

Engineering Notebook

355X

Team Number



Hexa-Force

Team Name

HEXA-FORCE

Fox Valley Robotics

School

06/11/2024

Start Date

N/A

End Date

1

Book #

of 1

V2.1 Date 6.12.24



Date 02/28/25

Event Name Illinois V5RC HS State Championship

Innovate Award Submission Information Form

Instructions for team: Please fill out all information, printing clearly. This form should be included as per the instructions given in the [Guide to Judging](#). Teams may only submit **one** aspect of their design to be considered for this award at each event. Submission of multiple aspects will nullify the team's consideration for this award.

Full Team Number: 355X

Brief description of the novel aspect of the team's design:

Our team derived a formula that allows us to utilize 1 auton function for all 4 possible starting positions. This formula takes 3 inputs: color (red or blue), sign (positive corner or negative corner), and the default blue positive value. It converts the inputted default blue positive value to the value needed for the starting position that is specified by the other 2 inputs. This allows us to make easy auton adjustments to all 4 programs at once while the code remains clean and easy to read.

Identify the page numbers and/or the section(s) where documentation of the development of this aspect can be found:

Reflecting Auton Entries: 130-137

Initial Auton Entries (*Original Auton Without Formula*):

Explain why your submission is unique from other approaches to the problem it solves or task it performs:

Our approach is unique because although some teams may have similar autons for all 4 starting positions, they often copy/paste the program 4 separate times. This makes even the smallest edits incredibly time-consuming and creates unnecessary variation between all 4 programs. Our approach not only saves time programming but also ensures minimal variation in aspects like accuracy between all 4 autons.

How To Read Our Notebook



Define the Problem (DP)



Brainstorming Solutions (BS)



Select and Plan Solution (SP)



Build and Implement Solution (BI)



Test Solution (TS)



Time Management (TM)



Competition Analysis (CA)



General/Other (GO)

Title

Font is used for entry titles

Section Headings

Font is used for section headings

Important

Font is used for text that is especially important

Normal

Font is used for normal text

“Name” section at bottom of notebook is all the team members that attended the meeting the entry was based on

“Full Cycle” in the table of contents is a collection of links to the full design cycle of a project.

Table of Contents

(Please Click on Titles)

Date	Title	Color	Page
Please Read	<u>How To Read Our Notebook</u>	N/A	2
-----	<u>All Design Cycles</u> (All Design Processes We Went Through)	N/A	8
06/11/24	<u>Our Design Process</u>	GO	9
06/11/24	<u>Team Profile</u>	GO	10
06/11/24	<u>Season Goals</u>	GO	11
06/11/24	<u>Season Budgeting</u>	GO	12
06/11/24	<u>Gantt Chart 1</u>	TM	13
06/11/24	<u>June 2024 Calendar</u>	TM	14
06/11/24	<u>Why Switch Back to C++?</u>	GO	15
06/11/24	<u>Game Overview</u>	GO	16
06/11/24	<u>Game Scoring</u>	GO	17
06/11/24	<u>Games Rules</u>	GO	19
06/11/24	<u>Skills Rules</u>	GO	21
06/11/24	<u>Game Measurements</u>	GO	22
06/12/24	<u>Initial Thoughts/Strategy</u>	DP	24

Table of Contents

(Please Click on Titles)

Date	Title	Color	Page
06/12/24	Initial Drivetrain: Define Problem	DP	26
06/12/24	Initial Drivetrain: Brainstorm Solutions	BS	27
06/14/24	Building + Programming Pneumatics	BS	30
06/19/24	Initial Drivetrain: Select and Plan	SP	33
06/24/24	LemLib Drivetrain Setup: Define Problem	DP	38
06/24/24	LemLib Drivetrain Setup: Brainstorm Solutions	BS	39
06/24/24	LemLib Drivetrain Setup: Select and Plan	SP	42
06/26/24	Basic LemLib Commands	GO	46
07/01/24	July 2024 Calendar	TM	48
07/08/24	Initial Intake: Define Problem	DP	49
07/08/24	Initial Intake: Brainstorm Solutions	BS	50
07/10/24	Initial Mogo Mech: Define Problem	DP	52
07/10/24	Initial Mogo Mech: Brainstorm Solutions	BS	53
07/12/24	Initial Mogo Mech: Select and Plan	SP	55
08/01/24	August 2024 Calendar	TM	57
08/02/24	Initial Intake: Brainstorm Solutions (Part 2)	BS	58
08/02/24	Initial Intake: Select and Plan	SP	59

Table of Contents

(Please Click on Titles)

Date	Title	Color	Page
08/06/24	Initial Drivetrain: Build and Implement	BI	61
08/21/24	Initial Drivetrain: Test Solution	TS	64
08/30/24	Initial Intake: Select and Plan (Part 2)	SP	65
09/01/24	September 2024 Calendar	TM	66
09/13/24	Initial Intake: Build and Implement	BI	67
09/20/24	Initial Mogo Mech: Build and Implement	BI	69
10/01/24	October 2024 Calendar	TM	71
10/08/24	Initial Auton: Define the Problem	DP	72
10/08/24	Initial Auton: Brainstorming Solution	BS	73
10/08/24	Initial Auton: Select and Plan	SP	77
10/11/24	Beginning Programming Tweaks	GO	82
10/25/24	Initial Intake: Test Solution	TS	85
10/29/24	Initial Mogo Mech: Test Solution	TS	87
11/02/24	Competition 1: Lake Park Analysis	CA	89
11/05/24	November 2024 Calendar	TM	95
11/05/24	Gantt Chart 2	TM	96
11/06/24	Intake V2: Define the Problem	DP	97
11/06/24	Intake V2: Brainstorming Solutions	BS	98

Table of Contents

(Please Click on Titles)

Date	Title	Color	Page
11/06/24	Intake V2: Select and Plan	SP	100
11/06/24	Intake V2: Build and Implement	BI	101
11/15/24	Intake V2: Test Solution	TS	103
11/16/24	LemLib Drivetrain Setup: Build and Implement	BI	104
11/16/24	LemLib Drivetrain Setup: Test Solution	TS	108
11/22/24	Competition 2: Speedway Signature Event	CA	110
12/01/25	December 2024 Calendar	TM	117
12/26/24	Wall Stakes Mech: Define the Problem	DP	118
12/26/24	Wall Stakes Mech: Brainstorm Solutions	BS	119
12/31/24	Wall Stakes Mech: Select and Plan	SP	121
12/31/24	Initial Auton: Select and Plan (Part 2)	SP	123
01/01/25	January 2025 Calendar	TM	124
12/31/24	Initial Auton: Build and Implement	BI	125
12/31/24	Wall Stakes Mech: Build and Implement	BI	126
01/04/25	Initial Auton: Test Solution	TS	128
01/04/25	Wall Stakes Mech: Test Solution	TS	129
01/02/25	Reflecting Auton: Define the Problem	DP	130
01/02/25	Reflecting Auton: Brainstorm Solutions	BS	131

Table of Contents

(Please Click on Titles)

Date	Title	Color	Page
01/02/25	Reflecting Auton: Select and Plan	SP	132
01/02/25	Reflecting Auton: Build and Implement	BI	133
01/02/25	Reflecting Auton: Test Solution	TS	134
01/02/25	Reflecting Auton: Build and Implement (Part 2)	BI	135
01/02/25	Reflecting Auton: Test Solution (Part 2)	TS	137
01/05/25	Competition 3: Great Lakes	CA	138
02/01/25	February 2025 Calendar	TM	144
02/03/25	Skills Auton: Define the Problem	DP	145
02/03/25	Skills Auton: Brainstorm Solutions	BS	146
02/03/25	Skills Auton: Select and Plan	SP	148
02/04/25	Skills Auton: Build and Implement	BI	149
02/05/25	Overall Competition Analysis	CA	151
02/28/25	Illinois V5RC HS State Championship INNOVATE AWARD SUBMISSION	GO	154

ALL DESIGN CYCLES

(Please Click on Titles)

Date	Title	Page(s)
06/12/24 to 08/21/24	Initial Drivetrain: Define the Problem , Brainstorming Solutions , Select and Plan , Build and Implement , Testing Solution	26-27 33-37 61-64
06/24/24 to 11/16/24	LemLib Drivetrain Setup: Define the Problem , Brainstorm , Select and Plan , Build and Implement , Test Solution	38-45 104-107
07/8/24 to 10/25/24	Initial Intake: Define the Problem , Brainstorming Solutions , Select and Plan , Build and Implement , Testing Solution	49-50 58-60 65 67-68 85-86
07/10/24 to 10/29/24	Initial Mogo Mech: Define the Problem , Brainstorming Solutions , Select and Plan , Build and Implement , Testing Solution	52-56 69-70 87-88
10/08/24 to 01/0/24	Initial Auton: Define the Problem , Brainstorming Solutions , Select and Plan , Select and Plan Part 2 , Build and Implement	72-81 123 125 128
11/06/24 to 11/15/24	Intake V2: Define the Problem , Brainstorming Solutions , Select and Plan , Build and Implement , Test Solution	97-103
12/26/24 to 01/04/25	Wall Stakes Mech: Define the Problem , Brainstorming Solutions , Select and Plan , Build and Implement , Test Solution	118-121 126-127 129
01/02/25 to 01/04/25	Reflecting Auton: Define the Problem , Brainstorming Solutions , Select and Plan , Build and Implement , Test Solution , Build and Implement (Part 2) , Test Solution (Part 2)	130-137
02/03/25 to 02/07/25	Skills Auton: Define the Problem , Brainstorming Solutions , Select and Plan , Build and Implement	145-150

Our Design Process

1. Define the Problem

- a. Description with words and pictures
- b. Solution requirements and goals

2. Brainstorm Solutions

- a. Research (if applicable)
- b. Possible solutions with pros and cons
- c. Labeled diagrams

3. Select and Plan Solution

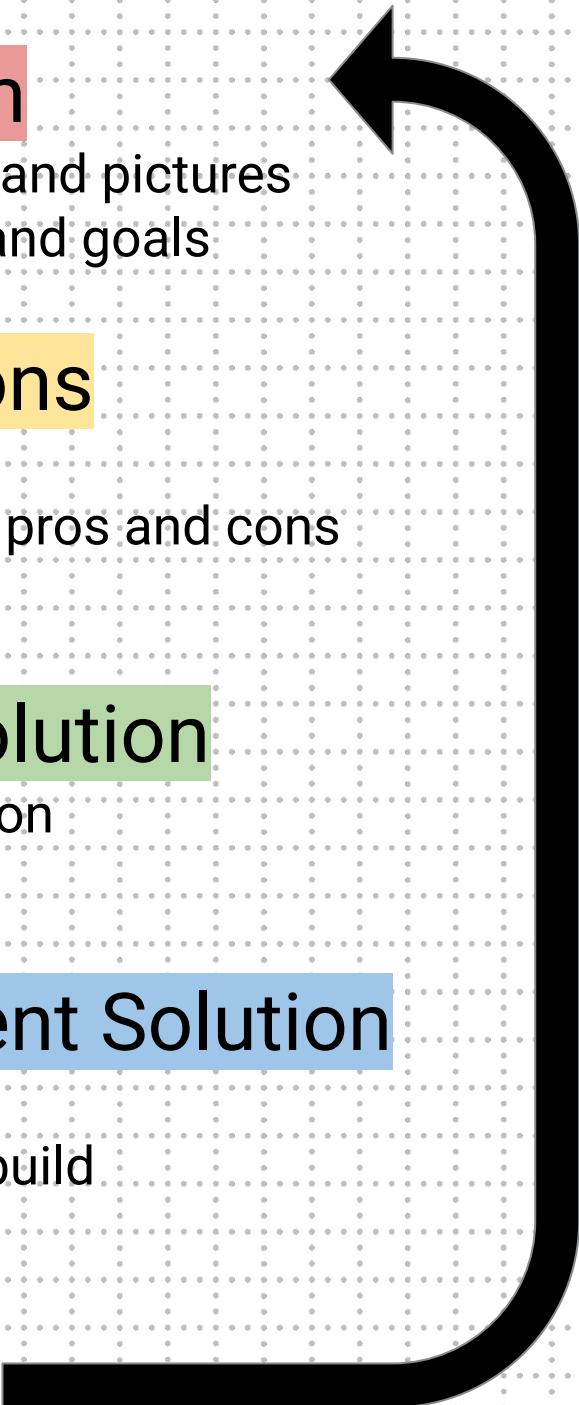
- a. Sketches/CAD of solution
- b. Decision matrix

4. Build and Implement Solution

- a. Steps to build solution
- b. Pictures of completed build

5. Test Solution

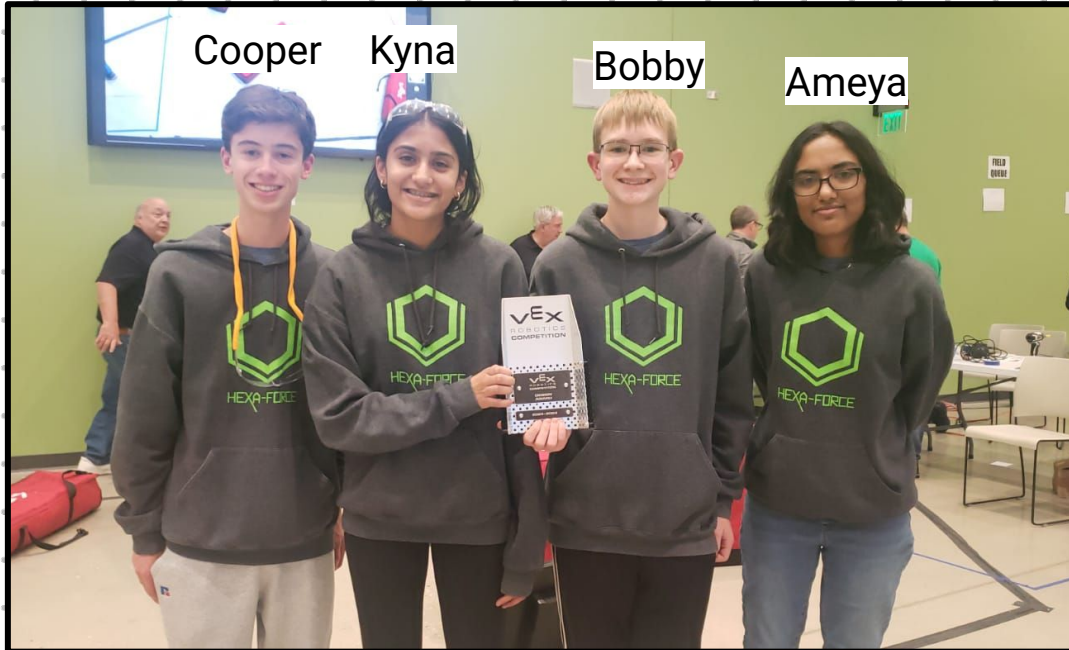
- a. Procedure for testing
- b. Data from testing
- c. Analysis of data/results
- d. Future plans



Robotics is centered around constantly improving and iterating, so this is the design cycle we utilize to create & modify our robot.

Team Profile

General/Other



Hello! We are a four-person VEX team, and this is our 3rd year of VEX! All of us are juniors from 2 different schools. We all have experience in building, programming, notebooking, and strategies, so we are all able to work together at all aspects of VEX.

We use a **digital** engineering notebook so that our entire team can contribute and access it anytime. Since it can be difficult for every team member to attend each meeting, we've chosen to utilize our engineering notebook as a tool to inform other teammates about what we accomplish.

Season Goals

General/Other

- Go to Signature Events (something we didn't do last year)
 - Understand and be exposed to the ever changing meta
 - Rank top 75%
- Network
 - Create reveals and post them on Instagram and YouTube
 - Meet new teams at competitions!
- Win an award at 75% of our regional competitions
- Rank top 40% in Skills at all of our regional competitions
- Go to State and win a Worlds qualifying award
- Go to Worlds and rank top 50% in our division



HEXA-FORCE

Season Budgeting

General/Other

Goal: Create a budget to optimize spending and fundraising for this year.

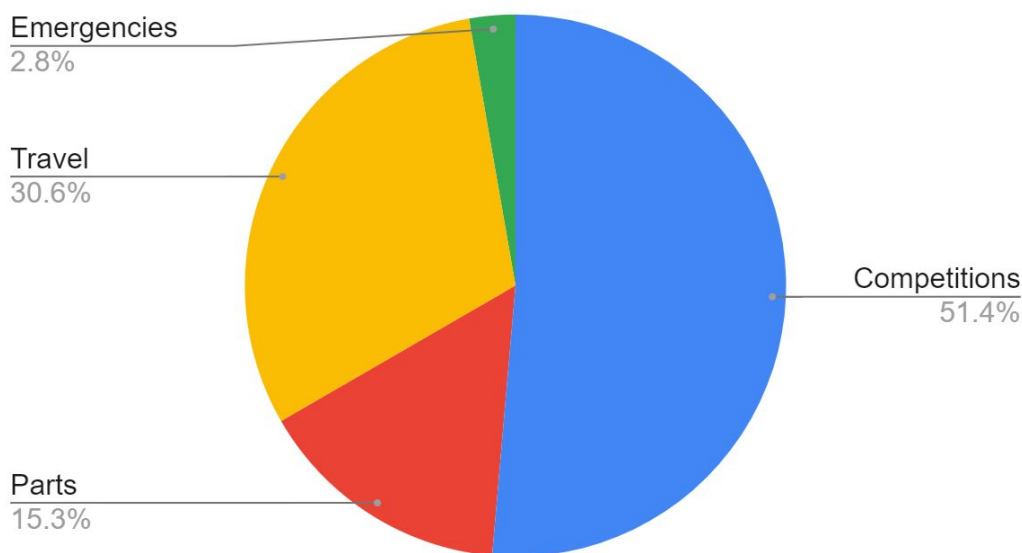
Item	Quantity	Cost
Parts	N/A	\$1,000
Regional Tournaments	7	\$700
Signature Events	2	\$685
State Championship	1	\$175
Worlds (Theoretical)	1	\$1,800
Emergency Fund	1	\$180
Travel	4	\$2,000

Season Total: \$6,540.00

Current Capital: \$3,600.00

Need from Fundraising/Sponsors: \$2,940.00

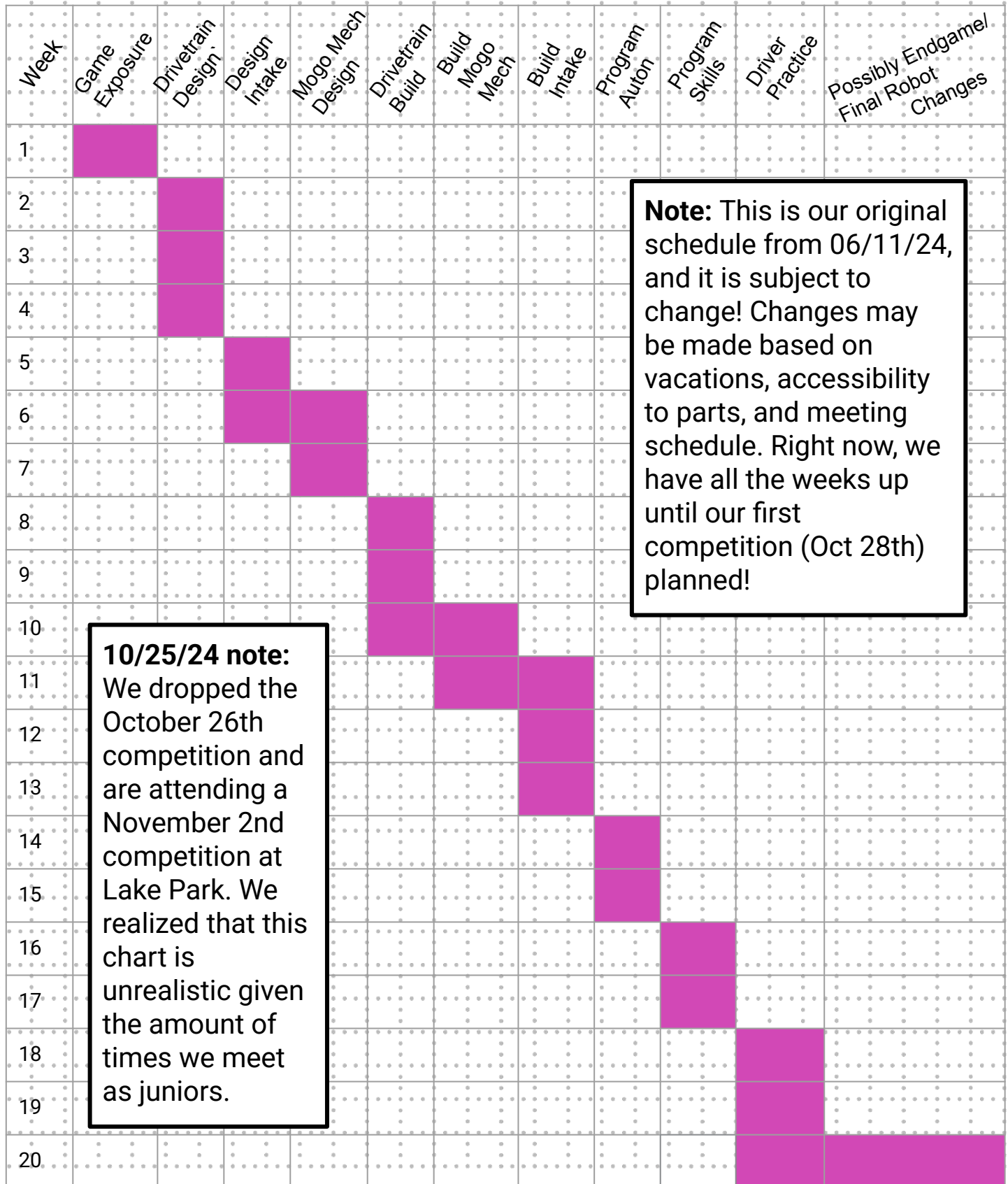
Breakdown of Spendings



Gantt Chart 1

Time Management

Goal: Plan out our project plans until our next competition (October 26th).



Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
						1 Zoom @ 9:30 am (Decide on Signature Events + Competitions)
2	3	4	5	6 Basement @ 1:00 pm (Take Apart Last Season's Robot + Order Testing Parts)	7	8
9	10	11	12 Library @ 1:00 pm (Initial Game Strategy + Design Ideas)	13	14 Basement @ 1:00 pm (Start Testing Pneumatics)	15
16	17 Library @ 4:30 pm (Finalize Drivetrain and Begin CAD)	18	19 Library @ 1:00 pm (Continue CAD)	20	21 Library @ 1:00 pm (Continue CAD)	22
23	24 Library @ 2:00 pm (Start Programming Drivetrain with LemLib)	25	26 Library @ 2:00 pm (Continue Programming Drivetrain with LemLib)	27	28	29
30		Upcoming Competitions: October 26th @ Glenbrook South November 2nd @ Lake Park November 22nd @ Speedway Signature Event				

Why Switch Back to C++?

General/Other

Last year, our team programmed in Python due to its greater versatility and real-world applications. We wanted to learn something and expand our toolbox in a way that would benefit us in the future.

This year we wanted to switch back to C++ like our first season because last year we encountered many useful features for our code that we were unable to implement because they were only offered through PROS which is currently only compatible with C++.

Some of the features we hope to implement with C++ this year include:

1. SD Card
 - a. The ability to replay a robot's actions that were performed during driver control, autonomously by storing specific values in the SD Card (Originally from Cukmekerb's Coding Class)
2. LemLib (Especially Pure Pursuit)
 - a. The ability to have a robot follow a path sketched electronically
3. PID
 - a. The ability to make a robot's movements more accurate and consistent



Game Overview

General/Other

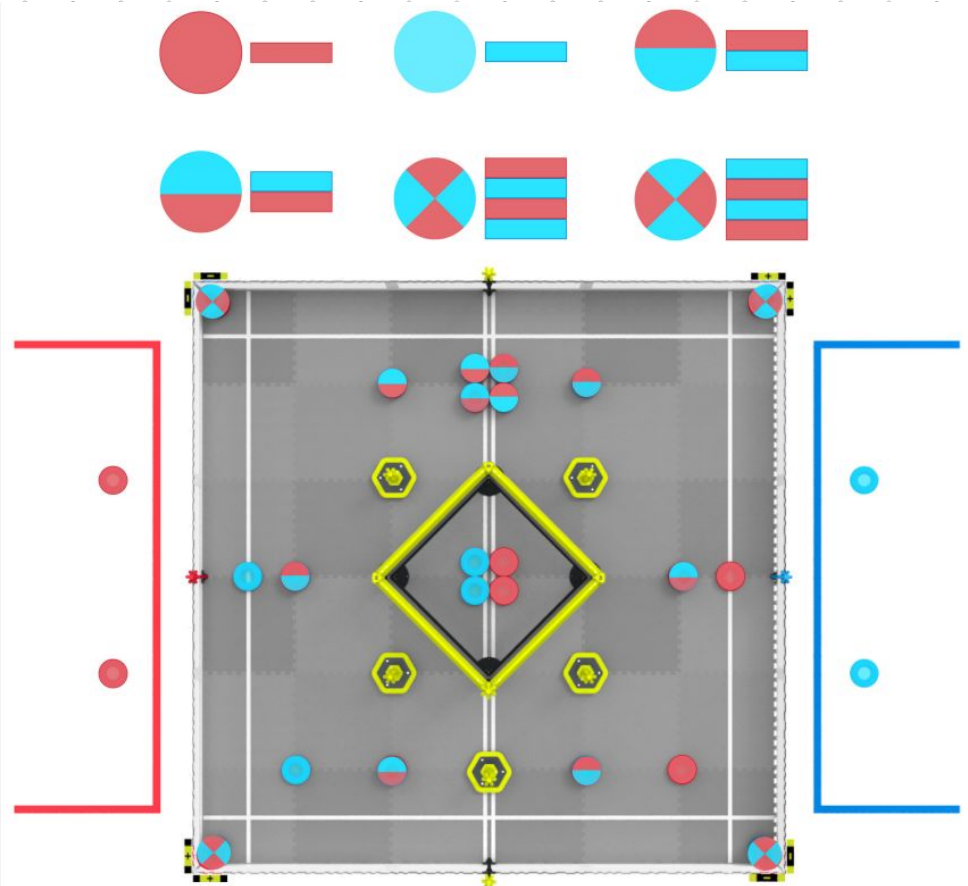
Goal: Familiarize ourselves with the game and its setup.

The competition "High Stakes" is played on a 12' by 12' field. Matches consist of two alliances: the blue alliance and the red alliance. Each alliance is composed of two teams that work to score and keep as many points as possible. Matches start with a 15 second autonomous period, followed by a 1 minute 45 second driver control period where the last 10 seconds are considered endgame.

The objective of the game is to end with a higher score than the opposing alliance by scoring rings on stakes, utilizing the positive and negative corners, and climbing the ladder at the end of the game.

This year's game setup:

- 5 Mobile Goals
 - 1 per alliance + 2 neutral
- 4 Wall Stakes
 - 1 per alliance + 2 neutral
- 1 Ladder
 - 3 levels + 1 high stake on top
- 48 Rings
 - 24 of each color
- 4 Corners
 - 2 positive and 2 negative

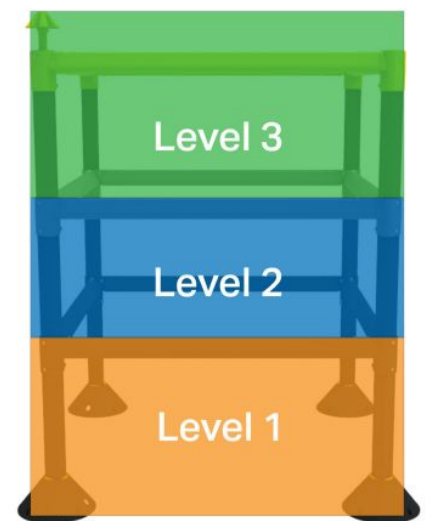


Game Scoring

General/Other

Goal: Familiarize ourselves with game scoring and scoring criteria (SCs).

- **Autonomous Bonus - 6 Points**
 - Climb points and corner modifiers are not included when calculating autonomous bonus
 - If autonomous period ends in tie, each alliance will receive an autonomous bonus of 3 points
 - Any rule violations, major or minor, during the autonomous period will result in the autonomous bonus being awarded to the other alliance. If both alliances violate rules during the autonomous period, no autonomous bonus will be awarded
- **Each Ring Scored on a Stake - 1 Point**
 - Ring is not contacting robot of same color alliance as ring
 - Ring is not contacting any gray foam tiles
 - Ring is “encircling” a stake
 - Stake is not exceeding the permitted number of rings
 - Mobile Goals - Fit 6 Rings Each
 - Alliance Wall Stakes - Fit 2 Rings Each
 - Neutral Wall Stakes - Fit 6 Rings Each
 - High Stake - Fits 1 Stake
- **Each Top Ring on a Stake - 3 Points**
 - The ring is scored on a stake
 - The ring is the furthest scored ring from the ground
 - No minimum number of rings scored in order
- **Climb**
 - Level 1 - 3 Points
 - Level 2 - 6 Points
 - Level 3 - 12 Points
 - Robot is contacting ladder
 - Robot is not contacting any other field elements, including foam tiles
 - Robot is not contacting mobile goals



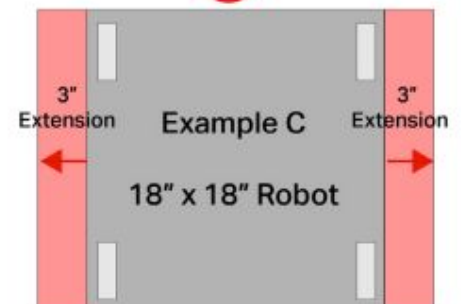
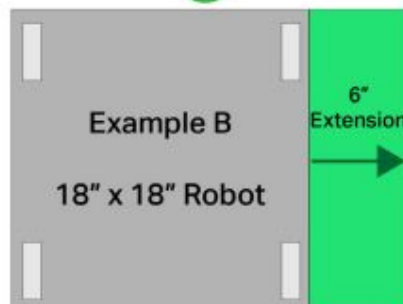
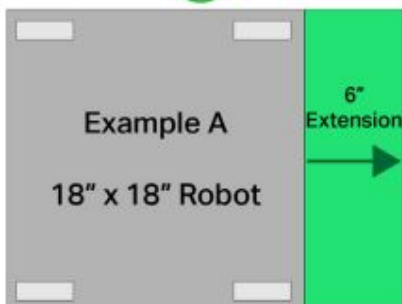
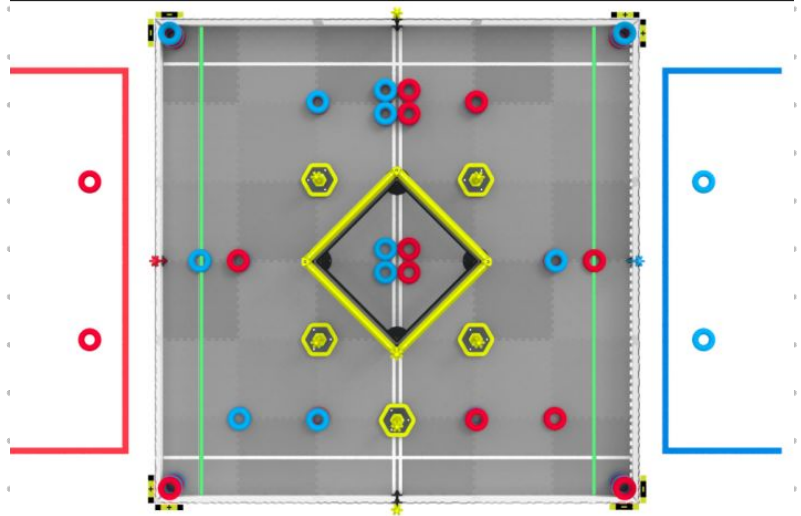
- Each levels' height corresponds to the top edge of a rung of the ladder. For example, a Level 1 Climb represents a robot whose lowest point is above the foam tiles, but not higher than the first rung of the ladder.
- **Corners**
 - Mobile Goal placed in Positive Corner
 - Value of Scored Rings is doubled
 - Mobile Goal Placed in Negative Corner
 - Value of Scored Rings is set to 0
 - For each ring, an equivalent amount of points will be removed from that alliance's other scored rings
 - The negator **only applies to "Ring Points"** and **cannot be applied to climbing or autonomous bonus**
 - Mobile goal's base is contacting the corner (the floor and/or white tape)
 - Mobile goal is "upright". "Upright" means there is no contact between the stake (and/or rings on this stake) and floor and floor perimeter
 - Contact with robot is irrelevant!!
- **Autonomous Win Point**
 - At least 3 scored rings
 - Minimum of 2 stakes, each with at least 1 ring scored
 - Neither robot is breaking/contacting the plane of the Starting Line
 - 1 robot contacting the ladder

Game Rules

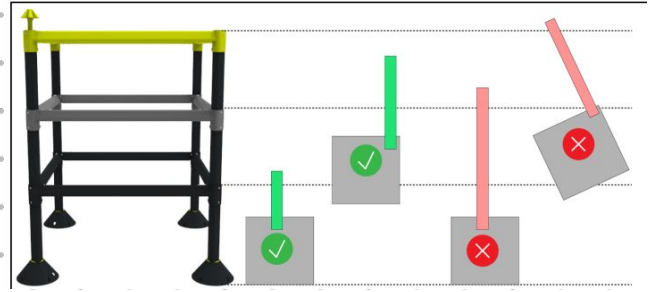
General/Other

Goal: Familiarize ourselves with specific game rules (SGs).

- Prior to the start of each match, the robot must be placed so it is:
 - Contacting/"breaking the plane" of their alliance's starting line (shown in green in the diagram below)
 - Not contacting any scoring objects other than the 1 preload
 - Not contacting any other robots
 - Completely stationary
- Horizontal Expansion is Limited
 - Robot may never exceed an overall footprint of 24" x 18"
 - Robot can only expand in one "X/Y" direction (from a single "side" of the robot)



- Vertical Expansion is Limited
 - Robot may never be breaking the plane of more than 2 levels of the ladder
 - So... height limit is 32"
 - Vertical limit is measured by perspective of field, it does not "rotate with the robot"



- Keep Rings and Stakes on field - don't throw them off the playing area
- Each Robot Gets 1 Preload
 - Must be placed contacting one robot of the same alliance color as the preload
 - Must be placed so it is not contacting the same robot as another preload
 - Must be placed in a non-scored location
- Possession is Limited to 2 Rings and/or 1 Mobile Goal
- Don't Cross Autonomous Line During Auton
 - Scoring objects and wall stakes that contact or are positioned above the autonomous line are able to be used by either alliance during auton
- Engage with Autonomous Line (scoring objects and wall stakes that are ON it) at Your Own Risk
- Alliance Wall Stakes (colored stakes) are protected
 - Robot should not interact with opposing alliance's wall stake

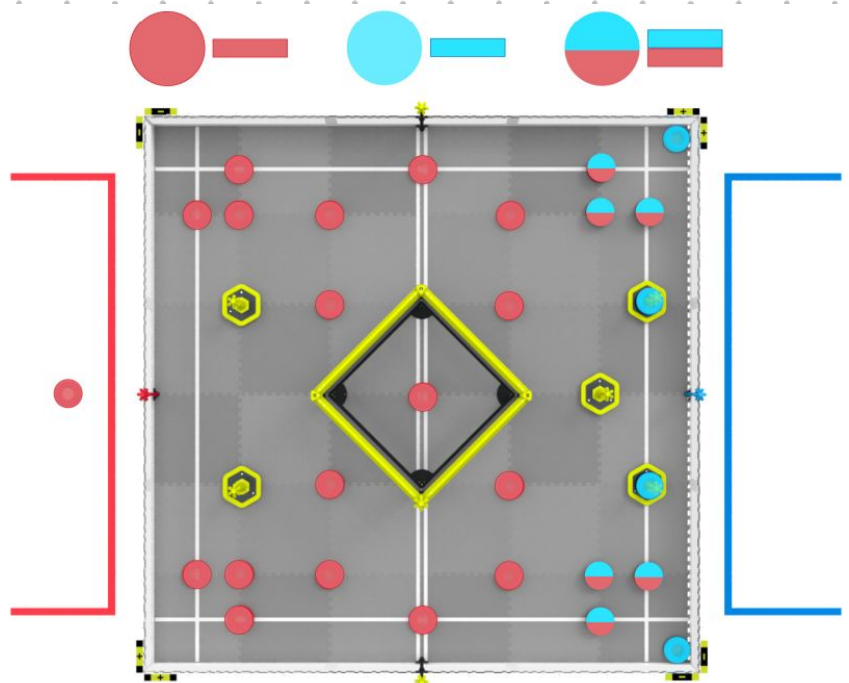
Skills Rules

General/Other

Goal: Familiarize ourselves with the skills rules and set-up.

