Team 3476
Code Orange
Technical Documentation

Presenting our 2024 Robot:

HALEIWA

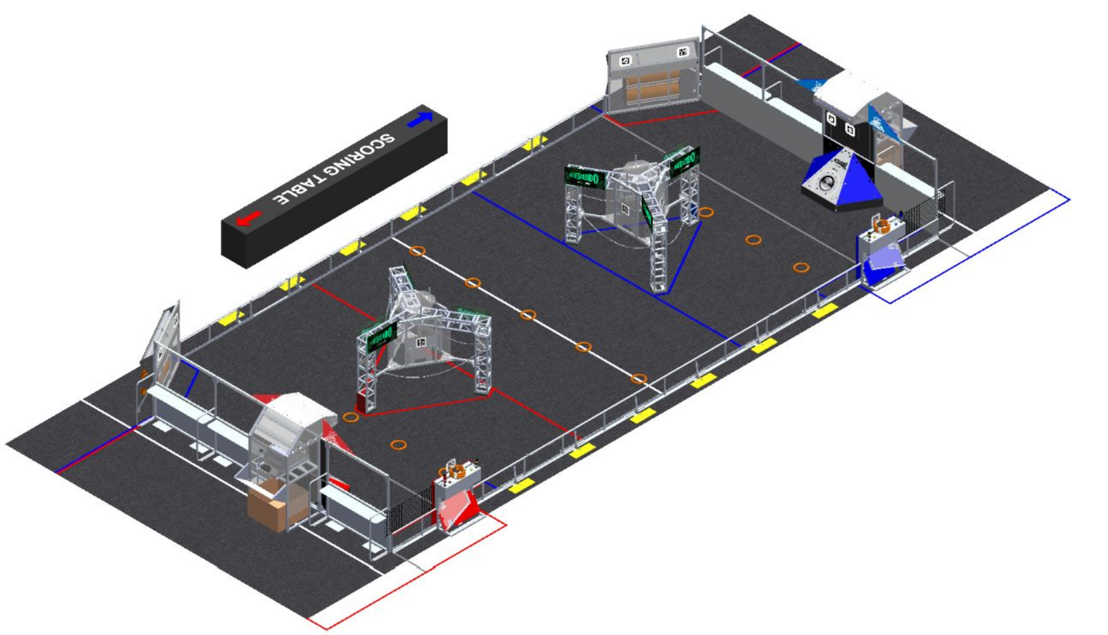
Table of Contents

I. Design Process

Design Requirements

Actions:

- Intake game pieces from the ground and human player substation
- Score notes into all point locations
- Score game pieces during auto, telop, and end game
- Climb and score during endgame for ranking points
- Fast cycles to maximize notes scored



Temporary: Wish, Prefer, Demand List

1	Specifications	W/P/D
2	hang on the chain	Demand
3	floor pickup of notes	Demand
4	score notes into speaker	Demand
5	fast cycle times	Demand
6	can drive fast	Demand
7	manuverable	Demand
8	accurate scoring	Demand
9	intake directly from source	Prefer
10	score note into trap	Demand
11	use an auto preload	Demand
12	robust auto, correct for errors	Prefer
13	don't run over notes	Prefer
14	score notes from our wing into speaker	Prefer
15	score notes from middle of field into speaker	Wish
16	score from in front of the subwoofer into speaker	Demand
17	score from behind alliance partner	Wish
18	score while touching podium with bumpers into speaker	Prefer
19	score notes from far into amp	Wish
20	score notes from close into speaker	Prefer
21	score notes from close into amp	Wish
22	score notes from wall into amp	Demand
23	don't tip	Demand
24	move out of starting zone in auto	Demand
25	buddy climb with 1	Prefer
26	buddy climb with 2	Wish
27	easily repairable	Demand
28	use apriltags for location	Demand
29	score into amp	Demand
30	high scoring auto	Demand
31	no-jamming floor intake	Demand
32	holonomic drivebase	Demand
33	don't drop pieces once controlled	Demand
34	robust robot	Demand
35	be able to remove pieces while unpowered	Demand
36	fit under the chain and bar	Prefer
37	light	Wish
38	low cog while driving	Demand
39	automatic scoring	Prefer
40	automatic ground pickup	Prefer

Overall Design Considerations

Design requirements:

- Score in all scoring locations
- Lightweight and fast
- Short cycle times
- Easy to repair
- Comply with all game rules

Designs considered:

- 2 segment arm
    - Pros:
        - Pass through the center of robot
        - Protected shooting against defense
    - Cons:
        - Complicated
        - Hard to repair
        - Unpreferable stow position
- Elevator Pivot wrist
    - Cons:
        - Complicated
        - Harder to repair
        - Center of CG towards one side of robot
    - Pros:
        - Nearly whole width intake
        - No Intake shooter handoff

- 25x25 drivebase
    - Pros:
        - Passes defence and easier maneuver
        - Lighter to go faster
    - Cons:
        - Limited space
        - Can easily tip

Overall Design Considerations

- 4 Bar intake
  - Cons:
    - Needs hopper
    - Extra time creating a intake shooter handoff
- Fit under the core
  - Cons:
    - Size restriction
  - Pros:
    - Faster cycle times
- Under Bumper Intake to shooter
  - Cons:
    - Maximum size constraint
    - Handoff time from intake to shooter using a hopper
  - Pros:
    - Protected from contact
    - Touch it own it

Carriage

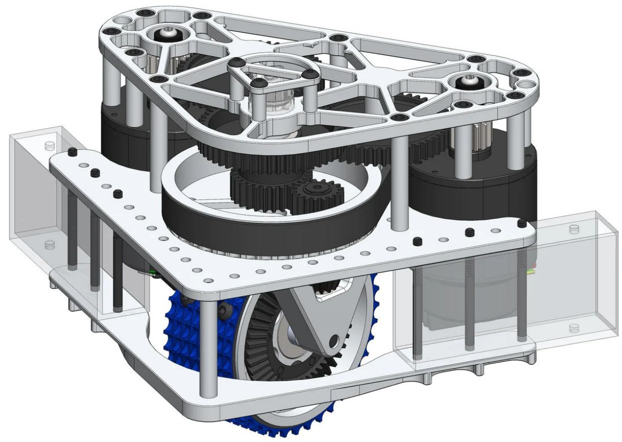
Design requirements:

- Lightweight
- Robust
- Low Backlash
- Compact
- Quick rotation

Designs considered:

- Chain and Sprocket
    - Pros:
        - High strength
    - Cons:
        - Some Backlash
        - Tensioning is required
        - Takes up more space
- Roller pinion Gear and Roller pinion
    - Pros:
        - High strength
        - No backlash
        - Compact
    - Cons:
        - More parts to assemble
        - Custom manufactured Pinion and Sprocket

Drivebase



Design requirements:

- Fast
- Easy to wire
- Comply with rules
- Fit all subsystems
- FIt all electronics
- Doesn’t tip
- Robust

Designs considered:

- Swerve Gearing
  - Quickly changeable from speed to torque w/Kraken X60

- Bumper mounting design
  - L brackets to prevent screw stripping
  - Pins to keep bumpers mounted in place
  - Handles for more portability
  - Lowered to prevent notes from getting under

- Note Blockers
  - Mounted at an angle so notes slide off easily
  - Hinge used to access battery and important electronics
  - At stowed climber height to ensure notes can slide from wherever they land

- ⅛ Box tubing vs. 1/16 Box tubing - more weight vs sturdiness

- 25x25
  - Pros:
    - Small and fast
  - Cons:
    - Not enough space for electronics and subsystems
    - Easier to tip

