Self-Balancing Food Tray

By

Jay Kim

Mitchell Kremer

Taylor Xia

Final Report for ECE 445, Senior Design, Spring 2023

TA: Xiangyuan Zhang

5/3 May 2023

Project No. 31

Abstract

The Self-Balancing Food Tray is a small, easy to carry electronic multi-axis gimbal stabilizing system to be inserted/carried between the server's hand and tray that will stabilize the serving tray in real time when encountering smaller/slower impacts and disturbances. Using four robot arms powered by four motors, two pairs of arms will balance the tray individually for their assigned axis. It includes a fully functioning Power/Battery subsystem, Control subsystem, IMU sensor subsystem, and Balancing subsystem.

Contents

1. Introduction

1.1 Problem

Even for waiters and waitresses with experience, it may be a struggle to carry out and balance trays of food and beverages at restaurants while navigating around customers and rows of tables, especially with heavier and unevenly loaded trays. This is especially true when going to lower the tray down onto a nearby table or onto a folding servers table as the transition in height introduces potential dangers in stability. With the recent growing popularity of robotic and automated waiters, the need for a self-stabilizing platform or tray could also prove valuable to this emerging technology. Modern day waiter robots are slow, boxy, and require the user to ultimately still take the food off of the robot's carrying trays.

1.2 Solution

Our solution is to make a small, easy-to-carry electronic multi-axis gimbal stabilizing tray system to be inserted/carried by the server which will stabilize the serving tray in real time according to a calibrated initial angle reference. This would ensure extra stability to prevent drinks and food tipping over while serving customers as the server encounters smaller/slower impacts and disturbances. This would effectively allow the restaurant to save long-term costs on lost food, drinks, and dishware while preventing dangers such as hot food being spilled on the nearby patrons.

With a small stabilizing platform, robots can be built to move faster with less risk and can actually serve food to a table like an actual waiter would, being resilient to small impacts and disturbances. In the following sections we will go over our design processes behind creating our solution, the various requirements we set for ourselves and how we achieved those goals.

