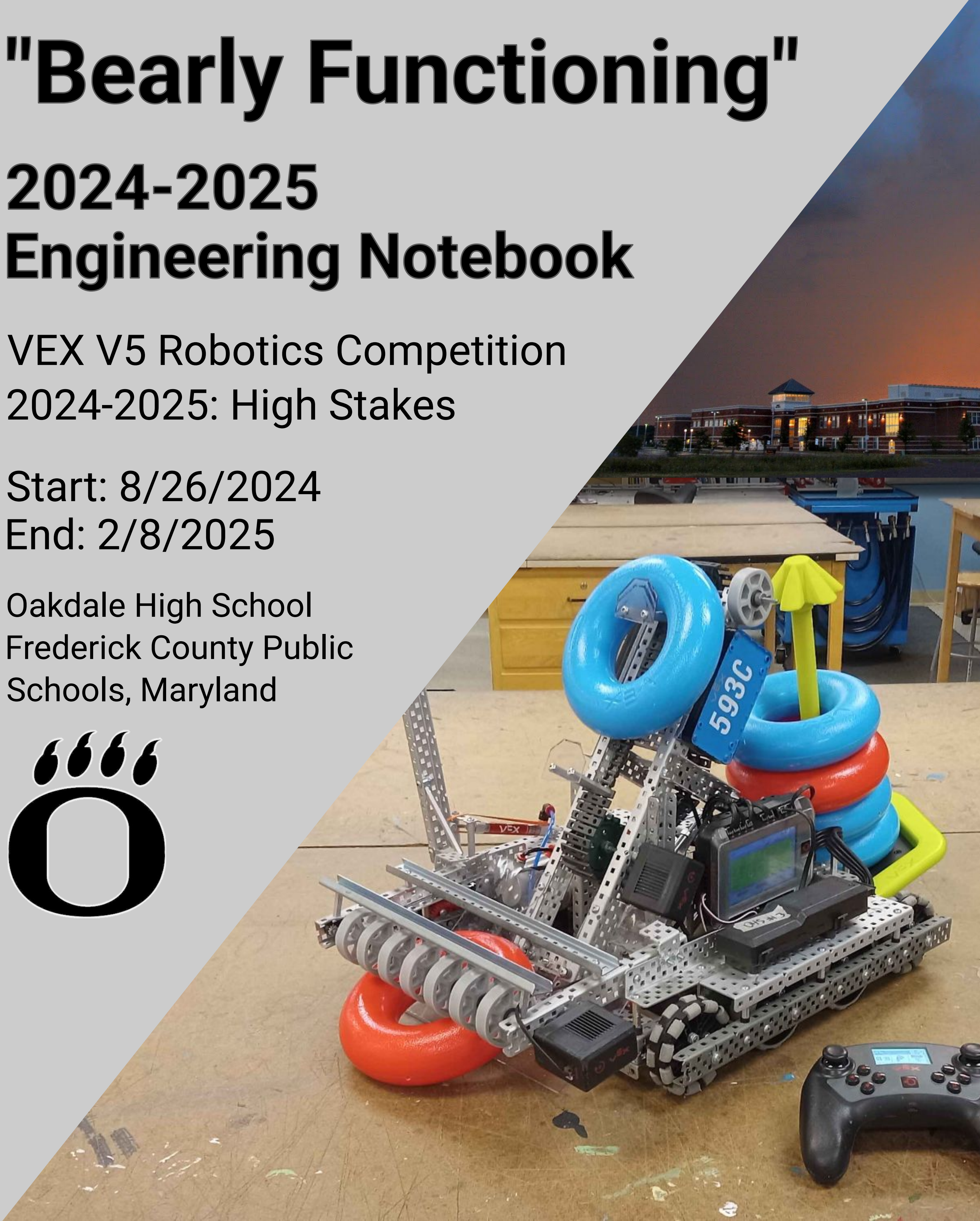
Oakdale 593C
"Bearly Functioning"

2024-2025
Engineering Notebook

VEX V5 Robotics Competition
2024-2025: High Stakes

Start: 8/26/2024
End: 2/8/2025

Oakdale High School
Frederick County Public
Schools, Maryland



Date 2/8/2025 Event Name SHHS Last Chance Tournament

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: 593C

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

Utilizing an optical sensor and some code, we designed a color sorter to sort the rings before they are scored onto stakes. Rings pass the optical sensor, if they are of an allowed color, rings continue onto their path of being scored onto a stake, otherwise they are forcefully ejected from the conveyor due to an abrupt stop by the hook carrying said ring at the top of its arc.

We have found a 90% effectivity rate with this system, where most failures are caused by a human, not by the mechanism.

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

Pages 60 - 62

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

Our system is designed to be as simple as possible. All it requires is one color sensor, and about 10 lines of code. The entire program is based on gravity and motors, instead of pneumatics which might run out of air, so it will be effective throughout the entire match, versus only being turned on for a portion of the match. We also have a killswitch, to prevent issues mid match, and to try to fill a goal to shove into the negative corner.

This notebook utilizes colors to designate its sections, utilizing the Engineering Design Process (EDP) outlined in the VEX Engineering Notebook Rubric to guide our design process.

Identify — **“Identify the Problem”** - Identifies challenges of the game/design and explains how they can be solved

Brainstorm — **“Brainstorm, Diagram, or Prototype Solutions”** - Create solutions to the problem, then prototype the best solutions

Select — **“Select Best Solution and Plan”** Select a solution, explain why and continue to plan from there

Build — **“Build and Program the Solution”** - Create the solution that was chosen, and describe the design and building process

Program — **“Build and Program the Solution”** - Program the solution that was chosen, document and explain the programming process

Test — **“Test Solution”** - Records testing of the solution, including test results and a summary

Iterate — **“Repeat Design Process”** - Records roses, thorns, and buds, what went right, wrong, and what we can improve.

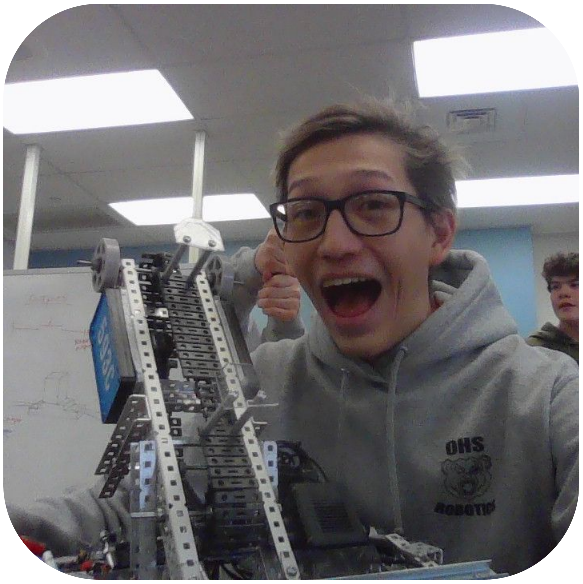
The Engineering Design Process is cyclic in nature, allowing for improvement of systems over time, through iteration. After identifying a problem, brainstorming, designing, and building a solution, we test our solution to identify what went right (Rose), what went wrong (Thorn), and how we can improve (Bud). We then repeat the Engineering Design Process, to improve our solution.

In addition to documenting the Engineering Design Process, this notebook will also document various other functions of our team, including team management, strategy, and reflections.

Notebook	**"Notebook"** - These pages include information about the formatting or setup of the notebook.
Team	**"Team Decisions"** - These pages will document whole team decisions, including budgets, meeting dates, and team members
Daily Log	**"Daily Logs"** - These pages will include a summary of the activities completed during team meetings
Management	**"Time and Team Management"** These pages will document the tasks each team member is in charge of, and the time allotted.
Strategy	**"Game Strategy"** - These pages will document our gameplay strategy during events, including routing during skills matches
Events	**"Event Plan and Analysis"** - These pages will analyze our performance at competitions, and decisions made during competitions.

Some text may be placed inside of a box, to emphasize those points.

Any included images will be labeled with the following system: Fig.#-#, where the first # is the page number and the second # is the image number.
All pages will include a footer, with the names of the team members who created that page, the date the page was created, and the page number.



Rowan Wood
Softphone
2nd Year Competing
Captain, Designer, Builder, Programmer, Driver

Chloe Vandevander
Junior
1st Year Competing
Designer, Builder, Driver

Karunya Ganga
Seinor
1st Year Competing
Designer, Builder, Driver

Ajaypranav “Ajay” Vinod
Junior
2nd Year Competing
Designer, Builder, Driver

Budgeting and Sponsors

- At our first team meeting, we developed a rough budget and started putting items onto our team expenses sheet.
- We found that this year we will need ~$6,000 to successfully buy parts and pay fees necessary for a competitive team this year.

Item	Qty.	Price per Unit	Sub-Total
Field Elements	1	$589.99	$589.99
Registration	4	$150, $100	$450
Pneumatics	4	$299	$1196
Competitions	12	$100	$1200
States	4	$150	$600
Parts			$1500
		Total:	$5,536

- All members on our team have been requested to either find a sponsor for at least $100, or pay a $100 fee to participate.
  - If all 16 members bring in $100, this will only cover $1,600. Any contributions will be welcome!
- We've applied for a few grants, each for around $5,000 to $6,000, so if we are awarded any of them it will cover almost all of our expenses!

8/26	2.75" Omni-Directional Wheel (2-Pack)	276-8106	1	$22.89	$22.89	Rowan
8/26	Rotation Sensor	276-6050	2	$41.79	$83.58	Rowan
8/26	Inertial Sensor	276-4855	1	$52.49	$52.49	Rowan
8/26	3-Wire Expander	276-5299	1	$41.79	$41.79	Rowan
8/26	Optical Sensor	276-7043	1	$47.99	$47.99	Rowan
8/26	Performance Tool Kit	276-1645	1	$45.99	$45.99	All
8/26	On-Field Robot Expansion Sizing Tool	276-5942	1	$59.99	$45.99	All
8/26	V5 Robot Brain	276-4810	1	$362.96	$362.96	All
8/26	Smart Cable Connectors (50-Pack)	276-5775	4	$3.69	$14.76	All
8/26	Smart Cable Stock (8m)	276-5774	1	$7.29	$7.29	All
8/26	Booster Kit (Structure)	276-2232	1	$199.99	$199.99	All
	Structure					

Events

We plan to attend 3 regional competitions this season, all hosted by team 9080, due to the shorter travel distance compared to all othe Baltimore competitions, and the price. Our overall goal is to qualify for states, and have fun! Any sort of award would be cool!

South Hagerstown Rebel Rumble (MS&HS) 2024

In-Person

Status: Closed
Spots Open: 0
Date: 14-Dec-2024
Event Region: Maryland
Event Code: RE-V5RC-24-7965
Type: Tournament
Location: 1101 South Potomac Street, Hagerstown, Maryland, 21740, United States

We missed this competition, not enough people signed their waiver. Whoops.

Western MD V5RC "High Stakes" Valley Mall Qualifier 2025 (MS&HS)

In-Person

Status: Open
Spots Open: 1
Date: 11-Jan-2025
Event Region: Maryland
Event Code: RE-V5RC-24-7966
Type: Tournament
Location: 17301 Valley Mall Rd, Hagerstown, Maryland, 21740, United States

Valley Mall was the event I (Rowan) enjoyed the most last year, and am looking forward to!

SHHS February High Stakes Tournament (MS&HS) 2025

In-Person

Status: Open
Spots Open: 0
Date: 8-Feb-2025
Event Region: Maryland
Event Code: RE-V5RC-24-7967
Type: Tournament
Location: 1101 South Potomac Street, Hagerstown, Maryland, 21740, United States

This was the competition we earned Skills Champions last year, and we hope to reach a similar level this year!

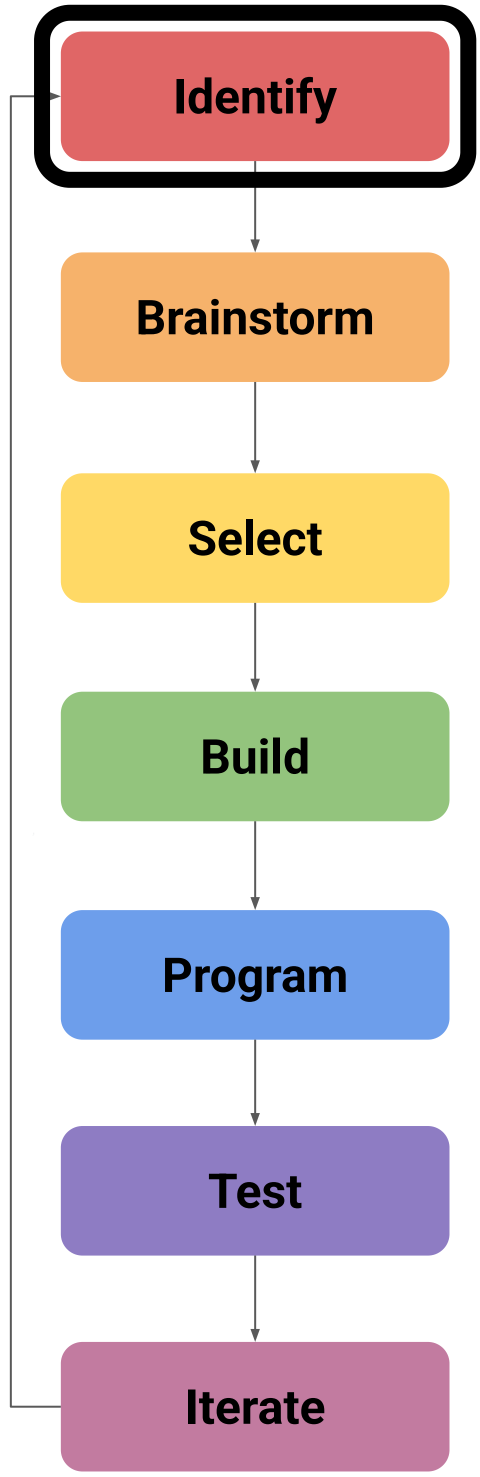
2024-2025 MD High School V5RC State Championship High Stakes

In-Person

Status: Open
Spots Open: 42
Date: 7-Mar-2025 - 8-Mar-2025
Event Region: Maryland
Event Code: RE-V5RC-24-8881
Type: Tournament
Location: 1901 Delvale Avenue, Baltimore, Maryland, 21222, United States

We may not qualify for states, but we're shooting for it!

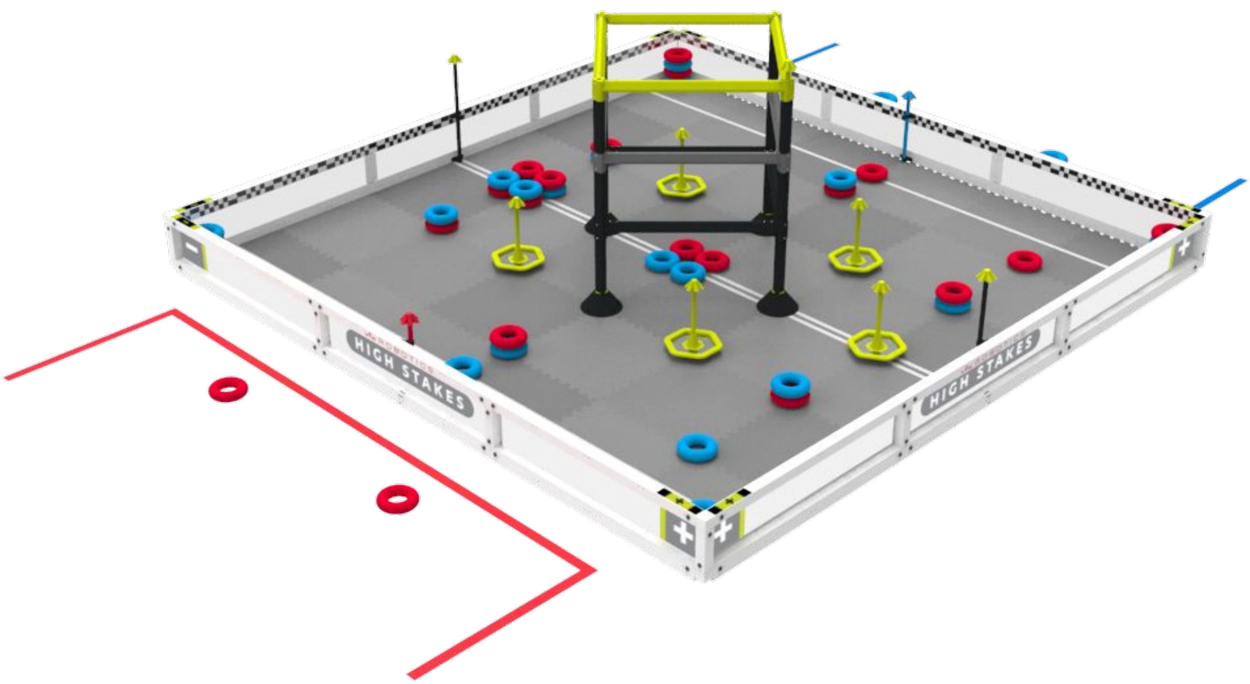
Identify the Problem
The EDP



“Identify the Problem” - Identify challenges of the game/design and explain how they can be solved

Field Setup

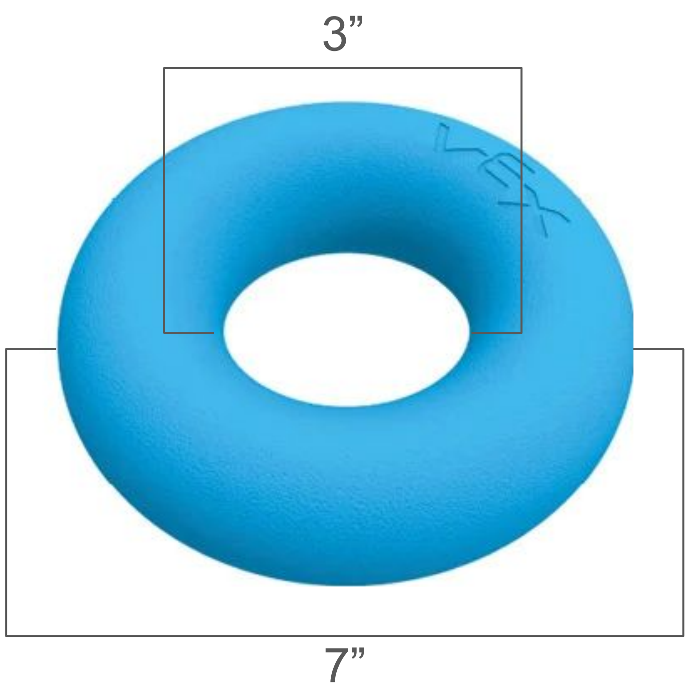
- High stakes is played on a 12’ x 12’ square field, on top of 18 24” x 24” grey foam tiles. The field is surrounded by a field perimeter, which is either 11.54” or 11.5” tall.
- 5 mobile goals are located on the field, each with one stake.
- 4 wall stakes are mounted to the field perimeter, 2 neutral and 2 alliance stakes.
- 1 ~46” tall 4 sided ladder is located in the center of the field, with tiers located 18”, 32”, and 46” from the field floor.
- 1 “High Stake” is located on top of the ladder, facing the audience.
- 48 rings are located on the field, 24 red and 24 blue. 44 rings start placed on the field during a head to head match, with 2 rings given to each team to serve as a preload.
- 2 positive and 2 negative corners are located on the field, positive corners are closest to the audience and double the score of a mobile goal, negative corners remove points.



Rings - Game Elements

Rings are torus shaped plastic objects, which are the primary scoring element in High Stakes

- Rings come in two colors, Red and Blue, similar to alliances.
- Rings have an outside diameter of 7”, inside diameter of 3”, and a thickness (or tube diameter) of 2”
- Rings are scored on stakes.
    - Not contacting a robot with the same alliance as the ring
    - Not contacting the floor/foam tiles
    - Encircling a stake
- Top rings are awarded to the highest stake on a stack where that ring is worth 3 points. All rings below that ring are worth 1 point.
- Scored at the end of a match, or the end of the autonomous period



Stakes - Field Elements

These field elements are the objects that rings are scored on. Stakes include Mobile Goals, Neutral Stakes, Alliance Stakes, and the High Stake.
Each stake is comprised of a 1” diameter PVC pipe, with a flexible cap that each ring needs to be scored on.
Caps are slightly larger than the inside diameter of rings, some force is required to score a ring.

Mobile Goals

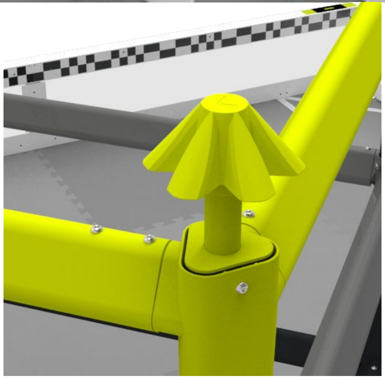
- 5 Mobile goals are located on the field
- Can contain up to 6 rings
- 10” wide, 14.5” tall
- Hexagonal base
- Weigh 2 lbs
- Robots are allowed possession of 1

Neutral Stakes

- 2 neutral stakes, located on either end of autonomous line
- Can contain up to 6 rings
- Immoble, 25” tall.

Alliance Stakes

- 2 stakes, located on either end of the field
- Can contain up to 2 rings
- Immoble, 16.72” tall.



