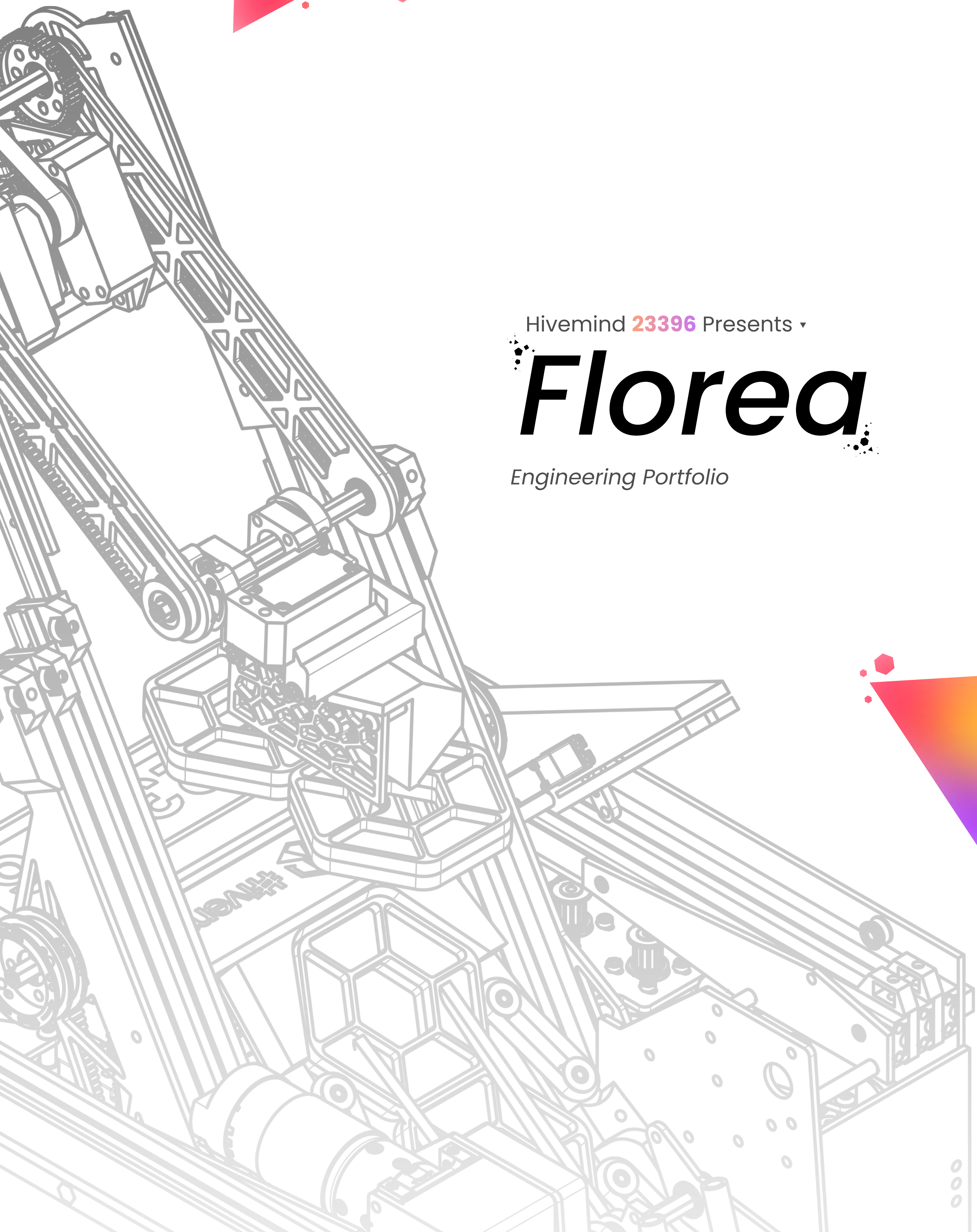
Hivemind 23396 Presents

Florea

Engineering Portfolio

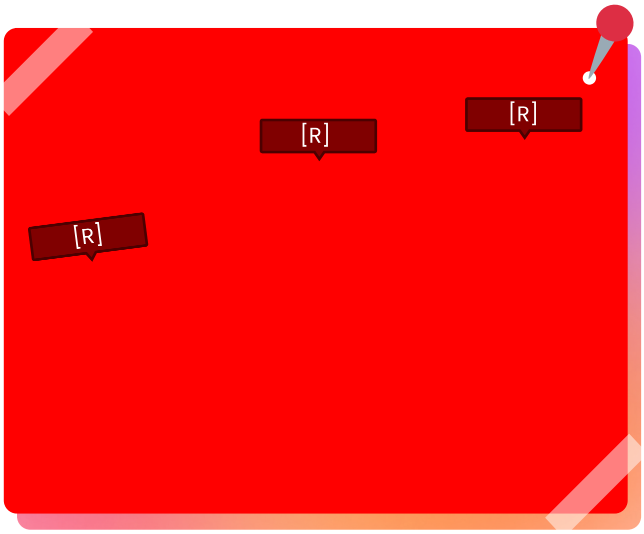


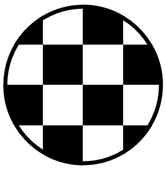
Please Note: This portfolio is a public release copy, some pictures have been removed and replaced with solid red.

Team Introduction

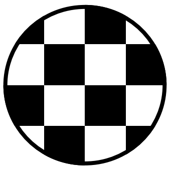
We are FTC Rookie Team #23396, Hivemind, from Spring Hill, Tennessee. We're a student-led team focused on creating quality, competitive robots for FIRST Tech Challenge, whilst leading initiatives to spread STEM into **technical and underrepresented communities**.

Whilst our team may be rookie, 2 of us have been on robotics teams prior, so we have experience and know how to get things done.