1.3 Block Diagram

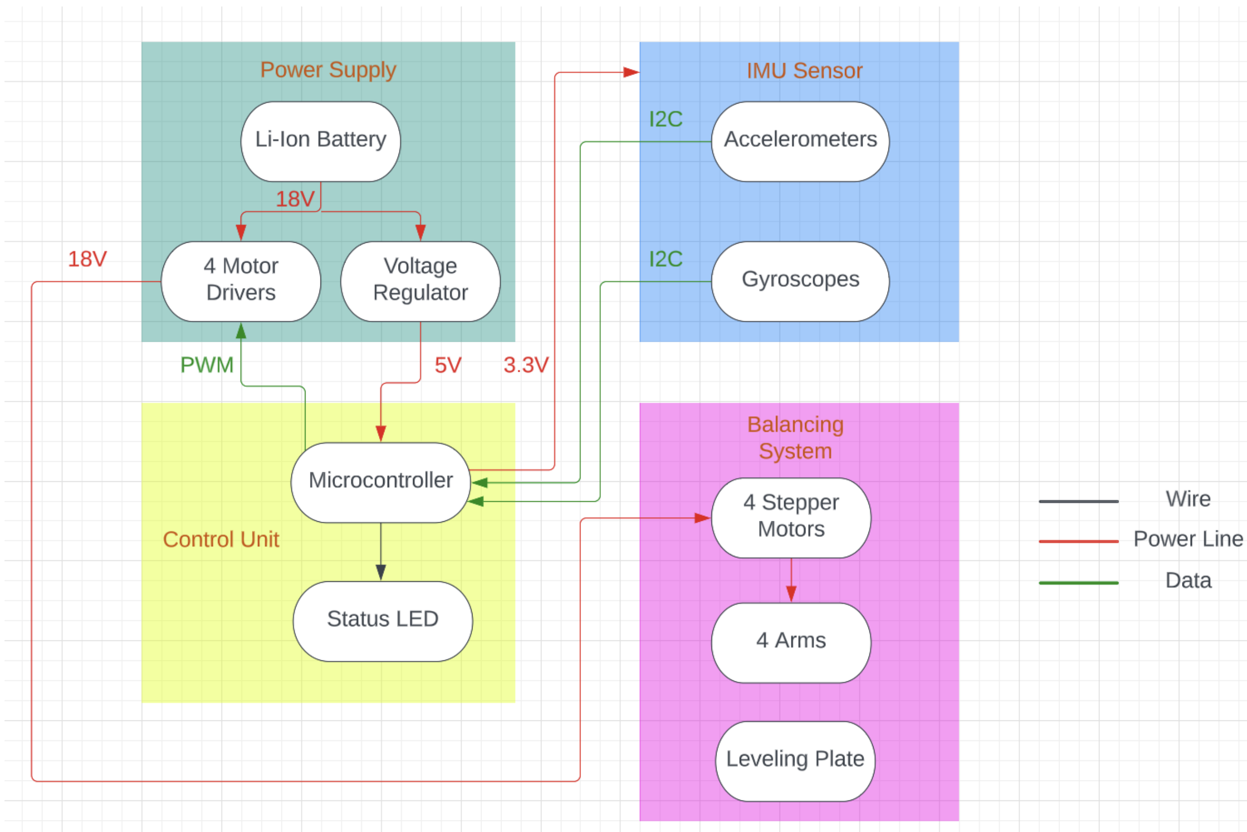


Figure 1 High level block diagram of final iteration of project.

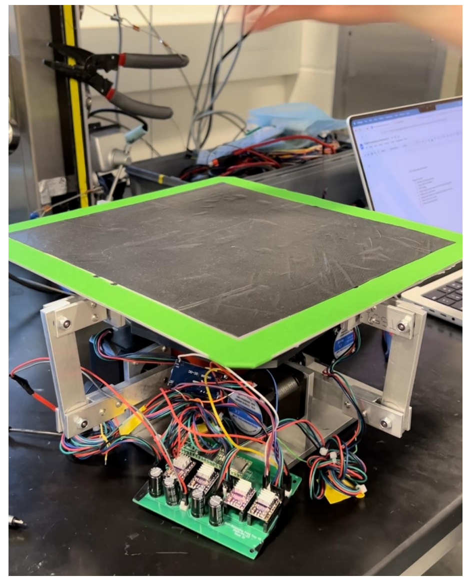
The design consists of primarily four subsystems: the Power Supply, Control Unit, IMU, and Balancing systems. The Power Supply would power the entire system using a Lithium Ion battery. The IMU sensor would provide the necessary angle data used for balancing purposes. The Control Unit would have the PID algorithm running using the inputted IMU data and send the necessary PWM signals to the stepper motors. The Balancing System would enact the balancing process using the stepper motors, arms, and the plate.

2 Design

2.1 Physical Design

Our design was primarily based off of an existing project titled the “Octo-Bouncer”. This project was created and shown in a video made by online user Electron Dust. In their iteration, the machine was made to simply keep a ping pong ball bouncing on the tray while also adjusting itself to keep the ball within the center. We decided to utilize the general design concept of this project and have our design also feature four stepper motor controlled arms that held up a food tray.

Initial Concept Design of Self-Balancing Food Tray

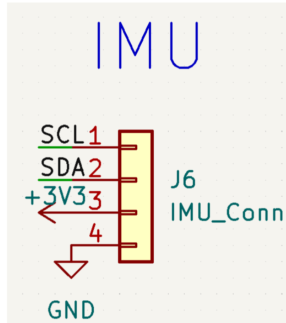


Final Design of Self-Balancing Food Tray

2.2 Sub-System Overview and Requirements

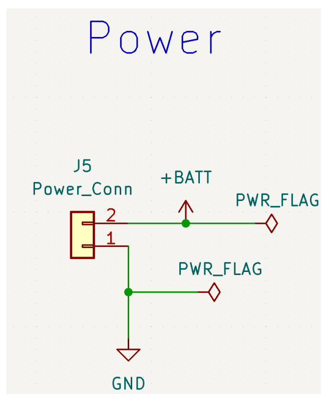
2.2.1 Inertial Measurement Unit

The inertial measurement unit (IMU) was our primary form of detecting the state of the tray and determining whether it is considered balanced for our system. The IMU contained gyroscopic and acceleration data, both of which we leveraged as part of our PID algorithm. The part that we chose to use for this project is the MPU-6050, of which we used its roll and pitch axis data. The IMU was attached to the bottom of the upper tray and connected to the PCB through jumper wires to allow for better mobility.



