Olivia Ma

Engineering Portfolio

My name is Olivia Ma, and I am a Robotics Engineering Master's student at the University of Michigan - Ann Arbor. I will be graduating in May 2026, and I am looking for full-time positions in robot/motor control. I have a background in hardware design and autonomous aircraft, and am passionate about solving unique, highly-dynamic controls problems and optimizing for energy efficiency.

Previously, I have been a GNC Intern at Reliable Robotics (2025), Hardware Engineering Co-op at Amazon Robotics (2024), and Motor Controls Intern at Stellantis (2023). I was also an Undergraduate Research Assistant at the A2Sys Lab working on cheap, precision hexacopter control. I was also Structures team lead for Michigan Autonomous Aerial Vehicles (MAAV) student project team, leading hardware design for various VTOL vehicles. This portfolio will take you through my recent projects.

Feel free to contact me with questions or comments at oliviiam1@gmail.com.

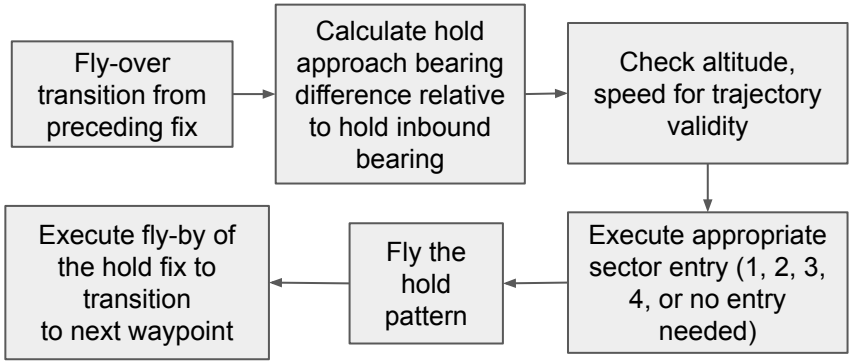
Controls Projects

Reliable Robotics GNC Intern Guidance Project

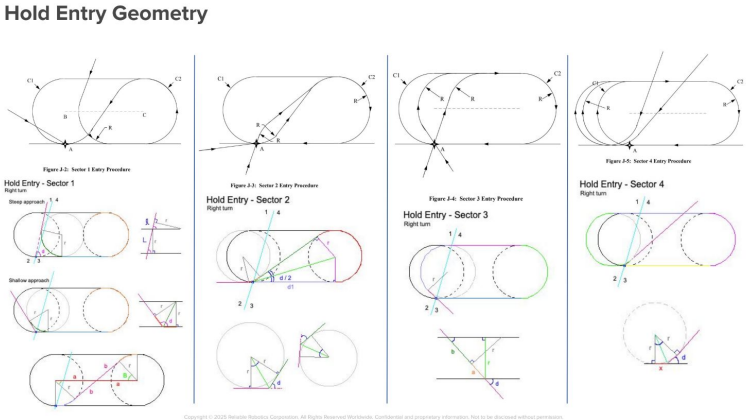
I was tasked with generating trajectories for autonomous holding patterns. A flight hold is a racetrack-shaped flight path planes might take while waiting (e.g. if a runway is backed up for landing).

I did background research to understand how the FAA defines a hold and what circumstances a pilot may enter a hold under. I worked within the guidance stack to generate and store hold information and set up the pipeline to insert hold legs into the flight plan. I added a state machine to appropriately handle all entry and exit conditions and edge cases to ensure the aircraft would only initiate viable holds at safe altitudes and speeds.

I delivered the hold leg trajectory generation capability, along with a suite of unit tests and integration tests that could be run nightly to check stability as the code base continued to evolve.



Rough flow chart for hold trajectory generation and execution



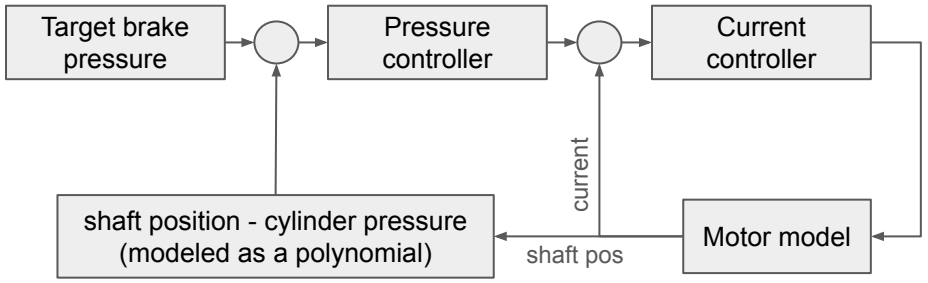
Calculations I made to break hold entries and racetracks into supported geometries

Reliable Robotics GNC Intern Modeling Project

For this project, my goal was to model the braking system and control surface systems in C++, to be integrated into the simulation.

I started from first principles, with motor electrical and mechanical dynamics. I used existing data to model inertial, viscous friction, and gear compliance effects. I modeled gear compliance as a mechanical damping. I then modeled mechanical dynamics of the aircraft systems driven by the motor. The aircraft system model requires a certain input torque to achieve its desired state, which is passed as an input to the motor model as an external torque on the output shaft. I applied PID controllers to the systems (assuming perfect sensor data).

I unit tested my models by passing step inputs to the motor model and system models inputs.



Block diagram of brake pressure model + control loop

Electrical dynamics for motor model

$$\dot{I} = \frac{V_a - K_e \omega_{motor} - IR}{L}$$

V_a = applied voltage
K_e = back emf const
ω = motor speed
I = current
R = resistance
L = inductance

Mechanical dynamics for motor model

$$J_{total}\dot{\omega}_{load} = \tau_{app} - \tau_{load} - \tau_{motor} - \tau_{ext}$$

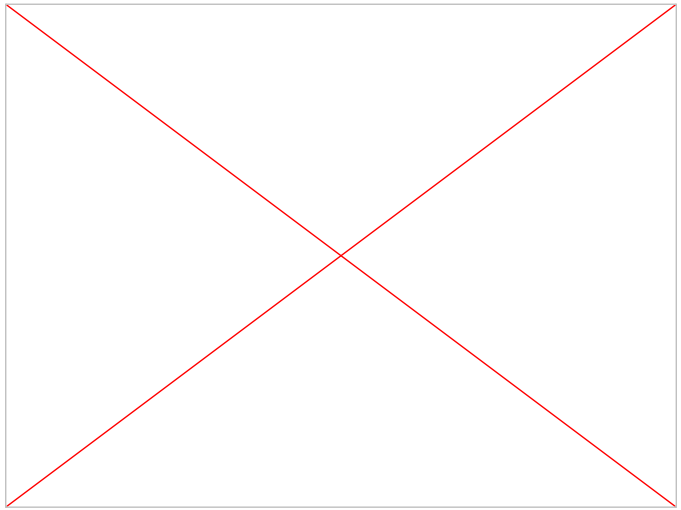
J = moment of inertia
ω = motor speed
τ = torque (applied by motor, load friction, motor friction, external)

