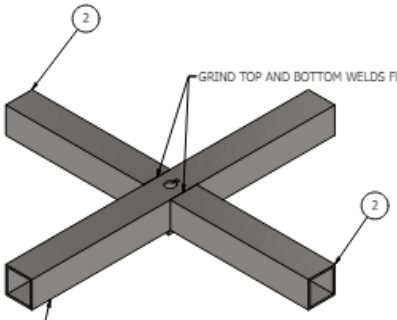
NOTES:
1. GRIND TOP AND BOTTOM WELDS FLAT
2. MANUFACTURE QUANTITY: 1



SIDE VIEW
SCALE 1 / 4



BOTTOM VIEW
SCALE 1 / 4



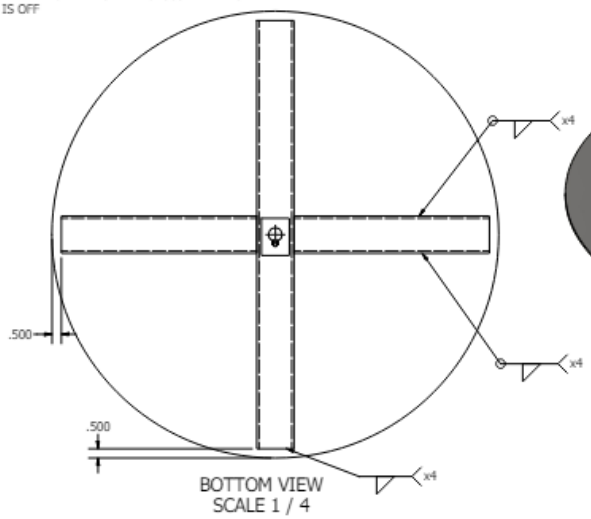
ISOMETRIC VIEW
SCALE 1 / 4

PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	P7	2x2x23-.120 THICK RECT TUBE
2	2	P8	2x2x10.5-.120 THICK RECT TUBE
3	1	Platform 2 Motor Shaft Connector	

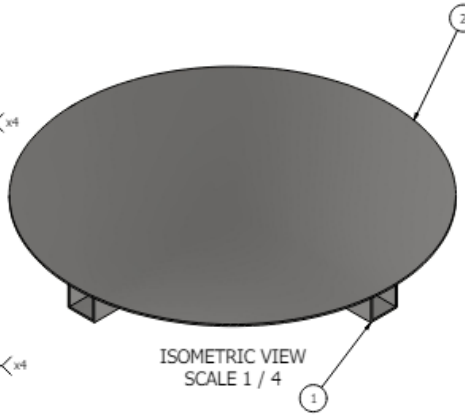
THE UNIVERSITY OF GEORGIA, ATHENS, GA

THIRD ANGLE PROJECTION	DRAWN BY: MATTHEW COFFEY	DATE: 4/30/2024
	TITLE: PLATFORM 2 SUBASSEMBLY 1	PART #: SA4
ALL DIMENSIONS ARE IN INCHES	MATERIAL:	
TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02, x.xx ±0.01, x.xxx ±0.005, x.xxxx ±0.0025; FRACTIONAL ±1/32, ANGULAR ±0.5°	MASS: 11.590 lbmass	SCALE: 1 / 4 | SIZE: B | SHEET: 13 of 30

NOTES:
1. MANUFACTURE QUANTITY: 1
2. CENTER THE DISC WITH THE CENTER OF THE CROSS MEMBERS IF DISC DIMENSION IS OFF



BOTTOM VIEW
SCALE 1 / 4



ISOMETRIC VIEW
SCALE 1 / 4

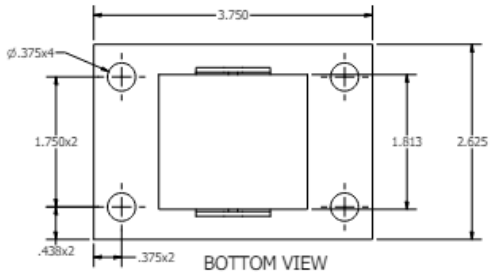
SIDE VIEW
SCALE 1 / 4

PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	SA5	PLATFORM 2 SUBASSEMBLY 1
2	1	P6	PLATFORM 2 DISC

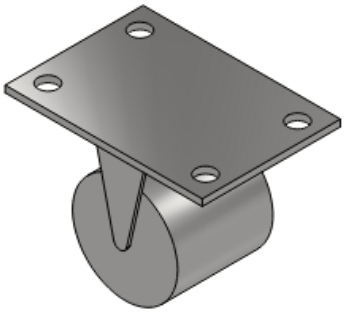
THE UNIVERSITY OF GEORGIA, ATHENS, GA

THIRD ANGLE PROJECTION	DRAWN BY: MATTHEW COFFEY	DATE: 4/30/2024
	TITLE: PLATFORM 2 SUBASSEMBLY 2	PART #: SA5
ALL DIMENSIONS ARE IN INCHES	MATERIAL:	
TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02, x.xx ±0.01, x.xxx ±0.005, x.xxxx ±0.0025; FRACTIONAL ±1/32, ANGULAR ±0.5°	MASS: N/A	SCALE: 1 / 4 | SIZE: B | SHEET: 14 of 30

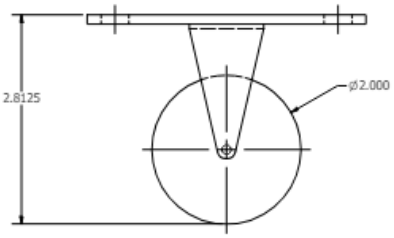
NOTES:
1. QUANTITY BOUGHT: 8
2. NO MANUFACTURING REQUIRED JUST A REFERENCE DRAWING



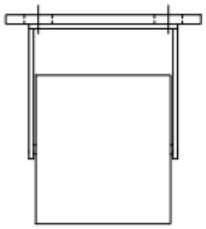
BOTTOM VIEW
SCALE 1 : 1



ISOMETRIC VIEW
SCALE 1 : 1



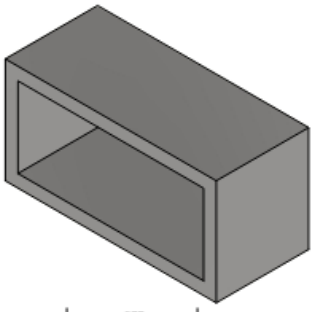
FRONT VIEW
SCALE 1 : 1

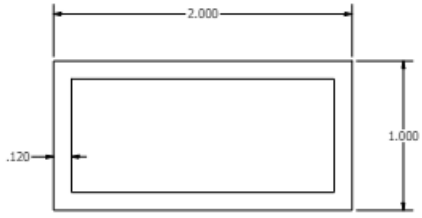


SIDE VIEW
SCALE 1 : 1

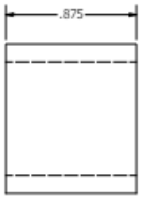
THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION	DRAWN BY MATTHEW COFFEY		DATE 4/30/2024
	TITLE CASTER WHEEL		PART # P9
ALL DIMENSIONS ARE IN INCHES	MATERIAL Steel, Carbon		
TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02, x.xx ±0.01, x.xxx ±0.005, x.xxxx ±0.0025, FRACTIONAL ±1/32, ANGULAR ±0.5°	MASS 2.021 lbmass	SCALE 1 : 1	SIZE B / SHEET 15 of 30

NOTES:
1. MANUFACTURING QUANTITY: 8
2. REMOVE BURRS FROM EDGES





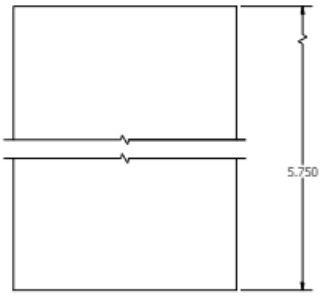
SIDE VIEW
SCALE 2 : 1



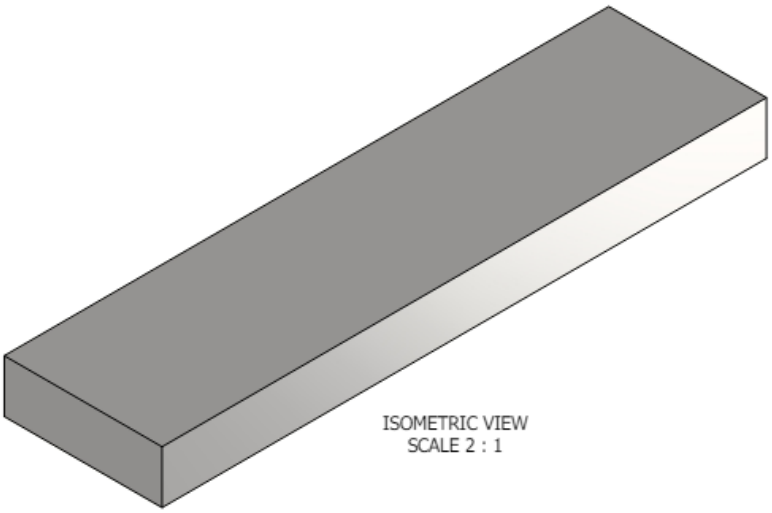
FRONT VIEW
SCALE 2 : 1

THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION	DRAWN BY MATTHEW COFFEY		DATE 4/30/2024
	TITLE CASTER SUPPORT 1		PART # P10
ALL DIMENSIONS ARE IN INCHES	MATERIAL Steel, Carbon		
TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02, x.xx ±0.01, x.xxx ±0.005, x.xxxx ±0.0025, FRACTIONAL ±1/32, ANGULAR ±0.5°	MASS 0.164 lbmass	SCALE 2 : 1	SIZE B / SHEET 16 of 30