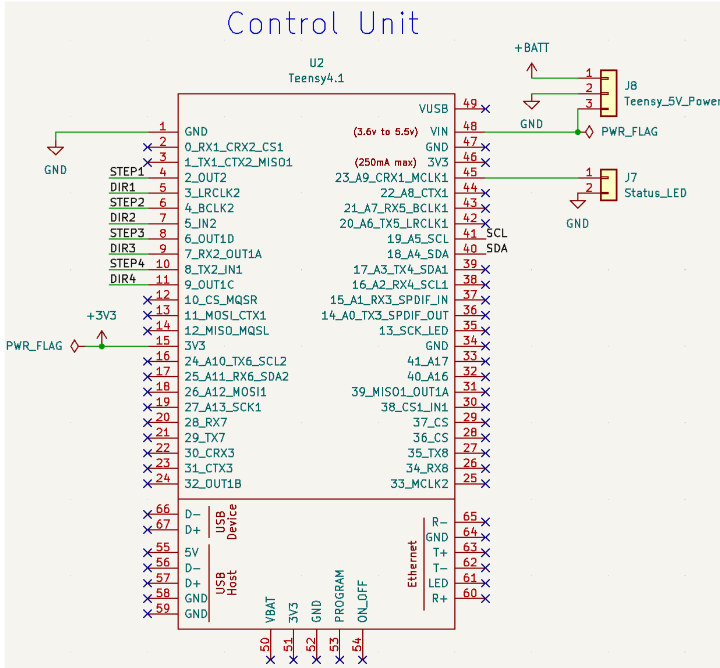
2.2.2 Power Supply

All four stepper motors were powered by a driver board. A key requirement of the driver was that it must be capable of delivering enough current and voltage to drive the motor it is assigned to while also being capable of handling the same or higher number of microsteps as the motor. This power driver was in turn connected to a rechargeable battery. To meet these as well as our other requirements (see section 3), we chose to use a Milwaukee 18V 2.0 Ah rechargeable battery which powered the four DRV8825 stepper motor drivers, Teensy 4.1 microcontroller, and MPU-6050 IMU.



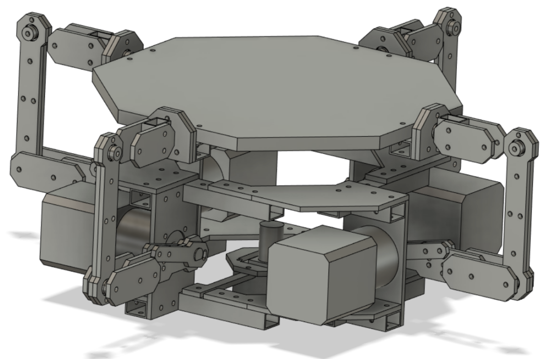
2.2.3 Control Unit

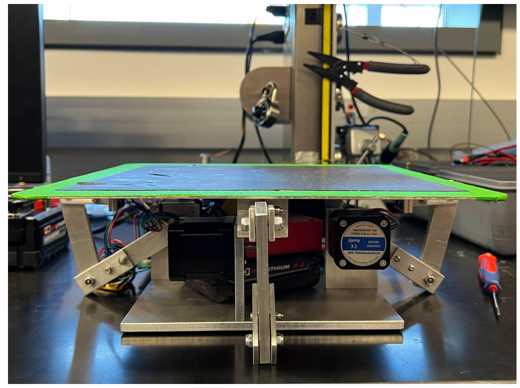
The Teensy 4.1 microcontroller takes the IMU's gyroscope and accelerometer data to measure angular tilt and velocity. This data is used to make the necessary adjustments as it is fed into the PID algorithm running onboard the microcontroller, which sends the adjustments in angle as PWM signals to the Balancing System based on the initial calibration of a desired angle. The start and end of the calibration process is indicated by the turning on and off of the Status LED. The reason we chose to use a Teensy instead of a more popular controller such as an Arduino board is due to the maximum analog output frequency the Teensy is capable of. A Teensy 4.1 can provide a clock speed of 600 MHz in contrast to the 16 MHz of an Arduino Uno, which we believed was better suited for better PID functionality. For our solution we would want the motors to move as smoothly as possible such as through microstepping. An Arduino board cannot produce analog signals in high enough frequency to facilitate microstepping but a Teensy can.