- All Driver Control and Autonomous Skills Matches are 1 Minute
- Each Team Gets 3 Attempts at Both Driver Control and Autonomous Skills
- Robot Must Start in a Legal Starting Spot for the Red Alliance
 - Teams may also use the 1 red alliance preload
- Blue Rings May Only Be Scored As Top Rings on Stakes
 - Each blue ring only has a point value if
 - All red rings in the match have been scored on stakes and have point values
 - At least 1 red ring is scored below that blue ring on that stake
 - There is only 1 blue ring on that stake
 - No red rings are scored above the blue ring on that stake
- Any Red Ring Scored Above a Blue Ring on the Same Stake Does Not Have a Point Value
- No Corner Modifications

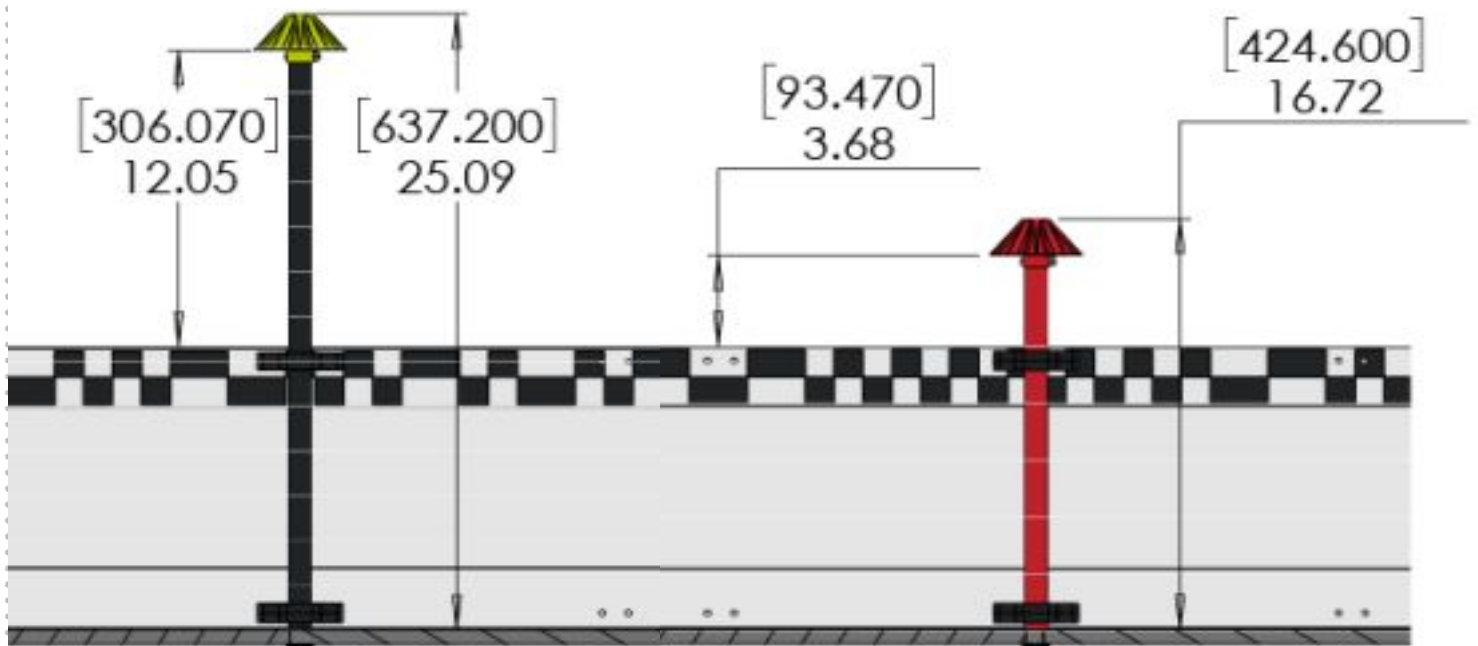
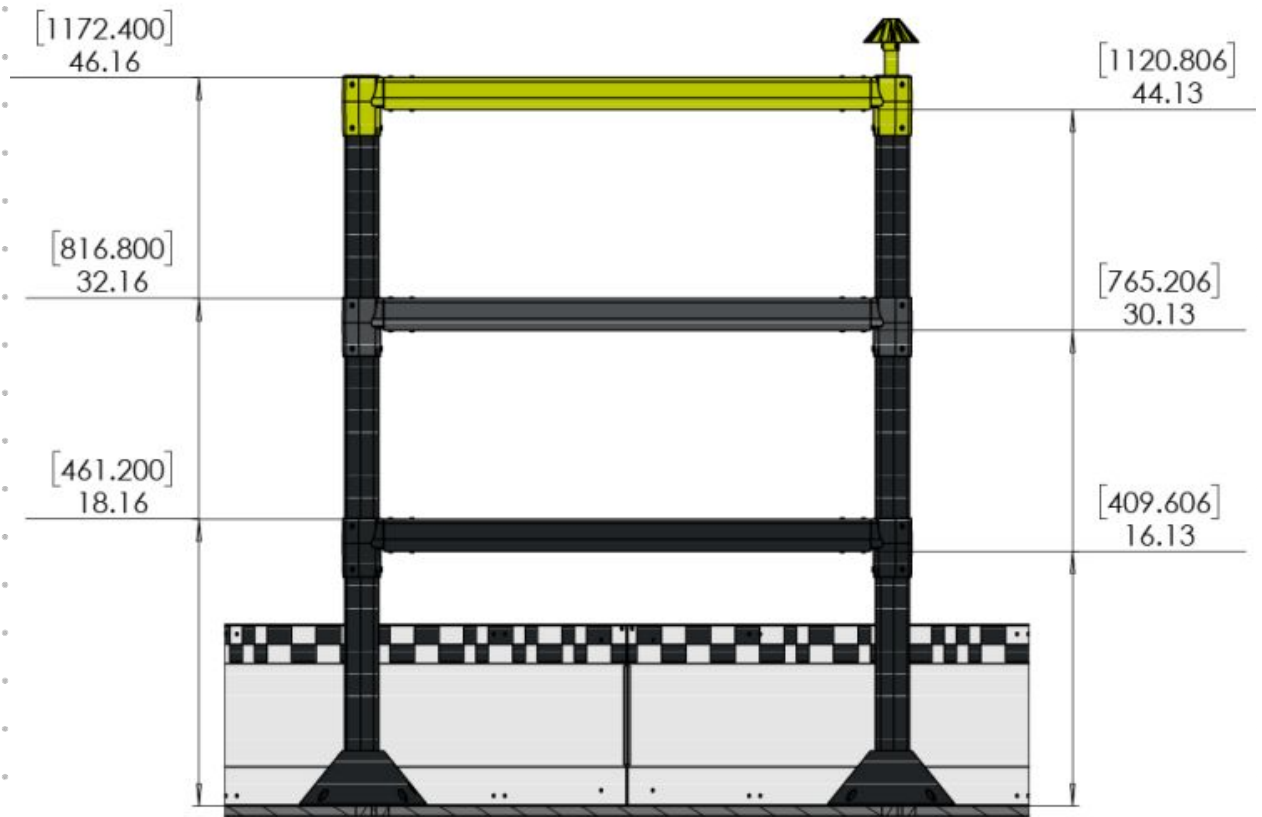
This year's skills setup includes all 24 red rings and 5 mobile goals but not all blue rings.

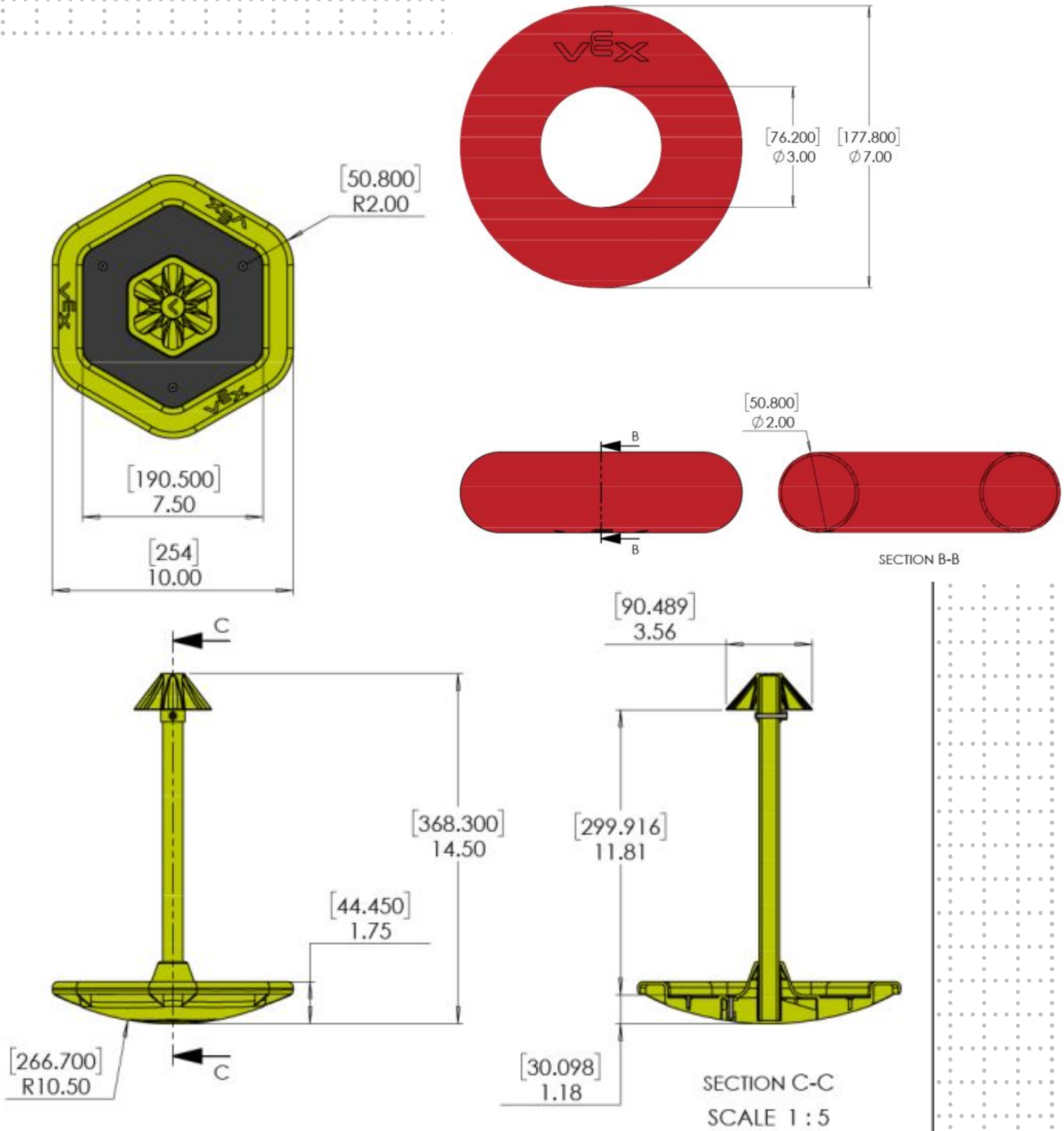


Game Measurements

General/Other

Goal: Familiarize ourselves with the game measurements.



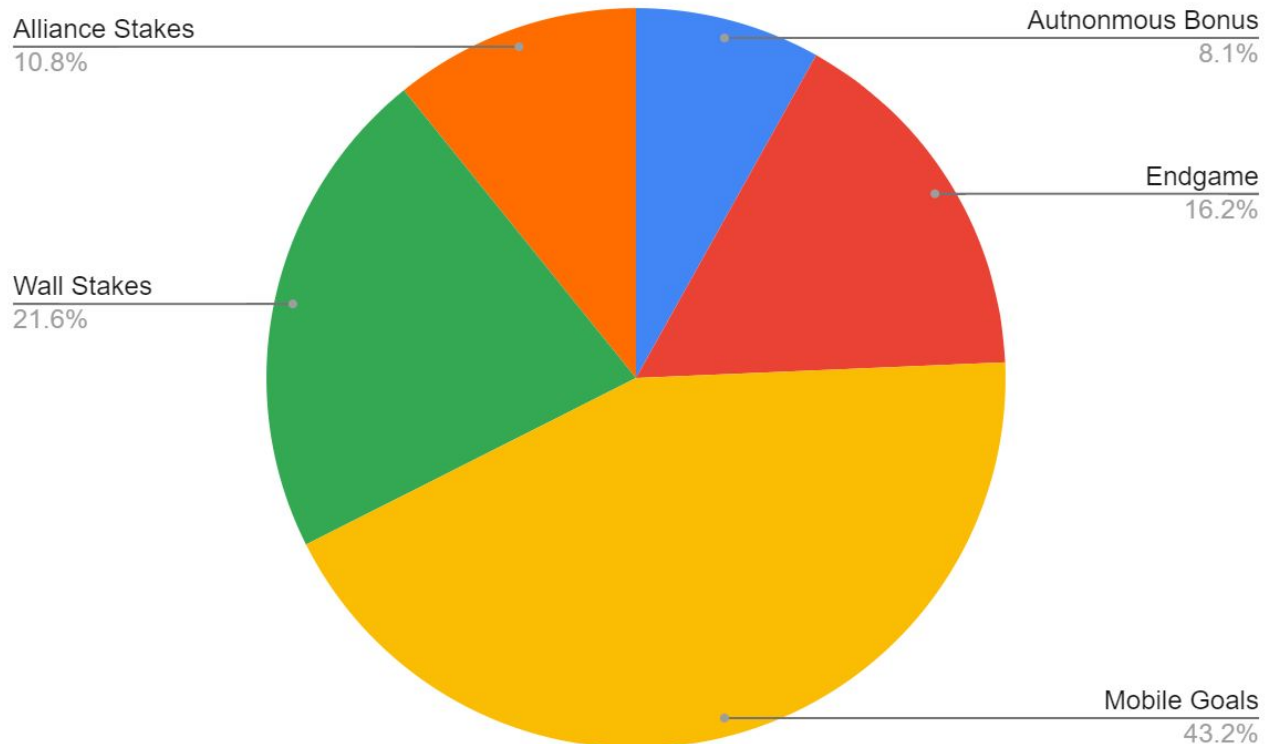


Initial Thoughts/Strategy

Define the Problem

Goal: Discuss and document our first thoughts as well as possible strategies.

Breakdown of Total Scorable Points:
(excluding positive and negative corners)



Initial Thoughts/Reactions:

- After looking at the breakdown of the scores, we saw many of the points come from the mobile goals, so it is important to have a strategy and mechanism that is able to reliably score the mobile goals.
- Negative Corners are not worth it
 - Positive corners will double your own score and negative corners will only take off half of what you could have earned for yourself from the other team's score

Define the Problem

- Mobile goals can easily be descored by knocking them over since the rings would touch the tiles
- AWP should 100% be programmed especially because it can be done solo
- Corners will most likely be puppy guarded
- Ring on top of ladder is not worth it because it is still only 1 point
- Alliance stakes can be done last since they cannot be descored (unless we want to do it during auton since we start right next to them)
- Try to maximize the number of stakes scored because the top ring on every stake is an extra 2 points

Initial Drivetrain

Define the Problem

Goal: Define the requirements and criteria for our drivetrain this year.

Problem Statement: We need a drivetrain so we are able to drive around the field and interact with the game pieces. Having a good drivetrain early on is important as it is the core and base of your robot. Changes made to a drivetrain in the middle of a season have the potential to affect the function of other mechanisms, so we want to have a good drivetrain now, rather than having a mediocre one and changing it in the middle of the season..

Solution Requirements:

- Must fit in 18" x 18" x 18"
- Must use legal VEX V5 Competition parts

Solution Goals:

- Be able to move quickly and efficiently across the field
- Be able to resist pushing from other robots - this year will involve a lot of fighting over the positive and negative corners
- Have enough motors left over to complete other tasks on the field
- Strong and robust
- Optimal weight (not too heavy, not too light)
 - For hang and withstanding pushing

Initial Drivetrain

Brainstorming Solutions

Goal: Brainstorm ideas for an effective drivetrain.

Possible Solutions - Number of Motors:

- 4 Motors

Positives	Negatives
<ul style="list-style-type: none">→ 44W leftover to use for other mechanisms→ Flexibility with positioning	<ul style="list-style-type: none">→ Not much torque→ We will be pushed around



x4

- 6 Motors

Positives	Negatives
<ul style="list-style-type: none">→ Increased torque→ Ability to fully gear up both sides	<ul style="list-style-type: none">→ Only 22W leftover to use for other mechanisms



x6

- 8 Motors

Positives	Negatives
<ul style="list-style-type: none">→ A very strong and fast drivetrain that can take any robot on the field	<ul style="list-style-type: none">→ 0W leftover to use for other mechanisms; Must use pneumatics for everything else→ Requires lots of space to place 8 motors on the base of the robot



x8

Possible Solutions - Motor Cartridges:

• Red Cartridge

Positives	Negatives
→ Very high torque (Ability to push other robots and withhold pushing)	→ Very low speed, which is a huge problem

Spins up to 100 RPM



• Green Cartridge

Positives	Negatives
→ Default cartridge → Good balance between speed and torque	→ Has neither high torque or high speed; no specialization → Slower than all the other teams; we won't be able to keep up with them

Spins up to 200 RPM



• Blue Cartridge

Positives	Negatives
→ Very fast - we put priority on speed → What most teams use: we will be able to keep up with the other robots	→ Very low torque (Easily pushed around) → Heats up fast so would need cooling system to avoid motor burnout

Spins up to 600 RPM



Possible Solutions - Wheel Sizes:

- 4" Omni + Traction Wheels (Four wheels)

Positives	Negatives
<ul style="list-style-type: none"> → Fast → Can use less RPM 	<ul style="list-style-type: none"> → Bigger - takes up more room

4" in Diameter



- 4 3.25" Omni + 2 Traction Wheels (Six Wheels)

Positives	Negatives
<ul style="list-style-type: none"> → Fast → Flexibility with number of wheels 	<ul style="list-style-type: none"> → Less traction

3.25" in Diameter



- 2 2.75" Omni+2 Traction Wheels (Eight Wheels)

Positives	Negatives
<ul style="list-style-type: none"> → Good traction → Compact → Fast 	<ul style="list-style-type: none"> → Heavier → More gears → Need more RPM

2.75" in Diameter



Building + Programming Pneumatics

Brainstorming Solutions

Goal: Explore ways to add motion to our robot without using motors. For example, we want to experiment with pneumatics and create a basic working system using the contents of a single pneumatics kit.

General Rules when Working with Pneumatics:

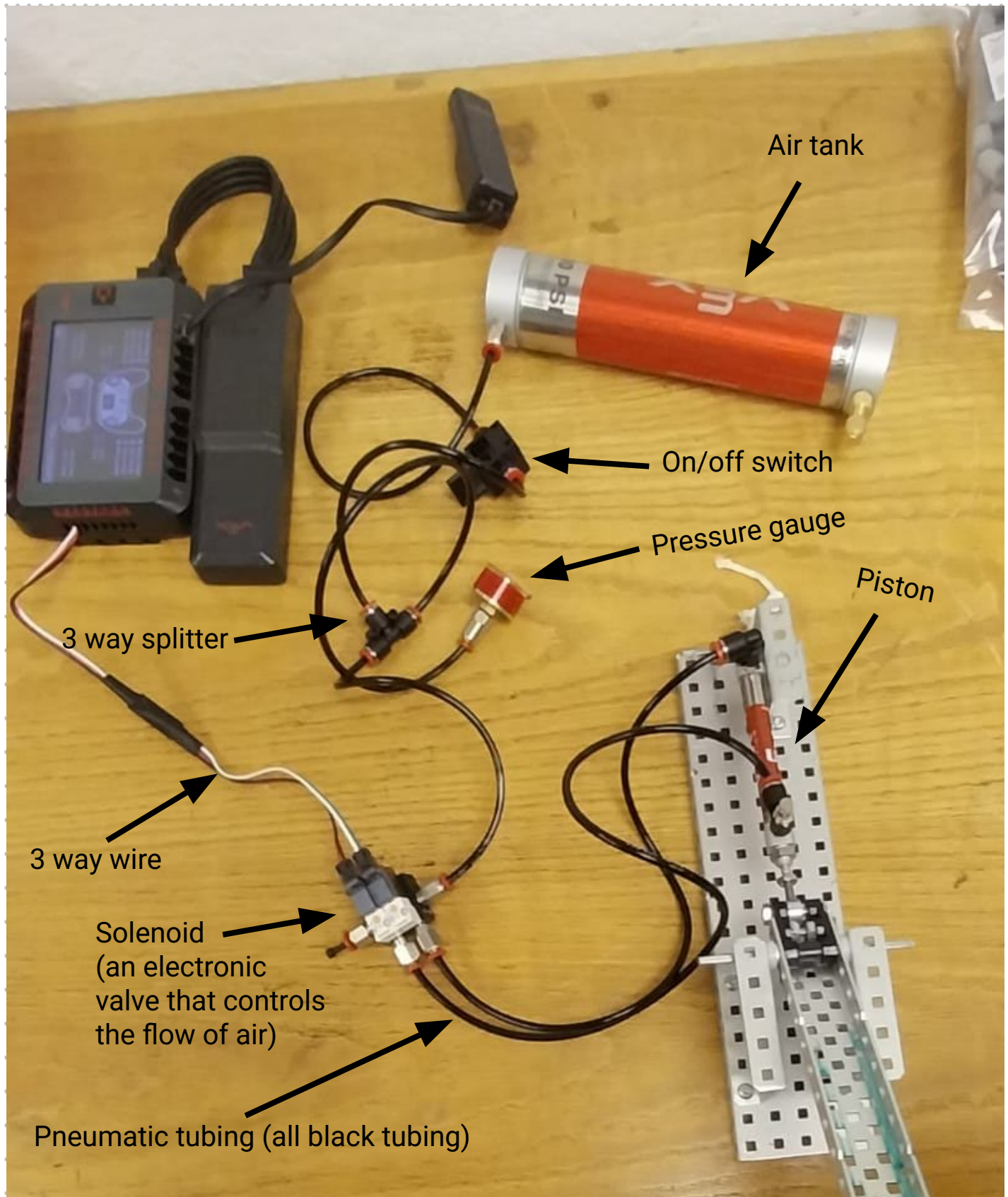
- Don't change or tweak the system until the air tank is empty
- The larger the piston, the more air used
- Each air tank can be filled to a maximum of 100 psi

Why Pneumatics?:

Since we have a six motor drivetrain, we only have two other motors for other mechanisms. Pneumatics essentially give us extra “motors”, allowing us to have more mechanisms to complete tasks on the field.

Building:

1. Add fittings to air tank, on/off switch, 3 way splitter, pressure gauge, solenoid, and piston
2. Add pneumatic tubing to connect air tank to on/off switch
3. Add pneumatic tubing to connect on/off switch to one part of the 3 way splitter
4. Add pneumatic tubing to connect 3 way splitter and pressure gauge
5. Add pneumatic tubing to connect 3 way splitter with the side of the solenoid and place a plunger into the fitting on the other side of the solenoid
6. Add pneumatic tubing to connect port A of solenoid to bottom of piston
7. Add pneumatic tubing to connect port B of solenoid to top of piston
 - a. One pushes the other pulls the piston



Pneumatics Code:

```
#include "main.h"
#define DIGITAL_SENSOR_PORT 'B'

pros::ADIDigitalOut piston(DIGITAL_SENSOR_PORT);
```

← Initializing
the
pneumatics
code

```
void autonomous()
{
    pros::lcd::set_text(2, "To get a 5 or not to get a 5 on the AP Exam");
    //above code is testing if our controller connects

    for(int i = 0; i < 10; i++) //a for-loop (Sets i to 0, as long as i is less than 10
    {                           //add one to i's current value each loop)
        piston.set_value(true); //moves piston out
        pros::delay(1000);      //waits for 1000msec
        piston.set_value(false); //moves piston in
        pros::delay(1000);
    }
}
```

- set_value() controls the state of the piston.
 - true means it is out
 - false means it is retracted
- Pros::delay is to add a wait after pushing and retracting
- We use a for loop to repeat the pushing and retracting process 10 times

Initial Drivetrain

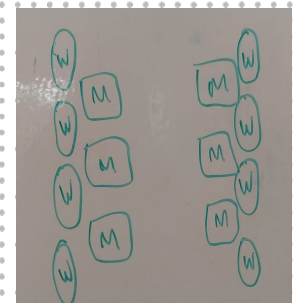
Select and Plan

Goal: Select a drivetrain option and CAD the design.

Decision Matrix for Number of Motors:

Design	Strength - x2 because strength is especially important to withstand pushing	Space - space taken up	Materials - amount of watts left	Total
4 Motors	1 The more motors the stronger our robot is	4 Less motors takes up less space	4 Still have 44W left for other mechs	10
6 Motors	3 The more motors the stronger our robot is	3 Less motors takes up less space	3 Still have 22W left for other mechs	12
8 Motors	4 The more motors the stronger our robot is	2 Less motors takes up less space	1 No motors left for other mechs	11

Why 6 motors? We chose to use a 6 motor drivetrain because it still has enough torque to withstand pushing and push other robots which is especially important in a game like High Stakes where the corners are going to be guarded and stakes may be fought over. Even with a high torque, we are still have 22W leftover to use for other mechanisms.



W = Wheel
M = Motor

Decision Matrix for Gear Cartridges:

Design	Strength (To resist pushing)	Speed (To move quickly)	Total
Red	4 Most Torque	1 100 RPM	5
Green	3 Middle Torque	2 200 RPM	5
Blue	2 Least Torque	4 600 RPM	6

Why blue cartridges? We chose to use blue cartridges because they are the fastest, and in an open field like High Stakes, speed is crucial. Although they have the least amount torque, we can gear them down to not only increase maneuverability but also increase torque. Even with less torque, our traction wheels should help us withstand pushing, and our speed should help us avoid being pushed in the first place.



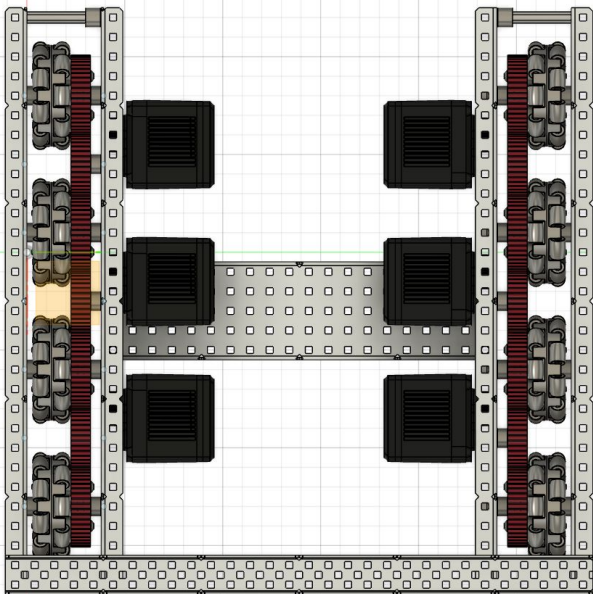
Decision Matrix for Wheel Size:

Design	2.75 inch wheels	3.25 Inch wheels
Speed - to avoid getting pushed	Equal Speed	Equal Speed
Ability to Withstand Pushing - very push heavy game this year.	4 - 2.75" wheels have more traction than 3.25" wheels. Also, if we use 2.75" wheels we have four traction wheels	3 - 3.25" wheels have less traction than the 2.75" wheels. If we use 3.25" wheels, we also only use 2 traction wheels
Robustness - we don't want our robot to break	4 - We get many places to place horizontal supports	3 - The bigger size gives us fewer places to place supports
Weight - for when we eventually climb	3 - With this build, we would be 456 grams $(70 * 2 + 44 * 2) * 2$	4 - With this build we would be 356 grams $(60 * 2 + 58) * 2$
Smaller Size - to fit in other mechanism	4 - We would be smaller than the 3.25" wheels	3 - We would be larger than the 2.75" wheels
Low Center of Gravity - to prevent tipping	4 - A lower center of gravity as the robot is lower to the ground	3 - A higher center of gravity than the 2.75" wheels
Total	19	16

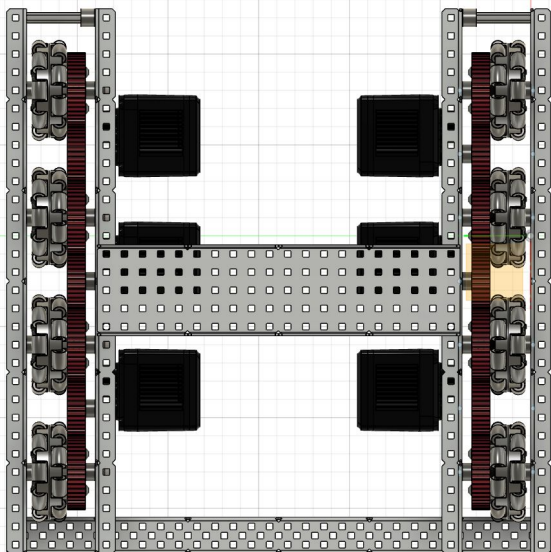
CAD Images of Drivetrain:

Note: Middle two wheels on both sides are going to be traction wheels!!

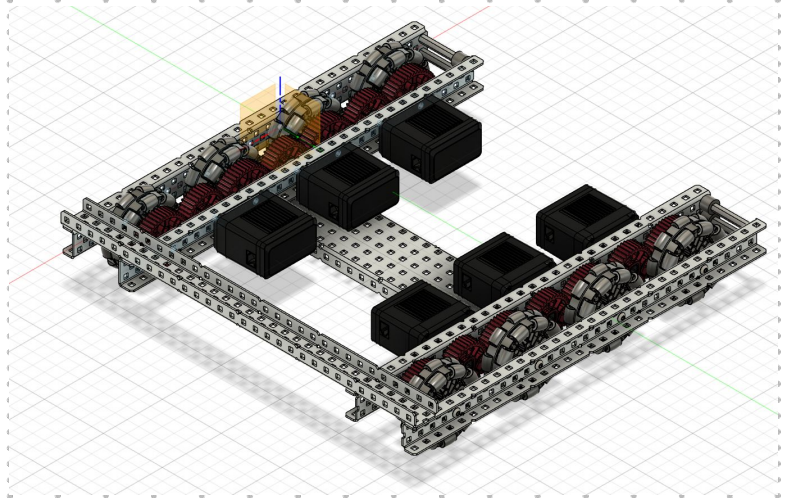
Top View



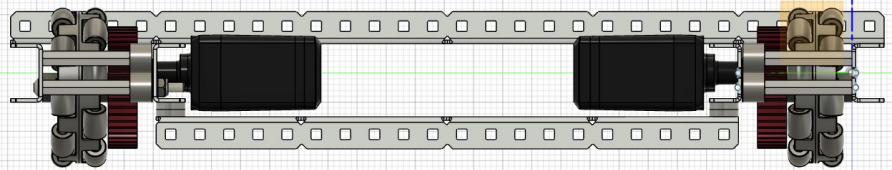
Bottom View



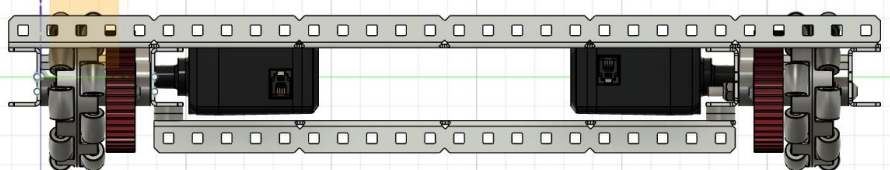
Angled View



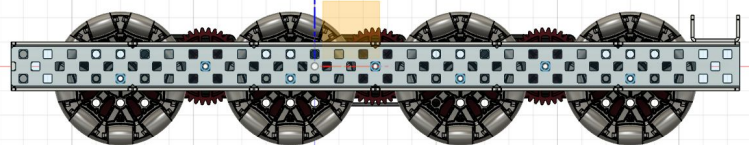
Front View



Back View



Side View



Plan for Our 6 Motor 450 RPM on 2.75" Omni + Traction Wheel Drivetrain:

- Screw joints for wheels to create less friction so wheels can spin more smoothly
- Rubberbanding motors in order to hotswap cartridges
 - Ensures our motors don't overheat and we can run back-to-back matches efficiently
- 2 outside wheels are omni while 2 inner are traction to create a good mix of speed and traction
- 6 motors (pg. 30 for more info)
- All blue cartridges (pg 31. for more info), but we geared it DOWN to 450 RPM with 36T and 48T gears
- 2.75" wheels (pg. 32 for more info)

LemLib Drivetrain Setup

Define the Problem

06-24-24: We took a break from CADing drivetrain to start learning LemLib because our main CADder was on vacation and we did not have access to the CAD (we use a school account).

Goal: Although we are currently in the process of CADing our drivetrain, we thought it would be beneficial to split up and have some people CAD the drivetrain while others began programming it in LemLib (something we are new to this year). LemLib is a C++ library that can be used in PROS to decrease coding time and enhance robot accuracy by utilizing Pure Pursuit and other unique features. We used these [LemLib tutorials](#) (linked) to help us get started.

(Full LemLib setup will be explained in Building and Implementing Step)

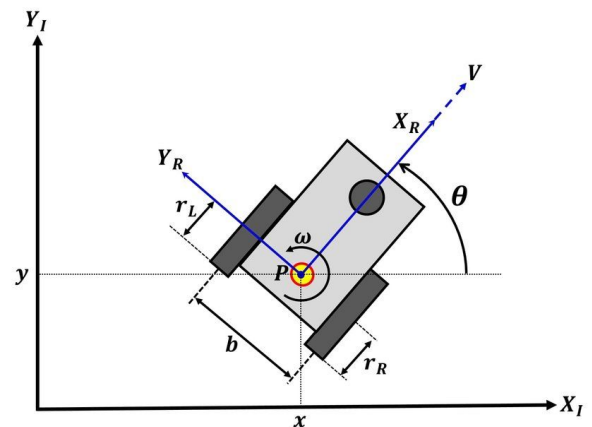
Problem Statement: LemLib uses odometry to make the robot's movements more accurate and consistent. We need to decide on the best sensors to use for this.

Solution Requirements:

- Must use legal VEX V5 Competition parts

Solution Goals:

- Heading tracking
- Vertical position tracking
- Horizontal position tracking



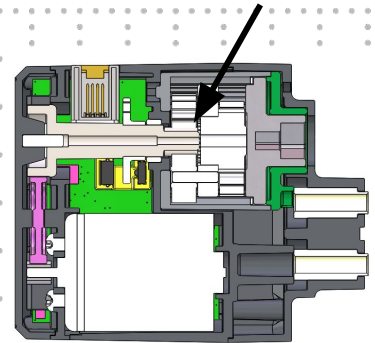
LemLib Drivetrain Setup

Goal: Brainstorm ideas for an effective odometry system

Possible Solutions - Heading Tracking:

- IME (Internal Motor Encoders)
 - Measures the amount of rotation on the shaft socket inside motor

Positives	Negatives
→ No extra mechanisms or installations	→ Not very accurate → Prone to wheel slippage and any inaccuracies outside the motor



- IMU (V5 Inertial Sensor)
 - Detects change in motion (acceleration) of robot

Positives	Negatives
→ Very accurate	→ Learning curve



- 2 Parallel Tracking Wheels
 - Wheels not included in the actual drivetrain that have a rotational sensors attached to them

Positives	Negatives
→ Medium accurate	→ Pricy - would need to buy two 2" inch omni wheels and two rotational sensors (~\$100)



Possible Solutions - Vertical Tracking:

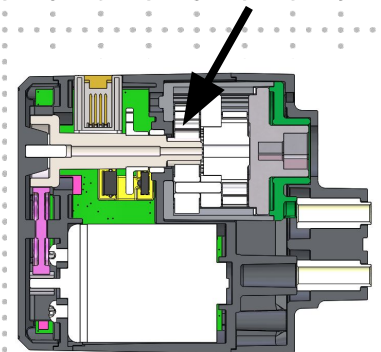
- 1 Vertical Tracking Wheel
 - Wheel not included in the actual drivetrain that has a rotational sensor attached to it (vertical - mounted parallel with drivetrain)

Positives	Negatives
→ Accurate	→ Pricy (~\$60) → Mounting - needs to be mounted well and far away from center of robot as possible



- IME (Internal Motor Encoder)
 - Measures the amount of rotation on the shaft socket inside motor

Positives	Negatives
→ No extra mechanisms or installations	→ Not accurate → Prone to wheel slippage and any inaccuracies outside the motor



- 2 Vertical Tracking Wheels

Positives	Negatives
→ Most accurate vertical tracking	→ More expensive than 1 tracking wheel → Takes up more space



Possible Solutions - Horizontal Tracking:

- 1 Horizontal Tracking Wheel
 - Wheel not included in the actual drivetrain that has a rotational sensor attached to it (horizontal - mounted perpendicular with drivetrain)

Positives	Negatives
→ Accurate	→ Pricy (~\$60) → Mounting - needs to be mounted well and far away from center of robot as possible



- Traction Wheels
 - Having traction wheels in drivetrain to prevent substantial horizontal drift

Positives	Negatives
→ We already have this implemented	→ Still allows some horizontal drift → No way to measure drift if it does happen so no way to correct it



- Add Mecanum Wheels
 - Having mecanum wheels in drivetrain to prevent substantial horizontal drift

Positives	Negatives
→ We have experience using them	→ Bulky → Would have to change current design



LemLib Drivetrain Setup

Select and Plan

Goal: Select odometry strategy for heading tracking, vertical tracking, and horizontal tracking.