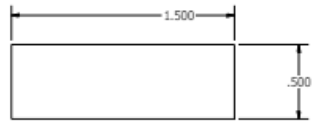
NOTES:
1. MANUFACTURING QUANTITY: 4
2. REMOVE BURRS FROM EDGES



TOP VIEW
SCALE 2 : 1



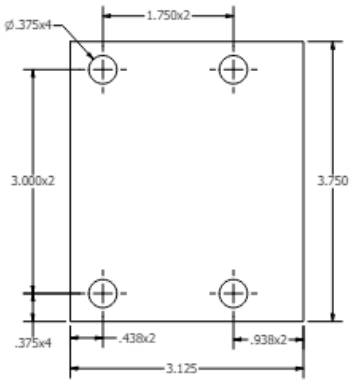
ISOMETRIC VIEW
SCALE 2 : 1

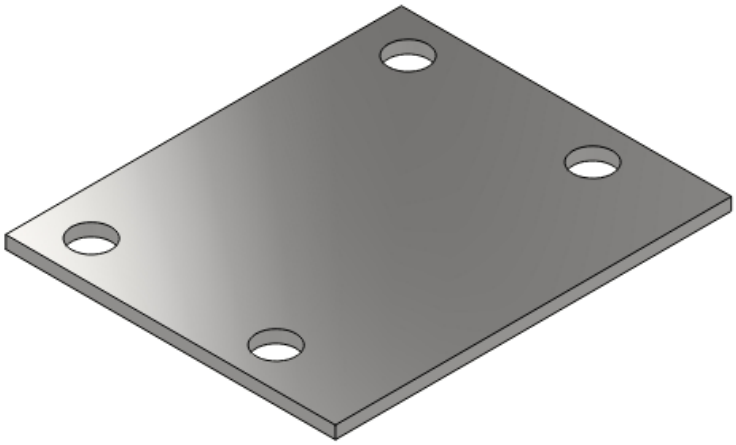


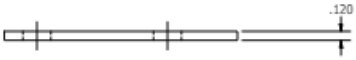
FRONT VIEW
SCALE 2 : 1

THE UNIVERSITY OF GEORGIA, ATHENS, GA				
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY MATTHEW COFFEY		DATE 4/30/2024	
	TITLE CASTER SUPPORT 2		PART # P11	
	MATERIAL Steel, Carbon			
	MASS 1.223 lbmass	SCALE 2 : 1	SIZE B	SHEET 17 of 30

NOTES:
1. MANUFACTURING QUANTITY: 8
2. REMOVE BURRS FROM EDGES

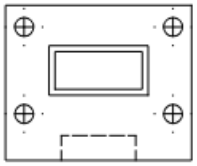




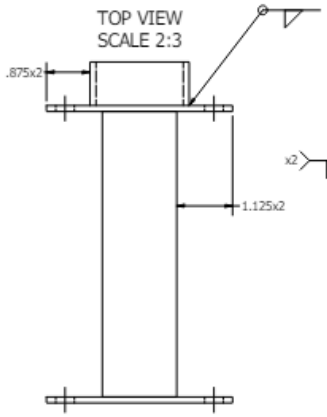


THE UNIVERSITY OF GEORGIA, ATHENS, GA				
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY MATTHEW COFFEY		DATE 4/30/2024	
	TITLE CASTER PLATE		PART # P12	
	MATERIAL Steel, Carbon			
	MASS 0.384 lbmass	SCALE 1 : 1	SIZE B	SHEET 18 of 30

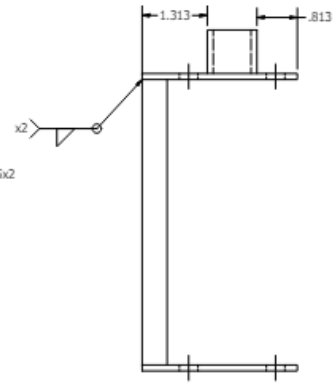
NOTES:
1. MANUFACTURING QUANTITY: 4



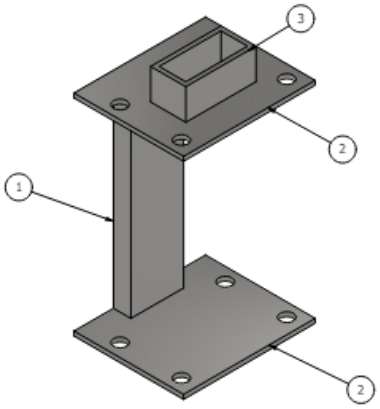
TOP VIEW
SCALE 2:3



FRONT VIEW
SCALE 2:3



SIDE VIEW
SCALE 2:3



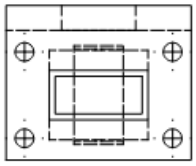
ISOMETRIC VIEW
SCALE 2:3

PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	P11	.5x1.5x5.75 BAR
2	2	P12	CASTER PLATE
3	1	P10	2x2x2-.12 THICK RECT TUBE

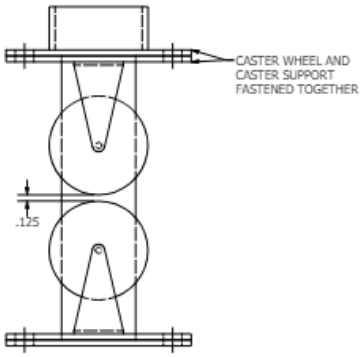
THE UNIVERSITY OF GEORGIA, ATHENS, GA

THIRD ANGLE PROJECTION	DRAWN BY: MATTHEW COFFEY	DATE: 4/30/2024
ALL DIMENSIONS ARE IN INCHES	TITLE: CASTER SUBASSEMBLY 1	PART #: SA6
TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.03, x.xx ±0.01, x.xxx ±0.005, x.xxxx ±0.0025, FRACTIONAL ±1/32, ANGULAR ±0.5°	MATERIAL	
	MASS: 2.155 lbmass	SCALE: 2:3 / SIZE: B / SHEET: 19 of 30

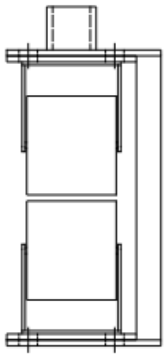
NOTES:
1. MANUFACTURING QUANTITY: 4



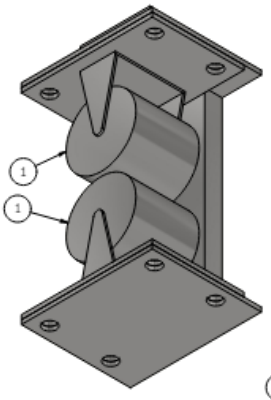
TOP VIEW
SCALE 2:3



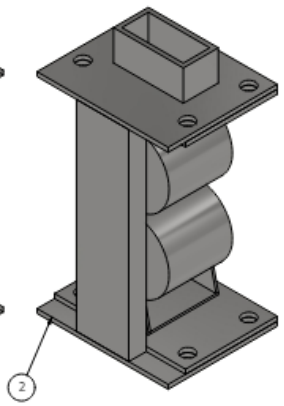
FRONT VIEW
SCALE 2:3



SIDE VIEW
SCALE 2:3



ISOMETRIC VIEW
SCALE 2:3



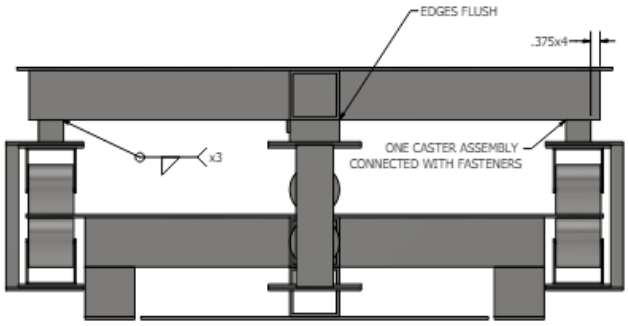
ISOMETRIC VIEW
SCALE 2:3

PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	2	P9	CASTER WHEEL
2	1	SA6	CASTER SUBASSEMBLY 1

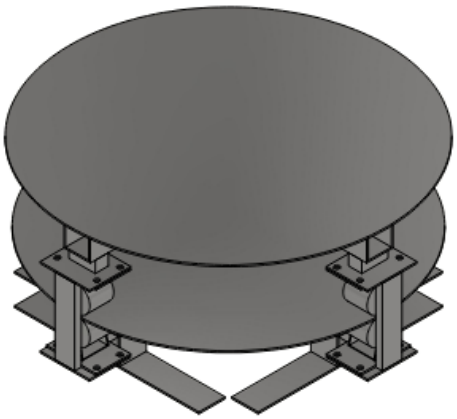
THE UNIVERSITY OF GEORGIA, ATHENS, GA

THIRD ANGLE PROJECTION	DRAWN BY: MATTHEW COFFEY	DATE: 4/30/2024
ALL DIMENSIONS ARE IN INCHES	TITLE: CASTER SUBASSEMBLY	PART #: SA7
TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.03, x.xx ±0.01, x.xxx ±0.005, x.xxxx ±0.0025, FRACTIONAL ±1/32, ANGULAR ±0.5°	MATERIAL	
	MASS: 6.197 lbmass	SCALE: 2:3 / SIZE: B / SHEET: 20 of 30

