2876 DevilBotz 2026
Technical Binder



Table of Contents

Initial Game Analysis **2**

Strategy 2

Research and Design 4

Subsystems **6**

Drivetrain 7

Intake 8

Hopper 10

Shooter 13

Programming **17**

Controls 17

Vision 18

Autonomous 22

Initial Game Analysis

REBUILT, the FRC game of the 2026 season, focuses on two main goals: shooting spherical FUEL into a HUB during the active periods, and climbing the three levels of the TOWER. ROBOTS are able to go through a TRENCH and over a BUMP to reach different sources of FUEL. Defense, hoarding, and passing are also viable options to help win matches.

Strategy

Kickoff Day - team members split into two groups that worked in tandem:

Rules Group:

- Scoured the game manual for relevant rules
- Presented streamlined findings to the whole team by category (ex. robot construction, field zones, fouls, etc.)
- Researched & addressed the strategy team's specific questions in real time

Strategy Group:

- Rated possible robot actions by value and difficulty
- Discussed tradeoffs (ex. trench vs. bump)
- Developed potential robot archetypes and rated them by both predicted value and execution difficulty

Research

	Mobility	points	difficulty (/10)	Arena Interaction	points	difficulty (/10)	Game Piece Interaction	points	difficulty (/10)	Robot to Robot Interaction	points	difficulty (/10)
	Bump	N/A	2	Give fuel to human player	N/A		score fuel (in active hub)	1	6	~~Block opposition from scoring from neutral zone~~	N/A	
	Trenches	N/A	5	Tower L1 (2 robot limit)	15	7	Shoot from human player station	1		Mess up all the balls in the neutral zone	N/A	
	Go over depot bump			2 robot climb L1	15pr	8	fuel from depot	N/A	5	Mess up opp's depot balls	N/A	
							Intake (ground)	N/A	4			
Auto							Intake (human player station)	N/A	3			
							Push Balls Towards our side (w/o mechanism)	N/A	0			
							Push balls towards our side (with mechanism)	N/A				
							Pass (from neutral to					
							Steal fuel from opps and shoot into your side (not hub)					
	Bump	N/A	2	Give fuel to human player	N/A		robot score fuel (in active hub)	1		Defense (neutral zone - hub)	N/A	6
	Trenches	N/A	5	Tower L1	10	6	Push Balls towards our side	N/A		defense (opposition side source)	N/A	4
	Go over depot bump	N/A		Lower L2	20	9	Intake (ground)		4	defense (opposition side hub)	N/A	7
	Auto-Align	N/A		Lower L3	30	10	Intake (outpost)		3	defense (neutral zone - bump)	N/A	3
Teleop							Push balls to human player			defense (neutral zone - trench)	N/A	3
	Bump	N/A		Tower L1	10	6	Push Balls towards our side w/o mechanism		0	Defense	N/A	6
	Trenches	N/A		Lower L2	20	9	Push Balls towards our side W mechanism			defense (opposition side source)	N/A	4
	Go over depot bump	N/A		Lower L3	30	10	1			defense (opposition side hub)	N/A	7
							Push balls to human player			defense (neutral zone - bump)	N/A	3
Endgame										defense (neutral zone - trench)	N/A	3

Strategy spreadsheet we created during kickoff to rank the importance of different robot actions to guide our design choices

Strategy Decisions

- The Student Executive Board led open meetings to organize the Master Robot Actions List into a MoSCoW list, which detailed our final robot archetype, with the goal of returning to the New England District Championship.
- We organized the actions into a MoSCoW list:

Must Haves	Should Haves	Could Haves	Won't Haves
• Score FUEL from a fixed range • Ground intake • Pass from the neutral zone to our alliance zone • Defense • Travel under the TRENCH OR over the BUMP	• Score FUEL from a variable range • Reversible intake • Travel over BUMP and under TRENCH • Intake from DEPOT	• Passing FUEL to the human player • Climbing L1	• Intake from the OUTPOST • L2 & L3 climb

- This helped us realize that we wanted to build a robot focused on shooting quickly and reliably over everything else and directed our research towards past robots from 2017 that had similar focuses.

Cycle Time Calculations

The strategy team developed a cycle time calculator that allows us to input different variables and output times for multiple cycle types. This tool has facilitated the setting of critical goals, including subsystem throughput speeds and hopper capacity.