Decision Matrix for Heading Tracking:

Design	Accuracy (x2 because the whole point of odometry is to be accurate and consistent)	Cost (Less money spent is better)	Space Used - we want to use as little space as possible on the drivetrain	Total
Internal Motor Encoder (IME)	$1 \times 2 = 2$ Not accurate	4 Already built into motors	4 Less parts take up less space	9
Inertial Sensor (IMU)	$4 \times 2 = 8$ Very accurate	2 \$52.49	3 Minimum space, only need sensor	13
Two Parallel Tracking Wheels	$3 \times 2 = 6$ Medium accurate	1 \$22.89 (4 pack of omni-wheels) + \$41.79 * 2 (2 rotational sensors) = \$106.47	1 Lots of space, need room for 2 wheels and 2 sensors	8

Decision Matrix for Vertical Tracking Options:

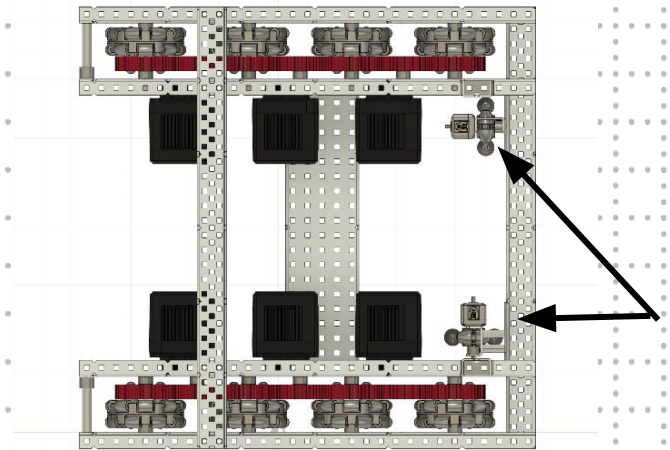
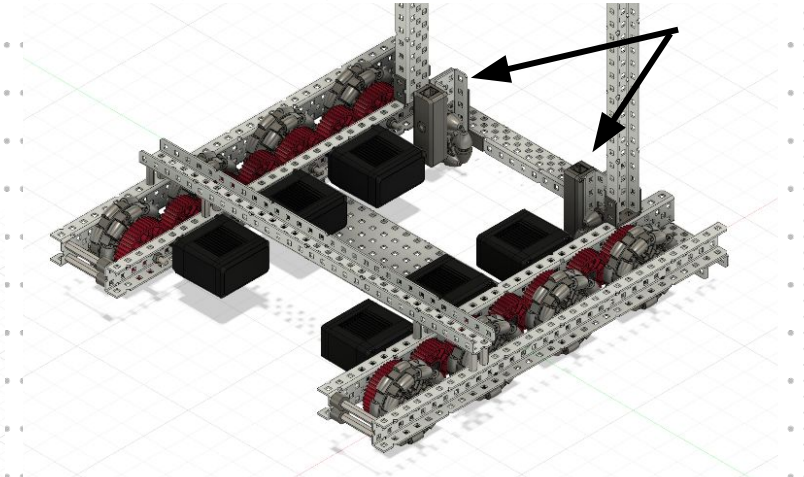
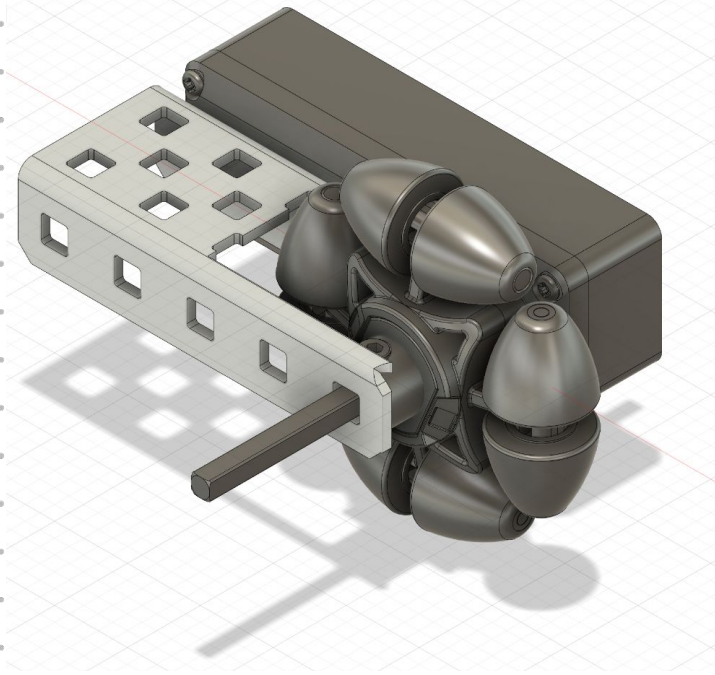
Design	Accuracy (x2 because the whole point of odometry is to be accurate and consistent)	Cost (Less money spent is better)	Space Used (We want to use as little space as possible on the drivetrain)	Total
One Vertical Tracking Wheel	3 x 2 = 6 More consistent than the IME's but less consistent than two tracking wheels	3 \$22.89 (4 pack of omni wheels) + \$41.79 (rotational sensors) = \$64.68	3 Less wheels mean more space	12
Internal Motor Encoder (IME)	1 x 2 = 2 The default option; Not very accurate at all	4 These are the default version, and we don't need to buy anything	4 No tracking wheels needed, no space used	10
Two Vertical Tracking Wheels	4 x 2 = 8 More consistent than both the IME's and one tracking wheel alone	1 \$22.89 (4 pack of omni wheels) + \$41.79 x 2 (2 rotational sensors) = \$106.47	1 More wheels used mean less space left	10

Decision Matrix for Horizontal Tracking Options:

Design	Accuracy (x2 because the whole point of odometry is to be accurate and consistent)	Cost (Less money spent is better)	Space Used (We want to use as little space as possible on the drivetrain)	Total
One Horizontal Tracking Wheel	4 x 2 = 8 The most accurate option of the three	3 We have to buy rotational sensors	3 We will have to make space, but only a little	14
Traction Wheels	3 x 2 = 6 While they will have less slippage, they will not be as accurate as a horizontal tracking wheel	3 We will use them for our drivetrain anyways, so not a huge loss	4 They will be on our drivetrain anyways, so no need to make space for them	13
Mecanum Wheels	1 x 2 = 2 We worked with these last year - mecanum wheels slide a lot, leading to less accuracy	4 We have these already	1 Takes up excessive space on the robot	7

CAD Images:

Note: We have not CADED the Inertial Sensor into our actual robot yet because it depends on where we have room after adding our other mechanisms.



Basic LemLib Commands

General/Other

Goal: Learn basic LemLib commands.

Note: Timeout parameter is the maximum amount of time the robot can take to perform the action. If timeout time is exceeded, robot will move on to next action.

setPose(x, y, z) - robot thinks it is at (x, y) with a heading of z degrees

turnToHeading(x, y) - Targets a heading on the field and point turns towards it, regardless of initial direction. Only turns while staying in place. Turn to face x degrees w/ a timeout of y msec

turnToPoint(x, y, z) - Same as above, but targets Cartesian coordinates. Turn to (x, y) with a timeout of z msec

swingToHeading(x, y) - Same as turnToHeading() but swing turn; not able to change amount of swing (only one side of drivetrain moves); swing to face x degrees w/ an timeout of y msec

swingToPoint(x, y, z) - Same as above, but targets a point instead of a heading; Not able to change amount of swing (only one side of drivetrain moves). Swing to face (x, y) with a timeout of z msec

moveToPoint(x, y, z) - Move to (x, y) on the field with a timeout of z msec. After initially turning to face it, angle stays the same.

moveToPose(x, y, z, t) - Same as moveToPoint, but also includes ending angle (z). Swing turns to face the angle by the time it reaches the end. Timeout is t msec

cancelMotion() - Stops another command before the timeout. Can be used with a sensor or custom logic.

Motion Chaining

- Same as a consecutive moveToPose() commands but without any pauses in between
- Used when speed is preferred over accuracy
- Struct is {optional parameters}
 - .minSpeed = x
 - Overrides everything but maxSpeed and robot's speed is always over x
 - .earlyExitRange = y
 - Distance away from destination where robot is allowed to go below minSpeed and decelerate
 - If y is 0, robot has to wait until point to decelerate
 - .maxSpeed = z
 - Won't go any faster
 - Between 0 and 127
 - .forwards = t
 - TBD - needs to be tested
 - .direction = w
 - Which way the robot should turn to reach an angle (clockwise or counterclockwise)
 - .lead = v
 - Amount of curvature

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
	1	2 Library @ 12:30 pm (Continue CAD)	3	4	5	6
7	8 Library @ 1:00 pm (Brainstorm Intake Solutions)	9	10 Library @ 2:00 pm (Brainstorm Mogo Mech Solutions)	11	12 Library @ 1:30 pm (Began CAD of Mogo Mech)	13
14	15 Library @ 1:30 pm (CAD Mogo Mech)	16	17 Library @ 1:30 pm (CAD Mogo Mech + Finalize Drivetrain)	18	19	20
21	22 Library @ 2:30 pm (CAD Mogo Mech)	23	24 Library @ 1:30 pm (Finish CAD of Mogo Mech)	25	26	27
28	29	30	31			
<p>Upcoming Competitions: October 26th @ Glenbrook South November 2nd @ Lake Park November 22nd @ Speedway Signature Event</p>						

Initial Intake

Define the Problem

Goal: Create an intake mechanism that we can use to suck in rings.

Problem Statement: In order to score the rings, we need to be able to pick them up somehow.

Solution Requirements:

- Only suck up 2 rings at a time
- Must use legal VEX V5 Competition parts



Solution Goals:

- Be able to pick up rings efficiently and fast
- Be able to pick the top ring in a stack
- Be able to score on mobile goals, neutral stakes, and alliance stakes (highest stake not necessary for now)
- Be able to descore other scored rings
- Be a "hook" intake

Initial Intake

Brainstorming Solutions

Goal: Brainstorm ideas for our intake mechanism.

Online Research (Videos are Linked):

- [2145Z](#)
- [LTC Robotics](#)
- [8076A](#)
- [4139B](#)
- [4886S](#)
- [Mankato Area Robotics Club](#)
- [23851A](#)

Note: We plan on brainstorming more intake design closer to MOA when the meta has changed and more designs are circulating.

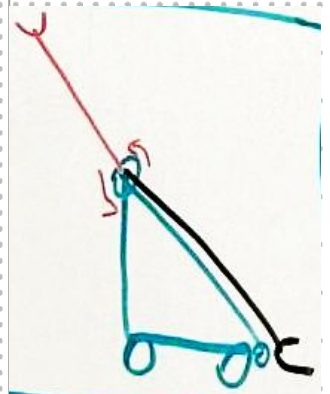
From our online research we found the following common trends... In other words, this is the current intake meta:

- Conveyor belt with standoff hooks to lift rings up
- Set of flex wheels at the beginning to suck up rings
- Polycarbonate at beginning to guide rings

Possible Solutions - Intake:

- Separate Claw Lift
 - Flex wheel + conveyor intake with pneumatic hinge for first set of flex wheels to grab first ring in stack and a separate lift with claw like mechanism at the end to score on the higher stakes

Positives	Negatives	Black is resting position and red is scoring position
<ul style="list-style-type: none">→ Ability to descore→ Easy placement for pneumatic tubing and air tank that is required for beginning intake hinge	<ul style="list-style-type: none">→ Claw like mechanism requires extreme precision when driving→ Requires air splitting between open/close claw and lowering/lifting claw→ Hard placement for pneumatic tubing and air tank that is needed for claw	

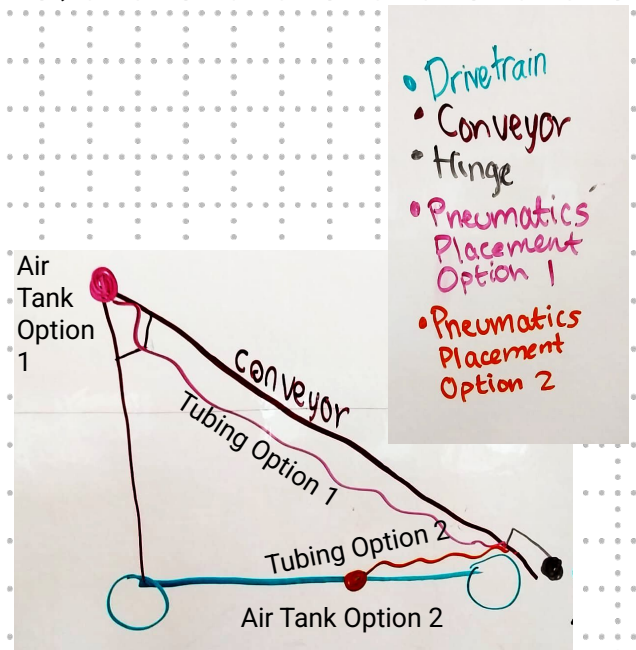


Brainstorming Solutions

• Lifiable Intake

- Flex wheel + conveyor intake with pneumatic hinge for first set of flex wheels to grab first ring in stack; whole intake lifts in order to score the higher stakes

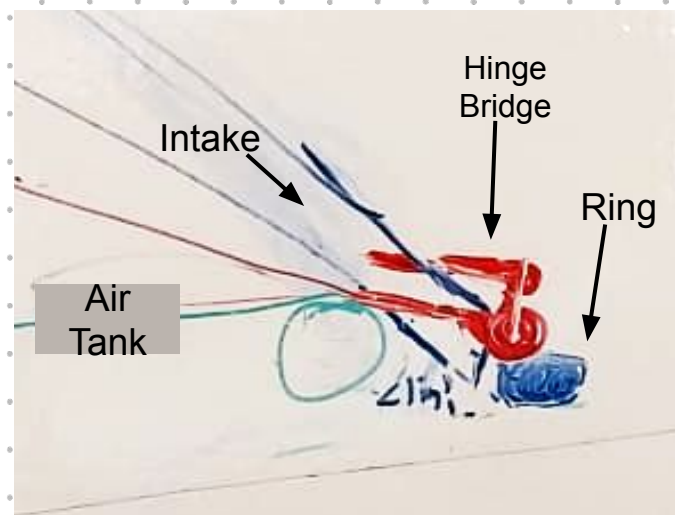
Positives	Negatives
<ul style="list-style-type: none"> → Simple → Unique - nobody has combined a hinge intake with a lifting one yet 	<ul style="list-style-type: none"> → Lifting lots of weight - unsure if it will lift → Hard placement for pneumatic tubing/air tank



• Hinge Bridge

- Flex wheel + conveyor intake that is liftable to score the higher stakes; pneumatic hinge is separate and does not lift

Positives	Negatives
<ul style="list-style-type: none"> → Easy placement for pneumatic tubing/air tank that is required for pneumatic hinge → Unique - nobody has combined a hinge intake with a lifting one yet 	<ul style="list-style-type: none"> → Very experimental - need to ensure polycarbonate is able to suck in rings AND lift without hitting stationary hinge



Initial Mogo Mech

Define the Problem

Mogo Mech = Mobile Goal Mechanism

Goal: Create a mechanism that can grab and hold the mobile goals in order to move them around and score rings on them.

Problem Statement: In order to score rings on a mobile goal and move it around the field to the positive/negative corners, we need to be able to grab and hold the goal.

Solution Requirements:

- Must use legal VEX V5 Competition parts
- Mobile goal must be able to move around with robot
- Mobile goal must be able to be grabbed and released by robot
- Mobile goal must in a suitable position for rings to be placed onto it (i.e. tilted)



Solution Goals:

- Be hard for other robots to “steal” the mobile goal
- Easily slide around the field
- Be able to hold on to the mobile goal regardless of its original orientation

Initial Mogo Mech

Goal: Brainstorm solutions for our mogo mech.

Online Research (Videos are Linked):

- [90385R - Striker Robotics \(1\)](#)
- [90385R - Striker Robotics \(2\)](#)
- [1032Z - Axis](#)
- [44252A - LTC Robotics](#)
- [8076A](#)

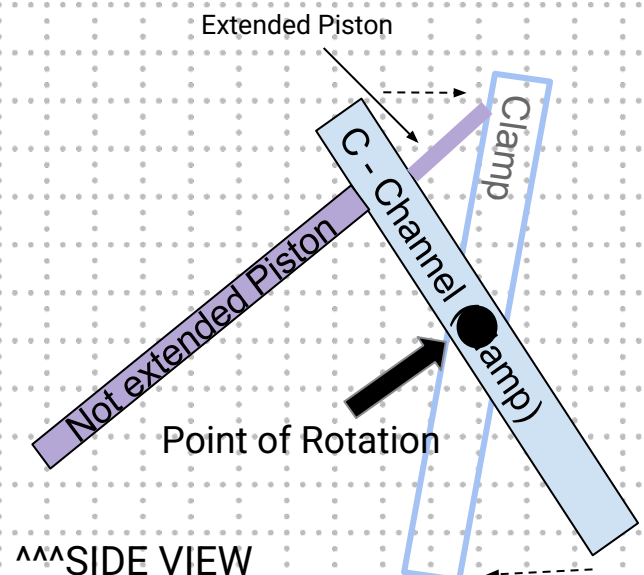
From our online research, we found the following common trends... In other words, this is the current mogo mech meta:

- Lip to slightly lift mogo mech of the ground
- Pneumatic controlled clamp - tilts and grabs mobile goal from a rotating point (see above image)

Possible Solutions - Mogo Mech

- 1 piston
 - Clamp that that grabs and tilts mobile goal (controlled by 1 piston)

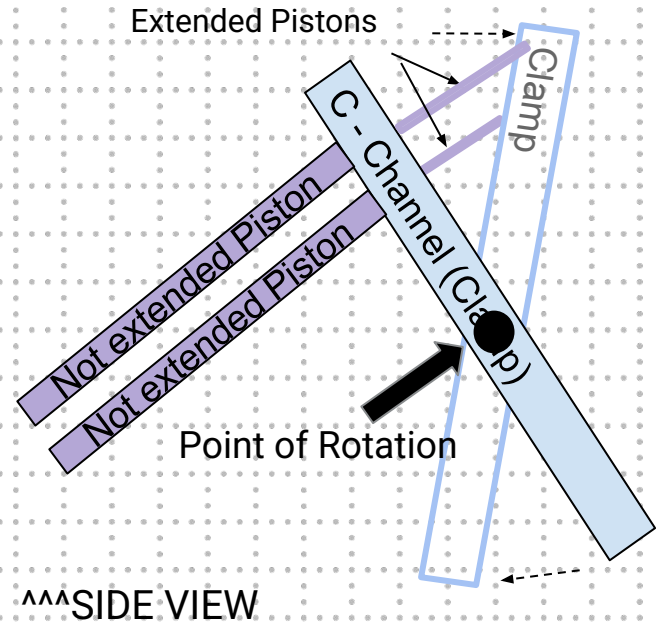
Positives	Negatives
<ul style="list-style-type: none">→ Less air used→ Can use the same air tank for something else	<ul style="list-style-type: none">→ Less strength→ May not be able to pick up the goal - everyone else uses 2 pistons



Brainstorming Solutions

- 2 piston
 - Clamp that that grabs and tilts mobile goal (controlled by 2 pistons)

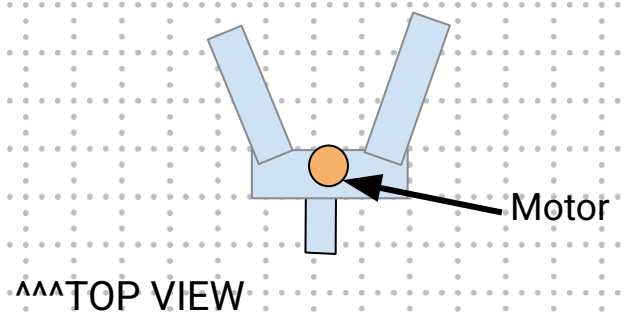
Positives	Negatives
→ Strong - prevents stealing from other robots	→ More air used



- Claw/Grabber Mechanism - Like pincers
 - Claw (similar to crab hands) that grabs the axle of the mobile goal

Positives	Negatives
→ No pneumatics	→ Lots of extra space needed
→ Can use to descore by knocking down goals	→ Would need something extra to tilt mogo mech
	→ Less strong in theory

All blue rectangles are c-channels



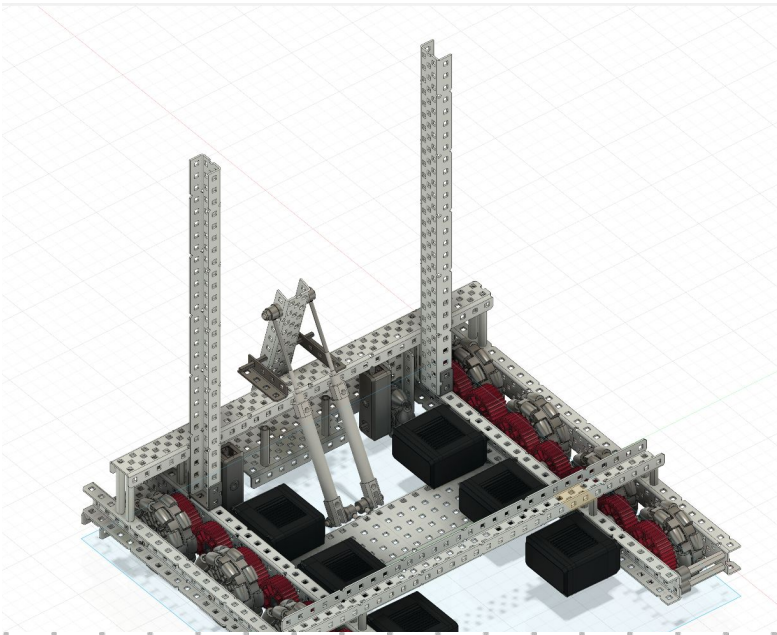
Initial Mogo Mech

Select and Plan

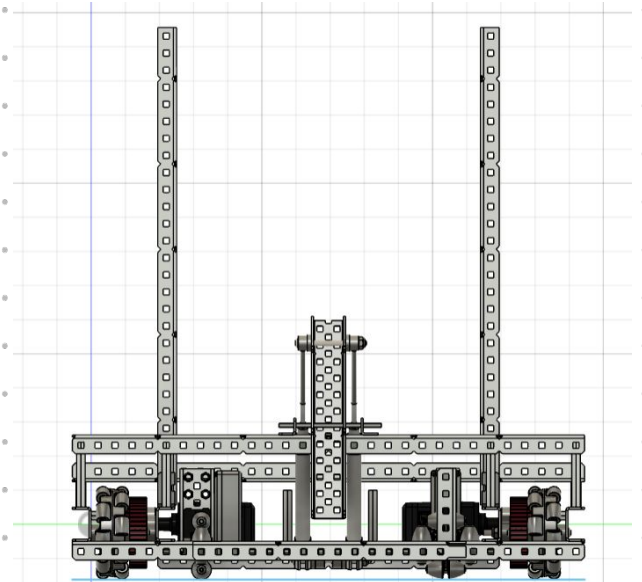
Goal: Select mogo mech design.

Design	Strength (We want to be able to grab the goal) x2 because we prioritise the ability to grab the goal	Air Used (We need air to do other things on the robot)	Space Used (We want less space used on the robot)	Total
One Piston	$2 \times 2 = 4$ Uses pneumatics, so will be a stronger clamp than a motor	3 One piston requires air but less than having multiple pistons	3 Need room for air tank + 1 piston + tubing	10
Two Pistons	$4 \times 2 = 8$ Twice as strong as one piston	2 Two pistons requires a substantial amount of air (depending on size of piston - we can use smaller pistons to counteract this)	2 Need room for air tank + 2 pistons + tubing	12
Claw	$1 \times 2 = 2$ Not very strong in theory as it uses a motor. Also, the claw extends "out" so robots can take away our goal	4 No air used	3 No air tank needed, but claw will need to extend "out" of the robot	9

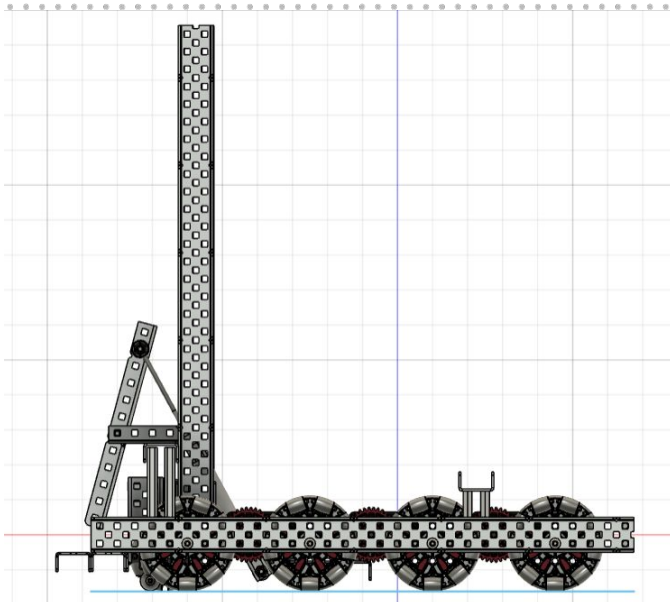
CAD Images of 2 Piston Mogo Mech:



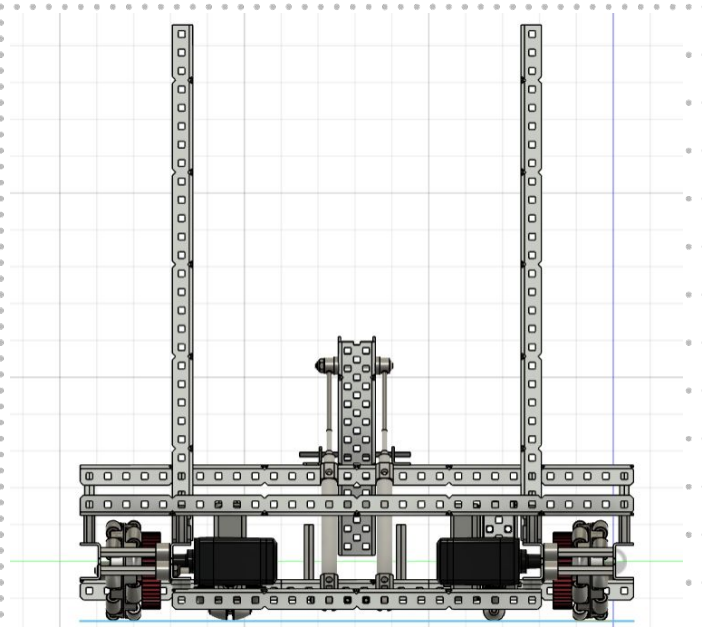
Isometric View



Back View



Side View



Front View

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
				1 Library @ 1:30 pm (Select Intake)	2	3
4	5	6 Basement @ 1:30 pm (Begin Building Drivetrain)	7	8 Basement @ 1:30 pm (Continue Building Drivetrain)	9	10
11	12 Basement @ 9:30 am (Continue Building Drivetrain)	13	14	15	16 Basement @ 5:45 pm (Continue Building Drivetrain)	17
18	19	20	21 Basement @ 1:30 pm (Finish Drivetrain)	22	23	24
25	26	27	28	29	30 Library @ 12:30 pm (CAD Intake)	31
		Upcoming Competitions: October 26th @ Glenbrook South November 2nd @ Lake Park November 22nd @ Speedway Signature Event				

Initial Intake

Brainstorming Solutions (Part 2)

Goal: Brainstorm more ideas for our intake now that the Mall of America (MOA) competition is happening and the meta is changing.

New Online Research (Videos are Linked):

- [1010W](#)
- [3131V](#)
- [229V](#)
- [8110W](#)
- [1233H](#)

Note: This is an add on to pages 45-46 with new ideas now that the meta is changing.

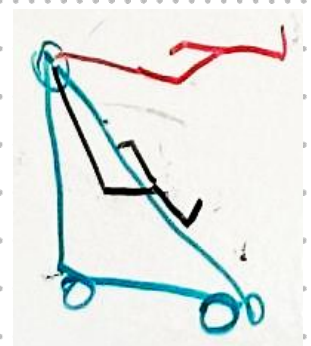
From our online research we found that the intake ideas from our previous entries (pg. 45-46) are still common, but a new strategy is emerging that requires “redirecting” the intake so that the ring lands in a box that is raised to score on the high stakes.

More Possible Solutions - Intake:

- Redirect Intake
 - Flex wheel + conveyor intake with pneumatic hinge for first set of flex wheels to grab first ring in stack. Rings can also be redirected into a second arm with the ability to extend to score in the high stakes.

Positives	Negatives
<ul style="list-style-type: none">→ Can score two rings at a time on high stakes→ Rings can be easily redirected to high stakes→ Ability to multitask	<ul style="list-style-type: none">→ Cannot unsuck a ring once it's past a certain point→ Box arm may be bulky

Black is resting position and red is scoring position



Initial Intake

Select and Plan

Goal: Select our intake design.

Note: We decided not to include liftable intake, because we realized lifting the front hinge does not have an advantage over leaving it stationary.

Decision Matrix for Intake Designs:

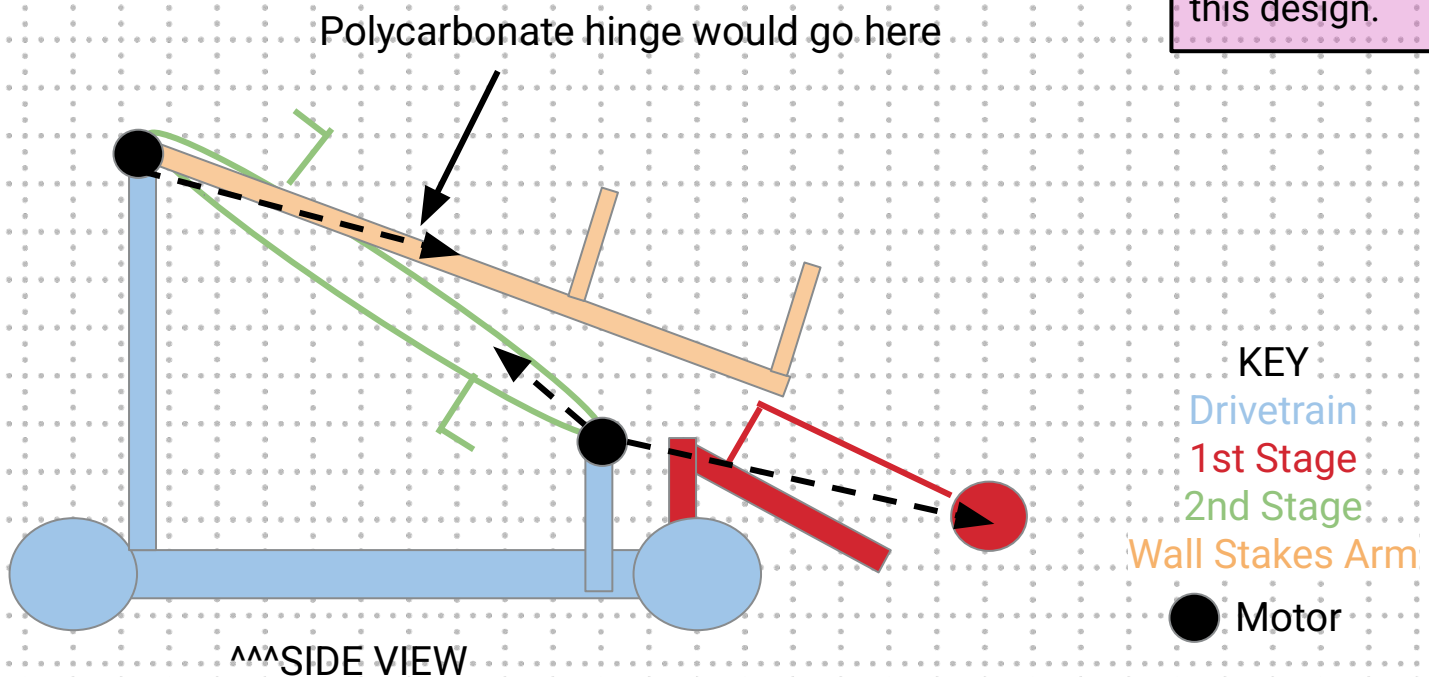
Design	Hinge Bridge	Separate Claw Lift	Redirect Intake
Stability	3 Pretty stable	1 Claw is wobbly and is prone to swaying	3 Pretty stable
Precision	3 Need precision to put on high stake because ring is being flipped onto it	2 Need extreme precision to pick up ring with claw and put on high stake	3 Need precision to put on high stake
Buildability	1 Have to make sure first and second stage align	4 Easy build except for making claw open and close	3 Going to need experimentation with redirecting intake
Size	3 Not much extra space taken because no extra mechanisms	2 Having a separate mechanism and claw takes up a good amount of room	1 Carrier for rings takes us the most space
Materials	3 Two motors + one pneumatic usage	2 Two motors + two pneumatics usages	2 Two motors + two pneumatics usages
Efficiency	2 Can only put one ring at a time but it's moderately fast	1 Can only put one ring at a time	4 Can put two rings at a time on high stakes
Total	15	12	16

Plan for our Initial Intake:

- 2 Stage Intake
 - 1st Stage: Flex wheel hinge controlled by pneumatics (flex wheels will be concentrated on inside to grab the ring faster) to suck in rings both on the ground AND on top of stacks
 - 2nd Stage: Conveyor belt made of chain with standoffs with plates at the ends to act as hooks to catch the rings; these hooks will flip the rings on to the mobile goal (flipping gives more force rather than just regularly sliding the ring onto the goal)
- Redirection Feature for High Stakes
 - Halfway up intake there is a hinge lined with polycarbonate that can go up but not down, allowing rings to go up intake normally but if intake is reversed they are redirected by the polycarbonate into a box that can hold a maximum of two rings
 - Arm connected to box can move up to place the rings on high stakes
 - Box can fold upwards using pneumatics so it is out of the way when not in use (extra feature we may want to implement)

08/02/24 - We have not CAD our intake yet because we need to build and test different angles for the flex wheels and polycarbonate.

11/5/24 - See Intake V2 for our new thoughts on this design.



Initial Drivetrain

Build and Implement

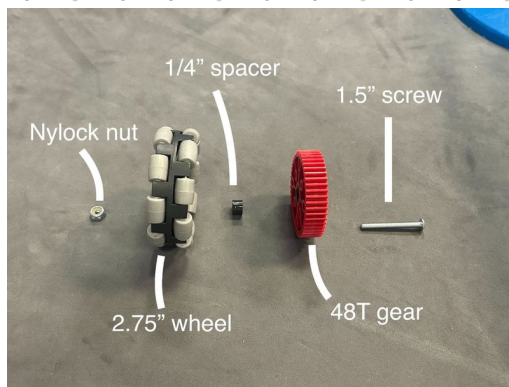
Goal: Build the drivetrain!

Steps by Step Build:

08/06/24: It took us some time to get to building drivetrain because of the lack of parts .

1. Assemble Wheels

- With a 1.5" screw, attach the 48T gears to the wheels with a $\frac{1}{8}$ " spacer in between the two
- Put circle gear inserts on both sides



2. Prepare Inner Drivetrain C-Channels

- Put flat bearings in the following pattern* (displayed below) on the two inner drivetrain c-channels

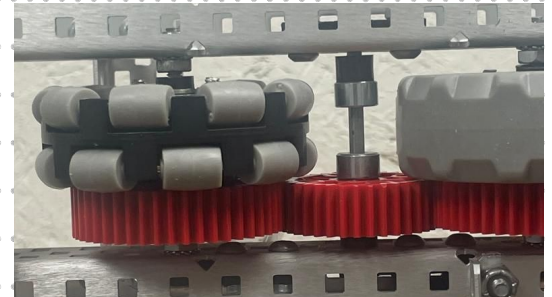
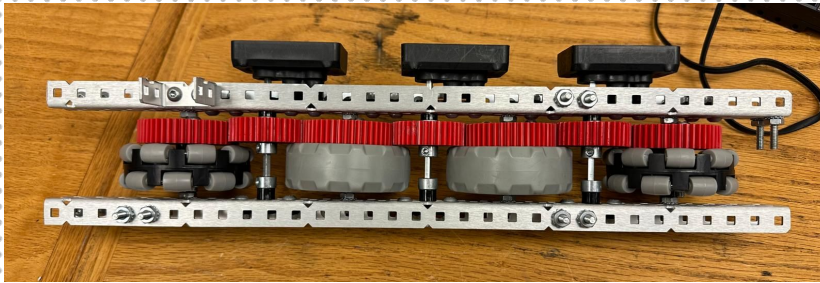


Note: The second c-channel should be a mirror image.