- 30x30
  - Pros:
    - Space for subsystem and electronics
    - More stability
  - Cons:
    - Slower and heavier

Pivot


Design requirements:

- Sturdy/highload
- Hold whole weight of robot
- Move the arm quick enough
- Low backlash

Designs considered:

- V groove bearing
    - Less wiggle
    - High load
    - Easier to switch out and repa
    - Complex
- Sprocket Gear
    - Pros:
        - Stable
        - 2 roller pinion design
        - Big enough to fit V groove on inner ring
    - Cons:
        - Needs to be made, not a COTS
        - Completed to assemble
- 2 Rigid Plates to mount
    - Elevator
    - Limlights
    - RSL
    - Radio
    - Ethernet Switch
    - Drag Chain
    - Enough space for an access hole
    - 2 motors
- Eccentric VS Concentric bearing shafts
    - Eccentric can be tuned to have more contact, vibration or contact will cause an offset
    - Concentric cannot be tuned but won’t move on contact

Intake/Shooter

Design requirements:

- Pick up Notes
- Make it a “Touch it own It” intake
- Lightweight
    - Elevator and Pivot load reduced

Intakes considered:

- Vertical Rollers
    - Pros:
        - “Touch it Own It” Intake
        - Consistent amount of force applied
    - Cons:
        - Lower strength
        - Uses more rollers
        - Catch debris from the floor
- Horizontal rollers
    - Pros:
        - “Touch it Own It” Intake
    - Cons:
        - Inconsistent
        - Variable amount of compression depending on note position
        - Hard to angle

Shooters considered

- Vertical Rollers
    - Pros:
        - Consistent amount of force applied
        - Contact on note throughout path
    - Cons
        - Lower Strength
- Horizontal rollers
    - Pros:
        - Able to use self cleaning flywheels
    - Cons:
        - Inconsistent amount of force applied
        - Variable amount of compression depending on note position

Climber

Design requirements:

- Not slide down the chain
- Climb in <5 sec
- Hold the robot in a position to score in trap

Designs considered:

- Hook on the Climber
    - Pros:
        - Uses the same motors as pivot
    - Cons:
        - Cantilever
        - Can't score trap
- 1 Hook in a Slot
    - Pros:
        - Thin
        - Low Profile
    - Cons:
        - Flimsy
        - Chain Goes through Pivot
- 1 hook
    - Pros:
        - 3 second climb
        - Non Slip design
    - Cons:
        - High pressure on rope
- Driverail climbers
    - Pros:
        - Compact
        - Fast deployment
    - Cons:
        - Weak
        - Complex design
        - Requires multiple motors
        - Unable to fit on Pivot Side

Climber

- Drivebase forks
  - Pros:
    - Stabilizes robot
    - Prevent robot from tipping out of trap position
    - Reduces Tilt
    - Surgical tubing and servo arm deployment
- Winch design
  - 1 motor climb
  - Gas shocks and torsion springs for deployment
  - String and winch to Climbr
    - Fair climbing speed with high torque
    - 2300lb rope

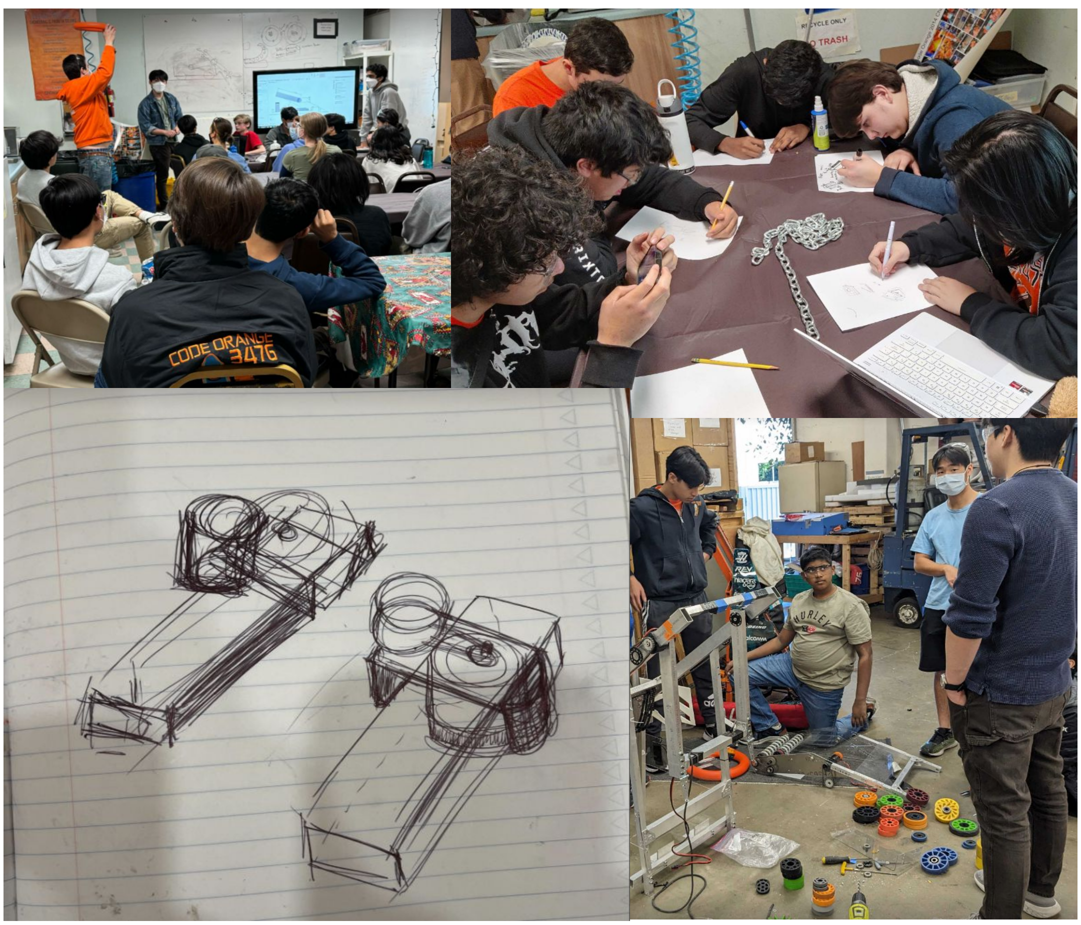
FINAL DESIGN:

For staying under the CoG, we chose to make multiple holes in the belly pan so we could shift the mechanism over if needed. To put the robot in a trap position we decided to add forks to the side so that we can stay flat while climbed. To stay less than 5” above the drive rails we went with a collapsing design with a winch in between the drive rails. This winch is a 20:1 Kraken X60 MaxPlanetary with a 1” spool. This Gear Ratio gave us the power to lift the robot fast and consistently. We also added a limit switch to stop the motor when it goes to the limit where we want.