A2Sys Pogo Drone Switching Controller + Tuning

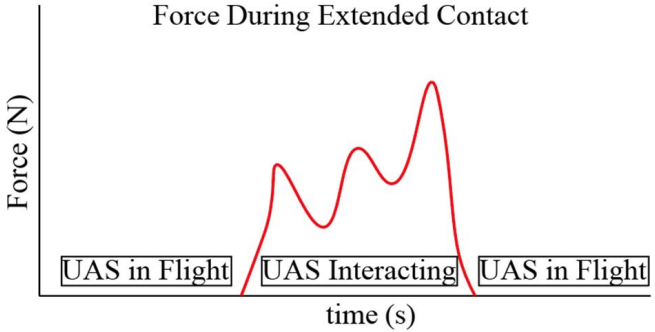
The goal of this project is to demonstrate a controlled, targeted bouncing technique for the application of collecting treetop samples in the Amazon rainforest. To this end, we developed a controller to safely recover an underactuated system under collision.

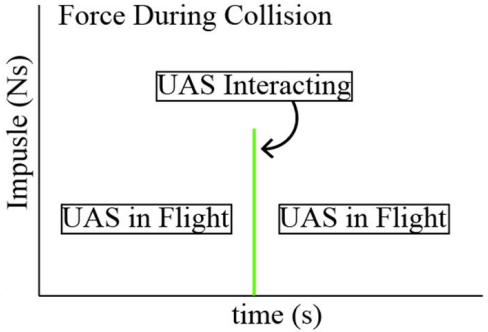
UAS typically struggle with meaningful physical interactions with environments due to the coupling between the UAS and the environmental dynamics, as external forces will disrupt the UAS's attempts to keep itself stable. This effect is doubly observed when attempting to interact with flexible/unpredictable targets, such as tree branches.

The idea is that planned collisions will enable UAS to maintain stability through environmental interactions via short, controllable impacts (a la Marc Raibert's hopping robots).

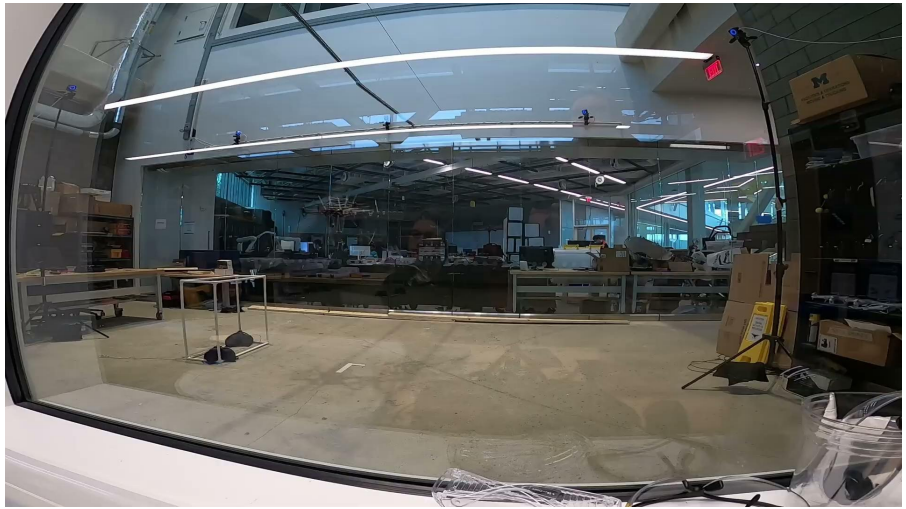


Switching control with bounce on "virtual ground" for safety



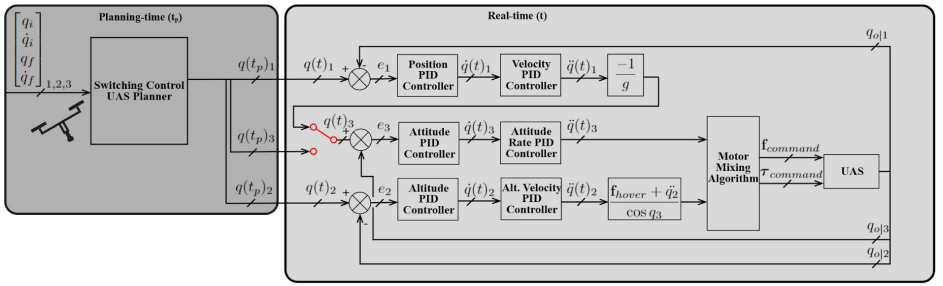


We first proceeded with a velocity-matching approach, which enabled collision recovery by controlling the attitude to align the pogo stick with the UAS's CoM velocity vector while matching a desired horizontal velocity. We identified a contour of recoverable flights (varying initial altitude, horizontal velocity) in simulation and validated experimentally. I conducted flight tests for PID tuning and data collection (paper here).



Velocity matching control with bounce

We then moved to a switching controller, for which I contributed to the development of the state machine architecture in C, and led flight testing to tune the cascaded PID controller. I also isolated mismatch between our simulation and flight test time delays created by the command → messaging/mocap → motor spin up pipeline, leading to poor performance of optimally generated trajectories.