*This pattern exists because of the gear size differences

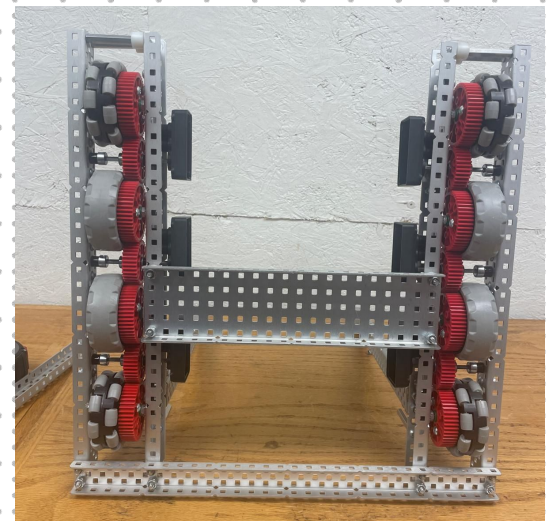
3. Assemble Both Sides of Drivetrain

- a. Starting on the first flat bearing and continuing on to every other, use 2.25" screws to connect the wheels to the two drivetrain c-channels.
 - i. In between the inner c-channel and gear, place a nylock nut and $\frac{1}{8}$ " spacer. In between the wheel and outer c-channel, place a thick washer, $\frac{1}{8}$ " spacer, and nylock nut. Make sure to tighten the nylock nut so it grips the c-channel. Add a nylock nut to the end of the screw.
- b. On the other flat bearings, use 3.5" axles to connect 36T gears to the motor. In between the gear and motor, place a $\frac{1}{8}$ " spacer and a thick washer. In between the gear and outer c-channel, add two collars and a $\frac{1}{2}$ " spacer.

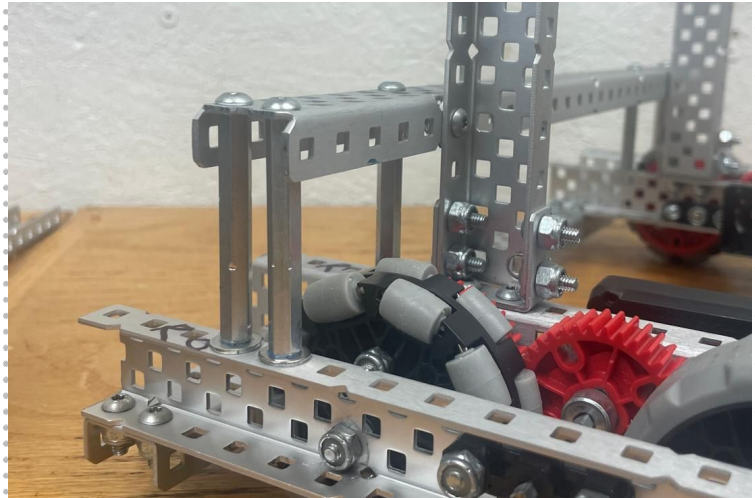
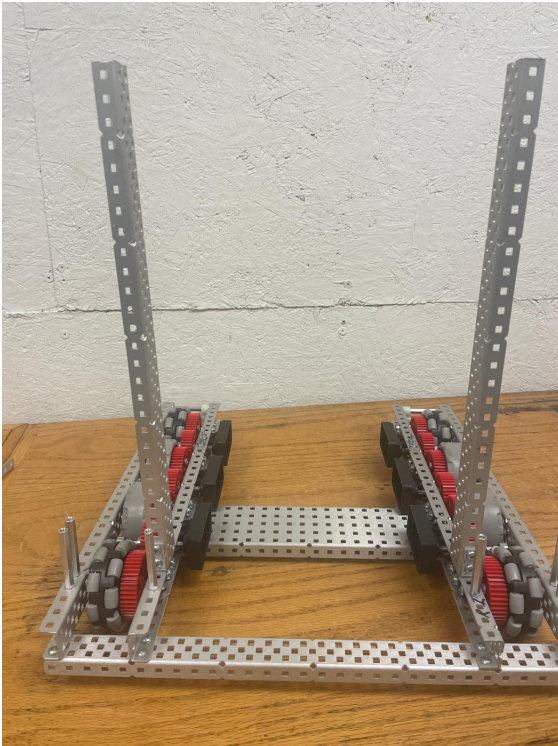


4. Connect Both Sides of Drivetrain

- a. On the bottom of drivetrain, use a 1x2x1x30 c-channel to connect the back end of both sides of the drivetrain and use a 1x5x1x20 c-channel to connect the middle of both sides of drivetrain.



5. Prepare Drivetrain for Intake and Mogo Mech
 - a. Add a 2x2x2x1 u-channel to the two inner drivetrain c-channels
 - a. Use the u-channel to vertically attach two 1x2x1x30 c-channels
 - b. On the top of the robot, connect both sides of drivetrain with a 1x2x1x30 c-channel raised by 2" standoffs on the back of the robot. Use the vertical c-channels to help support it



8/21/24: We didn't make too much progress today for a variety of reasons. We realized we could support the intake c-channel with one of the c-channels we use to connect both sides of drivetrain (see above). This took a while to undo and redo. We did finish drivetrain though! :D

Initial Drivetrain

Test Solution

Goal: Test our drivetrain.

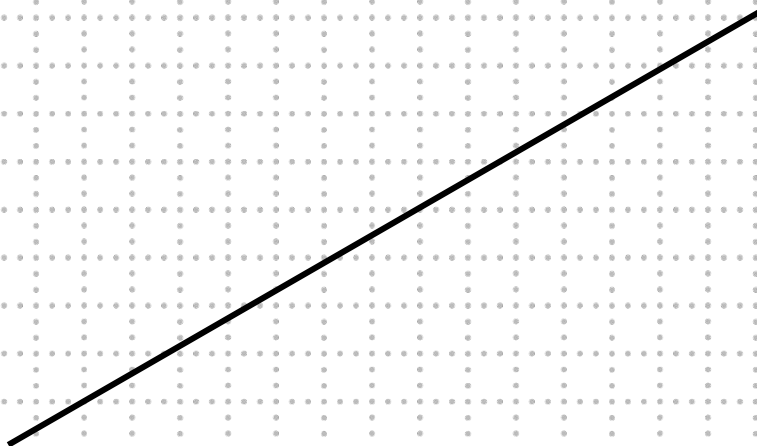
Procedure:

1. Drive around the field using an arcade drive
 - a. Left stick is forward/backward
 - b. Right stick is turning left/right
2. Record any errors or irregularities encountered

We had to use green motors for our testing since we don't have blue motors yet.

Data + Analysis:

- When we drove around the field, we noticed that our controls were swapped (i.e. forward was backward and turn left was turn right) due to a coding error. After we fixed that, it looks like everything ran smoothly.
- We noticed that our robot sometimes veers right and left even though we try to drive in a straight line. We realized that the screw joints on either side were either too tight or too loose which allowed different amounts of friction between both sides of drivetrain. **(See QR code link)** As a result, we used a wrench to equally tighten/loosen all the screw joints on both sides of drivetrain.

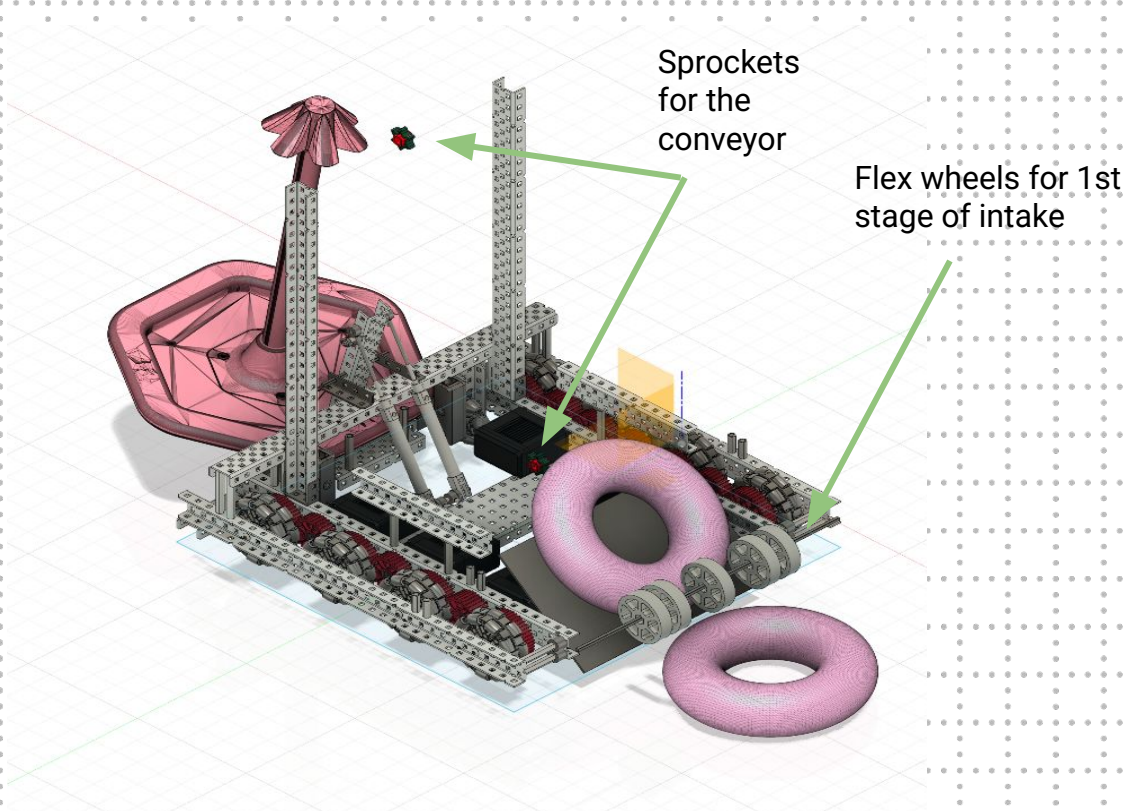
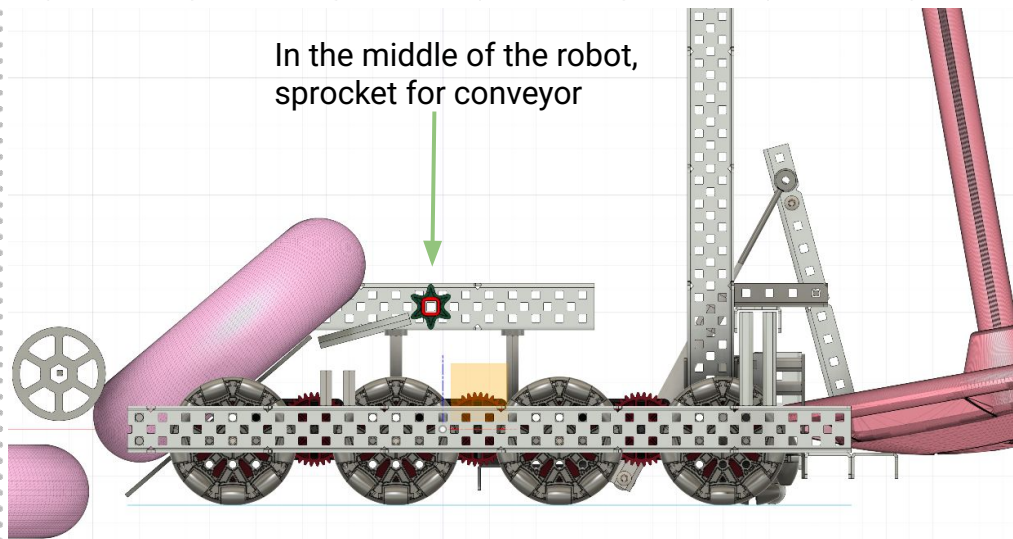


Initial Intake

Select and Plan (Part 2)

Goal: CAD our intake design.

CAD Images of Intake Designs



We finished the above and top left designs in the past 4 meetings. We divided intake into two parts, the 1st and 2nd stage. The first stage is the part with the flex wheels, used to get rings into the robot at a lower angle. The second stage brings rings up to mobile goal height at a steeper angle. (See pg. 58 for more info)

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
1	2 Basement @ 12:30 pm (Continue CADing Intake)	3	4 Library @ 6:00 pm (Continue CADing Intake)	5	6 Library @ 2:30 pm (Continue CADing Intake)	7
8	9	10	11 Library @ 2:30 pm (Continue CADing Intake)	12 Library @ 2:30 pm (Finish CADing Intake)	13 Basement @ 2:30 pm (Start building intake)	14
15	16	17	18 Basement @ 5:00 pm (Continue building intake)	19	20 Basement @ 5:00 pm (Building mogo mech)	21
22	23	24	25 Basement @ 5:00 pm (Continue Building Intake + Strategizing)	26	27	28
29	30					
		Upcoming Competitions: October 26th @ Glenbrook South November 2nd @ Lake Park November 22nd @ Speedway Signature Event				

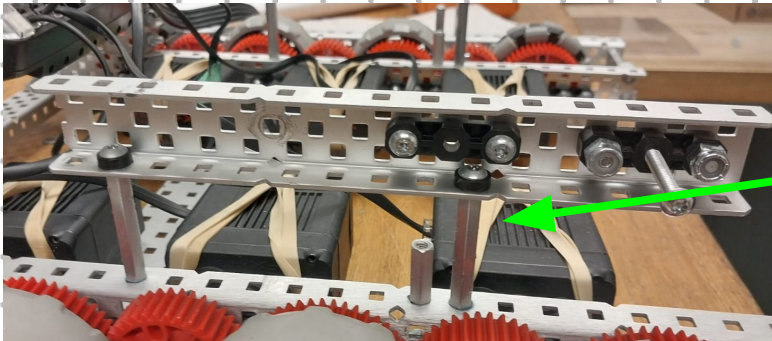
Initial Intake

Build and Implement

Goal: Build our intake!

Step-by-Step Process:

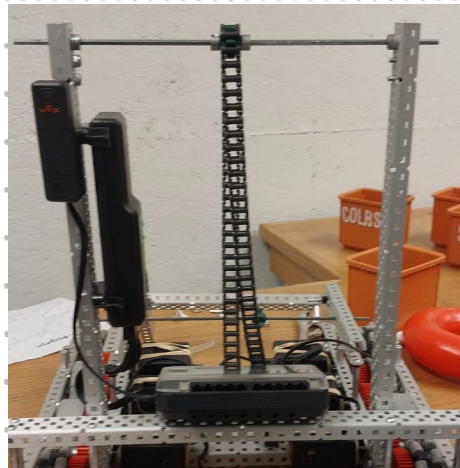
1. Build supports for the intake
 - a. Replicate image below on each side of robot



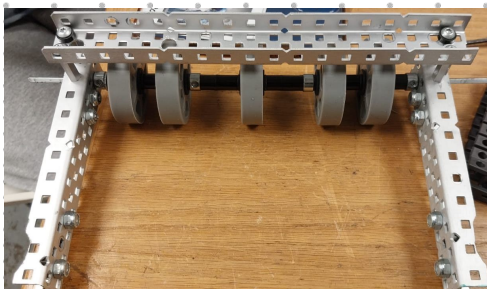
2" standoff

9-13-24 (STEPS 1 AND 2): We started building intake today. We finished building supports and added a placeholder chain. The supports hold up the front part/1st stage of our intake.

2. Add a chain
 - a. Add a chain.
 - b. Don't add proper spacing for now, this is just a placeholder



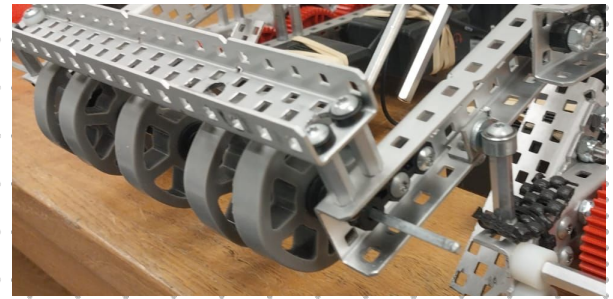
3. Build and attach the front part of intake
 - a. Build according to specifications below
 - b. Add standoffs to raise intake



9-18-24 (STEPS 3 AND 4) : We spent today building the 1st stage of intake.

4. Raise intake

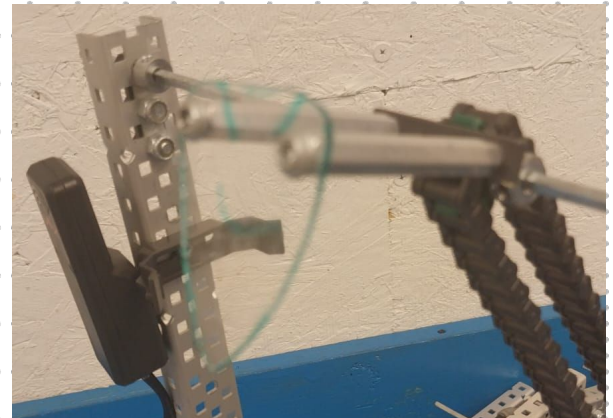
- a. Add standoffs to raise the intake (so it can grab rings)



9-20-24 (STEP 5): We only did step 5 today because cutting polycarbonate too time. We also built our mogo mech today. See details on that page.

5. Add hooks to the seconds stage of intake

- a. Attach 2 inch standoffs to chain, spreading them so a ring can fit between them
- b. Add polycarbonate hook so it fits halfway over the radius of the ring (not the whole ring, just the plastic part)
- c. Add smiley face on the polycarb with standoffs as eyes for character

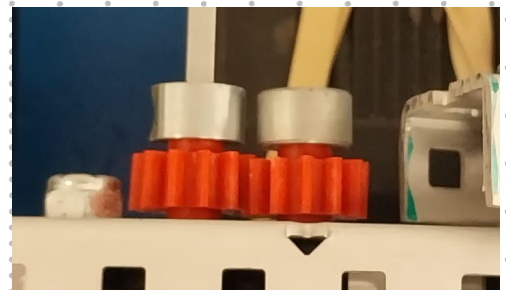


6. Gear together 1st and 2nd stage of intake

- a. Attach 6T sprockets to the outside of intake
- b. Add 12 tooth gears on the inside



9-25-24 (STEP 6): We finished step six and moved on to intake testing today. We also worked on some strategizing. See details on that page. ;D



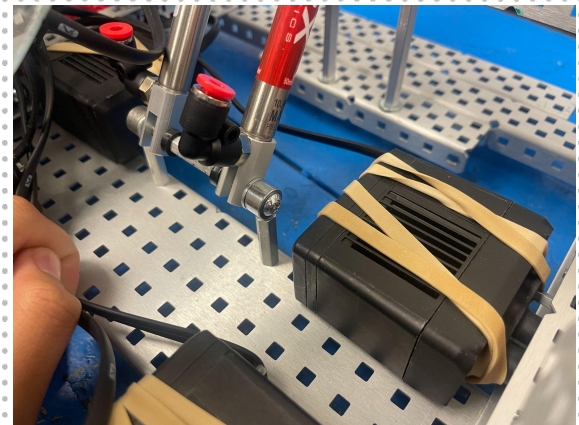
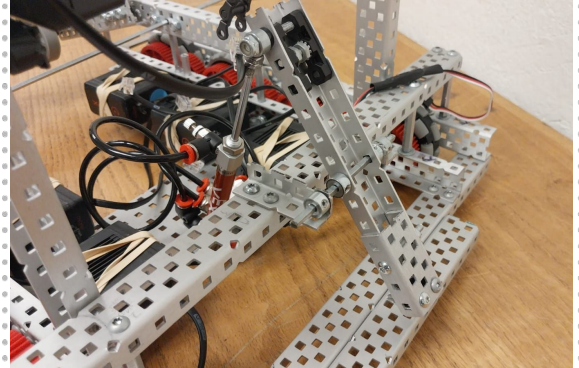
Initial Mogo Mech

Build and Implement

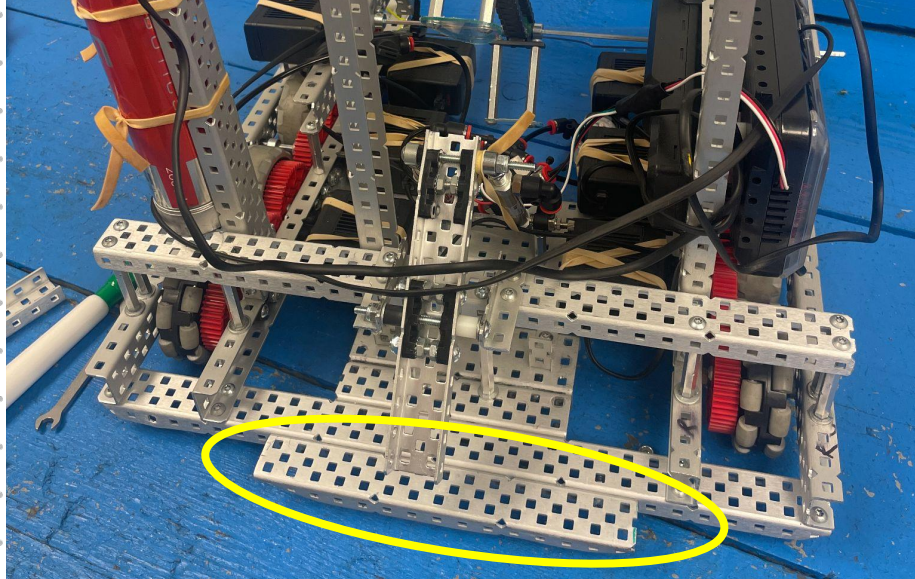
Goal: Build our Mogo Mech!

Step by Step Process:

1. Add L pieces to mogo mech support
 - a. Attach two 4 long L pieces in the middle of the mogo mech support c-channel as shown below
2. Add mogo mech c-channel
 - a. Put a 3.5" axle through the 7th hole of a 1x2x1x13 c-channel
 - b. Make sure to use collars as pictured to ensure the c-channel is secure
3. Add pneumatics
 - a. Make a pneumatics system similar to that on page 3
 - b. Attach a 50mm piston to the top hole of the mogo mech c-channel
 - i. Make sure to use flat bearings!
 - c. Connect the base of the piston to the 5 wide c-channel on drive train as shown in the picture
 - d. Repeat this with another piston for the opposing side of the mogo mech c-channel



4. Add lip for mogo mech
 - a. Add a 1x2x1x15 c-channel to the connector c-channel at the back of the robot (as displayed below) to create a lip for the mogo mech to lift onto



Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
		1	2	3	4	5
6	7	8 Library @ 3:00 pm (Create auton strategy)	9 Basement @ 5:30 pm (Strategy & Design Meeting with Mr. K)	10 Basement @ 1:00 pm (Continue building intake)	11 Basement @ 9:30 am (Finish building intake, + program motor temp)	12
13	14	15	16	17	18	19
20	21	22	23	24	25 Basement @ 3:30 pm (Test out and fix intake)	26 We dropped this competition due to concerns about timing
27	28	29 Basement @ 3:00 pm (Test mogo mech)	30 Basement @ 3:00 pm (Fix transition from intake to mogo mech)	31 Basement @ 3:00 pm (Finalize robot for comp and begin programming auton)	Basement @ 3:00 pm (Program auton and skills)	Lake Park Competition! :)
<p>Upcoming Competitions: October 26th @ Glenbrook South (No longer attending) November 2nd @ Lake Park November 22nd @ Speedway Signature Event</p>						

Initial Auton

Define the Problem

Goal: Define the requirements and criteria for our autonomous program.

Problem Statement: We need to an effective strategy for auton in order to win the auton bonus point which is worth 6 points! This is especially important in a low scoring game like this year.

Solution Requirements:

- Must break the plane of the starting line
- Must start contacting the starting tiles
- Must fit inside the 15 second time period

Solution Goals (AWP Requirements):

- At least 3 scored rings of the alliance's color
- A minimum of 2 stakes on the alliance's side of the auton line with at least 1 ring of the alliance's color scored
- Neither robot contacting/breaking plane of the starting line
- At least 1 robot contacting the ladder

Key Reminders

- Climb points and corner modifications are not included in the alliance's auton bonus calculation
- If the auton period ends in a tie, each alliance will receive 3 points

Goal: Brainstorm solutions for our auton strategy.

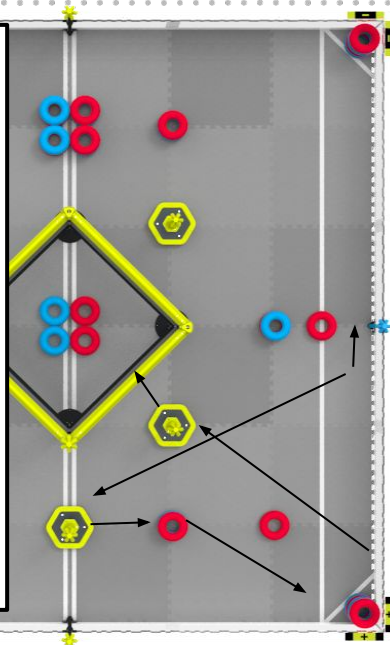
Independent Inquiry:

- Auton Bonus is crucial because in a low scoring game like this season, those 6 points make up a large percentage of the total points
- Utilize as many stakes as possible because the top ring on every stake gets an extra 2 points on top of its regular value
- Get the 3rd contested stake in the middle of the two positive corners. This is an advantage during driver control because then we have general control over 3 goals while the opposing alliance has only 2
- **Staging is important for driver control. It's important that we're positioned towards the positive corner and have a mobile goal in our possession. That way, we can "claim" our positive corner once auton begins**
- We have seen teams online at large competitions like MOA utilizing the positive corners during auton

Possible Auton Strategies for Positive Corner Side:

- Strategy 1 (9 Points)

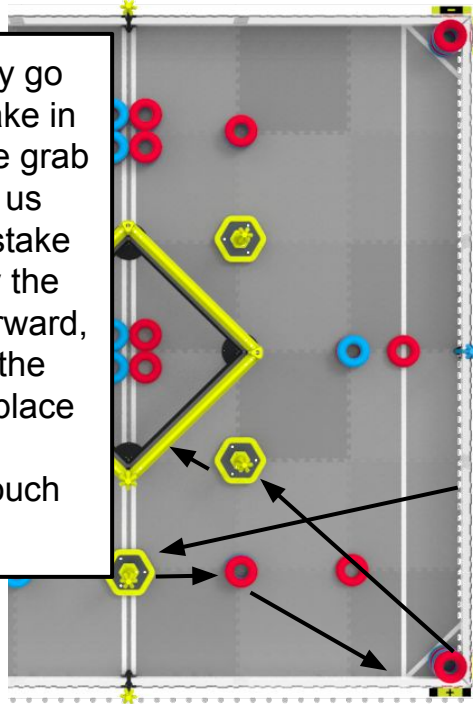
First, we place our preload on our alliance stake. Then, we get the contested middle goal. Afterward, we place the ring behind us on that stake. Then, we put that goal by the positive corners and get a ring from the positive corners and place it on the goal ahead. Finally, we touch the ladder.



Positives	Negatives
→ 3 stakes are scored (maximizing number of scored stakes)	→ Not immediately going for contested goal

• Strategy 2 (7 Points)

First, we immediately go for the contested stake in the middle. Then, we grab the ring right behind us and put that on the stake and put the stake by the positive corner. Afterward, we grab a ring from the positive corner and place it on the mobile goal ahead. Finally, we touch the ladder.



Positives

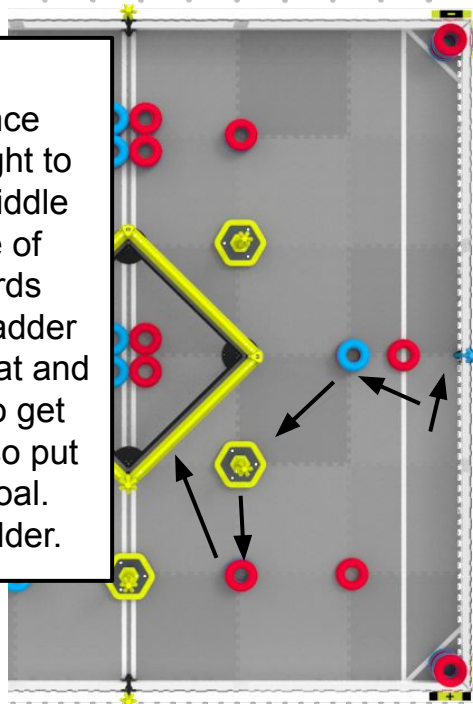
- Immediately getting contested stakes (increased chances of getting it)
- 3 stakes are scored

Negatives

- The contested stake might be stolen first, meaning we lose auton
- We do not utilize the alliance stake as an extra stake

• Strategy 3 (7 Points)

First, we place our preload on our alliance stake. Then, veer right to get the ring in the middle of our alliance's side of the field. Drive towards mobile goal by the ladder and place ring on that and drive slightly more to get the next ring and also put that on the mobile goal. Finally, touch the ladder.



Positives

- Everything this is in our hands because we are not attempting anything that is related to a contested object

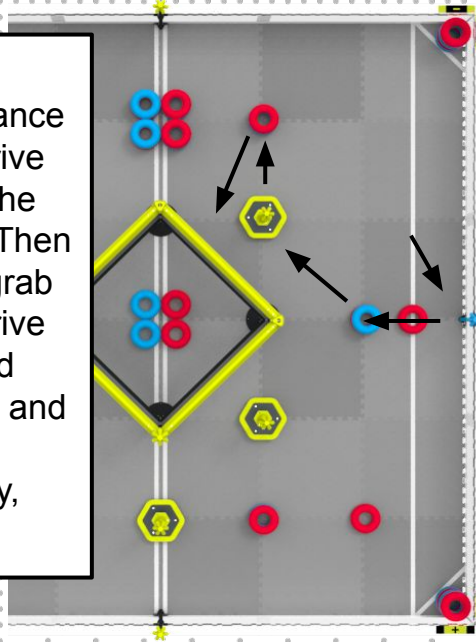
Negatives

- Not attempting to get contested stakes (we lose chance of having general control over 3 during driver control)

Possible Auton Strategies for Negative Corners:

• Strategy 1 (7 points)

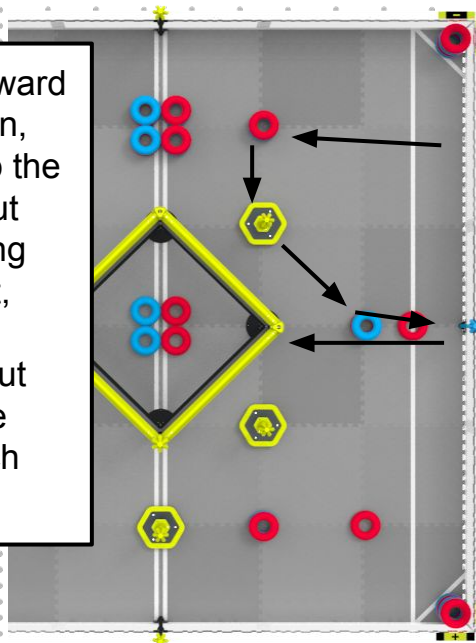
First, we put the preload on the alliance stake. Then, we drive forward and grab the ring in front of us. Then we turn right and grab the mobile goal, drive slightly forward and grab the other ring and both rings on the mobile goal. Finally, touch the ladder.



Positives	Negatives
<ul style="list-style-type: none"> → All actions are concentrated in one spot of the field → Solo AWP 	<ul style="list-style-type: none"> → No “messaging up” of opposing team → Positive corners are far from our ending location

• Strategy 2 (7 points)

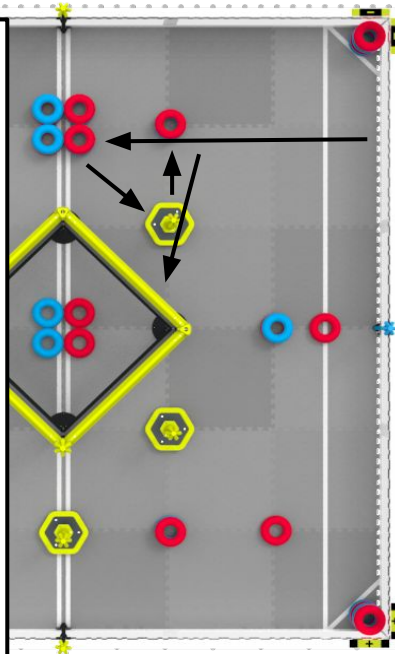
First, we move forward to grab a ring. Then, we turn left to grab the mobile goal and put our preload and ring on there. After that, turn left to grab another ring and put that on the alliance stake. Finally, touch the ladder.



Positives	Negatives
<ul style="list-style-type: none"> → Solo AWP → Close to positive corners as possible (when on negative side) 	<ul style="list-style-type: none"> → No “messaging up” of opposing team → Lots of turning since mogo mech is on back of robot and intake is on front

• Strategy 3 (5 points)

First, we go towards the contested stake in the middles to knock the stacks down, in turn decreasing the accuracy of the opposing team's auton. We then grab the bottom ring on the left stack and grab the mobile goal. We put our preload and ring on the mobile goal and slightly drive forward to grab the ring in front. Finally, we touch the ladder.



Positives	Negatives
→ Knocking down stacks may alter the accuracy of the opposing alliances' auton	→ Only one stake is scored

Initial Auton

Select and Plan

Goal: Decide on an auton strategy

Decision Matrix for Positive Side Auton:

Design	Strategy 1	Strategy 2	Strategy 3
Points (x2 because we want auton bonus!)	4 * 2 = 8 Nine points	3 * 2 = 6 Seven points	3 * 2 = 6 Seven points
Staging for Driver Control	4 Puts contested goal by positive corner	4 Puts contested goal by positive corner	2 Does not put any stakes near positive corner
Goals Controlled	3 Alliance stake + <i>possibly</i> contested goal + one more mobile goal	4 Gets contested goal first, higher likelihood of controlling 3 during driver control	2 Alliance stake + one more mobile goal
Attainability with Given Time	2 Everything is scattered	3 Everything is not close together	4 Very minimal everything is in one spot
Solo AWP?	1	1	0
Total	18	18	14

Decision Matrix for Negative Corner Side Auton Strategies:

Design	Strategy 1	Strategy 2	Strategy 3
Points (x2 because we want auton bonus!)	4 * 2 = 8 (7 points)	4 * 2 = 8 (7 points)	3 * 2 = 6 (5 points)
Staging for Driver Control	3 Far from positive corners	4 Close to positive corners	3 Far from positive corners
Goals Controlled	4 Alliance stake + one mobile goal	4 Alliance stake + one mobile goal	3 One mobile goal
Attainability with Given Time	4 Moderately Attainable	3 Too many turns bc mechs are on opposite sides o	4 Moderately Attainable
Solo AWP?	1	1	0
Total	20	20	16

For **positive corner** auton, why choose Strategy 2 over Strategy 1?

(Since they were tied in the decision matrix)

1. We are immediately heading to get the contested stake, increasing our likelihood of getting it. It is extremely important that we have control over 3 mobile stakes during driver control, and this is the perfect opportunity.
2. Although the strategy is not the highest scoring, we could add alliance stakes in before we touch the ladder if we are quick enough. This would increase our point value to 10 points which would be the highest out all of our strategies.

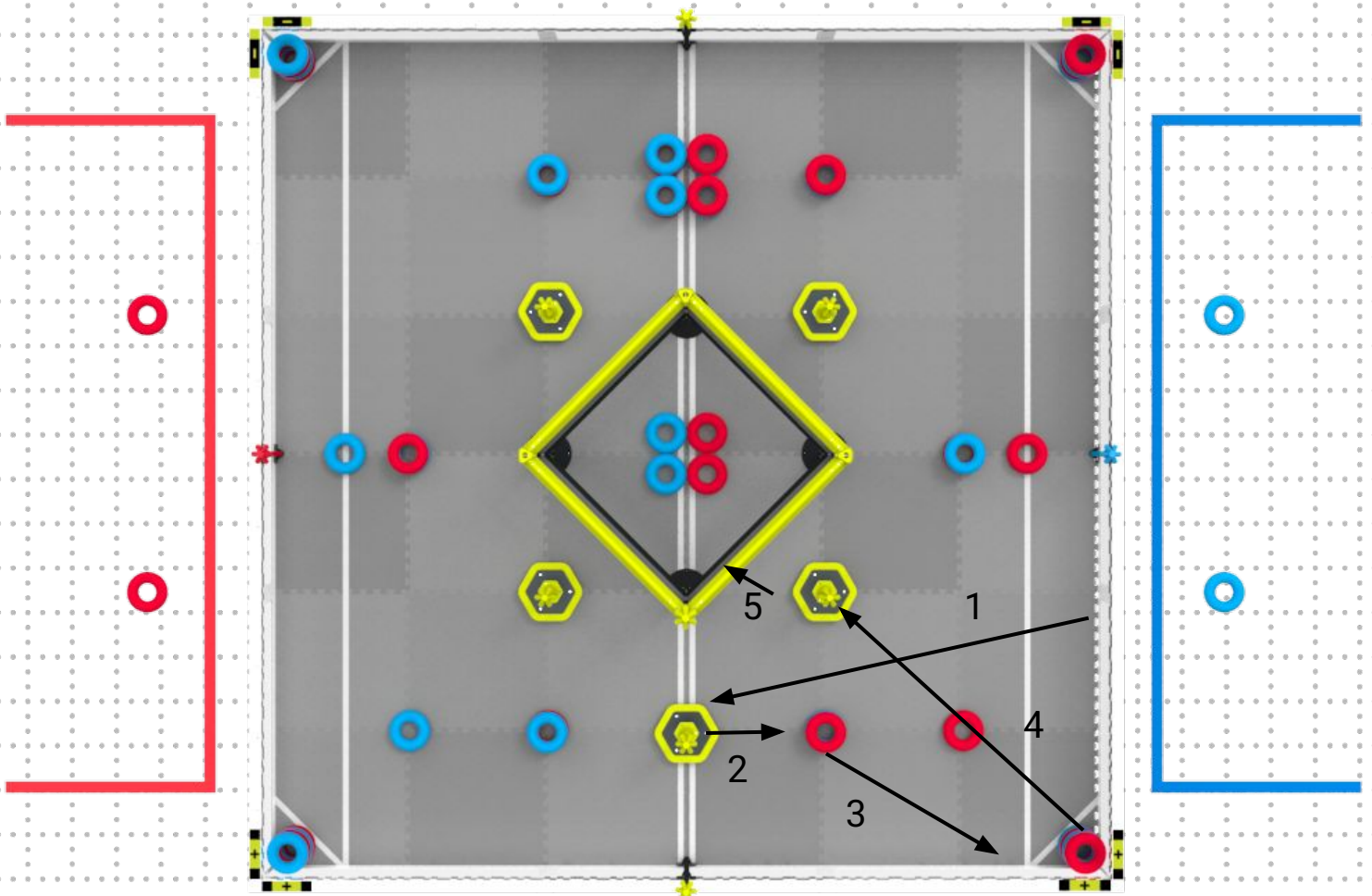
For **negative corner** auton, why choose Strategy 1 over Strategy 2?