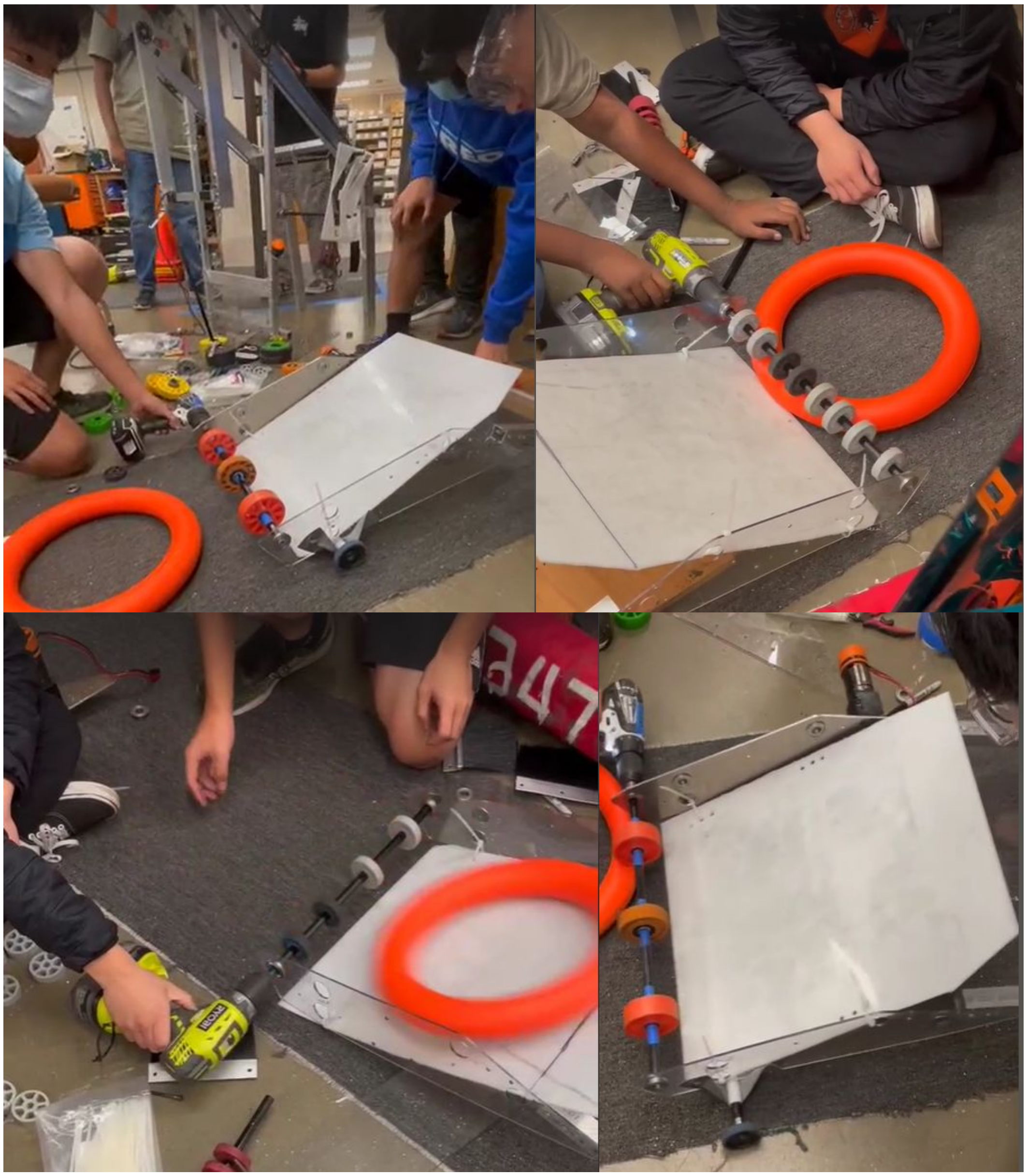
Prototyping

Next, our team splits into smaller subsystem groups to prototype ideas following our wish, prefer, demand list. These teams are composed of experienced and new students to promote passing down knowledge.

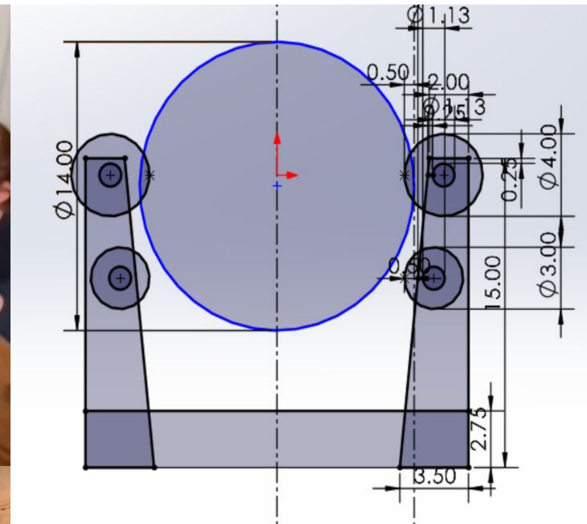
Subsystem groups brainstorm ideas and build prototypes with wood and scrap materials. It is important to collect **data** and document **functionality** through video in this step so designers can look back to determine subsystem specifications.