NOTES:
1. MANUFACTURING QUANTITY: 1



SIDE VIEW
SCALE 1 / 3



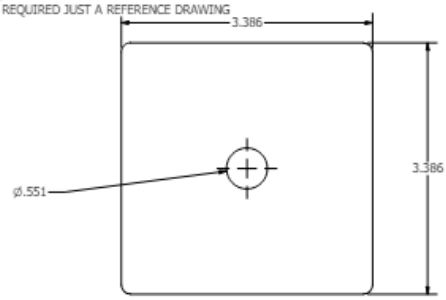
ISOMETRIC VIEW
SCALE 1 / 4

PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	SA3	PLATFORM 1 SUBASSEMBLY 3
2	1	SA5	PLATFORM 2 SUBASSEMBLY 2
3	4	SA7	CASTER SUBASSEMBLY 2

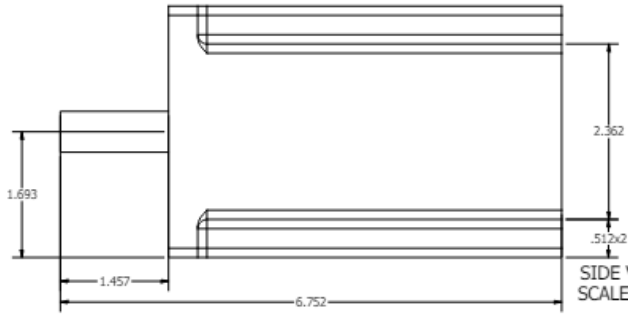
THE UNIVERSITY OF GEORGIA, ATHENS, GA

THIRD ANGLE PROJECTION	DRAWN BY: MATTHEW COFFEY	DATE: 4/30/2024
	TITLE: YAW PLATFORM ASSEMBLY	PART #: A1
ALL DIMENSIONS ARE IN INCHES	MATERIAL:	
TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02, x.xx ±0.01, x.xxx ±0.005, x.xxxx ±0.0025, FRACTIONAL ±1/32, ANGULAR ±0.5°	MASS: 81.022 lbmass	SCALE: 1 / 3 | SIZE: B | SHEET: 21 of 30

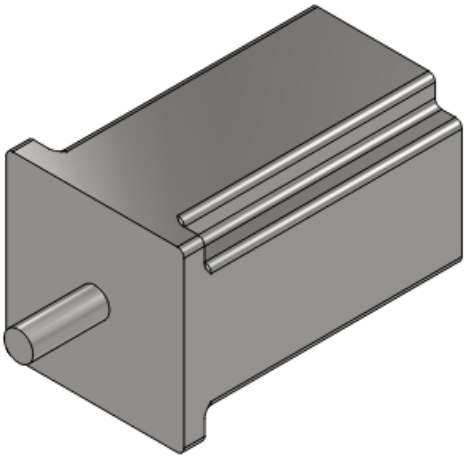
NOTES:
1. QUANTITY BOUGHT: 3
2. NO MANUFACTURING REQUIRED JUST A REFERENCE DRAWING



FRONT VIEW
SCALE 1 : 1



SIDE VIEW
SCALE 1 : 1

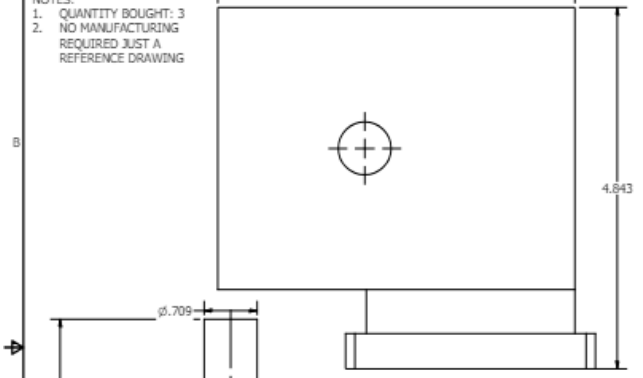


ISOMETRIC VIEW
SCALE 1 : 1

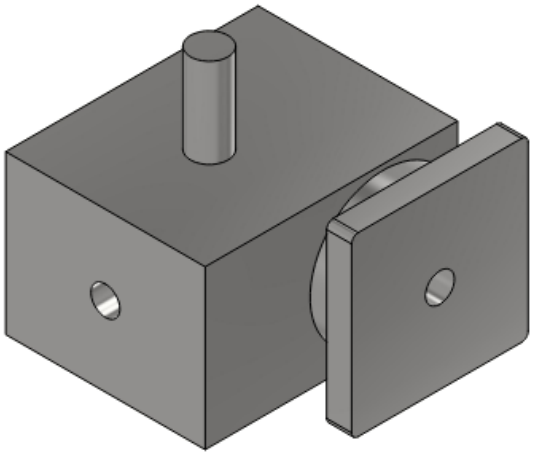
THE UNIVERSITY OF GEORGIA, ATHENS, GA

THIRD ANGLE PROJECTION	DRAWN BY: MATTHEW COFFEY	DATE: 4/30/2024
	TITLE: STEPPER MOTOR	PART #: P13
ALL DIMENSIONS ARE IN INCHES	MATERIAL: Steel, Carbon	
TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02, x.xx ±0.01, x.xxx ±0.005, x.xxxx ±0.0025, FRACTIONAL ±1/32, ANGULAR ±0.5°	MASS: 15.841 lbmass	SCALE: 1 : 1 | SIZE: B | SHEET: 22 of 30

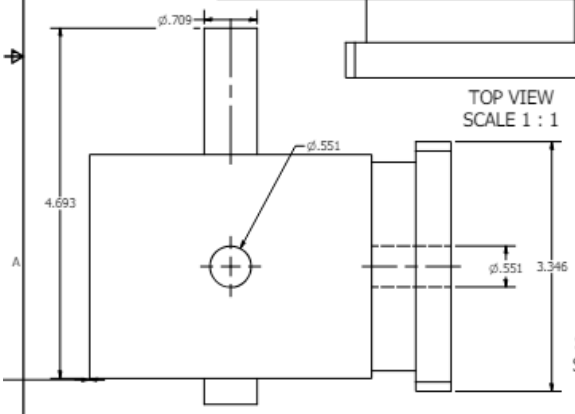
NOTES:
1. QUANTITY BOUGHT: 3
2. NO MANUFACTURING REQUIRED JUST A REFERENCE DRAWING



TOP VIEW
SCALE 1 : 1



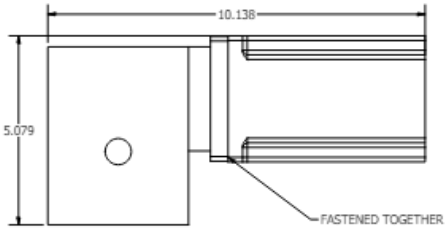
ISOMETRIC VIEW
SCALE 1 : 1



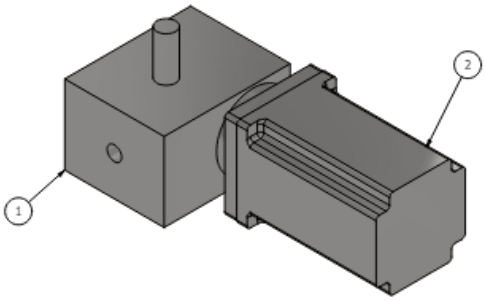
SIDE VIEW
SCALE 1 : 1

THE UNIVERSITY OF GEORGIA, ATHENS, GA				
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.03 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY MATTHEW COFFEY		DATE 4/30/2024	
	TITLE GEARBOX AND SHAFT		PART # P14	
	MATERIAL Steel, Carbon			
	MASS 17.953 lbmass	SCALE 1 : 1	SIZE B	SHEET 23 of 30

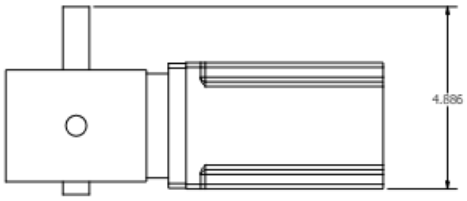
NOTES:
1. QUANTITY BOUGHT: 3
2. NO MANUFACTURING REQUIRED JUST A REFERENCE DRAWING



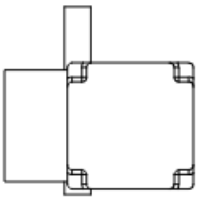
TOP VIEW
SCALE 1 / 2



ISOMETRIC VIEW
SCALE 1 / 2



SIDE VIEW
SCALE 1 / 2

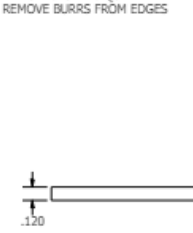


BACK VIEW
SCALE 1 / 2

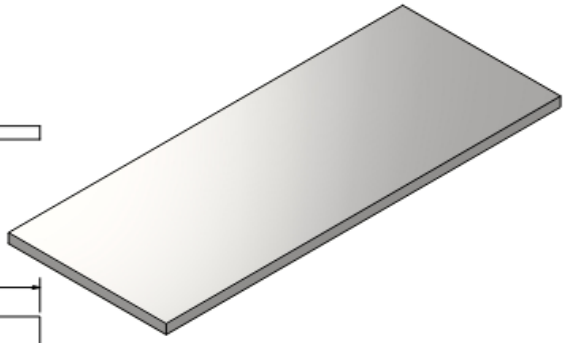
PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	P14	GEARBOX
2	1	P13	STEPPER MOTOR