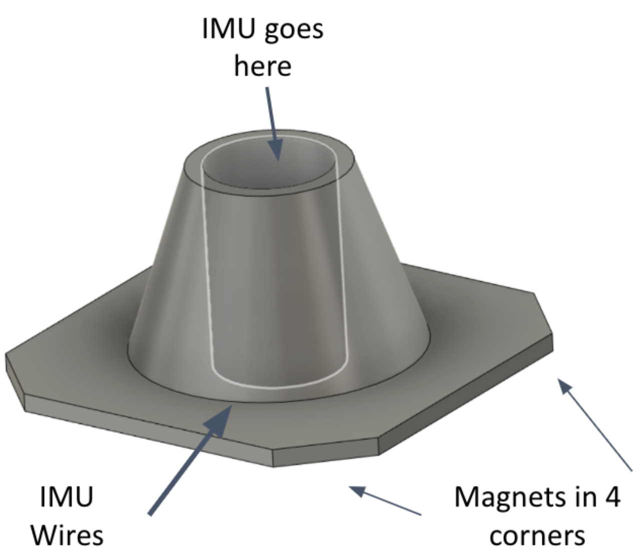
2.2.4 Balancing System

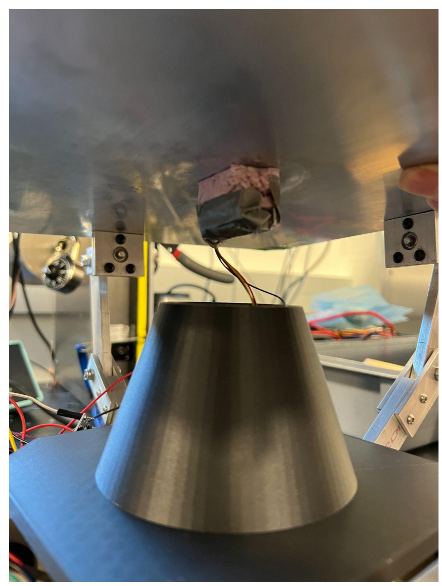
The balancing system will consist of the aforementioned four stepper motors. These motors control the distance the four arms will bend or extend. The arms themselves consist of three solid pieces: one piece connected to the tray, another connected to the motor shaft, and lastly a linkage piece between the two. The three pieces are connected via two revolute joints which allows the arm to freely rotate in one degree of freedom (up and down). The top plate is then connected to these four arms via a swivel joint which allows the table to be angled in another direction without ripping off the attachment to the neighboring arm. Overall balance plate adjustments will be made on the pitch and roll axis, not on yaw. The stepper motors we chose to use will be Nema 17s. These 18V 2 amp motors are powerful enough to lift the leveling plate up and down via turning of the arms in each direction. We run opposing motors in conjunction with one another as there are only two axes that need to be accounted for. While one motor is raising up the arm the opposing motor is pulling down an arm. This essentially provides us with double the lifting power and keeps the movement much more rigid than if only one motor were conducting the movements. All four of these motors can move in conjunction with one another allowing us to compensate for movement in both axes at the same time, which is critical in this use case.