Intake

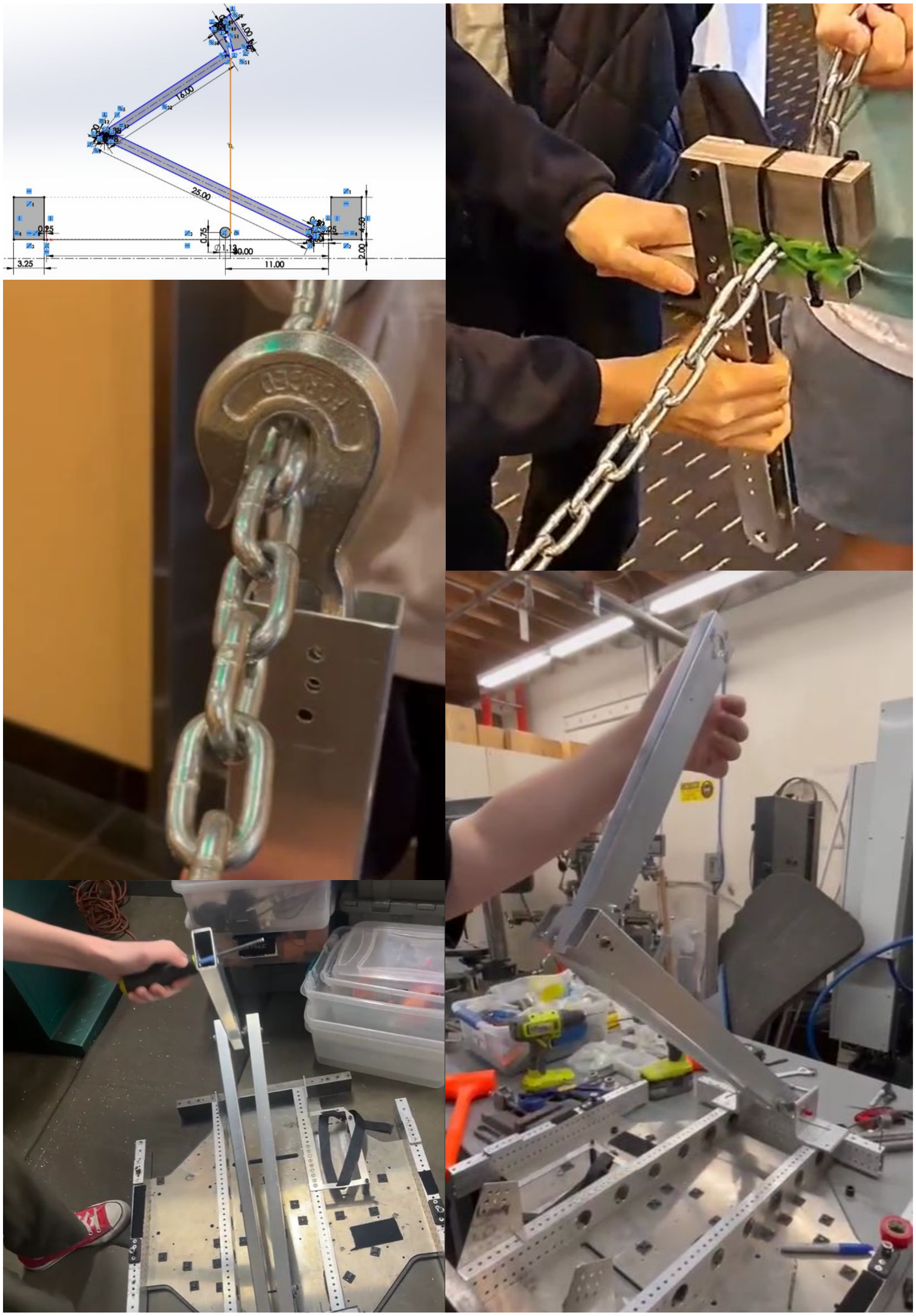


Shooter


Ø1.13
0.50
2.00
Ø14.00
Ø4.00
0.25
Ø3.00
15.00
2.75
3.50


PLEASE DO
NOT HANG
OR LEAN

Climber



CAD

Upon finishing prototyping we move on to the next step, CAD. We select our most successful prototypes of each mechanism and CADers begin the process of designing. Many meetings and open discussions take place to facilitate coordination between mechanisms. CADers adapt their designs based on constant feedback and new ideas. They then present their ideas to team members and mentors for approval.

After a design is approved, parts need to be made by machinists. Drawings are made for each part of the design so that machinists can fabricate the parts. The drawing is a 2D diagram with material specifications and dimensions so the part can be accurately machined.

New ideas and iterations are found throughout the season, so changes need to be implemented. CADers must quickly design and enact these changes so that the robot will be ready for competition. The CAD team is one of the most important divisions of our team because it results in a cohesive design that can be manufactured and assembled fluently.

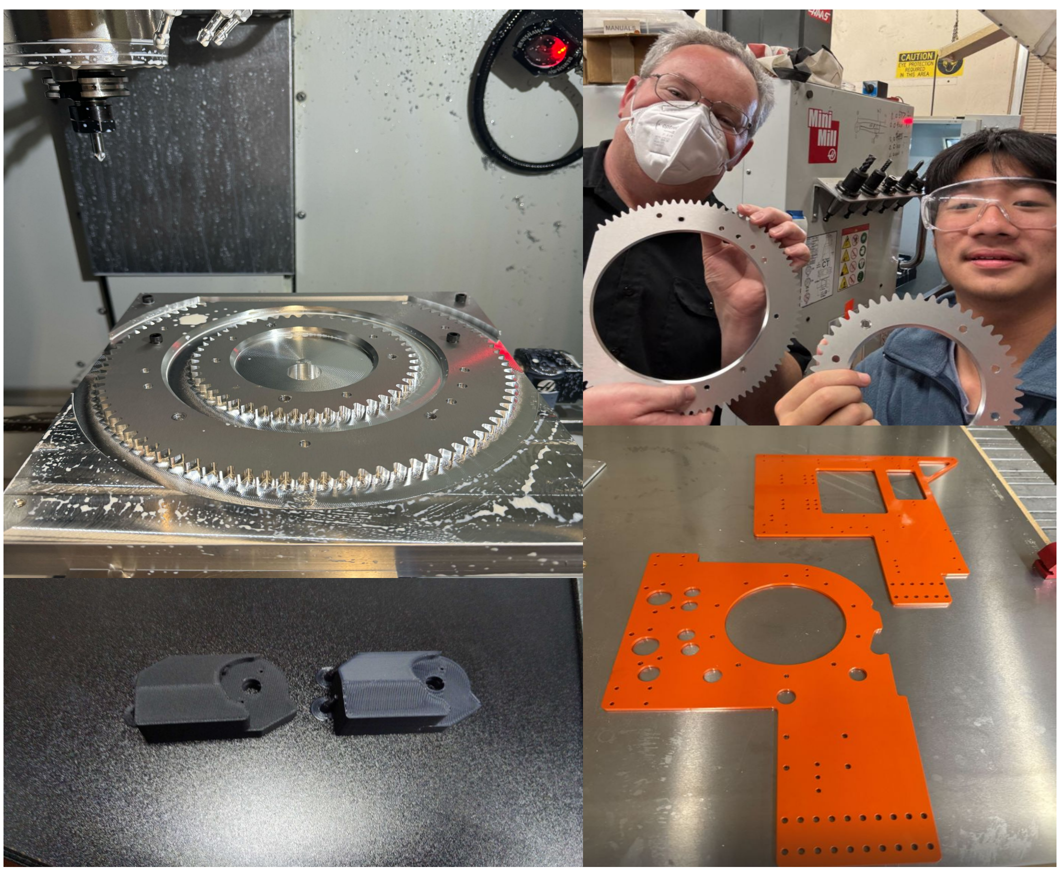
Fabrication

The manufacturing sub team has the most people on our team and works to bring the robot model, which is designed by the CADers, into reality.

With hundreds of parts to be manufactured, **organization** and **quality** is key. The manufacturing lead ensures each part is made to the tolerance specified on the drawing and completed parts are properly labeled for powder coating and assembly.

Our team uses **basic machinery** like the bandsaw, the chop saw, the bench grinder, and the belt sander as well as **advanced machinery** such as the mill, the lathe, the laser cutter, the CNC mill, and the CNC router to manufacture our robot.

Unique parts like camera mounts or custom pulleys can be created using our Ender, and Markforged 3D printers. The Markforged printer allows us to print with a carbon-fiber/nylon filament for load-bearing parts.