THE UNIVERSITY OF GEORGIA, ATHENS, GA				
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.03 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY MATTHEW COFFEY		DATE 4/30/2024	
	TITLE MOTOR AND GEARBOX ASSEMBLY		PART # SAB	
	MATERIAL			
	MASS 33.794 lbmass	SCALE 1 / 2	SIZE B	SHEET 24 of 30

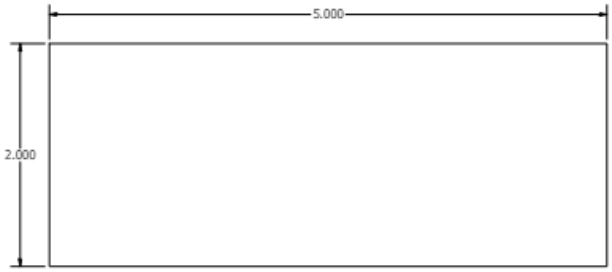
NOTES:
1. MANUFACTURING QUANTITY: 1
2. REMOVE BURRS FROM EDGES



SIDE VIEW
SCALE 3:2



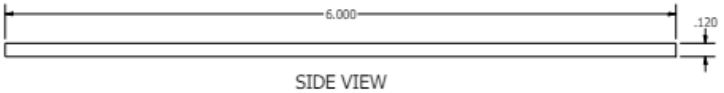
ISOMETRIC VIEW
SCALE 3:2



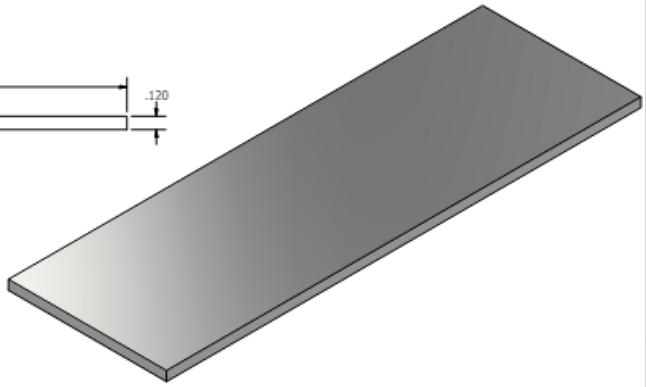
TOP VIEW
SCALE 3:2

THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION	DRAWN BY: MATTHEW COFFEY		DATE: 4/30/2024
	TITLE: MOUNT SUPPORT 1		PART #: P15
ALL DIMENSIONS ARE IN INCHES	MATERIAL: Steel, Carbon		
TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02; x.xx ±0.01; x.xxx ±0.005; x.xxxx ±0.0025; FRACTIONAL ±1/32; ANGULAR ±0.5°	MASS: 0.340 lbmass	SCALE: 3:2	SIZE: B / SHEET: 25 of 30

NOTES:
1. MANUFACTURING QUANTITY: 1
2. REMOVE BURRS FROM EDGES



SIDE VIEW
SCALE 3:2



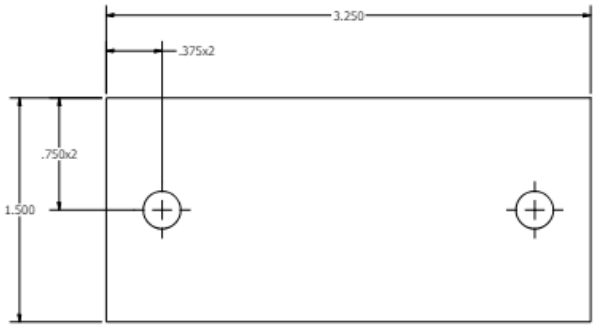
ISOMETRIC VIEW
SCALE 3:2

TOP VIEW
SCALE 3:2

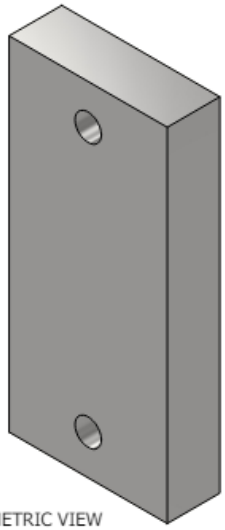
THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION	DRAWN BY: MATTHEW COFFEY		DATE: 4/30/2024
	TITLE: MOUNT SUPPORT 2		PART #: P16
ALL DIMENSIONS ARE IN INCHES	MATERIAL: Steel, Carbon		
TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02; x.xx ±0.01; x.xxx ±0.005; x.xxxx ±0.0025; FRACTIONAL ±1/32; ANGULAR ±0.5°	MASS: 0.408 lbmass	SCALE: 3:2	SIZE: B / SHEET: 26 of 30

NOTES:
1. MANUFACTURING QUANTITY: 3
2. REMOVE BURRS FROM EDGES

SIDE VIEW
SCALE 2 : 1



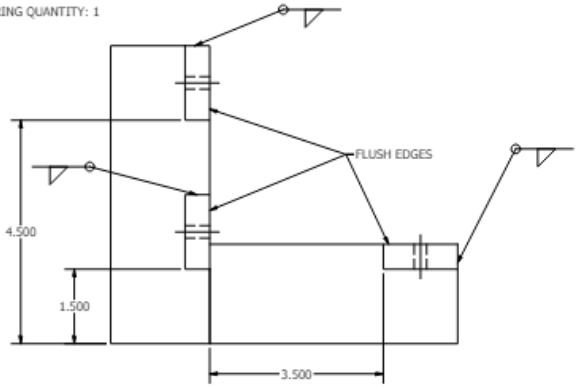
FRONT VIEW
SCALE 2 : 1



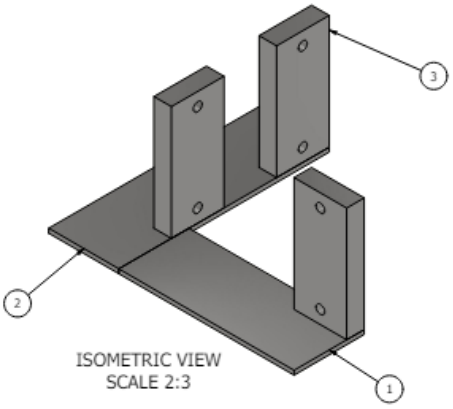
ISOMETRIC VIEW
SCALE 2 : 1

THE UNIVERSITY OF GEORGIA, ATHENS, GA		
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY MATTHEW COFFEY	DATE 4/30/2024
	TITLE MOUNT SUPPORT 3	PART # P17
	MATERIAL Steel, Carbon	
	MASS 0.677 lbmass	SCALE 2 : 1 / SIZE B / SHEET 27 of 30

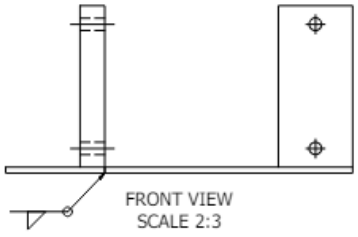
NOTES:
1. MANUFACTURING QUANTITY: 1



TOP VIEW
SCALE 2:3



ISOMETRIC VIEW
SCALE 2:3

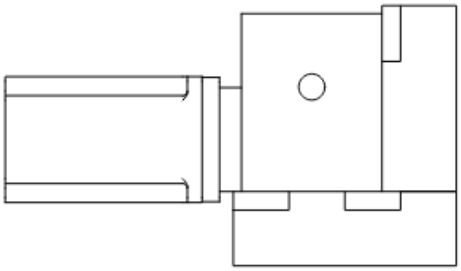


FRONT VIEW
SCALE 2:3

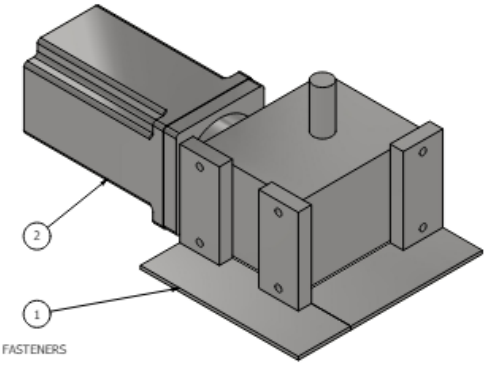
PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	P15	MOUNT SUPPORT 1
2	1	P16	MOUNT SUPPORT 2
3	3	P17	MOUNT SUPPORT 3

THE UNIVERSITY OF GEORGIA, ATHENS, GA		
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY MATTHEW COFFEY	DATE 4/30/2024
	TITLE MOUNT SUBASSEMBLY	PART # SA9
	MATERIAL	
	MASS 2.781 lbmass	SCALE 2:3 / SIZE B / SHEET 28 of 30