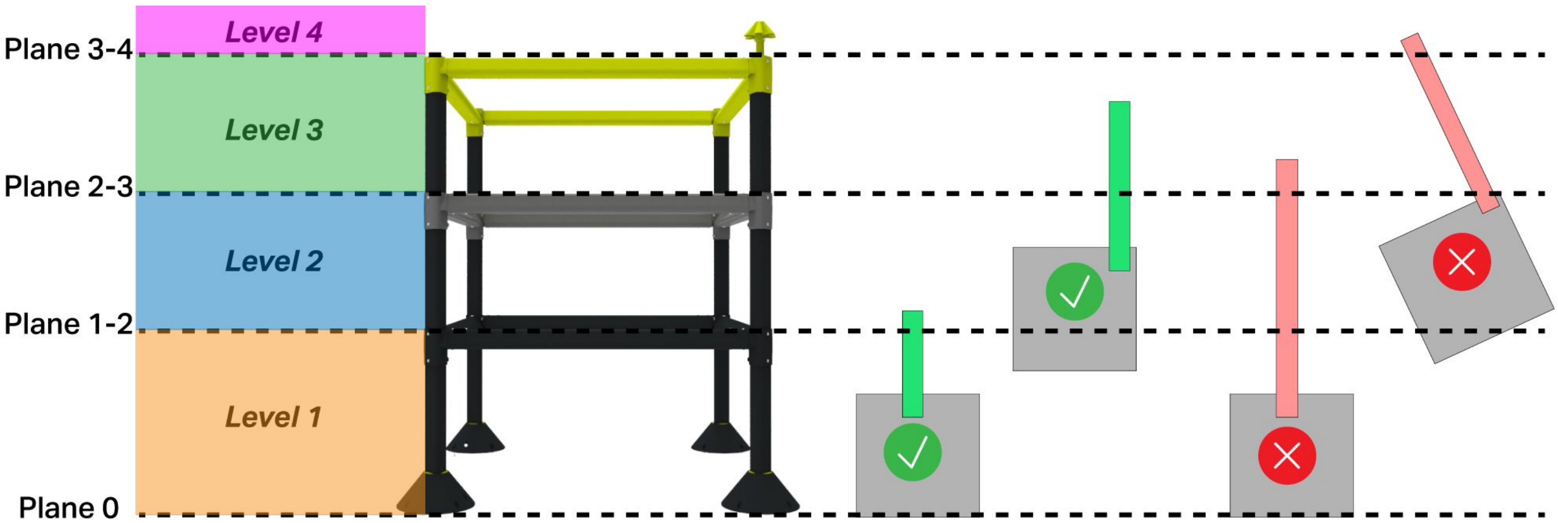
High Stake

- Located on top of the climbing tower (49.89”)
- Can contain 1 ring
- Facing audience (180°)

Ladder - Field Element

The ladder is a A 36” x 36” x 46” structure located in the center of the field.
The ladder consists of 4 vertical posts, and 3 sets of horizontal rungs spaced 18” apart.
A single “High Stake” is located at the top of the ladder, facing the audience.

At the end of the match, robots can use the ladder to elevate, to gain more points.
A robot is counted as elevated when it is no longer touching any other field elements, including the floor tiles, and is touching the ladder.
There are 4 levels, and a robot is considered elevated to a level once it passes the previous level’s highest rung.
Tier 0 begins on the floor, tier 1 is between the floor and 18 inches, tier 2 from 18-32”, tier 3 from 32”-46”
A robot may only break the plane of 2 tiers at once, which effectively gives a robot a vertical expansion limit of 32”
Bottom of the bottom rung is 16.13 in from the floor



Important Information at a Glance

Scoring

Elevation:	Rings:	Corners:
Tier 0: 0 pts	Scored Rings: 1 pt	+ Corner: Double pts
Tier 1: 3 pts	Top Ring: 3 pts	- Corner: Negative pts
Tier 2: 6 pts	High Stake: 3 pts, 2 pt	Autonomous Bonus:
Tier 3: 12 pts	bonus per elevated robot	6 pts

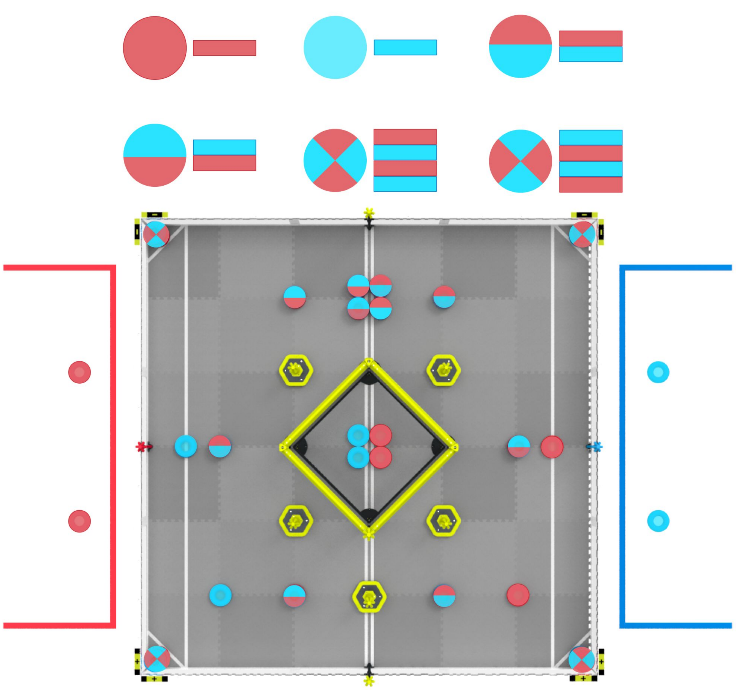
Autonomous

- 15 second autonomous period, 1:45 minute driver control period
- Auton bonus to team with highest score at end of auton period
- Auton WP: Both robots leave starting line, 3 alliance colored rings scored across 2 stakes, one robot ends touching ladder

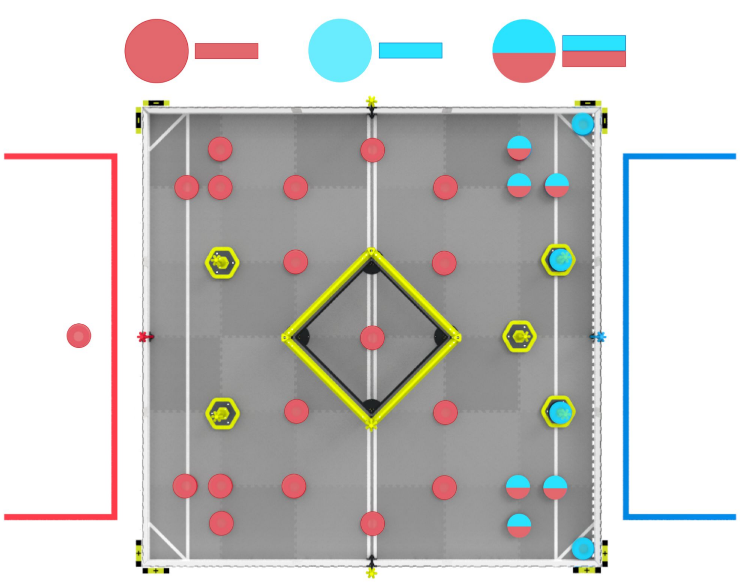
Field Setup

Robots start breaking the plane of the white alliance starting line

Head to Head Matches



Robot Skills Matches



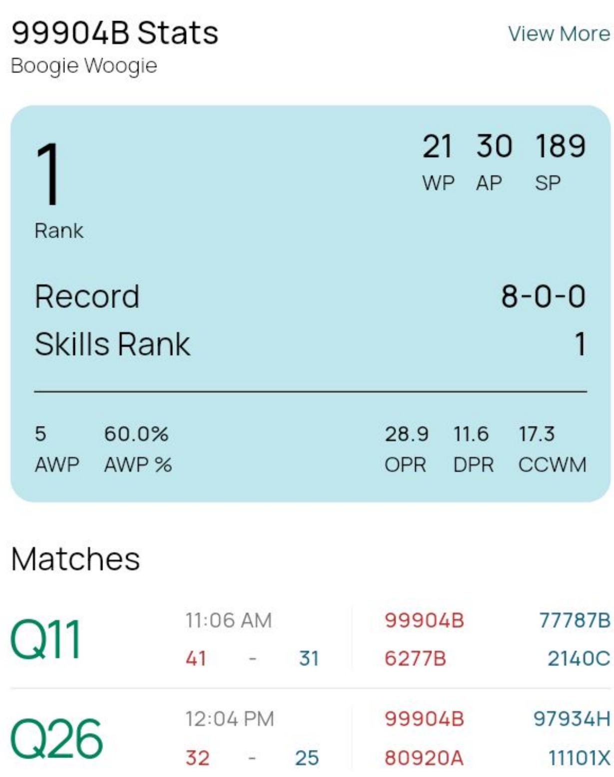
Important Terms

- **Holding** - A robot status. A robot is considered to be holding if it meets any of the following criteria during a match:
- **Trapping** - Limiting the movement of an opponent robot to a small or confined area of the field, approximately the size of one foam field tile or less, without an avenue for escape.
- **Pinning** - Preventing the movement of an opponent robot through contact with the field perimeter, a field or game element, or another robot.
- **Lifting** - Controlling an opponent's movements by raising or tilting the opponent's robot off of the foam tiles.
- **Disablement** - A penalty that forces the team affected to set down their controller for the remainder of the match
- **Disqualification** - A penalty that removes a team of all win points received in the match
- **Drive Team member** - A student who stands in the alliance station during a match
- **Field Element** – The foam field tiles, field perimeter, white tape, elevation bars, match load bars, goals, and all supporting structures or accessories
- **Autonomous Bonus** – A 8 point bonus awarded to the alliance that has earned the most points at the end of the autonomous period.
- **Robot** - A machine that follows set specifications to do game tasks, controlled by a team member or a program
- **Match** - A set time period in which the robot performs tasks.
    - Autonomous Period - First 15 seconds, robots run a pre programmed route to earn points
    - Driver Control Period - Robot is controlled by a team member to score points

Scouting Stats / Terminology

- **WP** - Win Point
    - Affects your overall ranking during qualifying matches
    - 1 WP for completing the Auton WP
    - 2 WPs for winning a match
    - Higher is better
- **AP** - Autonomous Points
    - Total points scored during the Autonomous period
    - In case of a WP tie, affects your overall ranking
    - Higher is better
- **SP** - Schedule points
    - When you beat a team, their score is added to your SP
    - In case of a WP and AP tie, SP is the tie breaker
    - Higher is better

Image credit Rowan Wood - Elapse App



- **AWP** - Auton Win Points
    - Total AWP you've scored
- **OPR** - Offensive Power Rating
    - How much you contribute to your alliance's score
    - Higher is better
- **DPR** - Defensive Power Rating
    - How much you stops your opponents from scoring
    - Lower is better
- **CCWM** - Calculated Contributed to Winning Margin
    - How much you contribute to the total winning margin
    - Higher is better

Design Brief

Problem Statement:

- High Stakes has 48 rings, 24 of each color, which can be scored on “stakes” to earn points. Each team starts with 1 ring as a preload, with the other 44 rings starting on the field. There are 4 stakes attached to the walls with varying heights, and 5 mobile stakes arranged throughout the field. In the center of the field is a “ladder” that robots can climb to score points. Each corner of the field is considered either a “positive” or “negative” corner, and can either double or eliminate the point values of the rings scored on those stakes.

Design Statement:

- We will design, build, program, test, and compete with a robot that can quickly and efficiently score game components in High Stakes

Constraints:

- Power consumption is limited to 88 watts
- Robot must be build with Vex permitted parts
- Robot must start in 18” x 18” x 18”
- Robot can expand up to 24” in one direction and, to 36” vertically
- Possession of rings are limited to 2
- Possession of stakes are limited to 1
- Custom plastic must fit within 12” x 24”
- **Time, Money, Resources**

Max motors, based on 88w limit

Example	A	B	C	D	E
Qty of 11W Motors:	8	7	6	5	0
Qty of 5.5W Motors:	0	2	4	6	16

Game Strategies

- Low level climbs don't seem to be the most game changing, passive hang would be nice. Make sure to elevate later, scores can change in an instant due to corner modifiers
- Max points is around 88 points, mostly from full stakes in the positive corners
  - Work with teammate to protect positive stakes at all costs
- Wall stakes don't seem to be the most important, but should be considered.
- Want a robot that can travel under the ladder to keep our movement options open

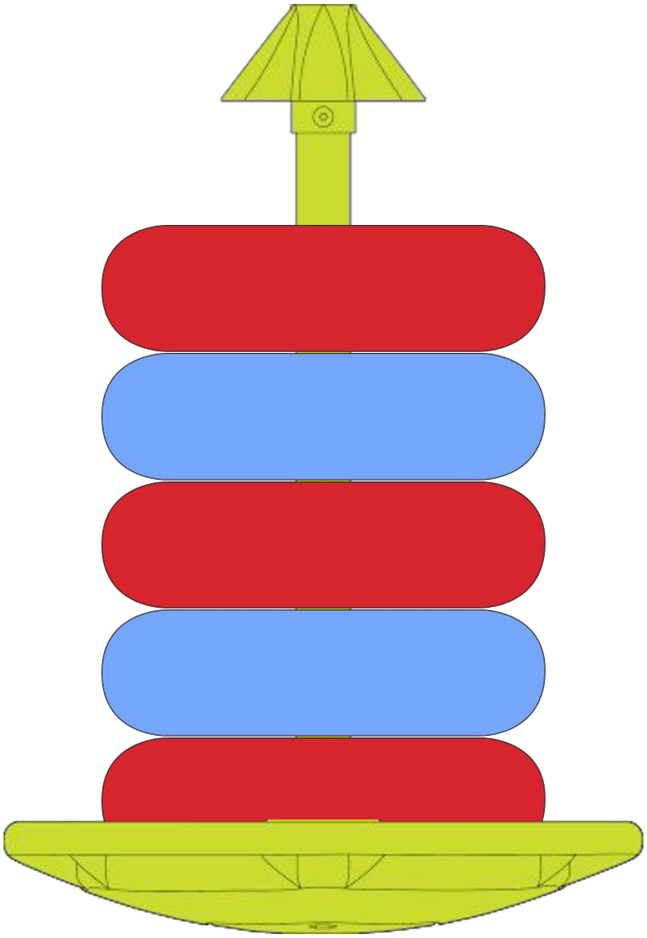
- Control over positive corners and stakes will be very important
  - Do not leave stakes unguarded, will be used against us
  - One stake in the negative corner negates any benefit from the positive corner
- Game seems that it includes much more interaction between competitors than previous games, similar to over under
- Practicing driving and using your robot is a must, this game seems to have a lot of High Stakes, one mess up could cost the match

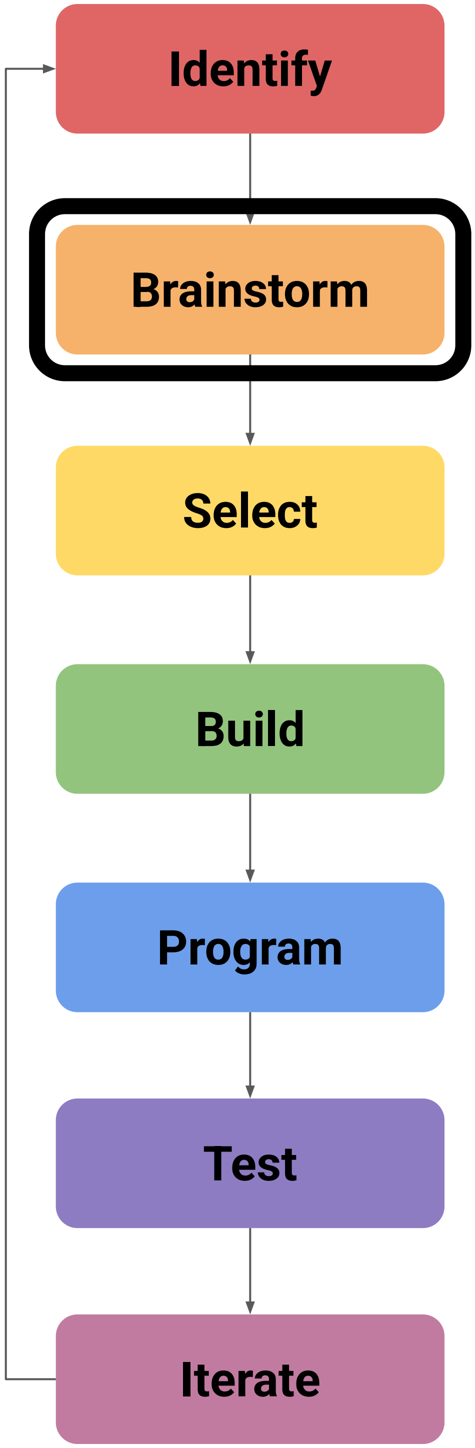
- The Autonomous Bonus is a massive help in matches, attempting to score the most points during the Auton Period is a must
  - Many teams at the Mall of America Signature Event only won due to the points they won during the Auton Period

- Network with other teams and talk with them! Don't hide in the pits, have conversations! Try to get into the eliminations!

Brainstorm Solutions
The EDP

“Brainstorm, Diagram, or Prototype Solutions” - Create solutions to the problem, then prototype the best solutions





Drivetrain Brainstorming

The drivetrain is the most important component of our robot, without one it is very hard to do anything while competing. Our goal is to pick the most effective drivetrain style to have a competitive robot.

A few things to keep in mind

- In years past, the most competitive team had fast, 6 motor (66 watt) drivetrains.
- Last year, we used a 55w drivetrain (4 11w motors, 2 5.5 w motors) which overheated very quickly due to the weight and uneven torque on the motors
- Drivetrain should be simple to maintain, so we can quickly fix any issues that arise between matches
- Hot swappable motors would be nice, but not necessary

Hot Swappable Motors

According to rule <R15c.ii>, “Teams may make the following modifications to the V5 / EXP Smart Motor’s … Removing or replacing the screws from the V5 Smart Motor Cap”

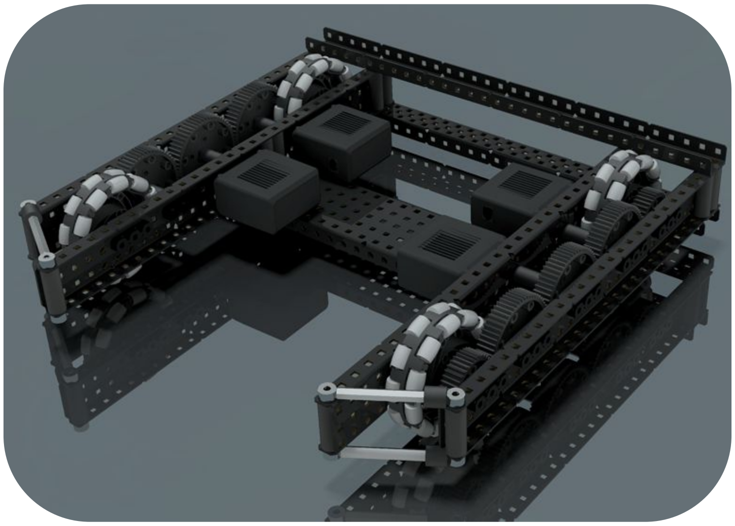
Essentially, a hot swappable motor does not have any of the linking the v5 motor cap to the motor itself, instead using zip ties and rubber bands to hold the motor in place. This allows for rapid swapping of motors between matches, so you always have cool motors going into matches; no need to worry about thermal throttling.