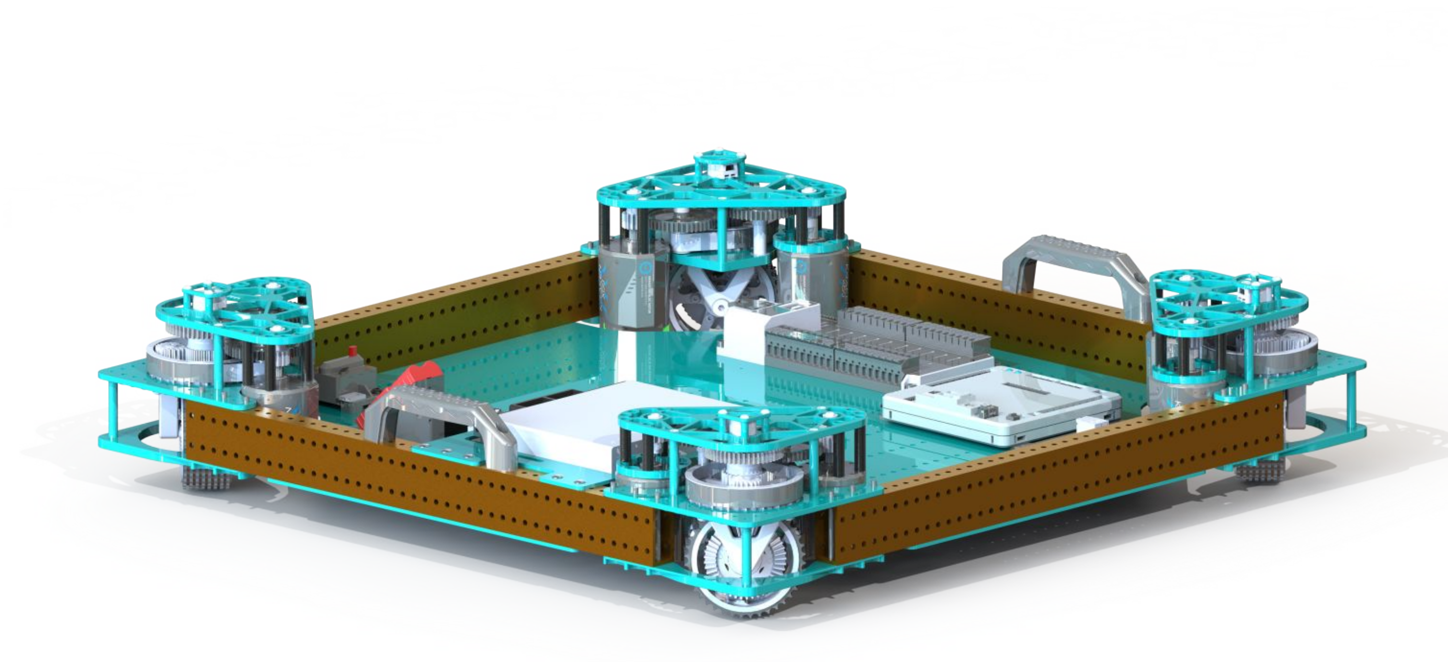


II. Mechanical Design

Final Design

Dimensions: 29.5x29.5

Drivebase



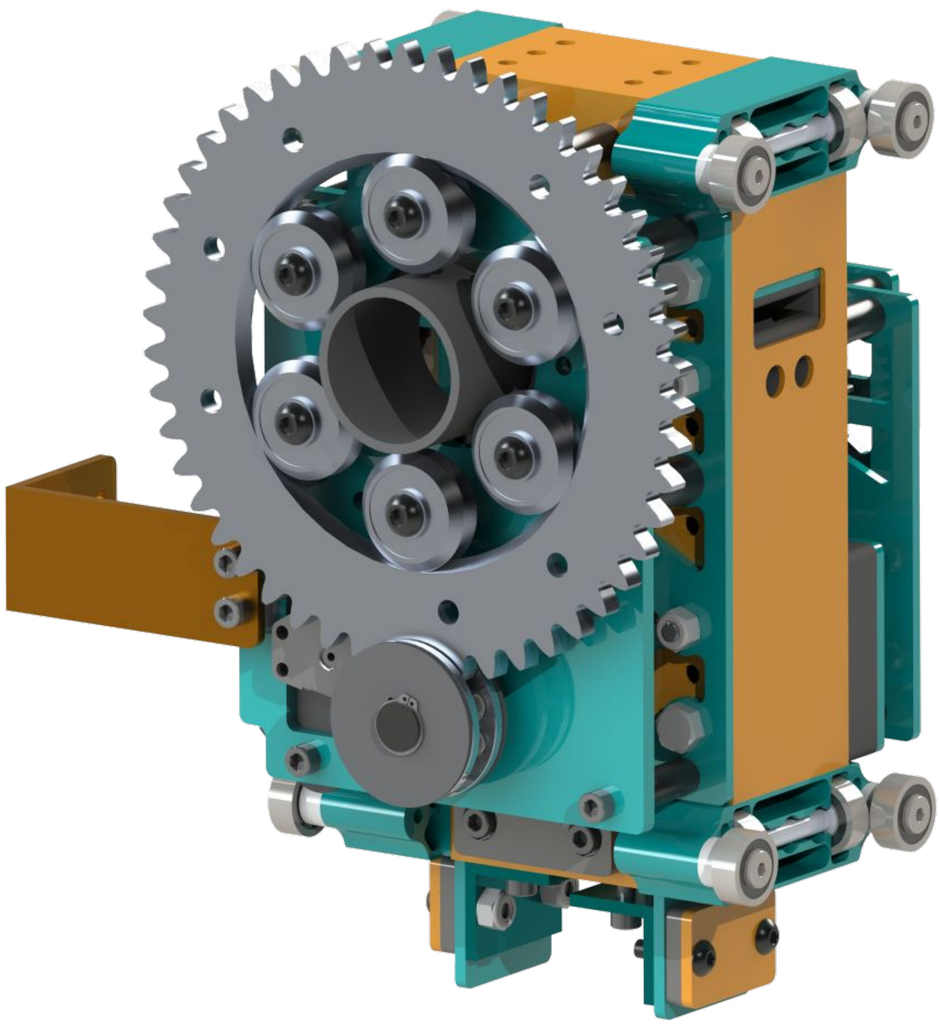
Structure

- 29.5x29.5 drivebase for space and stability
- Room for all electronics on drivebase
- Adjustable center box tubing for climber
- New L bracket design for bumpers
- Bumpers under 10lbs
- Flat battery to help balance robot
- Angled battery connector for ease of access

Swerve

- 4 MK4i swerve modules (2 Kraken X60s per module) - 16T Pinion adapter
- Black nitrile treads for traction
- Added custom swerve plates for easy ratio change
- CTRE CANcoder for location of swerve

Carriage



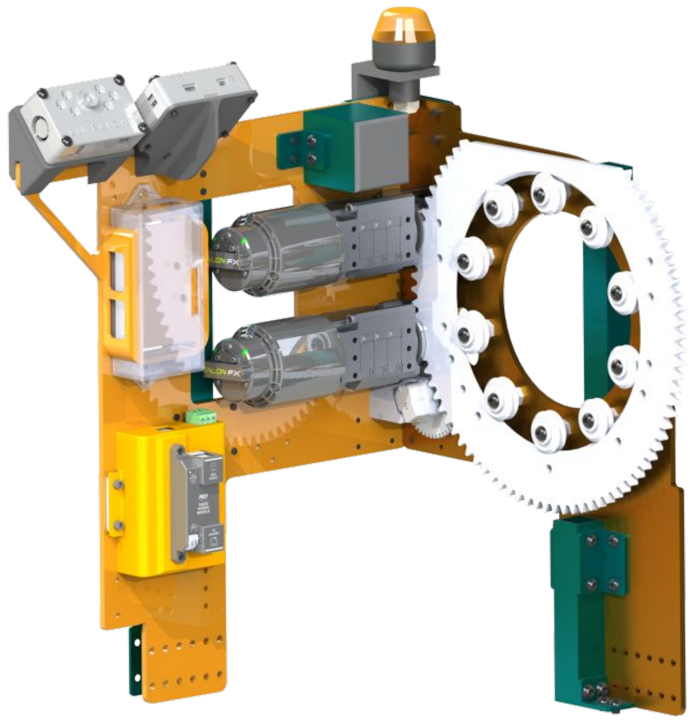
Roller Pinion and Gear Design

- Zero backlash for accurate positioning
- Gear with chamfer mounted on to v-groove bearings for rigidity
- 5:1 Ratio

Gearbox

- One Falcon motor with a 180° gear conversion to make it as compact as possible
- 9:1 Ratio