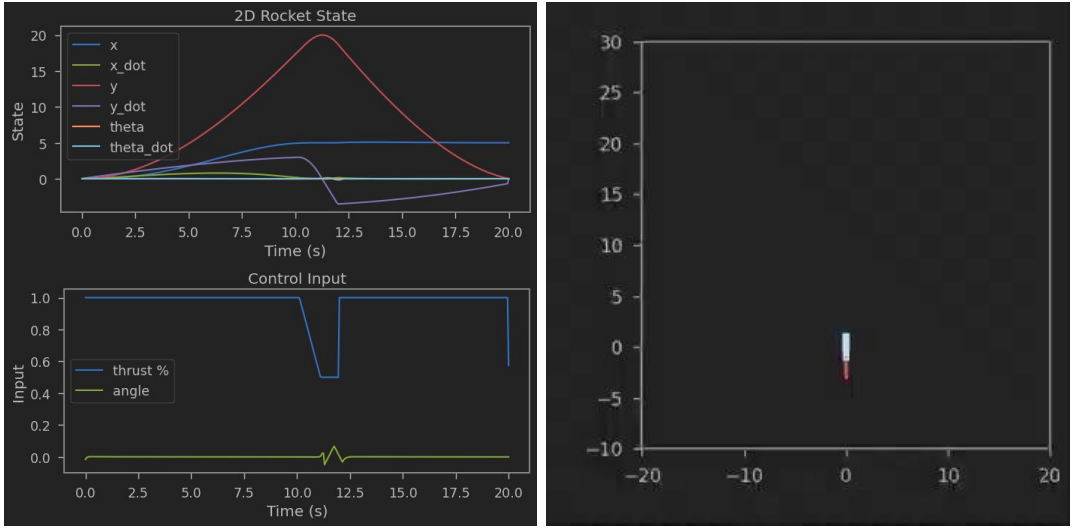


Cascaded PID Switching Controller Architecture

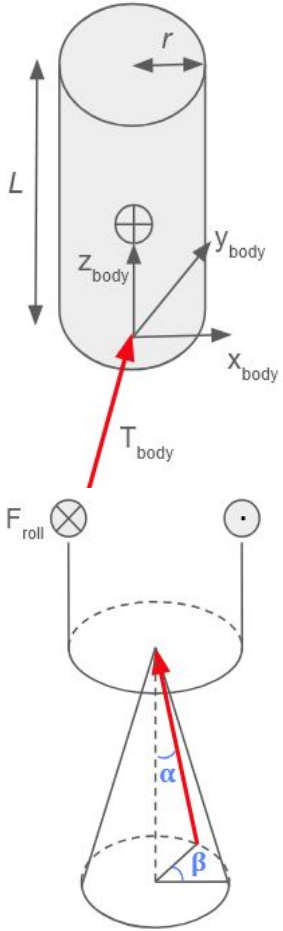
MERO Self-Landing TVC Rocket Sim + Control (WIP)

My friends and I started this project team to challenge ourselves to design and test a self-landing rocket over the span of our 1-year Masters. My role is to simulate and design a control system for the rocket to be able to perform a hop – fly up to a designated waypoint, and perform a controlled landing. The rocket will be controlled using one gimbaled, throttlable engine, and a series of ducted fans for roll control.

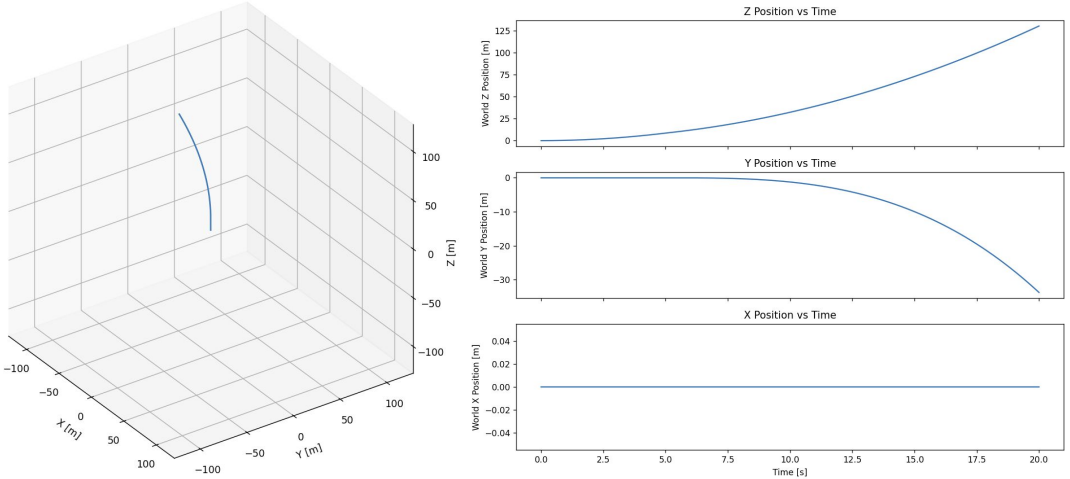
Given an engine's thrust curve, I used CasADi to generate optimal trajectories and evaluate feasibility. My optimization included the rocket dynamics, max gimbal angle and rate, and throttle limits of 50-100%. Through this method, we adjusted our apogee height and landing location requirements to match our engine's capability.



2D Rocket State and Control Input Plotted Over 20 sec. Burn Time (+Video)



I also developed a simple 3D simulation in Python, which models the rocket as a cylinder with uniform mass. I derived the dynamics from first principles, then integrated the rocket state forward through time using forward Euler. I validated my simulation by giving it open-loop control inputs and matching the results to my expectations based on hand calculations.



3D Simulation with Open Loop Control (max thrust for 0.5s at t=0, then a pitch-over command for 0.5s at t=5)

```python
# --- Translational dynamics ---
accel_w = T_w / self.mass + np.array([-self.g, 0, 0])
vel_next = vel + accel_w * self.dt
pos_next = pos + vel_next * self.dt
self.vel_series[:, t+1] = vel_next
self.pos_series[:, t+1] = pos_next

# angular accel in body: I*ωdot + ω×(Iω) = M_body
omega_b_dot = np.linalg.solve(self.I, M_b - np.cross(omega_b, self.I @ omega_b))
omega_b_next = omega_b + omega_b_dot*self.dt
self.omega_b_series[:, t+1] = omega_b_next
self.theta_b_series[:, t+1] = self.theta_b_series[:, t] + omega_b_next*self.dt
dtheta = np.linalg.norm(omega_b_next*self.dt)

# skew matrix of angular accel axis unit vector
K = hat(omega_b_next * self.dt / (dtheta + 1e-16))
# Rodrigues rotation formula: rotation of R by dtheta about axis omega_b_next
dR = (np.eye(3) + np.sin(dtheta)*K + (1 - np.cos(dtheta))*K@K)

# updated rotation matrix from world to body frame
R_next = R @ dR
```