Image credit Ascend Robotics

Drivetrains

This video by “Engineering Girl” does a great job comparing common drivetrains, much of this section utilizes their data.
https://youtu.be/Py14YTHCth0

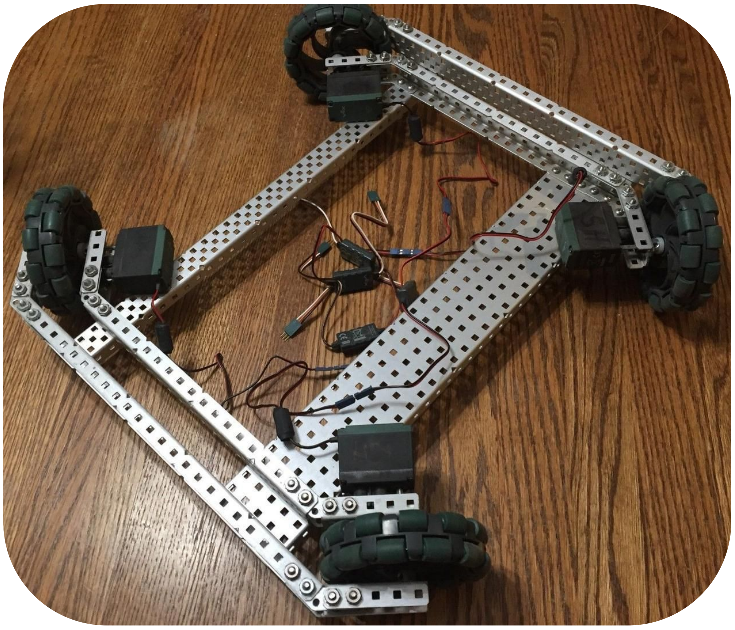
Image credit Xenon27



Tank Drive All wheels are parallel to the robot, most common drivetrain design	
Pros:	Cons:
- Fast - Consistent - Easy to build/code - Easy to maintain - Hard to shove around	- Less maneuverable - Can not strafe

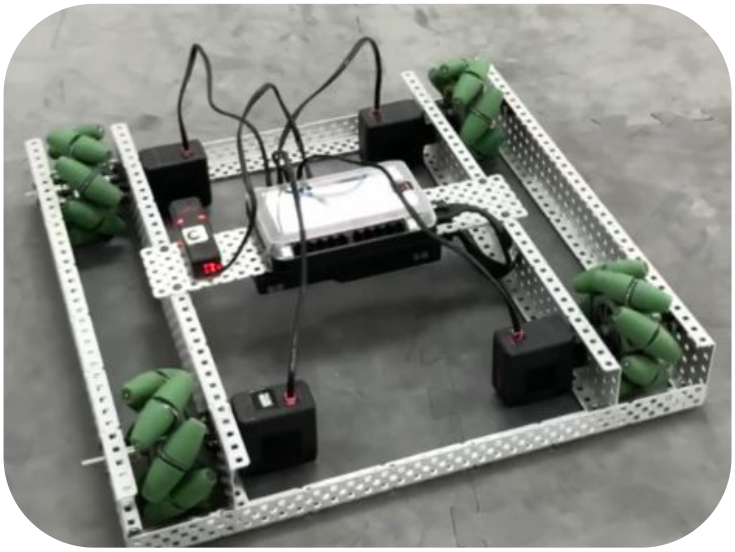
Honominic / X-Drive Wheels set at a 45 degree to the robot	
Pros:	Cons:
- Very maneuverable - Somewhat fast	- Difficult autonomous programming - Not very powerful - Hard to maintain - Easy to shove about - Not very precise in movement

Image credit Worth Point Robotics



Drivetrains

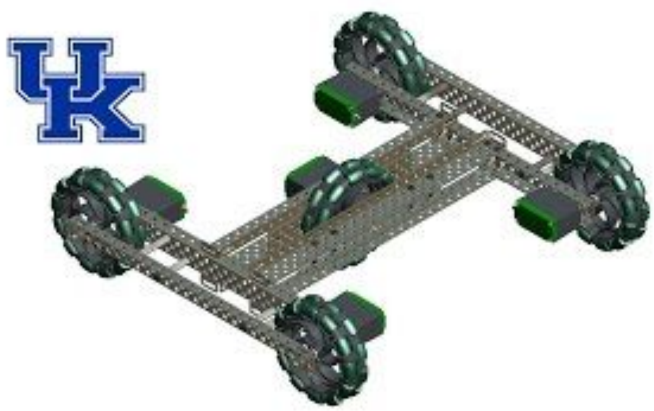
Image credit SuhJae



Mecanum Drive Uses mecanum wheels to allow for omnidirectional movement	
Pros:	Cons:
- Movement in all directions - Easy to build/code - Hard to shove - Much simpler than a Holonomic drive	- Low traction - Low power - Low speed

H-Drive Tank drive with sideways center wheel for sideways movement	
Pros:	Cons:
- Allows for sideways movement - Benefits of tank drive	- Uses an extra motor - Center wheel takes up valuable space - Easy to shove

Image credit Doug Klein



Asterisk Drive X Drive with more powah	
Pros:	Cons:
- More power than X Drive - Very versatile - Very roomy	- Easy to shove - Annoying to program - Not precise

Motor Cartridges

11 watt v5 Smart Motors utilize motor cartridges to control their output speed, which outputs different amounts of torque and speed. This can then be linked to external gear ratios to control the speed of your mechanisms.

5.5w motors are limited to an output of 200rpm, or a green insert.

	RPM	Torque under 60% velocity	Torque at 100% velocity
Red cartridge	100	~2.1 Nm	~0.7 Nm
Green cartridge	200	~1.1 Nm	~0.3 Nm
Blue cartridge	600	~0.4 Nm	~0.1 Nm

Image and Data Source: VEX Library
kb.vex.com/hc/en-us/articles/360044325872-Understanding-V5-Smart-Motor-11W-Performance

Essentially:
- Red: low speed, high torque
- Green: medium speed, medium torque
- Blue: high speed, low torque

This data is important to reference throughout the season so we can choose the best cartridge for each mechanism we're planning to use.

Ex: an intake needs to spin fast and is not under much load, a blue insert may be the best choice
Ex: an arm that supports the weight of the entire robot needs a lot of torque, but may not need speed : a red insert will work the best

Mobile Goal (MoGo) Clamp

An important aspect of this game is the control of mobile goals around the field. A clamp is important to hold onto these goals to keep them in a place where we can score onto a goal and to prevent the goal from being stolen.

Tilting the goal also makes it easier to score on and harder to steal

Pneumatics vs Motorized Clamps

Pneumatics		Motorized	
- Does not contribute to motor cap - Can tilt goal - Reliable (only 2 states)	- Relies on air in tank - More bulky - Not many teammates know how to use	- Always works - Can tilt goal - Easier to build - Less bulky	- Contributes to motor cap - Slower - Requires more moving parts - Unreliable (many states)

Image credit ADH_Team

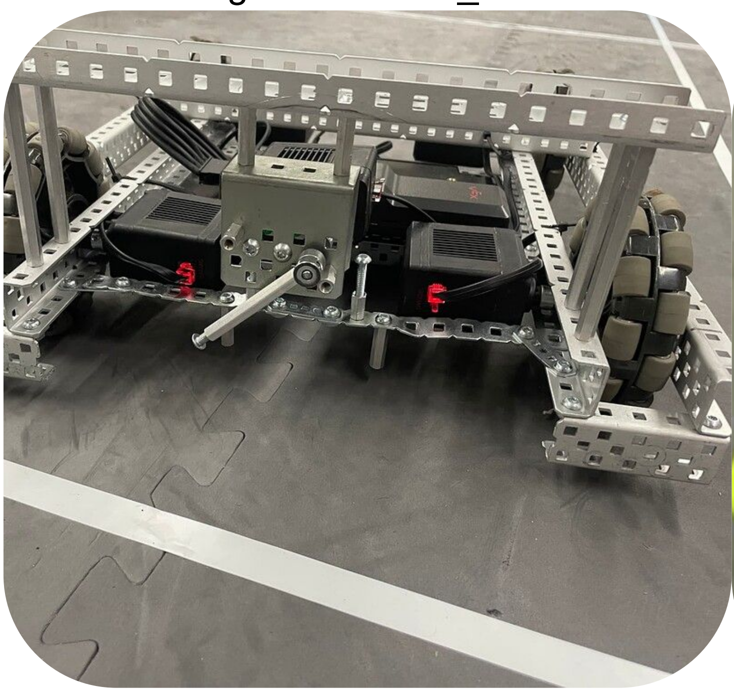
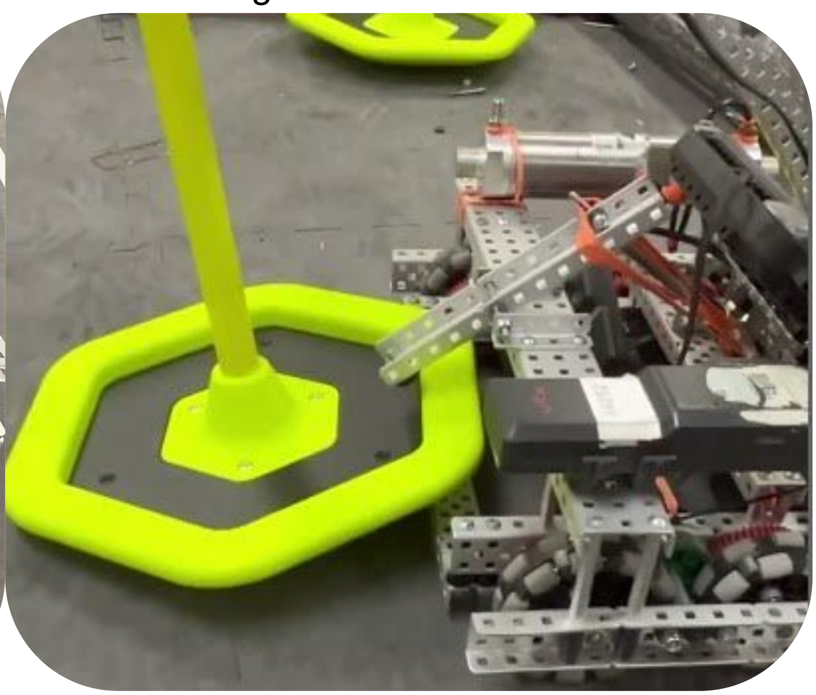


Image credit LTC Robotics



Scoring on Mobile Goals (MoGos)

The most important part of this challenge is actually scoring on mobile goals, being able to move these goals won't do much if we can't place rings onto stakes.

There are 3 main designs that seem viable, called a "plunger intake," "hood intake," and "hook intake" by much of the VEX Community.

Image credit VEX Robotics

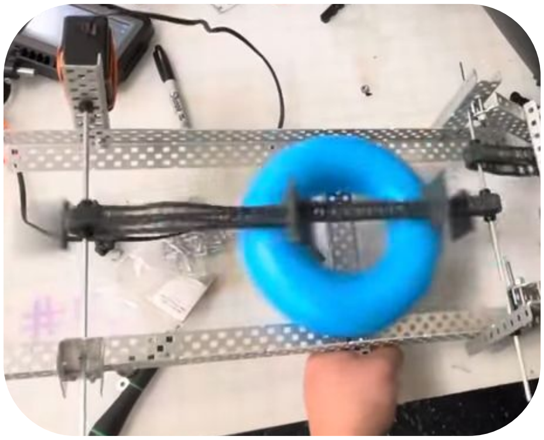
Plunger Intakes

A plunger intake, similar to the one used in the Hero Bot build by VEX this year, picks up rings through its center, much like a plunger, and forces them onto a stake.

- Very easy to build/plans already exist	- Not very fast - Uses a lot of valuable space

Hood Intakes

A "Hood Intake" gets its name from its shape, a "hood" that goes over the rings to pull them forwards.



- Somewhat easy to build - Very fast - Simple to maintain - Easy to modify to score onto stakes	- Uses a lot of valuable space - Hard to score on stakes - Lots of moving parts - Requires jutting out piece to push rings onto stake

Image credit LTC Robotics

Scoring on Mobile Goals (MoGos)

Hook Intakes

A “hook intake” gets its name from the “hook” used to move rings around. It could comprise of one or two stages, either using the hooks to directly pick rings up off the ground, or using a separate intake to bring rings up to the first stage.

- Easy to build - Compact (only need to cover center of ring) - Easy to score rings	- Hard to tune - Hard to pick up rings - Chains may break during matches - Uses custom plastic

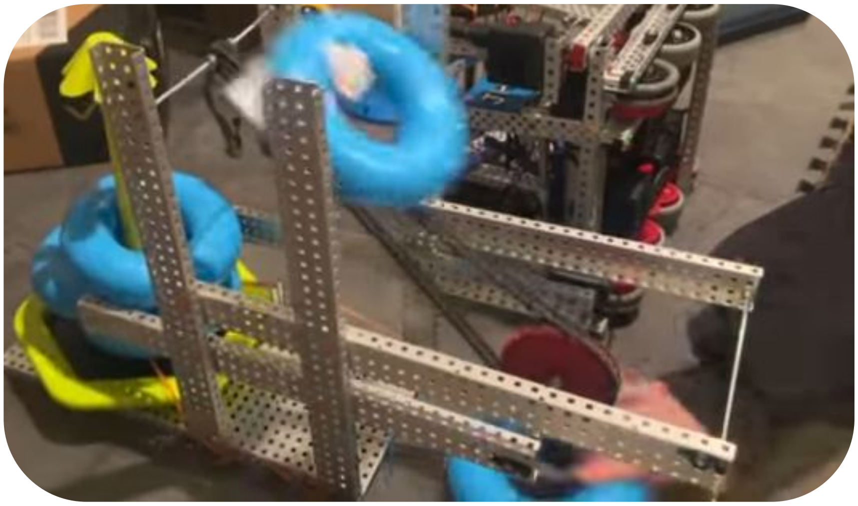


Image credit 9mg JHAWK

There are many more designs, but these were the ones we thought of and were being used by the community at this time. Many of these decisions are being made based on the Mall of America Signature Event earlier this season, which provided a great insight into what works and doesn’t work this season. Thank you to the entirety of the Maryland VEX Discord Server for your suggestions and help throughout the designing and building process, I thoroughly enjoy the very welcoming community built up there.

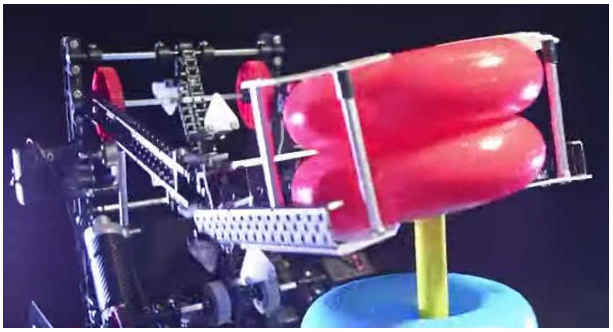
Scoring on Wall Stakes

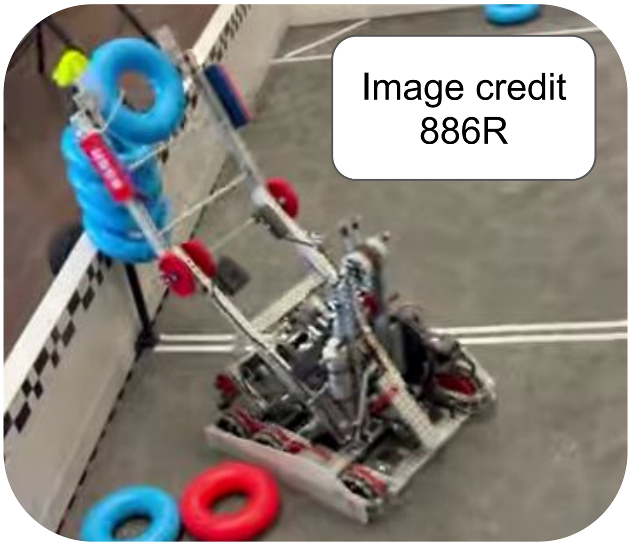
Scoring on wall stakes creates a way to ensure we have some points at the end of the game. While can not affect positive or negative modifiers, they still can create an edge over the opponent in scoring.

There are several designs the community has developed, including the “Ring Redirect mech” which pulls rings out of either a hook or hood intake and onto an arm, which lifts them up to score onto the wall stake. In addition, some team have rigged their entire intake system to hinge or lift up to score on stakes. Finally, several teams have started using a “Lady Brown mech,” or an arm that intercepts rings on their way up the conveyor, and flips them onto a wall stake.

Ring Redirect	
- Integrated with mobile goal scoring - Fast - Simple in theory	- Bulky - Mechanically complex

Image credit 1010W





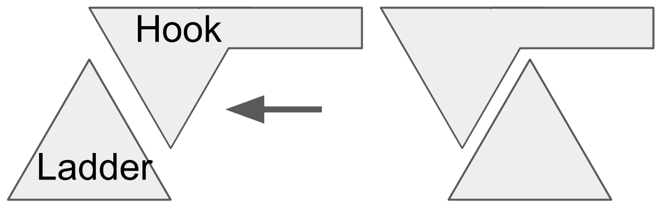
Intake lift	
- Entire scoring system lifts -	- Entire scoring system needs to me light - Motor cap

Lady Brown	
- Very simple - Easy to program -	- Uses extra metal - Contributes to motor cap

Climbing the Ladder

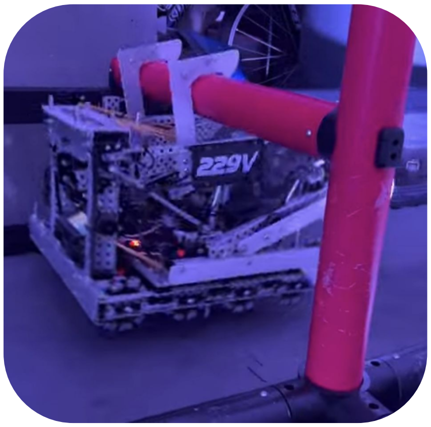
At the end of the game, you can score extra points by climbing the ladder: 12 pts for reaching the 3rd tier, 6 for tier 2, and 3 for tier 1.

Climbing to the third tier does not seem viable for our skill level, but there are a few ideas for climbing



Passive Hang (Tier 1)
Probably the easiest climb, a passive hang is just an arm that pops up just high enough that the momentum of your robot forces it to hang from the lowest bar of the ladder

Image credit ACE 229V



- Easy to build - Very fast hang - Can pop up when needed to be used	- Must be located above the center of mass of the bot - Requires a wind up to use

Motorized Hang (Tier 1)
Utilize the wall stake/scoring mechanism to hang on the lowest tier

- Easy to build - Very fast hang - No need to worry about air pressure - Can be integrated with pre existing scoring mechanisms	- Must be located above the center of mass of the bot - Contributes towards motor cap

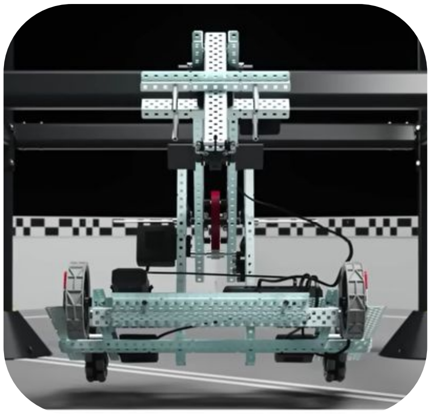


Image credit VEX Robotics

Climbing the Ladder

At the end of the game, you can score extra points by climbing the ladder: 12 pts for reaching the 3rd tier, 6 for tier 2, and 3 for tier 1.

Climbing to the third tier does not seem viable for our skill level, but there are a few ideas for climbing

Linear Slide Climb (Tier 3)
Utilizing linear slide rails, a robot can climb straight upwards, and reach a tier 3 climb in under 15 seconds, or just after positive corner protection starts.

- Tier 3 climb - 12 extra points - Likely earn awards - Simple in concept	- Not buildable at our skill level - Custom designed bot focused on climbing - Has to be in line with center of mass

Image credit 5203G Gremlin

Programming software

The VEX v5 system offers many programming libraries and CLIs (command line interface) to aid in programming our robot. VEX offers VEXcode v5 (https://codev5.vex.com/) as a starting point, which offers compatibility with block coding, similar to scratch, along with the C++ and Python programming languages. In addition, they offer the VEXcode VSCode extension, which allows for programming the robot with C++. Finally, students at Purdue University's Purdue ACM SIGBots created PROS (PROS Robotics Operating System or PROS), an entirely open source way to compile C++ software to a v5 brain.

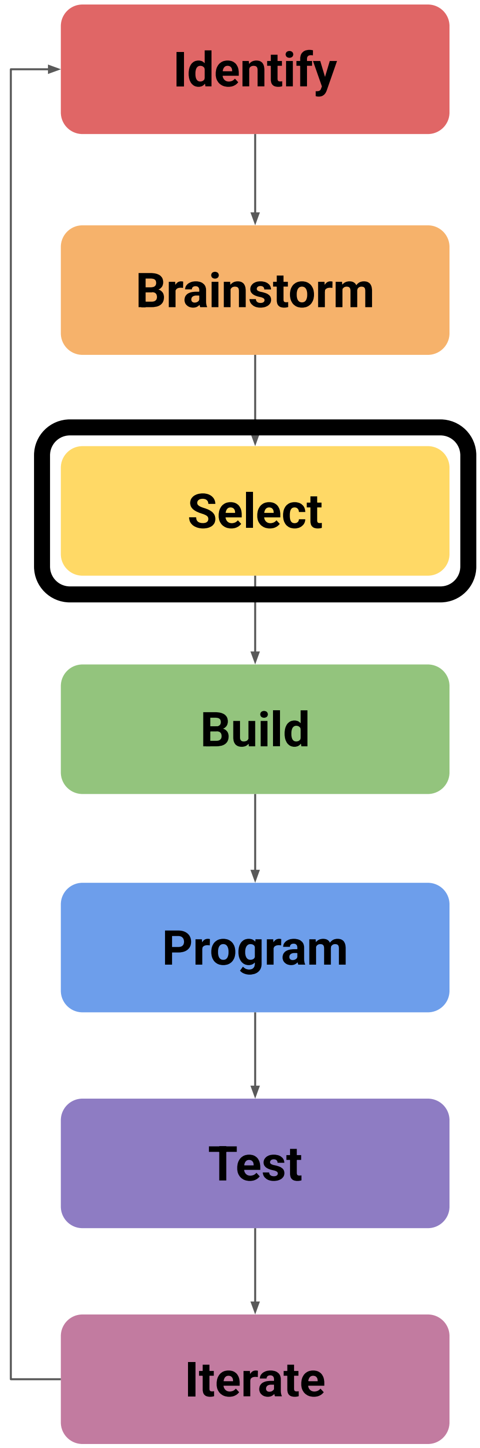
VEXcode	
- Very intuitive - Plenty of resources - Block, C++, Python - Works on any computer - Supported directly by VEX	- Limited use of libraries - Proprietary -

Image credit VEX Robotics

PROS	
- Plenty of community built, external libraries, including LemLib - Plenty of community resources - Full access to all of C++'s features	- Only supports C++ - Requires a personal device to run

Image credit Purdue ACM SIGBots

Select Solution
The EDP



"Select Best Solution and Plan" Select a solution, explain why and continue to plan from there

"Welcome to the Jungle"
12/17

Selecting our Drivetrain

We put drivetrain selection up to a vote, where each member of the team was to rank each aspect of each drivetrain on a scale of 1-5. Total points calculated at the end, highest score wins.
Only 3 members were present when votes were tallied.
Choices were made based on research in Brainstorming phase
Votes are biased, based on past experience.
Highest possible score is 75 points.