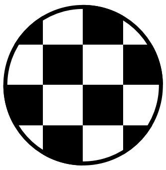




[REDACTED] - Captain
CAD Design, Programming, Business



[REDACTED]
Outreach, Assembly, Business



[REDACTED]
Outreach, Assembly, Business

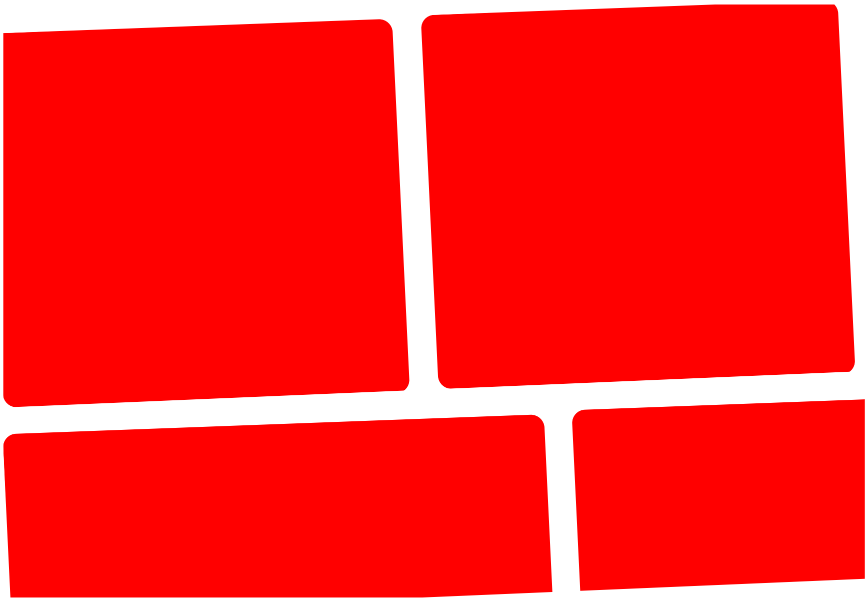
Team Management

Our team is **fully student-led**, which means that meetings are scheduled and ran by team leaders. We have a **captain** who oversees the team and its agenda, as well as **leads** for different divisions of the team.

Our coaches are there to manage bank accounts, make sure we get to competitions, and communicate to parents!

Recruiting and Sustainability

We are a small team, with reliable funding, and a dedicated workspace. Our goal is to continue to support this small team, whilst recruiting mentors for team growth, and contributing to FIRST knowledge for long-term information permanence within our team environment.

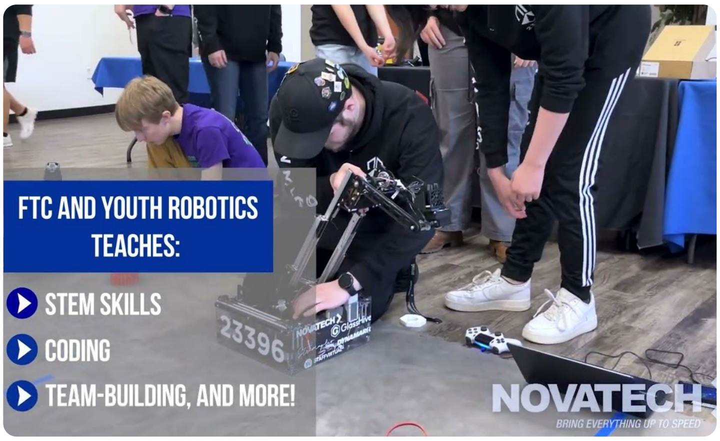


Getting Sponsors & Mentors

We created a **sponsorship packet** to send to potential sponsors over **emails** or other social networking platforms, like **LinkedIn**. We typically send these sponsorship messages to companies to which we already have preexisting connections and then network from there!

Our team has implemented this incredibly effective strategy since the early summer of last year, leading us to secure **13 sponsors so far**, and this number can get even larger.

We use a similar philosophy for **finding mentors**, by finding them through either pre-existing **industry connections**, or finding them **through online communities**.



