UNCA IEEE Senior Design Project

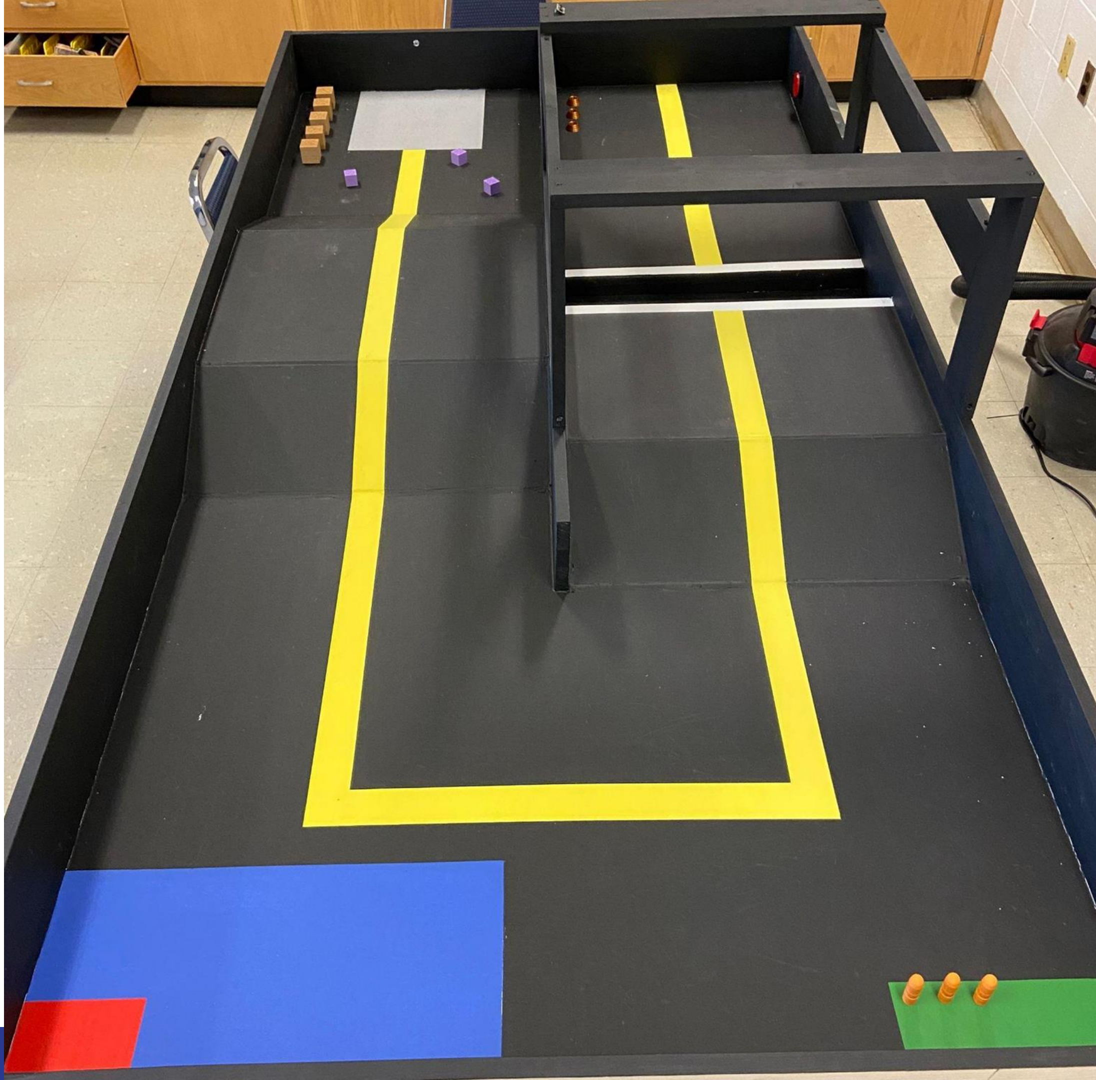
Carlos Anzola
Chris Bass
Alex Haines
Zachary Price
Jason Robinson
Maxwell Turcios

Project Overview

- Competed in the 2024 IEEE Southeastern Conference Hardware Competition
  - Institute of Electrical and Electronics Engineers
  - Build an autonomous robot that earns points for completing tasks
- Every southeastern university with an engineering program can compete

This is the playing field where our robot runs

Total points possible: 120 pts

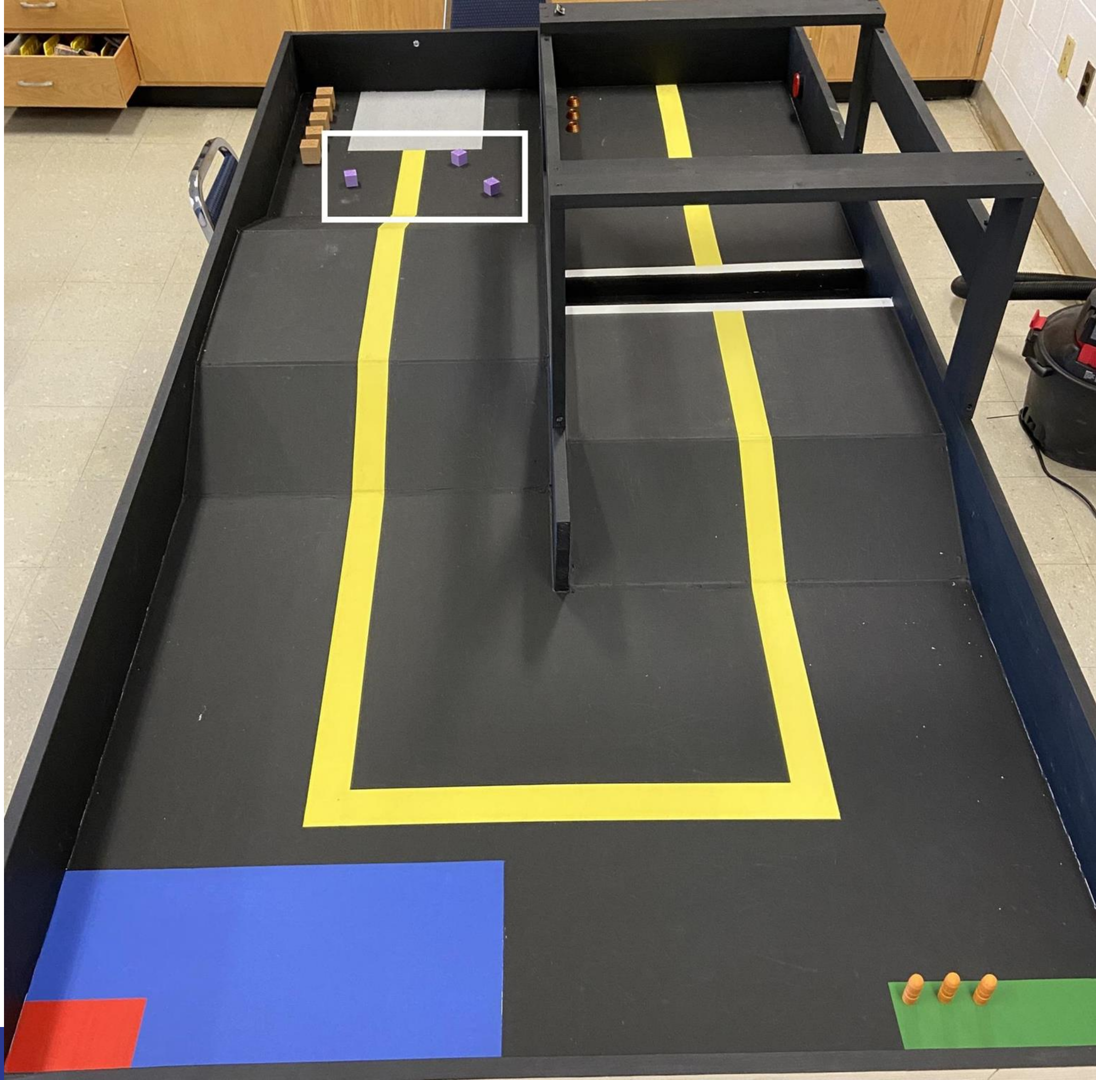


Task 1:
Move (5 pts)

Additional points if we start at the signalling of a green LED (5 pts)