Style	Speed	Manuvablity	Pushing Force	Ease of building	Footprint	Totals
Tank	5, 5, 5	3, 4, 4	5, 5, 5	5, 5, 5	5, 5, 5	71
Honominic	4, 4, 4	5, 5, 5	4, 3, 4	3, 4, 4	4, 4, 4	61
H-Drive	4, 4, 5	5, 4, 4	4, 4, 5	4, 3, 3	3, 3, 3	58
Mecanum	4, 4, 3	4, 4, 5	3, 3, 4	4, 4, 5	4, 4, 3	58
Asterisk	4, 4, 5	5, 5, 5	4, 5, 4	3, 4, 4	4, 4, 4	64

Our overall decision is to build a tank drive due to ease of building, speed, pushing force, and its footprint. Even though it may not be as maneuverable due to being unable to strafe, we believe it will be the most viable to build and integrate with the rest of our design.
We were surprised at how similar our thoughts were, but after analysing our biases, it quickly became clear why - one style which we see everyday jumped out at us, and was simple enough to easily build and understand how it works.
Rowan was also really excited to be able to build a drifting robot again this year :)

Drivetrain - Finer Details

Now that we've narrowed down our design, there are several concepts we need to narrow down, being motors in the drivetrain (4 motors or 6 motors), and size of wheels (2.75, 3.25, or 4in).

Drivetrain Motors: 4 motors vs 6 motors

4 motor drive	
• Uses less space • Uses 2 fewer motors	• Less powerful • Slower acceleration

6 motor drive	
• More powerful • Fast acceleration	• Uses more space • Uses more motors

Wheels: 2.75 in vs 3.25 in vs 4 in

2.75in wheels	
• Uses less space • Low center of gravity • Can fit 8+ wheels (more traxion)	• Don't own any • Robot is much lower to ground

3.25in wheels	
• Balanced size • Can fit 8 wheels (more traxion)	• Only own one set of 8 • Longer than 2.75 in wheels

4in wheels	
• Fast • We have many	• Larger • Only fit 6 wheels • Higher center of gravity

Overall, we decided on building a 6 motor drive for a stronger wheelbase when entering shove matches throughout our competitions, which is an issue we highly anticipate. In addition, we chose to use 8 3.25 wheels due to their happy medium; less likely to tip, and small enough to fit 4 wheels on either half for more traxion. We will use 4 omni and 4 traction wheels, to gain the turning benefit from omni wheels on the outskirts of our drivetrain, and traction wheels in the center to prevent being shoved from the side.

Drivetrain - Finer Details

We also have to choose gearing our drivetrain. Last year, we ran direct drive green motors to 4in wheels, which limited us to 200 rpm, or ~42 in/sec. We noticed many other competitive teams had much faster robots, to respond to stimuli faster than we could.

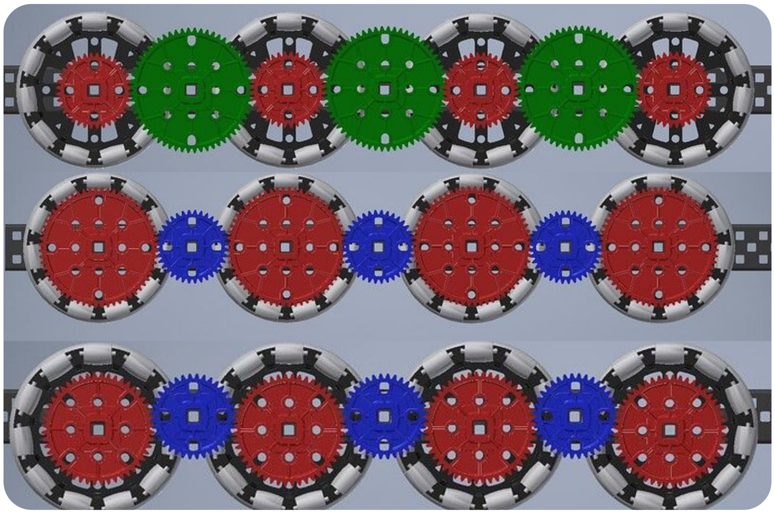
Important Formulas:

$$\frac{\text{\# of Driven Gear Teeth (Output)}}{\text{\# of Driving Gear Teeth (Input)}} = \text{Gear Ratio (Input:Output)}$$

$$\frac{\text{motor rpm * wheel diameter in * pi}}{\text{gear ratio * 60 sec/min}} = \text{Wheel Speed (in/sec)}$$

Based on previous brainstorming, we chose to use green cartridges for our drivetrain motors, to ensure we have enough torque without sacrificing speed.

We considered 3 drivetrain gearings, a 333 rpm 56.7 in/sec drivetrain, a 360 rpm 61.3 in/sec, and a 450 rpm 76.6 in/sec. While all are much faster than before, option 1 stood out to us the most, as the simplest to build, and meeting all our criteria.



Images credit Xenon27

We finally decided to screw joint our wheels, to eliminate friction between square axles and round holes, allowing for less friction on the motors, and prolonged battery / motor life.

Mobile Goal Clamp

Given our two options, the choice was very simple.
Again, each present member of the team ranked each idea on a scale of 1 to 5, highest score wins

Style	Speed	Motor Cap	Ease of building	Footprint	Totals
Pneumatic	5, 5, 5	5, 5, 5	3, 5, 4	3, 4, 4	53
Motorized	4, 4, 4	1, 1, 1	3, 4, 4	4, 4, 5	39

Image credit ADH_Team

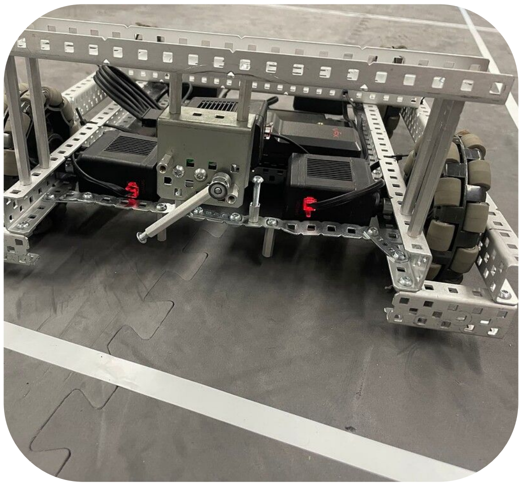
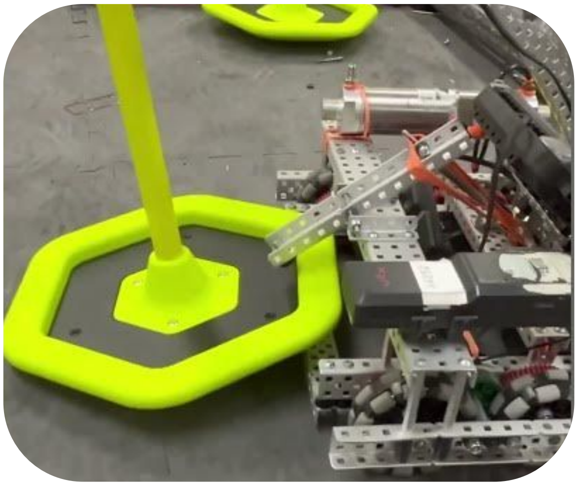


Image credit LTC Robotics



Single handedly the biggest issue facing us was contributing to the motor cap. Since we're already using 6 11w motors, we are only left 22 watts worth of power to play with, and we need to stretch those motors as far as possible.
Pneumatic Clamp it is!
We plan to tilt the mobile goal with our clamp to prevent mogo thieves and make it easier to score on,

Mobile Goal Scoring

Three options on ways to score on to mobile goals, each with its own set of challenges. Plunger, holding the center of rings, hood, pushing down onto the goal, and hook, flipping rings onto stakes. Each present member of the team ranked each idea on a scale of 1 to 5, highest score wins

Style	Speed	Integration	Reliability	Ease of building	Footprint	Totals
Plunger	2, 3, 2	2, 3, 4	3, 3, 3	5, 4, 4	2, 2, 2	44
Hood	4, 4, 4	4, 3, 3	4, 4, 4	3, 2, 3	3, 4, 2	51
Hook	5, 4, 4	3, 3, 4	4, 5, 4	4, 3, 5	3, 4, 4	59

And hook wins! It seems the simplest to build and integrate into our design, and accounts for a large margin of error when scoring rings onto stakes

Now, the question, is how to we pick rings up off the floor? Directly off the ground with a hook, or use a 2 stage intake?

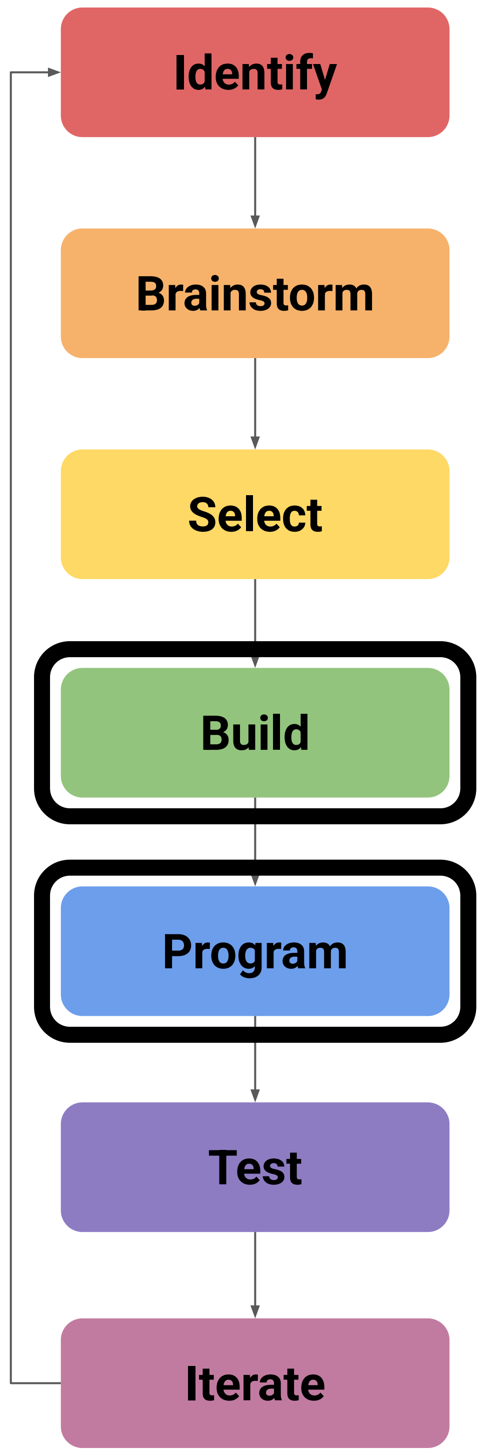
We voted to build a 2 stage intake, due to simplicity and ease to build (margin of error) vs an intake that would need to be perfect to pick up any rings.

Wall Stakes and Climb

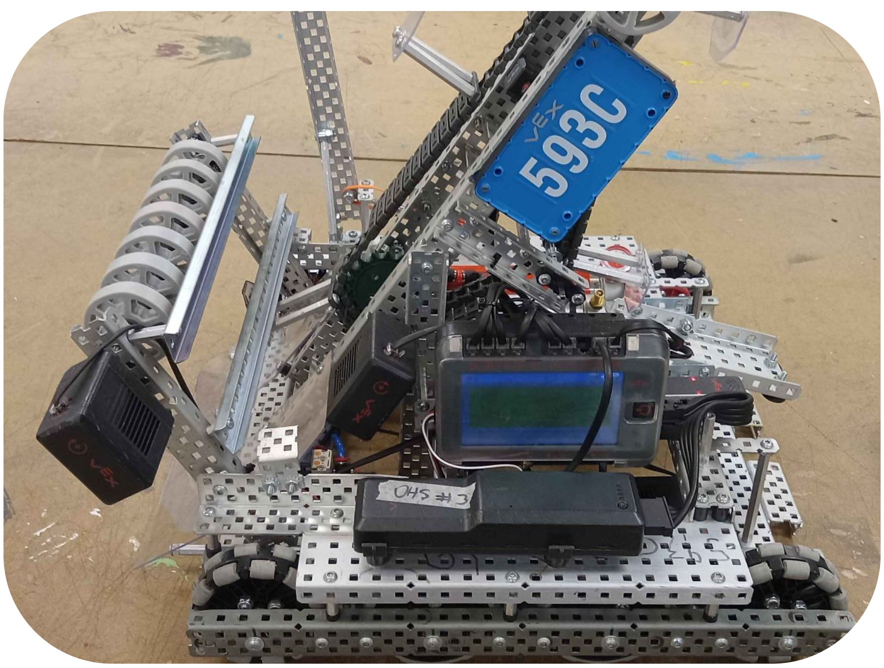
At this time, we’re unsure if we’ll even have a way to score on wall stakes. We have several options available to us, including the Lady Brown mech, which seems to be the easiest design to integrate into our design. Scoring on wall stakes seems a way off, we will revisit once we reach a stage in which we can score points on mobile goals effectively.

Same goes for a climb, while we feel that it is a very important aspect of this game, heck the ladder is the centerpiece of the field, we aren’t at a stage where a climb or a hang is a priority.
A passive hang seems the most doable, and will be looked into ASAP. Will revisit in the future.

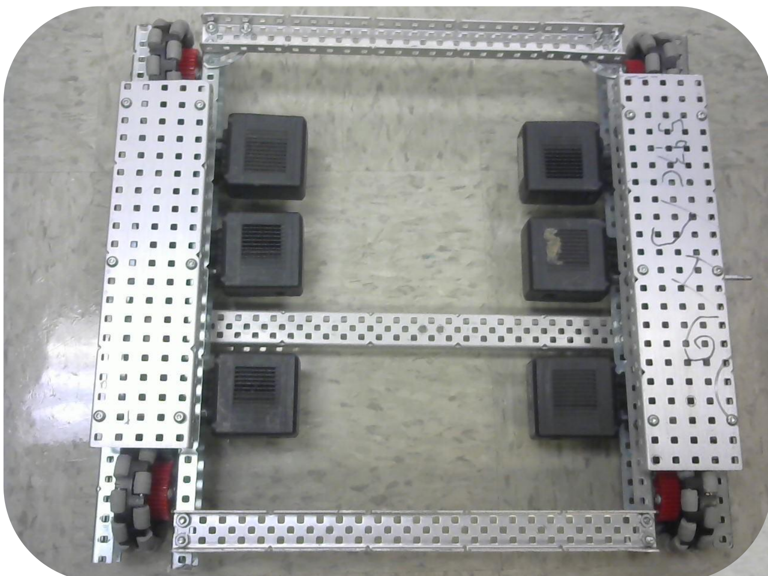
Build and Program
The EDP

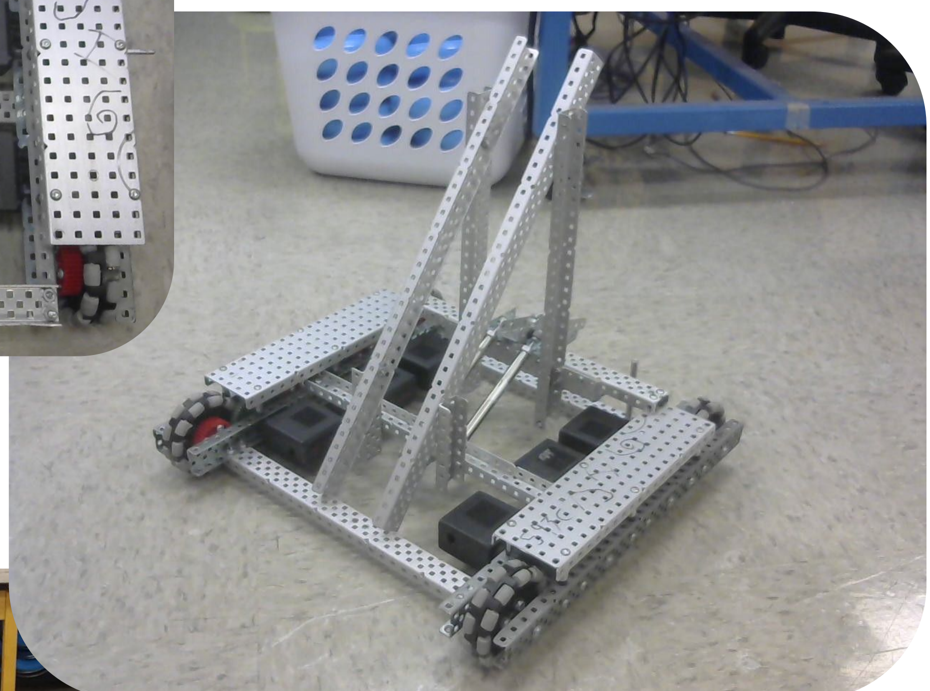


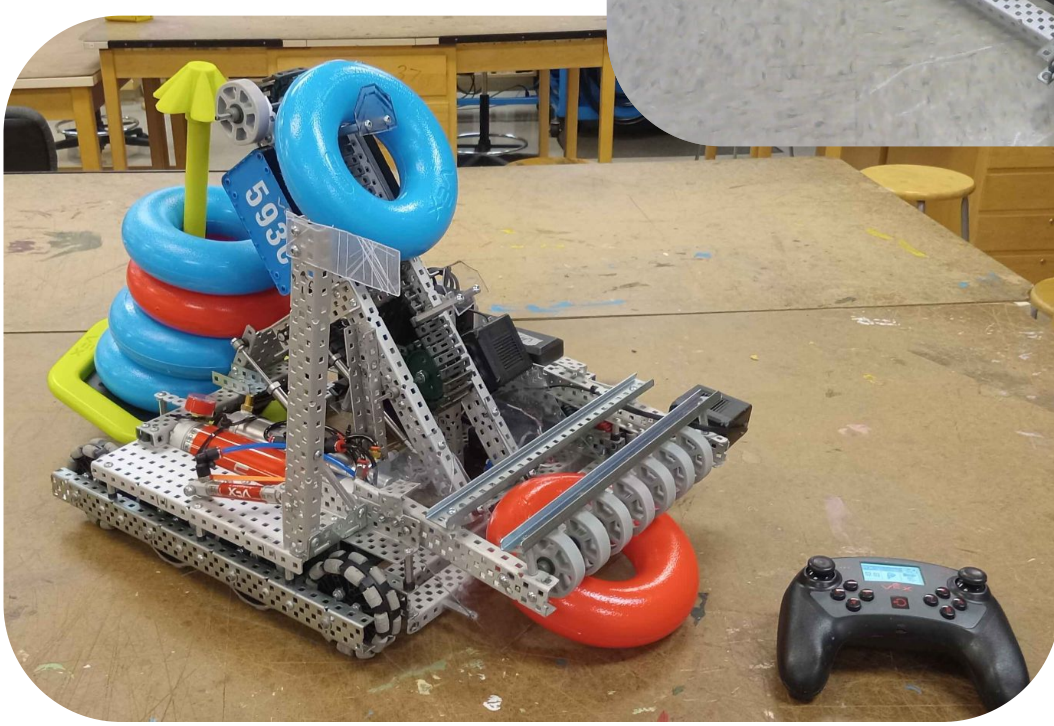
“Build and Program the Solution” - Create the solution that was chosen, and describe the design and building process. Program the solution that was chosen, document and explain the programming process



Daily Logs







8/26/24

- Today was our first meeting of the season!
- We discussed budgeting, watched/rewatched the game reveal, and discussed updates to the game over the summer.
- We also discussed meeting schedules (Mondays, Tuesdays, and Thursdays, from 2:15pm (end of school) until 4pm.
- Team captains were chosen, and team members were selected
- Groups split off, and read through the game manual.

8/27/24

- Today, we gained access to our team meeting/storage space - It is much smaller than it was last year, but no one else other than us has access to it.
- We moved much of our stuff into it, and used the remaining time discussing our research and game rules.