NOVATECH

GlassHive

MarketMuse

workflow

Stramaglio CONSULTING

DYNAMARK printing signs marketing fulfillment promo

THE CANNATA REPORT

Nexsales

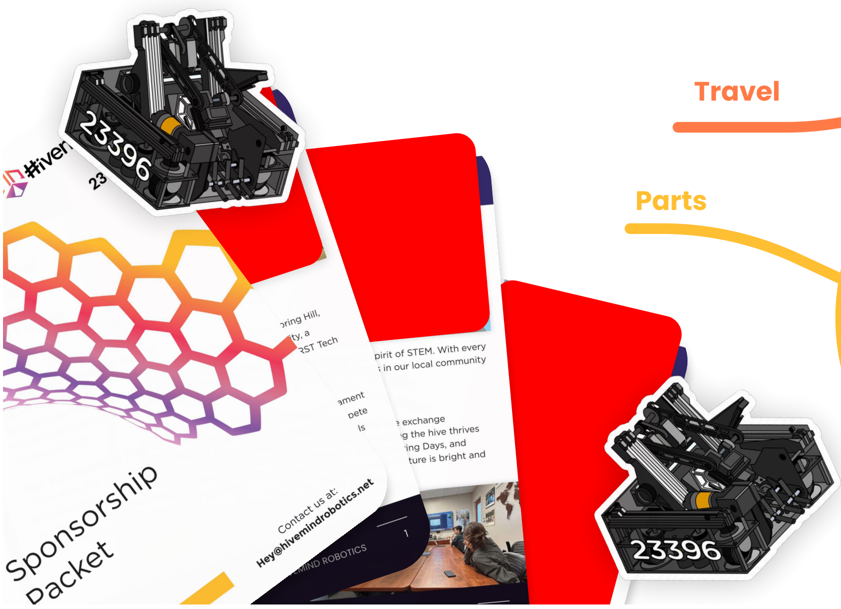
MARATHON CONSULTING

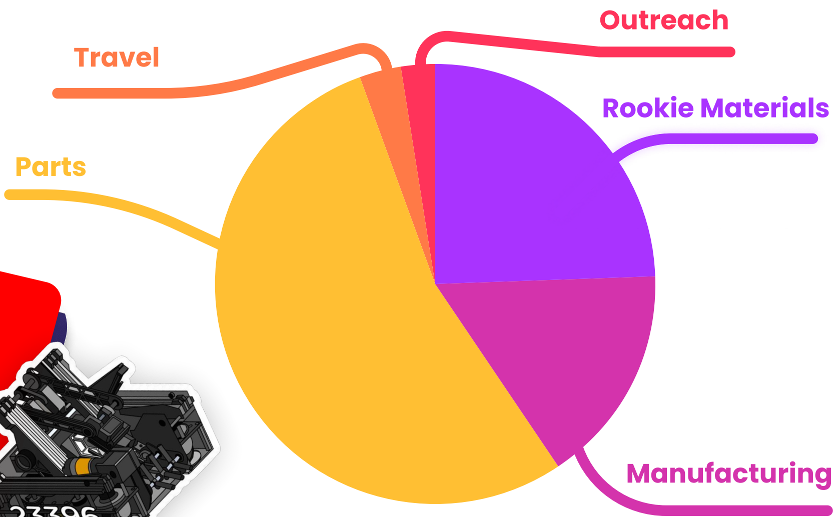
HAAS Gene Haas Foundation

SV STAFFVIRTUAL

ACDI

Jpixx





We recognize that assistance is a two-way street. While we prioritize mentoring other teams and students, we acknowledge the essential role of our mentors in providing support and expertise. These mentors serve as the foundation for **maintaining and developing** skills, playing a crucial role in sustaining both **technical** and **business knowledge**. They guide us in their respective fields, help build our skills, and foster critical thinking.

John Bateman

Engineering Mentor

Teaches: Mechanical, Electronics

Gained through Community

Davy Hallihan

Business & Engineering Mentor

Teaches: Mechanical, Business

Met at a FIRST Event

Jamie Kazemier

Engineering Mentor

Teaches: Mechanical, Design

Gained through Discord

Mason Stuart

Software Mentor

Teaches: Software, Control Theory

Gained through Discord

Industry Mentors

Industry Mentors are mentors who can only help with certain issues that we run into! Here are some of the stories of help we've received from mentors at different companies, we typically connect with industry mentors through **online communities**.

Google

A member of the **AI Engineering** team at **Google Cloud** helps us with tasks such as **Computer Vision** and **Object Oriented Programming** in Java.

GitHub

A **Product Manager** at **GitHub** advised us on how to get started with **April Tags** and **Mono-repositories**, they also have done code review for this year's robot!

AXON

The Founder of **Axon Robotics** actively reviews and advises us with **CAD Design**, programming, and how to design **sustainable, modular subsystems**.

iRobot

A **mechanical engineer** at iRobot taught us how a proper design process should go, and advised us to focus on **quality and robustness** with machining.

Microsoft

A **Project Manager** at Microsoft taught us about the importance of **Project Management** and using programs like **Trello** and Linear.

Utilizing our Public Resources program, we actively enhance the ever-growing knowledge base of FIRST by **offering valuable assets** in specialized fields like **graphic design**, **programming**, and **team management**.

Outreach Program Summary

Impact: **57,900+ Page Views**

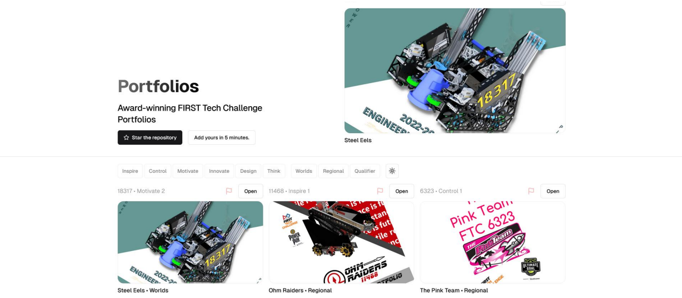
Hours: **200+**

Portfolios

11,500+ page views

Portfolios is a **open-source** resource for **FTC and FRC Teams** providing them resources to create award-winning e**ngineering documentation**. Teams which have won awards are able to submit their portfolios or provide documentation that has brought them success.

With it being open-source, means that anyone can view the source code behind it, and learn.

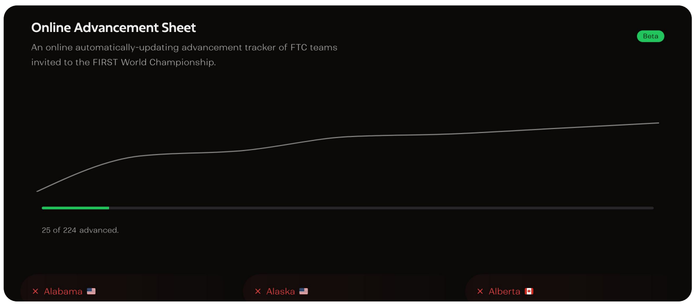


Worlds Advancement Sheet

44,700+ page views

Together alongside **#16461 Infinite Turtles** in **North Carolina**, we developed an automatically updating Worlds Advancement Sheet.

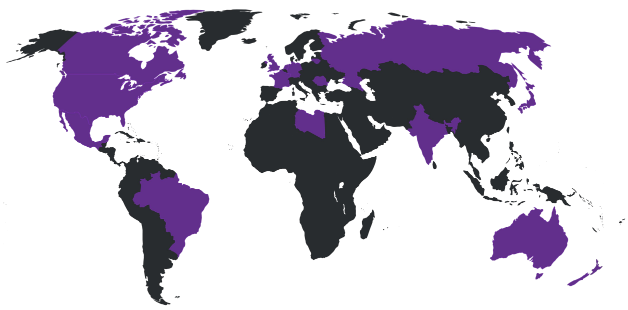
It pulls teams from FTC's Elastic Search and FTC-Events API to show teams, matches, streams, and more. It automatically updates through **real-time server infrastructur**e we **developed and deployed**.