Forward Euler Step for Translational and Rotational Dynamics

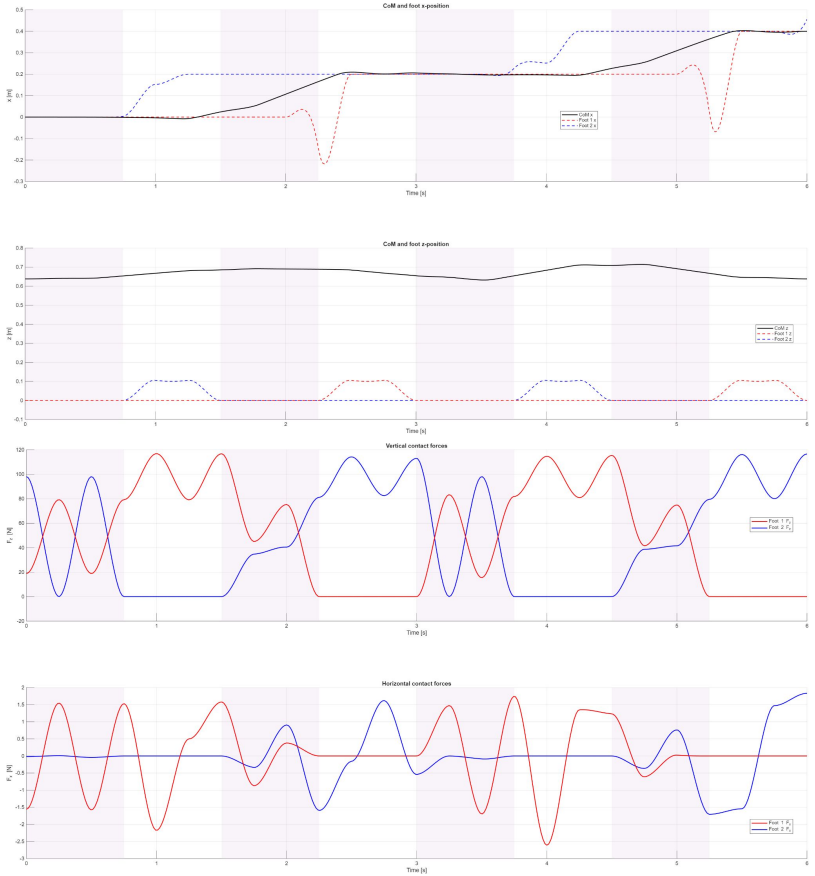
My next steps are to improve the 3DOF simulation (making it 6DOF, adding throttle rate, CoM changes as fuel is consumed, air resistance, more realistic geometry/mass distribution) and improve on the 6DOF cascaded controller.

ROB 599 (Legged Robot Control) Final Project

My team and I replicated Winkler et al.'s phase-based end-effector parameterization using CasADi and visualized results in MuJoCo. We evaluated the NLP on a 5-bar link planar bipedal robot model ("Rabbit") walking on an even surface.

We modeled the gait using a continuous phase-based parameterization for each foot. This allows the optimizer to smoothly adjust step timings to discover feasible gait patterns. We enforced rigid-body dynamics and kinematic limits at equal intervals across the trajectory to ensure physical feasibility. The position and forces of each foot were represented by a series of 3rd order Hermite polynomials, with constraints enforced based on the stance/swing phase. We represented the CoM trajectory as a series of 4th order polynomials, and constrained it to the forces and torque generated by the contact forces at the feet.

The optimizer successfully generated a coherent 4-step forward walking motion, demonstrating sensible vertical CoM oscillation and smooth stance-swing force activation (load transfer) between the two legs. We were able to successfully construct a whole-body controller to replay a simple leg swing in MuJoCo, but we were ultimately unable to replay more complex motion.



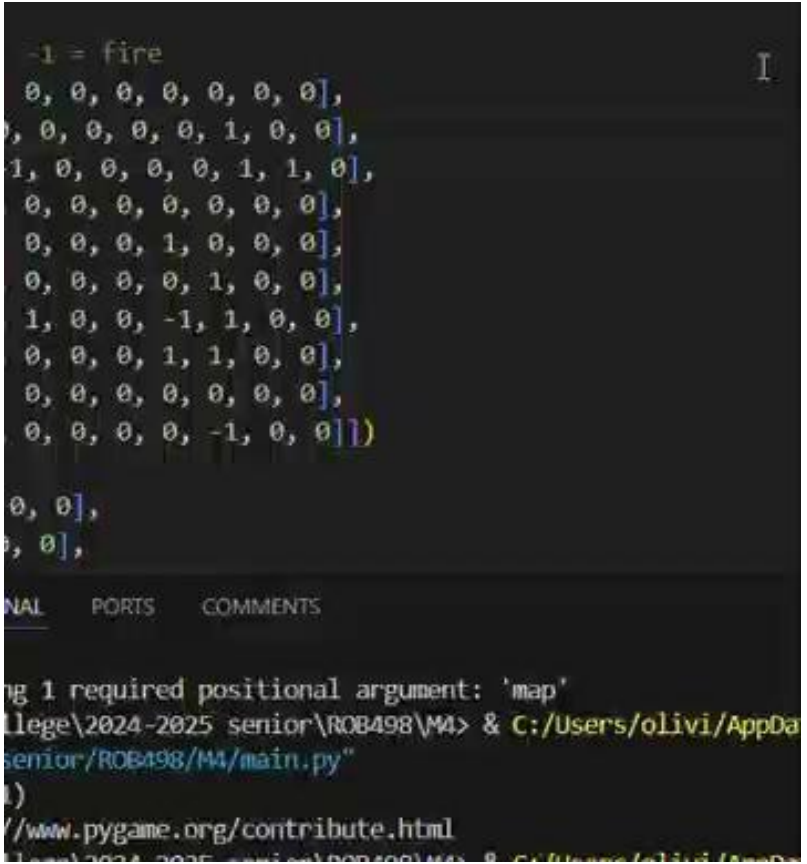
Foot and CoM trajectories and forces over a 4-step walking gait

ROB 498 (Multi-Robot Control) Final Project

My goal was to explore coordination across multi-robot teams by extending the homogeneous team coordination covered in class to heterogeneous teams – that is, teams comprised of distinct types of specialized robots. I constructed a forest-fire problem, in which teams of Surveyor robots (fixed-wing UAS) and Firefighter robots (quadrotors) were deployed to identify and extinguish fires.

I implemented a forest fire simulation in Python, representing the workspace as a 10x10 grid, with random cells of fire (increasing odds for cells adjacent to fire). I implemented the Surveyor robot using an information-gain exploration method with A* path-planning to efficiently map the evolving environment, then used a distributed optimization algorithm to assign Firefighter robots to the nearest, largest fires.

I implemented the drone swarm as a distributed system, meaning there is no central coordinator. Firefighters could exchange info with Surveyors and other Firefighters within their communication radius to maintain updated maps and solve the Multiple Traveling Salesman problem among themselves to find the optimal fire-assignment.