8/27/24

- Rowan started designing the formatting of the Notebook

8/29/24

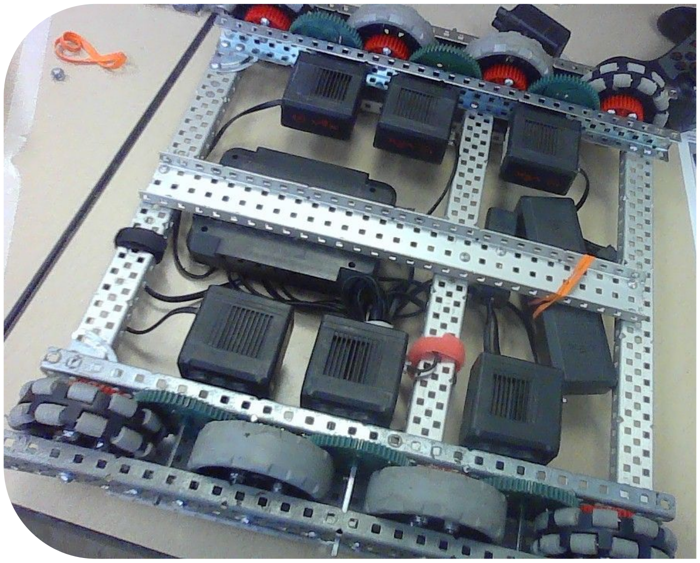
- Our coaches readjusted our teams, to more evenly distribute the veterans and new members, since more than 2/3rds of our team is new to VEX this year, or were not really involved last year.
- Preliminary sketches and brainstorming was started.
- Event attendance was discussed, and further budgeting was discussed

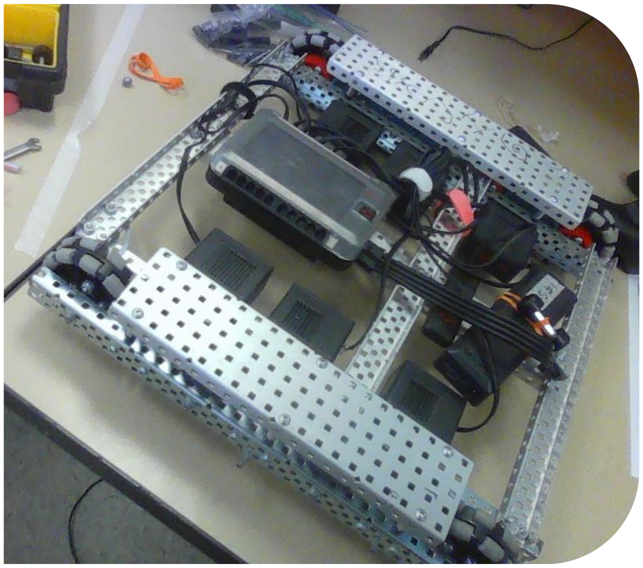
9/3/24

- Today was spent building a temporary drivetrain, using screw joints and gear ratios.
- The goal of this drivetrain is to have a basis to visualize our robot on, and to teach our new team members how the VEX construction system works.
- In the middle of the robotics meeting, the game manual was updated to Version 2.0. Rowan started the game analysis under the identify category.

9/5/24

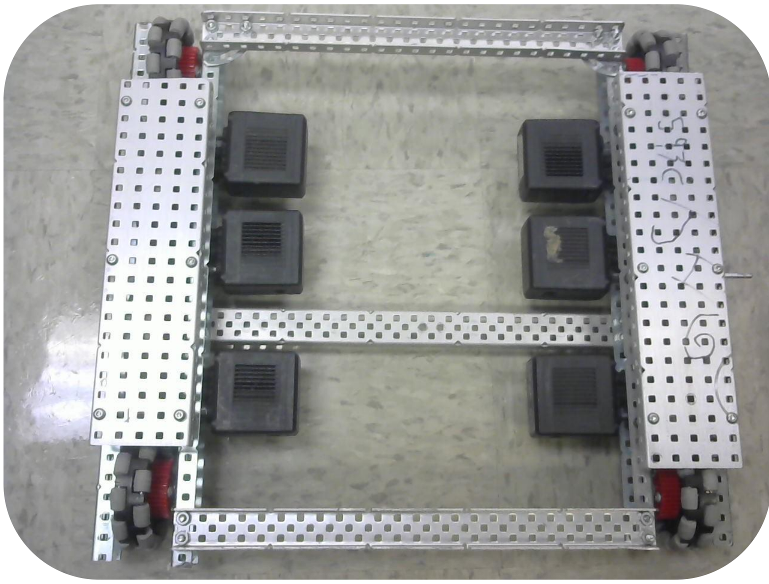
- Today, we completed both halves of our first drivetrain, linked them together and set up a very basic program to drive it in VEXcode
- We noticed there was a lot of friction on the left side of the drivetrain, and will need to address that issue on Monday.
- Sketches of our planned robot were continued, as was work on our notebook.

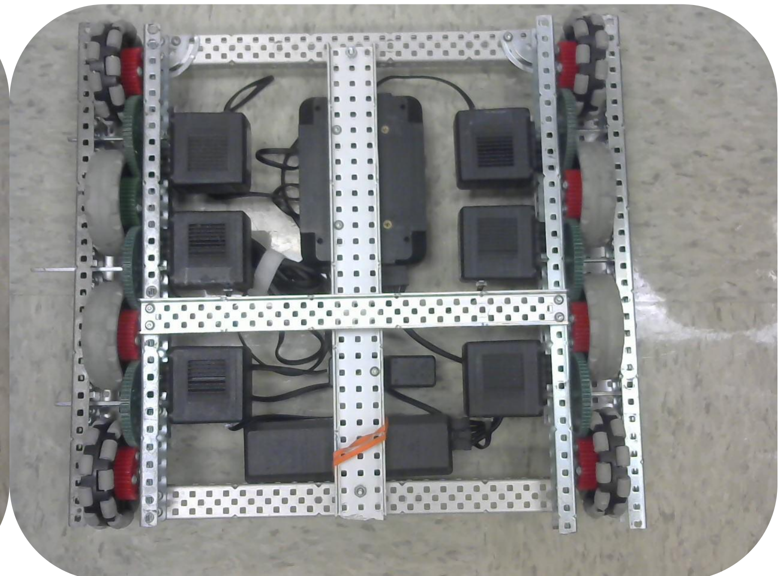




9/9/24

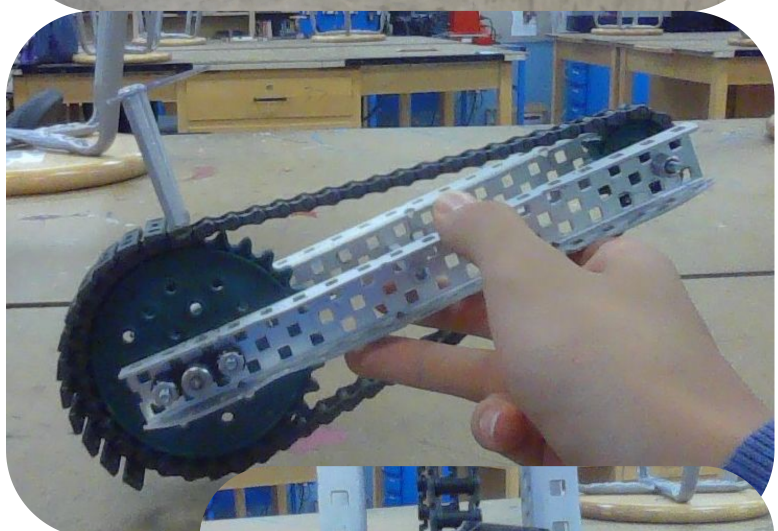
- The drivetrain was fixed, it was discovered that our brain had a dead port. It is well tuned now, and drives forward very straight.





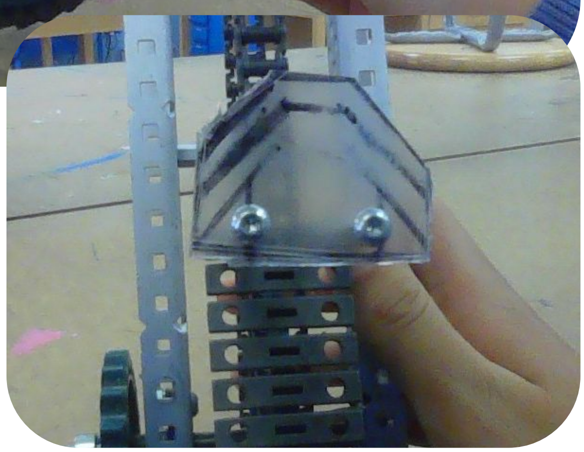
9/10/24

- We disassembled our drivetrain, and pitched our ideas to our coaches. A model hook intake was developed.



9/12/24

- We completed research on different hooks and intake mechanisms, and designed a prototype hook for our intake.

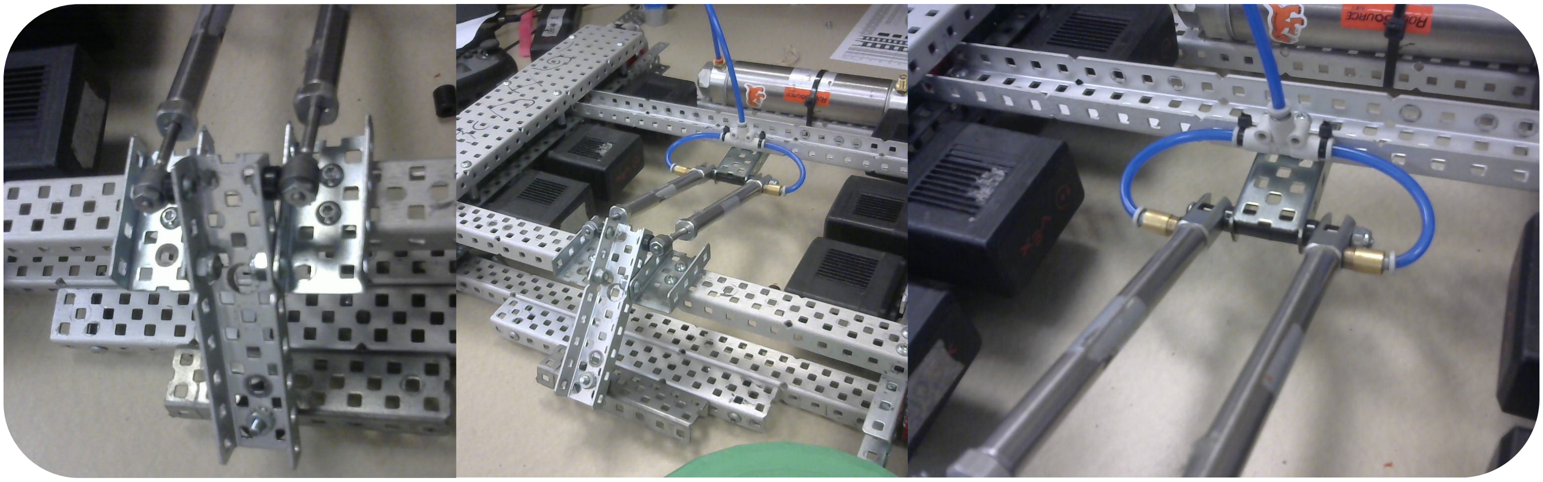


9/16/24

- Reassembled robot, with minor adjustments.
- Work on the design of our mobile goal clamp was started.

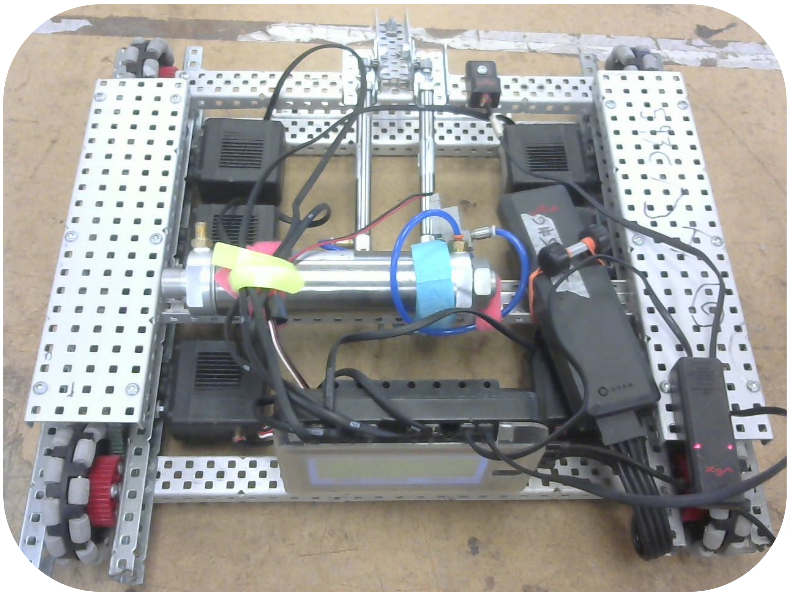
9/17/24

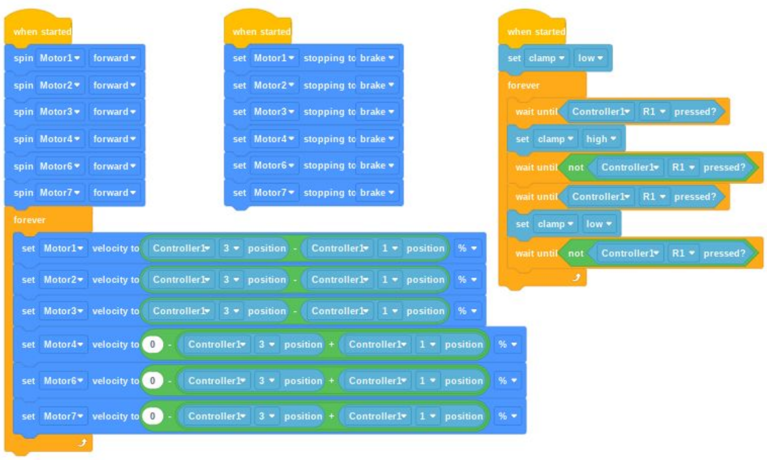
- A preliminary design for our mobile goal clamp was completed, stress tested, and leak tested.



9/19/24

- A simple programming drive was completed, with a simple drivetrain created in VEX Code





9/23/24

- We continued to complete research on different hooks and intake mechanisms.
- Rowan started converting our program into PROS, and completed preliminary research into LemLib

9/24/24

- We continued to complete research on different hooks and intake mechanisms.
- Rowan continued research on PROS, and LemLib

9/26/24

- Rowan finished converting our drivetrain program into PROS, and was able to drive the robot around.

```cpp
#include "main.h"
#include "lemlib/api.hpp" // IWYU pragma: keep

pros::MotorGroup rightMotors({1, 2, 3}, pros::MotorGearset::green);
pros::MotorGroup leftMotors({4, 6, 7}, pros::MotorGearset::green);
pros::Controller master (CONTROLLER_MASTER);

void initialize() {
        pros::lcd::initialize();
        pros::lcd::set_text(0, "Oakdale 593C: Bearly Functioning");

}

void disabled() {}

void competition_initialize() {}

void autonomous() {}

void opcontrol() {
        while(true) {
                rightMotors.move(master.get_analog(ANALOG_LEFT_Y) - master.get_analog(ANALOG_RIGHT_X));
                leftMotors.move(- master.get_analog(ANALOG_RIGHT_X) - master.get_analog(ANALOG_LEFT_Y));

        }
}
```

Import other libraries needed to function

Set up controller and both sides of the drivetrain

Show text on screen to ensure the correct program is running on start

Functions that start when sections of the match start

Math to control the velocity of the motors based on the position of the joysticks on the controller

10/1/24

- Our coaches pulled the plug on monday practices, not enough members were attending to make it feasible to meet on mondays.
- Chloe and Ajay continued research on a hook mechanism, using various sources on the web.
- Rowan continued programming, enabling remote control of our mobile goal clamp.

```
void driverClamp() {
    mogoClamp.set_value(false);

    while(true) {
        waitUntill(masterCont.get_digital_new_press(DIGITAL_R1));
        mogoClamp.set_value(true);
        waitUntill(!masterCont.get_digital_new_press(DIGITAL_R1));
        waitUntill(masterCont.get_digital_new_press(DIGITAL_R1));
        mogoClamp.set_value(false);
        waitUntill(!masterCont.get_digital_new_press(DIGITAL_R1));
    }
}
```

Open the clamp

“False” means the clamp is open, “true” means it is closed.

On the press of R1, close the clamp

On the press of R1 again, open the clamp

10/3/24

- Further programming progress was made, as Rowan finished setting up LemLib, a programming library for creating fancy autonomous paths.
- Chloe and Ajay rigged up a tester for our hook design, to ensure the idea had grounding in the real world.

10/8/24

- Rowan fixed some small issues with driving and speed control to make the controller less sensitive.

10/10/24

- Continued progress on programming, improved control with the controller

```
void driverDriver() {
        rightMotors.move(200);
        leftMotors.move(200);          Enable motors, do not allow to spin

        rightMotors.set_brake_mode(pros::E_MOTOR_BRAKE_HOLD);   Make the drivetrain stop and lock in place when it
        leftMotors.set_brake_mode(pros::E_MOTOR_BRAKE_HOLD);    stops receiving signal from the controller

        while(true) {   Math to make the drivetrain spin at full speed,
                rightMotors.move_velocity((masterCont.get_analog(ANALOG_LEFT_Y)*1.575) - (masterCont.get_analog(ANALOG_RIGHT_X)*1.575));
                leftMotors.move_velocity((masterCont.get_analog(ANALOG_LEFT_Y)*1.575) + (masterCont.get_analog(ANALOG_RIGHT_X)*1.575));
        }
}
```

10/15-17/24

- Continued research into designs and ideas
- Progress on notebook
- Whole team meetings about progress and planning for the future

10/22/24

- Continued notebooking
- Redesigned mogo clamp
- Waiting on new parts

10/24/24

- Created cardboard mobile goal to ensure design was actually feasible
- Driving practice
- Waiting on new parts

10/29/24

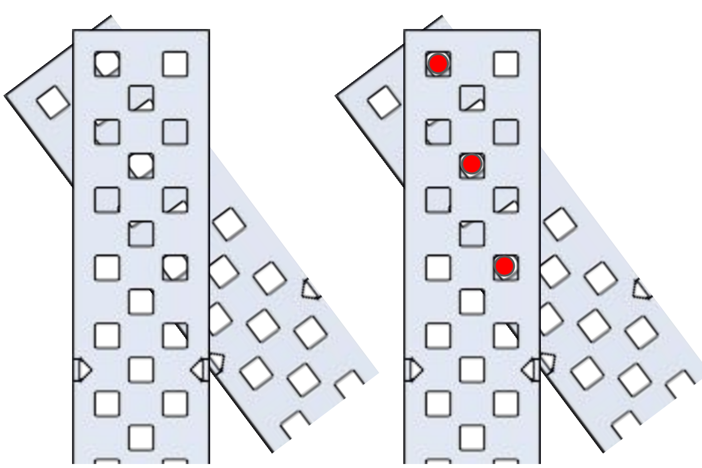
- New parts arrived!
- Rowan hosted a pneumatics workshop for the other Oakdale teams to try to get them up to speed with the new pneumatic kits.
- Drivetrain disassembled and reassembled to demonstrate how it works

10/31/24

- Rowan made a proof of concept odometry pod, to be used in the future
- Small friction issues fixed on the drivetrain

11/5/24

- Rowan started work to integrate Ajay and Chloe’s idea for the hook intake onto the chassis, at a larger scale.
- With VEX Structure, the holes are spaced in a way which you can get a perfect 45° angle if you align the holes in a certain way. This seems to work, and is how we plan to mount the two halves of our structure



Credit ACE 229V for the picture

11/7/24

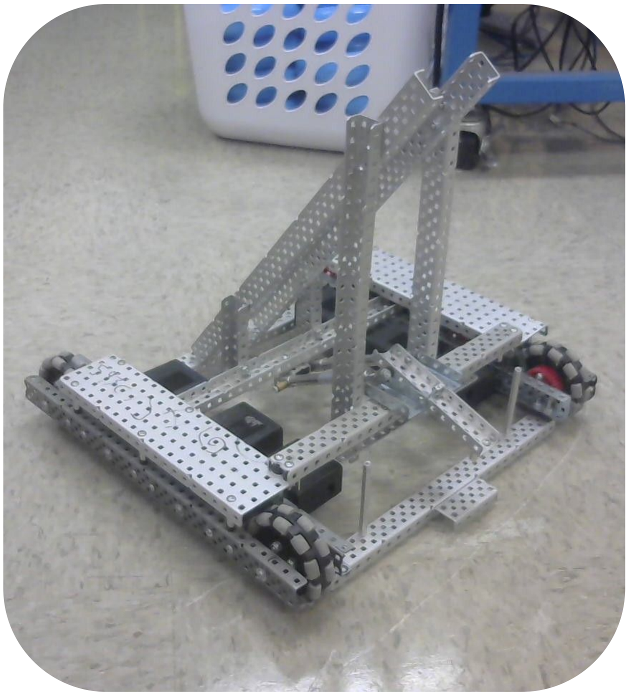
- Rowan continued work on the triangle, attaching the other side onto the chassis.
- Chloe and Ajay worked on creating the chain for the intake

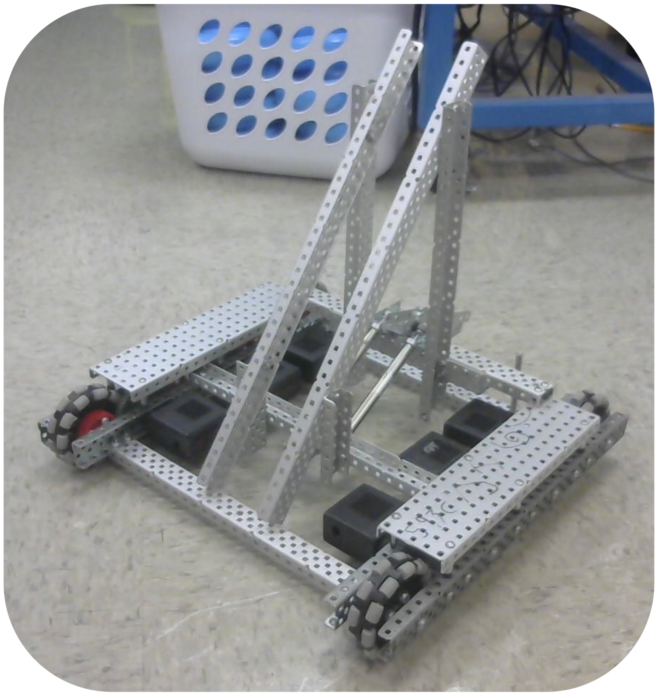
11/12/24

- Removed progress from last week, not feasible in the long run
- Made our clamp wider to account for inaccuracies
- Reattached triangle for hook, with a VEX Hinge this time to have a more secure connection

11/14/24

- Attached the chain, motors, and a tensioner to the triangle to create something functional
- Can not score on a stake currently, ring does not flip far enough to land on a stake





11/19/24

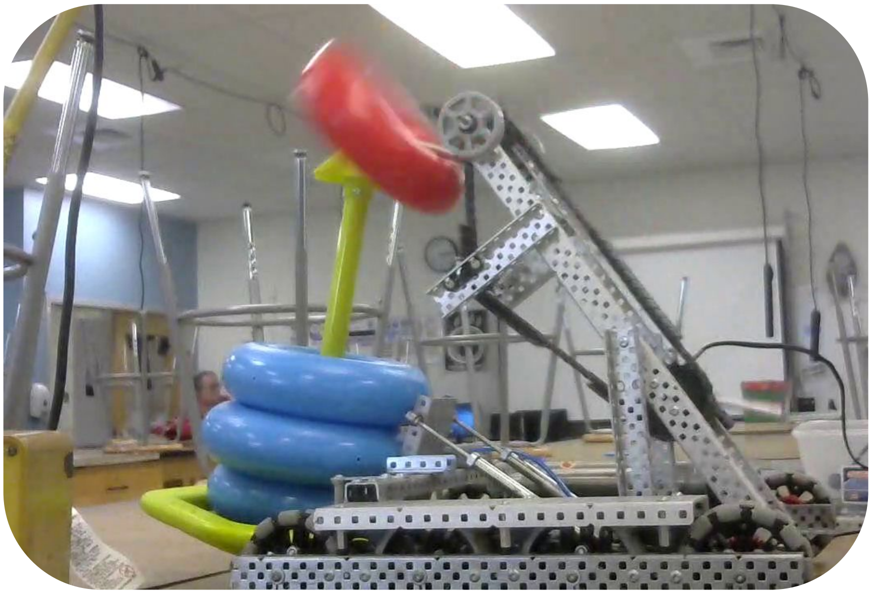
- Success! By placing flex wheels on either side of our top hook, we were able to score onto a clamped stake!
- No programming has been started, all motor control is manual currently

11/21/24

- Continuously tested the hook arm, made adjustments
  - Moved tensioning rod down
  - Replace tensioning gear with a spacer
- Work on an intake was started, by bending plastic and creating an arm to pull in rings

11/26/24

- Work on the intake continued, we are facing issues of constant jamming between the stages of the intake and hook
  - Chloe cut new hooks out of plastic



12/3/24

- Chloe started work on a sweeper arm, one designed to clear the corner of rings to we can place a mobile goal inside.
- Rowan fixed issues with the chain on our hook intake breaking, and continued tuning our intake

12/5/24

- Chloe continued work on the sweeper arm, and attached it/ removed it with Rowan’s help

12/10/24

- First successful full stake without any manual help! https://www.youtube.com/watch?v=wWe_xZpMluE
- Robot was programmed and tested, small changes to the clamp were made to make it easier to get consistent clamps on the mobile goals.