Variables	Values	Units
Hub to Bump	1.00	sec
Bump Traversal	0.30	sec
Bump to Neutral Zone	1.00	sec
Hub to Trench	1.50	sec
Trench Traversal	0.10	sec
Trench to Neutral Zone	1.50	sec
Hub to Depot	1.50	sec
Hub to Outpost	1.75	sec
Hopper Capacity	48.00	fuels
Intake Speed	8.90	fuel/sec
Shooter Speed	6.00	fuel/sec
Accuracy	0.80	percent

Computed Variables	Formula	Units
Shooting full Hopper	8.00	sec
Intake full Hopper	5.39	sec

Cycles	Time	Description
Prep Cycle 1	9.99	Hub -> Bump-> neutral zone -> bump-> hub
Prep Cycle 2	11.59	Hub -> Trench-> neutral zone -> trench-> hub
Prep Cycle 3	8.39	Hub-> Depot -> Hub
Prep Cycle 4	8.89	Hub -> Outpost -> Hub
Prep Cycle 5	10.79	Hub -> Bump-> neutral zone -> trench -> hub
Shooting Cycle	8.00	shooting full cycle
Prep Cycle 6	3.2	Trench -> Neutral Zone -> Trench

Points / Active Shift	Points	Time
Shoot + PC 1 + Shoot	76.8	25.99
Shoot + PC 2 + Shoot	76.8	27.59
Shoot + PC 3 + Shoot	76.8	24.39
Shoot + PC 4 + Shoot	76.8	24.89
Shoot + PC 5 + Shoot	76.8	26.79
PC 1 + Shoot	38.4	17.99
PC 2 + Shoot	38.4	19.59
PC 3 + Shoot	38.4	16.39
PC 4 + Shoot	38.4	16.89
PC 5 + Shoot	38.4	18.79
Shoot + PC 6 + Shoot	76.8	19.20
Shoot + PC 6 + Shoot + PC 6 +Shoot	115.2	30.4

Each variable is input on the left, and times for preset cycles and FUEL scores are output on the right. The red coloring represents cycles that cannot fit in the 25 sec ACTIVE period

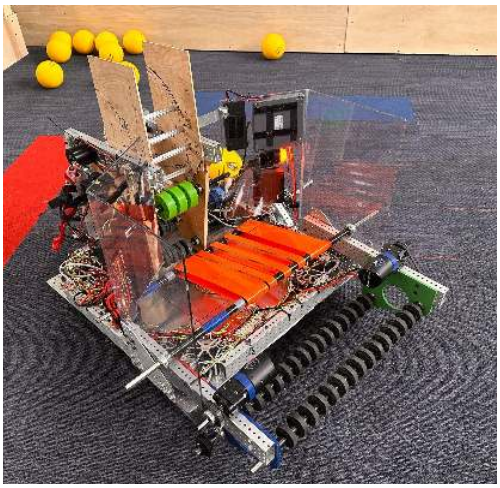
Research and Design

- The robot is divided into five subsystems:
- Each subsystem was designed with compatibility in mind to go under the trench.
    - Drivetrain -> movement and defense
    - Intake -> collecting FUEL
    - Hopper -> storage and transfer of FUEL
    - Shooter -> scoring FUEL
    - Vision -> autonomous routines and field alignment
- Noticing that REBUILT has many similarities with the 2017 game, STEAMWORKS, we took inspiration from a few of the best robots of that year.
    - One that stood out was team 254's robot. We loved the idea of a dual shooter to maximize FUEL throughput, as well as their extending hopper walls and roller hopper floor.
    - We also took inspiration from team 125's 2017 intake. On our own robot, we decided to use the idea of a roller on the top and bottom of the FUEL

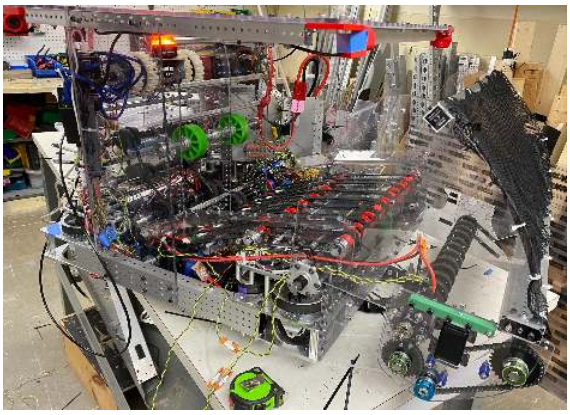
and modified it by removing the belt when we discovered the two rollers worked quite well.

We built prototypes for each subsystem and combined them into a functional test robot, while our CAD team refined designs based on prototype data. This approach also allowed programming to begin earlier.