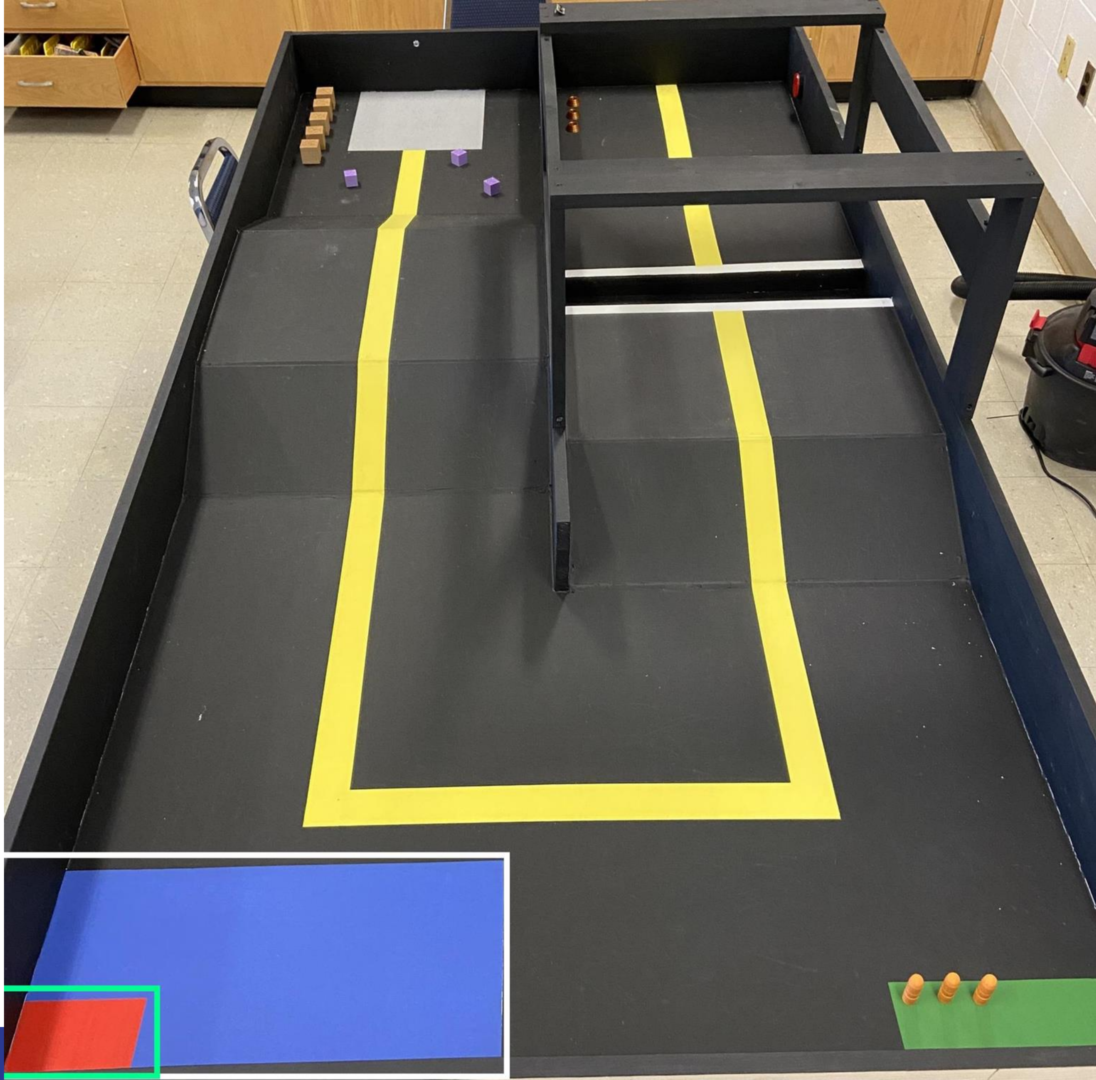
Task 2:
Collect 5 big, brown blocks

Task 3:
Collect 3 small, purple blocks

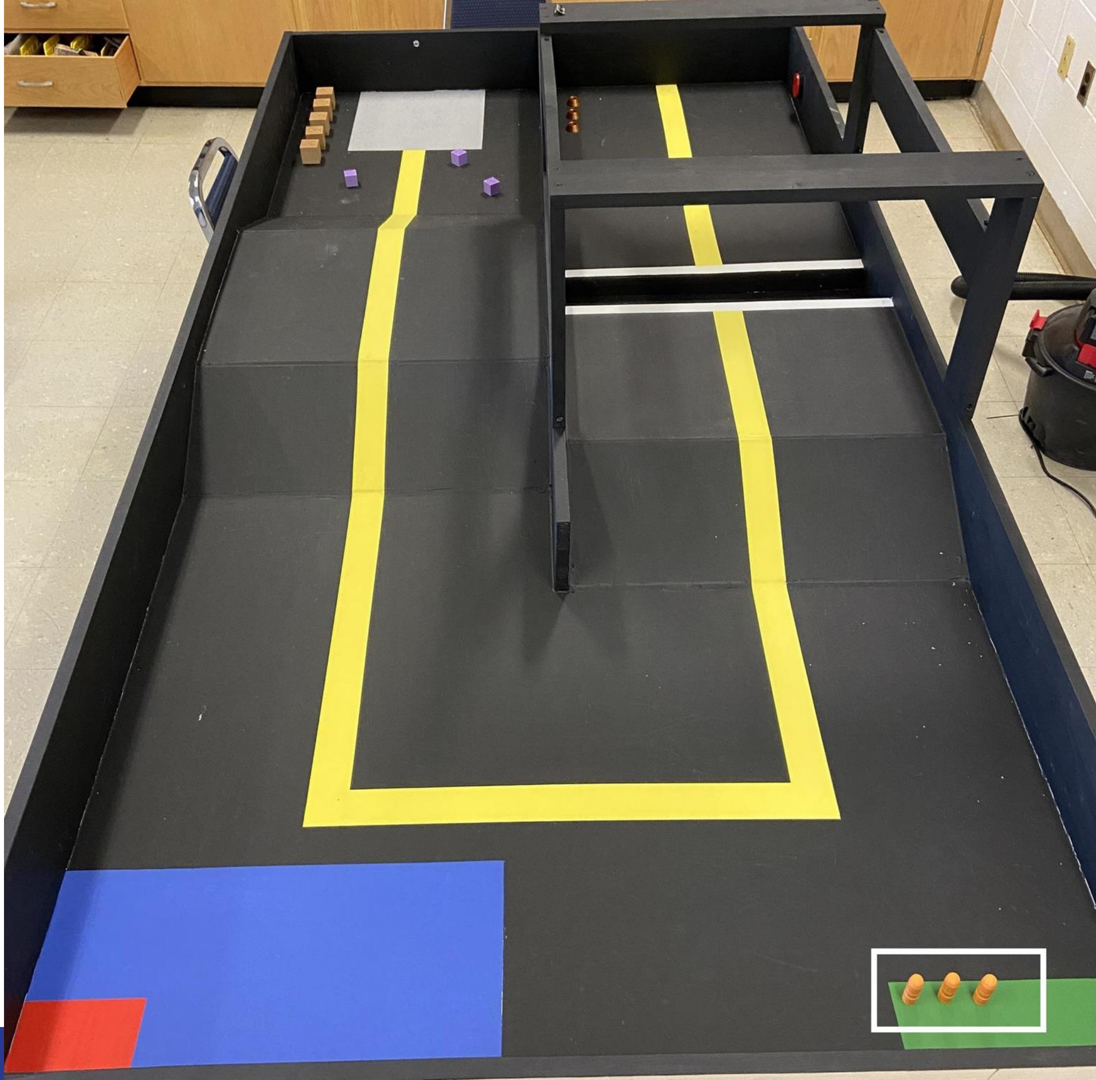


Task 4 & 5:
Place all big blocks in blue area
(15 pts)

Place all small blocks in red area
(15 pts)

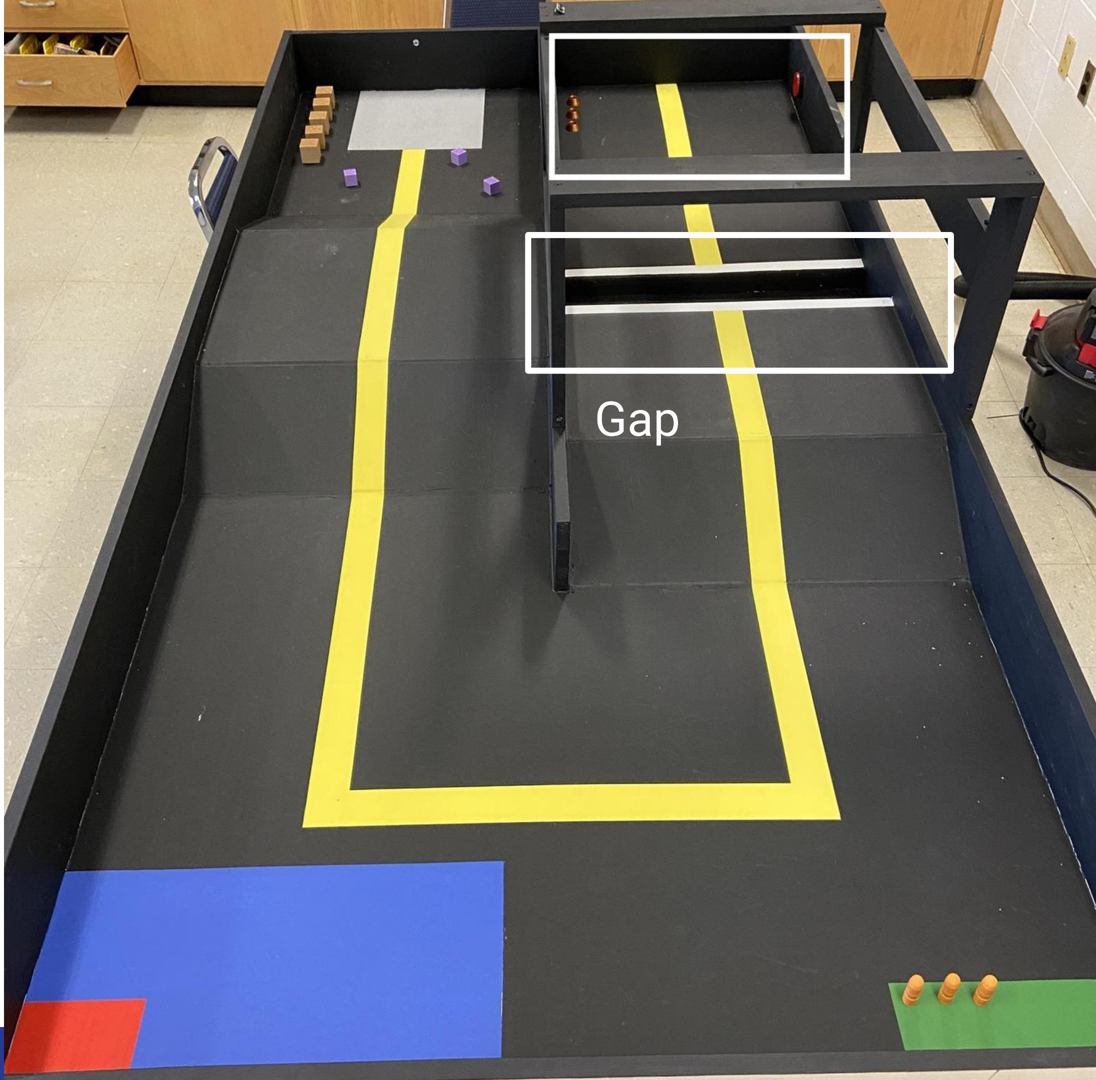


Task 6:
Navigate to green area and collect orange cylinders



Task 7 & 8:
Reach final area (20 pts)
Doesn't matter whether we cross the gap or find a different way

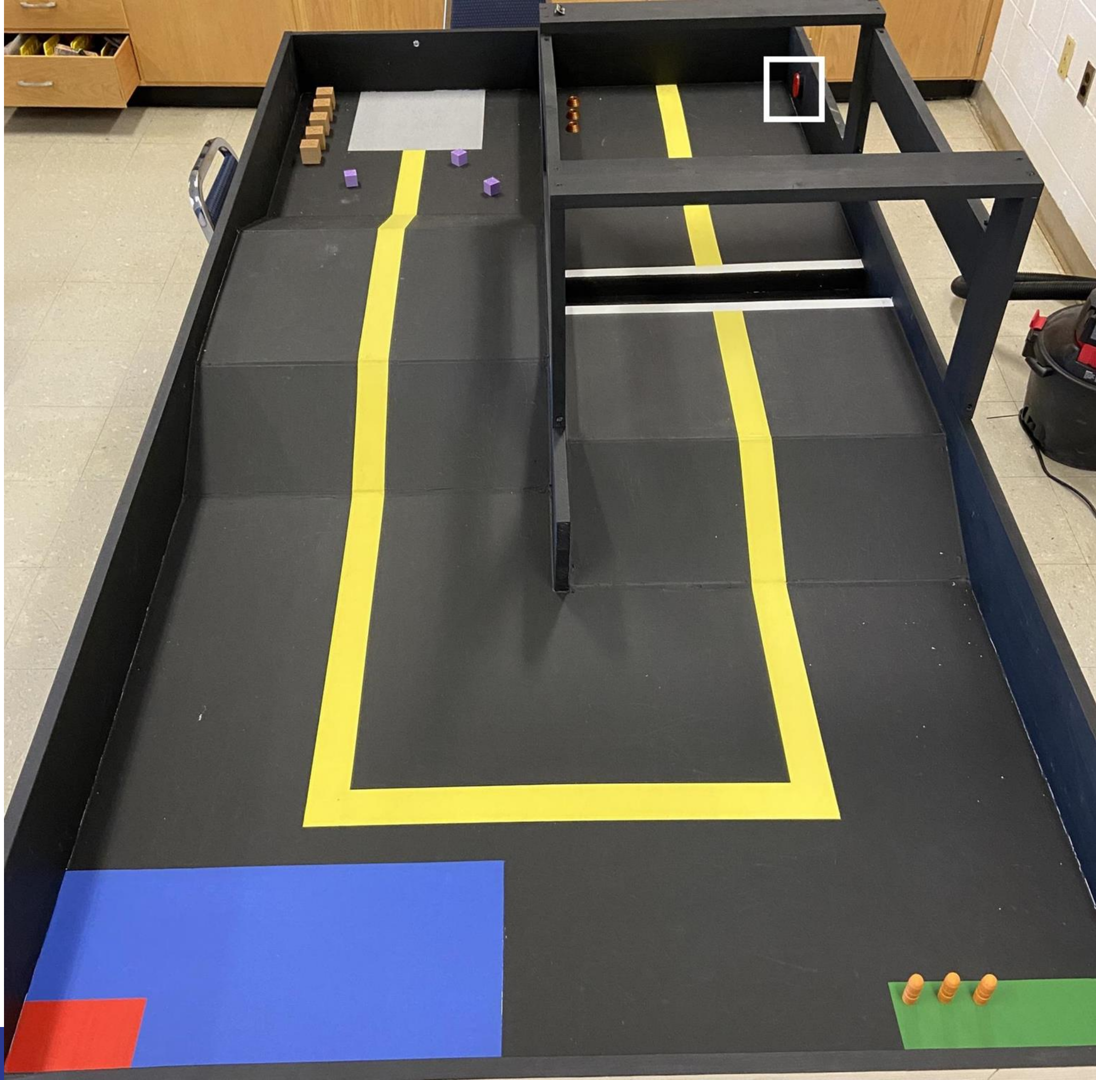
Display school pride while in final area (10 pts)



Task 9:
Place cylinders in bronze rings
(20 pts)

Task 10:
Press red button to end the run in under 1 minute 45 seconds
(30 pts)

Or let the clock run to 3 minutes (0 pts) at which the judges stop the run



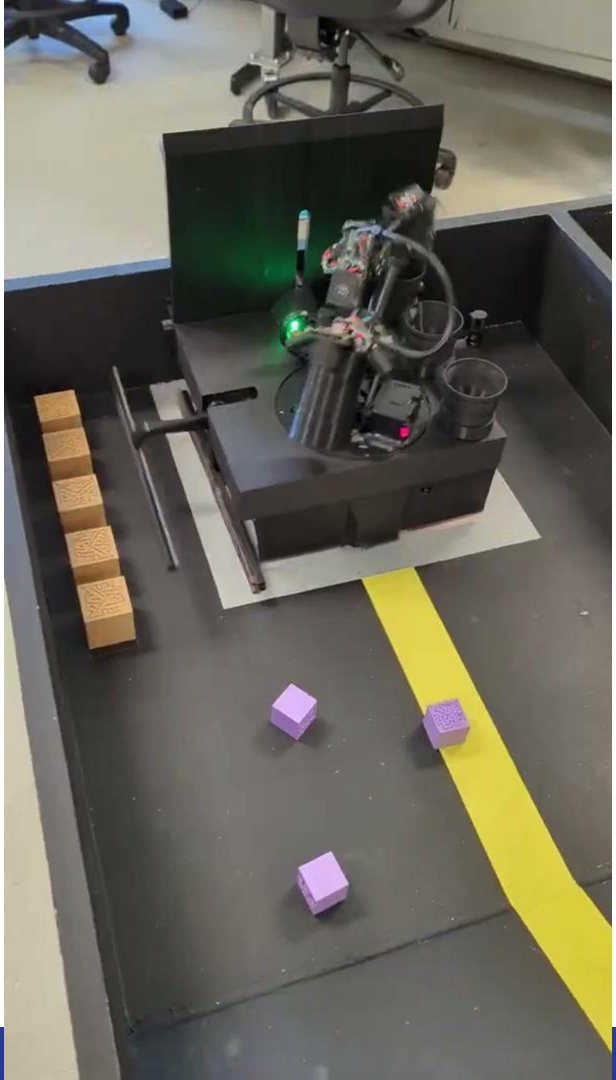
Rules and Limitations

- Must run autonomously
- Must fit in 1ft x 1ft x 1ft box
- Must weigh less than 25 pounds
- Must not pose a threat or damage the playing field

Why Compete?

- Maintain IEEE relations
- Demonstrate engineering proficiency
- Increase visibility of the UNCA/NCSU Joint Mechatronics Program
- Job opportunities

Video of Robot



Team Goals For Robot

- Wait until the green LED turns on then start moving (5 pts +5 pts)
- Place every small block in the red area (15 pts)
- Place every big block in the blue area (15pts)
- Reach final area then display school spirit (20 pts +10 pts)
- Place every cylinder in the rings (20 pts)
- Press the button in the final area in under 1 minute 45 seconds (30 pts)

FULL POINTS (120 pts), LOWEST TIME!