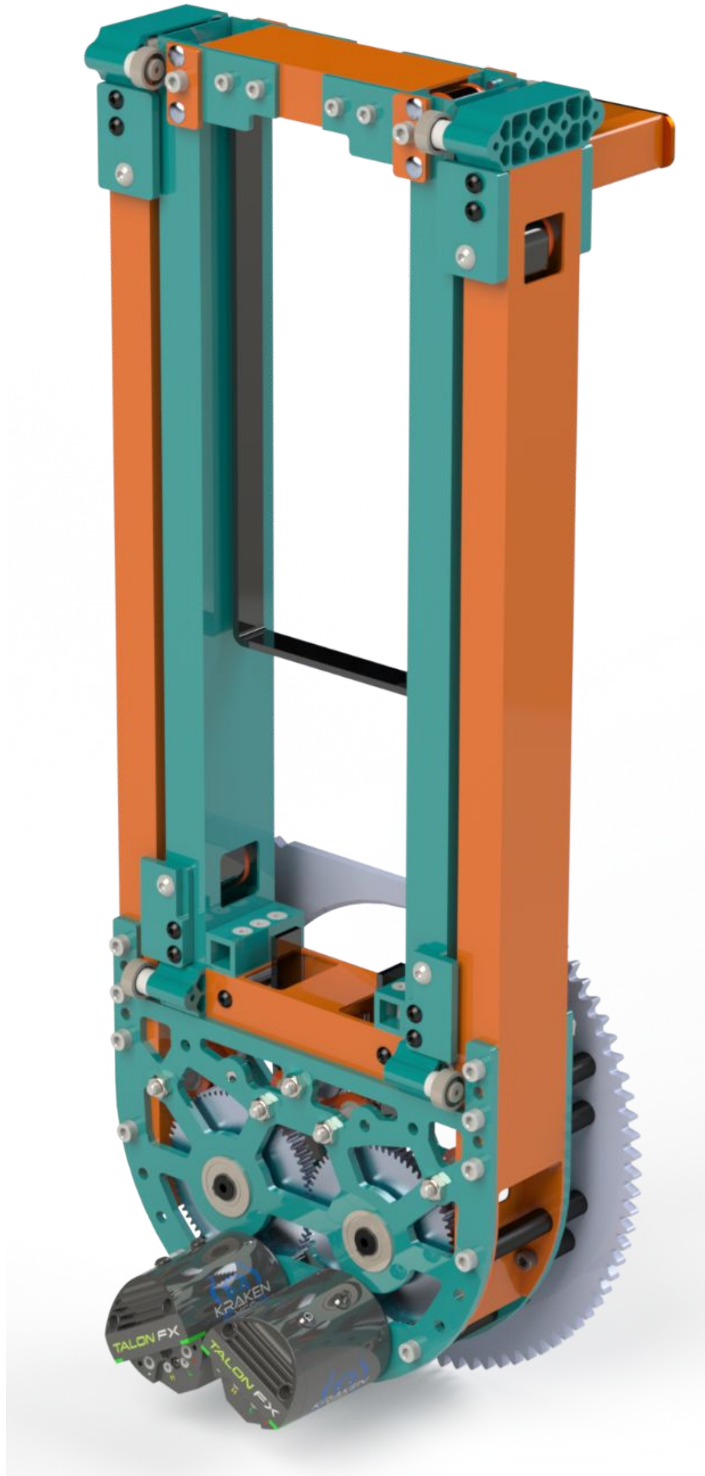
Pivot



Frame

- 11” Sprocket Gear
    - Held by 10 V groove bearings
- 2 Roller pinions Geared 9:1
- 2 Kraken Planetary Gearboxes 36:1
    - Final Ratio 324:1
- Side Box tubing support
- Gussets to drive rails for more support
- 2 Big plates
    - Mounts:
        - Motors & planetary gearbox
        - Radio and other electronics
        - Limelight Cameras
        - Nexen gear and pinions
        - Sprocket Gear Holds Elevator, Carriage, & Intake/Shooter

Elevator



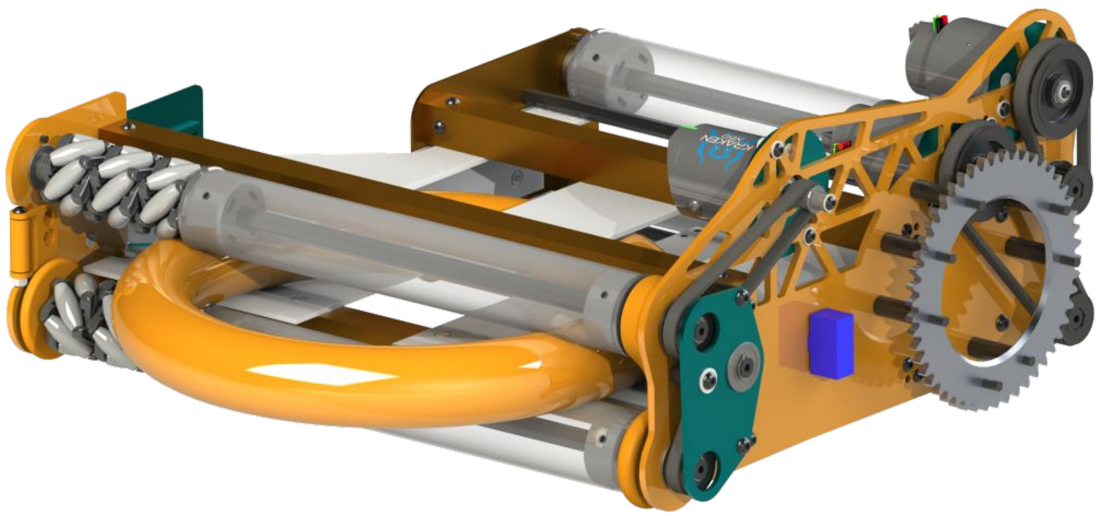
Elevator

- 2 Stage elevator driven by WCP (HTD 5) 18mm Wide Kevlar Backed Timing Belt
- Continuous rigging within elevator box tubes increase compactness, access ports decrease time to assemble
- 13 inches in less than a quarter of a second
- Utilizes 4 WCP inline bearing blocks with ¾ bearings to support radial and thrust loads

Gearbox

- Powered by 2 Kraken X60s for maximum current draw efficiency
- Mirrored 9:1 ratio to allow for 2 30T HTD 5mm pulleys used for driving the elevator
- 8 LB load WCP constant force spring preventing overextension of 1st stage during stow

Intake/Shooter



Structure

- Mounted onto zero backlash sprocket gear
- HDPE plates for low friction note sliding
- Intake Kraken X60 (1:2.38 gear ratio) for higher torque
- Falcon 500s for shooter (2:1 gear ratio)
    - Mecanum wheels in combination with a funnel plate for note centering
- Delrin rollers to reduce floor contact
- Plates to protect motion components

Rollers

- Polycarbonate rollers and mecanum wheels for shooter
- Colson and Banebot wheels for intake
- Slots in the plate to tension belts in case of loosening
- Uses two sided break beam for instant note detection
- Belts and gears to transfer motion