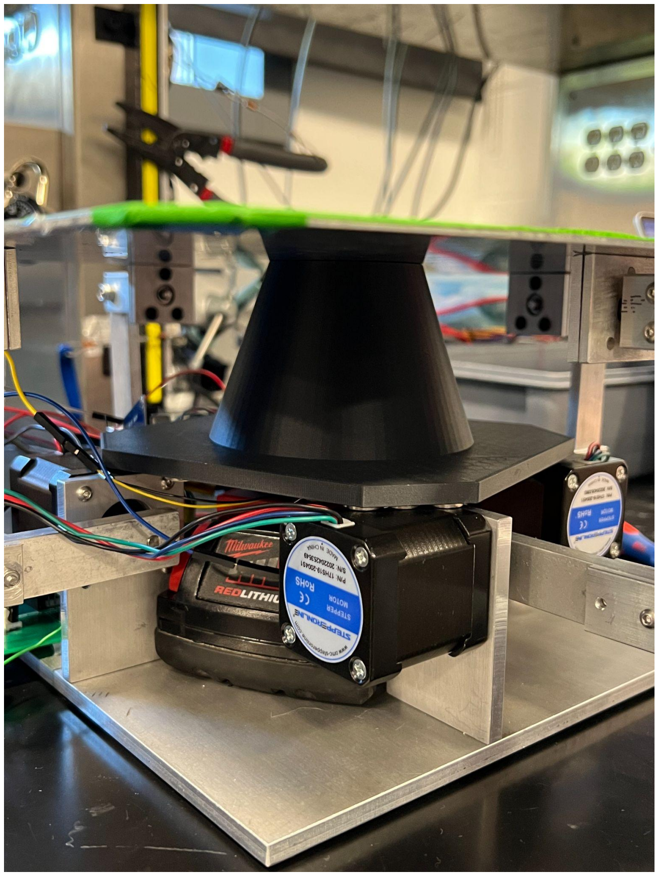




We encountered an issue where the motors had difficulties handling the weight of the arms, the metal plate, and the additional load placed on top of the tray. During tests, we found that supporting the arms with spools of wire allowed for the motors to focus their efforts primarily on the balancing of uneven weights while the spools handled the majority of the overall weight. This led to the creation of what our team calls, the Volcano. Shaped like a volcano, it is magnetically attached to the motors and handles a significant amount of weight while minimizing the effects on the balancing movements with a circular end. With a cylindrical vent, the IMU attached to the bottom of the upper plate wouldn't be affected and the vent allowed for wires to be fed through to connect the IMU and PCB.







The Volcano integrated into the final product

3. Design Verification

3.1 Inertial Measurement Unit

Requirements	Verifications
Deliver gyroscope and accelerometer data within accuracy threshold of 0.1 degrees and 0.01 meters per second squared.	Check gyroscope and accelerometer data with known constant angles/acceleration and confirm accuracy of 0.1 degrees and 0.01 meters per second squared.
No drift large enough to affect data accuracy and PID algorithm by more than 5% std.	Record data over a period of 10 seconds with constant load and plot error data to confirm accuracy of 5% std.

For testing these requirements, we ran our PID balancing code on the Teensy while a weight of around 1 lb was placed in the center of the plate. From our data graphs, our gyroscope had an accuracy of around 0.3 degrees, IMU giving readings between 0.3 to -0.3 with the plate being at verified 0 degrees. These values are after various attempts to reduce noise such as enabling the built in low pass filter. The accelerometer was closer to an accuracy of around 0.15 m/s^2 . Despite not meeting our verification goal, the plate functions fairly well under use so it may be acceptable to leave.

As for the standard deviation, we modified a part of our code to record the gyroscope readings for the first 10,000 milliseconds and used the sample standard deviation equation to solve for the value. We ended up with a standard deviation of around 2-4%, within our predefined acceptable range.

The rest of our gyroscope and accelerometer code dealt with converting the radian measurements given to degrees. We did this through a series of arctangent operations, adding PI to those output values, and then using radian to degree constant (180/π) to find the degree angle of the plate.

$$s = \sqrt{\frac{\sum (x - \bar{x})^2}{n - 1}}$$

Sample standard deviation formula using in verification

3.2 Power Supply

Requirements	Verifications
All four stepper motors should be able to run smoothly at 75% max voltage for at least one hour at half maximum load.	Confirm that the balancing system can run for one hour at half maximum load.

Quick swappable battery system will be replaceable by the user in less than one minute.	Run 10 trials of replacing the battery within a minute and have at minimum 8 successes. These trials will be done by volunteers who are informed of the proper way of replacing the batteries.

While we have definitely run the entire system for longer than an hour, we verified our requirement by checking the power rating of our battery. The 18v Milwaukee battery we chose to use can output a maximum of 2Ah thus giving it 36Wh worth of power. The Teensy's rated maximum power draw is 1.75W; in our use case it likely only draws less than a watt. Thus almost all of the 36W goes toward powering the motors. Each Nema 17 motor requires between 12v-24v and can draw up to 1.5A of current. Our project provides 18V and each motor uses around 0.5A based on our testing using the lab power supply. Thus the system will run for one hour and fulfills our requirement.

Our second requirement was verified exactly as we described; from our own various trials, we determined that the time to change the battery within the balancing system is around 35-50 seconds. This falls within our predefined time frame.

3.3 Control Unit

Requirements	Verifications
PID algorithm must be able to stabilize sudden increases in weight of up to 1 lb on outer edges of tray	Test by lightly dropping weights of up to 1 lb on an empty tray and see if it can counter the imbalance while not dropping the weight.
Microcontroller buses must be able to send data to all 4 motors simultaneously up to a PWM frequency of 200 MHz	Create a test script that demonstrates stepper motor capabilities to rapidly change speeds and/or direction based on inputted wave-like motions.