Newer photos from end of Dec, show sweeper concept well enough, albeit much improved. Old sweeper only had one attachment point, this has many more.

12/10/24

- Programming for our stake scoring and intake is very simple, and controlled by two triggers, R1 and R2

```
// driver control, control intake / conveyor
void driverHook() {
        conveyor.move(200);
        
        while(true) {
                if(masterCont.get_digital(DIGITAL_R1)) {
                        conveyor.move_velocity(200);
                } else if(masterCont.get_digital(DIGITAL_R2)) {
                        conveyor.move_velocity(-200);
                }
                else {
                        conveyor.move_velocity(0);
                }
        }
}
```

Spin motors forwards, set velocity to 0

While R1 is held, spin conveyor forwards

While R2 is held, spin conveyor backwards

Else, do not spin the intake or hooks

As programmed a while ago, the waitUntill() function is very simple, it waits until the opposite of an action happens

```
#define waitUntill(condition) while (!(condition)) {pros::delay(50);}
```

Define the function

While the condition is not met, wait until the condition is met. Check every 50 milliseconds

12/10/24

- Reprogrammed sweeper arm and clamp to use one button, which greatly simplifies the code
- driverClamp is code for the mogo clamp, driverSort is code for the sweeper arm

```
// driver control, control clamp
void driverClamp() {
        mogoClamp.set_value(false);
```

Keep clamp up

```
        while(true) {
                waitUntill(masterCont.get_digital_new_press(DIGITAL_L1));
                mogoClamp.set_value(true);
                waitUntill(masterCont.get_digital_new_press(DIGITAL_L1));
                mogoClamp.set_value(false);
        }
}
```

On press, close clamp, On press, open clamp

```
void driverSort() {
        sortClamp.set_value(false);
```

Keep arm up

```
        while(true) {
                waitUntill(masterCont.get_digital_new_press(DIGITAL_L2));
                sortClamp.set_value(true);
                waitUntill(masterCont.get_digital_new_press(DIGITAL_L2));
                sortClamp.set_value(false);
        }
}
```

On press, lower arm On press, raise arm

12/12/24

- Our sweeper arm was removed, again, to fix it
- Intake removed and reattached to strengthen it

12/17/24

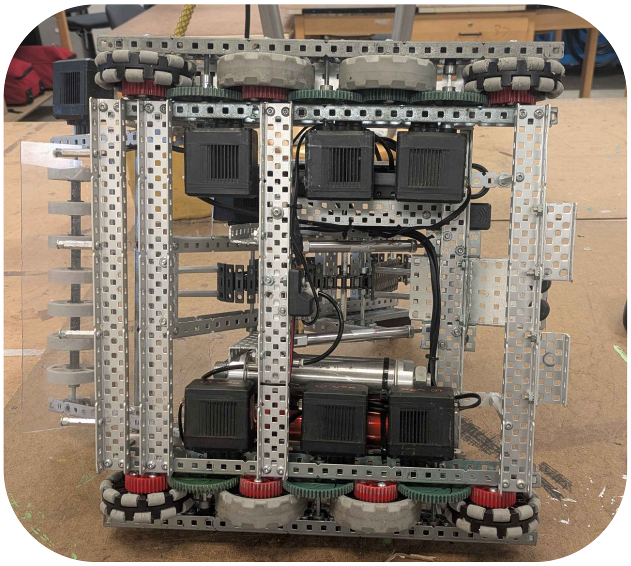
- Sweeper arm was reinforced and replaced by Rowan and Chloe
- Utilized a vex hinge to have a stronger attachment point than just one screw

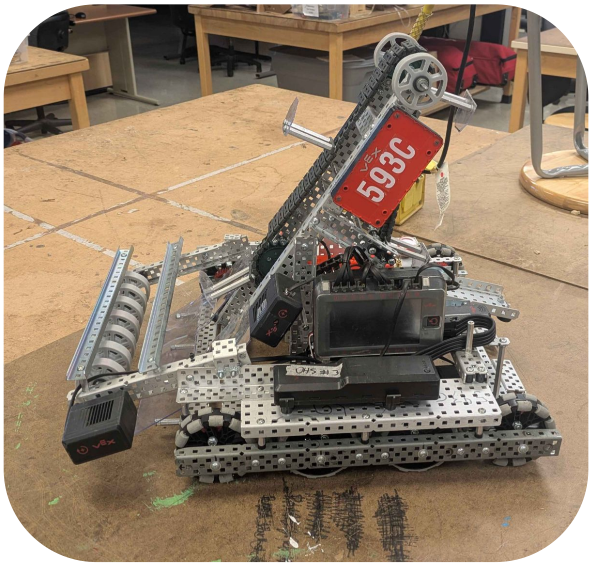
12/19/24

- Christmas party!
- Photos for the notebook were taken
- Driving practice was completed
- Power for the arm to sweep out the corner was added and wired/programmed.

12/25/24

- Merry Christmas! Let the notebooking bonanza commence!



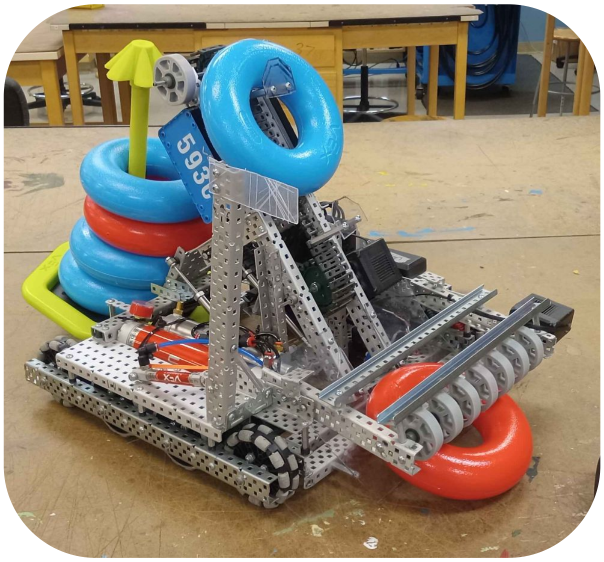


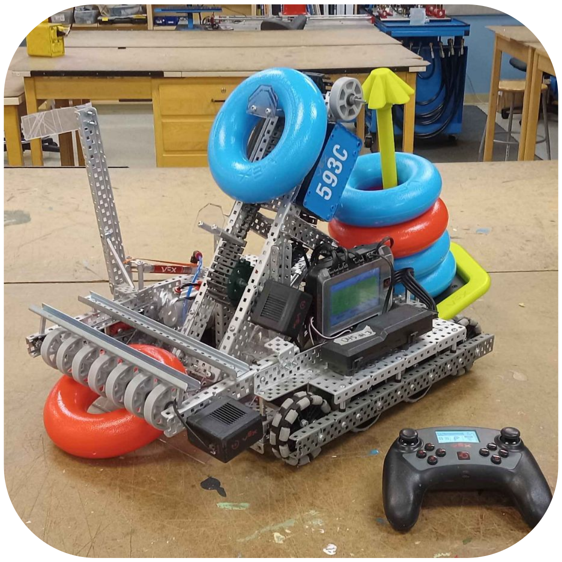
1/1/25
- Happy New Year! Rowan continued work on the notebook

1/2/25
- The color and logo for our team shirts was selected, with our old logo on a grey shirt, with black text.
- Old logo to the right ------------>
- A weird grinding and screeching noise was emanating from the drivetrain, a few screws came loose, and we think we stripped a motor. To be fixed.

1/6-7/25
- Snow day! Rowan continued work on the notebook.

1/9/25
- Issues with the drivetrain were resolved
- Programming autonomous and color sorter commenced





1/9/25

Current lemlib setup includes a inertial sensor, everything else is as described in the documentation.
Formatting is strange on Github, sorry

```
// odometry sensor settings
lemlib::OdomSensors sensors(nullptr,nullptr,nullptr,nullptr,&imu);

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

// angular PID controller
lemlib::ControllerSettings angular_controller(2, // proportional gain (kP)
                                              0, // integral gain (kI)
                                              10, // derivative gain (kD)
                                              3, // anti windup
                                              1, // small error range, in degrees
                                              100, // small error range timeout, in milliseconds
                                              3, // large error range, in degrees
                                              500, // large error range timeout, in milliseconds
                                              0 // maximum acceleration (slew)
);

// create the chassis
lemlib::Chassis chassis(drivetrain,
                                                lateral_controller,
                                                angular_controller,
                                                sensors
);
```

1/9/25

Autonomous Programming!

Rowan set up a helper function for movement, then set up a preliminary auton program to score 2 rings onto a stake.

```cpp
// auton drive controlls
void autonDrive(float he, float ie, int jv) {
	leftMotors.tare_position_all();
	rightMotors.tare_position_all();

	float hd = (he / 360);
	float id = (ie / 360);

	pros::lcd::print(4, "id: %f", id);
	pros::lcd::print(5, "ie: %f", ie);

	rightMotors.move_relative(id, jv);
	leftMotors.move_relative(hd, jv);
	// wait until motors reach pos
	while (!((rightMotors.get_position() < id+0.1) && (rightMotors.get_position() > id-0.1))) {
		pros::delay(2);
	}
}
```

Tare all encoder values to reset them

Convert degrees to revolutions

Bug fix, ensure conversion worked

Spin motors _ revolutions

Keep running program until motors reach target

```cpp
void autonomous() {
	rightMotors.set_brake_mode(pros::E_MOTOR_BRAKE_HOLD);
	leftMotors.set_brake_mode(pros::E_MOTOR_BRAKE_HOLD);
```

Lock motors in place, only move when told

```cpp
	autonDrive(-600, -600, 50);
```

Move forward to clamp mogo

```cpp
	// rightMotors.move_relative(-1.5,50);
	// leftMotors.move_relative(-1.5,50);
```

Bugfixes

```cpp
	mogoClamp.set_value(true);
```

Clamp mogo

```cpp
	conveyor.move_relative(2000, 100);
	pros::delay(1000);
```

Score ring on mogo within 1 second

```cpp
	conveyor.move_relative(20000, 100);
	autonDrive(330, -330, 50);
```

Turn on intake, turn to second ring

```cpp
	autonDrive(360, 360, 50);
```

Advance to 2nd ring, intake and score

```cpp
	autonDrive(-600, 600, 50);
```

Turn to face ladder, run into ladder to earn ¾ AWP

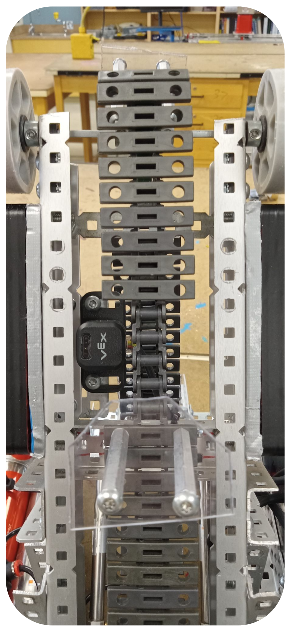
```cpp
	autonDrive(600, 600, 50);
```

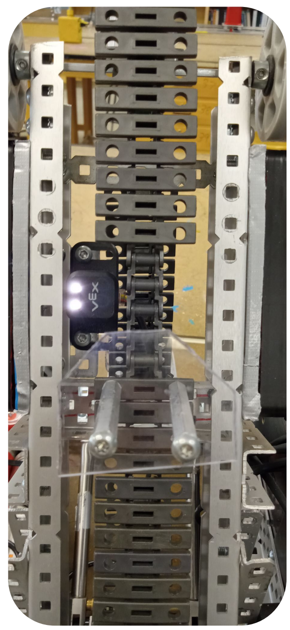
Color Sorting

Color sorting is an idea we came up with about half way through the season, and we've tried various designs. About mid way through December, Rowan spent an afternoon trying to use pneumatics to eject rings of the opposing color, to much avail. Instead, we've found a coding solution seems to work better than a physical solution.

Color sorting is important during a hectic match, where you only want to give points to yourself, and not score points for the opposing team. It takes much longer to outtake a ring of the opposing color, in addition to the driver needing to pay closer attention to the color of the rings they're scoring, vs an automatic program that ejects the rings faster for you.

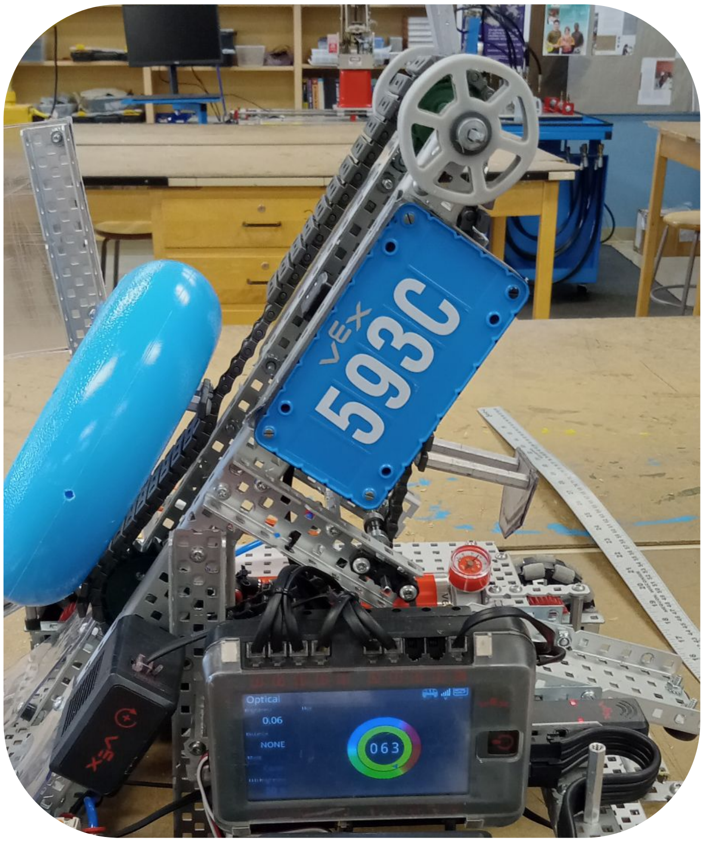
To enable our color sorting dreams, we installed a optical sensor about ¾ the way up our conveyor belt. This color sensor can detect the color of rings before they reach the top where it will pause briefly, causing the ring to go sailing into the air instead of being scored onto a stake.

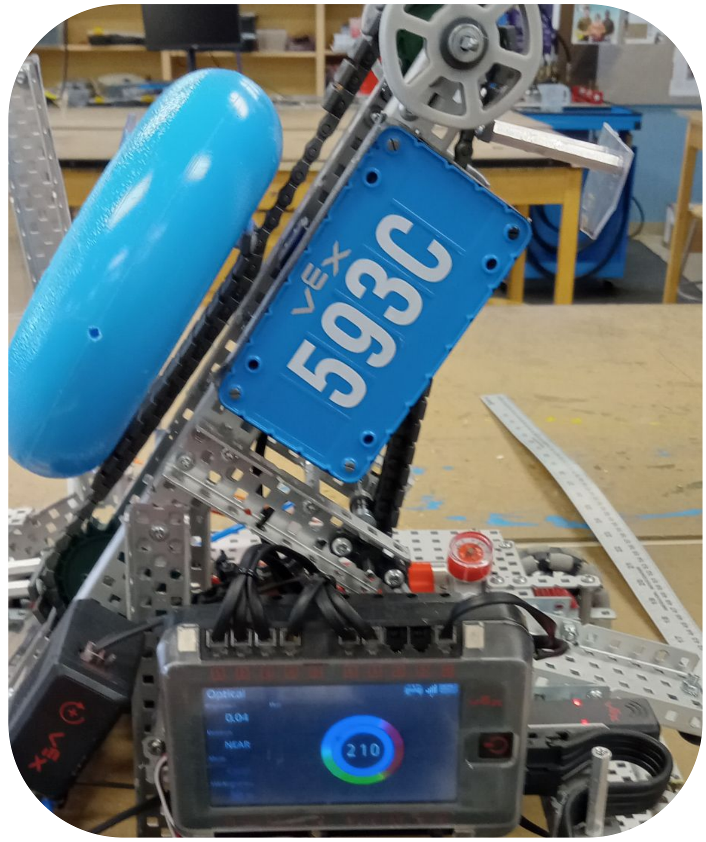




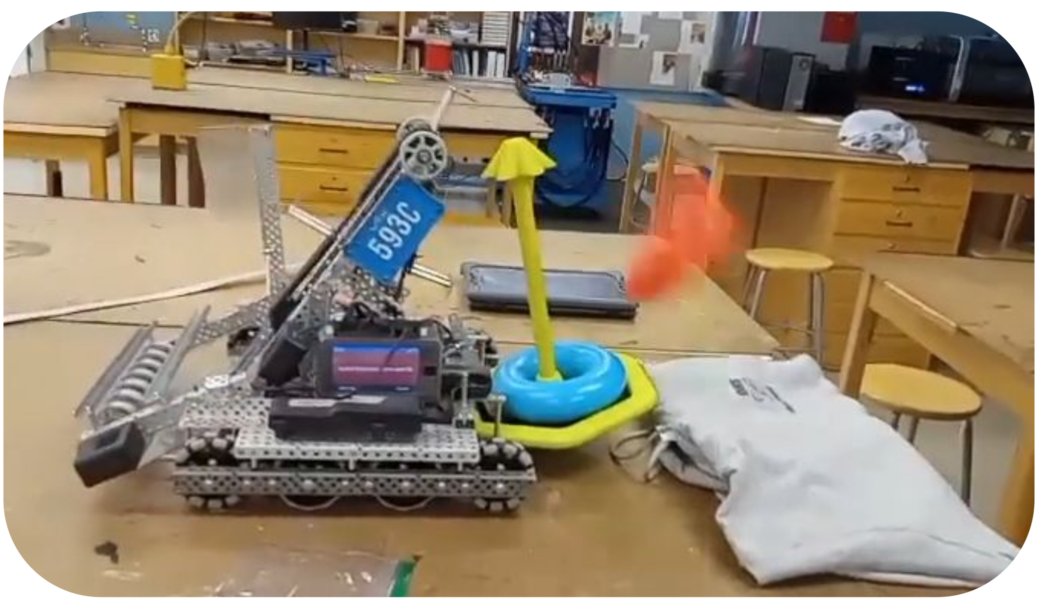
You can watch it in action at https://youtu.be/b-4utlWAViM