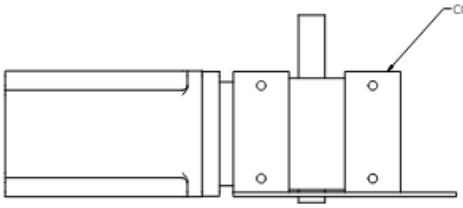
NOTES:
1. MANUFACTURING QUANTITY: 1



TOP VIEW
SCALE 1 / 2



ISOMETRIC VIEW
SCALE 1 / 2

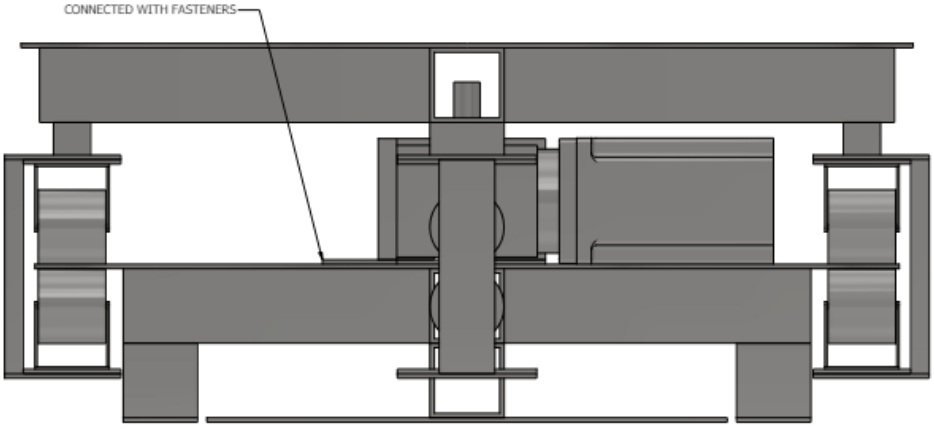


SIDE VIEW
SCALE 1 / 2

PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	SA9	MOUNT SUBASSEMBLY
2	1	SA8	MOTOR GEARBOX SUBASSEMBLY

THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY MATTHEW COFFEY		DATE 4/30/2024
	TITLE MOUNTED MOTOR		PART # SA10
	MATERIAL		
	MASS 36.575 lbmass	SCALE 1 / 2	SIZE B
			SHEET 29 of 30

NOTES:
1. MANUFACTURING QUANTITY: 1



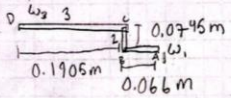
SIDE VIEW
SCALE 1 / 2

PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	A2	YAW PLATFORM ASSEMBLY 1
2	1	SA10	MOUNTED MOTOR SUBASSEMBLY

THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY MATTHEW COFFEY		DATE 4/30/2024
	TITLE YAW PLATFORM AND MOTOR		PART # A2
	MATERIAL		
	MASS 117.597 lbmass	SCALE 1 / 2	SIZE B
			SHEET 30 of 30

Pitch and Roll Platform Calculations

At 225 RPM from motor 5:1 ratio
45 RPM from gearbox



$V_B = \omega_1 \times r_1$

$= \omega_1 \hat{k} \times -0.066\hat{i}$

i	j	k
0	0	ω_1
-0.066	0	0

$V_B = -0.066\omega_1 \hat{j}$

$V_{B/C} = \omega_2 \hat{k} \times r_2 \hat{j}$

i drops out

$V_C = \omega_3 \times r_3$

$= \omega_3 \hat{k} \times 0.1905\hat{i}$

i	j	k
0	0	ω_3
0.1905	0	0

$V_C = -\omega_3 0.1905 \hat{j}$

$V_{B/C} = V_B - V_C$

$-0.066\omega_1 - \omega_3 0.1905$

$-\omega_1 = 2.89\,\omega_3$

Bar 1 rotates 2.89 times faster than bar three

At 225 RPM from motor
5:1 ratio
45 RPM from gearbox

$-45\ RPM = 2.89\,\omega_3$

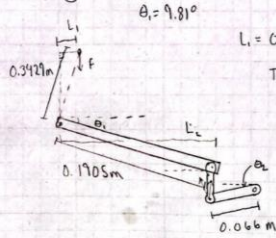
$\omega_3 = 15.57\ RPM$

$\omega_3 = 93.42\ deg/s$

we will use this as our max speed

* at 225 RPM motor outputs 4.5 N·m
from gearbox 22.5 N·m

Case ① maximum backwards tilt in Yaw
$\theta_1 = 9.81°$



$L_1 = 0.3429 \sin(9.81)$ $L_2 = 0.1905\cos(9.81)$
$L_1 = 0.0584m$ $L_2 = 0.1877m$
Torque from each gearbox: 22.5 N·m

$\frac{22.5\ N\cdot m}{0.066\ m} = 340.9\ N$

At end of motor arm

$\theta_2 = 35°$

$340.9\cos 35° = 279.25\ N$

Quliable due to angle

Summing moments to find net torque

$F = 120kg \cdot 9.81\ m/s^2 = 1177.2N$

$\tau_{net} = -1177.2N \cdot 0.0584m + 2 \cdot 279.25N \cdot 0.1877m$

$\tau_{net} = 36.08\ N\cdot m$

Moment of inertia of rotating frame + person
$I = 120kg \cdot 0.3429^2 m = 14.11\ kg\cdot m^2$

* center of gravity of top frame + person estimated to 0.3429m above pivot point

Using $\tau_{net} = I\alpha$

$\frac{36.08\ N\cdot m}{14.11\ kg\cdot m^2} = 2.56\ rad/s^2$

$\Delta\theta = 9.81° = 0.1712\ rad$

Using $\omega^2 = \omega_0^2 + 2\alpha\Delta\theta$

$\omega = \sqrt{0^2 + 2(2.56)(0.1712)}$

$\omega = 0.936\ rad/s = 53.63\ deg/s$

$\omega = \omega_0 + \alpha t$

$0.936\ rad/s = 0 + 2.56\ rad/s^2\, t$

$t = 0.366\ s$

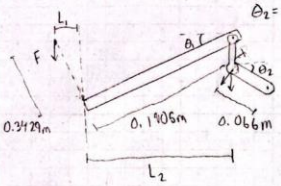
From the position shown to the flat position the person + top frame would accelerate at 2.56 rad/s² to a speed of 53.63 deg/s and it would take 0.366 seconds

* this uses the min net torque and keeps it constant

* Note this is an underestimation of the speed + acceleration because as the frame rotates to the flat position the net torque on the rotating body will increase.

Case ② maximum fowards tilt in yaw

$\theta_1 = 9.21°$ $\theta_2 = 35°$



$L_1 = 0.3429\sin(9.21°)$ $L_2 = 0.1905\cos(9.21)$

$L_1 = 0.055m$ $L_2 = 0.188m$

Torque from each gearbox 22.5 N·m

$\frac{22.5\ N\cdot m}{0.066m} = 340.9N$ At end of motor arm

$340.9\cos 35° = 279.25N$

Avliable force due to angle

Summing moments to find net torque

$\tau_{net} = -1177.2N \cdot 0.055m + 2 \cdot 279.25N \cdot 0.188m$ $F = 120kg \cdot 9.81\ m/s^2 = 1177.2\ N$

$\tau_{net} = 40.25\ N\cdot m$

Moment of inertia of rotating frame + person

$I = 120kg \cdot 0.3429^2 m = 14.11 kg\cdot m^2$

★ center of gravity of top frame + person estimated to 0.3429m above pivot point.

Using $\tau_{net} = I\alpha$

$\frac{40.25\ N}{14.11\ kg\cdot m^2} = 2.85\ rad/s^2$

$\Delta\theta = 9.21° = 0.1607\ rad$

Using $\omega^2 = \omega_0^2 + 2\alpha\Delta\theta$

$\omega = \omega_0 + \alpha t$

$\omega = \sqrt{0^2 + 2(2.85\ rad/s^2)(0.1607\ rad)}$

$\omega = 0.957\ rad/s = 54.83\ deg/s$

From the position shown to the flat position the person + top frame would accelerate at 2.85 rad/s² to a speed of 54.83 deg/s + it would take 0.34 s

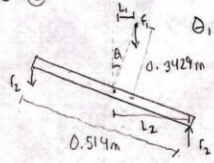
$0.957\ rad/s = 0 + 2.85 rad/s^2 \cdot t$

$t = 0.34\ s$

- this is assuming minimum torque that is constant
- in reality the motor torque would be greater because the motors would not run at our set max speed at the extreme positions