(Since they were tied in the decision matrix)

1. Because we most likely won't have a PID program by our next competition due to time constraints, we need something that is accurate and reliable even without utilizing PID. Strategy 2 requires many more turns since mogo mech and intake are on opposite sides of the robot, which increases the precision needed.
2. With Strategy 1, although we don't end near the positive corners, we end near the stack of rings that we can easily grab as soon driver control starts.

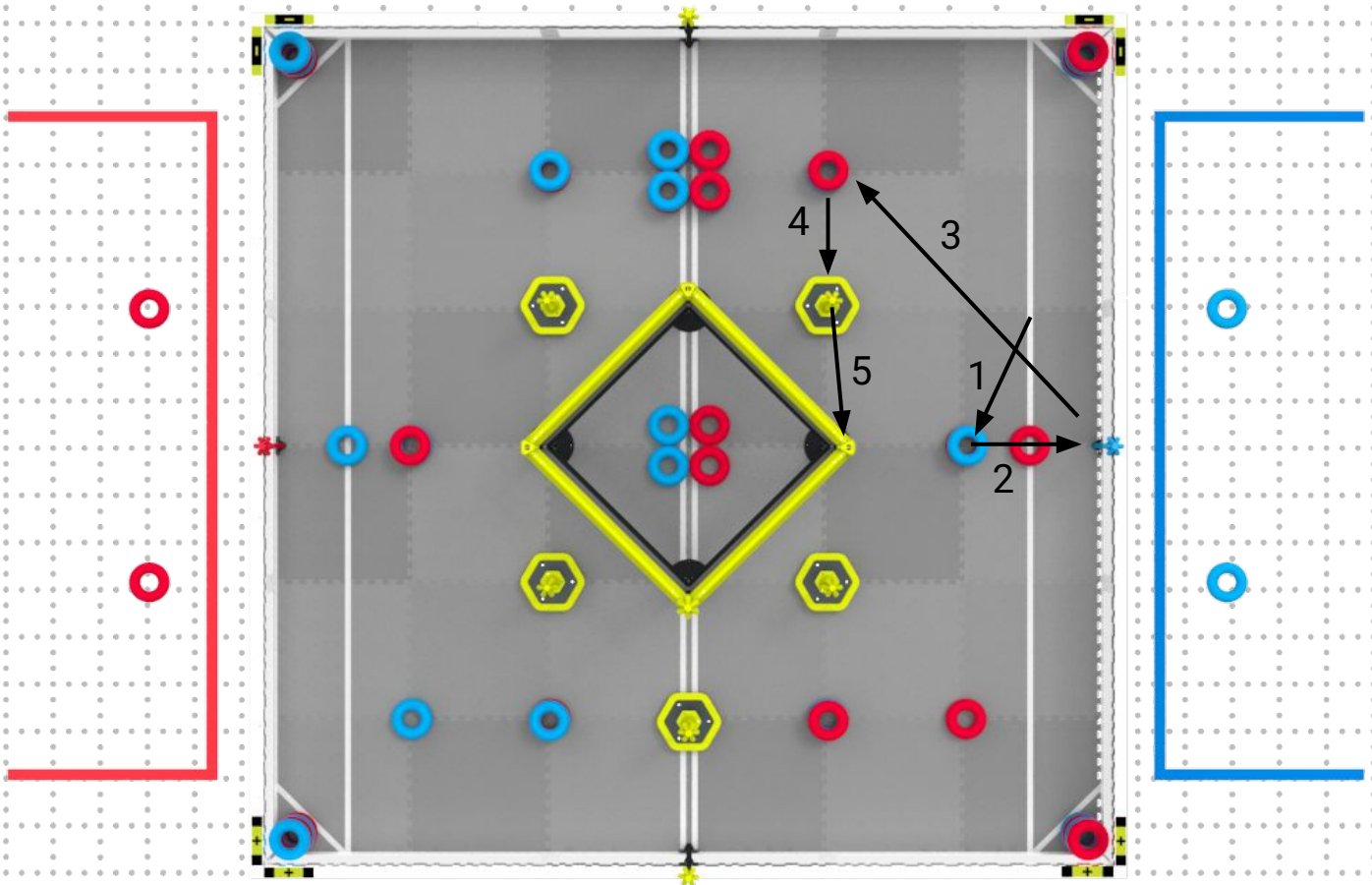
Plan for Positive Corner Auton (7 Points + Solo AWP):

1. With the preload, immediately drive towards and grab the contested middle mobile goal. We want to have control of this because then our alliance team has general control over 3 stakes while the opposing alliance has general control over 2, giving us an advantage for driver control. We can then put our preload on the mobile goal.
2. We then drive backward towards another stack of rings and grab the bottom one of our color. We can then put that ring on the mobile goal.
3. Turn and drive towards the positive corner and leave the mobile goal there and grab a blue ring from the stack in the positive corner.
4. Drive towards the ladder and grab the mobile goal on the way and place the ring on the goal.
5. Continue driving to touch the ladder.



Plan for Negative Corner Auton (7 Points + Solo AWP):

1. With the preload, drive toward the stack to grab the top ring.
2. Back into the alliance stake and put the preload on it.
3. Drive to the stack and intake the bottom ring.
4. Turn around and back into the mobile goal. Then, activate mogo mech to take possession of the goal.
5. While driving towards the ladder to touch it, run intake stage 2 to drop both rings onto the mobile goal



Beginning Programming Tweaks

General/Other

Goal: Add temperature checking for drivetrain motors, drivetrain motor locking, and toggles for intake and mogo mech

Temperature Checking:

In our program's initialize task, we added code to print the temperature of our drivetrain motors to the controller screen and update it every 500 ms.

1. Inside a while loop set to true, we create an array of temperatures for both the left and right drivetrain motor groups.
2. Because the temperatures of all six motors would be too long to fit on a single line of the controller screen, we split them into two lines. The left motors are on the first line, and the right motors are on the second. We printed all three elements of each temperature array to the controller screen.
3. When testing, we ran into a problem where a long decimal number of repeating zeroes was printed for each motor. This happened because the `get_temperature` and `get_temperature_all` functions in Pros returns doubles, which are decimal variables. This was unnecessary because VEX motors only measure temperatures in increments of 5 degrees Celsius. To solve this, we cast each temperature to an integer before printing.
4. We also included a 50 ms delay between printing the first and second lines to the controller because the screen needed time to refresh.

```
void initialize() {
    pros::Task background_task(function: [&]() -> void {
        while (true) {
            //Print temperature of each drivetrain motor to the controller screen
            //Create array of temperatures for each motor group
            std::vector<double> leftTemps = leftMotors.get_temperature_all();
            std::vector<double> rightTemps = rightMotors.get_temperature_all();
            //Print temperature from each element of the left and right arrays
            controller.print(line: 0, col: 0, fmt: "%i, %i, %i", (int) leftTemps[0], (int) leftTemps[1], (int) leftTemps[2]);
            pros::delay(milliseconds: 50); //Minimum of 50ms delay is needed between controller print commands
            controller.print(line: 1, col: 0, fmt: "%i, %i, %i", (int) rightTemps[0], (int) rightTemps[1], (int) rightTemps[2]);
            // delay to save resources
            pros::delay(milliseconds: 500);
        }
    });
}
```

Mogo Mech Toggle:

We also changed our mogo mech code to run by toggling a button instead of holding it. This was a matter of personal preference. We wanted it to stay extended until we pressed a button a second time to retract it. To do this, we created a variable called “mogoTogg” to store the current state so we can toggle it. If the L1 button on the controller is pressed, the code checks the current toggle state. If mogoTogg is set to false, the pneumatic piston extends, and the variable is changed to true to reflect the current state. If the variable was already set to true when the L1 button is pressed, it will be changed back to false, and the mogo mech piston will be retracted.

```
void driverButtons() {
    bool mogoTogg;
    bool intakeTogg;
    while (true)
    {
        //Mogo mech piston
        if (controller.get_digital(button: pros::E_CONTROLLER_DIGITAL_L1))
        {
            if (!mogoTogg)
            {
                mogoMech.set_value(true); //out
                mogoTogg = true;
            }
            else
            {
                mogoMech.set_value(false); //in
                mogoTogg = false;
            }
        }
    }
}
```

Intake Toggle:

We used a similar toggle system for the our intake, this time using a variable called “intakeTogg” to track the state of the intake toggle. Pressing the R1 button toggles intake on and off. When it’s on, the intake motor is set to 50 volts.

```
//Intake
if (controller.get_digital(button: pros::E_CONTROLLER_DIGITAL_R1))
{
    if (!intakeTogg)
    {
        intake.move(voltage: 50); //on
        intakeTogg = true;
    }
    else
    {
        intake.brake(); //off
        intakeTogg = false;
    }
}
```

Motor Locking:

We also added programmed one of our controller buttons to lock all our drivetrain motors using the “Hold” braking mode. We used this in our driver control last season to prevent other robots from pushing us during a match. This is especially useful when our robot is on defense because our motors will actively resist any motion when this is on. It can be activated by holding the B button on the controller. When the button is released, the motors unlock, switching back to the default coast mode to avoid putting unnecessary strain on

```
//Motor locking
if (controller.get_digital(button: pros::E_CONTROLLER_DIGITAL_B))
{
    wheels.set_brake_mode_all(mode: pros::E_MOTOR_BRAKE_HOLD);
    wheels.brake();
}
else
{
    wheels.set_brake_mode_all(mode: pros::E_MOTOR_BRAKE_COAST);
}
```


Initial Intake

Test Solution

Goal: Test our intake.

We noticed the first stage of our intake wasn't able to pick up rings initially because the polycarbonate was too steep. For our first competition, we are prioritizing picking up rings, so this was a major problem for us. As a result, we decided to test the effects of the polycarbonate angle on the seconds it takes to pick up a ring.

Procedure:

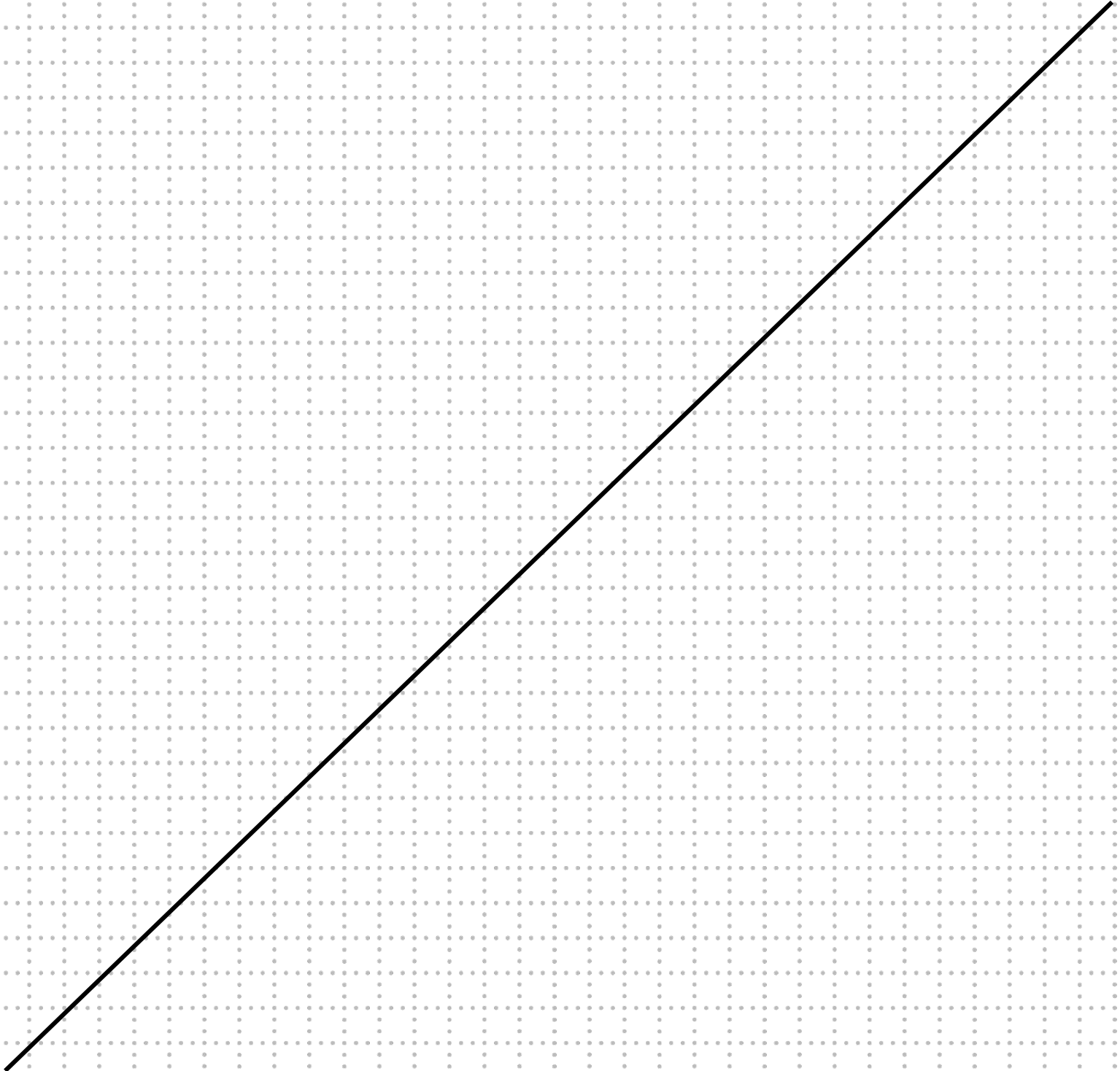
1. Place the robot so that intake is lined up with the edge of a tile
2. Place the center of the ring on the tile edge that is parallel and across from the robot's current position
3. Start intake and the stopwatch
4. Drive forwards towards the ring
5. Stop the stopwatch once the ring has gone through both stages of intake
6. Record the time displayed
7. Repeat steps 1-6 three more times
8. Repeat steps 1-7 two more times, each time adjusting the current slope of the polycarbonate by 5 degrees

Data/Results:

Slope of Polycarbonate	Seconds to Pick up a Ring From Tile			
	Trial One	Trial Two	Trial Three	Average
Original	6.97	13.22	5.32	8.503
-5 degrees	4.33	5.29	4.23	4.617
-10 degrees	1.30	.74	1	1.013

Analysis:

On average, the slope with a decrease of 10 degrees worked the best (average of 1.013 seconds per ring) As a result, we are going to keep our polycarbonate at this angle.



Initial Mogo Mech

Test Solution

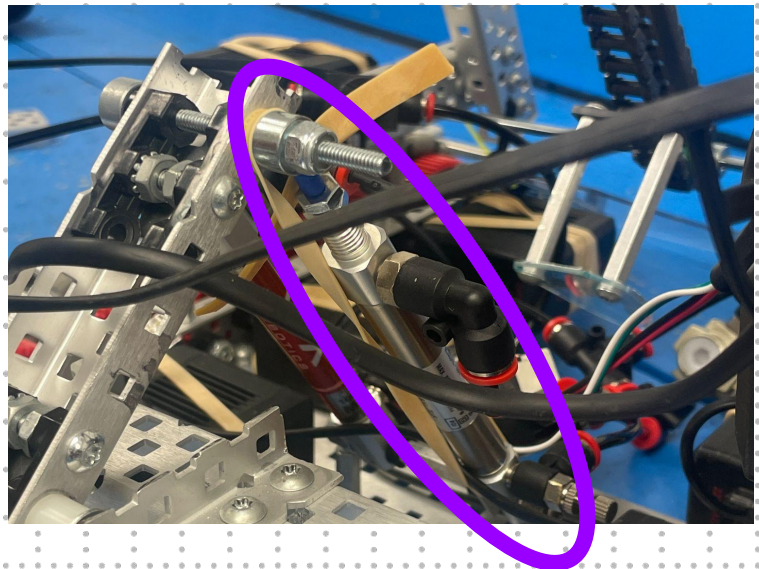
Goal: Test our mogo mech to see the effect of number of rubber bands on the psi used and pneumatic efficiency and to see its overall capabilities.

10/29/24: We came back to this today - we realized we never actually tested our MogoMech after we fixed it. :/

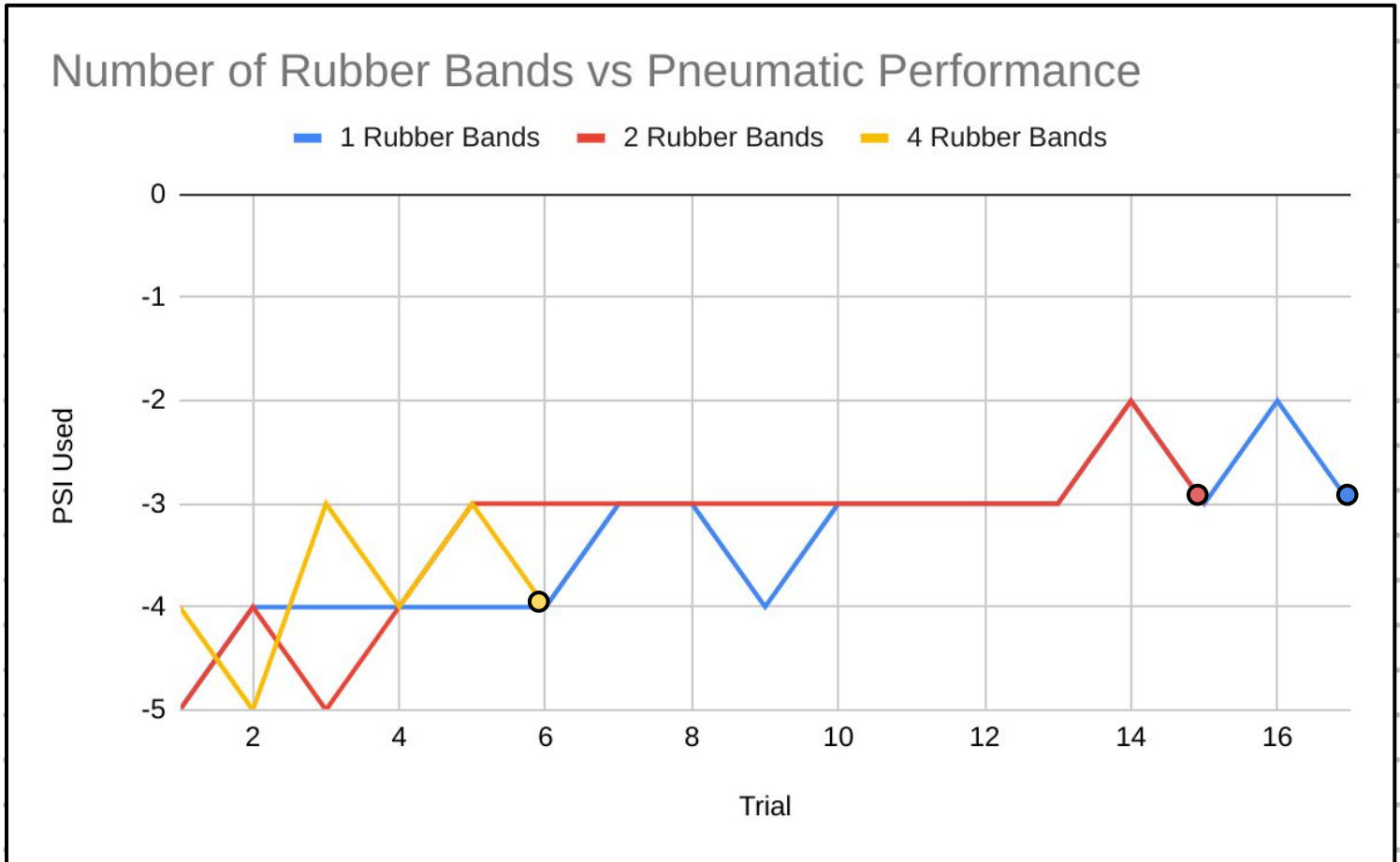
Procedure:

1. Fill the pneumatic air tank with 99 psi
2. Place the mobile goal underneath mogo mech, so it is easily clampable if the piston is extended
3. Use 1 rubber band on the right piston to connect its base and the top of its extension (pictured below)
4. Activate the extension of the pneumatic piston (ours is done with button L2 on the controller)
5. Wait 5 seconds
6. Check the the amount of psi left in the air tank
7. Record the change in psi after one activation
8. Press L2 again and piston should automatically retract because of rubber bands
9. Repeat steps 4-8 until activation no longer lifts mogo mech into a suitable position
10. Repeat steps 1-9 with 2 rubber bands (1 on each side) and 4 rubber bands (2 on each side)

This is how the rubber bands attached to the piston. (Step 3)



Data/Results:



# of Rubber Bands	1 Rubber Band	2 Rubber Bands	4 Rubber Bands
Average PSI Used Per Activation	-3.352941176	-3.333333333	-3.833333333
# of Activations	17	15	6

Analysis:

From the data, it is apparent that having 2 rubber bands, on average, uses the least amount of psi per activation (3.333 PSI). However, having 1 rubber band lasted the most mogo mech activations (17 activations). As a result, we are using 1 rubber band on the right piston to ensure the most activations possible during a single match. In general, mogo mech worked as expected.

Competition 1

11/02/24



HEXA-FORCE

Lake Park Competition

Goal: To qualify for state and scope out the meta (general strategy + design)

We won the Design Award and qualified for State!

Qualification Matches - 26th Place of 40 :(
Skills - 5th Place out of 23 :D

Ranking Data Post-Competition:

3-3-0

6 WP - (win points)

12 AP - (alliance points)

53 SP - (Strength of Schedule Points)

Skills Score: 38 (35 Driver, 3 Auton)

OPR - 13.24 (Offensive Power Rating)

DPR - 8.58 (Defensive Power Rating)

CCWM - 4.66 (Calculated Contributed to Winning Margin)

True Skill Ranking - #3628/9,533

World Skills Ranking - #603/1,834

What is our Ranking Data?

1-1-1 → 1 Win, 1 Loss, 1 Draw

Win Points - Winning a match gives 2 points, tying a match gives 1, and losing a match gives 0. Gaining an Autonomous Win Point gives 1 point.

Autonomous Points - Winning auton gives 6 points, tying gives 3, losing gives 0

Strength of Schedule Points - The combined scores of the losing alliances in a match. For example, if we lose a match with 10 points scored, we have 10 SP. If we then win a match with 20 points, and the losing team gets 5 points, we will have 15 SP.

OPR - How much each team contributes to the opposing alliances scores (higher is better)

DPR - How much teams score when you are against them (lower is better)

CCWM - The team's total contribution to their alliances (higher is better) calculated

Qualification Matches:

Alliance Partner	Opponents	Result	Score	Analysis
321E	1413C 60441T	Loss	10-41	<p>Autonomous: We were in the red side negative corner. Our autonomous program that put one ring on the alliance stake worked. We however lost auton.</p> <p>Driver Control: We lost a little time because we start driver control with our mogo mech inwards. It took 30 seconds for us to get a full stake. 45 seconds into the match, we try to release our stake but our mogo mech got stuck oddly on top of rings. We got stuck in a corner as a result of that..</p>
333V	60172W 60441K	Win	22-9	<p>Autonomous: We were in the blue side positive corner. Auton worked as normal. We lost auton</p> <p>Driver Control: Our polycarbonate slipped and started skidding against the ground, stopping us from being able to intake rings :(. We were able to get five rings on at 32 seconds left and put the stack in the positive corner with 19 seconds remaining.</p>
60441C	4454C 333S	DQ	31-27	<p>Autonomous: We were in the red negative corner. Auton worked as normal. We won auton.</p> <p>Driver Control: We tried to stop a team from putting rings on a wallstake, but that lead to us putting our fully filled mobile goal into the positive corner at exactly 15 seconds, causing us to get disqualified. We need to prioritize positive corners over wall stakes.</p>

Competition Analysis

Alliance Partner	Opponents	Result	Score	Analysis
60172X	60172Z 60441A	Win	15-7	<p>Autonomous: We were in the red side negative corner. Auton worked as normal. We won auton</p> <p>Driver Control: Our intake was hitting our mobile goal, so we released and grabbed the goal to make them not collide. We had three rings on the stake 22 seconds before the match ended.</p>
60172T	60172V 60441Y	Win	10-6	<p>Autonomous: We were in the blue side positive corner. Auton worked as normal. We tied auton.</p> <p>Driver Control: We weren't able to pick up rings for 15 seconds because our intake didn't fall down at the beginning of the matches. We got six rings on the stake and put the rings in with 17 seconds remaining.</p>
20612A	355A 38535C	Loss	11-29	<p>Autonomous: We were in the red side positive corner. Our auton worked as intended. We lost auton though.</p> <p>Driver Control: It took 10 seconds to grab a mogo mech. At 56 seconds, we only had 3 rings on our stake. Our intake simply wasn't able to pick up rings efficiently.</p>

Skills Analysis (5th Place):

Driver: 35

Auton: 3

Total: 38

Although we did not have a Skills originally planned, we decided to run our regular auton and attempt driver control. The plan we quickly developed for Driver Skills was as follows: get the mobile goal on our left side, put all the rings clustered together on the left side on the goal, put the goal in the corner. After that, we repeat this same process on the right side. Then we go onto the opposing side of the field to put the 2 mobile goals in a corner. Here are two key takeaways from Skills:

1. Putting the mobile goal in ANY corner is an extra 5 points regardless if it has rings scored or not
 - a. This helped us get 20 out of the 35 of our Driver Skills points
2. A team can call a time stop if they plan to run their Skills for less than the given maximum of 1 minute. Teams with time stops rank higher if there is a tie
 - a. We called a time stop at 3 seconds for our Auton Skills because it only ran for 1 second.

Overall Competition Analysis:

Compared to previous competitions, we were severely underprepared. Here are some of the overall problems we encountered:

- Because we did not have a mechanism to take the rings out of the positive corners, it required a significant amount of time to place the mobile goals in the positive goals
- Our intake has several problems:
 - The polycarbonate is changing angle mid-match which makes picking up rings highly inconsistent. Additionally, sometimes the polycarbonate is skidding/brushing against the floor.
 - Sometimes the rings also come out of intake
 - Intake is very slow
- We did not have a well-programmed auton due to time constraints, so we were unable to get solo AWP
 - When making our timeline, we feel we overestimated the amount of times we could meet as juniors in high school, taking up to 5 AP's and partaking in other extracurriculars. We will remedy this in our next gantt chart.
- The mogo mech at the competition was different than the one at the basement, so we had to add an extra lip to account for the different shape (see picture below)



Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
					1	2 Lake Park Competition
3	4	5 Basement @ 10:00 am (Competition Analysis)	6 Basement @ 2:00 pm (Fix 1st Stage of Intake)	7	8 Basement @ 4:00 pm (Continue 1st Stage)	9
10	11	12 Basement @ 3:00 pm (Improve Intake)	13	14 Basement @ 4:00 pm (Improve Intake)	15 Basement @ 4:00 pm (Improve Intake)	16 Basement @ 10:00 am (Try to finish fixing intake, see if we can add tracking wheels for PID)
17	18	19	20	21 Leaving after school for Speedway!	22 Speedway	23 Speedway
24	25	26	27	28	29	30
		Upcoming Competitions: November 22nd @ Speedway Signature Event				

Gantt Chart 2

Time Management

Goal: Plan out our project plans for the competition on December 7th + Speedway on November 21st

Meeting	Fix First Stage	Fix Second Stage	Build Tracking Wheels + Mount Sensors	Calibrate PID	Program Auton	Driver Practice	Speedway	Program Skills	Tweaks	Dec. 7th Comp
1										
2										
3										
4										
5										
6										
7										
8										
9										
10										
11										
12										
13										
14										

Intake V2

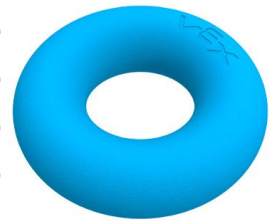
Define the Problem

Goal: Create an improved intake mechanism that can efficiently suck up rings.

Problem Statement: We need an intake mechanism that can pick up rings consistently and fast. In the past, we've had problems with the polycarbonate angle on our intake changing leading to skidding and inconsistency in picking up rings. Additionally, the rings sometimes escape second stage intake. Lastly, our current intake is on the slower side.

Solution Requirements:

- Only suck up 2 rings at a time
- Be able to score on mobile goals and alliance stakes
- Must use legal VEX V5 Competition parts



Solution Goals:

- Be able to pick up rings efficiently and fast
 - Polycarbonate angle is stable
 - Rings are not completing the transition from stage one to stage two of intake effectively (see QR code)
 - Be able to pick a ring and put in on the mobile goal, on average, in 1 second

Note: For Intake V2, we're only focusing on scoring on mobile goals (and alliance stakes). We believe these are the most valuable for now because their point value can double. As a result, it is important we have a strong mechanism that score on these goals.

Goal: Brainstorm ideas for our improved intake mechanism.

Online Research (Videos are linked):

- [34020A](#)
- [1138B - Part 1](#)
- [1138B - Part 2](#)

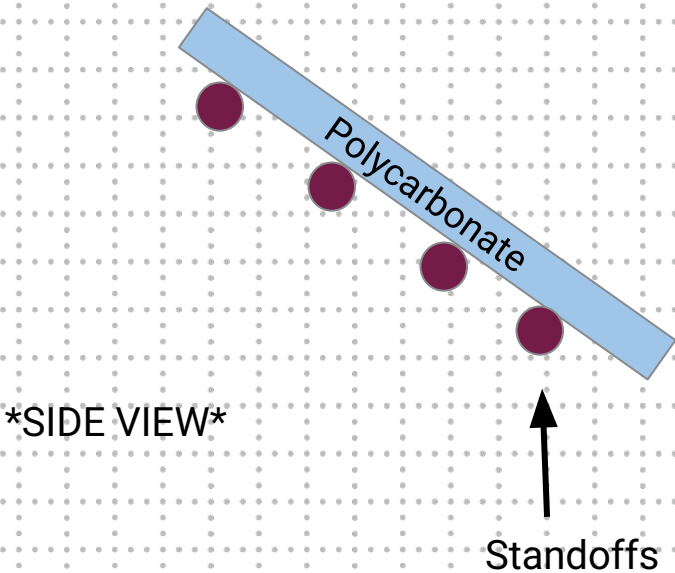
While researching, we found a few common trends:

- Using a separate motor for each stage of intake
- Flex wheels on 2nd stage intake
- Flaps above 2nd stage intake to prevent rings from escaping
- Thick treads for more ring stability

Possible Solutions:

- Add multiple standoffs directly under the polycarbonate to support it

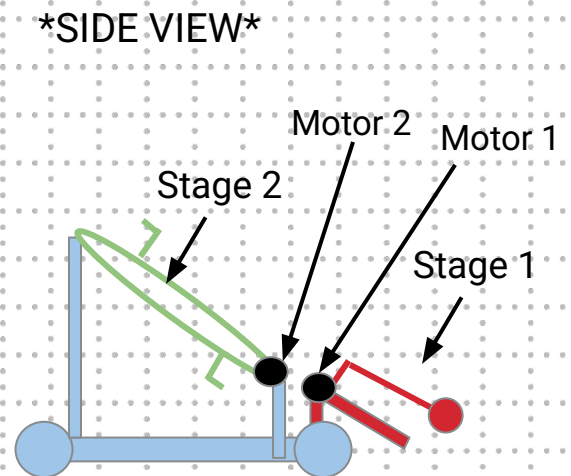
Positives	Negatives
→ Polycarb will not be allowed to change angle; very sturdy	→ Standoffs have to be mounted extremely sturdy otherwise polycarb will not be well-supported



Brainstorming Solutions

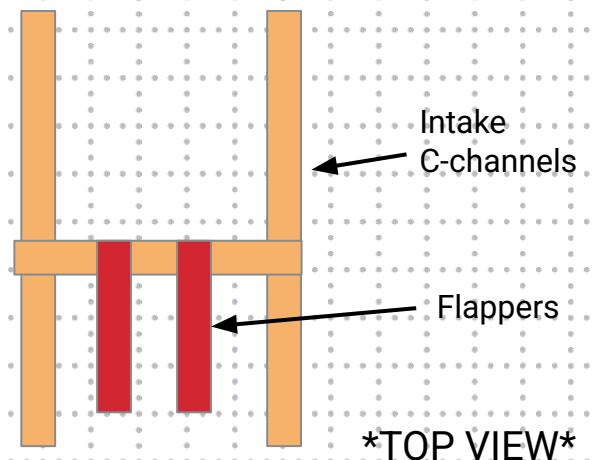
- Separate intake into stage 1 and stage 2 - have each stage controlled by an individual motor

Positives	Negatives
<ul style="list-style-type: none"> → Each stage can move independent of the other → One stage getting stuck does not lead to other stage getting stuck → More torque 	<ul style="list-style-type: none"> → Losing 11W of power that could otherwise be used somewhere else



- Put polycarb flappers on top of intake to prevent rings from escaping

Positives	Negatives
<ul style="list-style-type: none"> → Simple and robust 	<ul style="list-style-type: none"> → Hard to find material (need really thin polycarb for flexibility)



Intake V2

Select and Plan

Goal: Select the design of our improved intake and make a plan for what changes we are going to make.

Decision Matrix:

Design	Standoffs	2 Stages	Flappers
Helpfulness	4 Keeping the polycarb at a constant angle was a huge problem!	4 Intake getting stuck because of one ring was a huge problem!	4 Rings escaping intake after being sucked in was a huge problem!
Buildability	3 Standoffs may need to be secured in hard to reach spots	3 Motor mounting may be difficult	4 Very simple
Materials	4 Just standoffs	2 One extra 11W motor	2 Hard to find ultra thin polycarbonate
Total	11	9	10

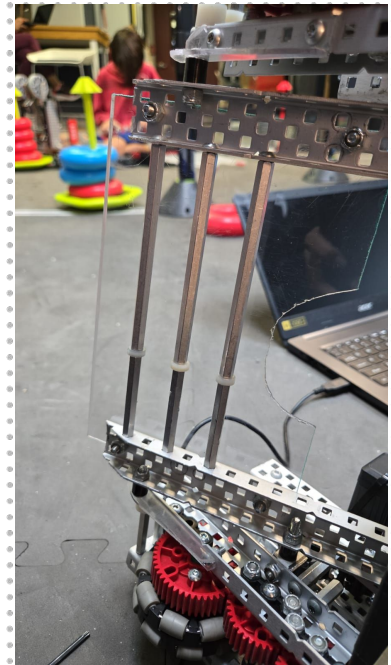
The Plan:

Because all 3 solutions scored very similarly on the decision matrix, we decided to implement all of them! All the solutions tackle different problems with our initial intake, so this is an efficient plan. We're going to add standoffs underneath the intake polycarbonate to keep it at a constant angle, separate intake into 2 stages that work independently of each other (each with their own motor), and add flappers to stage 1 to prevent rings from escaping after being intaked.

Goal: Build and implement features to improve our intake.

1. Attach standoffs underneath polycarbonate
 - a. Attach a 6" standoff to the inner side of the left intake c-channel using a 1" screw
 - b. Attach a 3" standoff to the inner side of the right intake c-channel by using a 1/2" screw
 - c. Use a 1/8" spacer and headless screw to connect the two standoffs
 - d. Repeat steps a-c two more times

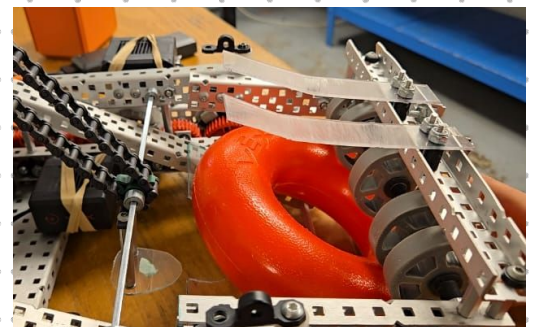
(Utilize the hole pattern displayed to the right)



This is what the implemented standoff supports look like.