Color Sorting



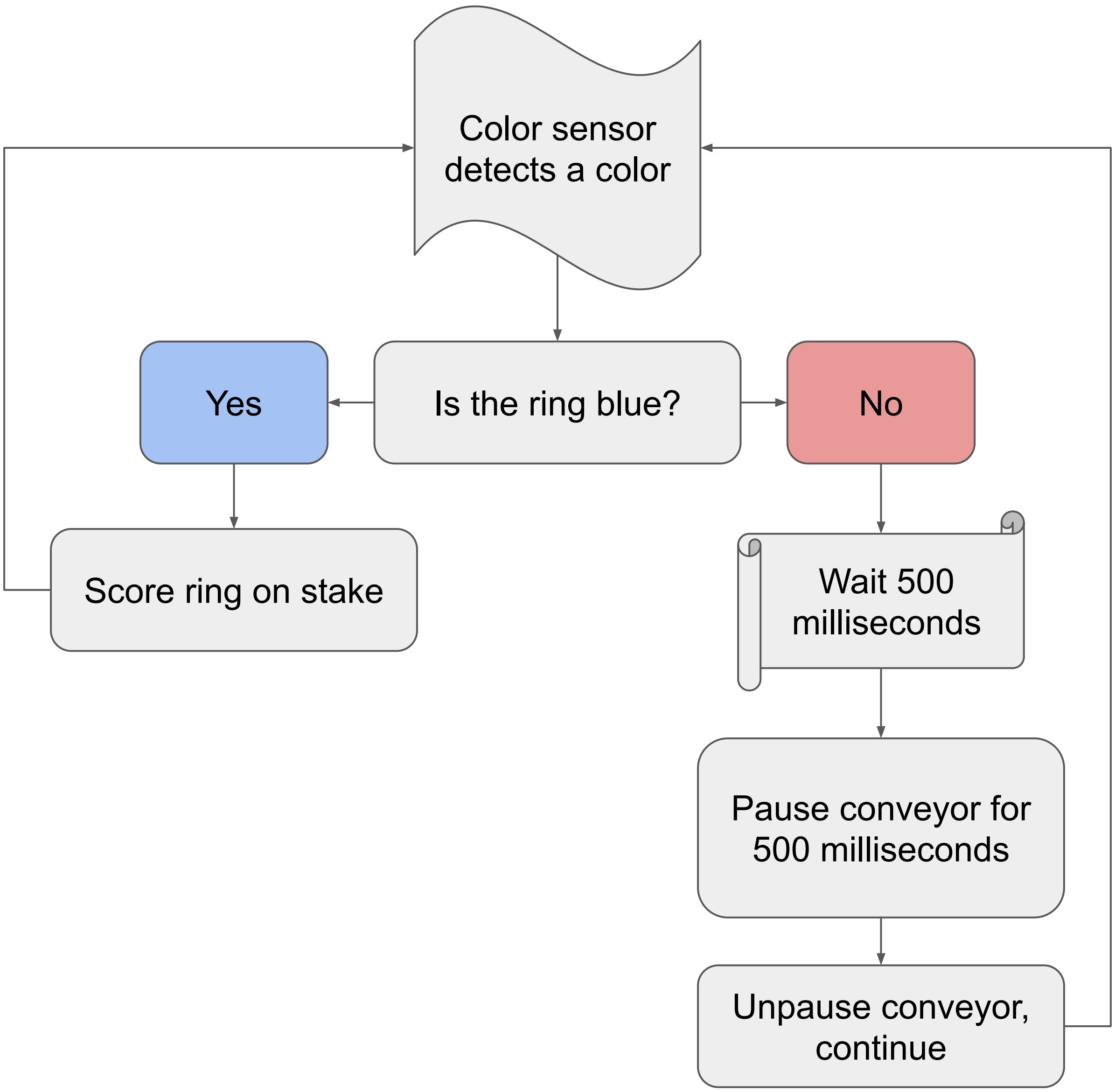


Here, you can see the optical sensor before and after detecting the color of the ring - if it is a color it allows, the ring gets scored, otherwise, fling!!, the ring gets flung off the hook

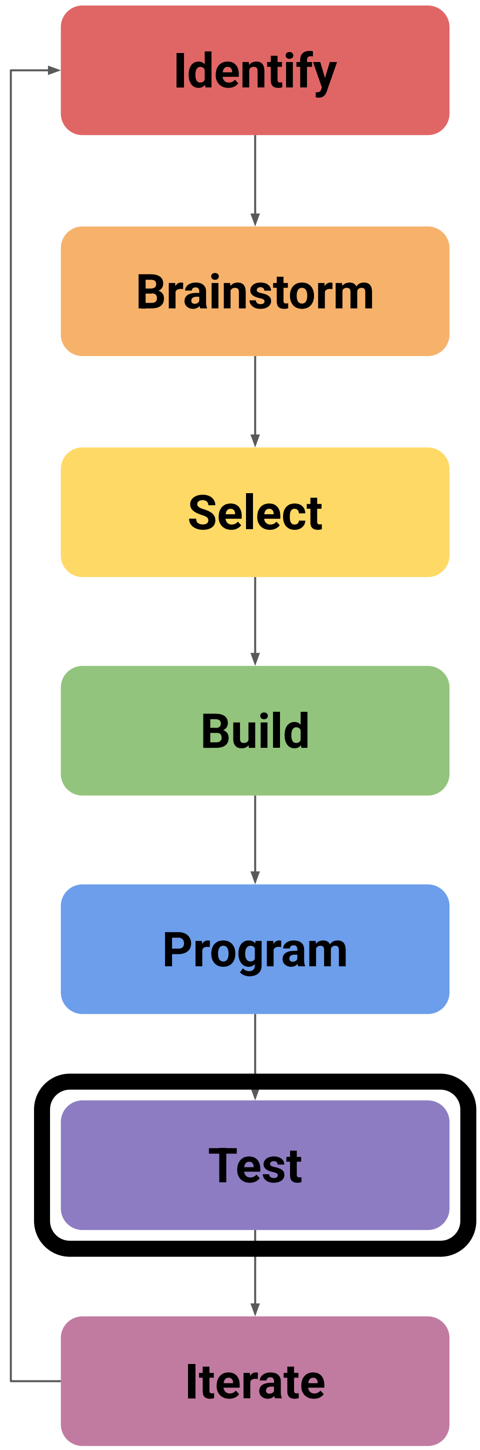


Color Sorting

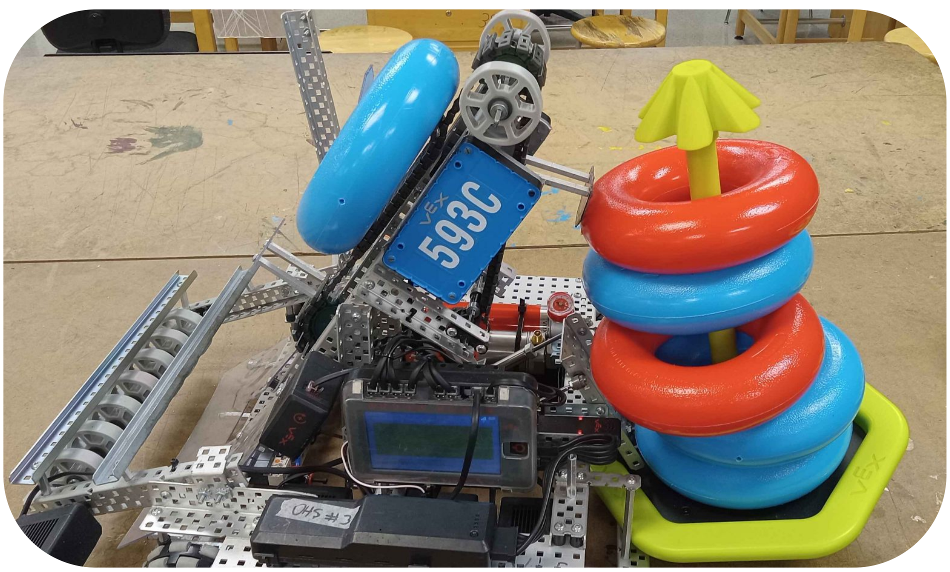
Programming is very simple, see the flowchart below for a simplified version of the code, if we want to score blue rings. The opposite is true for if we want to score red rings.
This logic is controlled by a toggle set to the X button, which is easy to toggle in a match, between the three stages. Text is displayed on the controller of which ring it is ejecting and which it is keeping.



Test Solution
The EDP



“Test Solution” - Records testing of the solution, including test results and a summary

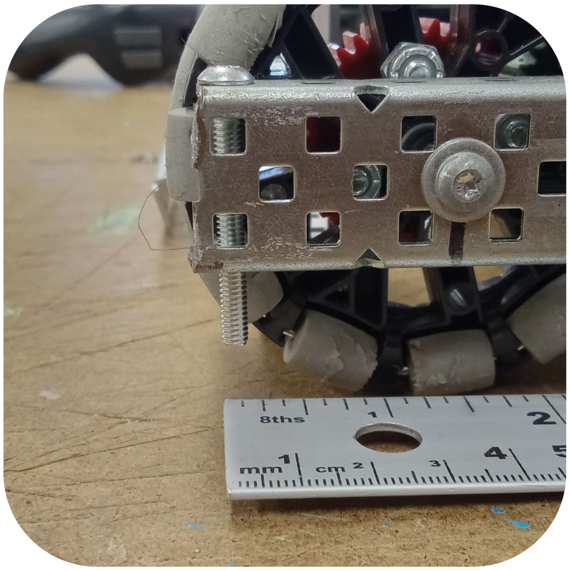


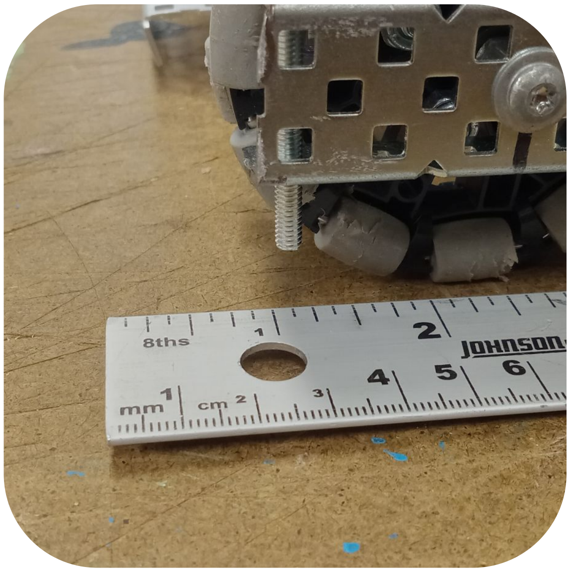
Testing effectiveness of robot

We are unable to effectively test our robot in scrimmage matches, as there are no other teams that have effective robots currently.

However, we can test for slop within our drivetrain, precision, and scoring viability.

First, testing for slop.
We placed a screw through the drivetrain, and pulled the drivetrain as far forwards as it would go, before the motors caught it. We then placed a ruler beneath the screw, and pushed the drivetrain as far back as it would go.





From this test, we find that our drivetrain has ~1in of slop, or area where the robot is not very precise, due to spacing between teeth of gears outside and inside motors, and around axles.

This data means our robot's positional accuracy will always be ± 0.5 inches from where we expect it to be, which we will need to account for in auton. This can be proven by movement tests

Testing Accuracy

For the setup, we programmed our robot to travel 1, 3 and 6 feet forwards, stop, then return back to the start, to test for positional accuracy.

	Test 1	Test 2	Test 3
1 ft	1/16 in ahead	1/8 in behind	1/16 in ahead
3 ft	1/4 in ahead	1/4 in behind	1/2 in ahead
6 ft	1/2 in ahead	3/16 in behind	1/16 behind

This test shows that our robot is more precise than we realized, it never reached worst case scenario for slop in testing.
While this may not be the best test, as the robot was traveling at 75% velocity without any turns, it shows that short autonomous programs are possible to be completed accurately.

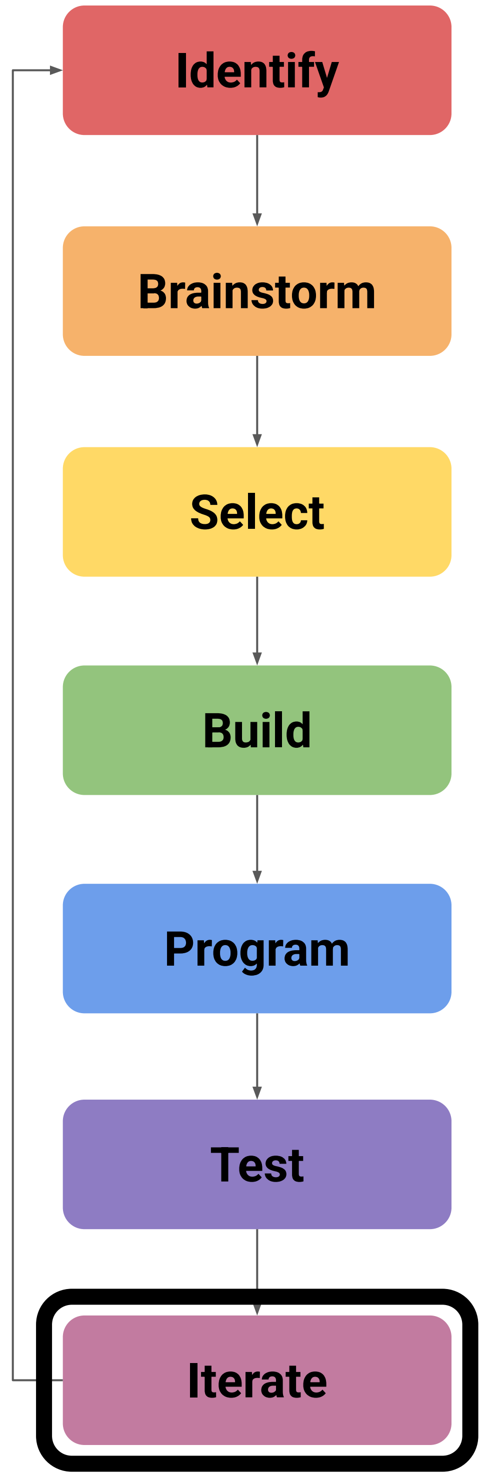
Testing Scoring

To test scoring percentage, we tested various angles where we grabbed the stake from, where our robot would score on the mogo

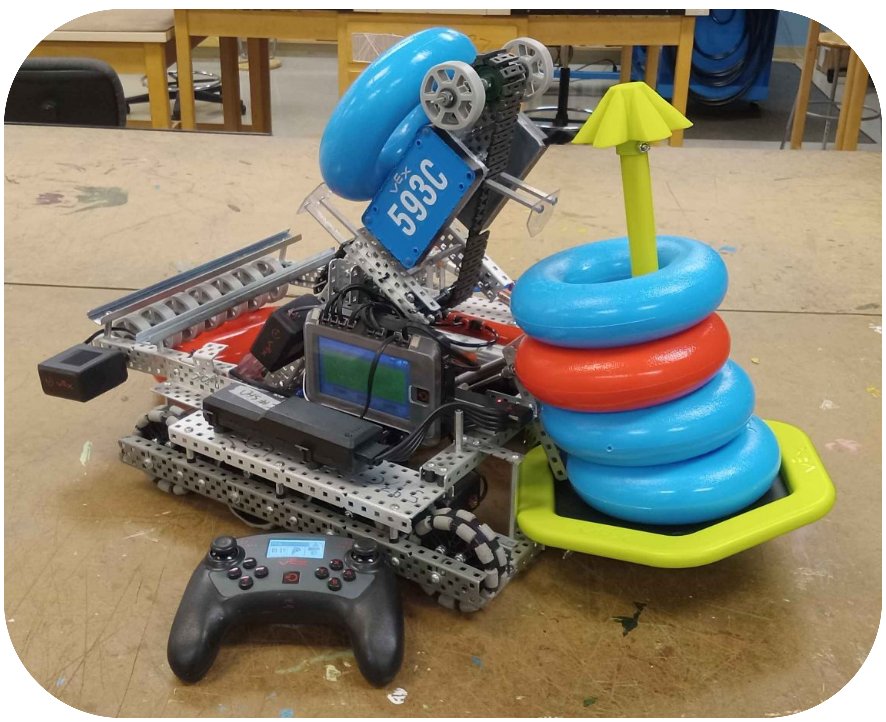
	Test 1	Test 2	Test 3
Clamp on rim	Fail	Fail	Fail
Clamp on flat edge, tilt	Success	Success	Success
Clamp on corner	Success	Success	Fail
Clamp catty corner	Success	Fail	Fail

Our hook intake can score rings 100% of the time when clamped correctly, but only half the time when incorrectly. This is important to keep in mind, to ensure rings are being scored correctly.

Iterate
The EDP



“Repeat Design Process” - Records what went right, wrong, and what we can improve.



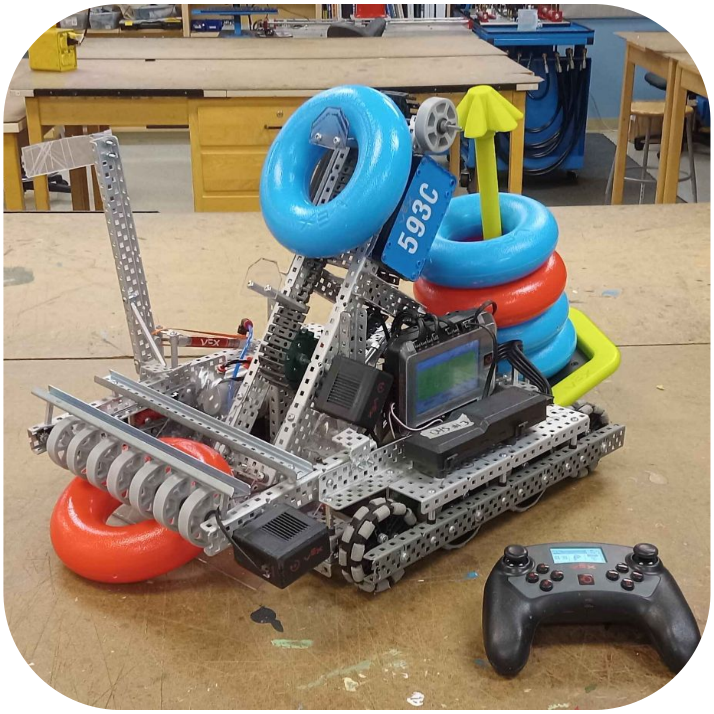
Iteration

Iteration is present throughout the entire design process, and is documented throughout. We've encountered many issues we've had to solve; ramp is too steep to carry rings up, mogo clamp is bending pneumatic pistons, sweeper arm is flexing too much or is too wide, etc. We've had to return to the drawing board hundreds of times throughout building our robot, sometimes without even noticing it.

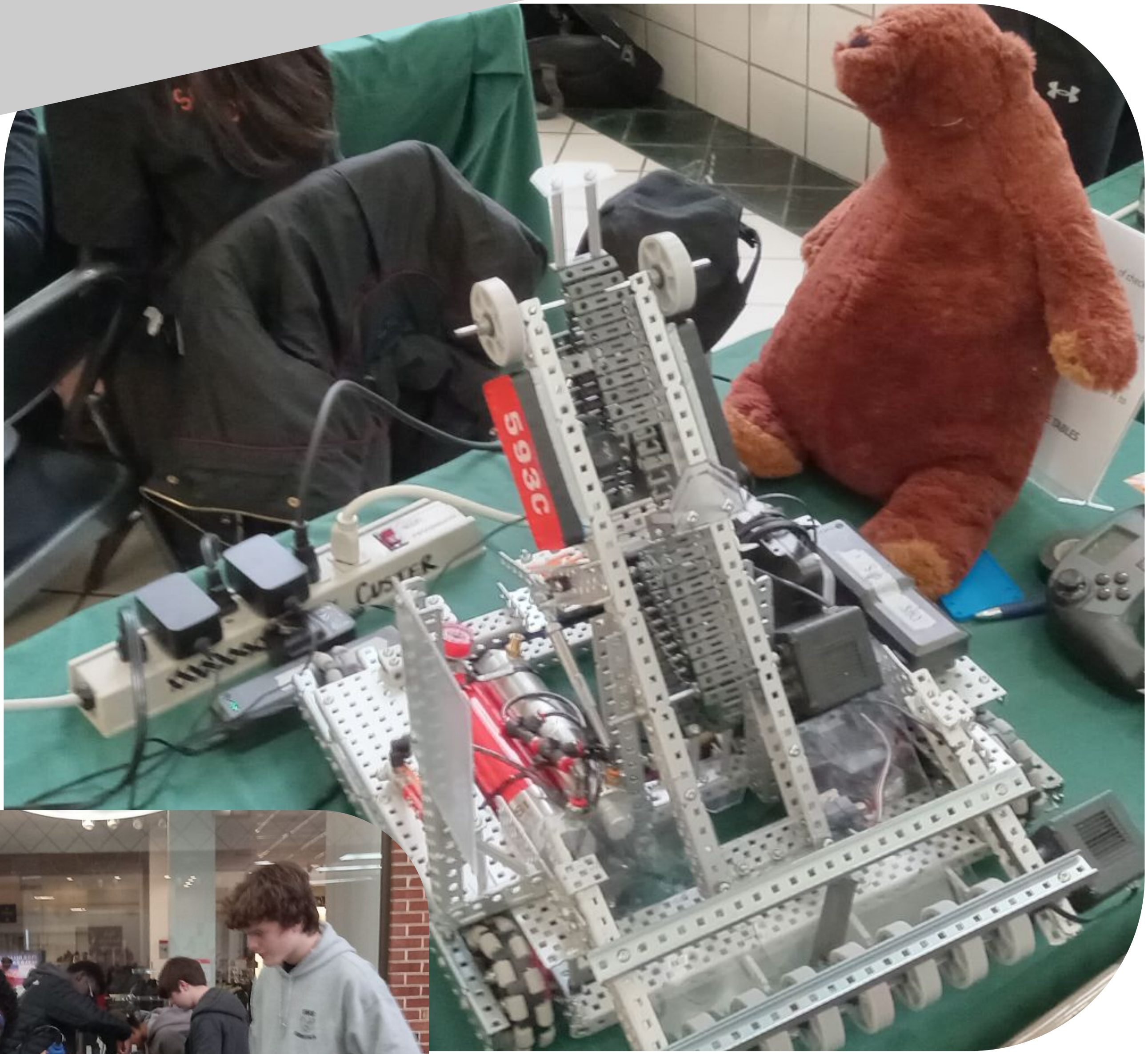
A massive thing we do need to work on throughout our iteration cycles, is documentation. We'll admit it, we are not that good at it. Much of the iteration is cut out, because we found it boring.