Educational Blog Posts

1,700+ views

We pride ourselves on creating high-quality designs for both robotics, websites, and portfolios. To assist with teaching people new skills, we **create blog posts** which explain our design process for specific things, big ones being graphic design, web development, and robot design.



Reached users in **16** countries

Mentoring & Assisting

4 Teams

We adhere to the FIRST guidelines for Mentoring for teams **FTC 18221** from Trumbull, Connecticut, and **FRC Team 3597** from Kittery, Maine. We also assist **FTC 7444** from Winston-Salem, North Carolina, and **FTC 7842** Browncoats from Huntsville, Alabama.

We have letters from these teams available on request, confirming details throughout our relationship and assistance throughout the season.

Lipscomb University

Active Relationship

We actively network with universities in our community, one prominent one being Lipscomb, they allowed us to utilize their CNC mills, we've done robot demonstrations with their **VEX-U** and **BEST** Robotics teams.

FTC Discord

20,000 Members

We actively engage in productive conversations with other teams inside the FIRST Tech Challenge Discord Server. We are a part of a student body which helps **run the server**. We help ensure that the community is a **safe and welcoming** place for all members to develop their skills.

- **771,922 Messages** Since Kickoff
- **20,900** Members
- **5,000** Weekly Visitors

Girls in STEM

10 Troops

We actively try to engage our next generation of **girl scouts to become engaged inside FIRST Robotics**. We actively invite Girl Scout troops to attend inside our meetings, and learn life-long skills like CAD, Engineering, and 3D Printing.

With this, we hope to increase their interest in joining a robotics team in the future, and also teach them life-long skills, whilst helping them earn their robotics badges in the process.

Season Begins

Robot Demonstrations Begin

We began doing robot demonstrations **to spread STEM and FIRST awareness** in December, with our Beta Bot, continuing for the **rest of our season** leading up to our State Competition

Girl Scouts' STEM Badges

We ran robot demonstrations and **taught 10 local Girl Scout troops about FIRST**, CAD, and software design.

Scrimmage

We ran a scrimmage in Nashville in February, with the assistance of Novatech for the facility.

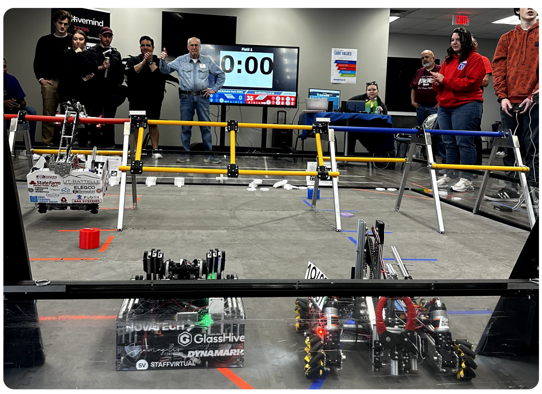
Now we're here

Scrimmage

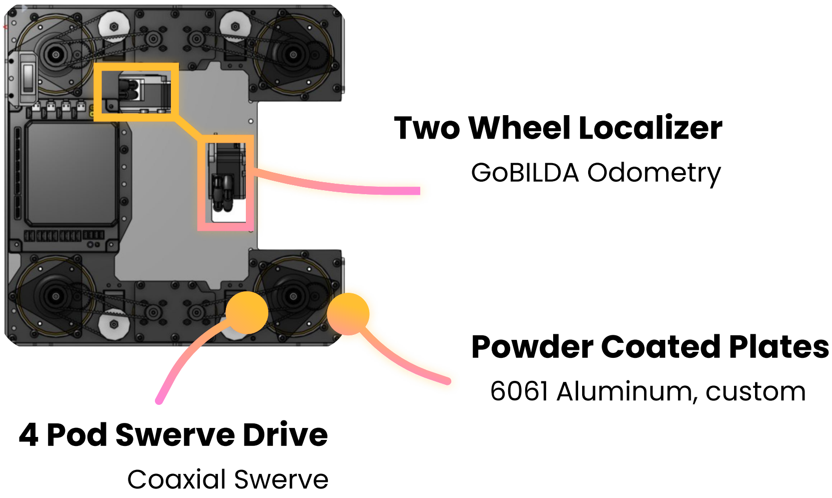
In collaboration with NOVATECH

Along with the help with facilities from our gracious sponsor, **Novatech**. We **ran, streamed, and managed**, our own **scrimmage**, in their Nashville facility.

We had **8 teams attend**, making it one of the **largest scrimmages ran in Tennessee this year**!



Drivetrain

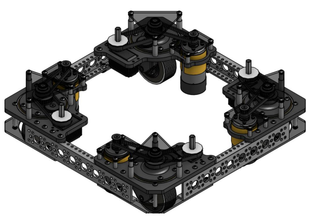


Custom Manufacturing

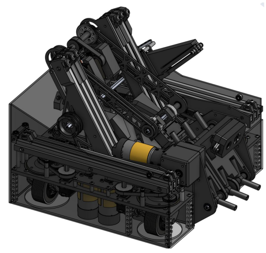
This year, we decided to fully custom manufacture our robot. We did this through industry leading platforms such as **Fabworks** and **Send Cut Send**. Materials such as **Lexan**, **aluminum**, and **carbon fiber** keep our robot strong while staying light.

fabworks. + SendCutSend

Iterations



V1 Swerve Drivetrain
Proof of concept Drivetrain, Made over the summer.



V2: Centerstage Robot
Cleaner design, rigidity, full assembly.