Project Approach

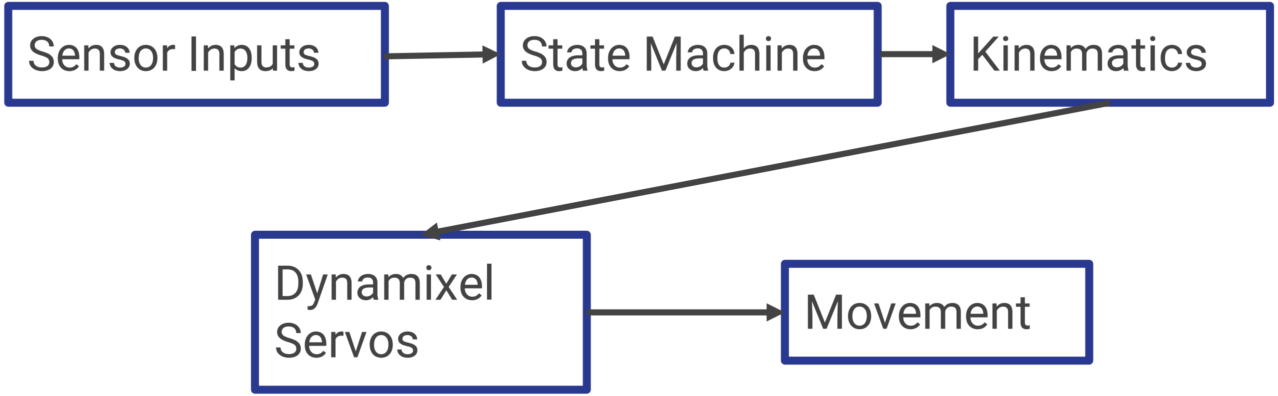
- Developed a Minimum Viable Product (MVP)
  - Start fulfilling minimum requirements then improve
- Divided ourselves into subteams
  - Mechanical
  - Electrical
  - State machine
  - Robotic arm and bulk manipulation
  - Computer vision

Programming Methods

- Kinematics (Robot local velocity to wheel velocities)
- GitHub (Version Control)
- Modular code
  - Reuse multiple functions and sections of code
- Control Algorithm
- ROS (Robotic Operating System)
  - Uses Nodes (code that executes a program) to communicate with other systems.

Control Algorithm

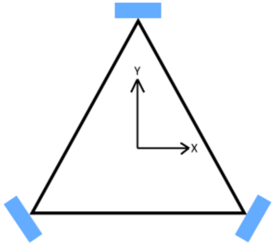
- Uses sensor inputs from Time of flight sensors and Inertial measurement unit to get real world data
- Uses inputs from sensors to change state in state machine
- Dynamixel Servos have built in controller to handle error and accuracy

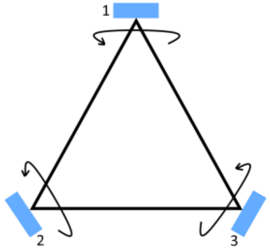


Inverse Kinematics

- The inverse kinematics convert robot local velocity to individual wheel velocities

Robot Local Velocity → Inverse Kinematics → Wheel One Velocity / Wheel Two Velocity / Wheel Three Velocity

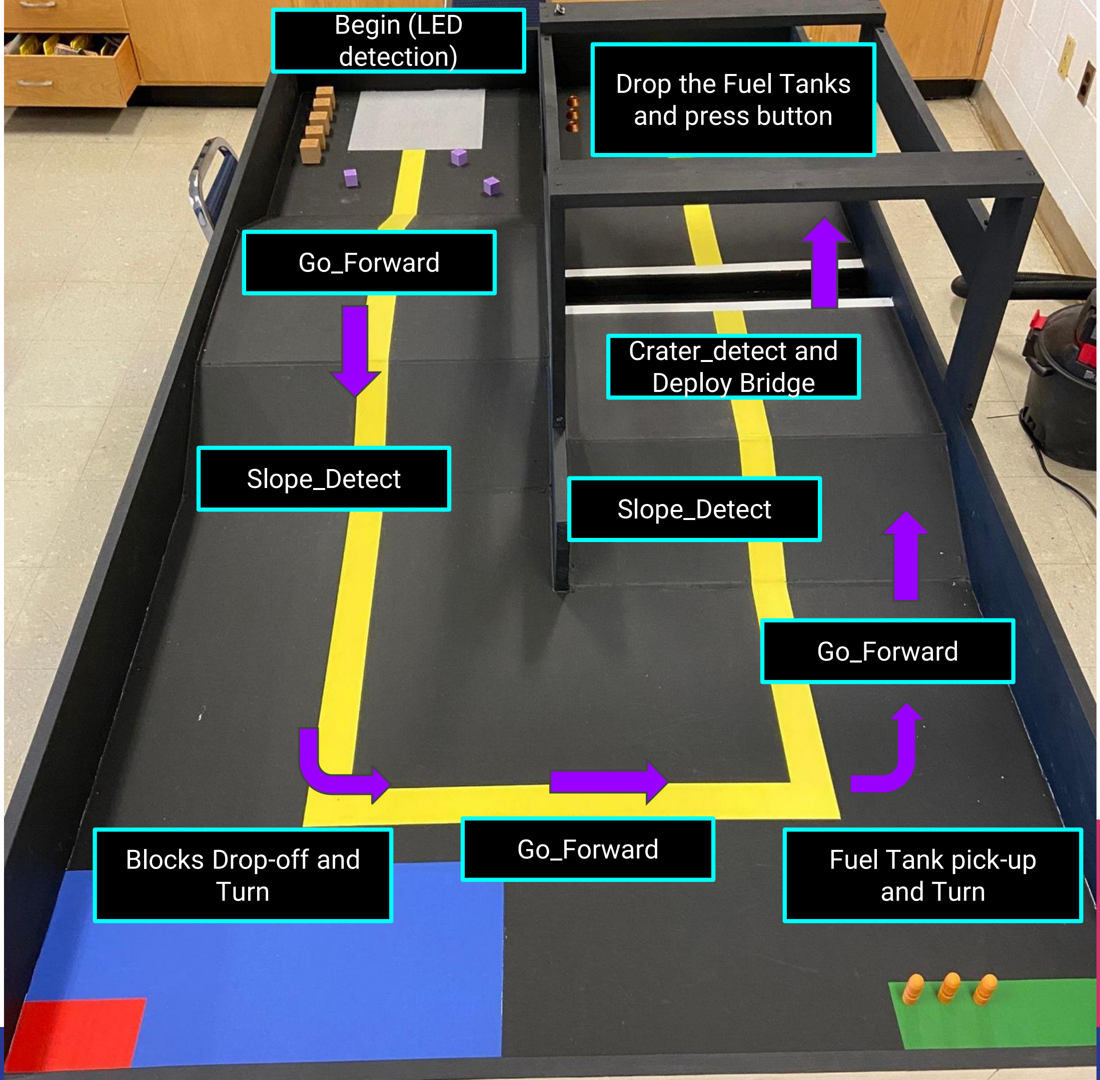




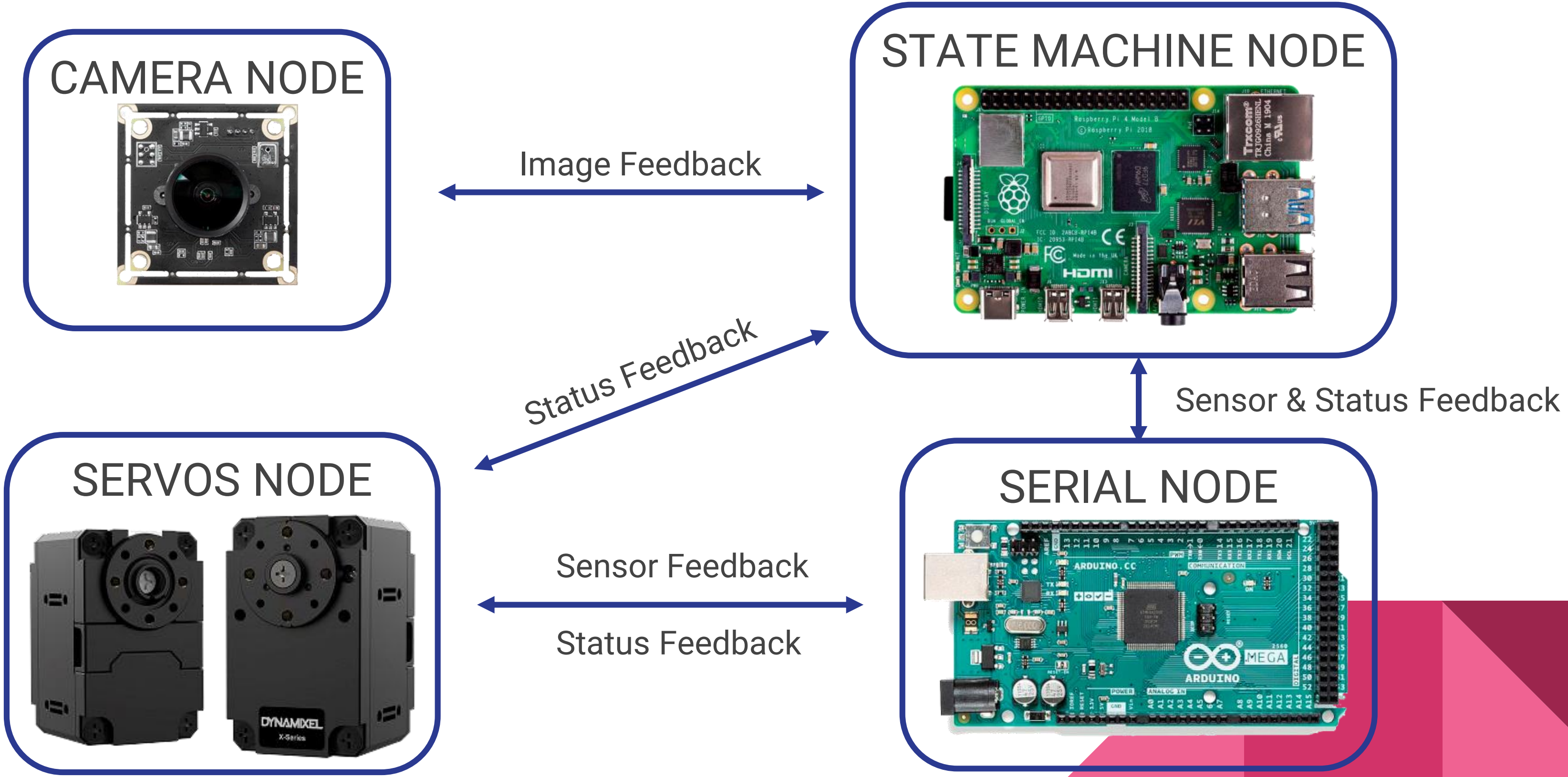
State Machine

Overall State Machine in Python

- The state machine is a behavioral model that changes its status based on inputs.
- Uses Robotic Operating System (ROS) to communicate with Servos and Sensor Nodes.
- The structure controls the instructions given to the robot.


Begin (LED detection)
Drop the Fuel Tanks and press button
Go_Forward
Crater_detect and Deploy Bridge
Slope_Detect
Slope_Detect
Go_Forward
Blocks Drop-off and Turn
Go_Forward
Fuel Tank pick-up and Turn

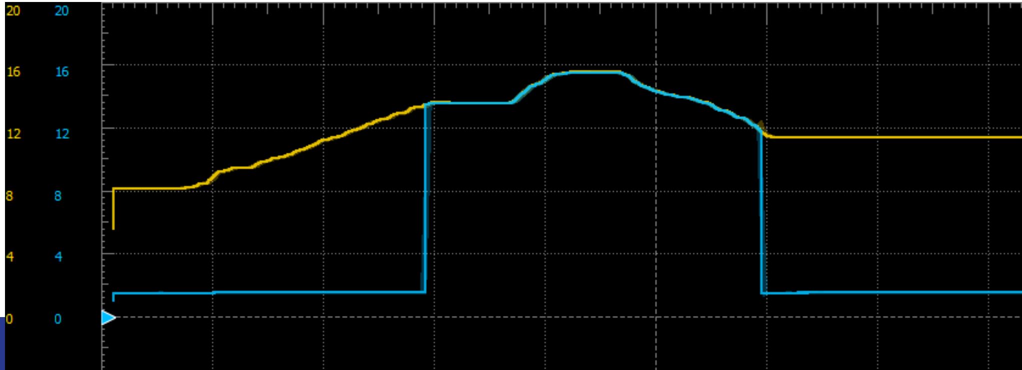
ROS Architecture



Electrical Systems - Power Supply

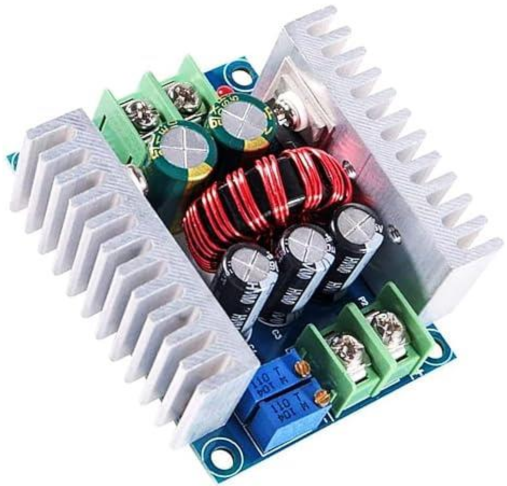
- Lithium Ion battery (4 cells) (16.4v to 12.8v)
  - Low weight for high energy capacity
- Batteries become irreparably damaged if voltage gets too low
- Systems can be damaged if current gets too high
- Power management circuit disconnects power when needed