Case ③ maximum tilt in roll on either side

$\theta_1 = 7.94°$



$L_1 = 0.3429\sin(7.94°)$ $L_2 = \frac{0.514}{2}(\cos(7.94)$

$L_1 = 0.0474m$ $L_2 = 0.2545m$

torque from each gearbox 22.5 N·m

$F_2 = 279.25N$

avliable torque at ends

$F_1 = 120kg \cdot 9.81\ m/s^2 = 1177.2N$

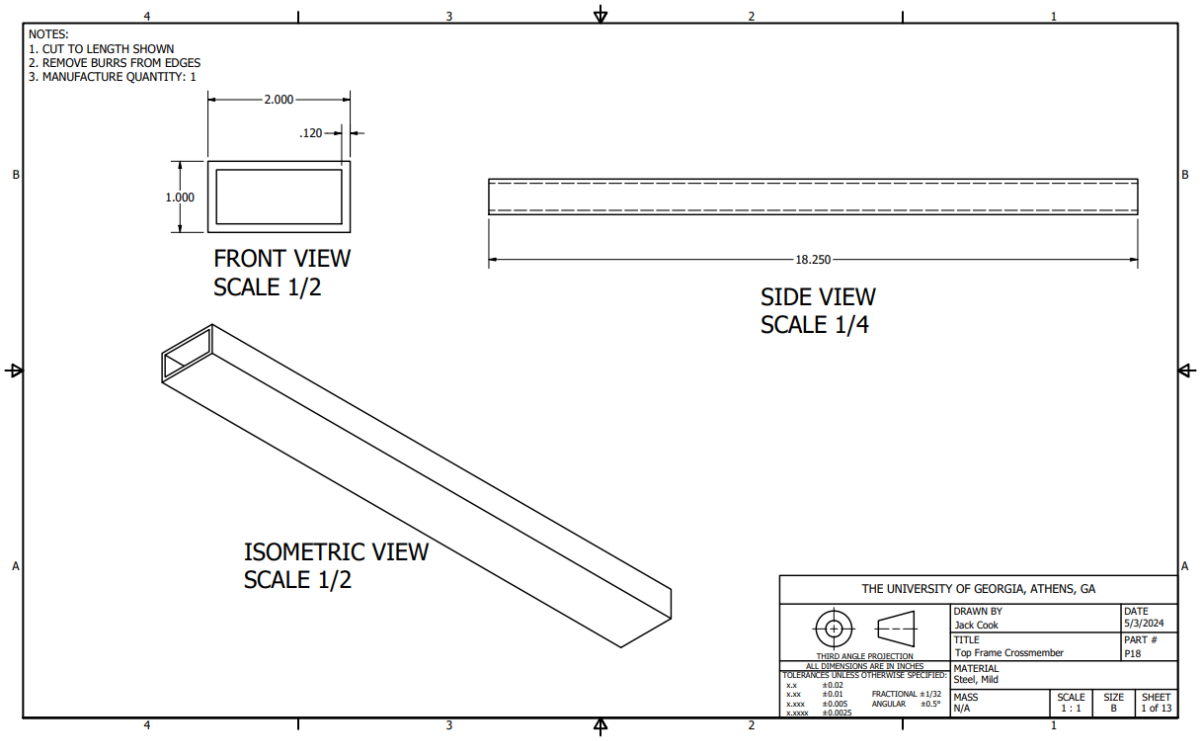
Summing moments to find net torque

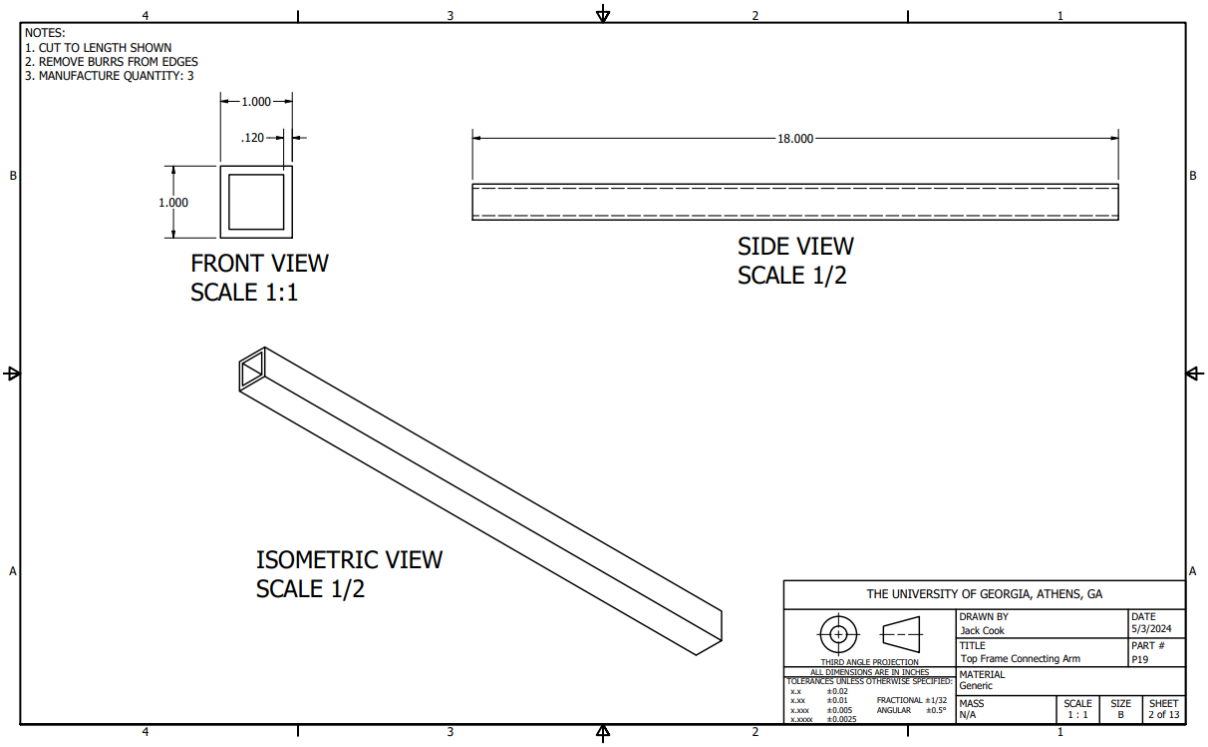
$\tau_{net} = -1177.2N \cdot 0.0474m + 2 \cdot 279.25N \cdot 0.2545m$

$\tau_{net} = 86.34\ N\cdot m$

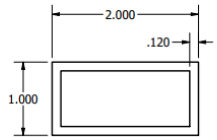
Net torque is positive because torque supplied by motors is greater than torque produced by weight of top frame. This means frame will be able to move with lowest torque value. This tells us that the design will work as intended.

Pitch and Roll Platform Manufacturing Drawings

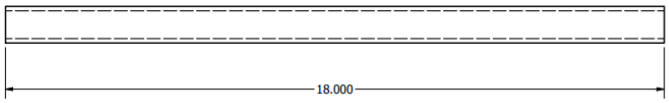




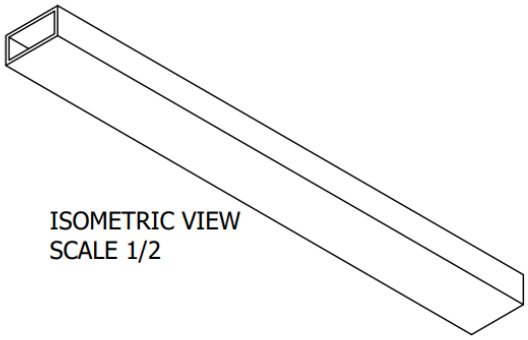
NOTES:
1. CUT TO LENGTH SHOWN
2. REMOVE BURRS FROM EDGES
3. MANUFACTURE QUANTITY: 3



FRONT VIEW
SCALE 1:1



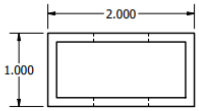
SIDE VIEW
SCALE 1/2



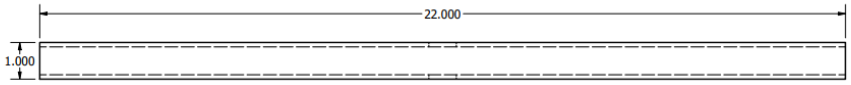
ISOMETRIC VIEW
SCALE 1/2

THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY Jack Cook	DATE 5/3/2024	
	TITLE Base Frame Support	PART # P20	
	MATERIAL Generic		
	MASS N/A	SCALE 1 : 1	SIZE B
		SHEET 3 of 13	

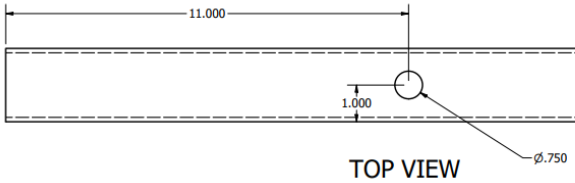
NOTES:
1. CUT TO LENGTH SHOWN
2. REMOVE BURRS FROM EDGES
3. MANUFACTURE QUANTITY: 1
4. HOLE IS CENTERED



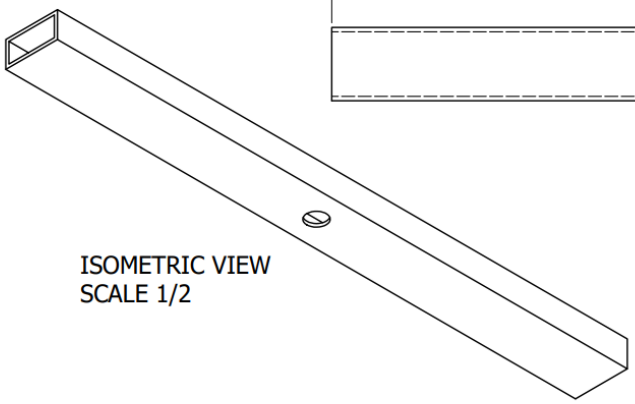
FRONT VIEW
SCALE 1:1



SIDE VIEW
SCALE 1/2



TOP VIEW
SCALE 1/2

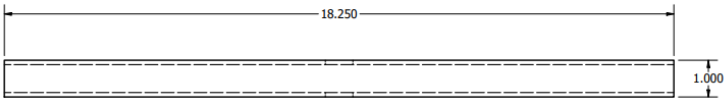


ISOMETRIC VIEW
SCALE 1/2

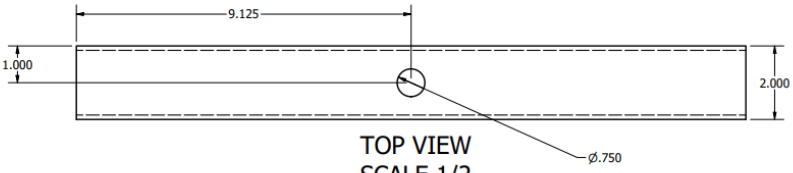
THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY Jack Cook	DATE 5/3/2024	
	TITLE Base Frame Crossmember	PART # P21	
	MATERIAL Generic		
	MASS N/A	SCALE 1 : 1	SIZE B
		SHEET 4 of 13	