Climber

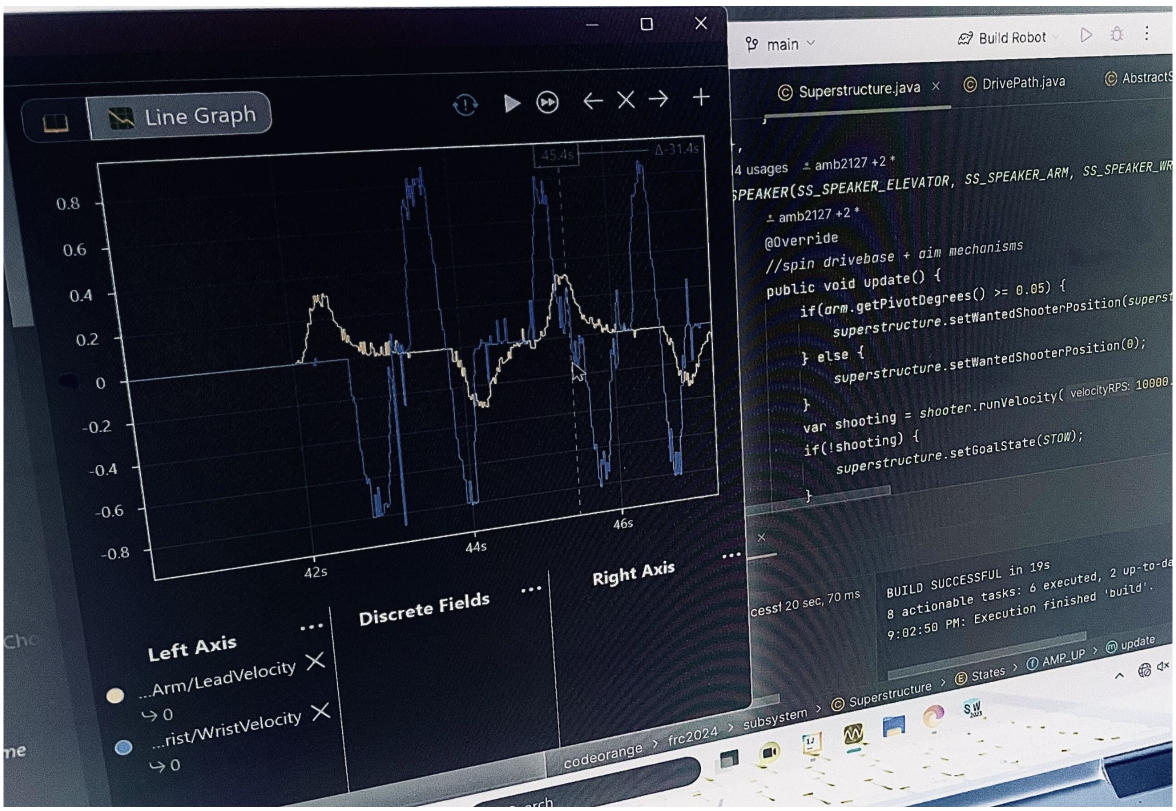
Gas Shock Supported Design

- Two arm linkage extends with gas shocks and torsion springs, deployed with a winch and utilizes 2300lb rope
- Torsion springs connect the climbing arm to the linkage allowing for proper movement and retraction
- Clevis hook design maximizes chain grip
- Spring supported forks are controlled by a Rev Servo, placement under stage provides stabilization during climb

Electronics and Mounting

- Compact drive rail attachment design allows for climber electronic space whilst leaving room for wiring of other subsystems
- This Kraken X60 powered design lets Haleiwa climb within 3 seconds

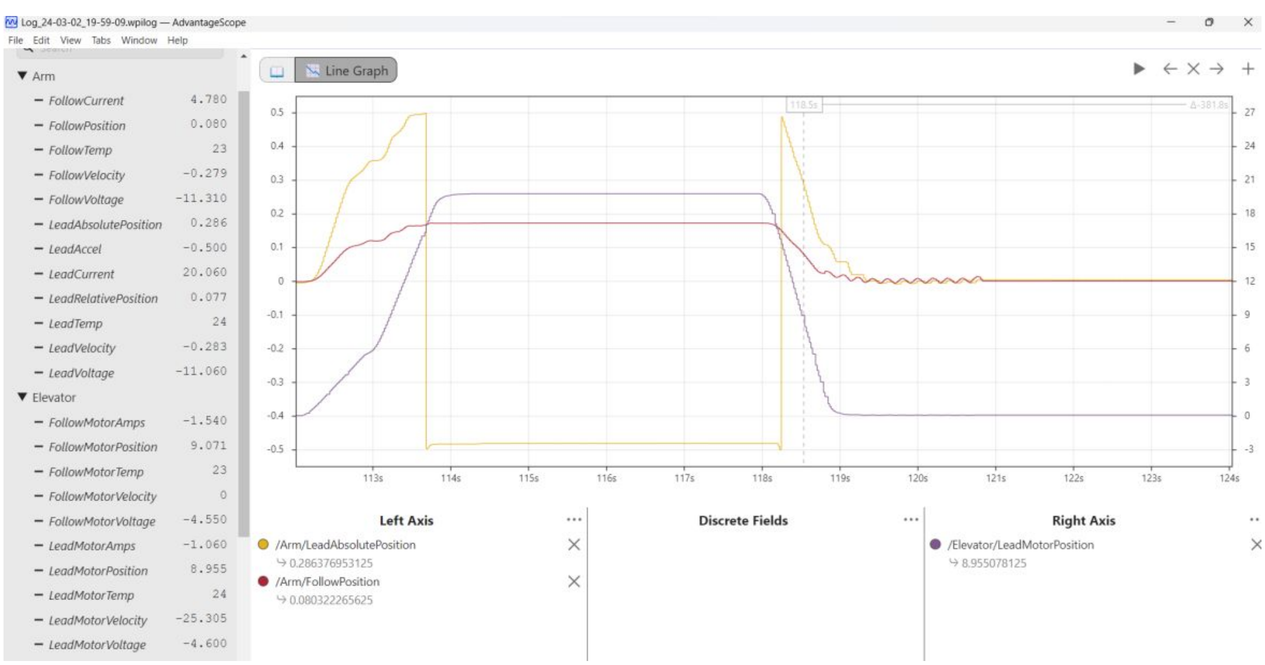
III. Programming



Programming Management

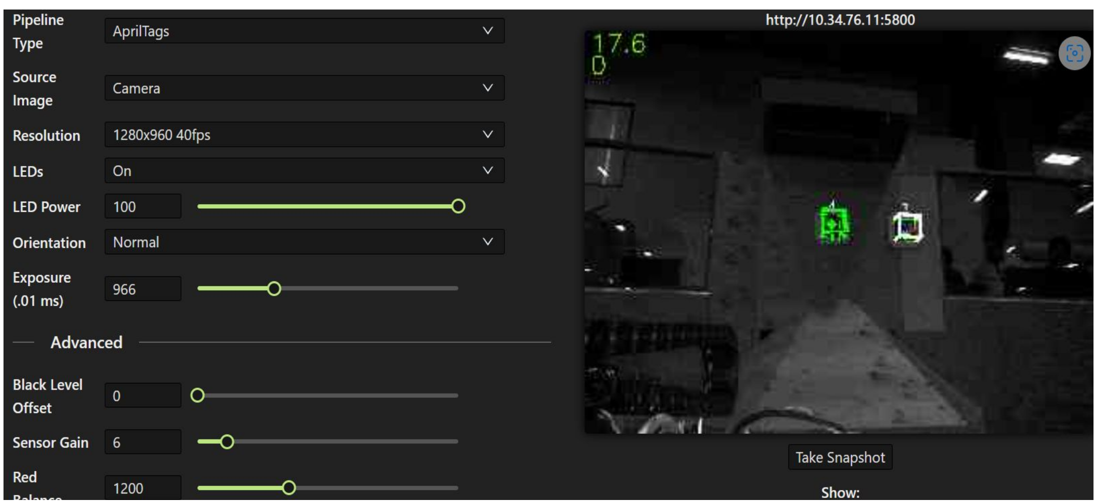
Since we have a **large programming team**, **organizing tasks** was a crucial step to success. Only two people in our team had previous programming experiences with FRC, so it was definitely a great learning experience for everyone.