The first requirement we completed just as we were building our balance plate was the second one regarding the motors being able to rapidly change speeds. We wrote many such scripts during the construction and programming of the system and thus we can confidently say that the requirement has been met.

The other requirement was also completed after the PID algorithm was fully developed and tuned. Testing showed that the plate could handle a sudden shift in weight and readjust albeit a little slowly as the settling time for the PID algorithm was a little long. Nevertheless, the plate settles and so the requirement is verified.

3.4 Balancing System

Requirements	Verifications
The system can balance itself to within while the user is walking at an average adult walking pace of 3 mph.	Place weights totalling 2 lbs dispersed along the tray while walking at a speed of 3mph. We will measure the distance walked and the time walked to calculate the speed.
Create four arms of identical specifications with at most 3 mm of variance between all parts.	Measure finished parts with digital calipers to verify accuracy of dimensions between all parts.
Use stepper motors which are rated at greater than .25 ft pounds of torque each to carry a load of 2 lbs on the tray.	Test by placing a weight of 5 lbs within the central area of the tray and seeing if load can be handled adequately.

Our balancing system contained a lot of requirements mainly due to the fact that it is a very central and integral part of our project. Thankfully, most of these requirements were verified simply by us testing our high level requirements. While testing our high level requirement for walking around with 5 water cups, we decided to ensure all the cups and water together were around 2 lbs in weight. Thus, this requirement is fulfilled.

The 5 lb weight was tested separately by simply doing what we wrote in our verification. We placed a 5 lb weight on the center of the balance tray and it held up.

The parts are all within acceptable tolerances purely due to the fact they were manufactured using a CNC machine. Throughout our usage and testing of the system we didn't notice any extensive friction or drift due to the parts themselves, so we decided that this requirement

4. Costs

4.1 Parts

Description	Price Per Unit	Total
4 Nema 17 Stepper Motors	$13.00	$52.00
4 TI DRV8825 Driver Chips	$18.00	$72.00
1 Teensy 4.1	$31.50	$31.50
1 Adafruit 3886 IMU	$12.95	$12.95
Raw Aluminum Frame Material	$10.00	$10.00
Fabrication of Frame	$100.00	$100.00
Total		$278.45

4.2 Labor

Description	Price per Unit	Total
Taylor Labor	$50.50 per hour 8 hours per week for 8 weeks	$3,232
Mitchell Labor	$50.50 per hour 8 hours per week for 8 weeks	$3,232
Jay Labor	$50.50 per hour 8 hours per week for 8 weeks	$3,232
Total		$9,696

4.3 Total

Description	Total
Labor	$9,696
Parts	$278.45
Overall Total	$9,974.45

4.4 Schedule

Week	Task	Taylor	Jay	Mitchell
Feb 20 - Feb 24	• Begin ordering all parts	•	•	•
	• Finish Design Document	•	•	•
	• Begin Receiving Parts	•	•	•
	• Complete mechanical design of components			•
Feb 27 - Mar 3	• Send out parts to be machined			•
	• Start Looking into PCB design	•	•	•
Mar 6 - Mar 10	• Submit PCB design Order	•	•	•
	• Plan to receive most of machined parts at during the week	•	•	•
Mar 20 - Mar 24	• Assemble frame of balancing system			•
	• Program and prototype Teensy for stepper motor	•	•	

	control			
	● Ensure reliable I2C data reading IMU	●	●	
Mar 27 - Mar 31	● Program PID control loop for control unit and IMU subsystems	●	●	
	● Assemble physical parts of the balancing subsystem.			●
	● Second PCB order if necessary	●	●	●
April 3 - April 7	● Anticipated troubleshooting of software	●	●	●
	● Miscellaneous adjustments	●	●	●
April 10- April 14	● Filler week for handling errors	●	●	●
April 17 - April 31	● Filler week for handling errors	●	●	●

5. Conclusion

5.1 Accomplishments

Overall our project was quite successful. We ended up with a functional product capable of completing the 3 high level requirements we set out to complete in our design document. Our final product is capable of doing active stabilization of objects exceeding 2 pounds to within 3 degrees of accuracy. This was done with a completely handheld design with a working PCB and power supply able to be used completely untethered.