NOTES:
1. HOLE IS CENTERED
2. MANUFACTURE QUANTITY: 1

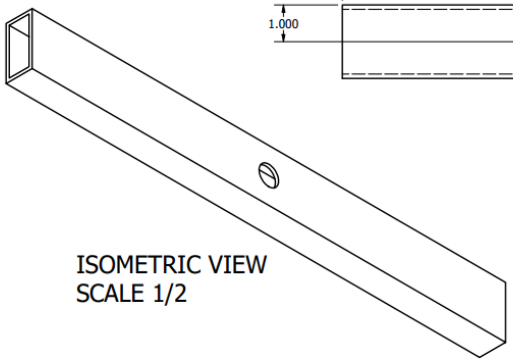
FRONT VIEW
SCALE 1:1



SIDE VIEW
SCALE 1/2



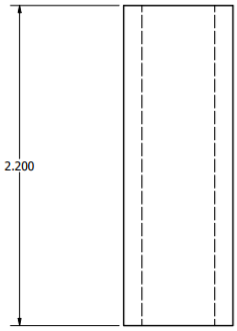
TOP VIEW
SCALE 1/2



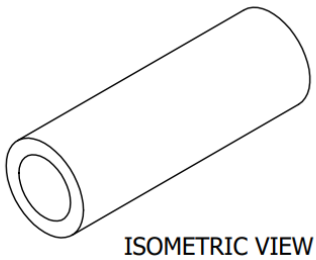
ISOMETRIC VIEW
SCALE 1/2

THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION	DRAWN BY: Jack Cook		DATE: 5/3/2024
	TITLE: Crossmember with mounting hole		PART #: P22
ALL DIMENSIONS ARE IN INCHES	MATERIAL: Steel, Mild		
TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02, x.xx ±0.01, x.xxx ±0.005, x.xxxx ±0.0025; FRACTIONAL ±1/32; ANGULAR ±0.5°	MASS: N/A	SCALE: 1 : 1	SIZE: B / SHEET: 5 of 13

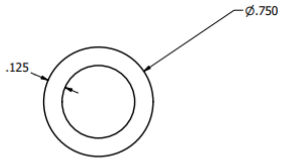
NOTES
1. CUT TO LENGTH
2. MANUFACTURE QUANTITY: 2



SIDE VIEW
SCALE 2:1



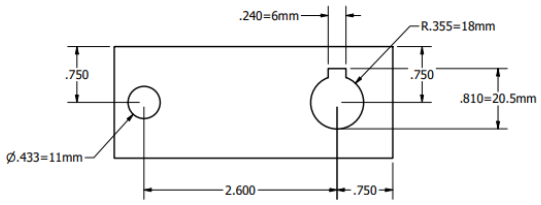
ISOMETRIC VIEW
SCALE 2:1



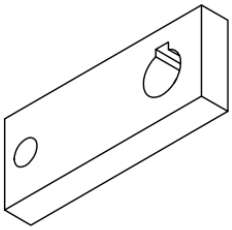
FRONT VIEW
SCALE 2:1

THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION	DRAWN BY: Jack Cook		DATE: 5/3/2024
	TITLE: Mounting Bracket for U-Joint		PART #: P23
ALL DIMENSIONS ARE IN INCHES	MATERIAL: Generic		
TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02, x.xx ±0.01, x.xxx ±0.005, x.xxxx ±0.0025; FRACTIONAL ±1/32; ANGULAR ±0.5°	MASS: N/A	SCALE: 2 : 1	SIZE: B / SHEET: 6 of 13

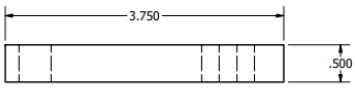
NOTES:
1. MANUFACTURE QUANTITY: 2



TOP VIEW
SCALE 1:1



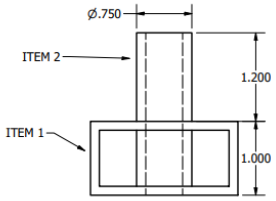
ISOMETRIC VIEW
SCALE 1:1



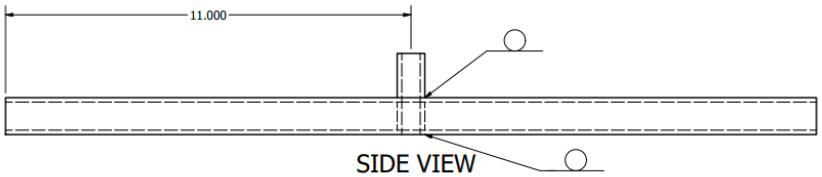
SIDE VIEW
SCALE 1:1

THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY Jack Cook	DATE 5/3/2024	
	TITLE Shaft Arm	PART # P24	
	MATERIAL Generic		
	MASS N/A	SCALE 1 : 1	SIZE B
		SHEET 12 of 13	

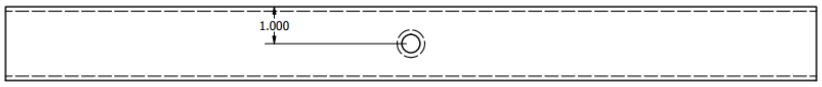
NOTES
1. MANUFACTURE QUANTITY: 1



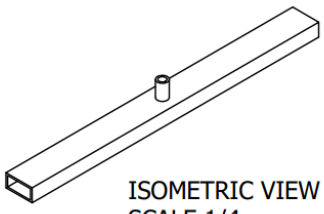
FRONT VIEW
SCALE 1:1



SIDE VIEW
SCALE 1/2



TOP VIEW
SCALE 1/2

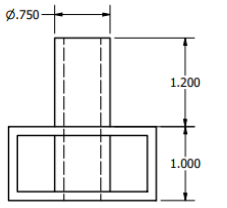


ISOMETRIC VIEW
SCALE 1/4

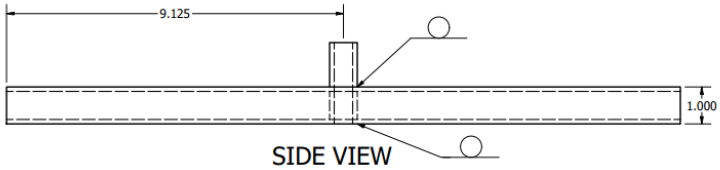
PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	P21	1in x 2in 11ga 22in length
2	1	P23	2.2in length 0.75dia tubing

THE UNIVERSITY OF GEORGIA, ATHENS, GA			
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY Jack Cook	DATE 5/3/2024	
	TITLE Crossmember with tube mount BOT	PART # SA11	
	MATERIAL Generic		
	MASS N/A	SCALE 1 : 1	SIZE B
		SHEET 7 of 13	

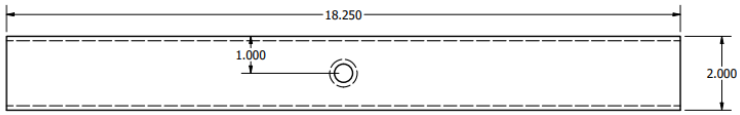
NOTES
1. MANUFACTURE QUANTITY: 1



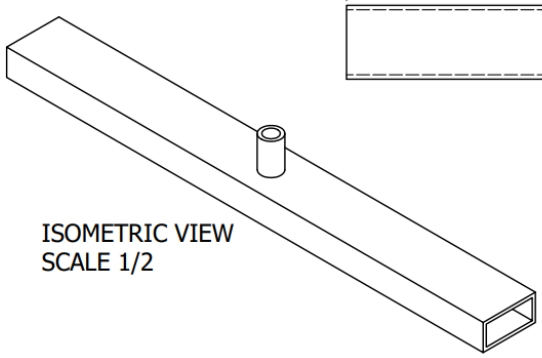
FRONT VIEW
SCALE 1:1



SIDE VIEW
SCALE 1/2



TOP VIEW
SCALE 1/2

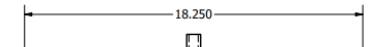


ISOMETRIC VIEW
SCALE 1/2

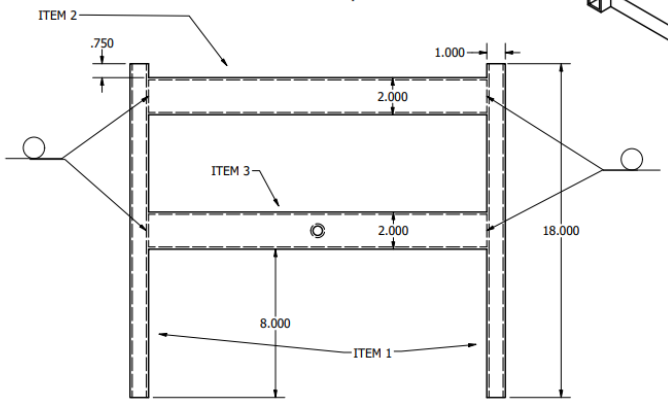
PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	P22	1in x 2in 11ga 18.25in length
2	1	P23	2.2in length 0.75dia tubing