2. Add flappers
 - a. Cut out a 1x6" rectangle of 1/16" thick polycarbonate
 - i. Bend it upwards to ensure flexibility
 - b. Use a 1/2" screw with a 1/4" spacer to attach the polycarb to the c-channel that lays over the set of flex wheels
 - c. Repeat a-b for a second flapper

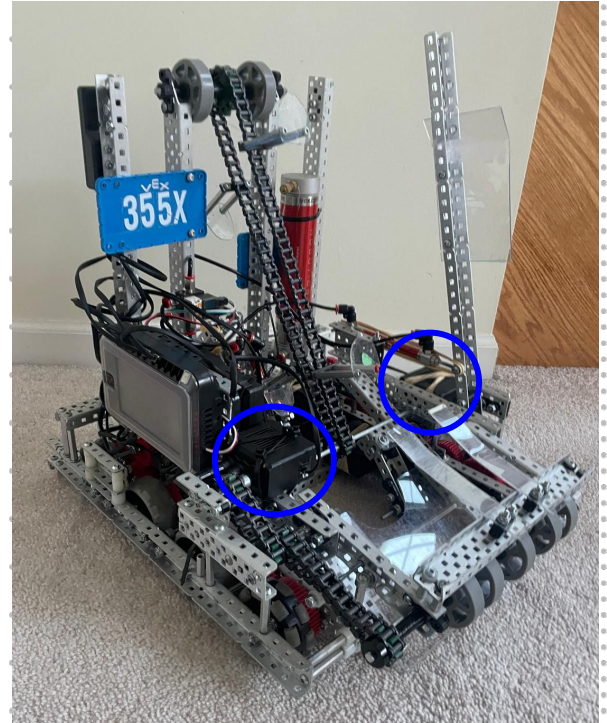
(Utilize the hole pattern displayed to the right)



This is what the attached flappers look like.

3. Separate intake into 2 stages

- a. Use four 12T sprockets + chain for the 1st stage motor
 - i. Controls initial set of flex wheels
- b. For the 2nd stage motor, directly connect the motor to the axle upholding stage 2
 - i. Controls conveyor belt



Goal: Test our improved intake mechanism.

Testing Procedure:

We ended up not properly testing our new improved intake because we were being rushed for time in order to prepare for Speedway. We didn't really meet the week before Speedway since we had a lot of tests. As an informal test, we drove around the field and picked up as many rings as we could as fast as we could. We found that the polycarbonate no longer skidded on the floor, rings no longer got stuck, and rings no longer escaped intake which means our new and improved intake met the requirements and goals we had set for it.

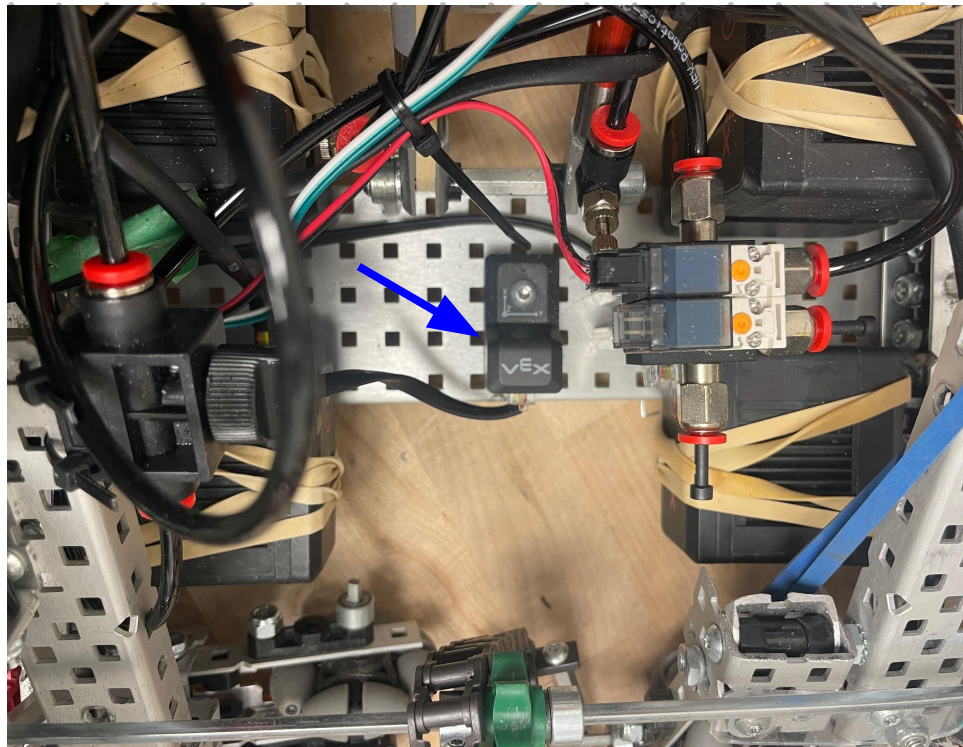
LemLib Drivetrain Setup

Build and Implement

Goal: Set-up the odometry system for our robot.

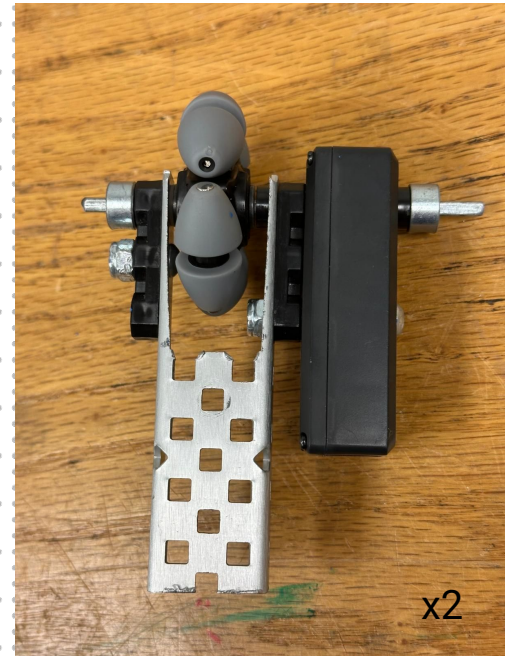
Inertial Sensor:

We want to mount the inertial sensor in the middle of our robot to receive the most accurate heading measurements. As a result, we mounted it on the base 1x5x30x1 c-channel that connects both sides of our drivetrain. Its orientation and placement is shown in the image below.

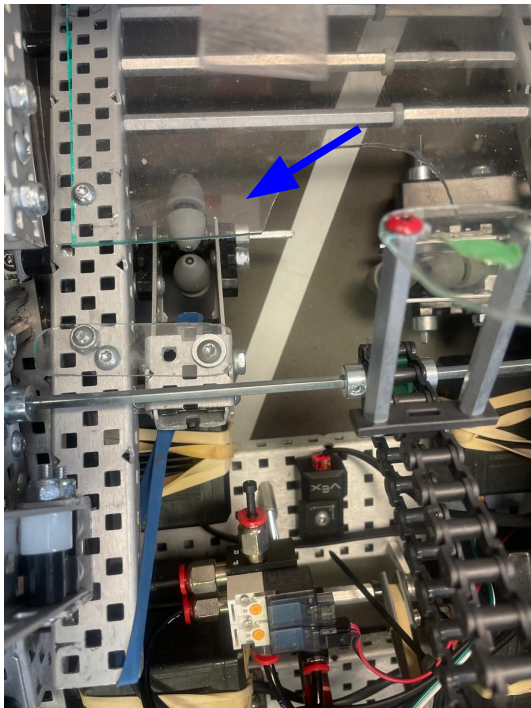


Tracking Wheels:

1. Separately build each tracking wheel
 - a. Cut a 1x2x1x8 c-channel
 - b. Cut a 2x4 rectangle out of the base of the c-channel
 - c. Use flat bearings on the outer sides of the c-channel
 - d. Use a 3- inch axle to connect the 2" omni-wheel to the c-channel and rotational sensor
 - e. Secure the axle with 2 flat bearings, one on either side
 - f. Repeat steps a-e to make the 2nd tracking wheel



This is what the assembled tracking wheel looks like.



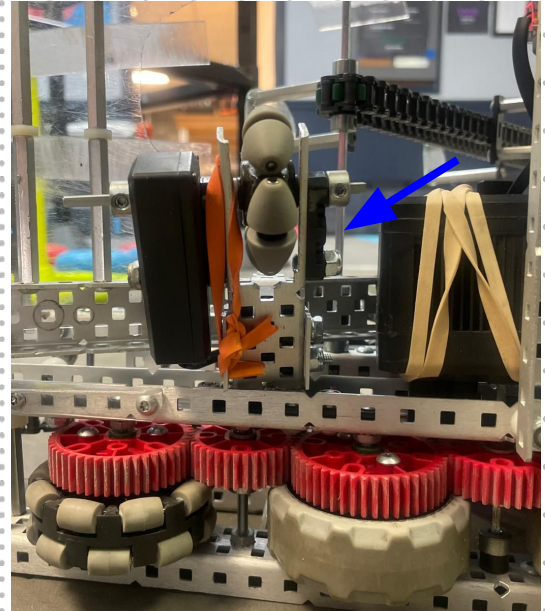
This is what the mounted vertical tracking wheel looks like.

2. Mount vertical tracking wheel
 - a. Cut out a 0.5x2 inch polycarbonate rectangle
 - b. Use ½" screws to attach that piece to both the tracking wheel and the intake c-channel that the intake polycarb is resting on (drill holes in the polycarb as necessary for attachment)
 - c. Attach a rubber band to the tracking wheel c-channel and upper intake-channel
 - i. Allows the tracking wheel to stay on ground at all times

3. Mount horizontal tracking wheel

- a. Attach the convex surface of the c-channel underneath the top lip of the right inner drive train c-channel

(The rubber band utilized to the right is not necessary and was used solely for testing purposes.)



This is what the mounted horizontal tracking wheel looks like.

Programming Odometry + PID:

```
//SENSORS
pros::Rotation rotation_vert(16); //vertical rotational sensor
pros::Rotation rotation_horz(18); //horizontal rotational sensor
pros::Imu imu(12); //inertial sensor

//TRACKING WHEELS
//negative is left of center, positive is right of center
//Track width for horz. tracking wheel (distance from tracking center) is -5.75
lemlib::TrackingWheel horizontal_tracking_wheel(&rotation_horz, lemlib::Omniwheel::NEW_2, -5.75);
//Track width for vert. tracking wheel (distance from tracking center) is -2.5
lemlib::TrackingWheel vertical_tracking_wheel(&rotation_vert, lemlib::Omniwheel::NEW_2, -2.5);

//ODOMETRY SETTINGS
lemlib::OdomSensors sensors(&vertical_tracking_wheel, // vertical tracking wheel 1, set to null
                           nullptr, // vertical tracking wheel 2, set to nullptr as we are using IMEs
                           &horizontal_tracking_wheel, // horizontal tracking wheel 1
                           nullptr, // horizontal tracking wheel 2, set to nullptr as we don't have a second one
                           &imu // inertial sensor
);
```

This is the code we wrote to initialize the sensors and tracking wheels.

```
//TODO: Calibrate PID

//lateral PID controller
lemLib::ControllerSettings lateral_controller(10, // proportional gain (kP)
0, // integral gain (kI)
3, // derivative gain (kD)
0, // anti windup
0, // small error range, in inches
0, // small error range timeout, in milliseconds
0, // large error range, in inches
0, // large error range timeout, in milliseconds
0 // maximum acceleration (slew)
);

//angular PID controller
lemLib::ControllerSettings angular_controller(2, // proportional gain (kP)
0, // integral gain (kI)
10, // derivative gain (kD)
0, // anti windup
0, // small error range, in inches
0, // small error range timeout, in milliseconds
0, // large error range, in inches
0, // large error range timeout, in milliseconds
0 // maximum acceleration (slew)
);
```

The following commands set-up the angular and lateral controls which utilize PID. We plan on tuning these values during the testing phase.

Goal: In order to achieve maximum system performance, we need to tune our PID.

Important Terms:

kP (Proportional Component)

- How quickly the system reacts to an error
 - Higher kP leads to a quicker response but a more likelihood of overshoot

kD (Derivative Component)

- Anticipates future error based on rate of change
 - Dampens oscillations + improves stability and reduces overshoot

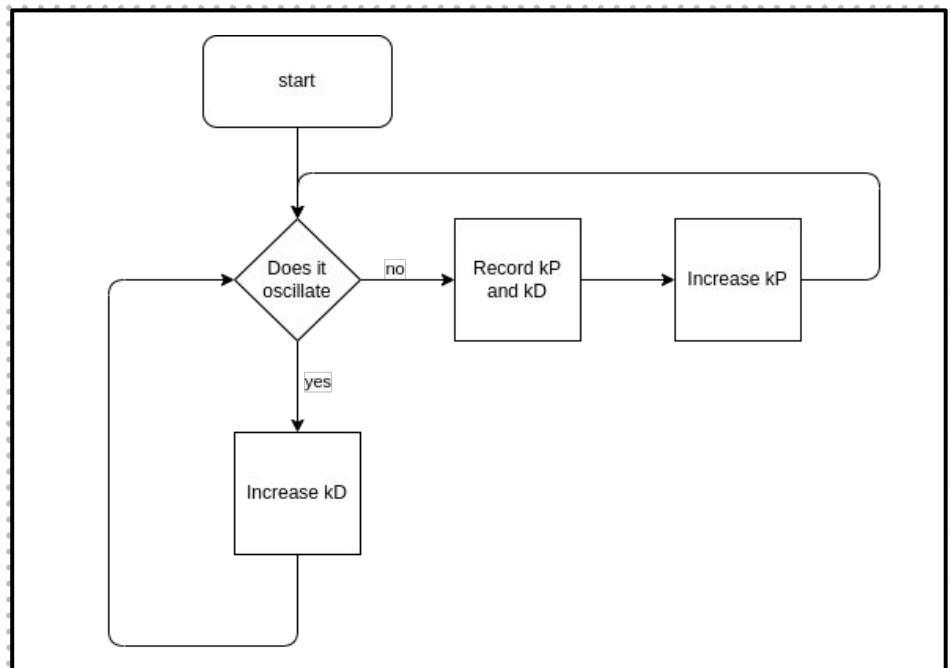
kI (Integral Component)

- Determines how much the controller responds to the accumulated error over time
 - Higher kI leads to stronger corrections applied to past errors

Procedure for Tuning Angular + Lateral PID:

The following testing procedure was provided in the LemLib documentation and is the same one we used.

Note that this cycle is used twice, once for the angular PID values and a second time for the lateral PID values. It is recommended kI is not tuned unless absolutely necessary.



Data Collection:

Angular Tuning Data

Observations	kP	kD
Looked perfect	2	10
Oscilated a little	3	10
Oscilated a lot	4	10
Oscilated slightly less	4	11
Still oscilated a lot	4	13
Oscilated less	4	16
Unnoticable change	4	18
Little oscilation	4	23
Barely any oscilation	4	28
Almost Perfect	4	30
Medium oscilation	5	30
Less oscilation	5	32
Maybe worse?	5	35
Less oscilation	5	39
Still some oscilation	5	42
Perfect	5	44

Lateral Tuning Data

Observations	kP	kD
Medium oscilation	10	3
Less oscilation	10	5
Unnoticeable difference	10	7
Less oscilation	10	12
Slightly less oscilation	10	17
Very little oscilation	10	22
Slightly less oscilation	10	25
Almost no oscilation	10	28
Seemed worse	10	30
Almost perfect, some drift at the end	10	33
More drift at end	11	33
No more drift, oscilation returned	12	33
Overshot by a lot, then slowly drove back to correct position	12	38
No more overshooting, very minor oscilation	12	45

Because kP and kD could be tuned to a satisfactory state for both the angular and lateral controls, we do not need to tune kI.

Analysis/Conclusion:

For our angular controls, a kP value of 5 and kD value of 44 worked the best. For our lateral controls, a kP value of 12 and kD value of 45 worked the best. These values will allow us to utilize PID in our autonomous programs and make them more consistent, accurate, and precise.

Competition 2

11/22/24 - 11/23/24



HEXA-FORCE

Speedway Signature Event

Competition Analysis

Goal: To place top 50% out of 100 of the top teams in the world

Autonomous Update: Going into the tournament, we didn't have a good auton. This was mostly due to time restraints. However, we did find some benefits with this. We had an autonomous that we programmed on the fly, making us adaptable and compatible of any team. However, we found this very draining, and we were always rushing between runs. We'd like to have a true auton before our next competition

Qualification Matches - 54th Place of 100

Skills - 56th Place out of 86

Ranking Data Post-Competition:

4-4-0	OPR - 12.11 (Offensive Power Rating)
9 WP - (win points)	DPR - 12.35 (Defensive Power Rating)
24 AP - (alliance points)	CCWM - -0.24 (Calculated Contributed to Winning Margin)
140 SP - (Strength of Schedule Points)	True Skill Ranking -
Skills Score: 38 (35 Driver, 3 Auton)	World Skills Ranking -

What is our Ranking Data?

1-1-1 → 1 Win, 1 Loss, 1 Draw

Win Points - Winning a match gives 2 points, tying a match gives 1, and losing a match gives 0. Gaining an Autonomous Win Point gives 1 point.

Autonomous Points - Winning auton gives 6 points, tying gives 3, losing gives 0

Strength of Schedule Points - The combined scores of the losing alliances in all matches. For example, if we lose a match with 10 points scored, we have 10 SP. If we then win a match with 20 points, and the losing team gets 5 points, we will have 15 SP.

OPR - How much each team contributes to the opposing alliances scores (higher is better)

DPR - How much teams score when you are against them (lower is better)

CCWM - The team's total contribution to their alliances (higher is better) calculated

Competition Analysis

Qualification Matches:

Alliance Partner	<u>9065P</u> (Match linked)	<p>Auton (Blue Negative): We successfully grabbed a mobile goal and put one preload on it. We won auton. The auton bonus is 6 points, which was our margin of victory that match.</p> <p>Driver Control (DC): Pathing is initially good, but 10 seconds in, we accidentally grab a red ring, and while trying to get away from it, we back into the tower. Hitting the tower makes us lose our grip on the mobile goal. We even let go of the goal to clear out the corner. Once in the corner we accidentally spin our intake containing two red rings the wrong way. Luckily our intake didn't work and the ring was flung off, but if it did it would have been 6 points to the other team, tying us. It took us 18 seconds to put 5 rings in the positive corner (4 in DC). The red alliance attempt wallstakes at 35 seconds remaining. It is unclear if we could have stopped them. We grab the last mobile goal (no rings on) with 12 seconds remaining. We pick up a ring but it is knocked out due to pushing by other robots. That was a loss of three potential points.</p>
Opponents	17788F 73641R	
Result	Win	
Score	23-17	

Alliance Partner	<u>100A</u> (Match linked)	<p>Auton (Red Negative): We put two red rings on a stake and touch the tower. We get an AWP this match thanks to our alliance</p> <p>Driver Control (DC): Our less experience becomes apparent in this match. Our alliance partner moves smoothly, while it takes us 8 seconds to get from the tower to the red positive corner. The other alliance robot takes 5 seconds to get to the same corner from the diagonal opposite of the field, going around the tower. That robot actually is able to get into the corner, but they move back for an unknown reason, and we are able to defend it again. With 35 seconds remaining, we (for what the writer is assuming to be for point scoring reasons) completely leave the positive corner undefended. The corner is almost stolen from us by a full stake - if we lost the corner the score would have been 29-29, a tie. In the endgame seconds, we put a goal with one blue ring into the negative corner. It takes us around 8 seconds to do so</p>
Opponents	85142Z 3150C	
Result	Win	
Score	35-11	

Competition Analysis

Qualification Matches:

Alliance Partner	<u>2011K</u> (Match linked)	<p>Auton (Blue Positive): We put one ring on the mobile goal. When trying to grab the second ring we rammed into the wall and got both rings in the stack stuck in our intake. We positioned our mobile goal in the corner as setup for driver control. We did not get the six point auton bonus, which was the margin that we lost by. Getting the 2nd ring on the stake would have also helped us.</p> <p>Driver Control (DC): In contrast to last match, our driving skills are incredible this match. The opposing alliance and us both immediately go for the contested stake, but we are able to grab it because we didn't have to turn around. Our alliance partner put their already filled stake from auton into the goal. While they are occupied with the goal, we smoothly put 4 rings onto a stake in 5 seconds. Us and our alliance team then switch off guarding the positive corner. With 60 seconds left, 11101B tries to get into the positive corner, but we don't let them. The rest of the game is left to 2011K and 11101B fighting for wall stakes. There is not much that we could have done better in driver control during this match.</p>
Opponents	11101B 6210T	
Result	Loss	
Score	33-39	

Alliance Partner	<u>2719A</u> (Match linked)	<p>Auton (Red Positive): We attempted to get an AWP but we missed the alliance stake. If we got it we would have gotten the autonomous bonus of 6 points, and our margin of loss was by seven.</p> <p>Driver Control (DC): The other alliance is able to grab the contested stake before us. We spend much of the match fighting for the positive corners because we were too slow to get one. We did nothing and appeared to lose control of our robot in the last seconds of the match as the robot didn't respond to inputs. If we had gotten the corners earlier, won auton, and moved in the last 15 seconds, we may have won this match. Apart from that no notes, driving and ring pick up time was average.</p>
Opponents	1165A 1115E	
Result	Loss	
Score	17-24	

Qualification Matches:

Alliance Partner	<u>6008D</u> (Match linked)	<p>Auton (Blue Negative): We put one ring on the mobile goal and put that goal near the negative corner since the code was mirrored from. We miss the ring on the alliance stake, but nothing would have saved auton anyways.</p> <p>Driver Control (DC): Our intake stopped working during the match. While we were trying to make it work, the other alliance took both positive corners. The issue here wasn't really a strategical one (although there were some strategical and driver mistakes), it was a mechanical one. We also got stuck on top of rings 36 seconds in. We likely would have lost due to pathing, but the reason this match was lost was the intake.</p>
Opponents	299V	
Result	Loss	
Score	3-40	

Alliance Partner	<u>8675V</u> (Match linked)	<p>Auton (Blue Positive): We put one ring on the mobile goal. When trying to put the preload on, we turned too fast, disrupting the rings position and causing us to only get one ring rather than two. We positioned our mobile goal in the corner as setup for driver control. We won auton as well</p> <p>Driver Control (DC): We immediately grab the contested stake - good. We get some rings on the stake - also good. We leave the positive corner completely undefended - incredibly bad. There were multiple seconds where the opposing alliance could have taken both positive corners. We then tried to get into the opposing color's corner while the opposing color was already there, and the other robot tried to get our corner. They did succeed, but luckily for us they didn't seem interested in holding the positive corners either. We steal a fully filled red stack 40 seconds in and put it in the negative corner, and our partner puts a fully filled stake in the positive corner with exactly 15 seconds remaining. We won, but it was pretty lucky and we need to remember to watch the positive corners.</p>
Opponents	1239E 99904C	
Result	Win	
Score	39-10	

Competition Analysis

Qualification Matches:

Alliance Partner	<u>45618A</u> (Match linked)	<p>Auton (Red Positive): We successfully put two rings on our mobile goal and put it near the positive corner. Blue won auton however</p> <p>Driver Control (DC): It takes up 49 seconds to fill up a stake. The biggest reason we lost this match was overcrowding. We had two stakes in the corner with one ring each, but we really wanted the full stake in the corner. The overcrowding of stakes and rings caused us to tip the stack, losing our change of getting 8 points. With 38 seconds left, we try to steal a positive blue goal. We successfully grab it with 29 seconds remaining, but we are blocked going into the negative corner. Instead of moving to the opposite negative corner where we could have easily gotten the goal into, we spent our time on a fruitless task. This is another case where the drive team could have been more used instead of all tunnel visioning on one corner.</p>
Opponents	2140C 1115B	
Result	Loss	
Score	18-34	
Alliance Partner	<u>6842V</u> (Match linked)	<p>Auton (Red Negative): We put one ring on the mobile goal and put that goal near the negative corner since the code was mirrored from. When trying to put the preload on, we turned too fast, disrupting the ring's position and causing us to only get one ring rather than two. We won auton and got the 6 point bonus.</p> <p>Driver Control (DC): We immediately head for our positive corner and clear it out 7 seconds in. Our alliance partner doesn't move until 19 seconds in. By then we have 3 rings on our stake. Our alliance partner fills up a stake in 30 seconds and we swap stakes in the corner (4 ring to 6 ring stake) 37 seconds in. We leave the positive corner completely unguarded for some seconds, and we get lucky that the opposing robot closest doesn't take advantage. The other robot further away notices, but we return to the corner in time. 45 seconds in, wallstakes are attempted by the opposing team and we do a great job pushing them away and defending them. That team goes to the other side to do wallstakes but we stalled them for a few seconds, which was really good strategically and our first successful wallstakes defense. Unsure what the purpose of our actions are in the last 15 seconds but they would be better off doing the alliance stake, which is 3 points and unfilled.</p>
Opponents	57249C 2719F	
Result	Win	
Score	40-31	

Overall Competition Analysis:

- The livestream didn't show the heads of the driveteam so it's unclear if we were watching other robots (not just "tunnel-visioning" on our robot)
- Auton related notes
 - We need to win auton. **Every single time we won auton, we won the match.** The six point bonus makes a huge difference, and we often lost by small margin
 - We would also really like to get AWP in our matches, as it means even when we lose, we win.
 - It was very tiring to constantly change our programming between matches to make an adaptable autonomous. We will try our best to not have it happen again
- Guard the positive corners needs to become our robot's name
- We should get more driver practice

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
1	2	3	4	5	6	7
8	9	10	11	12	13	14
NOTE: We did not meet for the first 3 weeks of December due to finals season. It was a very busy time for schoolwork, so we decided to take a small break from robotics to prioritize school and resume during winter break.						
15	16	17	18	19	20	21
22	23	24	25	26	27	28
				Basement @ 10:00 am (Program Positive Auton for Jan 5th)		
29	30	31				
		Basement @ 8:00 am (Finish Positive Auton for Jan 5th)				
Upcoming Competitions: January 5th @ Great Lakes January 19th @ Mundelein January 25th @ Kalahari Signature Event						

Wall Stakes Mech.

Define the Problem

Goal: Create a mechanism that can score on wall stakes.

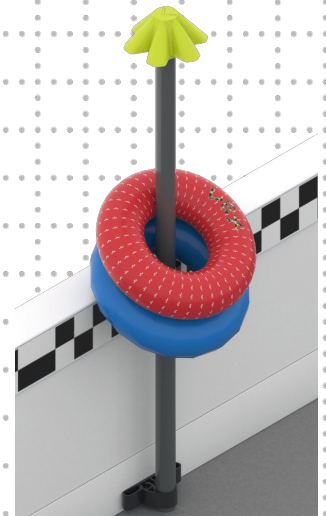
Problem Statement: We want to be able to score on wall stakes because they are another source of points that many teams can not utilize. They can be thought of as 2 extra mobile goals (without the corner modifications) which is a significant opportunity for points.

Solution Requirements:

- Must be able to take rings off of intake and place them on wall stakes
- Must use legal VEX V5 Competition parts

Solution Goals:

- Be able to put 2 rings on wall stakes at a time
- Not require too much precision
 - Taking rings off intake + putting on wall stakes should take a maximum of 4 seconds



Note: In the past (Initial Intake), we have considered wall stakes mechanisms as a part of intake when brainstorming and planning. However, we are now considering it as a separate mechanism because it can function independently of intake.

Wall Stakes Mech.

Goal: Brainstorm possible designs for a wall stakes mechanism.

Online Research (Videos are Linked):

- [Joseph973](#)
- [The Moderately Sized Wave](#)
- [2145Z - Smorg Brown](#)
- [44252A - Cockys Cola](#)
- [1412E Robotics](#)

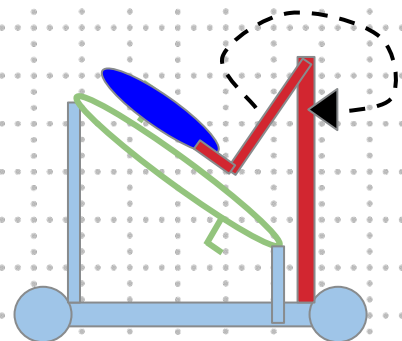
Common Trends:

- Fish Mechanism
 - Using an arm that can travel 360 degrees to take rings off of intake. ("Fishing" rings off of intake.)
- Lady Brown
 - Arm with a container that rests at the top of intake to collect rings
- Alignment mechanism that lines up to the circumference of wall stakes

Possible Solutions:

- Fish Mechanism
 - An arm that can rotate 360 degrees to grab rings off of intake and put them onto the wall stakes

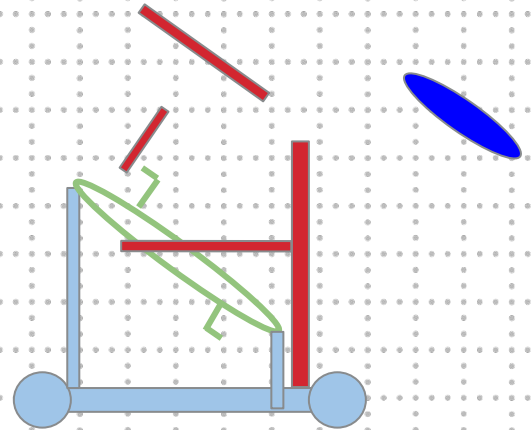
Positives	Negatives
→ Very fluid motion	→ Only one ring at a time
	→ Ring's position on intake has to be precise in order to "hook" the ring



Brainstorming Solutions

- Lady Brown
 - An arm with a container at the end that sits on top of intake
 - When arm is all the way down and resting on intake, the rings feed into the container

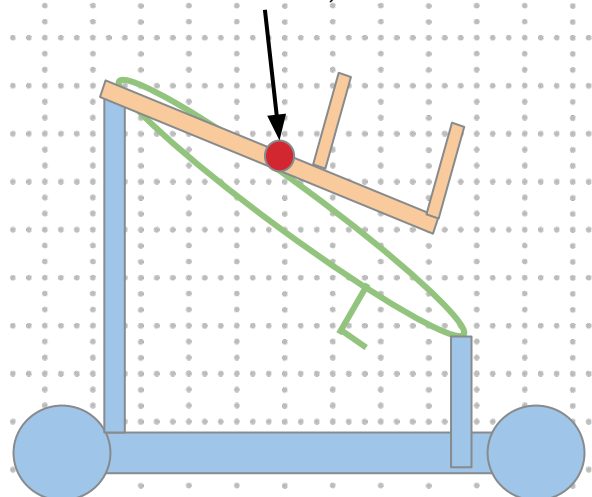
Positives	Negatives
→ Can put 2 rings at a time	→ Slow



- Redirect Intake (similar to previous idea from Initial Intake)
 - Have an arm with a container at the end. Instead of feeding into the container when arm is down, the ring feeds in when intake is reversed

Positives	Negatives
→ Can put 2 rings at a time	→ Tricky to make arm long enough for wall stakes

Polycarb hinge that can only move up and not down (allows rings to go up intake but redirects them into box when intake is reversed)



Wall Stakes Mech

Select and Plan

Goal: Select the wall stake design we want to utilize and make a plan for implementing it.

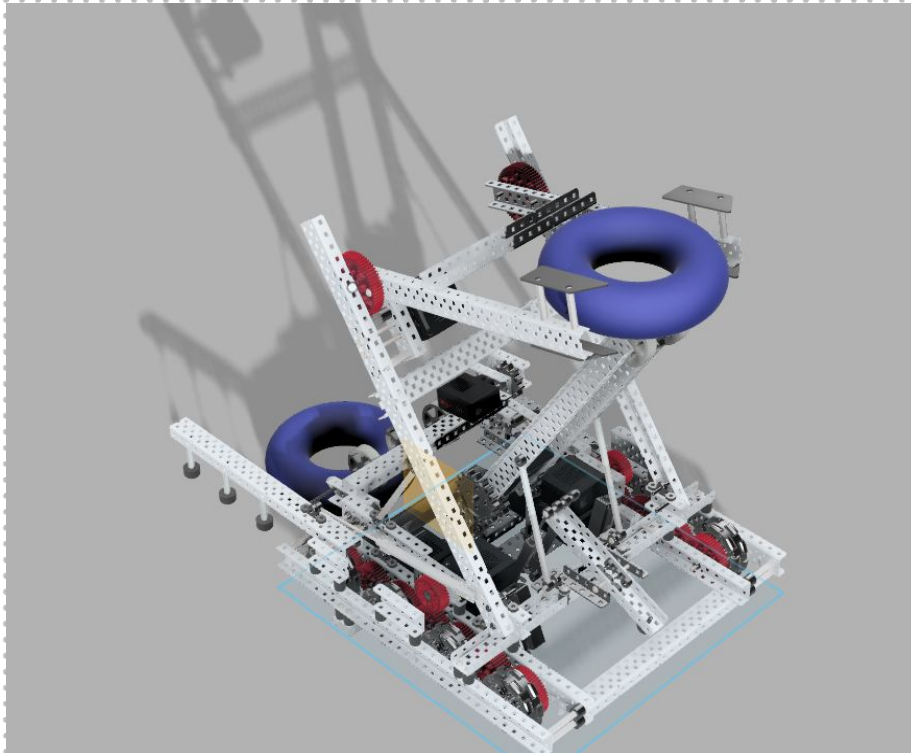
Decision Matrix:

Design	Fish Mechanism	Lady Brown	Redirect Intake
Speed	4 Very fast because only swing	4 Very fast because only lift	3 Slower because need time for redirect + lift
Efficiency	3 One ring at a time	4 Two rings at a time	4 Two rings at time
Accuracy Required	2 Rings needs to be in precise spot on intake to be hooked and lining up with wall stakes is also difficult	3 As long as arm is down, ring will go where we want but putting ring on wall stake may be tricky	3 As long as box is down, ring will go where we want but putting ring on wall stake may be tricky
Total	9	11	10

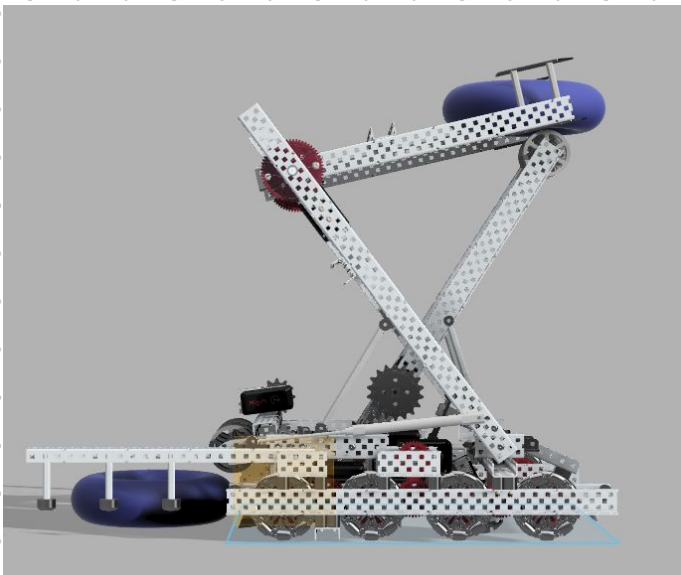
The Plan:

We plan on building a lady brown mech onto our existing intake. Our plan is to have this ready by our next competition on January 5th.

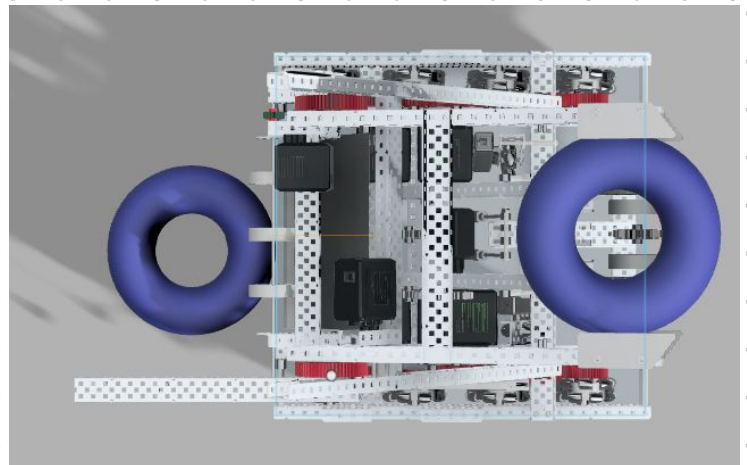
CAD Images of Lady Brown Mech:



Isometric View



Side View



Top View

Initial Auton

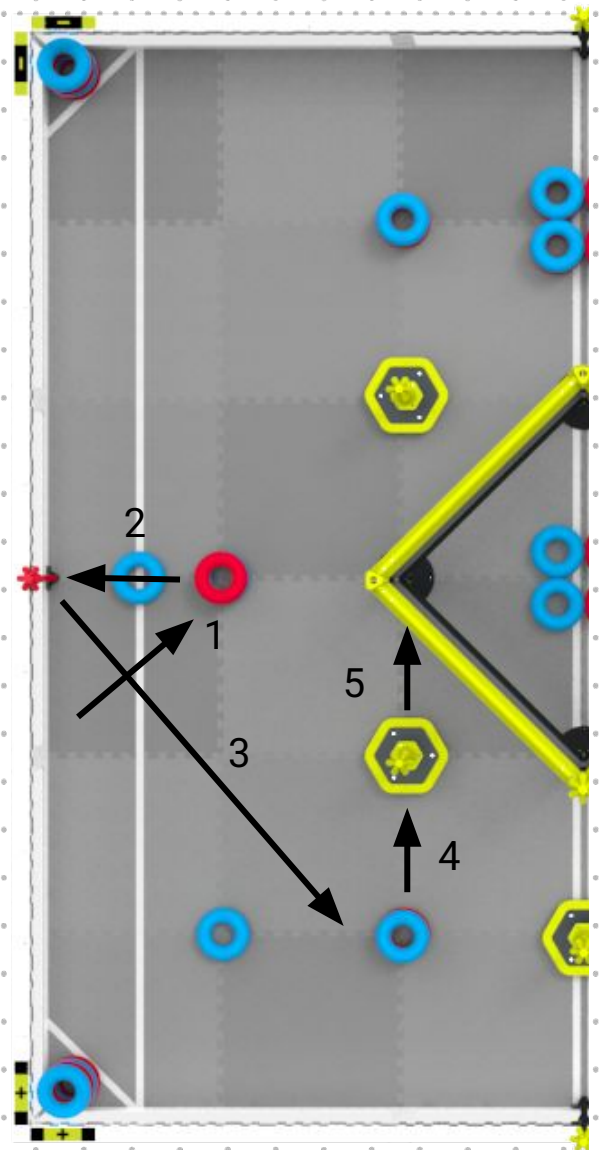
Select and Plan (Part 2)

Goal: Program a solo AWP auton.