Lessons Learned

Through extensive **design reviews** with **mentors**, we determined that we need more routes for cable management, and that in the future, we should focus on making our robot a bit bigger to allow for better clearances for **motors, servos, and wires**.

Swerve Drivetrain

Swerve is a custom-built drivetrain that features holonomic motion with the benefits of full-power in **every conceivable vector direction.** This is in direct contrast with mecanum drivetrains, where strafing does not provide full power from the motor. Because there is no slipping from mecanum rollers, swerve is capable of **faster acceleration.**

Active Intake

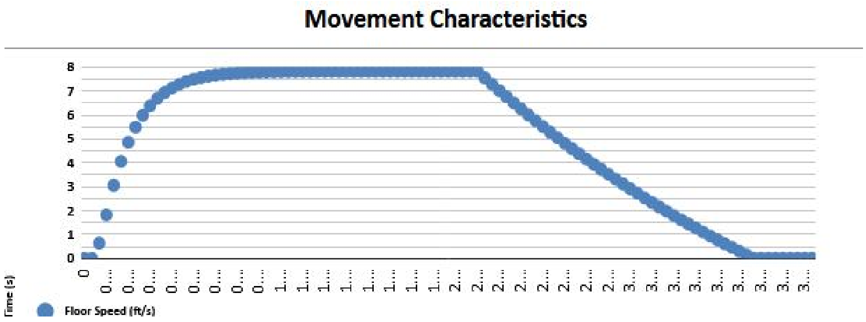
Our active intake is capable of **passively adapting to the height of the stack**, while also holding pixels **through sprung flipper arms**. We are also capable of picking up from the ground, making our intake truly dynamic.

Deposit & Claw

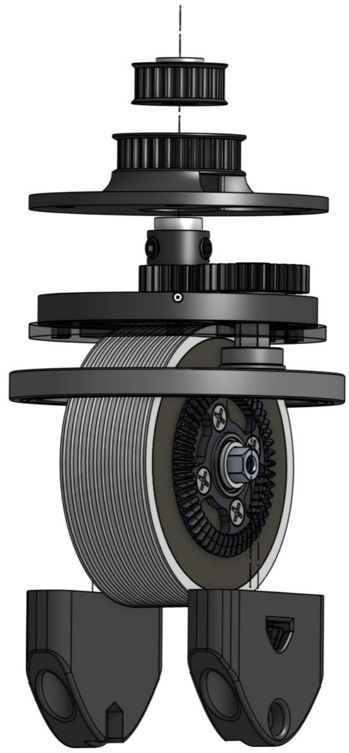
Our deposit contains a **virtual four bar** where the pixels maintain their **relative angle** from the intake, while also rotating outwards to be scored. Our claw is capable of pivoting **360 degrees** to score and create mosaics. This gives us the flexibility to score anywhere on the backdrop, and also allows us to edit the rotation of pixels on the board.

Design Goals

Many robots in years past have attempted to create a swerve drive, whether it be coaxial, differential, or swomni. But they always fall into the same pitfalls of not having enough design rigidity. This year, we made **no compromises,** we researched issues, and improved almost every aspect of modern swerves in FTC.



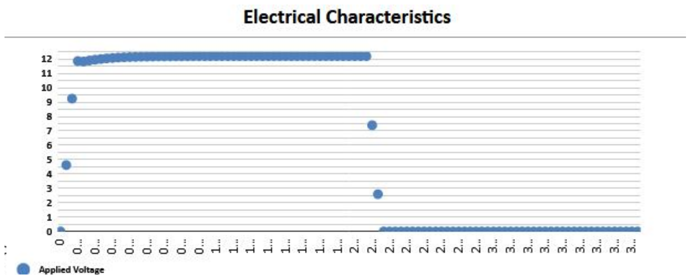
ILite FRC Drivetrain Simulation



Electrical Simulations

Due to far away wings, optimizing the efficiency of our drivetrain was a **huge** focus of ours during this year's season.

We discovered that because our wheel radius is smaller than conventional swerves (**64mm** vs **72mm**), we were able to reach similar speeds with a smaller reduction (**8:1 vs 10:1**). This helps us keep our modules small, while keeping the same performance.



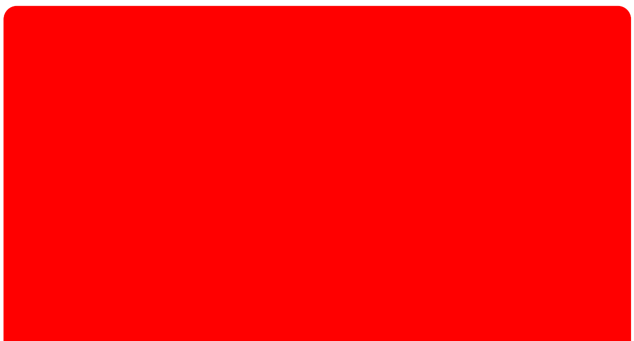
ILITE Drive Ratio Comparison

8:1	vs	10:1
• 2s Cycles • Less Power, Same Torque		• 3s Cycles • More Power

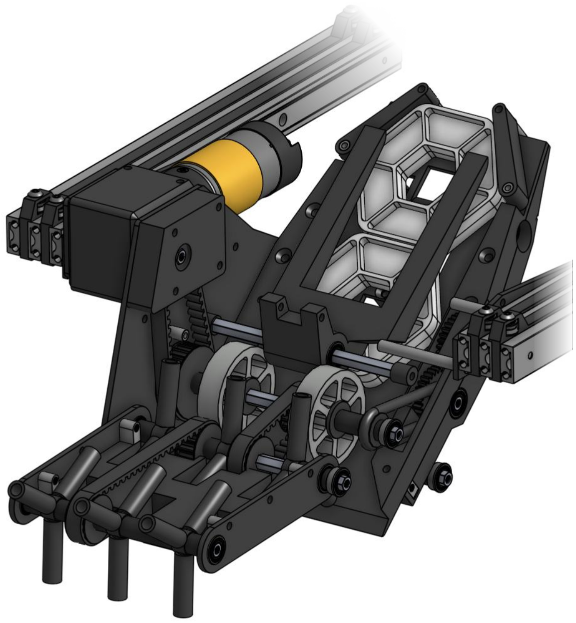
Why Swerve?