- Organized tasks on Github
  - Prioritized based on **wish/prefer/demand**
  - Simple management with **easy team communication**
  - Able to submit code for review before merged into main branch
  - Ensures that all code run on the robot is ready for testing and mostly **bug-free**
- Programming update spreadsheet
  - Listed what each programmer was doing each day, kept **everyone updated**
- Assigned each member to a specific subsystem
  - Able to quickly get subsystem programs written and completed
  - Several new programmers **gained experience**
- AdvantageScope was always open while testing
  - **Records all inputs/outputs** for review after
  - Ensures fixed bugs



Vision & Pose Estimation

- Limelight Vision
  - **Tracks AprilTags** on field quickly
  - Detects robot **field location**
  - Easy integration and configuration
  - For **increased accuracy**, each vision measurement is assigned a level of 'trust,' and the least trusted values are thrown out
- Auto rotation towards speaker
  - Based on alliance speakers
  - Activates with **left trigger** pressed while **driving near speaker**



Limelight detection of two AprilTags on the speaker

Driver Assists

- Shooter/intake autostow
  - **Break beam** sensor attached on the wrist
  - Senses if there is a note currently in the shooter/intake
- Climb in two buttons
  - Button 1
    - Prepares elevator, wrist, and arm to climb
    - Extends climber and runs servos to extend balance arms for climb
  - Button 2
    - Hooks onto the chain and pulls robot up
  - To **avoid accidents** as climbers are single-use, drivers must hold down two buttons at once for either function, and movement stops if buttons are released
- Superstructure
  - Allows for movement to **preconfigured states** with a single button press
  - Some states, like speaker, take input from the vision to aim
  - Drivers **don't have to manually move mechanisms**

```
public static final double SS_SOURCEINTAKE_ELEVATOR = 6;
public static final double SS_SOURCEINTAKE_ARM = 0.13;
public static final double SS_SOURCEINTAKE_WRIST = 0.008888888888888888888;
public static final double SS_SOURCEINTAKE_CLIMBER = isPrototype() ? 0 : 0;
public static final double SS_AMP_ELEVATOR = isPrototype() ? 21.6 : 20;
public static final double SS_AMP_ARM = isPrototype() ? 0.16 : 0.175;
public static final double SS_AMP_WRIST = isPrototype() ? -0.24 : -0.275;
public static final double SS_AMP_CLIMBER = isPrototype() ? 0 : 0;
public static final double SS_SPEAKER_ELEVATOR = isPrototype() ? 10 : 6;
public static final double SS_SPEAKER_ARM = isPrototype() ? 0.125 : 0.125;
public static final double SS_SPEAKER_WRIST = isPrototype() ? 0 : 0;
public static final double SS_SPEAKER_CLIMBER = isPrototype() ? 0 : 0;
```

Superstructure states are defined with constants for the positions of each subsystem so that we can easily change them as well as have different numbers for each robot.

Autonomous

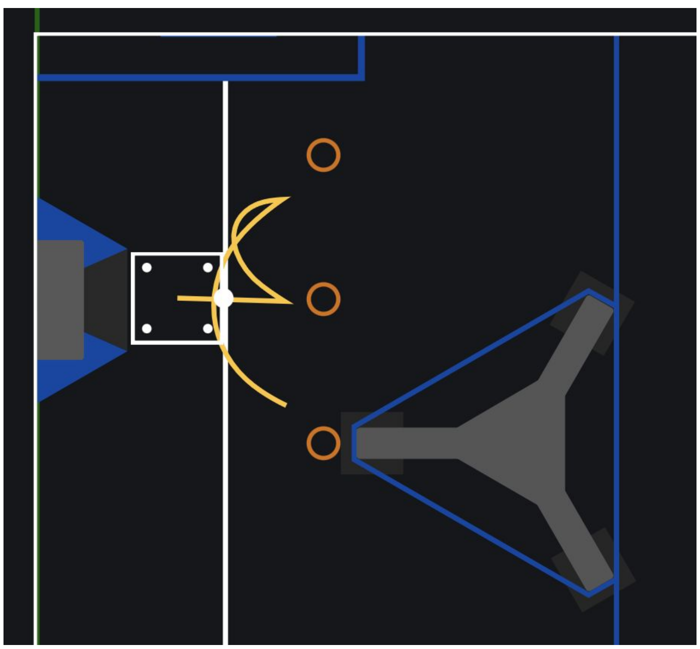
- Uses Choreo to generate time-optimized paths
  - **Object-oriented integration** of the Choreo trajectories into our code allows for easy addition of new autos whenever they are needed
  - Spreadsheet used for organization and prioritization when designing new paths
- Automates movements through superstructure
  - **Simplifies** the code even further – each auto trajectory's class file is easy to read and understand
  - Avoids **unnecessary repetition** between auto paths

```java
package org.codeorange.frc2024.auto.routines;

import com.choreo.lib.Choreo;
import com.choreo.lib.ChoreoTrajectory;
import org.codeorange.frc2024.auto.AutoEndedException;
import org.codeorange.frc2024.auto.actions.*;

public class FourPiece extends BaseRoutine {
    final ChoreoTrajectory driveToFirstNote;
    final ChoreoTrajectory driveToSecondNote;
    final ChoreoTrajectory driveToThirdNote;

    public FourPiece() {
        driveToFirstNote = Choreo.getTrajectory("4_pc_in_wing.1");
        driveToSecondNote = Choreo.getTrajectory("4_pc_in_wing.2");
        driveToThirdNote = Choreo.getTrajectory("4_pc_in_wing.3");
    }

    @Override
    protected void routine() throws AutoEndedException {
        runAction(new ParallelAction(new ResetOdometry(driveToFirstNote.sample(0)), new Shoot(52)));
        runAction(new ParallelAction(new SeriesAction(new GroundIntake(), new Shoot(33)), new SeriesAction(new Wait(0.25), new DrivePath(driveToFirstNote))));
        runAction(new ParallelAction(new SeriesAction(new GroundIntake(), new Shoot(28)), new SeriesAction(new DrivePath(driveToSecondNote))));
        runAction(new ParallelAction(new SeriesAction(new Wait(0.5), new GroundIntake(), new Wait(0.1), new Shoot(30)), new DrivePath(driveToThirdNote)));
    }
}
```

The above program creates a trajectory that shoots the preloaded note, then picks up and scores each of the three notes in the alliance zone. The entire autonomous movement code is contained in just four lines, making it very simple to make changes whenever necessary.

The below Choreo screenshots display one of our **basic autos**, which shoots the preloaded note, then picks up and shoots each note in the alliance zone. Based on its starting position, the robot picks up the notes in the **optimal order** to maximize our speed.

The dot on the side of the robot (the square) is in the location of the front of the robot. Our wrist allows us to rotate the intake and shooter mechanisms so that we can shoot out the back of our robot as well as the front, which removes time for rotating the robot and speeds up our paths.



Final Remarks

Time really flies by so fast. It only feels like weeks ago when we held our first meeting after kickoff. Our days spent working with each other have been filled with learning, collaboration, innovation, and excitement. We are so proud of the robot we've developed together as a team, and couldn't be more excited to compete with it!

Thanks to our parents, mentors, and sponsors for helping us bring our robot to life, and for believing in us.

And most importantly, we'd like to thank you for reading our technical binder, and for expressing interest in our robot. Feel free to visit our website: "teamcodeorange.com" for more information about us and our previous robots, CAD files, Github repository, and guides.

Once again, we're so grateful that you're here with us on our journey.

Thanks,
Code Orange, FRC Team 3476