Maximizing our scoring ability through a focus on software was very important to us this season. We wanted to aid our mechanical design with auto-align using AprilTags, strong auto routines, FUEL tracking, and creating fluidity of the FUEL path through our robot from a programming perspective.

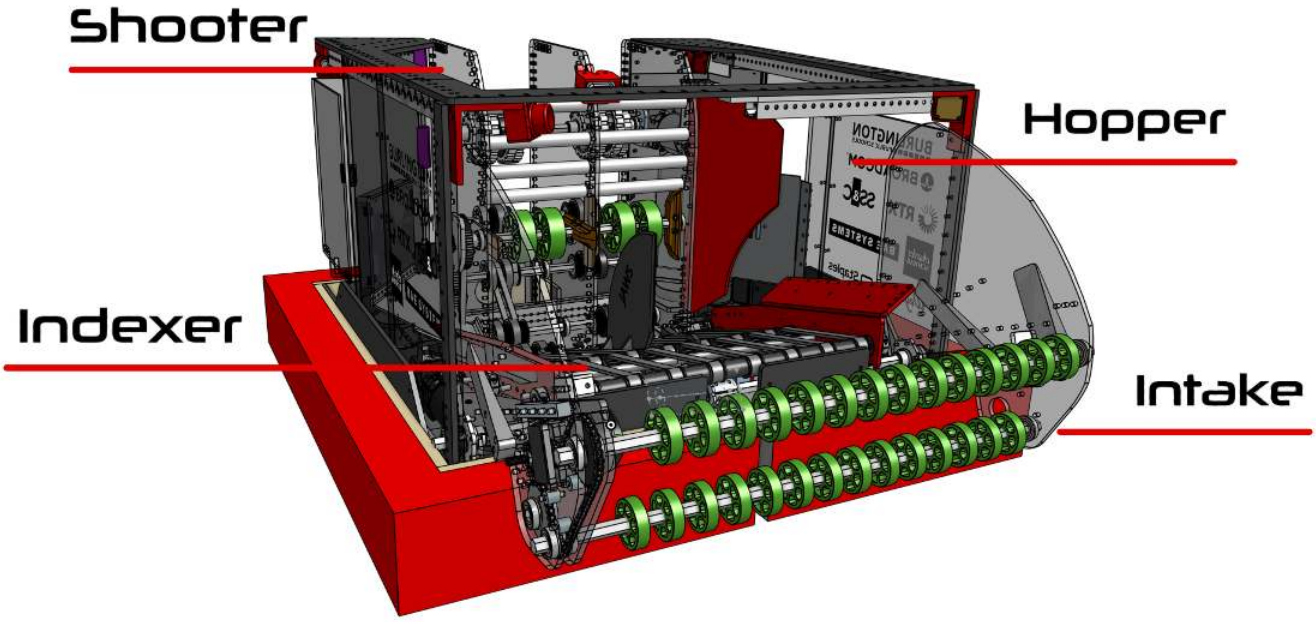


The scrappy protobot (named The RIG) with all individual prototypes integrated



Our 2nd protobot (named LeBrot)

Subsystems



Drivetrain

The robot drivetrain is the most critical part of any robot. Without a robust drivetrain, it is impossible to build a reliable and competitive robot. The importance of a robust drivetrain is further exacerbated by the defensive-heavy possibilities of REBUILT.

Goals

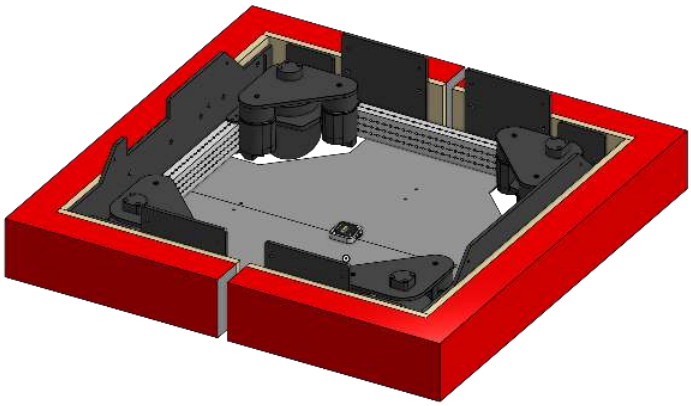
We therefore set the following drivetrain requirements for our robot this season:

- Robust and reliable
- Fast and agile
- Modular and easy to replace

Final Design

Given this, we decided on the following drivetrain configuration:

- Swerve Drive Specialties Mk4i with Kraken X60's, L2 Gearing
    - Provides a balanced torque-to-speed ratio, enabling strong acceleration while maintaining a competitive top speed
- Black Nitrile Tread
    - Allows for more wear resistance than blue neoprene tread
    - Allows for more grip compared to Colson wheels



Assembled swerve module with Kraken motors for the protobot

- 26" x 26" Size
    - Allows for optimized agility of swerve, maximizes space for FUEL, mechanisms, and electronics, while leaving space for polycarbonate hopper walls and bumper mounts to fix outside the drive frame rails.
- ⅛" Aluminum Bellypan
    - Kept our drivetrain lightweight and agile

Intake

Goals

We wanted to create an intake system that rapidly and reliably collects FUEL from the floor with minimal driver assistance, while ensuring consistent transfer to the hopper for continuous scoring.

Prototyping



Single-roller testing

Intake testing began immediately after kickoff, testing roller configurations and different wheel sizes. We drew inspiration from Team 4481's top and bottom roller intake, which picked up large amounts of FUEL quickly, and used super-compliant wheels to improve FUEL capture. We moved from a single-roller to a dual-roller layout and tested different hex-shaft positions with varying compressions and angles. We confirmed that two rollers with ~0.5" compression and a 24" intake width gave the most reliable, smooth intake without proper power to feed the indexer

Final Design

The intake subsystem acquires FUEL from the floor and transfers it into the hopper using dual counter-rotating compliant rollers.

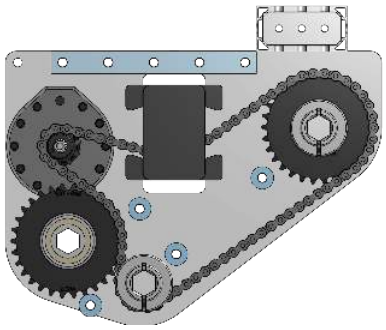
- 24" Wide
    - Allows FUEL to be intaked easier by autonomous routines and drivers
- 2" Black Super Compliant Wheels, with 2" wheel to wheel compression and 0.5" shaft to shaft
    - Compresses the FUEL to ensure reliable intake
- 2 Rollers, Steel Hex Shafts
- 1 Steel Hex Shaft in the rear to rotate the intake upwards to fit inside the 26x26 robot frame

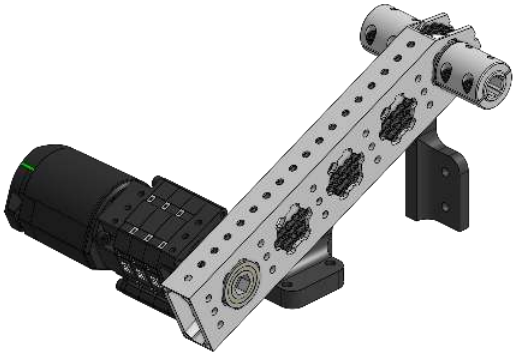
- Intake Deploy Actuator
    - 3D printed bracket that holds 2x1, which connects to the shaft on the intake and is bolted down to the belly pan and frame rails
    - Driven by a chain connected to a Kraken
- Chain Drive System to Drive Rollers
    - Counter-rotating rollers pull FUEL inward and direct it to the conveyor
    - Intake gear ratio is 1:1, and the actuator is 25:1
- Adjustable Chain Tensioner
    - Adjustable chain tensioner maintains proper tension throughout matches, and allows the chain tension to be quickly adjusted without the need to disassemble the mechanism
- 2 x 24” Churro
    - Ensures the intake stays stable when moving around and intaking
- Manufacturing: CNC-machined and 3D-printed components to minimize tolerance buildup
- Electronics
    - Intake Rollers powered by 1:1 Kraken
    - Intake Actuator powered by 25:1 Kraken



Final Intake Design

Chain Drive for Intake Rollers





Intake Actuator

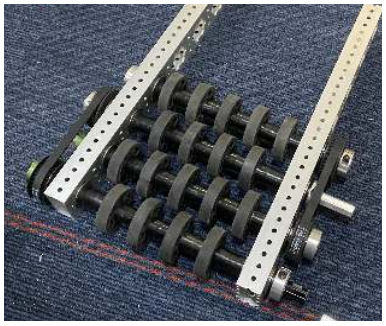
Hopper

Goals

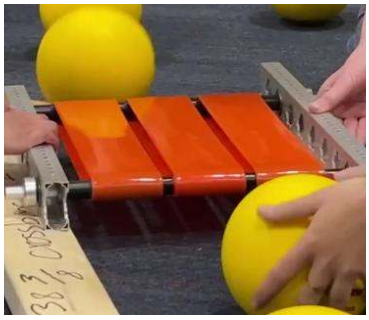
The “hopper” refers to the conveyor that transports FUEL from the intake to the indexer, as well as the structure that contains the fuel, electronics, and cameras. Our goal was to maximize the throughput of the FUEL from intake to the shooter.