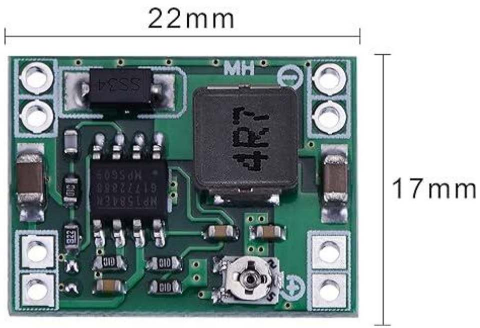
Yellow: input
Blue: output



Electrical Systems - Power Regulation

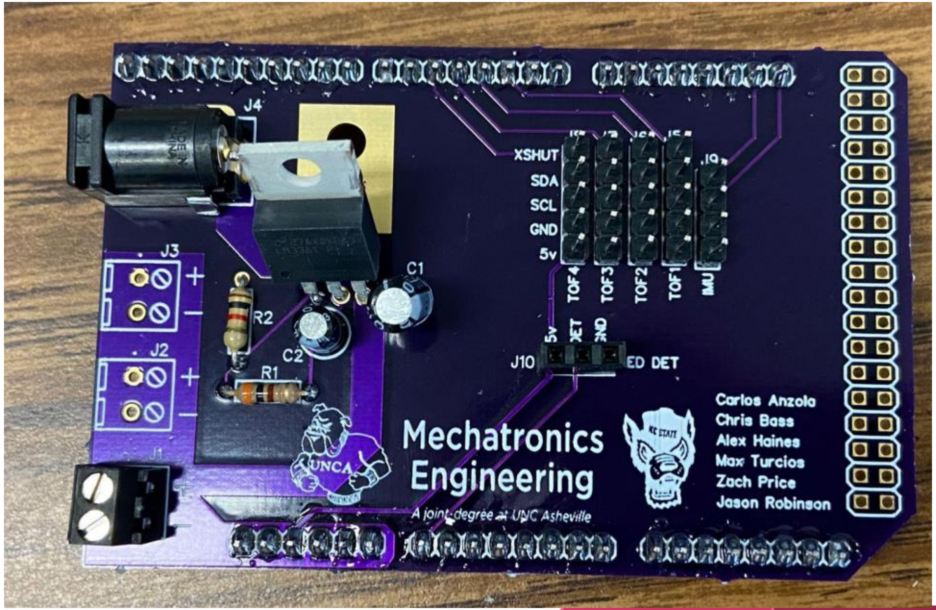
- Not every subsystem has same power requirements
- Battery voltage is from 16.4v to 12.8v
- Subsystems use either 12v or 5v

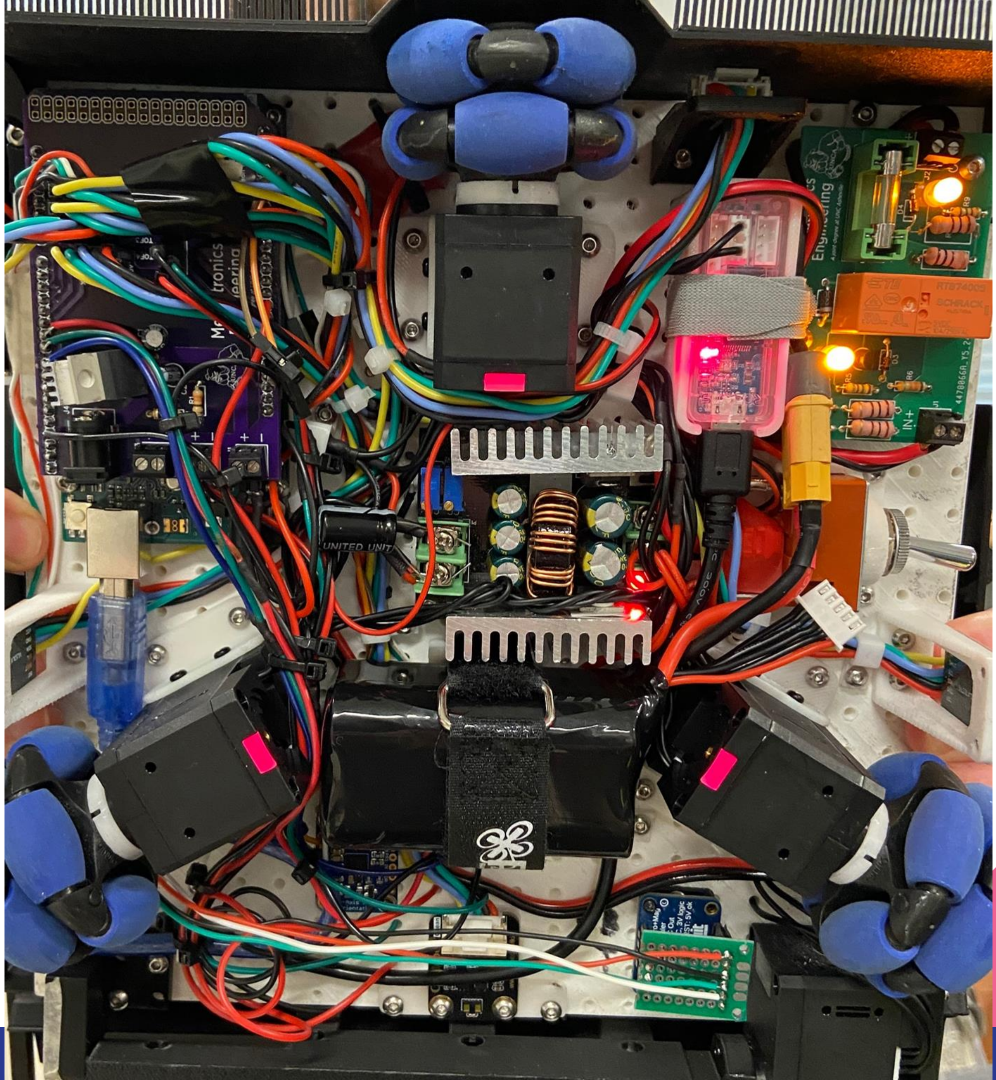




Electrical Systems - Sensor Node

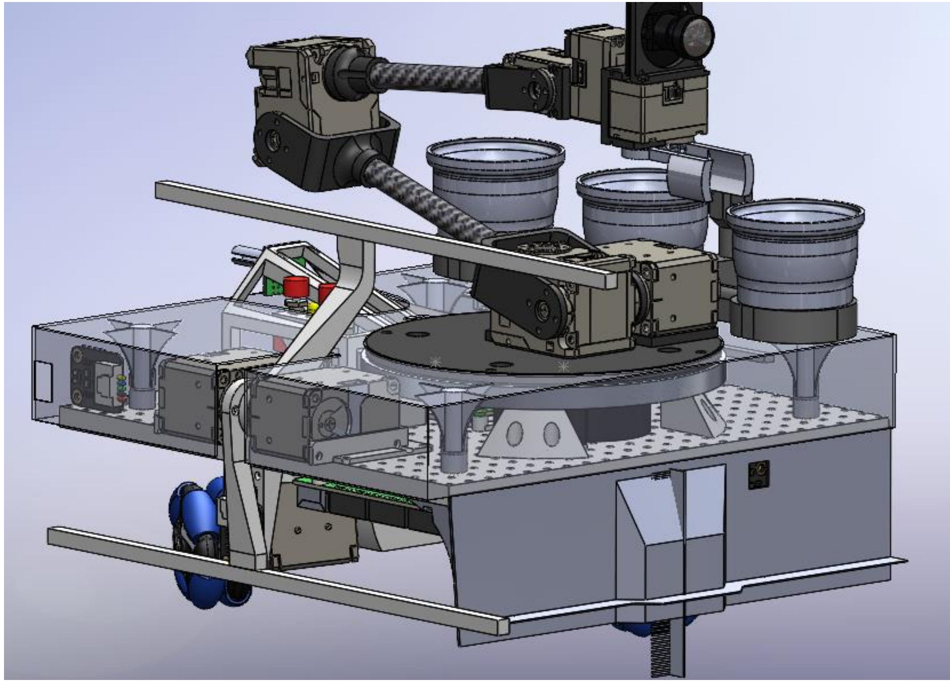
- Arduino Mega interfaces with sensors to collect data on surroundings
- I2C communication is used to simplify sensor communication
- Additional power regulation for external systems





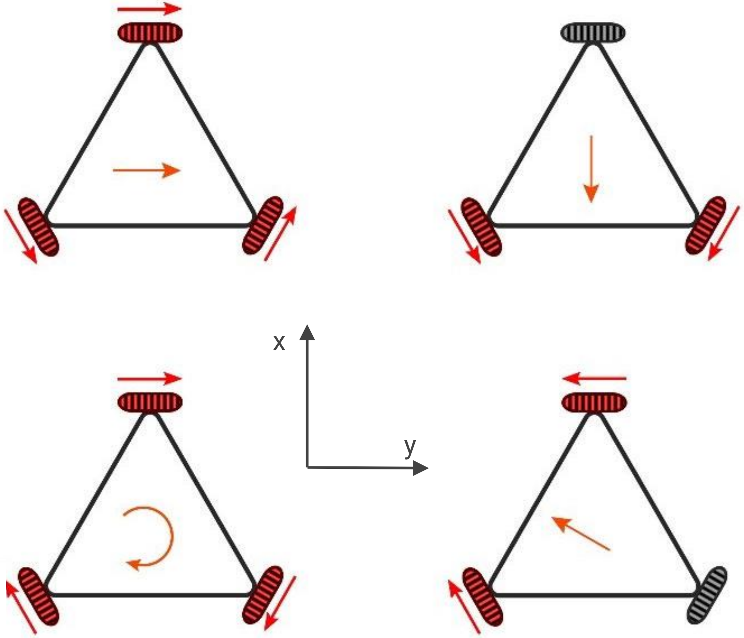
Mechanical Methods

- Modular chassis
  - Square, 3D printed
  - M3 holes, 10mm grid
- Fast prototyping
  - New bracket not new frame
  - Position, derive, print, and assemble in less than 30 min
  - No need for duct tape or hot glue
- Modular design for easy changes



Mechanical Methods

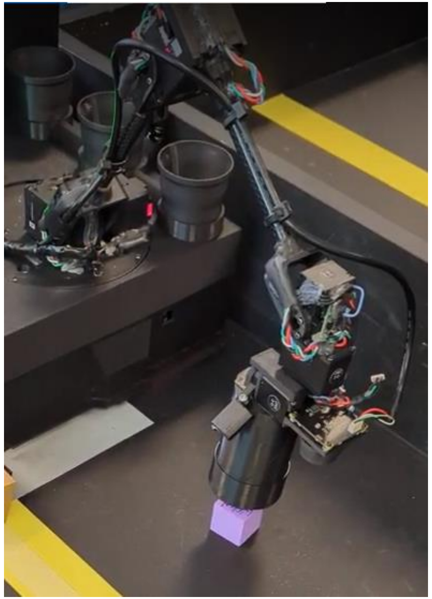
- Three Omni-wheels
  - Normal rolling wheel
  - Blue rollers spin freely
  - Free perpendicular movement
- Only three points of contact for reliability in uneven terrain
- Full 3 degrees of freedom
  - X velocity
  - Y velocity
  - Z rotation