At the start of the season, we already knew that we wanted to do something **"out of the box"** and **innovative**; Swerve is effective in both categories due to its **hardware and software** challenges.

However, it has rewards, mainly **faster acceleration** and **more traction**.



[R] and [R] assemble a Swerve Pod



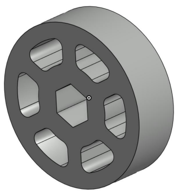
Passive Transfer

Through the use of **sprung flipper arms** and many iterations prior to our final design, we found that the best way to create a transfer was to do it **passively**.

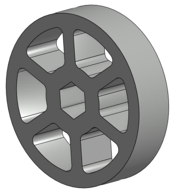
These flipper arms keep the pixels in place and oriented correctly while we await our deposit's **Virtual Four Bar** to pick up the pixels and orient them on the backdrop.

Conveyor Belt

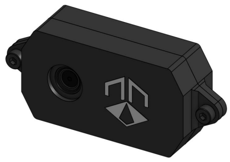
A round belt conveyor helps the pixels move all the way up to the top.



V1 - Small Wheel
Causes jams, too stiff.



V2 - Bigger Wheel
More Compliant

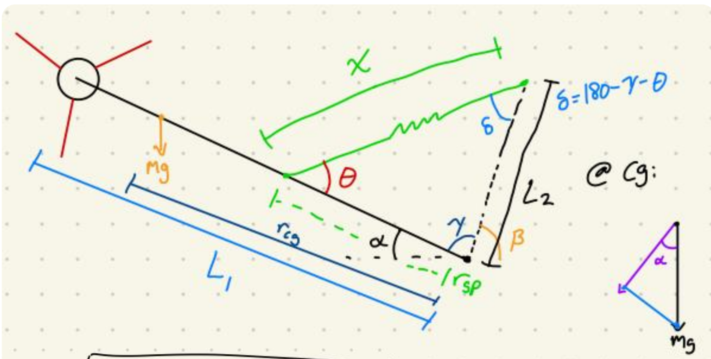


Camera is mounted to the intake for ease of use in Autonomous

Math & Theory

Our theory was that the spring counteracts the **gravitational pull against our intake**; however, with enough torque, the motor can pull down the spring thanks to the **coaxial effect**.

This allows our intake to **adapt to the height of the stack or ground passively** because it will stop itself when it reaches a solid, which blocks it from moving further until the pixel is inside the intake, or out of the way from the intake.



$$x = \sqrt{r_{sp}^2 + L_2^2 - 2 r_{sp} L_2 \cos\gamma}$$

$$r_{sp}\, k\, x \sin\theta = r_{cg}\, mg \cos\alpha$$

$$\frac{\sin\theta}{L_2} = \frac{\sin\gamma}{x}$$

$$r_{sp} k \cancel{x} \frac{L_2}{\cancel{x}} \sin\gamma = r_{cg}\, mg \cos\alpha$$

$$\theta = \sin^{-1}\left(\frac{L_2}{x}\sin\gamma\right)$$

$$\sin\theta = \frac{L_2}{x}\sin\gamma$$

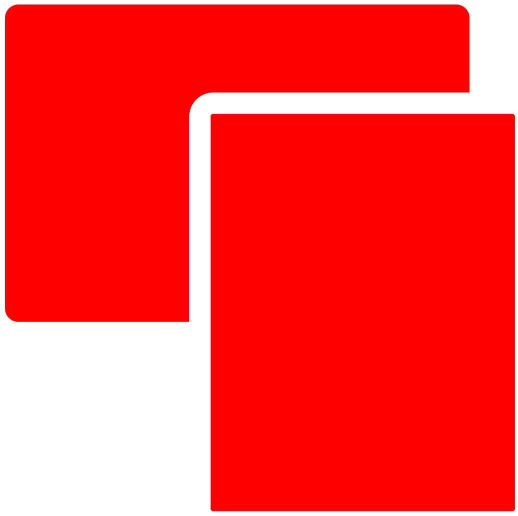
$$\boxed{k = \frac{r_{cg}\, mg \cos\alpha}{r_{sp} L_2 \sin(180-\alpha-\beta)}}$$

Why Linear Slides?

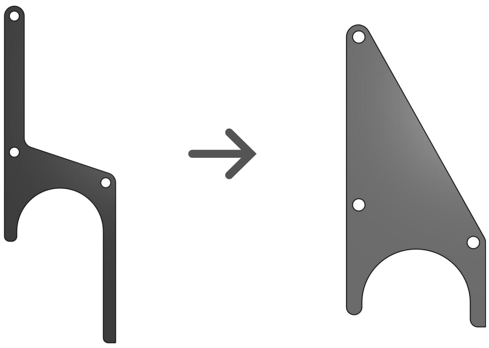
We chose **linear slides** over alternate options like **double reverse four bars** for several reasons. however, as in other team's past seasons, the key reasons remain their superior **durability and speed**.

Why climb from your slides, though?