Prototyping

We were inspired by Team 6324’s vertical powered wheel columns feeding the shooter and Team 3465’s V-groove bearing sliding hopper walls. We tested various hopper floors (wheels, wide/narrow polybelt (1”& 3”), and HTD belts) and chose HTD belts for their simplicity and effectiveness.



Wheels



Polybelt (3”)



HTD Belts

Final Design

The conveyor is composed of two stages of HTD belts. The first stage is on a slight slant over the swerve modules, and the slant of the second stage increases to both increase throughput to the indexer and save hopper space. It is run on a 5:1, REV Planetary, right-angle gearbox. For stability, there are threaded rods and polycarbonate plates running between the side plates. In the prototyping phase, we developed additional features:

- Removable angled side plates for easy access to wires below and next to the conveyor
- The central “fin” that splits the balls into two streams as they enter the shooter prevents blockages

- Fuel agitators that prevent an occasional dead space that was identified between the conveyor and indexer, and improve throughput



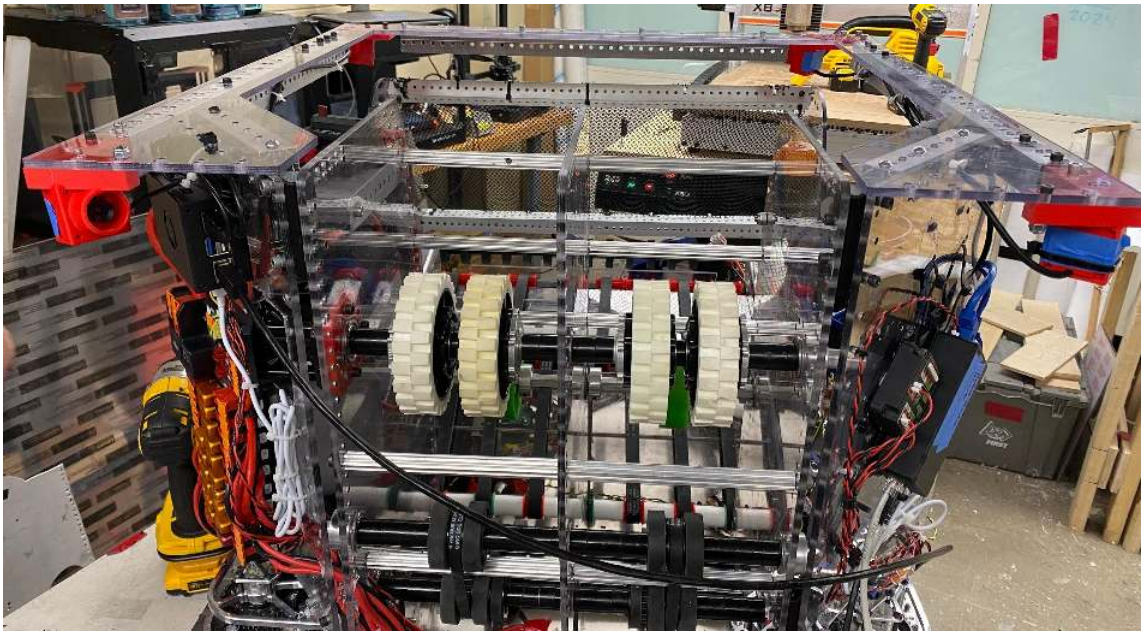
Final Conveyor Render

Structure

The hopper structure was built with ¼” and 1/8 “ polycarbonate plates, CNC’d by the team, and provides protection to internal parts as well as locations for attachment on other subsystems, and serves as a place to hold FUEL.

- Diagonal plates
    - These plates both protect the vertical electronic boards and guide the FUEL towards the indexer with the movement of the conveyor, preventing FUEL from jamming
- PDP cover plates
    - As our PDP and other electronics are situated on the exterior of the robot, we wanted to protect them during matches, so we created panels that cover them using polycarbonate.
- Side panels
    - These are the polycarbonate panels reaching from the bellypan to the top of the robot which hold in the FUEL
- ”Picture Frame” top cover

- This is the top layer of polycarbonate that strengthens the side panel structure and provides places for the cameras to attach to, as well as a large hole for FUEL to be placed into



Partially-built protobot with picture frame top cover attached and external electronics shown

Shooter

Goals

For our shooter, we had a goal of creating a high-accuracy shooter that is capable of scoring FUEL from multiple distances by adjusting flywheel speed alone, ensuring consistent scoring while minimizing missed shots.