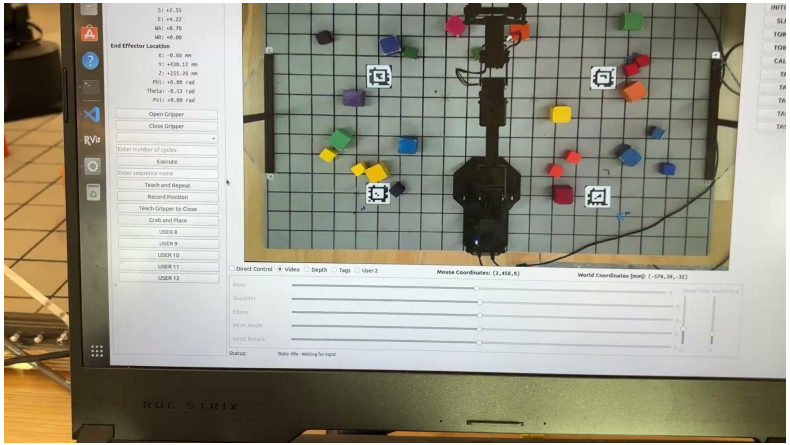
Recording of 1 Surveyor robot mapping the environment

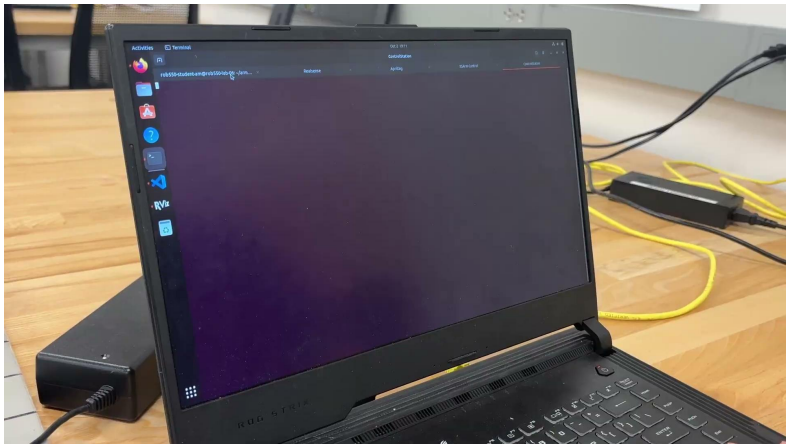
ROB 550 (Armlab) Final Project

Through this project I practiced my skills using forward/inverse kinematics, coordinate transforms, and computer vision on a 5DOF actuated arm.

My teammate and I used ROS2 and OpenCV's built-in functionalities to find the depth camera's internal and external matrices, which we used to normalize the workspace image and convert coordinates between the camera frame and world frame. We also used OpenCV to filter and segment the image by color and depth values in order to identify small and large blocks, with their centers and orientations.

I calculated the forward and inverse kinematics analytically. I used the Product of Exponentials method for FK by identifying the end effector's matrix in the arm's home pose, then finding the screw vectors generated by each joint. Then, by leveraging the exponential representation of a rotation about axis ω by angle θ ($e^{[\omega]\theta}$), I could find the end effector's state given arbitrary joint angles. I computed the inverse kinematics by considering the arm as a 2-link sub-arm and 2-axis wrist joint, with the yaw of the arm determined by the base joint. This greatly simplifies the IK, but I had to take extra care to address degenerate solutions (elbow up vs elbow down).





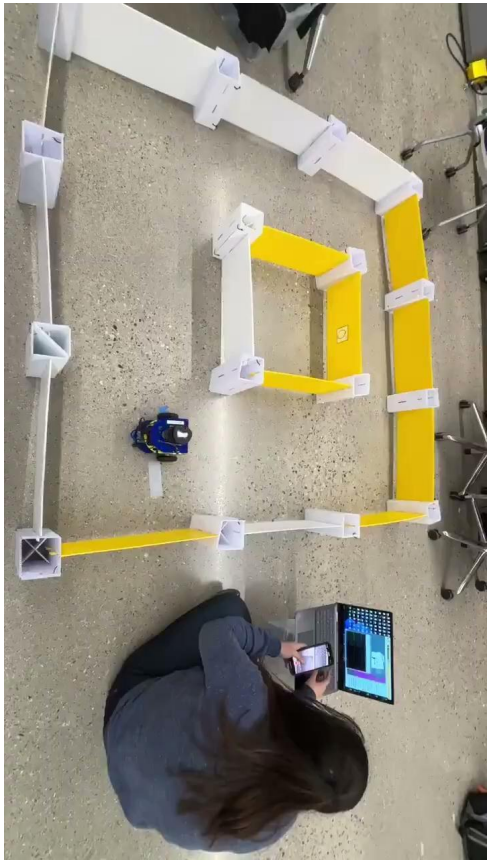
Demonstration of "Click and Place" and Block Detector

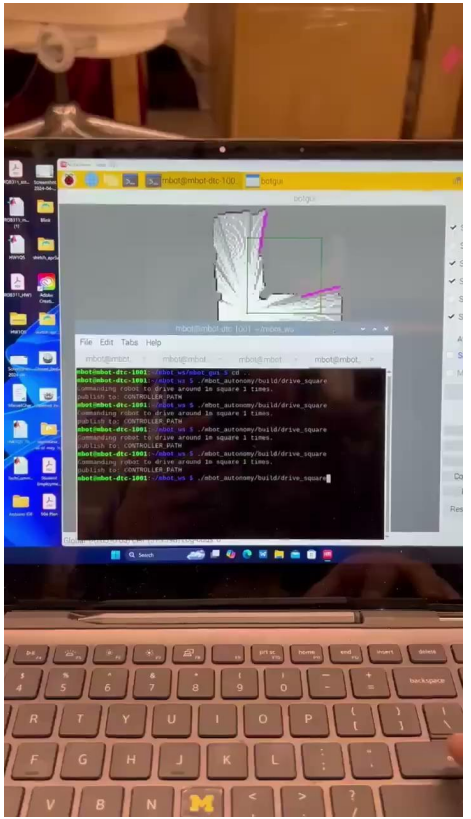
ROB 330 (SLAM and Navigation) Final Project

The goal of this project is to implement a SLAM algorithm on a two-wheeled robot and demonstrate autonomous mapping and exploration of unknown mazes.

I worked in a group to implement a PID controller for driving, a particle filter to fuse odometry and LiDAR data for position estimation, and a frontier-based exploration algorithm. As the robot drives and accumulates LiDAR scans, it builds an map segmenting its surroundings into a grid of occupancy probabilities. It also has a particle cloud about its estimated position, and propagates particle motion using its motor encoder data (+ Gaussian noise). It then weights particles by how closely their estimated position relative to obstacles match the latest LiDAR scan. By taking the weighted average of the best particles, it generates a more accurate understanding of its position.

The exploration algorithm chooses the next location to explore by identifying clusters of “free” cells adjacent to unknown cells (frontiers) and employs A* with associated costs to penalize long paths and encourage reachability.