This removes another aspect of the robot we would otherwise have needed to design separately. Further, the rated load capacity of our linear slides far exceeds the design weight of our robot, ensuring reliability even under rapid hangs



V1 | **V2**

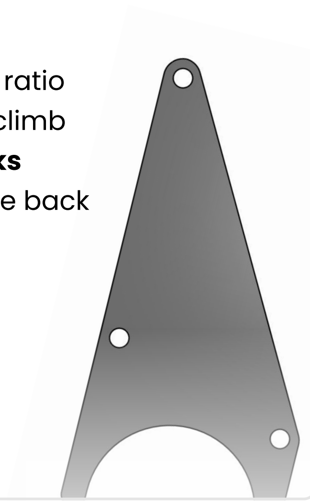


3D Printed, Flexes and Warps

Carbon Fiber, Stiff and Rigid

Climbing

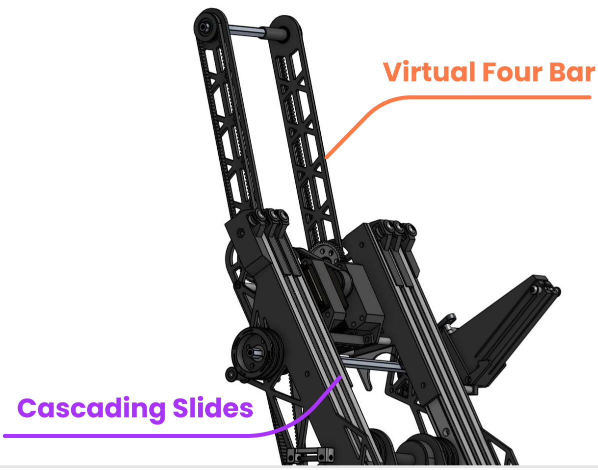
We utilize our high gear ratio to our advantage, and climb using **carbon fiber hooks** which are located on the back of our **virtual four bar assembly**.



Linear Slides

Our custom cascading linear slides allow us to deposit quickly and efficiently in combination with our **virtual four bar**.

They run at a **15.2:1 ratio**, making them fast enough to deposit, yet strong enough to hold up our robot when they are used for **climbing during endgame**.

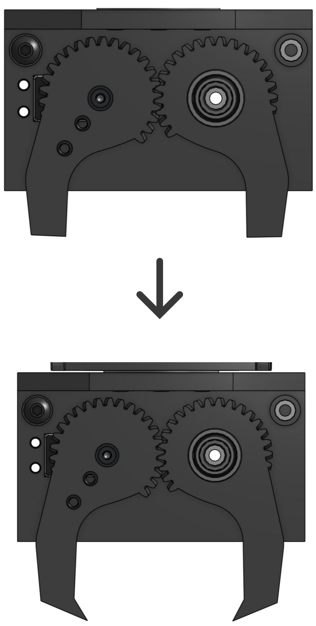


Claw Overview

Our claw can pivot up to 180 degrees, which lets us create mosaics quickly and efficiently. We can also change the pitch of our claw to align with the backdrop or intake more efficiently.

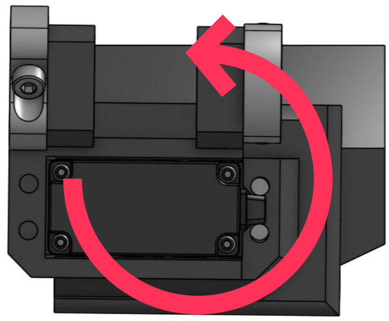
Drone Launcher Overview

Our drone launcher is a cost-effective and reliable device that utilizes latex tubing to quickly and efficiently propel the drone when kinetic energy is released.



V1
Short fingers,
fails transfers.

V2
Longer fingers,
+ internal claw.



Claw Rotation

Allows us to rotate our claw in 30 degree increments to create mosaics

Drone Launcher

Our drone launcher utilizes an micro servo, which releases the pin and shoots the drone, usually allowing it to land consistently in zone one or two.

Loop Time Optimizations

Bulk Reading

We made it a prominent goal of ours **only to have to bulk read one of our hubs, the Control Hub**, as bulk reading both hubs is redundant and adds a lot of **delay between loops**.

IMU Threading

Threading the IMU is essential as it **shaves off ~2ms** from our loop times; considering that we run field-centric and swerve, we need our **IMU reads to be as fast as possible**.

Power Cache

We cache our motor & servo powers to ensure that we don't end up:

A: Stalling our motors
B: Pulling extra voltage and sending extra reads/writes

The code to make this happen is pretty simple:

```
if (power != lastPower) {
 motor.setPower(power);
}
```

Command Parallelization

We utilize an open source optimization engine, **Photon**, by an FTC Alumni, which optimizes the reads and writes of motors by utilizing Parallel Commands, which allows us to send a maximum of **8 commands per loop** that do more than one thing at once.

Photon also helps with other things in FTC. However, this is our primary use case for it.

Photon and the rest of **our optimizations almost doubled our loop-times**.

Loop Order of Operations

We have a specific order of operations to ensure we only **write, read, and update once per loop**. Our robot utilizes a **custom hardware class** that handles all the logic for us, making this super simple to implement in most cases.

We run our loops 300 times per second.

```
robot.read();
robot.loop();
robot.write();
robot.clearBulkCache();
```

Overview

We utilize 3 **calculus based** PID (Proportional, Integral, Derivative) controllers for our **pathfollower**.

We utilize a custom **Lookahead Algorithm** which uses the robot's position and compares it to a target position, then we smooth the path out.

Due to our **command based architecture**, we must wait until we reach our target position within a certain threshold, before we are able to perform other tasks, such as depositing our purple or yellow pixels.

2 Wheel Odometry

We utilize two GoBILDA odometry pods and the Control Hub's **Internal Measurement Unit** to get our robot's current position

Lookahead Algorithm

Our custom lookahead algorithm allows us to predict the path that we are going to be using compared to our current localized position.

First we subtract our current position from the target position.

```
return target.subtract(robotPose);
```

Then we use 3 PID Controllers to chase that target position. Once we reach the position we mark the task as completed, allowing us to do more commands in autonomous.

```
Math.hypot(error.x, error.y) > max;
```

Math & Theory

Our algorithm utilizes linear algebra in order to quickly and efficiently follow specified paths with less than a few centimeters of error.