Prototyping

We were inspired by Team 254's 2017 flywheel with backboard design and Team 5026's flywheel prototypes. We got a 0.91" baseline compression, and our testing 0.7"-0.10" confirmed ~0.9" worked best. We also referenced the Thriftybot indexer: a static incline with an omni-wheel shaft guiding FUEL into the shooter.



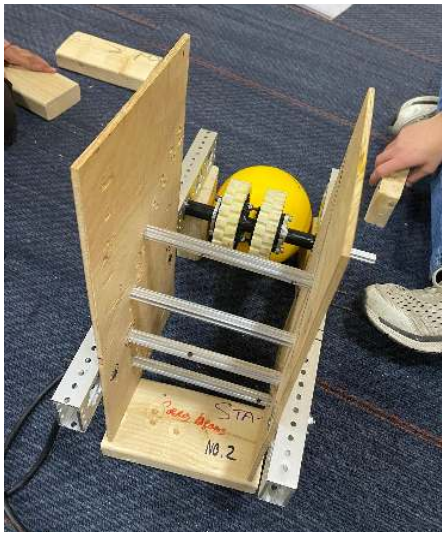
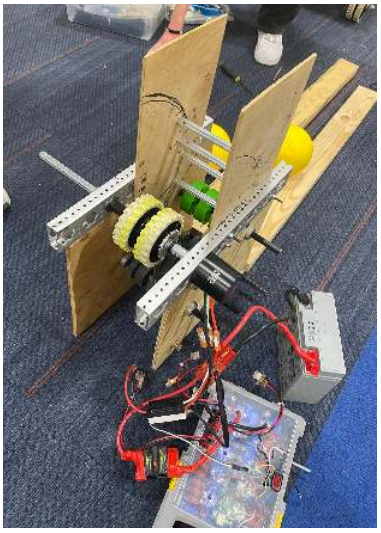
First shooter prototype, which was drill-powered

<u>Our prototype versions:</u>

V1: Churros in an arc, drill-powered flywheel (proof of concept)

V2: NEO-powered flywheel with an indexer added of 4 rows of compliant wheels beneath the FUEL path and an unpowered compliant wheel compressing from the top

V3: CAD-designed dual shooter with polycarbonate sides and more precise churro arc for greater consistency.


V.1 *V.2* *V.3*

Smooth vs Textured Flywheel Testing

While most prototypes we saw used smooth wheels, we tested textured 4” HiGrip wheels to improve traction on the slippery FUEL. Tests were run on our V3 prototype at full motor power, attached to a drivetrain with hopper walls. Horizontal distance was measured from the shooter. For vertical distance, we marked heights on a wall with tape and used slow-motion video to find the FUEL’s highest point frame by frame

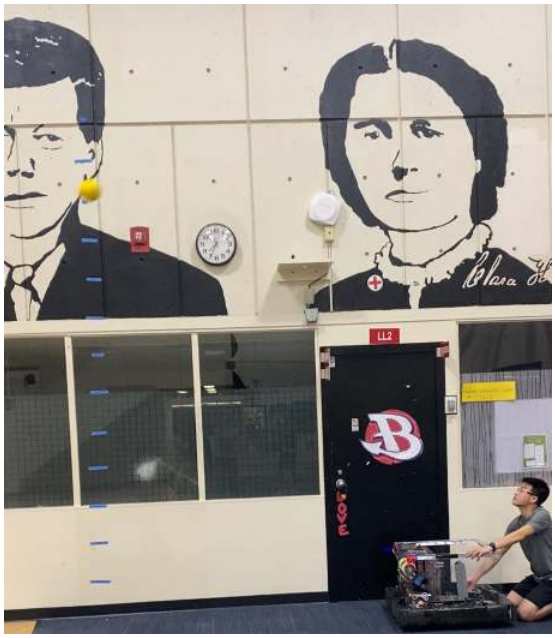
	Smooth	Textured
Maximum height	108”	132”
Mean	96”	128”
Standard Deviation*	7.2”	4.4”

	Smooth	Textured
Maximum distance	99”	126”
Mean	88”	119”
Standard Deviation*	6.8”	5.6”

Standard deviation reveals consistency, lower is more accurate



Smooth Wheel Testing



Textured Wheel Testing

Textured wheels dramatically outperformed smooth in both range and consistency, making smooth wheels suboptimal for scoring in the hub.