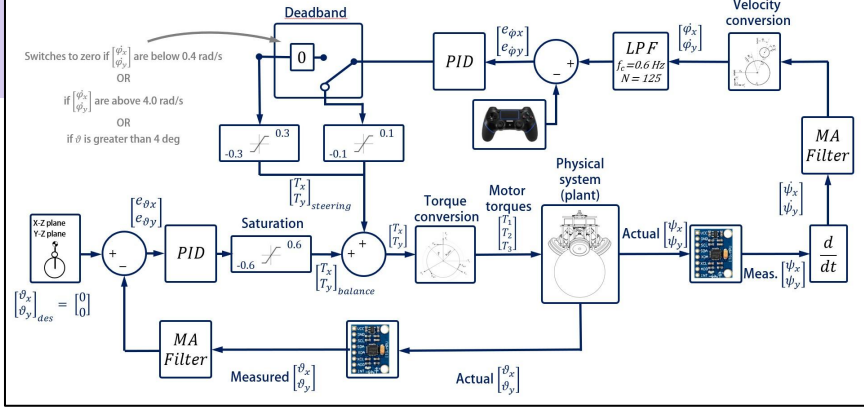


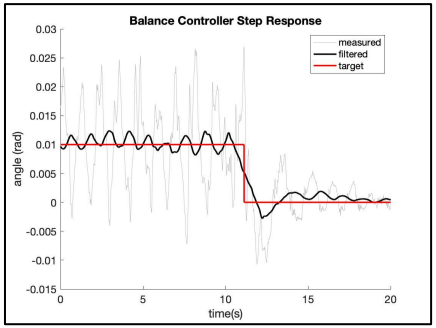


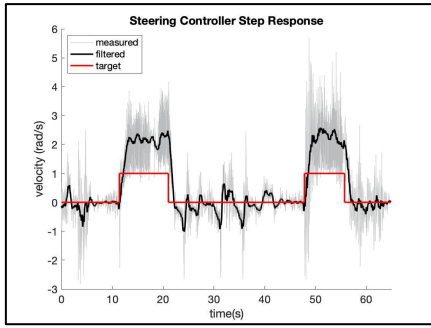
Demonstration of autonomous navigation and mapping of unknown maze

Ball-Bot (ROB 311 Final Project)

In ROB 311 I learned how to characterize a complex system's dynamics and develop a custom PID controller for it, which culminated in applying these concepts to building an omnidirectional ball-bot. The ball-bot is an underactuated system which is naturally unstable, making it a very difficult controls problem. I started with a planar model of the robot, governing equations of brushed DC motors, and velocity and acceleration specs. From there, I determined conservative estimates for the torque required to balance, turn, and drive the robot.







Ballbot balancing unassisted

I tuned the PID gains by first running balance tests. I used lean angle data from an MPU-6050 6-axis IMU, which was smoothed with a moving average filter. The balance controller pushed the robot's body angle towards 0, ensuring the motors would keep it upright as it continuously falls over under gravity. The motor commands from this controller were then weighted and added to a velocity controller, which took in wheel position and velocity data from the motor encoders, and calculated the ball's rotation and velocity. The velocity controller pushes these values towards the desired velocity from user (commanded from the PS4 console).

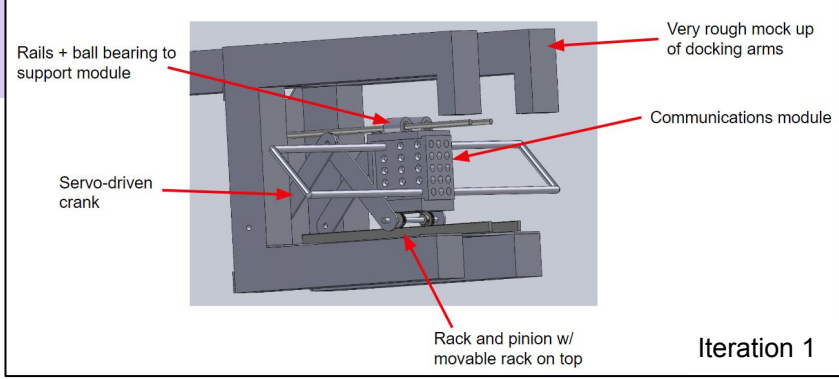
Hardware Projects

MAAV Payload Deployment Mechanism

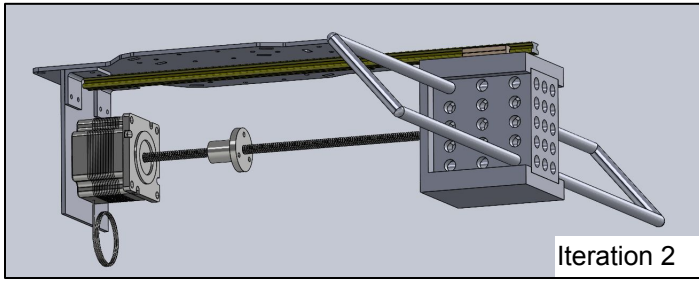
One of my first big projects on MAAV was designing the payload mechanism for our IARC Mission 9 submission. My task was to precisely place a 2 kg “communication module” after docking with a moving platform, minimizing weight and deployment time while maximizing accuracy and reliability.

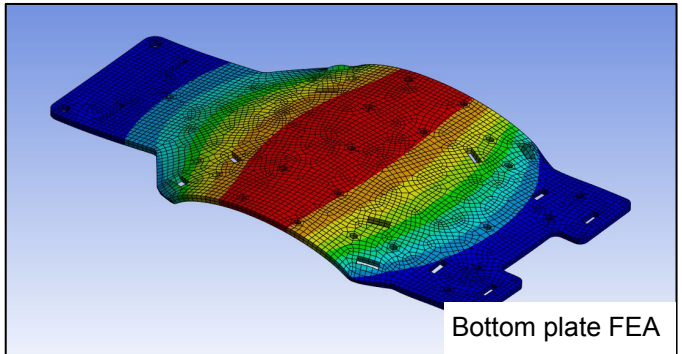
My first iteration tried to combine the compactness and low weight of a spring-loaded mechanism with the reliability and control of an off-the-shelf linear actuator. I designed a servo-driven crank which slid the payload along a pair of linear rails and a rack and pinion. This evolved into iteration 2, which was simplified to just a lead screw driven by a small DC motor, which would slide the payload along a rail, making it simpler, more reliable, and takes advantage of the payload’s threads.

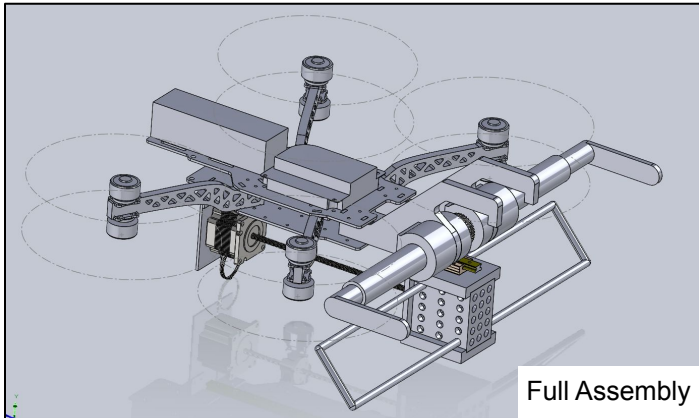
I validated my design by performing hand calcs and FEA using ANSYS. Shown to the right is the total deformation of the drone’s bottom plate under maximum load from the 4 motors and the payload fully extended.