5.2 Uncertainties

The main uncertainty we have with our current implementation is simply the speed with which it reaches a level state. Although we did not explicitly set a desired time to reach a level position, to make this product practicable in a restaurant setting, we would need to continue to improve the PID control and speed of the motors to be able to reach a steady state quicker than we are currently able. With the slow stabling time of the system currently, when walking at a normal rate the system isn't able to properly compensate for movement before a movement occurs in another direction that leads to a domino effect of shaking back and forth continually trying to find a level state.

5.3 Ethical considerations

Code I.1 of the IEEE ethics code states, "to hold paramount the safety, health, and welfare of the public, to strive to comply with ethical design and sustainable development practices, to protect the privacy of others, and to disclose promptly factors that might endanger the public or the environment". Our gimbal system will strive to provide a safe experience, in accordance with code number one of the IEEE code of ethics, to both the user and customers who may be at the receiving end of the product's capabilities.

Code II.5 of the IEEE ethics code states, "to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, to be honest and realistic in stating claims or estimates based on available data, and to credit properly the contributions of others". We will provide credit to sources we received inspiration from and or received pertinent information from in the construction of our project in accordance with the code of ethics.

5.4 Future work

We believe this project has some exciting possibilities for future use cases in various industries. However to truly become useful certain changes would need to be made. One of the more intriguing potential use cases is active leveling dining tables on board cruise ships. To do this we would obviously need to make a largely scaled up version and likely significantly change the design itself. We would likely transition to a ball screw design to control the up and down movement of the table, shifting away from the arms we have currently been using. This design change would lead to carrying much heavier loads while at the same time being able to move faster and more accurately. Obvious drawbacks to this design change would be the increased price but for a dining room table sized device it would be a necessary change. For the server tray use case, continued improvement of the PID algorithm and decreased weight of the unit itself would be necessary improvements.

References

[1] (n.d.). "Female Hand Holding Something." Depositphotos. https://depositphotos.com/77620065/stock-photo-female-hand-holding-something.html

[2] Kuhn. (2021). "The Octo-Bouncer: Advanced Bouncing Patterns." YouTube. https://www.youtube.com/watch?v=lYyAMDYzJQM

[3] Food Tray: (n.d.). "Oval Restaurant Serving Trays NSF Certified Non-Skid Food Service Hotel Bar Tray." Overstock. https://www.overstock.com/Home-Garden/Oval-Restaurant-Serving-Trays-NSF-Certified-Non-Skid-Food-Service-Hotel-Bar-Tray/30827966/product.html

[4] "IEEE code of ethics," IEEE, Jun-2020. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 6-Feb-2023].

[5] CarbonAeronautics, Github. 2022. *Gyroscope Calibration*, ver. 1. [Online]. Available: https://github.com/CarbonAeronautics/Part-V-GyroscopeCalibration/blob/main/ArduinoCode

[6] luni64, Github. 2020. *TeensyStep4*, ver. 1. [Online]. Available: https://github.com/luni64/TeensyStep4

[7] J. Bennett, A. Bashin, J. Grant, and W. C. Lim, "9.3: PID tuning via classical methods," *PID Tuning via Classical Methods*, 11-Mar-2023. [Online]. Available: https://eng.libretexts.org/Bookshelves/Industrial_and_Systems_Engineering/Chemical_Process_Dynamics_and_Controls_(Woolf)/09%3A_Proportional-Integral-Derivative_(PID)_Control/9.03%3A_PID_Tuning_via_Classical_Methods. [Accessed: 03-May-2023].

[8] Last Minute Engineers, "In-depth: Interface MPU6050 Accelerometer & Gyroscope sensor with Arduino," *Last Minute Engineers*, 29-Oct-2022. [Online]. Available: https://lastminuteengineers.com/mpu6050-accel-gyro-arduino-tutorial/. [Accessed: 21-Mar-2023].

[9] P. Thangavel, "DIY Arduino digital protractor using MPU6050 Gyroscope," *Circuit Digest - Electronics Engineering News, Latest Products, Articles and Projects*, 03-Jul-2019. [Online]. Available: https://circuitdigest.com/microcontroller-projects/arduino-based-digital-protractor-using-mpu6050-gyroscope. [Accessed: 18-Apr-2023].