This is almost the same robot we started with 6 months ago, ship of theseus and all that, but the general idea we started with all those months ago stayed much the same in general. Many of the subsystems we had planned originally we found just would not work, and had to be improved on.

There are several things we would like to improve on, namely being our clamp, we believe that will be the biggest failing point for us.



Event Review



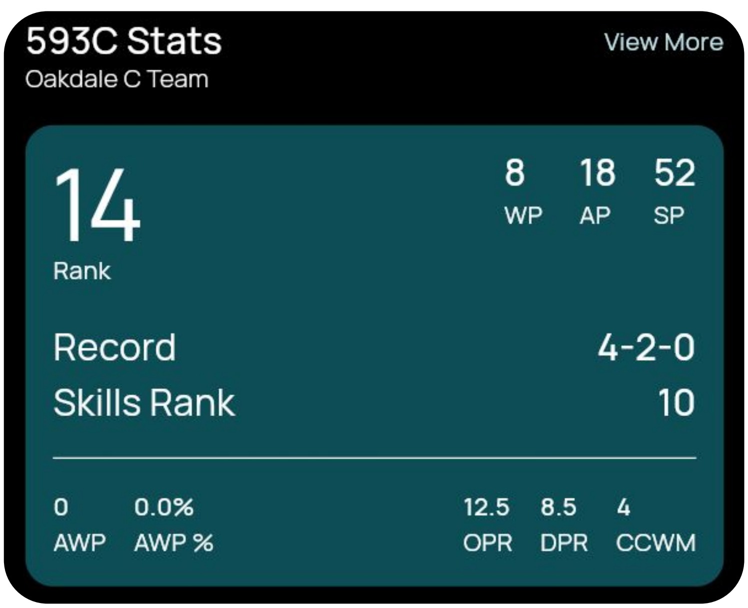
Western MD Valley Stakes Mall High Qualifier

All in all, we didn’t do all that badly at this tourney, for our first competition! There were some things that we could most definitely improve on, but we had a great time!

We ended up ranking 14 overall, with a 4-4-0 record. We ranked 10 in skills, with 36 points in driver and 3 points in auton.

Nothing broke during the competition, which is exciting!

Our autonomous program never fully functioned, so we were never able to earn AWP. This is something we’re looking to improve on in the future. This is due to a few causes, namely being an unreliable clamp and slow conveyor belt during auton.
We suspect our clamp is very unreliable when clamping on corners, and may move our starting point so it clamps onto the side of a goal instead of the corner.



Western MD Valley Stakes Mall High Qualifier

Our clamp was very unreliable and bipolar, where it would work perfectly some matches but not on others. We attribute this to the mogo not being in the exact spot where we needed it to be, so it would not function properly. We hope that by cradling the mogo as it enters will hold it more properly and become more reliable.

In addition, we felt our intake and elevator were slow compared to other teams. We hope that by adding a gear ratio instead of just direct drive from the green motors we can score faster.

The intake was also very flexible and bound up often, as it was not square. The motor for the intake was unprotected and hanging out, so mid way through a match the axle became unsecured.

Finally, we noticed rings getting caught between the stage transfer between the intake and the elevator, so we seek to fix this.

During driver control, there were a lot of small mistakes that cost us matches, such as almost getting DQ'd by not leaving the positive corner fast enough, or dropping a goal by accident. A lot more practice is needed.

1/14/25

- Chloe worked to upgrade the sweeper from plastic to metal, to try to make the arm more reliable
- Chloe added a counterweight to the intake in an attempt to fix problems with it randomly not working, and trimmed the plastic scoops on the bottom to prevent it from stuttering on the foam ground.

1/16/25

- No school due to the end of term.
- Rowan worked on the notebook, and the team reviewed matches from the last competition to try to improve

1/21/25

- Chloe and Ajay started work on a new clamping mechanism to try to improve reliability
- Rowan removed and disassembled the intake, to create a more squared and faster intake.

1/23/25

- Rowan reattached the improved intake, which span much more smoothly. A 2:1 gear ratio was mounted onto the side, with a 5.5w motor powering the intake. The motor is better protected, and is faster.
- Ajay created a large mogo “cup”, which will need to be shrank down

1/28/25

- Rowan replaced one of the missing hooks on the intake
- The motor for the intake was more securely mounted, and

1/30/25

- Rowan replaced the small 2” flexwheels at the top of the conveyor with 3” flexwheels, sinch another team needed them, and we though this might allow for more reliable flippage of the rings onto the goals
- Chloe continued work on the clamp

1/31/25

- Chloe continued work on the more reliable clamp, attaching a plastic “cup” onto the clamp, but was much too big

2/4/25

- Rowan adjusted the elevator in an attempt to speed it up.
- Chloe attached the final clamp “cup”

2/5/25

- Rowan removed all work on the elevator and returned to direct drive; with the sped up intake there was not enough torque to score onto the mogo, the motor heated up much too quickly, and there was a lot of vibration
- Rowan improved the autonomous program, and made it much more reliable.
- Rowan broke the plastic, to be replaced ASAP

2/5/25

Improved auton program!
The program stayed much the same, except for value changes due to the improved auton setup.

```
void autonLeft() {
	rightMotors.set_brake_mode_all(pros::E_MOTOR_BRAKE_HOLD);
	leftMotors.set_brake_mode_all(pros::E_MOTOR_BRAKE_HOLD);
	autonDrive(-700, -700, 50);
	mogoClamp.set_value(true);
	pros::delay(1000);
	conveyor.move_relative(2000, 200);
	pros::delay(1000);
	conveyor.move_relative(20000, 200);
	autonDrive(250, -250, 50);
	// autonDrive(330, -330, 50
	autonDrive(250, 250, 50);
	autonDrive(-600, 600, 50);
	autonDrive(700, 700, 50);
}
```

New program

Old program -------->

```
void autonomous() {
	rightMotors.set_brake_mode(pros::E_MOTOR_BRAKE_HOLD);
	leftMotors.set_brake_mode(pros::E_MOTOR_BRAKE_HOLD);

	autonDrive(-600, -600, 50);

	// rightMotors.move_relative(-1.5,50);
	// leftMotors.move_relative(-1.5,50);

	mogoClamp.set_value(true);

	conveyor.move_relative(2000, 100);
	pros::delay(1000);

	conveyor.move_relative(20000, 100);
	autonDrive(330, -330, 50);

	autonDrive(360, 360, 50);

	autonDrive(-600, 600, 50);

	autonDrive(600, 600, 50);
```

2/5/25

Here are all the functions we call, and what they do:

```
func driverDriver
func driverClamp
func driverSort
func driverHook
```

- During the driver period these first 4 functions define actions the driver can take, driving the robot, activating the clamp and sweeper arm, and the conveyor belt

```
func autonDrive
func autonLeft
func autonRight
```

- These 3 functions are called during the autonomous period, the autonDrive function being called in the left and right programs to control the location of the bot

```
func initialize
func disabled
func competition_initialize
func autonomous
func opcontrol
```

- These final functions are called throughout the match, running each aspect of the program.

  Below is the initialize function, which sets up our bot and displays its location on the screen.

```
// initalize function, runs on program startup
void initialize() {
	pros::lcd::initialize(); // initalize brain screen
	chassis.calibrate(); // calibrate sensors

	// print position to brain
	pros::Task ws1(positionPrint);
}
```

Testing Weight and Speed

We wanted to weigh our robot to compare it to our other teams, and to analyse how much the robot would sink into the floor.

Since we didn't have a big enough scale, we used a Vernier Force Plate to weigh our robot. Using said Force Plate, we found our robot had a weight of about 74 newtons. To convert to kilograms, we divided by 9.81 to find our robot had a mass of about 7.54 kg. Finally, to convert to pounds, we multiplied by 2.205, which gave a final weight of about 16.62 pounds.

We know the foam tiles are about 0.63" (1.6 cm) and are made of Anti-Static EVA Foam, and has a density of about 2.0 lb/ft^3. From here, none of us knew how to calculate the absolute sinkage into the tiles, since we didn't know the spring constant of the tiles.

We were able to measure the robot as sinking about 3/16" into the tiles though.

From a standstill, our robot was able to cross the whole field in just about 2.5 seconds. Taking into account the length of our robot, that results to about 51 in/s. On page 37, we calculated our theoretical max speed was about 56.7 in/s, but that number did not include the time it took to get up to speed. We were shocked at how close the theoretical to actual speeds were!
There was likely some human error in timing and controlling, but we don't believe that would have caused much of a difference.

SHHS Match Strategy

With the recent 3.0 game manual update, positive corner protection has been increased to 30 seconds. We plan to fill a positive corner as quickly as possible and hold it there, so as soon as possible after the 30 sec timer, we can fill a goal with the opposing team's rings and put that into a negative corner. We can not score on wall stake or the alliance stake, so we will see how that turns out.

We also have a goal to work with our alliance to ensure they have a functioning robot, and are able to score. Talking strategy beforehand is very important!

Skills will be a large part of this competition for us, as we see it as the only way to make it to states. We are not optimistic about earning any awards, but believe if we do well enough in skills again we may make it to states. Since there is not enough time at home, we may have to program our skills route while at the competition. That will not be fun.

If we do better than at valley mall, we'll be happy! (10 in skills, 14th overall, lost in the Round of 16)

Let's do this!

Rowan W.

Chloe V.

Karunya G.

Ajay V.

Engineering Notebook Rubric

Team # ____________ **Grade Level ☐ ES | ☐ MS | ☐ HS | ☐ University** **Judge Name** ____________________

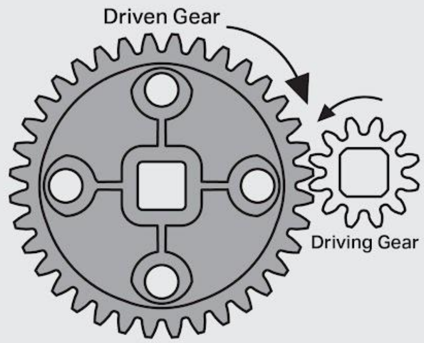
Directions: Determine the point value that best characterizes the content of the Engineering Notebook for that criterion. Write that value in the column to the right. This rubric is to be used for all Engineering Notebooks regardless of format (physical or digital).

CRITERIA	PROFICIENCY LEVEL			
ENGINEERING DESIGN PROCESS	**EXPERT** (4-5 POINTS)	**PROFICIENT** (2-3 POINTS)	**EMERGING** (0-1 POINTS)	**POINTS**
IDENTIFY THE PROBLEM	Identifies the game and robot design challenges in detail at the start of each design process cycle with words and pictures. States the goals for accomplishing the challenge.	Identifies the challenge at the start of each design cycle. Lacking details in words, pictures, or goals.	Does not identify the challenge at the start of each design cycle.	
BRAINSTORM, DIAGRAM, OR PROTOTYPE SOLUTIONS	Lists three or more possible solutions to the challenge with labeled diagrams. Citations provided for ideas that came from outside sources such as online videos or other teams.	Lists one or two possible solutions to the challenge. Citations provided for ideas that came from outside sources.	Does not list any solutions to the challenge.	
SELECT BEST SOLUTION AND PLAN	Explains why the solution was selected through testing and/or a decision matrix. Fully describes the plan to implement the solution.	Explains why the solution was selected. Mentions the plan.	Does not explain any plan or why the solution or plan was selected.	
BUILD AND PROGRAM THE SOLUTION	Records the steps to build and program the solution. Includes enough detail that the reader can follow the logic used by the team to develop their robot design, as well as recreate the robot design from the documentation.	Records the key steps to build and program the solution. Lacks sufficient detail for the reader to follow the design process.	Does not record the key steps to build and program the solution.	
TEST SOLUTION	Records all the steps to test the solution, including test results.	Records the key steps to test the solution.	Does not record steps to test the solution.	
REPEAT DESIGN PROCESS	Shows that the design process is repeated multiple times to improve performance on a design goal, or robot/game performance.	Design process is not often repeated for design goals or robot/game performance.	Does not show that the design process is repeated.	
INDEPENDENT INQUIRY	Team shows evidence of independent inquiry from the beginning stages of their design process. Notebook documents whether the implemented ideas have their origin with students on the team, or if students found inspiration elsewhere.	Team shows evidence of independent inquiry for some elements of their design process. Ideas and information from outside the team are documented.	Team shows little to no evidence of independent inquiry in their design process. Ideas from outside the team are not properly credited	
USEABILITY AND COMPLETENESS	Records the entire design and development process in such clarity and detail that the reader could recreate the project's history.	Records the design and development process completely but lacks sufficient detail.	Lacks sufficient detail to understand the design process.	
RECORD OF TEAM AND PROJECT MANAGEMENT	Provides a complete record of team and project assignments; team meeting notes including goals, decisions, and building/programming accomplishments; design cycles are easily identified. Resource constraints including time and materials are noted throughout.	Records most of the information listed at the left. Level of detail is inconsistent, or some aspects are missing.	Does not record most of the information listed at the left. Not organized.	
NOTEBOOK FORMAT	Five (5) points if the notebook has evidence that documentation was done in sequence with the design process. This can take the form of dated entries with the names of contributing students included and an overall system of organization. For example, numbered pages and a table of contents with entries organized for future reference. Partial points may be awarded if this is inconsistent or incomplete.		ZERO POINTS (DOES NOT MEET CRITERIA) If awarding zero points, please include details in the "NOTES" area below	
NOTES:				**TOTAL POINTS**

All judging materials are strictly confidential. They are not shared beyond the Judges and Judge Advisor and shall be destroyed at the end of the event.

Gear Formulas

$$\textbf{Gear Ratio} = \frac{\text{\# of } \textbf{Driven Gear Teeth (Output)}}{\text{\# of } \textbf{Driving Gear Teeth (Input)}}$$



Power Transfer is a 1:1 gear ratio where the driving and driven gear have the **same number** of teeth.

Increasing Torque (lowering speed) is a gear ratio where the driving gear has **fewer teeth** than the driven gear.

Increasing Speed (lowering torque) is a gear ratio where the driving gear has **more teeth** than the driven gear.

Compound Gear Ratio = (Gear Ratio 1) × (Gear Ratio 2) × (...)

Motion Formulas

$$\textbf{Average Speed} = \frac{\text{Total Distance}}{\text{Total Time}}$$

Distance is from the **axis of rotation**

$$\textbf{Rotational Speed} = \frac{\text{\# of Turns}}{\text{Time}} = \frac{\text{Degrees}}{\text{Time}}$$

Circumference = π × Diameter

Power = Force × Velocity

π ≈ 3.14

Torque = Force × Distance

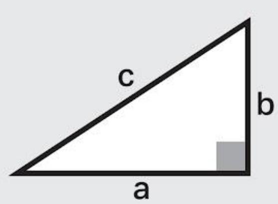
Force = Mass × Acceleration

Mathematical Formulas

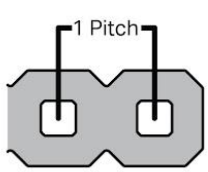
Complimentary angles are angles that sum to 90°

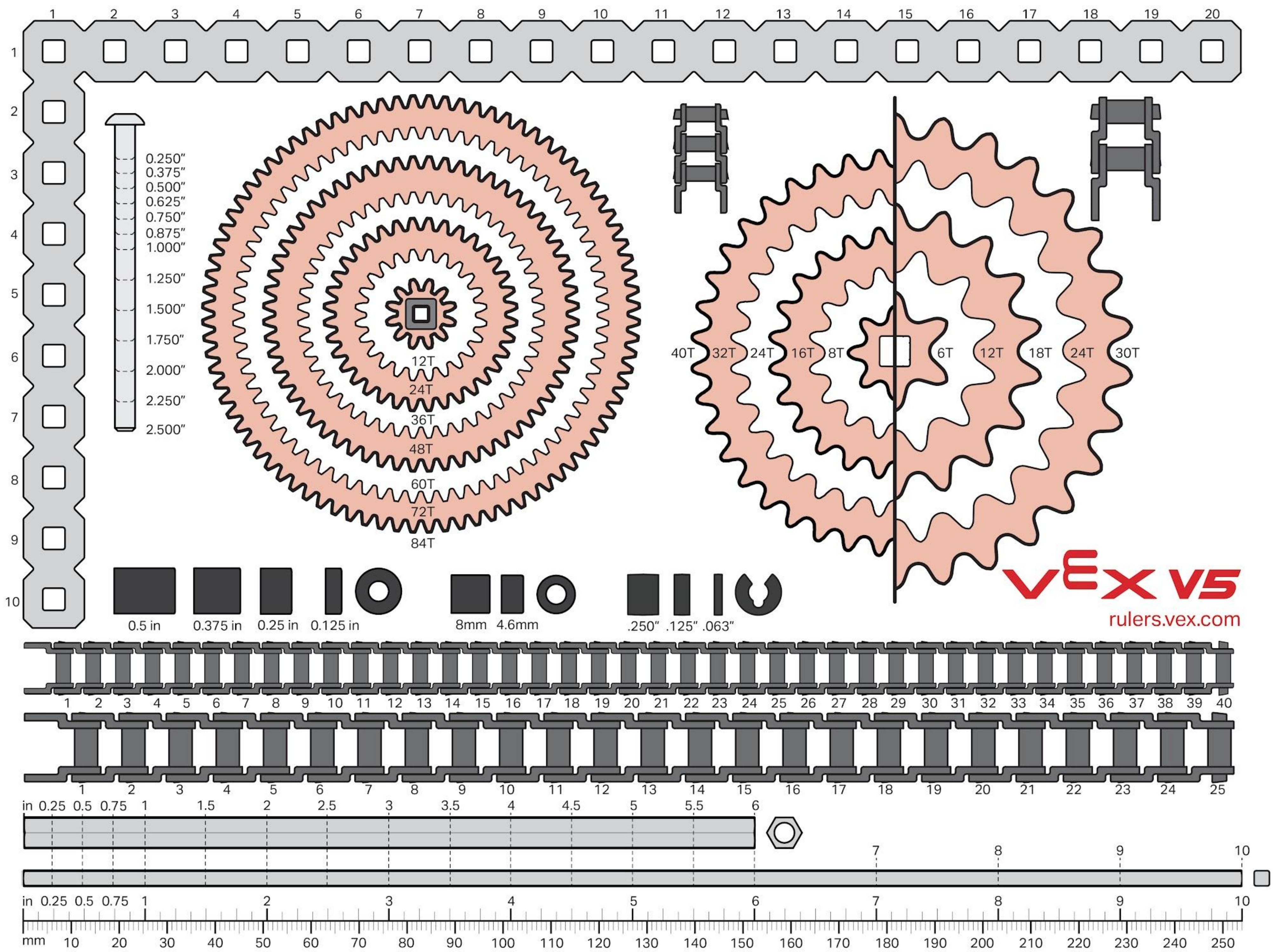
Supplementary angles are angles that sum to 180°

Pythagorean Theorem: $c^2 = a^2 + b^2$



1 Pitch = 0.5 in =12.7 mm




0.250"
0.375"
0.500"
0.625"
0.750"
0.875"
1.000"
1.250"
1.500"
1.750"
2.000"
2.250"
2.500"
12T
24T
36T
48T
60T
72T
84T
40T 32T 24T 16T 8T 6T 12T 18T 24T 30T
0.5 in
0.375 in
0.25 in
0.125 in
8mm
4.6mm
.250"
.125"
.063"
VEX V5
rulers.vex.com
in
mm

elapse

Easy to navigate and
find what you're looking for

The Smart V5RC app

download now
elapseapp.com

V5RC Team 593C
Oakdale High School
2024-2025, "High Stakes"

Rowan W. Karunya G
Chloe V. Ajay V

Oakdale 593C Engineering Notebook Template

- Welcome to the Oakdale 593C Engineering Notebook!
- This template was designed to be easy to use for all members of 593C, and to be easily readable.
- The goal of this notebook template was to create an easy way of logging data, yet looks more unique than the notebooks provided by VEX.
- Another goal was to prevent walls of text that you need to pull a magnifying glass out to view, much like previous years notebooks.
- Design philosophy is to integrate colors and images to break up walls of text to make it easier to read
- Writer names and entry dates will be visible on every page to demonstrate chronological documentation!