Iteration 1

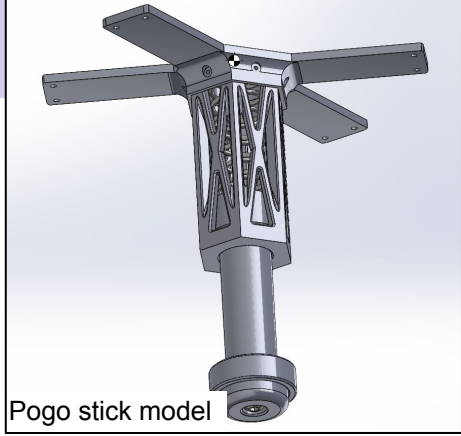

Iteration 2


Bottom plate FEA

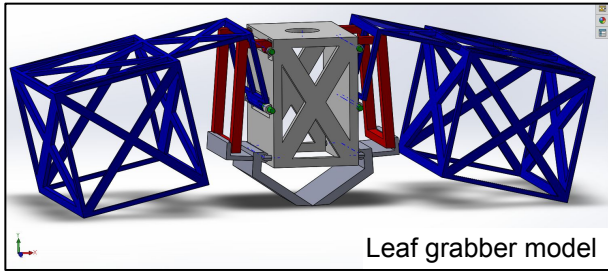

Full Assembly

Pogo Drone CAD and PID Tuning (A2Sys Research)

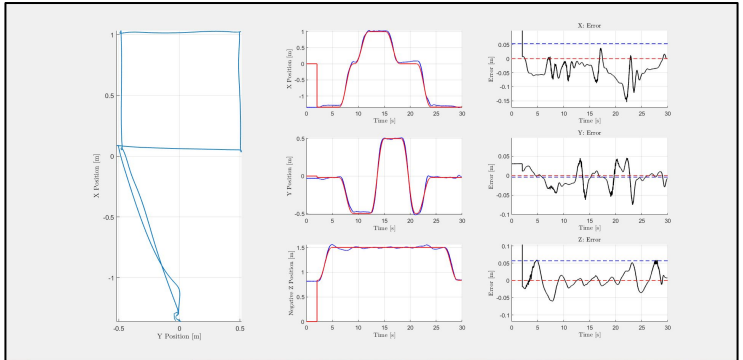
The goal of this project is to demonstrate a controlled, targeted bouncing technique for the application of collecting treetop samples in the Amazon rainforest. To enable these bouncing tests, I designed and fabricated a “pogo stick” attachment to be mounted on the drone’s bottom plate. I created a hexagonal cage with circular interior for easy mounting and added cutouts for weight saving. The cage and leg were resin printed, and the foot was resin printed in silicone. I validated this design by conducting flight tests and recording bounce-back heights for various inbound velocity and height setpoints.

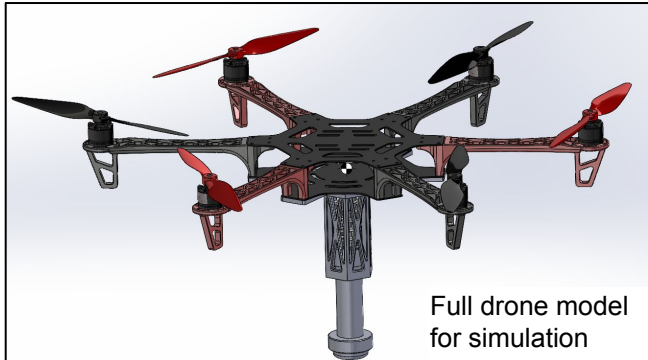

Pogo stick model

Next, I developed an end effector proposal which could be easily mounted in the same place as the pogo stick foot. The grabbing mechanism was triggered by a bounce, which would depress the triggers and release a torsion spring on each side, snapping the two halves shut. The two halves of the grabber are lined with razors for a clean cut from the branch, minimizing tearing of the leaf sample.


Leaf grabber model

I also conducted flight tests for cascaded PID controller tuning. I documented flight test procedure for safer and more consistent data collection, added flight state transition functionality, and improved waypoint fidelity. The xyz graphs are shown to the right.

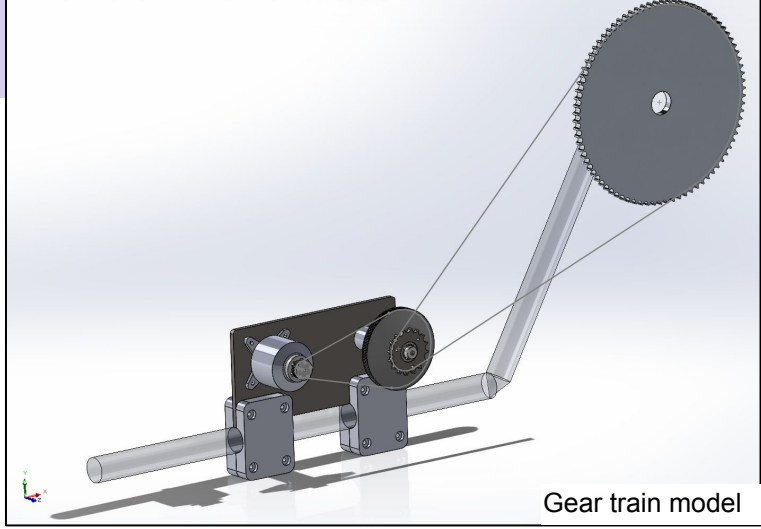



Full drone model for simulation

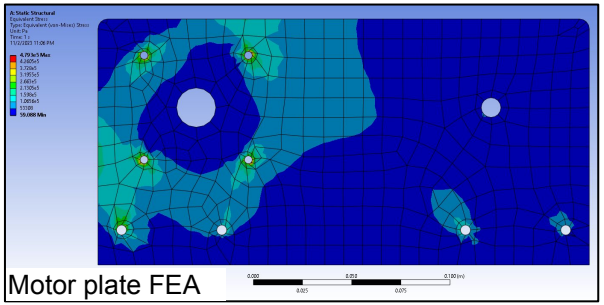
Electric Bike Conversion

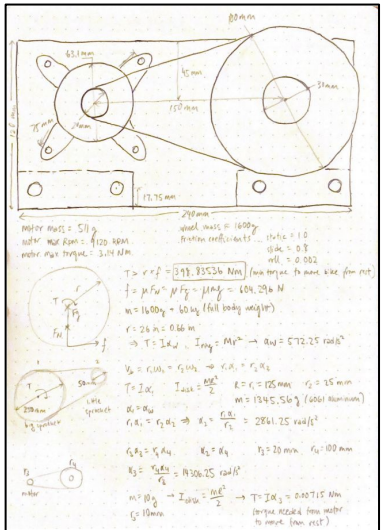
This is a personal project I started to challenge myself. My goal was to convert an old mountain bike into an electric bike I could use to get around campus, with as low a budget as possible. I modeled this project after this Tom Stanton YouTube video because I had access to several old drone motors from MAAV.