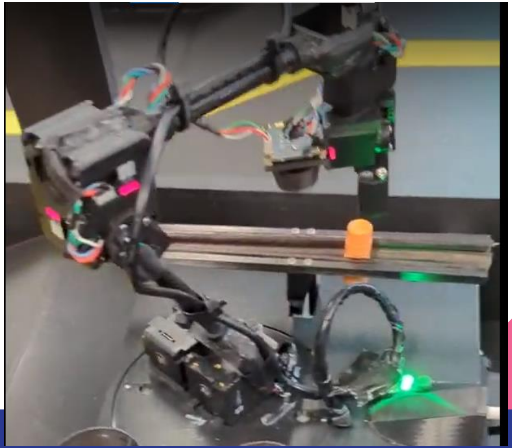


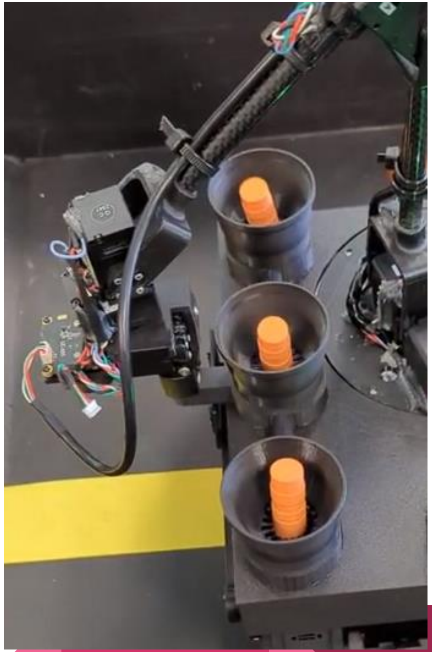
Not for sandy martian terrain

Robotic Arm

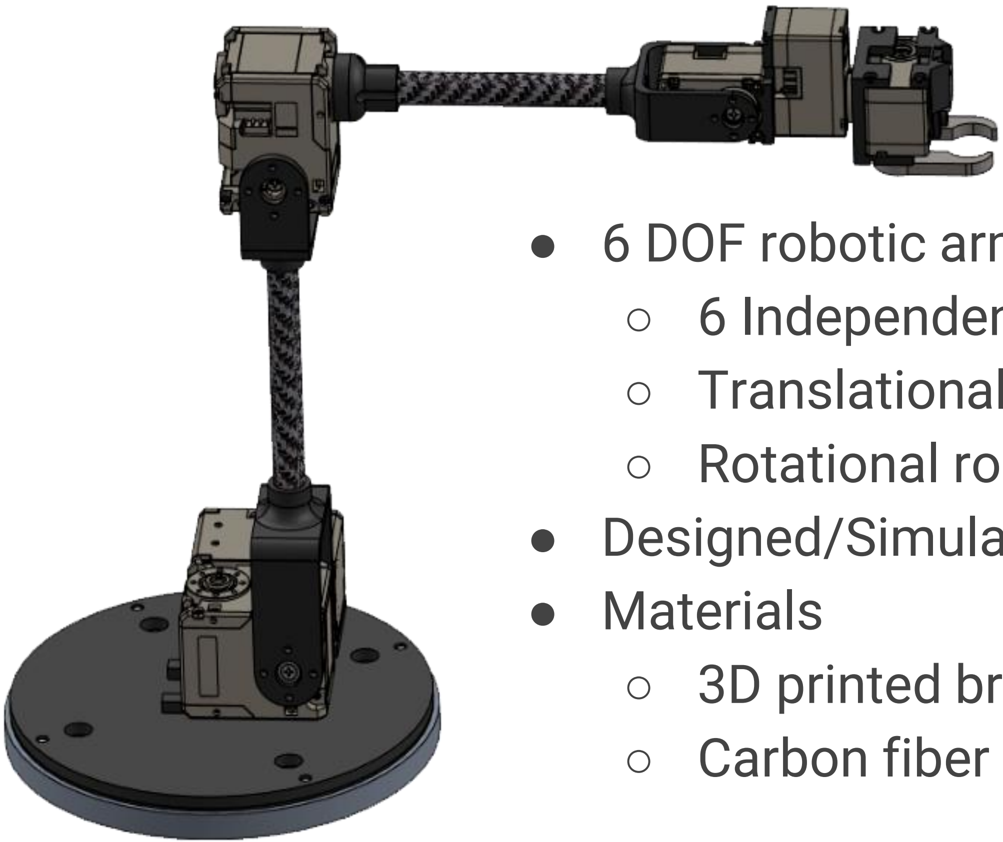
- 6 DOF (Degrees of Freedom)
- Custom Designed and built
- Using computer vision to
    - Pickup small boxes
    - Pick/place cylinders



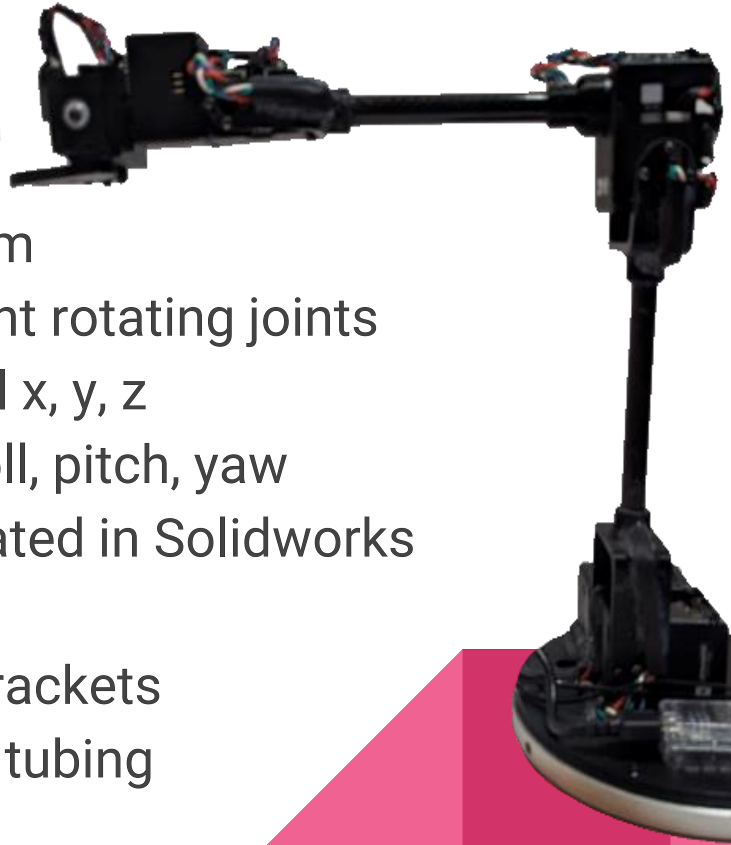




Robotic Arm

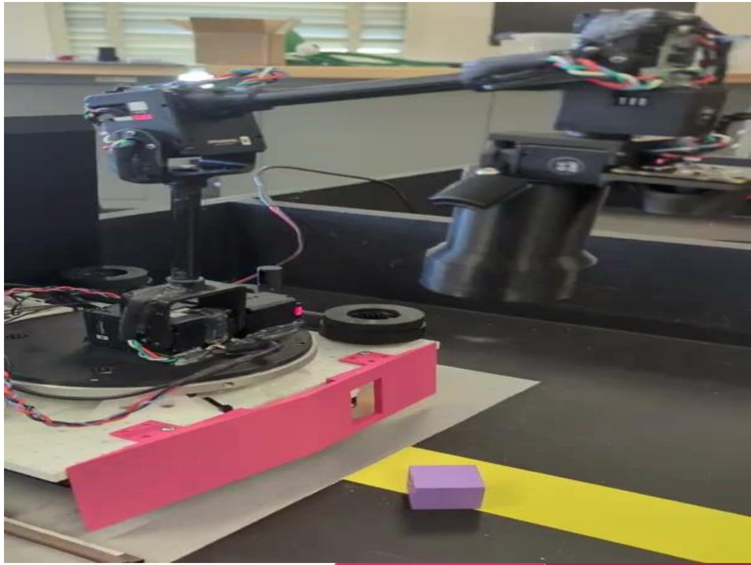


- 6 DOF robotic arm
  - 6 Independent rotating joints
  - Translational x, y, z
  - Rotational roll, pitch, yaw
- Designed/Simulated in Solidworks
- Materials
  - 3D printed brackets
  - Carbon fiber tubing



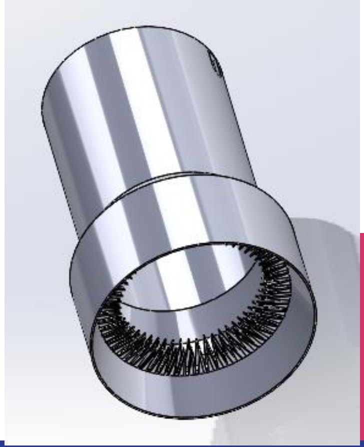
Robotic Arm

- 6 Dynamixel servos
  - 12 Bit absolute encoder
  - <0.1 Deg. of accuracy
  - Individually tuned controllers
- Forward/Inverse kinematics
  - Find xyz from joint angles or vice versa
- Implemented in ROS
  - Controlled with coordinates or angles
  - Control speed and accuracy
  - Linear interpolation



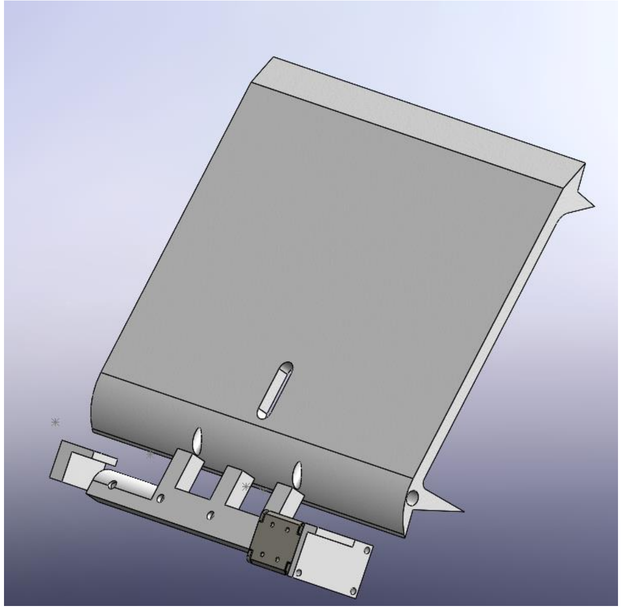
Other Mechanical systems

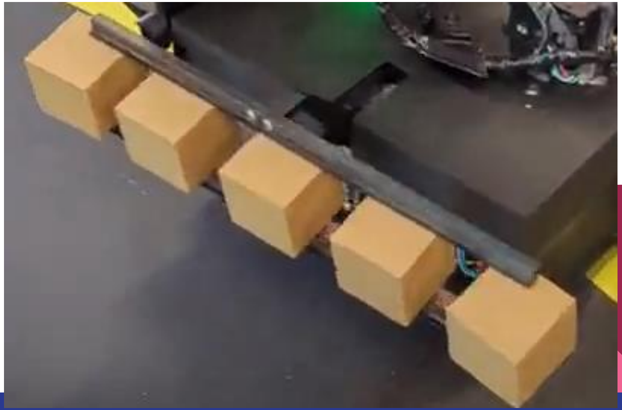
- Jaws attachment
  - Clips to robotic arm for start
  - Uses 3D printed bristles to allow items to push items in without falling out
  - Holds all three small blocks
- Cylinder holder
  - Funnels to precise location
  - Hold all cylinders for placement at once
  - Uses the 3D printed bristles



Other Mechanical systems

- Bridge
  - Reliable system for crossing gap
  - Made light weight as to not tip over
- Bulk grabber
  - Used to grab big boxes and cylinders
  - Uses silicone to conform and grip

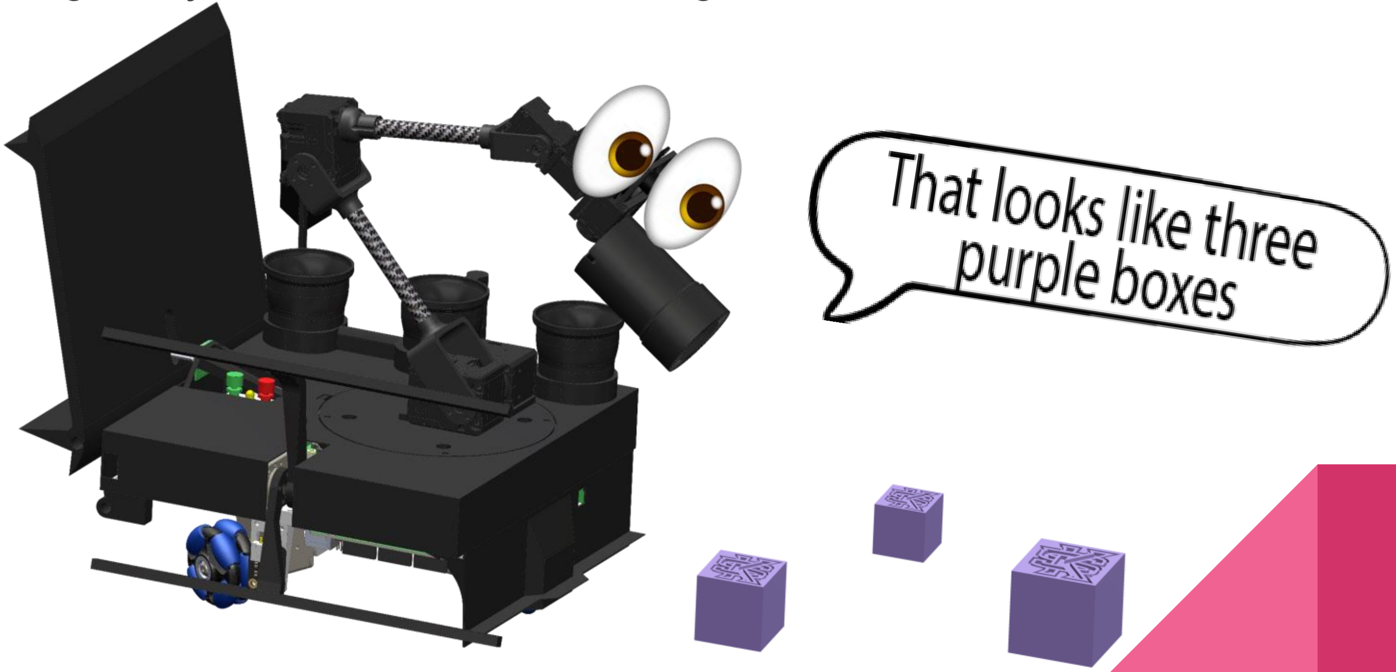




Computer Vision

What is the purpose of computer vision?

- To give eyes to the robot, allowing it to see and understand the visual world.



Why a Computer Vision system?