Passive vs Powered Large Indexer Wheel Testing

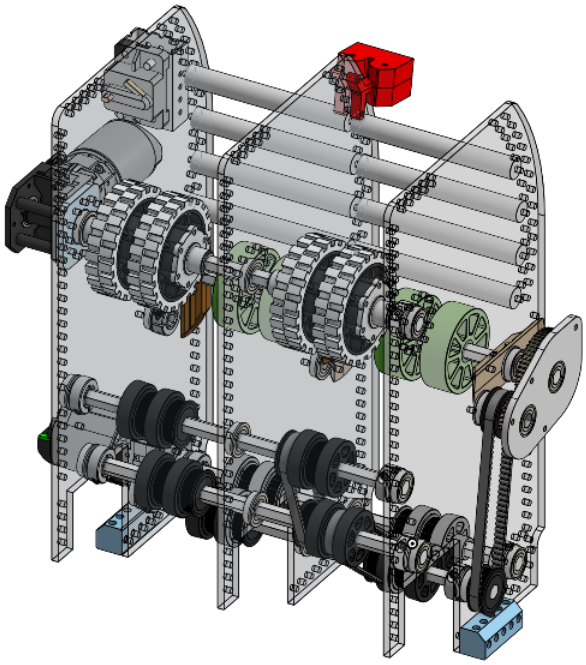
We tested the impact of powering the large (green) indexer wheel on accuracy and distance.

Indexer Wheel	Minimum Distance to HUB possible	Maximum distance to HUB possible	Shots Scored (25 FUEL)	Time to Empty
Powered	5”	204”	86%	16 sec
Unpowered	4’4”	126”	89%	56 sec

Only the powered version reached the HUB from the ~17ft OUTPOST area, and the unpowered wheel couldn't get the FUEL high enough to enter the HUB from close range, limiting scoring locations. Despite slightly better accuracy, unpowered, the restrictions that would be imposed on the shooting location would not be conducive to our strategy, and we ultimately decided to power the large indexer wheel.

Final Design

- **0.9” Flywheel Compression**
  Maintains consistent pressure for accurate, repeatable shots.
- **6.25” Side Horizontal Chute Length**
  Prevents lateral movement, reducing shot skew.
- **4” 80A HiGrip Wheels**
  High mass increases inertia for consistent shot power.
  80A durometer ensures firm, reliable contact.
- **Wheel Configuration: 2 center per channel, 0.75” gap**
  Eliminates side spin for straighter shots.
- **8 Aluminum Churros with Hex Covers**
  Smooth contact surfaces improve shot consistency.
- **Aluminum Hex Shaft**
  Lightweight for faster spin-up time.
- **1 Kraken Motor Control**
  Reduces power usage and simplifies maintenance.
- **90° Kraken Gearbox**
  Compact design fits within tight space constraints.
- **4” 35A Compliant Wheels**
  Provides grip to guide FUEL into the shooter.
- **2” 60A Compliant Wheels**
  Compresses and feeds FUEL into the flywheel efficiently.



CAD Model of Completed Shooter

Programming

All code is available and published on our GitHub repository:
https://github.com/DevilBotz2876/Rebuilt2026

The following critical software tools and libraries were used in the DevilBotz 2876's REBUILT 2026 software:

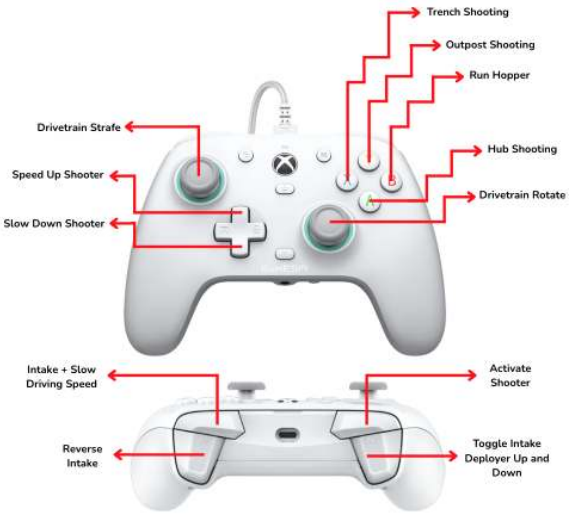
- WPILib
- AdvantageKit + AdvantageScope
- Pathplanner
- Phoenix Tuner X
- Photonlib + PhotonVision