12/31/24: We already had a Select and Plan for Initial Auton (see pgs. 75-79) but after our experience at Speedway we want to have 2 available options the we can run depending on the match. We previously selected a gold rush program, but now also want to have an auton capable of getting solo AWP. Being able to get solo AWP means we can get win points regardless of if we win the match or who our alliance partner/opponents are.

Plan for Solo AWP Auton:

1. Drive forward and pick up the top ring in the stack. This ring stays in stage 1 intake because our preload is in stage 2.
2. Turn and back into the alliance stake. Put preload onto alliance stake. (May need an extra turn to get blue ring out of the path.)
3. Turn towards the right stack and drive forward. Intake the bottom ring.
4. Turn and drive backwards to grab mobile goal. Place the 2 rings on the goal.
5. Turn towards ladder and drive towards ladder to touch it.



Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
			1 Basement @ 10:00 am (Program Negative Auton + Wall Stakes)	2 Basement @ 1:30 pm (Simplify Auton Formula + Wall Stakes)	3	4 Basement @ 9:00 am (Tweak Auton + Skills Auton + Wall Stakes)
5 Great Lakes Competition!!	6	7	8 Engineering Notebook Due for Kalahari @ 10:59 pm!	9	10	11
12	13	14	15	16	17	18 Purdue Competition!
19	20	21	22	23 Leaving for Kalahari!	24	25
26	27	28	29	30	31	
			Upcoming Competitions: January 5th @ Great Lakes January 19th @ Mundelein No longer attending because of spot availability! Instead, we are attending the Purdue competition on January 18th. January 25th @ Kalahari Signature Event			

Initial Auton

Build and Implement

Goal: Programming the positive red side auton.

Our code for AWP on the red side:

```
340 void mainAWP(int color, int sign) //1 is blue, -1 is red //1 is positive, -1 is negative
341 {
342     int shortTimeout = 1000;
343     int medTimeout = 3000;
344     int longTimeout = 5000;
345     int delay = 500;
346
347     //angleMod changes headings based on if we're blue/red (+0 or +180)
348     int angleMod = 0;
349     if (color == -1)
350     {
351         angleMod = 180; //adding 180 makes the angle opposite of what it was previously
352     }
353
354     chassis.setPose(60 * color, -24 * sign, 206.565 - angleMod); //starting position||RING CHECK: We have 1 ring
355                                     // in second stage
356     //move towards stack
357     chassis.moveToPoint(53 * color, -10 * sign, medTimeout, {.forwards = true}, true);
358
359     //start running intake when the robot is a certain distance
360     chassis.waitFor(5);
361     intake1.move(127);
362     chassis.waitForDone();
363     pros::delay(100);
364
365     //move a little more forward to grab the top ring of the stack
366     chassis.moveToPoint(52 * color, -9 * sign, medTimeout, {.forwards = true}, false); //org true
367     pros::delay(500);
368
369     //move over the one-stack opposing color ring
370     chassis.moveToPoint(55 * color, 17 * sign, longTimeout, {.forwards = false, .maxSpeed = 60}, false);
371     intake1.brake();
372     chassis.turnToHeading(270 - angleMod, shortTimeout, {}, false);
373
374
375     //back into stake for real
376     chassis.moveToPoint(66 * color, 14.5 * sign, 2000, {.forwards = false, .maxSpeed = 80}, false);
377     //Set accurate pose here
378
379     //We want to move forward 1 inch, but the other chassis has a small error range of
380     //1 inch, making it not move forward at all. We created a new chassis
381     lemLib::Pose current = chassis.getPose();
382     chassis2.setPose(current.x, current.y, current.theta);
383     //Then move a little bit more forward
384     chassis2.moveToPoint(current.x - (color * 1.25), current.y, shortTimeout, {.forwards = false, .maxSpeed = 80}, false);
385
386     //set regular chassis pose to the new chassis2 pose
387     current = chassis2.getPose();
388     chassis.setPose(current.x, current.y, current.theta);
389
390     intake2.move(127);
391     pros::delay(650);
392     intake2.brake();
393     //RING CHECK: 1 ring in stage 2 hopefully
394
395     //Part 2:
396
397     //move away from wall stake
398     chassis.moveToPoint(60 * color, 10 * sign, medTimeout, {.forwards = true}, false);
399
400     //Move towards stack of rings in centerish of corner
401     chassis.turnToPoint(40 * color, -15 * sign, shortTimeout, {}, false); //turn towards centerish stack
402     intake1.move(127);
403     chassis.moveToPoint(40 * color, -15 * sign, longTimeout, {.maxSpeed = 100, .minSpeed = 30, .earlyExitRange = 4}, true);
404     chassis.moveToPoint(24 * color, -35 * sign, longTimeout, {.maxSpeed = 50}, false);
405     pros::delay(500);
406
407     chassis.turnToPoint(16 * color, -10 * sign, shortTimeout, {.forwards = false}, false); //org y = 24
408     intake1.brake();
409     chassis.moveToPoint(16 * color, -10 * sign, shortTimeout, {.forwards = false, .maxSpeed = 60}, false);
410
411     mogoMech.set_value(true);
412
413     //touch ladder
414     chassis.turnToHeading(0 + angleMod, shortTimeout);
415     intake1.move(127);
416     intake2.move(127);
417     chassis.moveToPoint(12 * color, 0 * sign, medTimeout, {.maxSpeed = 60}, false);
418     intake1.brake();
419     intake2.brake();
420 }
```

Wall Stakes Mech

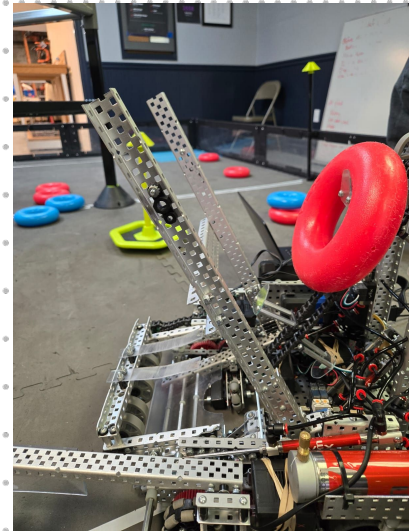
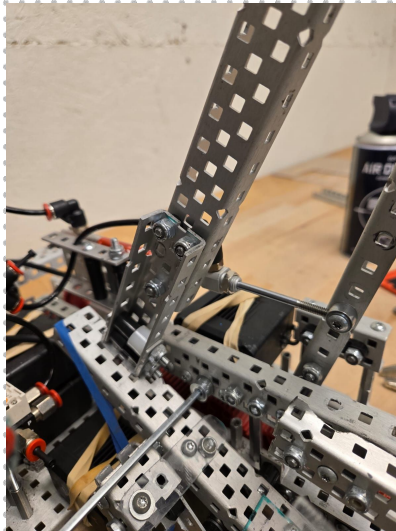
Build and Implement

Goal: Build our wall stake mechanism

Step-by-Step Process:

Step One: Take two 28 x 2 c-channels and screw them to the robot at a diagonal angle

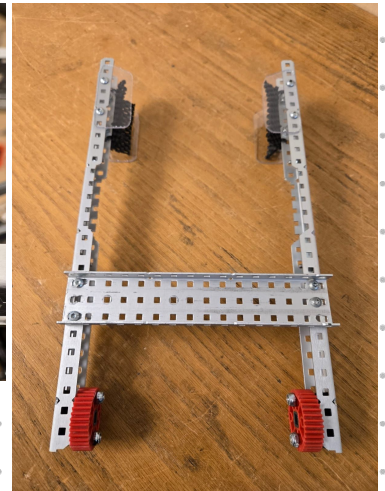
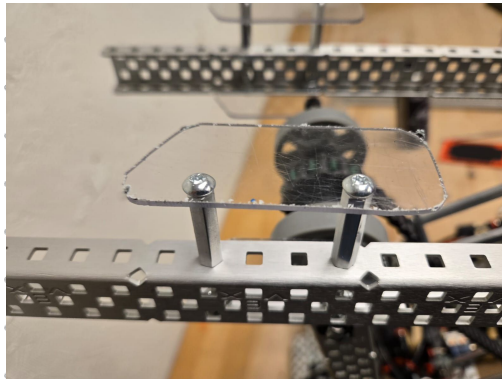
- They are on the support beams for stage one of intake
- Add spacers in between the lower c-channel for support
- Make sure to add a support 20 x 3 c-channel between the two to stop wobble



12-31-24 (Step One): We did this in tandem with programming autonomous

Step Two: Separately, create the moving part of the mechanism

- Take two 27 x 2 channels and connect them with a 16 x 3 support beam
- Put 1-inch standoffs on the beams to raise two polycarbonate trapezoids up. There are two similar trapezoids below them
- Add two 36T gears

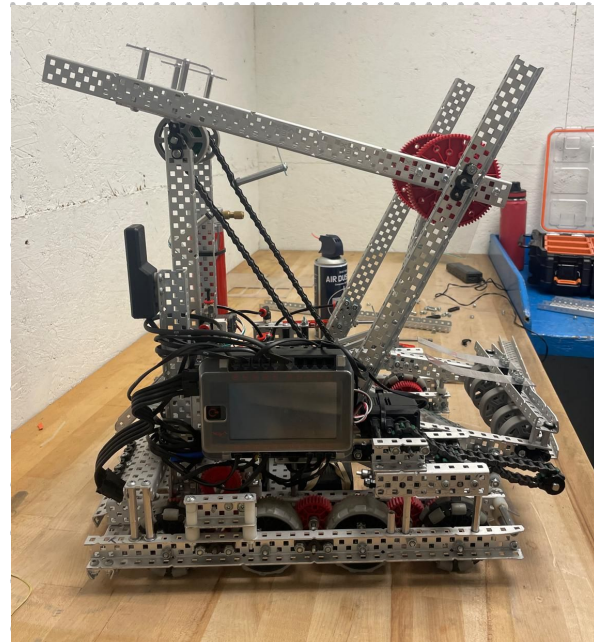


01-01-25 (Step Two): We did this in tandem with programming autonomous. Happy New Year! :D

Mini-Design Cycle 01/04/25: We used 72T gears, but the problem was that they would hit the tower. We brainstormed a solution where we reduced the size of the gears to 36T. To implement the solution, we took the whole mechanism off the robot, changed it, and put it back.

Step Three: Attach the mechanism onto the robot.

- a. Use a 15" inch axle to connect both parts of wall stakes together as show in the picture to the right



Goal: Tweak and test our AWP auton.

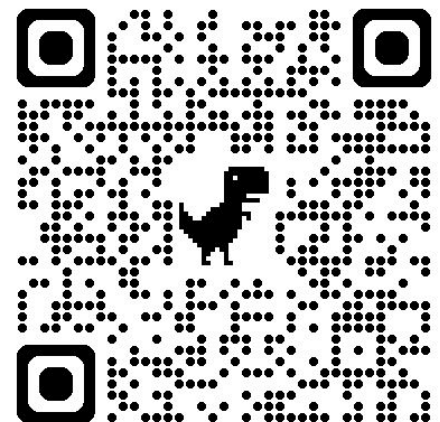
Testing Procedure:

After having the basic outline for our code, we had to make small tweaks to some of the values (i.e. velocity, coordinates, headings, etc.). We used the following process to do this:

1. Run the program
2. Take the first inaccuracy, and then run the program 3 times to see if the inaccuracy persists
 - a. If so, tweak the inaccuracy
 - i. Don't move onto the next inaccuracy until the previous one is fixed
 1. We do this to save time and be efficient with tweaking. If we don't go in the order of the inaccuracies in the program, tweaking may become redundant.
3. Stop once the program runs 3 consecutive times without inaccuracies

Results/Conclusion:

The QR Code to the right is the final tweaked version of our 7 point AWP Auton. It meets all the criteria we specified earlier. We may need to make small adjustments at competitions to accommodate the differences in fields since the field we practice on is not as sturdy as actual competition ones.

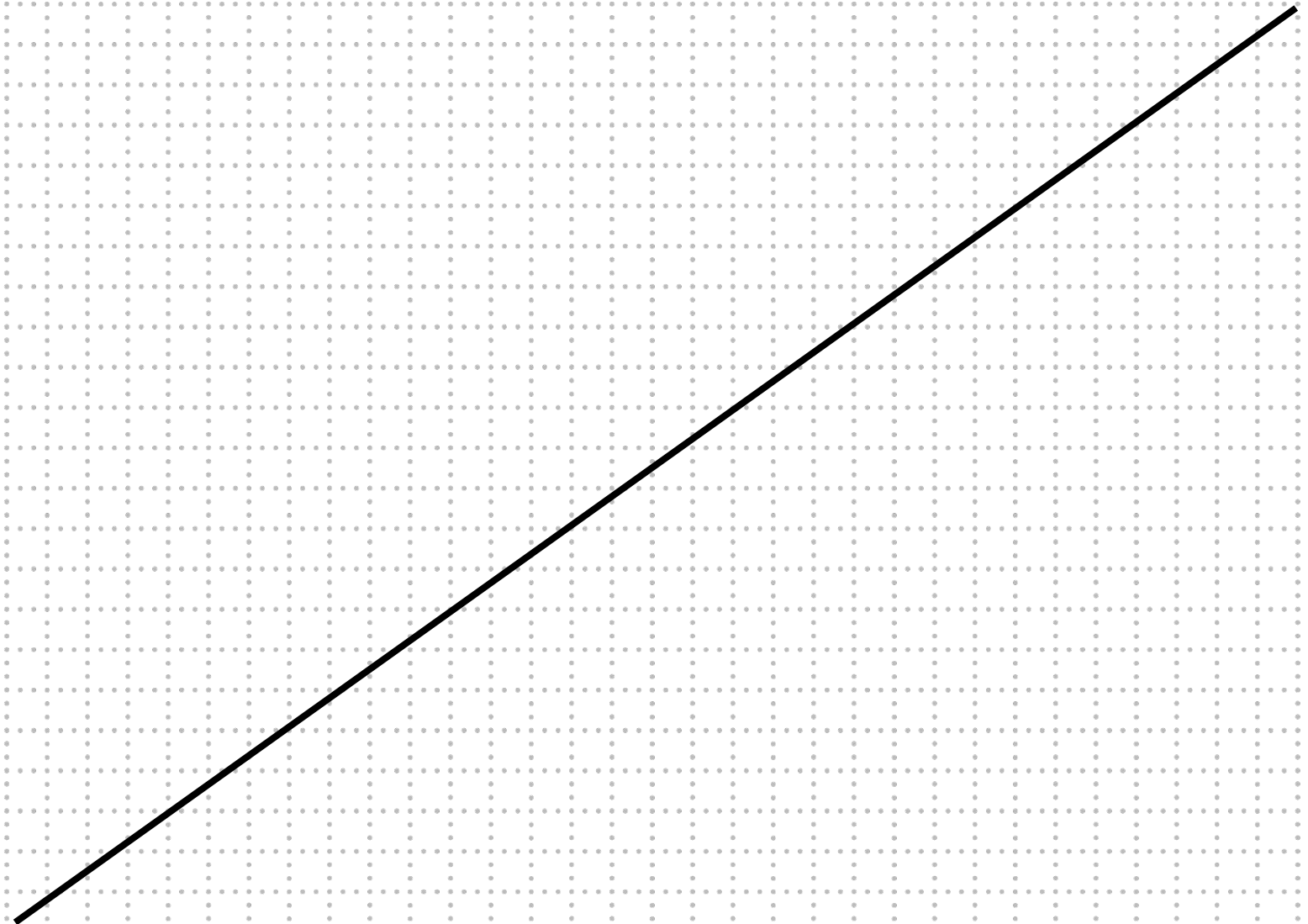


Wall Stakes Mech

Test Solution

Goal: Test our wall stakes mechanism to determine if it can score on wall stakes.

We ended up not properly testing our wall stake mechanism because we were being rushed for time in order to prepare for our Great Lakes Competition. As an informal test, we practiced taking rings off of intake and scoring them on the wall stakes. We found that we were able to do so, so the requirements we had set had been met. We plan on testing this new mechanism more at the Great Lakes Competition.



Reflecting Auton

Define the Problem

Goal: We want to replicate our auton for all 4 possible starting positions (blue positive, blue negative, red positive, and red negative).

Problem Statement: We have our auton programmed on the red positive side, but we need it to work in other starting positions as well. We could manually modify the code for each corner of the field, but this would be time-consuming, and it would be hard to keep track of changes we made to auton if they were in four different places.

Solution Requirements:

- Adjustments to auton need to be easy to make
- Changing code for one starting position should affect all starting positions
- The program should be equally accurate and consistent regardless of where the robot starts
- We should be able to download all four versions of the program to the brain quickly
- The code must remain clean and easy to read

Reflecting Auton

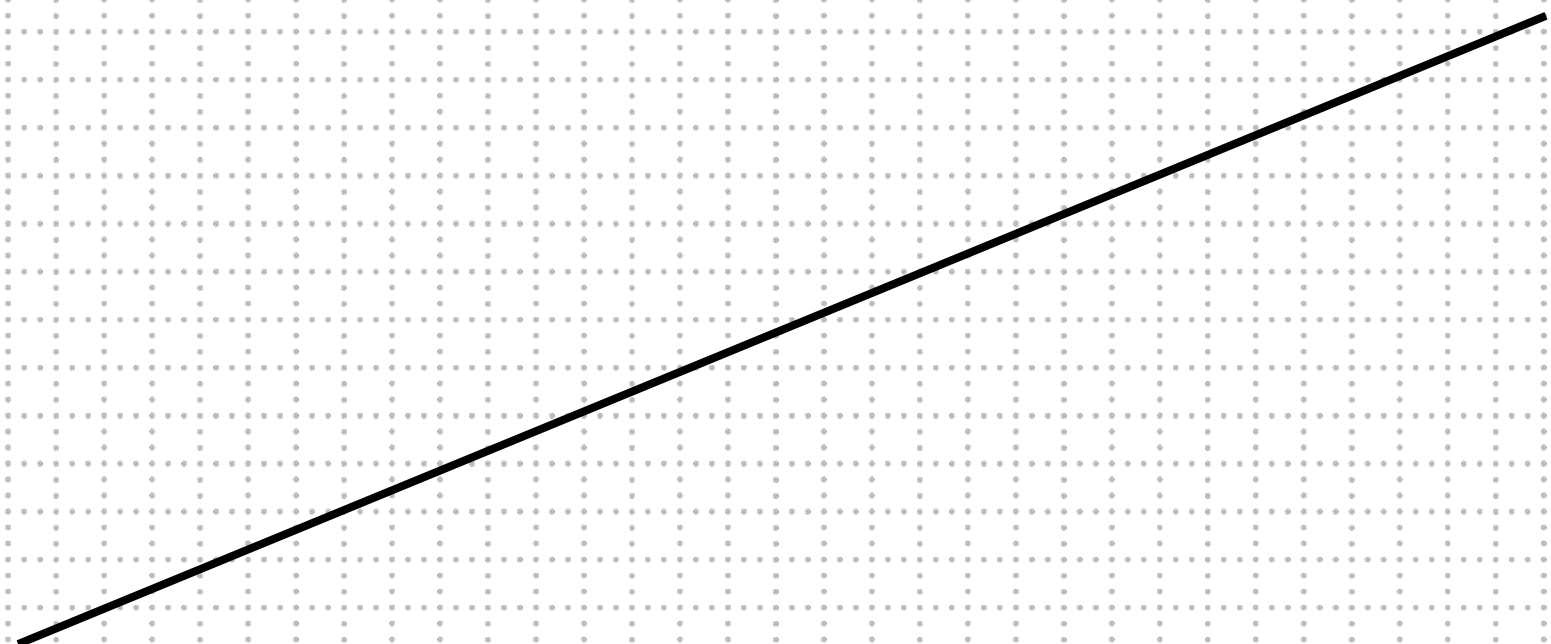
Goal: Brainstorm solutions for replicating our blue positive auton on all four sides.

Online Research:

For reflecting autons, we didn't do much online research because we have not seen other teams implement a unique solution/approach to reflecting their autons. As a result, the following brainstorming is composed of purely our original ideas.

Possible Solutions:

- Duplicate it manually (copy/paste + change all values)
 - We can copy and paste the positive blue auton to the 3 other sides and change the angles/coordinates manually
- Derive a formula
 - We can create a formula that allows the same program to be used for all four autons based on the inputted placement (sign and color)
- Utilizing path.jerryio.com + flipping tools
 - We can draw our path on path.jerryio.com and use their built in flipping tools to determine the values for the 3 remaining sides



Reflecting Auton

Goal: We want to replicate our auton for all 4 possible starting positions.

There are three aspects of our program that determine what the robot does during auton:

- X coordinates
- Y coordinates
- Angle

In order to fully reflect our auton to different corners of the field, we need to modify all of these. We decided to program our entire auton for the blue positive side of the field, then reflect it to the side we need to run by passing parameters into the *mainAWP* function. The *autonomous* function uses the *side* variable to determine which slot on the brain the program is being run from, then calls *mainAWP* with the appropriate parameters.

```
492 void autonomous() {
493     if (side == 1) //POSITIVE AUTON BLUE SIDE
494     {
495         mainAWP(1, 1);
496     }
497     else if (side == 2) //NEGATIVE AUTON BLUE SIDE
498     {
499         mainAWP(1, -1);
500     }
501     else if (side == 3) //POSITIVE AUTON RED SIDE
502     {
503         mainAWP(-1, 1);
504     }
505     else if (side == 4) //NEGATIVE AUTON RED SIDE
506     {
507         mainAWP(-1, -1);
508     }
509 }
```

mainAWP has two integer parameters: *color* and *sign*. They each serve a different purpose in reflecting coordinates and angles across the field, and they are each represented as either 1 or -1. On the blue side, *color* is set to 1. On the red side, *color* is set to -1. The *sign* variable is either positive or negative depending on whether the robot is closer to the positive or negative corner.

```
364 void mainAWP(int color, int sign)
```


Reflecting Auton

Build and Implement

Goal: We want to replicate our auton for all 4 possible starting positions.

We started by figuring out how to reflect x and y coordinates on the field. Since LemLib's coordinate plane is centered on the field, all x coordinates can be flipped to the opposite color by multiplying by -1. All y coordinates can also be flipped between the positive and negative corners by multiplying by -1. In our code, we do this by multiplying x coordinates by *color* and y coordinates by *sign*.

```
401 chassis.moveToPoint(53 * color, -10 * sign, medTimeout, {.forwards = true}, true);
```

While this solves most of the problem, there are still a few situations where the robot's heading needs to be reflected. Through trial and error, we were able to come up with a formula that reflects positive blue angles to other parts of the field using *color* and *sign*. We wrapped this formula inside a function called *angleLogic* for ease of use:

```
340 float angleLogic(int color, int sign, float posBlueAngle)
341 {
342     //SignMod -- Positive (1) is 0
343     //         -- Negative (-1) is 180
344     //ColorMod -- Blue (1) is 0
345     //         -- Red (-1) is 360
346
347     //angleMod changes headings based on if we're blue/red (+0 or +180)
348     int colorMod = 0;
349     if (color == -1)
350     {
351         colorMod = 360; //adding 180 makes the angle opposite of what it was previously
352     }
353
354     int signMod = 0;
355     if (sign == -1)
356     {
357         signMod = 180;
358     }
359
360     float angle = (-1 * color * sign * (colorMod - posBlueAngle)) + signMod;
361     return angle;
362 }
```

Here's an example from our auton:

```
419 chassis.turnToHeading(angleLogic(color, sign, 270), shortTimeout, {}, false);
```

Reflecting Auton

Test Solution

Goal: We want to replicate our auton for all 4 possible starting positions.

Throughout the process of developing our original formula, we tested it with a few example angles so we could modify it as needed. Using the starting angle of our original positive red auton, we manually determined the angle the robot would need to face at each side of the field:

Positive Blue - 333.435°

Negative Blue - 206.565°

Positive Red - 26.565°

Negative Red - 154.435°

We then plugged the positive blue angle into the formula to calculate the angle of each side:

	color	sign	colorMod	signMod	Expression	Result
Positive Blue	1	1	0	0	$(-1 * 1 * 1 * (0 - 333.435)) + 0$	333.435
Negative Blue	1	-1	0	180	$(-1 * 1 * -1 * (0 - 333.435)) + 180$	206.565
Positive Red	-1	1	360	0	$(-1 * -1 * 1 * (360 - 333.435)) + 0$	26.565
Negative Red	-1	-1	360	180	$(-1 * -1 * -1 * (360 - 333.435)) + 180$	154.435

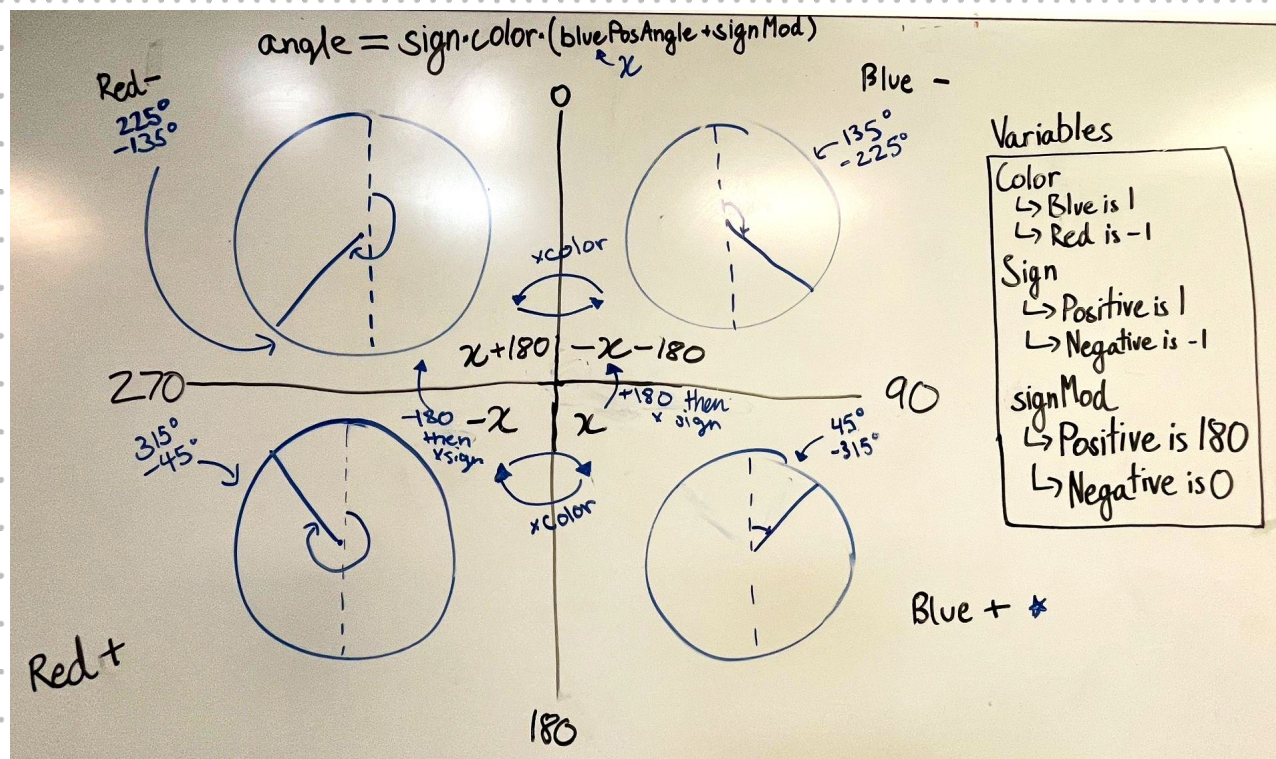
After verifying the formula for each test angle, we added it to the *angleLogic* function, updated our auton program to use it whenever an angle is defined, and tested the full program by running it on the robot from all four starting positions on the field. Although we suspected the first iteration of this function was overcomplicated, it calculated every angle correctly and assisted in the successful reflection of our auton, allowing us to run the same code no matter where the robot started.

Reflecting Auton

Build and Implement (Part 2)

Goal: We want to replicate our auton for all 4 possible starting positions.

Even though the first version of our *angleLogic* function correctly reflected angles across the field, it was unnecessarily complicated. We suspected that some parts of it were redundant due to the trial and error approach we originally took. In an effort to make our code more efficient, we decided to break down the operations required to reflect an angle across the x-axis and y-axis visually and attempt to develop a simpler version of the function.



As the diagram shows, reflecting an angle across the y-axis (flipping the color) can be done by multiplying the angle by -1. An angle can be reflected across the x-axis (flipping from positive to negative corner) by adding 180, then multiplying by -1. Since the *color* and *sign* variables already store either 1 or -1, they can be multiplied by each other, then the product can be multiplied by the sum of the *posBlueAngle* and *signMod*, which is either 180 or 0. Given this knowledge, we simplified the angle reflection formula to “*sign* * *color* * (*posBlueAngle* + *signMod*).”

In our simplified version of *angleLogic*([original on page 127](#)), we eliminated the need to factor a *colorMod* variable into our calculations. Additionally, the newer version no longer needs to be multiplied by -1. These improvements may seem insignificant, but they resulted in simpler, more efficient code that's easier to read and faster to execute. Now that we have a better understanding of how it works, we will be able to better modify it in the future if needed.

```

348 float angleLogic(float posBlueAngle)
349 {
350     //SignMod  -- Positive (1) is 0
351     //          -- Negative (-1) is 180
352
353     int signMod = 0;
354     if (sign == -1)
355     {
356         signMod = 180;
357     }
358
359     //          ( 1 or -1 ) ( original ) (180 or 0)
360     float angle = sign * color * (posBlueAngle + signMod);
361     return angle;
362 }
363

```

We also improved the usability of the function by cutting down on unnecessary parameters. While the *color* and *sign* variables are necessary for the function to run, they never change in the middle of an autonomous program. Therefore, they don't need to be provided to the function every call. Global variables are used to store this data instead, as they persist between function calls. Due to this change, *angleLogic* now only requires a single parameter. It can be called like this:

```

396 chassis.turnToHeading(angleLogic(270), shortTimeout, {}, false);

```

Goal: We want to replicate our auton for all 4 possible starting positions.

To test the simplified version of our formula, we followed the same process we used to test the original version([testing on page 128](#)). We used the same test angles to confirm that the two formulas were equivalent:

Positive blue - 333.435°

Negative blue - 206.565°

Positive red - 26.565°

Negative red - 154.435°

We then plugged the positive blue angle into the new formula to calculate the angle of each side:

	color	sign	signMod	Expression	Result
Positive Blue	1	1	0	$1 * 1 * (333.435 + 0)$	333.435
Negative Blue	1	-1	180	$1 * -1 * (333.435 + 180)$	206.565
Positive Red	-1	1	0	$-1 * 1 * (333.435 + 0)$	26.565
Negative Red	-1	-1	180	$-1 * -1 * (333.435 + 180)$	154.435

Like before, we tested the code on an actual robot on an actual field in addition to verifying it mathematically. We confirmed that the simplified formula accurately reflects angles across the field.

Competition 3

01/05/24



HEXA-FORCE

Batavia Great Lakes

Goal: To place top 10 in qualifications and make it to semis.

Qualification Matches - 8th of 37

Skills - 6th of 22

Eliminations: Tournament Finalists

Ranking Data Post-Competition:

Quals - 5-2-0

10 WP - (win points)

33 AP - (alliance points)

85 SP - (Strength of Schedule Points)

Skills Score: 38 (32 Driver, 3 Auton)

OPR - 11.88 (Offensive Power Rating)

DPR - 7.15 (Defensive Power Rating)

CCWM - 4.73 (Calculated Contributed to Winning Margin)

True Skill Ranking - 2,377 of 12,256

World Skills Ranking - 1,327 of 4,350

What is our Ranking Data?

1-1-1 → 1 Win, 1 Loss, 1 Draw

Win Points - Winning a match gives 2 points, tying a match gives 1, and losing a match gives 0. Gaining an Autonomous Win Point gives 1 point.

Autonomous Points - Winning auton gives 6 points, tying gives 3, losing gives 0

Strength of Schedule Points - The combined scores of the losing alliances in all matches. For example, if we lose a match with 10 points scored, we have 10 SP. If we then win a match with 20 points, and the losing team gets 5 points, we will have 15 SP.