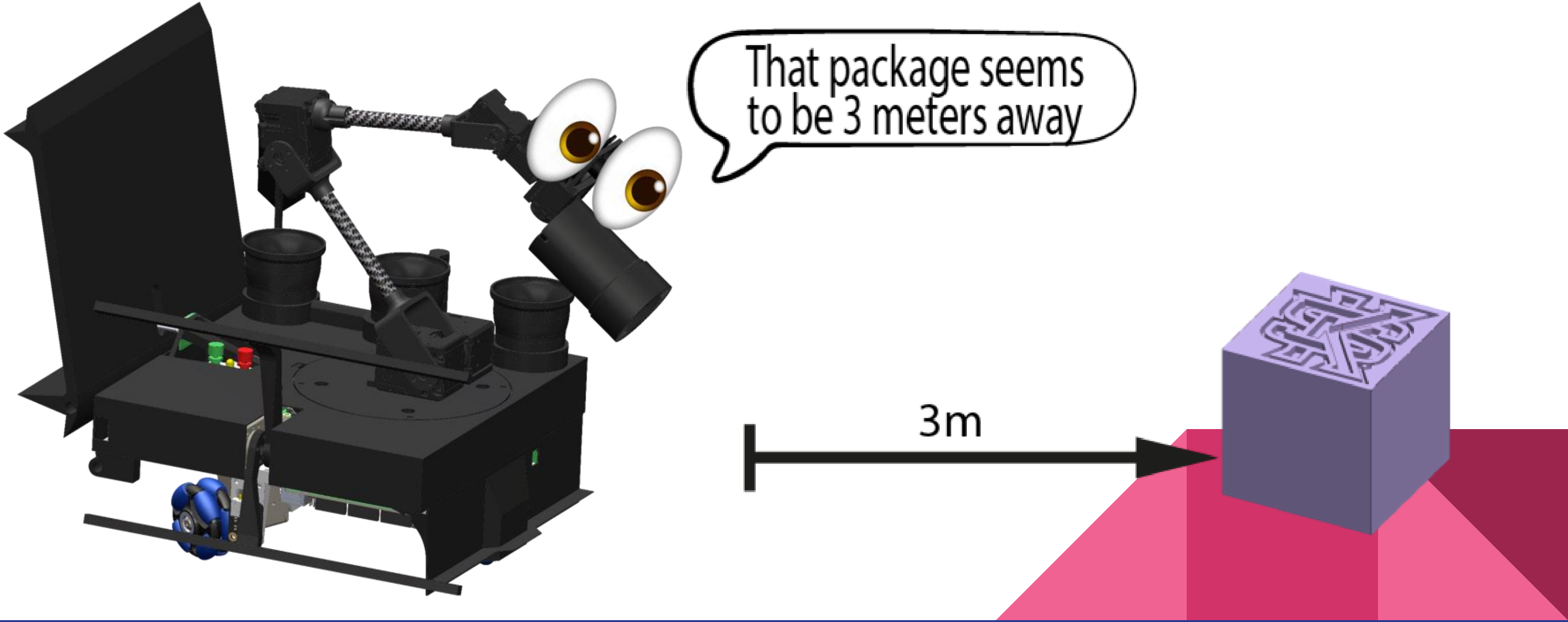
- The robot must be able to identify different types of objects in its surroundings.

Why a Computer Vision system?

- The robot must be able to identify small blocks and fuel tanks.

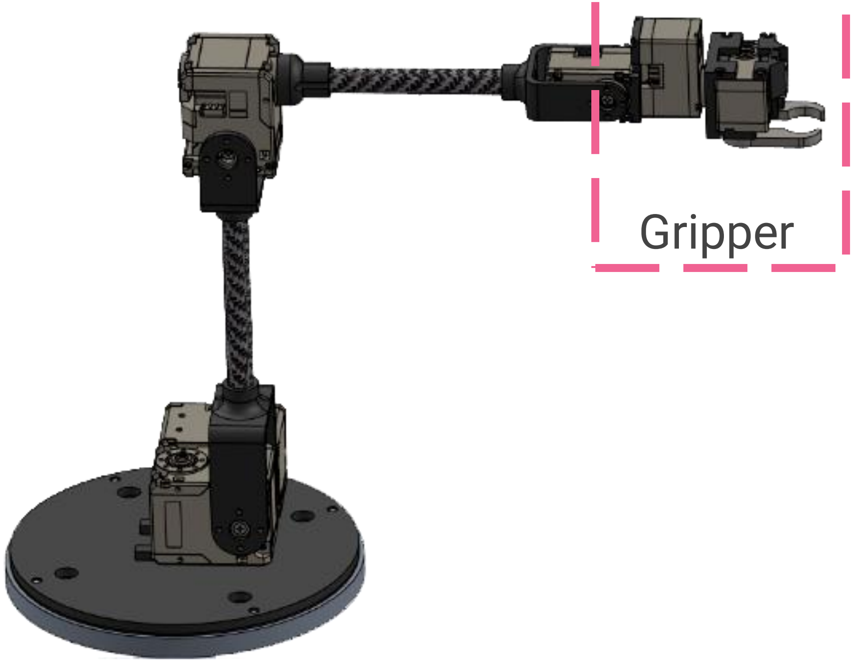
Why a Computer Vision system?

- The robot must have spatial awareness to tell where items are located.



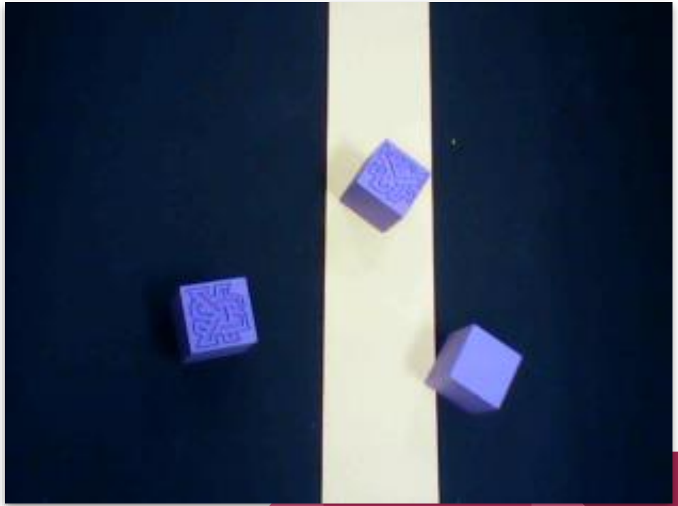
Vision System Process: Image Acquisition

- The robot uses a camera located in the gripper of the arm to take pictures of its surroundings.



Gripper

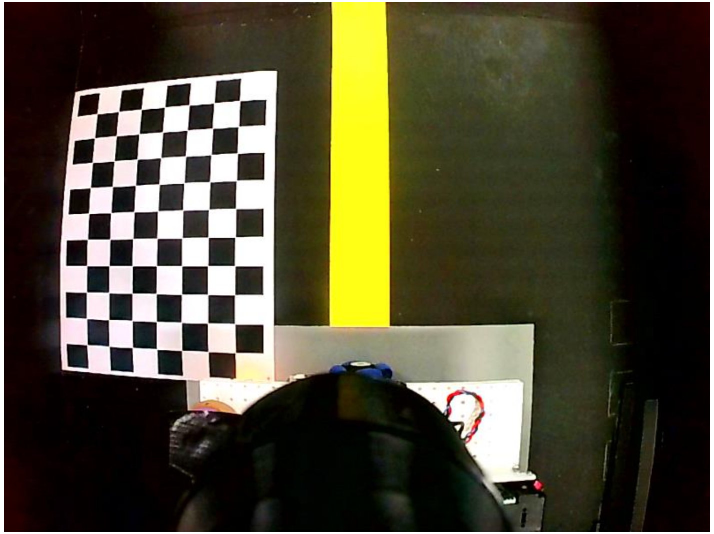
Camera



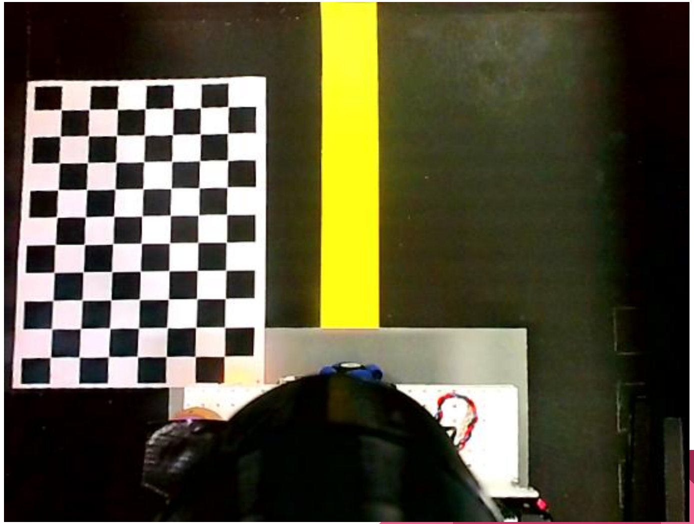
Output Image

Vision System Process: Distortion Removal

- Radial distortion causes straight lines to appear curved.



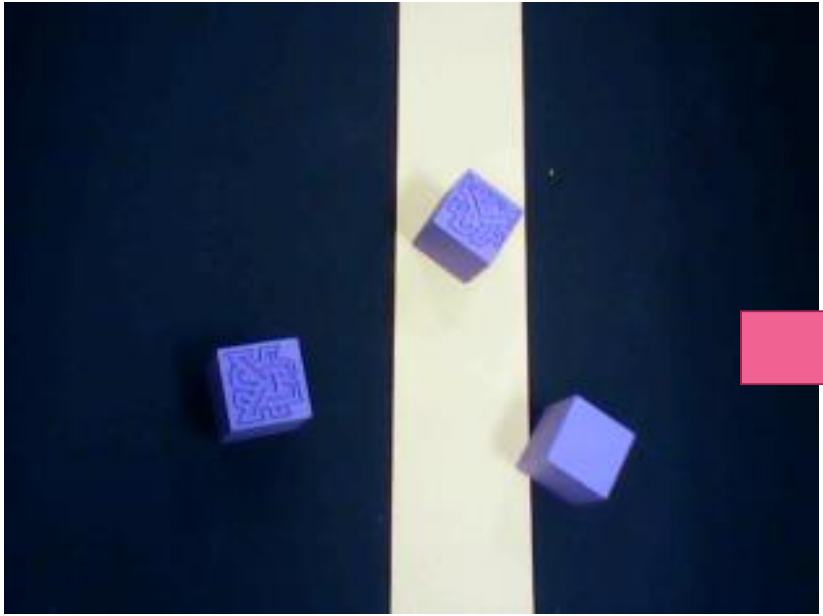
Original Image

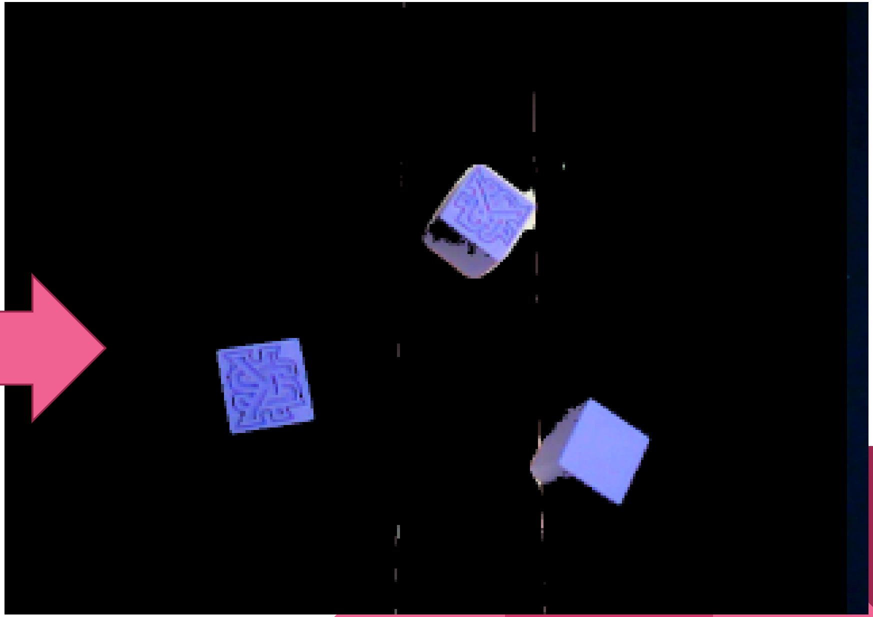


Undistorted Image

Vision System Process: Color Filtering

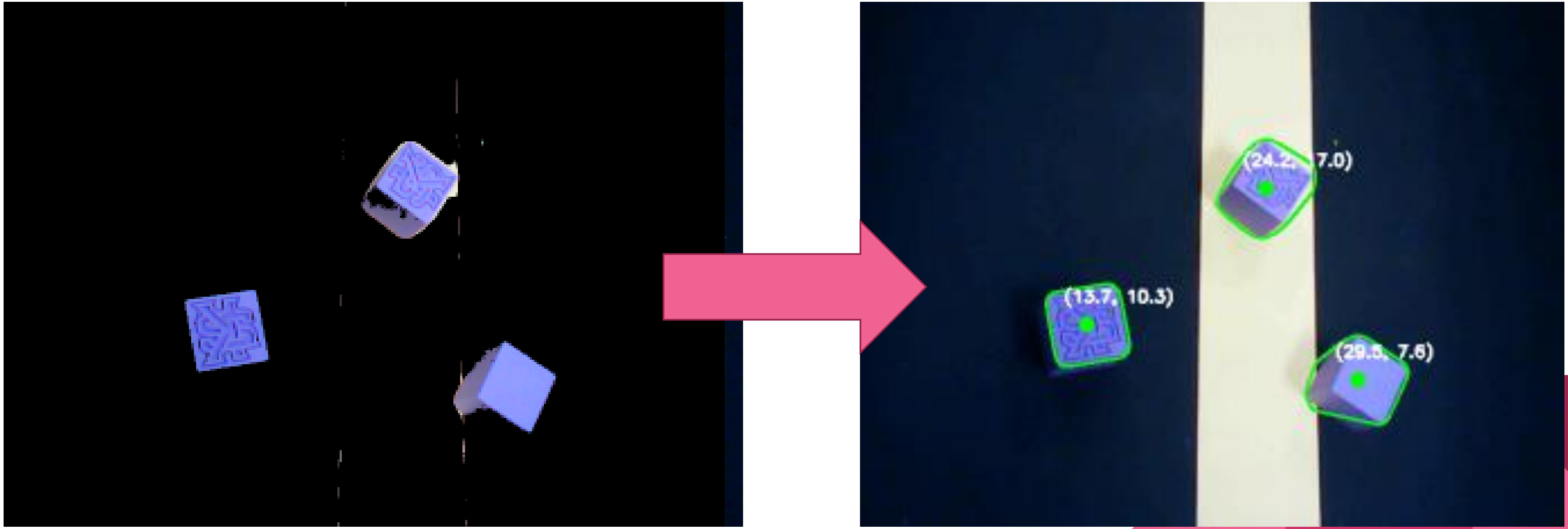
- Color-based recognition helps the robot to find specific colors in an image.