THE UNIVERSITY OF GEORGIA, ATHENS, GA

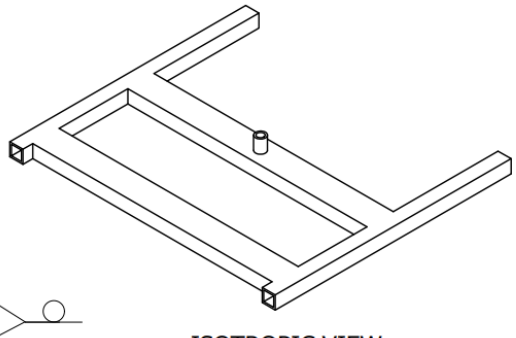
THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY Jack Cook	DATE 5/3/2024
	TITLE Crossmember with tube mount TOP	PART # SA12
	MATERIAL Steel, Mild	
	MASS N/A	SCALE 1 : 1 / SIZE B / SHEET 8 of 13



SIDE VIEW
SCALE 1/4



BOTTOM VIEW
SCALE 1/4



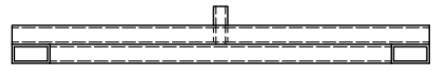
ISOTROPIC VIEW
SCALE 1/4

PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	2	P19	1in x 1in 11ga 18in length
2	1	P18	1in x 2in 11ga 18.25in length
3	1	SA12	1in x 2in 11ga 18.25in length w/ mount

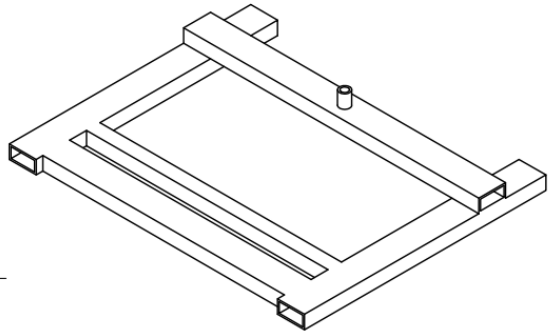
THE UNIVERSITY OF GEORGIA, ATHENS, GA

THIRD ANGLE PROJECTION ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	DRAWN BY Jack Cook	DATE 5/3/2024
	TITLE Top Frame	PART # SA13
	MATERIAL Generic	
	MASS N/A	SCALE 1 / 4 / SIZE B / SHEET 9 of 13

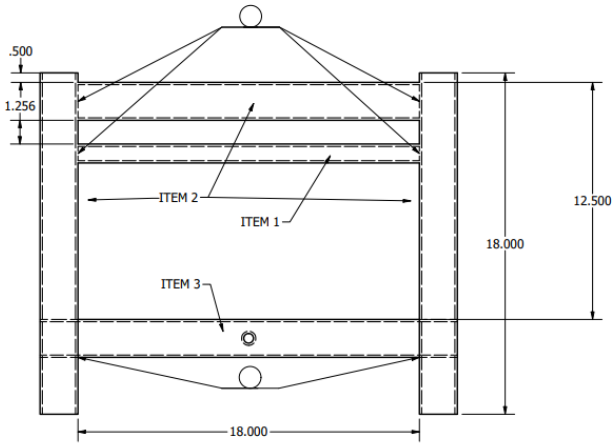
NOTES:
1. GRIND WELDS ON BOTTOM



SIDE VIEW
SCALE 1/4



ISOMETRIC VIEW
SCALE 1/4



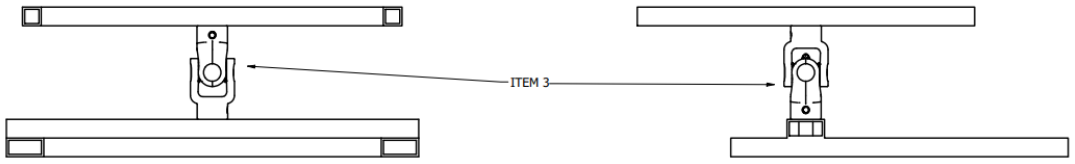
BOTTOM VIEW
SCALE 1/4

PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	P19	1in x 1in 11ga 18in length
2	3	P20	1in x 2in 11ga 18in length
3	1	SA11	1in x 2in 11ga 22in length w/ mount

THE UNIVERSITY OF GEORGIA, ATHENS, GA

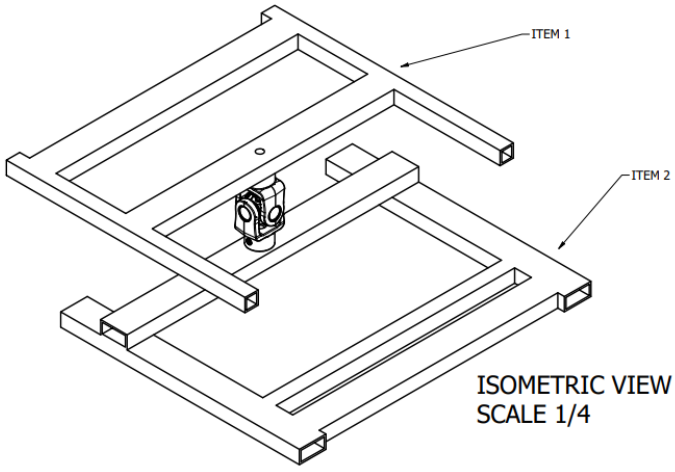
THIRD ANGLE PROJECTION
ALL DIMENSIONS ARE IN INCHES
TOLERANCES UNLESS OTHERWISE SPECIFIED:
x.x ±0.02
x.xx ±0.01 FRACTIONAL ±1/32
x.xxx ±0.005 ANGULAR ±0.5°
x.xxxx ±0.0025

DRAWN BY Jack Cook			DATE 5/3/2024
TITLE Bottom Frame Assembly			PART # SA14
MATERIAL Generic			
MASS N/A	SCALE 1 / 4	SIZE B	SHEET 10 of 13



FRONT VIEW
SCALE 1/4

SIDE VIEW
SCALE 1/4



ISOMETRIC VIEW
SCALE 1/4

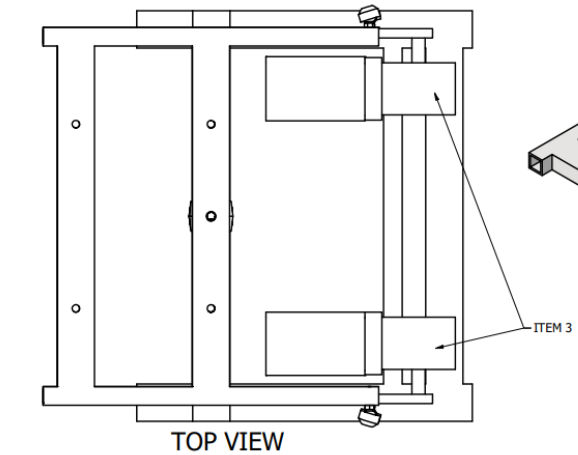
NOTE POSITION OF SET SCREWS ON U-JOINT FOR PROPER ORIENTATION

PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	SA13	Top Frame Assembly
2	1	SA14	Bottom Frame Assembly
3	1	--	U-Joint

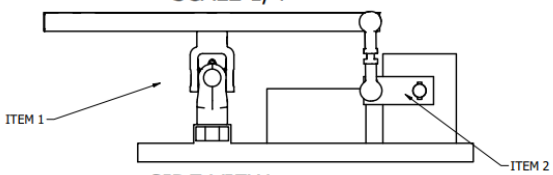
THE UNIVERSITY OF GEORGIA, ATHENS, GA

THIRD ANGLE PROJECTION
ALL DIMENSIONS ARE IN INCHES
TOLERANCES UNLESS OTHERWISE SPECIFIED:
x.x ±0.02
x.xx ±0.01 FRACTIONAL ±1/32
x.xxx ±0.005 ANGULAR ±0.5°
x.xxxx ±0.0025

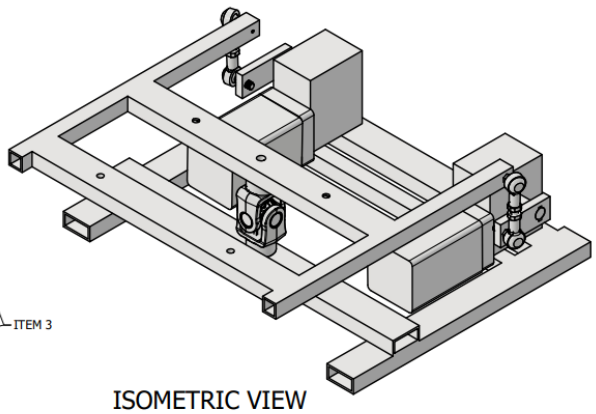
DRAWN BY Jack Cook			DATE 5/3/2024
TITLE Pitch and Roll Assembly			PART # SA15
MATERIAL			
MASS N/A	SCALE 1 / 4	SIZE B	SHEET 11 of 13



TOP VIEW
SCALE 1/4



SIDE VIEW
SCALE 1/4



ISOMETRIC VIEW
SCALE 1/4

PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	SA15	Pitch and Roll Full Assembly
2	2	P24	Shaft Arm
3	2	SA8	Motor and Gearbox Assembly

THE UNIVERSITY OF GEORGIA, ATHENS, GA

THIRD ANGLE PROJECTION	DRAWN BY: Jack Cook	DATE: 5/3/2024
	TITLE: Pitch and Roll Full Assembly	PART #: A3
ALL DIMENSIONS ARE IN INCHES TOLERANCES UNLESS OTHERWISE SPECIFIED: x.x ±0.02 x.xx ±0.01 FRACTIONAL ±1/32 x.xxx ±0.005 ANGULAR ±0.5° x.xxxx ±0.0025	MATERIAL	
MASS: N/A	SCALE: 1 / 4	SIZE: B — SHEET: 13 of 13