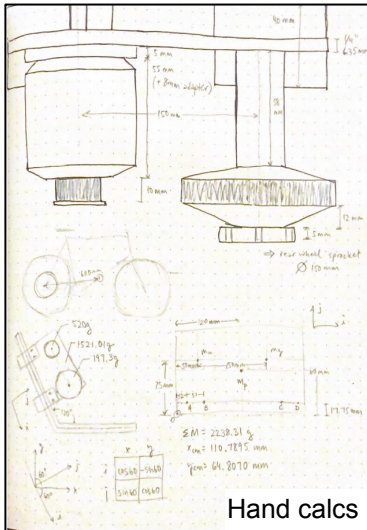
I started by calculating the gear ratio I would need to achieve the same back wheel RPM as Tom Stanton, which was 25:1. To check that the motor I had would work, I did a rudimentary torque calculation to find the upper bound of the torque the motor needed to supply to move the back wheel from rest. I then started modeling a two-stage gear train composed of two 5:1 gearsets.


Gear train model

To minimize cost, I planned to 3D print the first gear set and mill the second. I also wanted to mill the motor mounting plate. I did FEA for the plate, and chose to mill it from steel based on the results.


Motor plate FEA



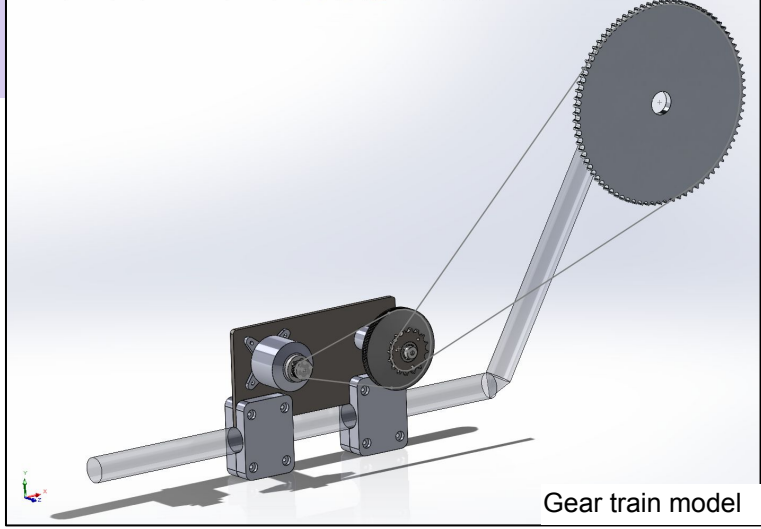

Hand calcs

At this point in the project, I began selecting ESCs, batteries, and other hardware. However, the budget had started adding up, and so I enlisted the help of my former professor for advice and support. With his help, I am now developing this project to become apart of a new robotics course to teach students about system design, motor spec'ing, and power principles.

Amazon Robotics Conveyor Test Stand

This is a personal project I started to challenge myself. My goal was to convert an old mountain bike into an electric bike I could use to get around campus, with as low a budget as possible. I modeled this project after this Tom Stanton YouTube video because I had access to several old drone motors from MAAV.

I started by calculating the gear ratio I would need to achieve the same back wheel RPM as Tom Stanton, which was 25:1. To check that the motor I had would work, I did a rudimentary torque calculation to find the upper bound of the torque the motor needed to supply to move the back wheel from rest. I then started modeling a two-stage gear train composed of two 5:1 gearsets.



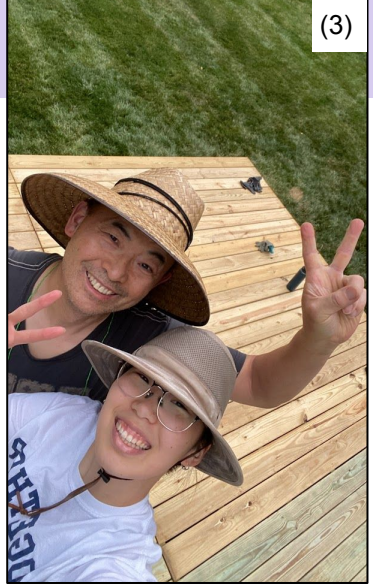
Gear train model

Bonus: Floating Deck

My mom has wanted a deck ever since we moved into our house over 12 years ago. My dad wanted to build it himself, but hasn't had time between work and just life in general. For my sister's high school graduation party, I decided to help him finally make this long-standing dream a reality.

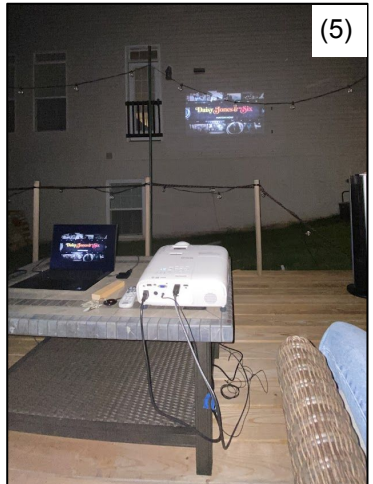
With a week to design and build this floating deck, my dad and I began this project by reviewing our town's building codes, which we used to determine a 8x12 ft^2 deck would be the most cost-efficient, serviceable, and legal. We mapped out our design in Lowe's Deck Designer, watched some Youtube videos, and bought our materials.

We spent 3-5 hours every evening for the next week building. First, we marked the corners of the deck to bury posts. Because the yard is inclined, we used a laser ensure it would be level. We installed the metal brackets and joist hangers, then placed the frame to double check the fit. Once fit was verified, we installed the frame and joists (1). We continuously checked that the joists were level as we went (2). Then, we began installing the decking boards (3). We barely finished in time, as we installed the railing a mere hour before my sister's guests arrived. Still, the party was a success (4), and the deck became a great place to watch movies in cool summer nights (5).


(3)


(1)


(2)


(5)


(4)