OPR - How much each team contributes to the opposing alliances scores (higher is better)

DPR - How much teams score when you are against them (lower is better)

CCWM - The team's total contribution to their alliances (higher is better) calculated

Competition Analysis

Qualification Matches:

Alliance Partner	<u>8995E</u> (Match linked)	<p>Auton (Blue Negative): We accidentally overfilled pneumatics,, so our goal was held too tightly in our mogo mech, causing our rings to not go on the stake.</p> <p>Driver Control (DC): We let go of the goal because the rings are stuck in between the goal and the robot. We then accidentally let go of the goal due to a misinput. It takes us seven seconds to grab the goal again. 16 seconds in, our alliance partner clears the corner. It takes us 32 seconds to fill up a stake, a long time. It takes us another 14 seconds to put the stake in a positive corner. At 29 seconds remaining, we try to grab a ring, but our intake doesn't work. The ring kept getting pushed by the robot rather than doing into the robot. It took us nine seconds to grab that ring. With thirteen seconds left, the same thing happens again, and it takes us three seconds to grab a ring. Mistakes attributed to mechanical and human error.</p>
Opponents	355D 355V	
Result	Win	
Score	35 - 9	
Alliance Partner	<u>4979A</u> (Match linked)	<p>Auton (Blue Negative): We didn't move forward enough when trying to grab the first ring, and weren't able to pick it up. The alliance stake worked however. Grabbing the ring from the stack worked as well. That made our auton worth six points. It allowed us to win the autonomous bonus of 6 extra points.</p> <p>Driver Control (DC): Initially, everything went well. The pathing was not optimal, but we were able to get two rings positioned across the board in 10 seconds (3 on the goal), which is pretty good. Unfortunately for us, while grabbing the fourth ring, we got a ring stuck inside our intake. While moving back and forth to shake it out, we accidentally grab another ring and get it stuck inside our intake. We get it out quickly, but we spend 23 seconds on trying to get the first ring out of our intake before we prioritize and put our stack of three into the positive corner. This prioritization is a good move; we get control of the corner and the ring is stuck until 26 seconds later, so we saved time in dropping the goal off. We are able to put two rings on a mobile goal we grabbed . We miss grabbing a third one, and we stop trying to grab rings because we notice an opposing team coming towards the positive corner we are near with a full stake. We appear to move to guard the positive corner. Unfortunately, due to what appears to be a misinput, we move out of the way, allowing the team to get their full stack in the corner. 17 seconds later, we had a very clear opening to steal the red goal from the positive corner but we don't take it. With 5 seconds remaining, we put one ring on a wall stake. The mistakes here are attributed to mechanical and human error.</p>
Opponents	85142Z 3150C	
Result	Win	
Score	35-11	

Competition Analysis

Qualification Matches:

Alliance Partner	<u>2360X</u> (Match linked)	<p>Auton (Red Positive): No rings worked (see DC below) but we got the auton bonus of six points thanks to our alliance.</p> <p>Driver Control (DC): Before the match, we (unnecessarily) closed the pneumatic switch to fill our air and forgot to open them up again, leaving us without pneumatics the entire match. This left us at a huge disadvantage the entire match, although upon further analysis the match was winnable. During the match we aggressively pushed around the other robots and stalled them from getting rings while our alliance partner grabbed rings. When our partner was having trouble getting their full stack into the positive corner, we went to support them by trying to push them in. We went to defend the other positive corner with 53 seconds (a couple seconds late, but otherwise good) left because we noticed the other team trying to get the corner with a full stake. We very successfully defended the corner but our alliance partner abandoned their mobile goal and went across the field, unable to support us. This allowed one of the opposing teams to grab our full stake and put it into the negative corner. With seven seconds remaining, we were able to grab the goab and tip it, making it worth 8 points again. Considering the circumstances, the match was very well played. Loss attributed to human error and alliance miscommunication.</p>
Opponents	1755K 3695B	
Result	Loss	
Score	19-20	

Alliance Partner	<u>2360S</u> (Match linked)	<p>Auton (Red Positive): We didn't move forward enough when trying to grab the first ring, and weren't able to pick it up. The alliance stake didn't work either. We were also too far left of the third ring, so we didn't grab any rings this auton. However, our partner got one ring on a stake so we won autonomous bonus of 6 extra points.</p> <p>Driver Control (DC): The "fling" of our intake didn't seem to be working properly. We seemed to have issues picking up rings as well. We had a fully filled stake in 43 seconds. We spend 39 seconds trying to fight a team guarding the corner before our stake tips over by accident. The issue is that we didn't clear the positive corner beforehand, and weren't able to while fighting for the corner. There was an opportunity to do that around 23 seconds in but we didn't take it. Clearing the positive corner when we can is something we should do In the last 15 seconds, we try to put some rings on a stake but one of our rings is flung off our robot and we are unable to grab any other (misinput or intake issue unclear). Most mistakes attributed to mechanical issues and strategy issues.</p>
Opponents	4979D 2558A	
Result	Win	
Score	35-11	

Competition Analysis

Qualification Matches:

Alliance Partner	<u>3695E</u> (Match linked)	<p>Auton (Red Negative): We accidentally grabbed the blue ring instead of the red ring on the first stack, so we put 1 blue ring on our mobile goal, but it didn't matter too much since our top ring was still red. We almost got AWP but didn't touch the ladder.</p> <p>Driver Control (DC): Before the match, we (unnecessarily) closed the pneumatic switch to fill our air and forgot to open them up again, leaving us without pneumatics the entire match. This left us at a huge disadvantage the entire match, although upon further analysis the match was winnable. During the match we aggressively pushed around the other robots and stalled them from getting rings while our alliance partner grabbed rings. When our partner was having trouble getting their full stack into the positive corner, we went to support them by trying to push them in. We went to defend the other positive corner with 53 seconds (a couple seconds late, but otherwise good) left because we noticed the other team trying to get the corner with a full stake. We very successfully defended the corner but our alliance partner abandoned their mobile goal and went across the field, unable to support us. This allowed one of the opposing teams to grab our full stake and put it into the negative corner. With seven seconds remaining, we were able to grab the goal and tip it, making it worth 8 points again. Considering the circumstances, the match was very well played. Loss attributed to human error and alliance miscommunication.</p>
Opponents	1755K 3695B	
Result	Win	
Score	30-5	
Alliance Partner	<u>2360S</u> (Match linked)	<p>Auton (Red Positive): We didn't move forward enough when trying to grab the first ring, and weren't able to pick it up. The alliance stake didn't work either. We were also too far left of the third ring, so we didn't grab any rings this auton. However, our partner got one ring on a stake so we won autonomous bonus of 6 extra points.</p> <p>Driver Control (DC): One of the opposing alliance teams immediately tries to get control of our positive corner. We notice two seconds in, and by that time the other robot is nearly at the corner. A faster response time would be initiated with help from the drive team.</p>
Opponents	4979D 2558A	
Result	Loss	
Score	16-22	

Overall Competition Analysis:

- One thing we noticed watching the match was the under usage of the rest of the drive team. Successful teams have drive teams that constantly look around the field for what is happening with the other robots, while the driver is focused on the robot. Our drive team is (for the most part) focused only on our robot. We could have stopped some mistakes we made if the team was looking out for other robots rather than focusing on our own. Match 4 is a pretty good usage of our driver team, but it happened match 4 because our robot was locked and fighting for the positive corner.
- Auton related notes
 - We need to win auton. **Every single time we won auton, we won the match.** The six point bonus makes a huge difference, and we often lost by small margin
 - We would also really like to get AWP in our matches, as it means even when we lose, we win.
 - It was very tiring to constantly change our programming between matches to make an adaptable autonomous. We will try our best to not have it happen again
- Guard the positive corners needs to become our robot's name
- We should get more driver practice

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
						1
2	3 Basement @ 3:00 pm (Work on Skills Auton)	4 Basement @ 3:00 pm (Work on Skills Auton)	5 Basement @ 3:00 pm (Work on Skills Auton)	6 Basement @ 3:00 pm (Work on Skills Auton)	7 Basement @ 3:00 pm (Work on Skills Auton)	8 Reffing at Great Lakes Competition @ 7:30 am
9 Great Lakes Skills Competition!	10 Team Interview for U.S. Open @ 7:00 pm!	11	12	13 Engineering Notebook due for State @ 6:00 pm!	14	15 Chicago Competition!
16	17	18	19	20	21	22
23	24	25	26	27	28 Illinois State Championship!	
		Upcoming Competitions: January 5th @ Great Lakes January 19th @ Mundelein No longer attending because of spot availability! Instead, we are attending the Purdue competition on January 18th. January 25th @ Kalahari Signature Event				

Goal: Define the requirements and criteria for our skills autonomous program.

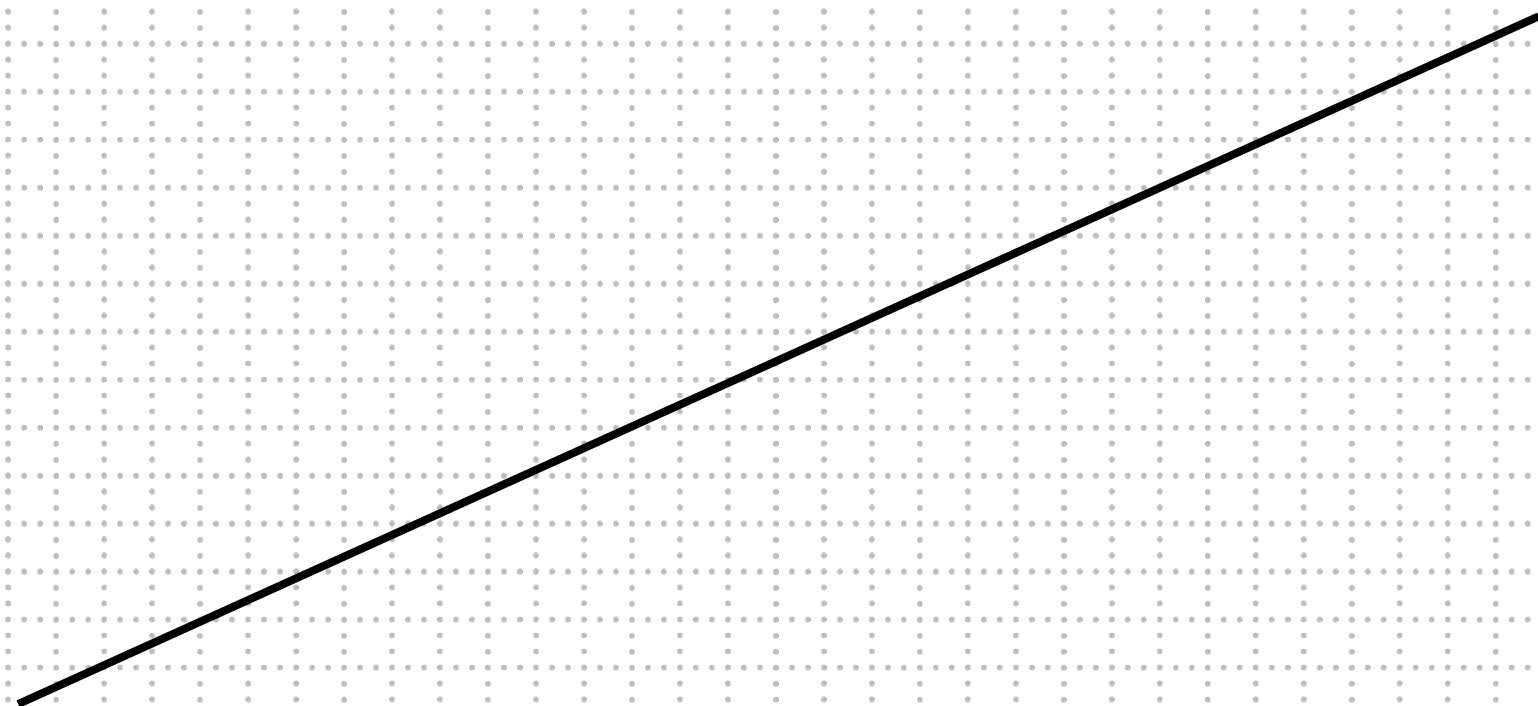
Problem Statement: We realized that skills is crucial because it is often used to determine awards and qualifications, especially when it comes to qualifying for worlds. Our robot excels offensively, so skills would match well with our team's strengths, giving us a strong likelihood of scoring among the top Illinois teams.

Solution Requirements:

- Must be run within 1 minute
- Robot must start not touching any game elements
- Robot's plane must start breaking the plane of the starting line

Solution Goals:

- Our driver skills is currently 44 points, so we hope to double this score with skills auton (Total Skills: 88 points)



Goal: Brainstorm solutions for our skills auton strategy.

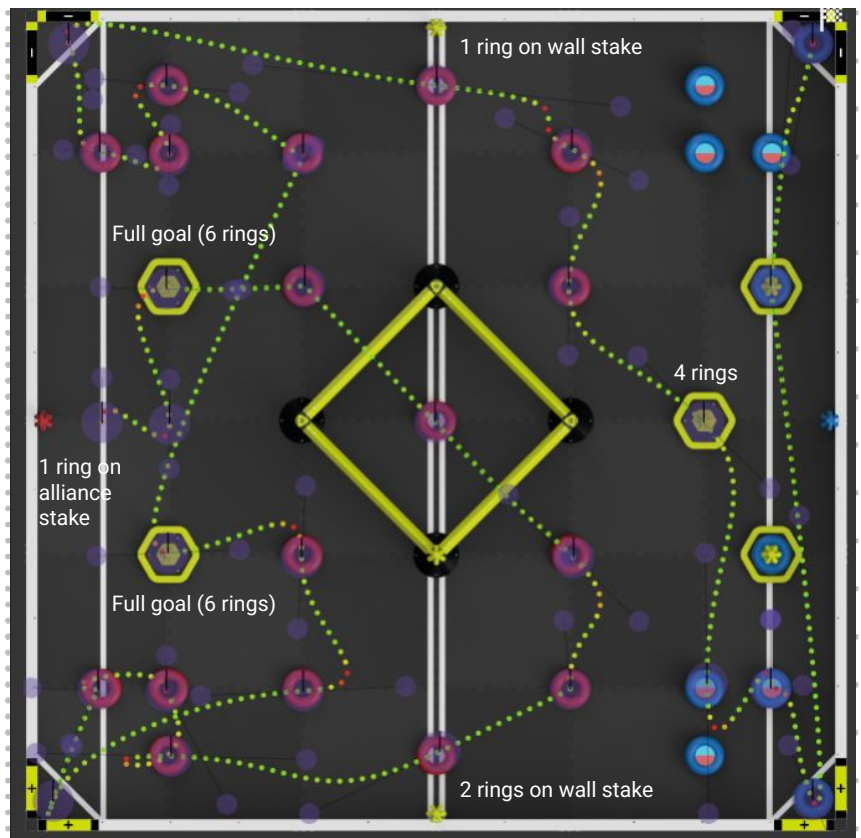
Independent Inquiry:

- Having rings on as many stakes as possible is ideal since the top ring on every stake is worth 3 points
- It is best to have minimal overlap in the path to increase efficiency
- Score all red rings in order to use blue rings
- Start robot positioned in front of alliance stake
 - Meets all starting criteria
 - Simply spin intake to score 3 points in <1 second
- Have a mogo in each corner because regardless of rings, the goal itself is 5 points when put in the positive corner (BUT NO CORNER MODIFICATIONS BASED ON RINGS IN SKILLS)

Possible Skills Auton Strategies:

- Strategy 1 (52 Points)

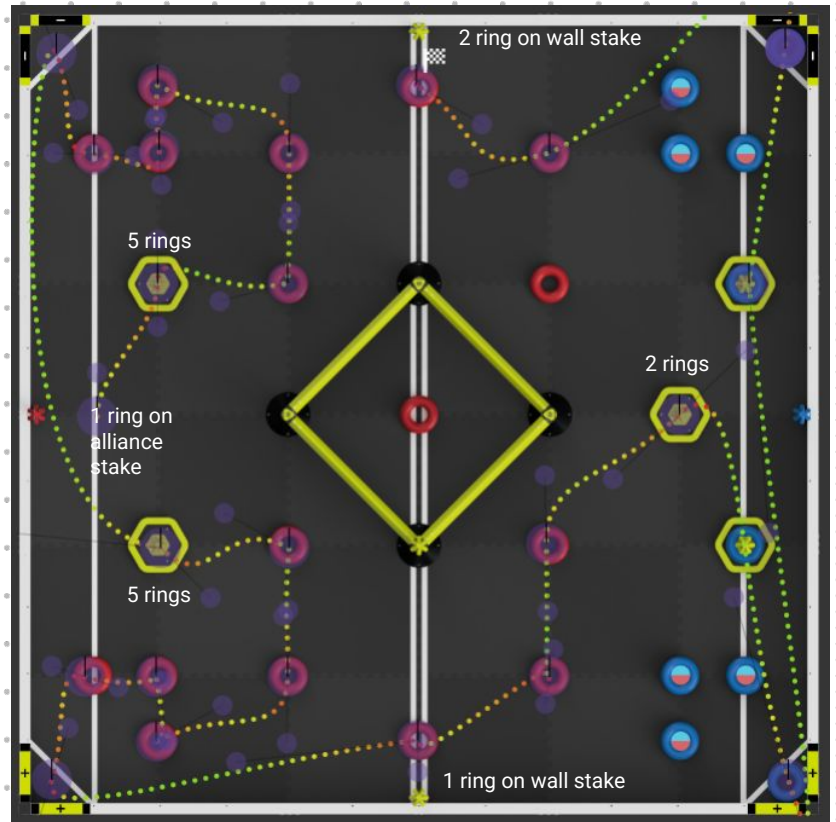
Positives	Negatives
→ All mogos in a corner	→ Blue alliance stake is not scored
→ Very minimal overlap	→ Difficult to complete <1 minute



Brainstorming Solutions

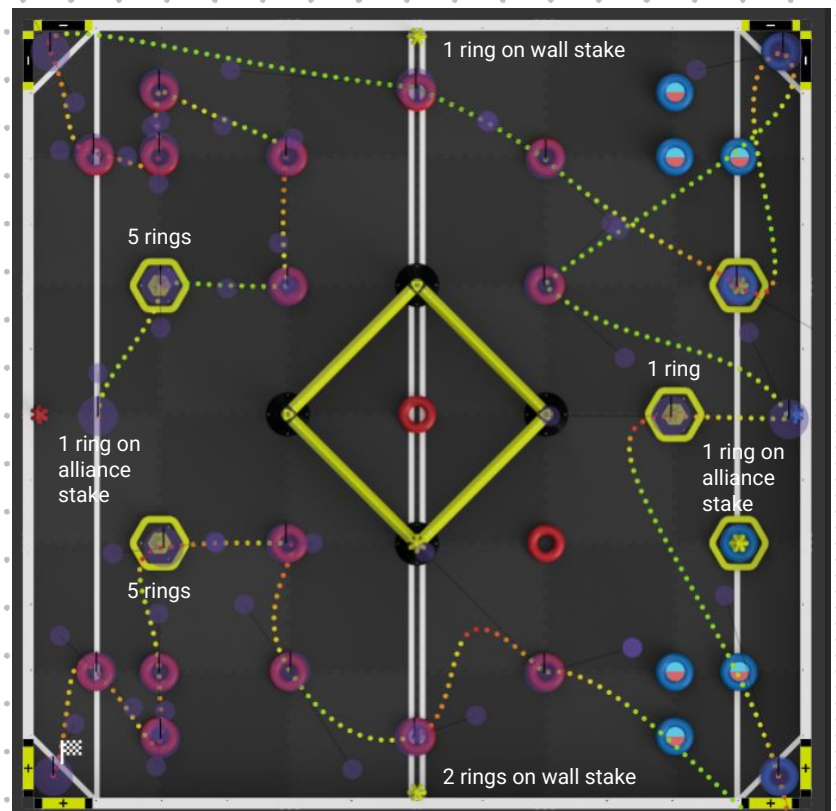
- Strategy 2 (48 Points)

Positives	Negatives
<ul style="list-style-type: none"> → Very doable in <1 minute → Very minimal overlap 	<ul style="list-style-type: none"> → Many missed rings, losing out of 8 red rings and 8 blue rings



- Strategy 3 (50 Points)

Positives	Negatives
→ Minimal overlap	→ Blue alliance
→ All stakes and goals are scored	→ stake is in tough spot
	→ Missing 5 red rings, 8 blue rings



Skills Auton

Select and Plan

Goal: Select our skills auton strategy.

Decision Matrix:

Strategy	Strategy 1	Strategy 2	Strategy 3
Time	2	3	2
Minimal Overlap	2	3	3
Points (x2)	$4 * 2 = 8$	$2 * 2 = 4$	$3 * 2 = 6$
Total	12	11	11

The Plan:

We plan on utilizing Strategy 1 which gets us 52 points. For our skills competition on February 9th, our plan is to program up until the part where we put the 2nd mobile goal in the positive corner. We will finish up the rest of the path for State.

Goal: Program skills auton.

Below is our commented skills auton code up until the 2nd mobile goal in the positive corner. Our lady brown mechanism code is a little finicky right now, so we programmed auton without wall stakes for now. We plan on fixing the lady brown mech code after the skills competition due to time constraints.

```
//ALIGNMENT: back of robot is 3.75 inches from the wall while centered with alliance stake
void realSkills()
{
    //Start at alliance stake
    chassis.setPose(x: -58.75, y: 0, theta: 90);
    intake2.move(voltage: 127);
    pros::delay(milliseconds: 300);
    printPose(message: "Starting Pose");
    chassis.moveToPoint(x: -47, y: 0, timeout: longTimeout, params: {}, async: false);
    printPose(message: "Forward from alliance stake");
    intake2.brake();

    //Turn towards mobile goal
    pointTurn(x: -47, y: 24, timeout: shortTimeout, forwards: false, maxSpeed: 70);
    printPose(message: "After turn to mogo");
    //Grab mobile goal while moving
    chassis.moveToPoint(x: -47, y: 24, timeout: longTimeout, params: {.forwards = false, .maxSpeed = 100}, async: true);
    chassis.waitForDistance(15);
    mogoMech.set_value(true);
    pros::delay(milliseconds: 200);
    chassis.waitForDone();
    printPose(message: "Grabbed mogo");

    //Turn towards 1st ring
    pointTurn(x: -24, y: 24, timeout: shortTimeout, forwards: true, maxSpeed: 127);
    printPose(message: "Turned to 1st ring");
    //Grab 1st ring
    intake1.move(voltage: 127);
    intake2.move(voltage: 127);
    chassis.moveToPoint(x: -24, y: 24, timeout: longTimeout, params: {}, async: false);
    printPose(message: "Grab 1st ring");

    //Move to 3rd ring while crossing under ladder to grab middle (2nd) ring
    //Intake speed fluctuates to make sure intake doesn't hit ladder when going through ladder
    pointTurn(x: 24, y: -24, timeout: shortTimeout, forwards: true, maxSpeed: 127);
    printPose(message: "Turn to middle of ladder");
    intake2.move(voltage: 35);
    float distance = chassis.getPose().distance(other: lemlib::Pose(x: 0, y: 0));
    printf("Distance: %f\n", distance);
    chassis.moveToPoint(x: 24, y: -24, timeout: longTimeout, params: {}, async: true);
    chassis.waitForDistance(distance - 1);
    printPose(message: "Under ladder, speeding up");
    intake2.move(voltage: 127);
    chassis.waitForDistance(distance + 15);
    printPose(message: "Slowing back down");
    intake2.move(voltage: 35);
    chassis.waitForDone();
    printPose(message: "Crossed under ladder");
}
```


Build and Implement

```
//Ring check: 3 Rings on mobile goal
intake2.move(voltage: 127);
//Move to 4th ring/1st ring on pos. wall stake
pointTurn(x: 24, y: -48, timeout: shortTimeout, forwards: true, maxSpeed: 127);
chassis.moveToPoint(x: 24, y: -48, timeout: longTimeout, params: {}, async: false);

//Pick up 5th ring/2nd ring on pos wall stake
pointTurn(x: 0, y: -61, timeout: shortTimeout, forwards: true, maxSpeed: 127);
chassis.moveToPoint(x: 0, y: -61, timeout: longTimeout, params: {}, async: false);

/*INSERT WALL STAKE*/
intake1.move(voltage: 127);
intake2.move(voltage: 127);
printPose(message: "At wall stake");

/**GRAB LAST THREE RINGS IN BOTTOM LEFT
//Pick up 6th ring/4th ring on mogo
pointTurn(x: -40, y: -64, timeout: shortTimeout, forwards: true, maxSpeed: 127);
printPose(message: "Turned to 6th ring");
chassis.moveToPoint(x: -40, y: -64, timeout: longTimeout, params: {.maxSpeed = 50}, async: false); //org -48
printPose(message: "At 6th ring");
//Pick up 7th ring/5th ring on mogo
pointTurn(x: -46, y: -54, timeout: shortTimeout, forwards: true, maxSpeed: 127);
chassis.moveToPoint(x: -46, y: -54, timeout: longTimeout, params: {}, async: false);
pros::delay(milliseconds: 1600);
intake2.brake();
//Pick up 8th ring/6th ring on mogo
pointTurn(x: -60, y: -54, timeout: shortTimeout, forwards: true, maxSpeed: 127);
chassis.moveToPoint(x: -60, y: -54, timeout: longTimeout, params: {}, async: false);

//Move mogo into corner
pointTurn(x: -64, y: -64, timeout: shortTimeout, forwards: false, maxSpeed: 127);
chassis.moveToPoint(x: -60, y: -60, timeout: shortTimeout, params: {.forwards = false}, async: false);
mogoMech.set_value(false);
intake2.move(voltage: -35);
pros::delay(milliseconds: 1000);
intake2.move(voltage: 40);

//Back out of corner and grab 2nd mobile goal
chassis.moveToPoint(x: -55, y: -55, timeout: longTimeout, params: {}, async: false);
intake2.brake();
pointTurn(x: -48, y: -24, timeout: shortTimeout, forwards: false, maxSpeed: 127);
chassis.moveToPoint(x: -48, y: -24, timeout: longTimeout, params: {.forwards = false, .maxSpeed = 80}, async: false);
chassis.waitUntil(dist: 10);
mogoMech.set_value(true);
pros::delay(milliseconds: 200);
chassis.waitUntilDone();
intake2.move(voltage: 127);

//Push 2nd mobile goal into the other corner
printPose(message: "Before turn to corner");
pointTurn(x: -65, y: 65, timeout: shortTimeout, forwards: false, maxSpeed: 127);
printPose(message: "Turned to corner");
chassis.moveToPoint(x: -65, y: 65, timeout: longTimeout, params: {.forwards = false}, async: false);
mogoMech.set_value(false);
printPose(message: "In corner");
printPose(message: "Away from corner");
```


Overall Competition Analysis

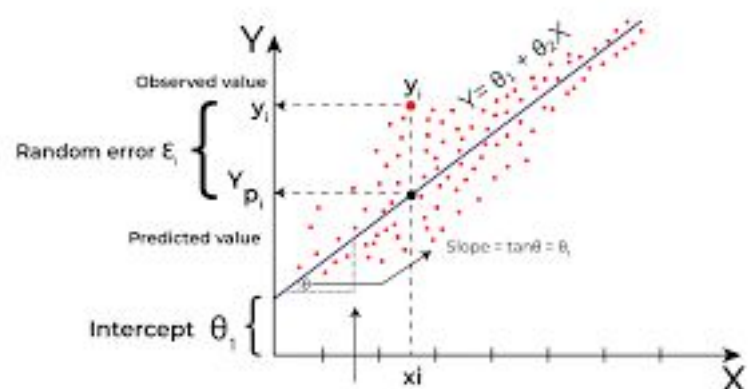
We wanted to do an overall competition analysis of our performance and the performance of other Illinois teams before we went to state, so we did a competition analysis on all of our past competitions, as well as an analysis of how top teams were performing and their statistics.

We took the data of the top 45 teams in Illinois who have competed in more than 20 matches, ranked by their true skill rating. The data can be found at this link: [Link to our data](#)

This is a screenshot of some of the data/statistics for other teams in Illinois.

tsRanking	teamNum1	teamName	event	Region	locCountry	trueSkill	totalWins	totalLosses	totalTies	totalMatch	totalWinn	qualWins	qualLosses	qualTies	qualWinn	apPerMatc	opr	dpr	ccwm	awpPerMatc
136	355Y	Loose Scre	Illinois	Illinois	United Sta	24.7	50	24	2	76	67.1	43	20	2	67.7	3.2	13.8	7.7	6.1	0.12
287	355Z	Falcons	Illinois	Illinois	United Sta	23.7	28	7	1	36	79.2	18	6	1	74	4.2	17.1	8.6	8.4	0.24
309	321A	Apollo	Illinois	Illinois	United Sta	23.6	26	4	0	30	86.7	15	4	0	78.9	4.6	16.1	5.8	10.3	0.11
309	333R	Los Robos	Illinois	Illinois	United Sta	23.6	24	5	0	29	82.8	16	3	0	84.2	4.4	19.4	4.1	15.3	0.26
355	355A	2Xstream	Illinois	Illinois	United Sta	23.4	30	7	3	40	78.8	20	4	3	79.6	3.2	14	6.1	7.9	0.15
481	321D	Deimos	Illinois	Illinois	United Sta	22.9	31	7	0	38	81.6	23	3	0	88.5	5	16.2	6.3	9.9	0.19
592	321B	BLAST OFF	Illinois	Illinois	United Sta	22.5	29	8	1	38	77.6	20	5	1	78.8	3.9	19	5.8	13.2	0.62
653	333S	Sigma	Illinois	Illinois	United Sta	22.3	27	10	0	37	73	19	7	0	73.1	3.6	14	6.7	7.2	0.27
778	333Y	Four Dolla	Illinois	Illinois	United Sta	21.9	28	10	0	38	73.7	19	7	0	73.1	4.6	15.7	8.6	7.1	0.15
778	355U	Umm...Act	Illinois	Illinois	United Sta	21.9	39	24	3	66	61.4	34	21	3	61.2	2.9	9.7	8.1	1.6	0.07
878	23880B	B Dubs	Illinois	Illinois	United Sta	21.6	12	9	0	21	57.1	11	8	0	57.9	4.1	8.7	1.8	7	0.11
1228	2360Z	ZOOM	Illinois	Illinois	United Sta	20.8	19	8	1	28	69.6	17	6	1	72.9	2.9	11.4	-0.1	11.5	0.04
1228	8995B	Blaze	Illinois	Illinois	United Sta	20.8	18	9	0	27	66.7	17	7	0	70.8	3.5	12.4	6.2	6.2	0
1342	23880C	CHROME	Illinois	Illinois	United Sta	20.6	15	8	0	23	65.2	12	7	0	63.2	4.6	9.4	4.5	4.8	0.21
1377	2360S	Singularity	Illinois	Illinois	United Sta	20.5	21	8	0	29	72.4	18	6	0	75	3.4	12.4	4.4	8.1	0.04
1519	2360A	Alpha	Illinois	Illinois	United Sta	20.2	22	6	0	28	78.6	20	4	0	83.3	4.1	17.7	2.9	14.8	0.08
1519	2360N	Nemesis	Illinois	Illinois	United Sta	20.2	22	9	0	31	71	16	8	0	66.7	3.2	14.8	7	7.8	0.12
1666	20612A	Blazin' Bot	Illinois	Illinois	United Sta	19.9	15	13	1	29	53.4	13	11	1	54	2.5	10.4	7.2	3.1	0
1666	38535A	TitanTron	Illinois	Illinois	United Sta	19.9	16	9	0	25	64	14	6	0	70	3.5	12.7	9.6	3.1	0.15
1850	725K	Kronos	Illinois	Illinois	United Sta	19.6	15	13	1	29	53.4	14	11	1	55.8	2.5	5.8	3.6	2.2	0
1967	8995E	Electric	Illinois	Illinois	United Sta	19.4	15	4	1	20	77.5	13	3	1	79.4	3.7	12.8	0.8	12	0.11
1967	38535C	TOMM-E	Illinois	Illinois	United Sta	19.4	14	12	0	26	53.8	11	10	0	52.4	2.9	9.9	8.5	1.4	0.19
1967	60172W	WHAM	Illinois	Illinois	United Sta	19.4	15	11	0	26	57.7	11	8	0	57.9	3.3	8.4	6.9	1.5	0.05
1967	99371B	Electrical	Illinois	Illinois	United Sta	19.4	19	10	0	29	65.5	12	8	0	60	1.9	11.9	8.5	3.4	0.05
2032	355P	PowerHou	Illinois	Illinois	United Sta	19.3	15	21	2	38	42.1	13	18	2	42.4	3.3	8.6	11.4	-2.8	0.03
2103	2360V	Vortex	Illinois	Illinois	United Sta	19.2	16	10	1	27	61.1	15	8	1	64.6	3.6	5.2	5.1	0.1	0.08
2562	355X	Hexa-Force	Illinois	Illinois	United Sta	18.5	15	11	0	26	57.7	12	9	0	57.1	3.3	14.8	11.6	3.2	0.05
2562	23880A	Techno Teg	Illinois	Illinois	United Sta	18.5	9	10	1	20	47.5	9	9	1	50	3.3	4.8	3.7	1.1	0.11
2813	333Z	Can't we j	Illinois	Illinois	United Sta	18.1	11	13	0	24	45.8	10	10	0	50	2.9	2.9	8.9	-6.1	0
2813	60172U	URAQT	Illinois	Illinois	United Sta	18.1	12	12	1	25	50	9	9	1	50	3.2	11.3	7.3	4	0
2886	355T	Thunderbi	Illinois	Illinois	United Sta	18	18	21	1	40	46.2	15	17	1	47	3.5	8.9	12.2	-3.3	0.06
2872	20612B	Blazin' Bot	Illinois	Illinois	United Sta	17.9	9	12	1	22	41.3	6	12	1	34.2	2.2	3.7	7.5	-3.8	0.05

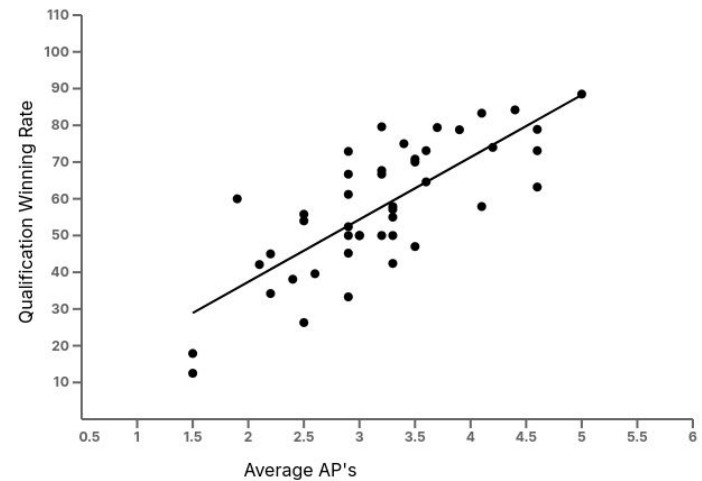
The model on the right is called a linear regression model. There are two variables we want to consider: the correlation coefficient (r) and the coefficient of determination (r^2). R tells us the strength of the relationship and r^2 tells us what amount of variance in one variable is caused by the other, ranging from 0 to 1.



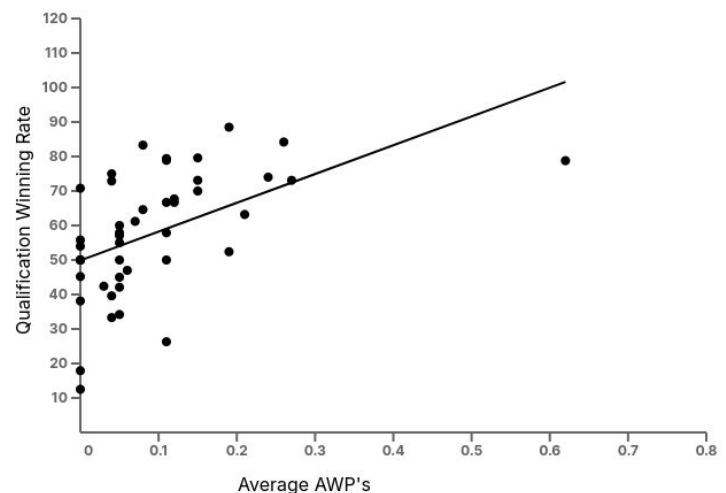
Competition Analysis

Before carrying out any test, we set a condition saying that a meaningful result had an r^2 value above 50%

The graph on the right compares teams in Illinois' average autonomous bonus points with their qualification winning rate. Our r value was 0.769 and our r^2 value was 0.5915. This is statistically significant for us, and we decided we want to prioritize getting autonomous bonus for state. One thing to consider with this data is if both teams don't have an autonomous, they both get three points, which may skew the data.



This graph compares teams in Illinois' average AWP points with their qualification winning rate. Our r value was 0.513 and our r^2 value was 0.2633. This is not statistically significant. However, we want to go forward with programming an AWP. This graph shows that just because a team wins AWP, it doesn't mean that they are likely to win a match.

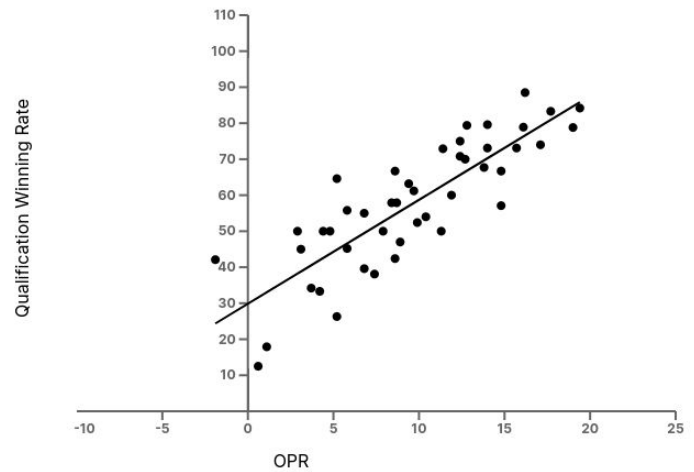


However, we want to have AWP because it means even when we lose a match, it still benefits us, so we realized this data doesn't really affect our opinion on AWP. A better comparison would be a team's Average AWP with their average spot on the leaderboard, but we cannot find a way to acquire that data.

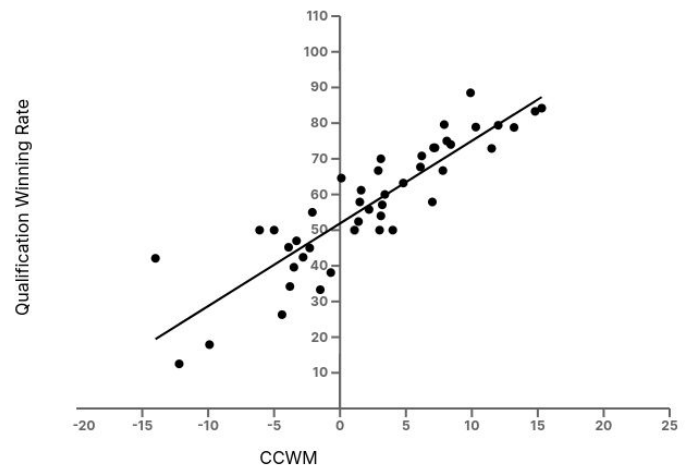
Competition Analysis

We also wanted some data to see how effective CCWM and OPR are in seeing how competitive a team is.

This graph compares teams in Illinois' OPR with their qualification winning rate. Our r value was 0.839 and our r^2 value was 0.7042. This is statistically significant for us, and shows to us CCWM is a good way of determining team competitiveness



This graph compares teams in Illinois' CCWM with their qualification winning rate. Our r value was 0.513 and our r^2 value was 0.8850. This is statistically significant, and means that CCWM is also a good way of determining a team's competitive strength at competitions



Date 02/28/25

Event Name Illinois V5RC HS State Championship

Innovate Award Submission Information Form

Instructions for team: Please fill out all information, printing clearly. This form should be included as per the instructions given in the [Guide to Judging](#). Teams may only submit **one** aspect of their design to be considered for this award at each event. Submission of multiple aspects will nullify the team's consideration for this award.

Full Team Number: 355X

Brief description of the novel aspect of the team's design:

Our team derived a formula that allows us to utilize 1 auton function for all 4 possible starting positions. This formula takes 3 inputs: color (red or blue), sign (positive corner or negative corner), and the default blue positive value. It converts the inputted default blue positive value to the value needed for the starting position that is specified by the other 2 inputs. This allows us to make easy auton adjustments to all 4 programs at once while the code remains clean and easy to read.

Identify the page numbers and/or the section(s) where documentation of the development of this aspect can be found:

Reflecting Auton Entries: 130-137

Initial Auton Entries (*Original Auton Without Formula*):

Explain why your submission is unique from other approaches to the problem it solves or task it performs:

Our approach is unique because although some teams may have similar autons for all 4 starting positions, they often copy/paste the program 4 separate times. This makes even the smallest edits incredibly time-consuming and creates unnecessary variation between all 4 programs. Our approach not only saves time programming but also ensures minimal variation in aspects like accuracy between all 4 autons.