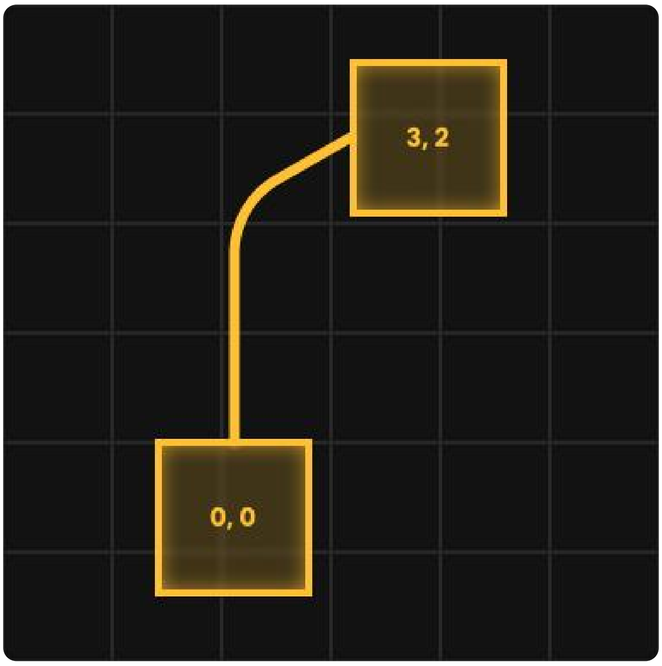
(not taking into account odometry drift)

We first take into account the robot's current position and subtract it from our target position, as can be seen in the example methods also on this page.

Then we calculate the difference and **smooth out our trajectory.**

Then we construct a pose, set the drivetrain's power to go to the target pose and use our 3 PID controllers to check for error, if error is minimal, we mark the task as **completed**.

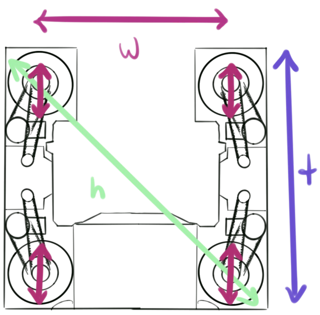
This movement algorithm is quick, efficient, and doesn't suffer under low loop-times or voltage due to extensive tuning.



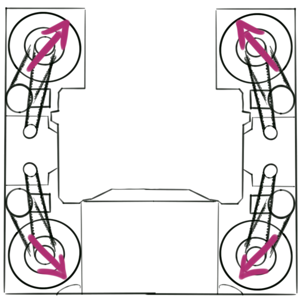
To achieve holonomic motion, Swerve equipped with 4 independently-controlled pods, enables precise and versatile movements in any direction. This translates to a **125% boost in directional speed** and a **150% boost in rotational speed** compared to traditional mecanum wheels, offering unparalleled maneuverability and agility in tight spaces.

$$\text{Output} = K_p \cdot e(t) + K_1 \int_0^t e(t)dt + K_d \cdot \frac{de(t)}{dt}$$

Forward/Backward



Rotating



PIDF Control

To control each pod, a continuous rotation servo is powered by a custom **PIDF Controller**, utilizing the built in Analog Encoder on the **Axon Mini+** Allows us to easily get the rotation of each pod. A PIDF Controller allows the pods to **rotate as quickly** as possible to a **given rotation**.

Motor Flipping

Due to the **symmetry** of each pod's motion, a pod can travel to the *opposite pod angle and reverse the wheel* allowing for the quickest and most efficient rotation possible.

Second Order Kinematics

This drivetrain utilizes **Second Order Swerve kinematics** which **minimizes skew** while controlling the drivetrain.

Desired Motion

F = Forward
S = Strafe
R = Rotate

wA = Pod Angle
wS = Wheel Speed

h = sqrt(w^2 + t^2)

Step 1

A = S - R * W/H

B = S + R * W/H

C = F + R * T/H

D = F - R * T/H

Step 2

wA_1 = atan2(B, C)

wA_2 = atan2(B, D)

wA_3 = atan2(A, D)

wA_4 = atan2(A, C)

Step 3

wS[1] = sqrt(B^2 + C^2)

wS[2] = sqrt(B^2 + D^2)

wS[3] = sqrt(A^2 + D^2)

wS[4] = sqrt(A^2 + C^2)

What is a PID Controller?

A PID Controller utilizes calculus and complex control theory. You provide the controller with three components:

- The **Proportional**: Controls how the subsystem will react to change
- The **Integral**: Considers how long it has been since the last change
- The **Derivative**: Acts as a dampener to your **Proportional** and **Integral** components.

Where do we use them?

In our case, we use PID controllers whenever we can. Currently, we run **over 9** PID controllers, all for different tasks.

A couple examples are:

- Our Swerve's rotation controller
- Our Deposit arm's lift controller
- Our Path Follower
- Our **heading lock** in driver control
- Our **field centric**

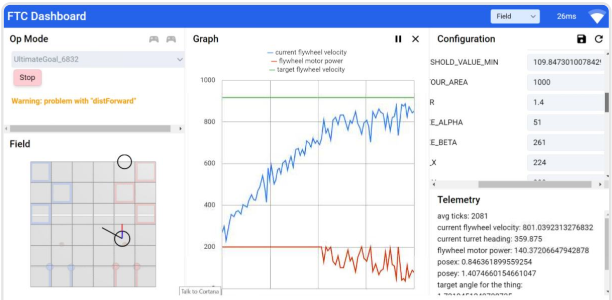
How do you tune one?

We utilize **FTCDashboard** to tune our PIDs as it allows us to easily graph and edit our components in real-time.

We typically start off with **super small numbers**, and then raise these numbers until we see a reaction.

Then we continue increasing until we receive the behavior we expect.

If there is behavior that is unexpected, we usually try to **raise the derivative value**, or **increase** our **feedforward** value and **decrease proportional**.



A map of our PID implementation