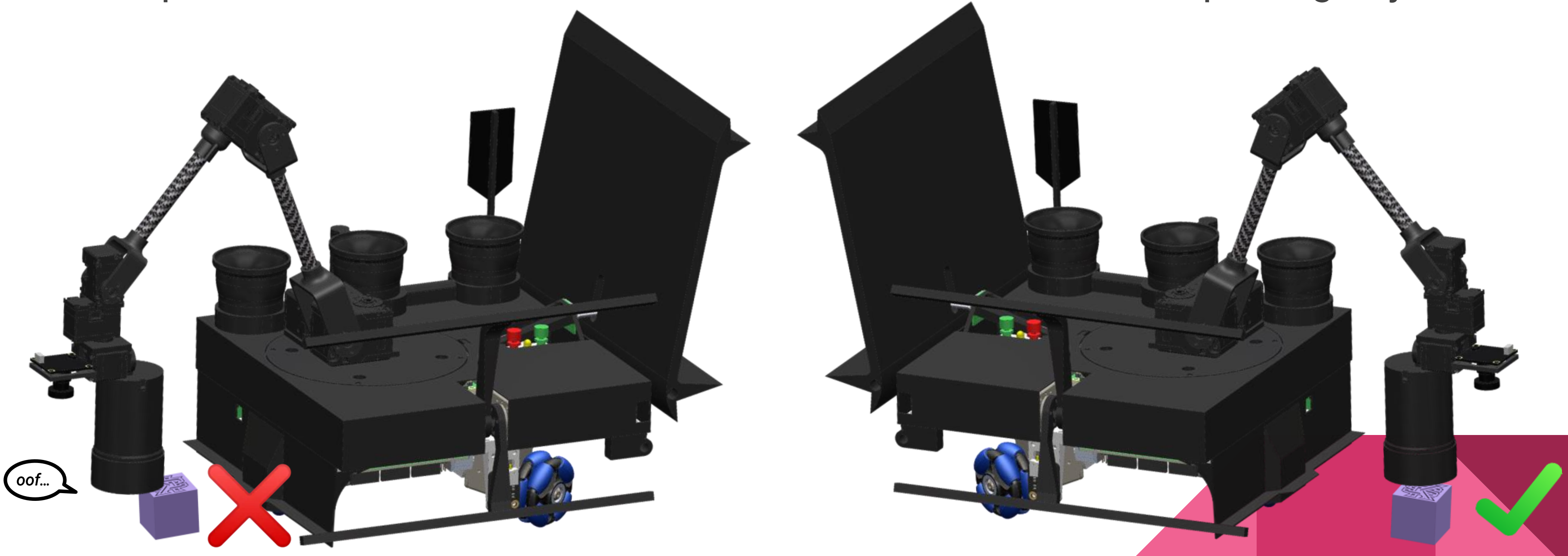
Vision System Process: Coordinate Calculation

- Spatial insight helps the robot recognizing shapes and finding distances (in the form of x-y coordinate pairs).



Vision System Process: Error Correction

- Experimental data lets us model and reduce the error when picking objects.



Vision System Demo

Competition Summary

Day 1:

We tested the bot several times before our trip. Then, we got to Atlanta and we settled at the hotel.

Day 2:

We realized one of the wires connecting to the camera was cut during travel, we immediately started in a solution for this. Fortunately, we had back up parts for the camera.

Day 3:

We finished calibrating for the day and participated.

Competition Results

- Design competition
  - Competition for design of robots
  - 1st place
- Hardware competition
  - Competition for performance of robot
  - 9th place out of 54 teams
    - Still 2nd highest total points
  - Top 2 scores of entire competition

	School	1st	2nd	3rd	total	Final	Average
1	Virginia Tech Main	96	99	99	294	30	81.0
2	UNCA	104	107	30	241		80.3
3	Clemson University	42	72	79	193	99	73.0
4	Jackson State University	30	100	60	190		63.3
5	High Point University	35	35	70	140	80	55.0
6	Bob Jones University	35	47	90	172	30	50.5
7	Western Carolina University	30	38	73	141		47.0
8	University of Kentucky #2 Main	35	40	60	135		45.0
9	The University of Sothern Mississippi	55	35	30	120		40.0
10	University of Flordia	30	30	55	115		38.3
11	Georgia Sothern University	35	35	30	100		33.3
12	Tennessee Technological University	35	35	30	100		33.3
13	UNC Charlotte	60	0	30	90		30.0
14	Flordia State University	10	55	25	90		30.0
15	Mississippi State University	25	30	25	80		26.7
16	Valencia College	15	44	20	79		26.3
17	Miami Dade College	25	25	20	70		23.3
18	Christian Brothers University	25	25	10	60		20.0
19	University of technology, Jamaica	20	20	20	60		20.0
20	University of Sothern Indiana	10	40	5	55		18.3

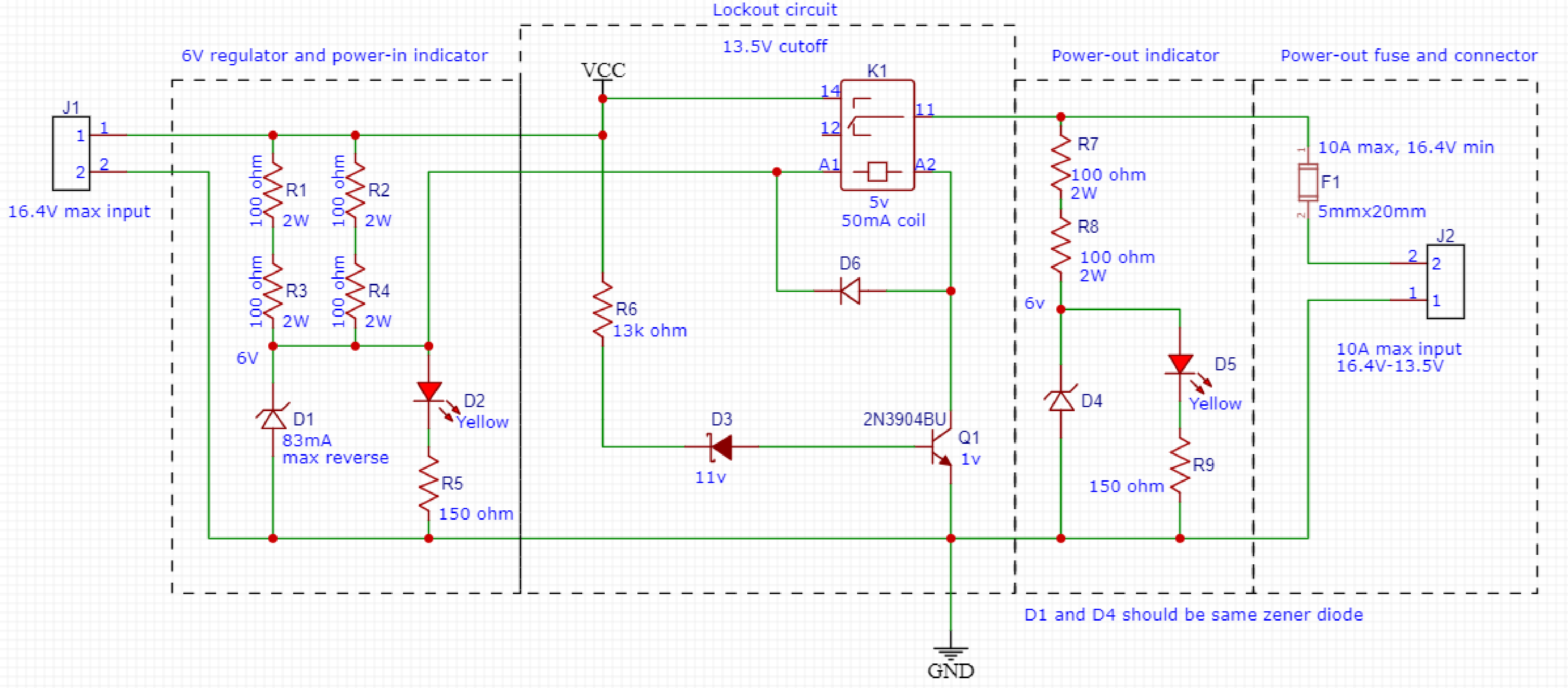

1st
IEEE
REGION 3
HARDWARE
DESIGN
COMPETITION
SoutheastCon

Conclusion

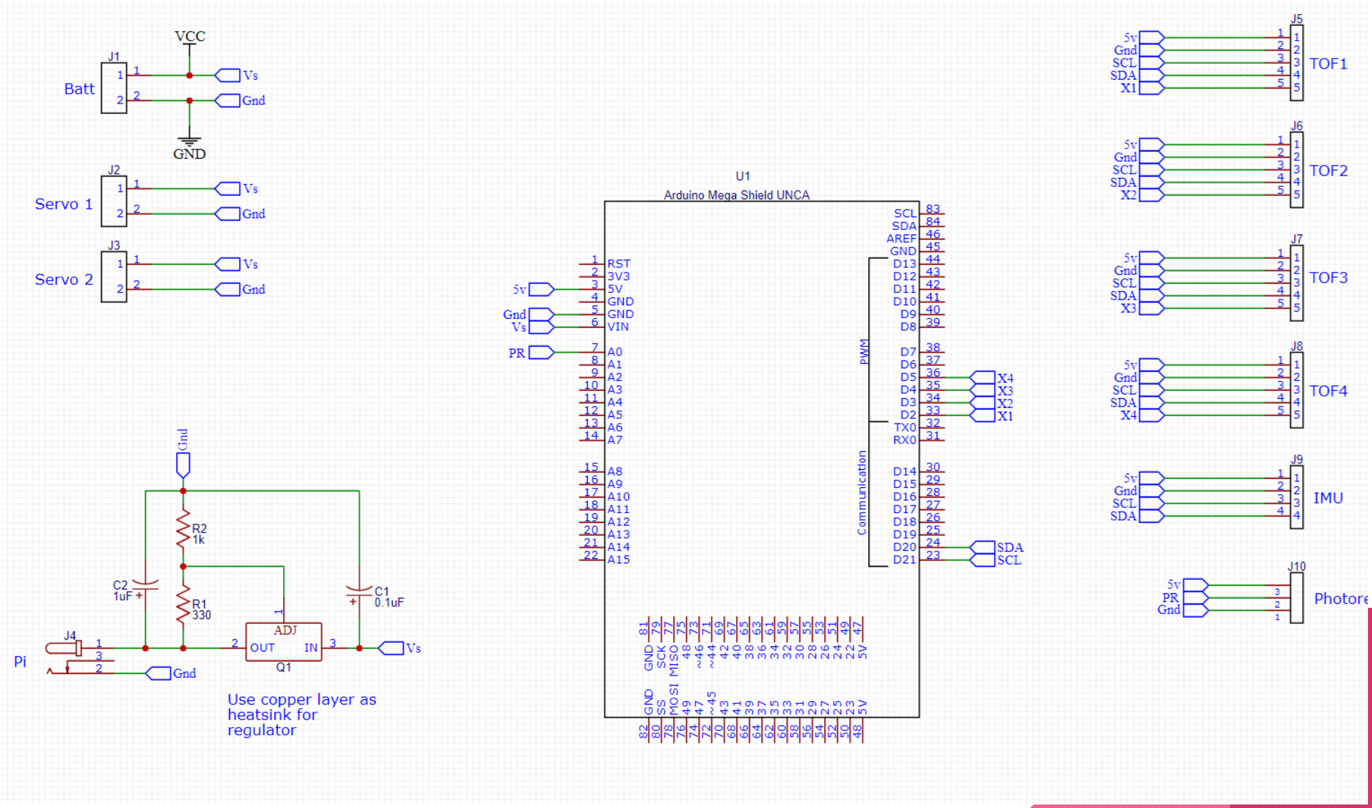
- We started with an MVP and ended with a completed robot.
- We went to the IEEE competition to prove ourselves and represent the UNCA/NCSU Joint Mechatronics program.
- We were able to place top 10 in the competition. Also, our team won 1st place in hardware design for best engineered robot.
- As a senior design team, we were able to work together and complete the objectives of the competition.

Questions?

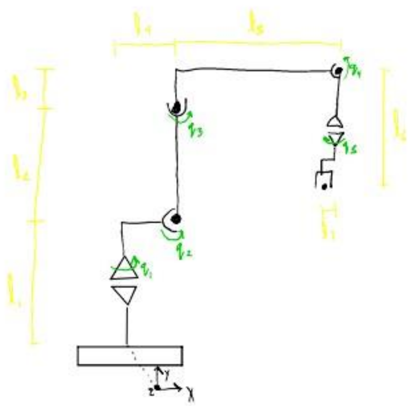
Appendix 1 - Li-Ion Protection Circuit



Appendix 2 - Power and Sensor Circuit



Appendix 3 - Analytical Inverse Kinematics



input (x, y, z, θ)

$$q_1 = \text{atan}\left(\frac{-z}{x}\right)$$

$$q_2 =$$

$$q_3 =$$

$$q_4 = -\pi/2 - q_3 - q_2$$

$$q_5 = \theta$$

output $x\ y\ z\ \theta\ \phi$

$$r_2 = (l_4 - l_2 \sin(q_2) - l_3 \sin(q_3 + q_2) + l_5 \cos(q_3 + q_2) - l_7)$$

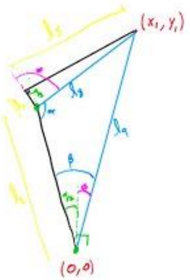
$$x_2 = r \cos(q_1)$$

$$y_2 = l_1 + l_2 \cos(q_2) + l_3 \cos(q_2 + q_3) + l_5 \sin(q_2 + q_3) - l_6$$

$$z_2 = -r \sin(q_1)$$

$$\theta_2 = q_5$$

$$\phi_2 = q_6$$



$$x_1 = \sqrt{x^2 + z^2} + l_7 - l_4$$

$$y_1 = y - l_1 + l_6$$

$$l_8 = \sqrt{l_3^2 + l_5^2}$$

$$l_9 = \sqrt{x_1^2 + y_1^2}$$

$$q_3 = \alpha + \omega - \pi$$

$$q_2 = \beta - \phi$$

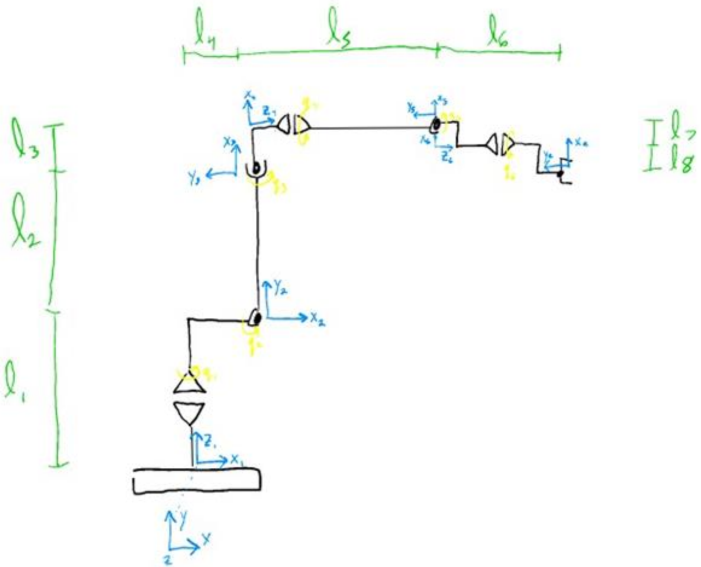
$$\omega = \cos^{-1}\left(\frac{l_3}{l_8}\right)$$

$$\phi = \tan^{-1}\left(\frac{x_1}{y_1}\right)$$

$$\beta = \cos^{-1}\left(\frac{l_2^2 + l_9^2 - l_8^2}{2 l_2 l_9}\right)$$

$$\alpha = \cos^{-1}\left(\frac{l_8^2 + l_2^2 - l_9^2}{2 l_8 l_2}\right)$$

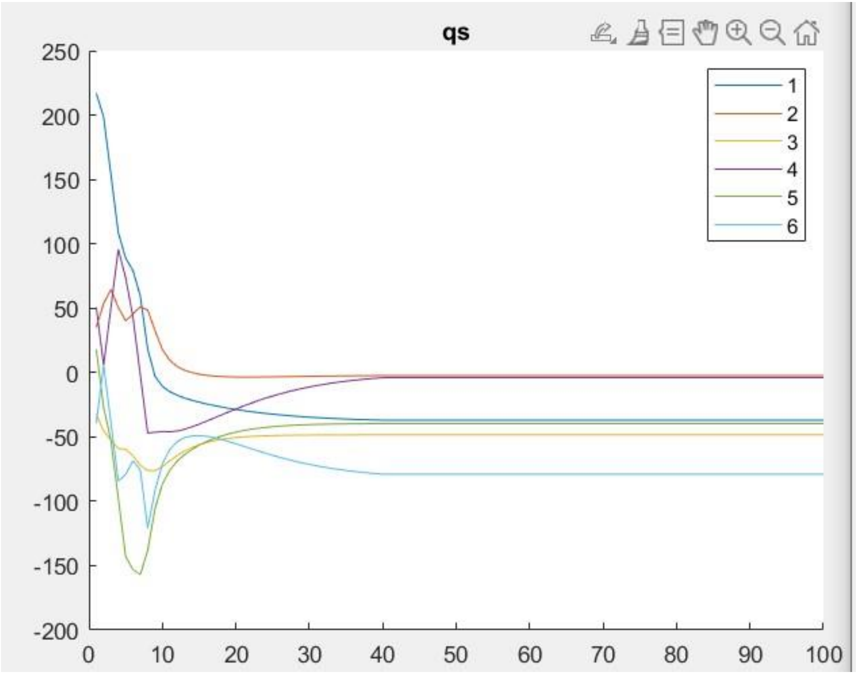
Appendix 4 - DH Parameters

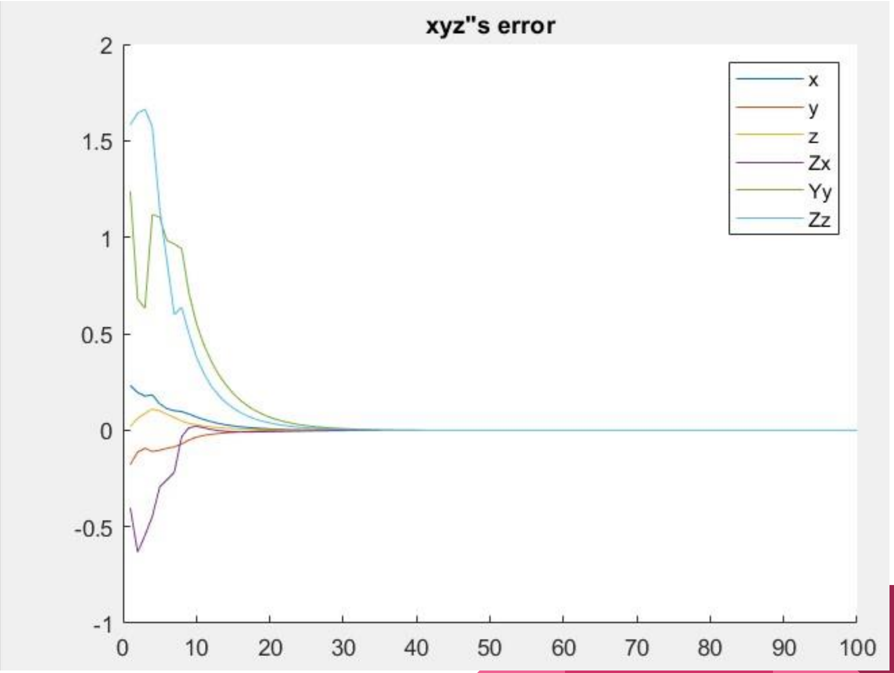


	θ	d	a	α
0	0	0	0	$-\pi/2$
1	q_1	l_1	l_4	$\pi/2$
2	$q_2+\pi/2$	0	l_2	0
3	q_3	0	l_3	$\pi/2$
4	q_4	l_5	0	$-\pi/2$
5	q_5	0	$-l_7$	$\pi/2$
6	q_6	l_6	$-l_8$	$-\pi/2$

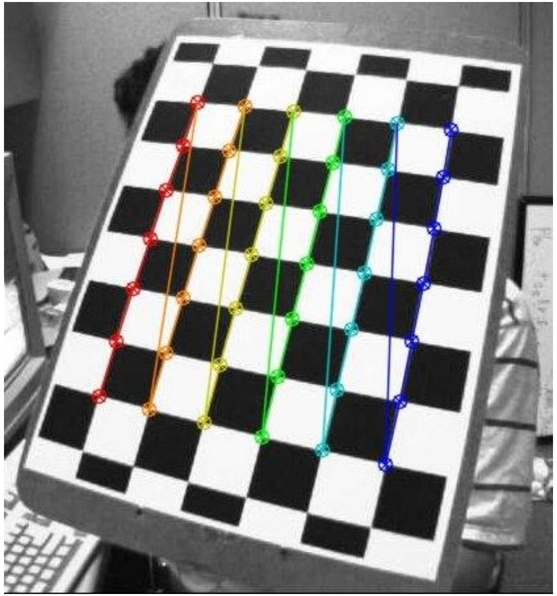
	mm
l_1	32.5
l_2	162
l_3	24
l_4	24
l_5	148.5
l_6	75.34
l_7	0.5
l_8	16.5

Appendix 5 - Gradient Descent





Appendix 6 - Computer Vision: Distortion Removal



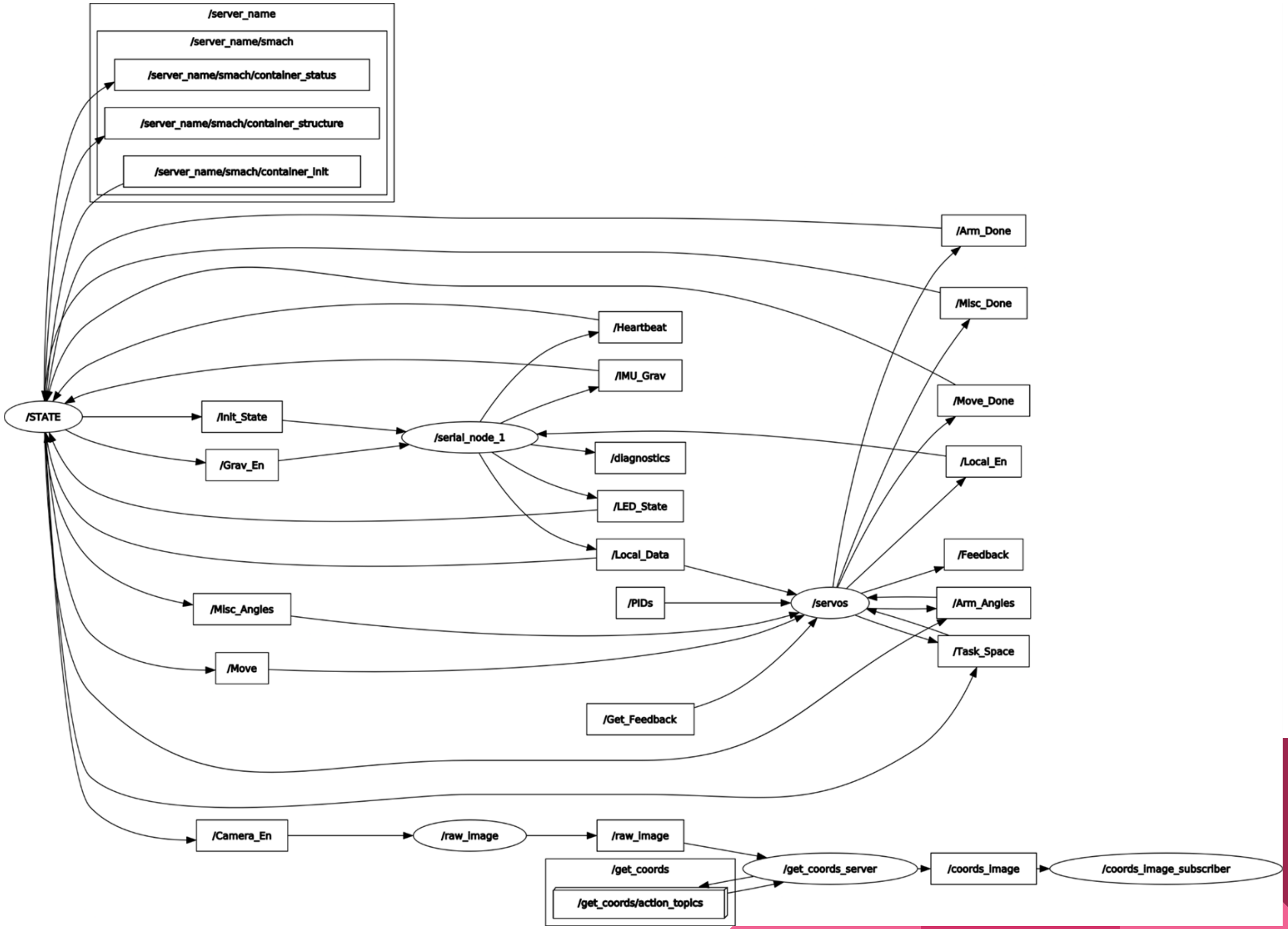
Two major kinds of distortion are radial distortion and tangential distortion:

- Radial distortion causes straight lines to appear curved. Radial distortion becomes larger the farther points are from the center of the image.

- Tangential distortion occurs because the image-taking lense is not aligned perfectly parallel to the imaging plane. So, some areas in the image may look nearer than expected.

[1] “Camera Calibration” OpenCV, https://docs.opencv.org/4.x/dc/dbb/tutorial_py_calibration.html (accessed Apr. 11, 2024).

ROS graphs


/server_name
/server_name/smach
/server_name/smach/container_status
/server_name/smach/container_structure
/server_name/smach/container_init
/Arm_Done
/Misc_Done
/Heartbeat
/IMU_Grav
/Move_Done
/STATE
/Init_State
/serial_node_1
/Local_En
/Grav_En
/diagnostics
/LED_State
/Local_Data
/Feedback
/PIDs
/servos
/Arm_Angles
/Misc_Angles
/Task_Space
/Move
/Get_Feedback
/Camera_En
/raw_image
/raw_image
/get_coords
/get_coords_server
/coords_image
/coords_image_subscriber
/get_coords/action_topics