Controls

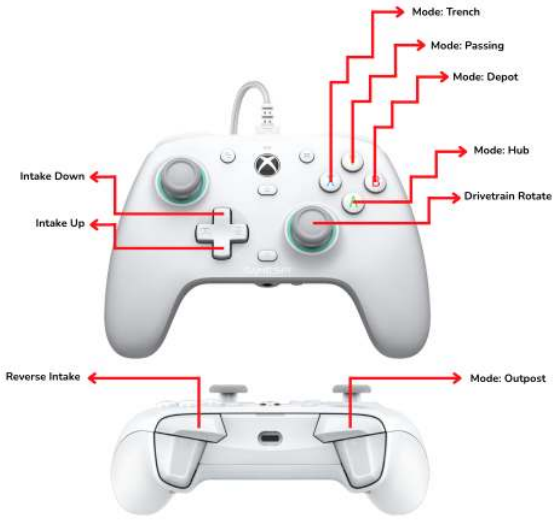
Driver Controls

In order for the *programming* and *drive* team to stay in sync and implement driving controls, a "Driver Controls" document was maintained to ensure the controls and functionality are understood by all members. The document included the following diagrams, along with more details on specific functionality.

Main Driver Controls



Operator Controls

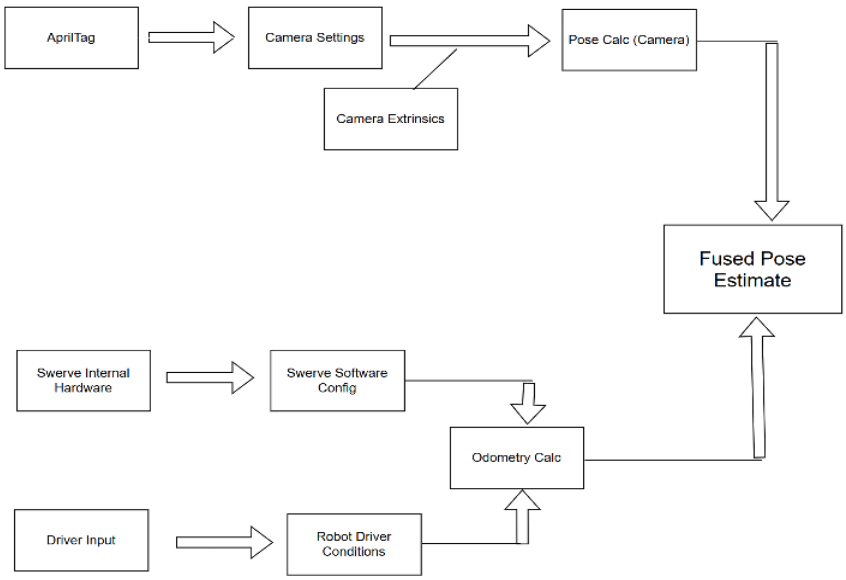


Vision

Goals

Robot derives smooth & accurate estimated pose from AprilTag detection & odometry

Kill Chain:



Final Design

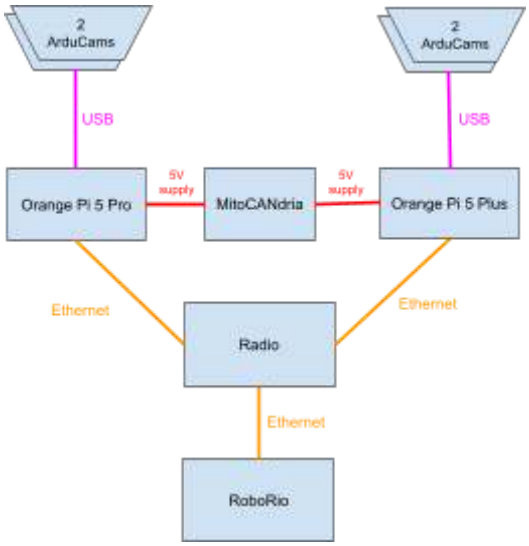
After realizing and testing optimal camera features, we landed on:

- 1 Orange Pi 5 Pro
    - Connected to RoboRIO via Ethernet
    - Communicates via NetworkTables
    - Running PhotonVision
- 1 Orange Pi 5 Plus
    - Connected to RoboRIO via Ethernet
    - Communicates via NetworkTables
    - Running PhotonVision

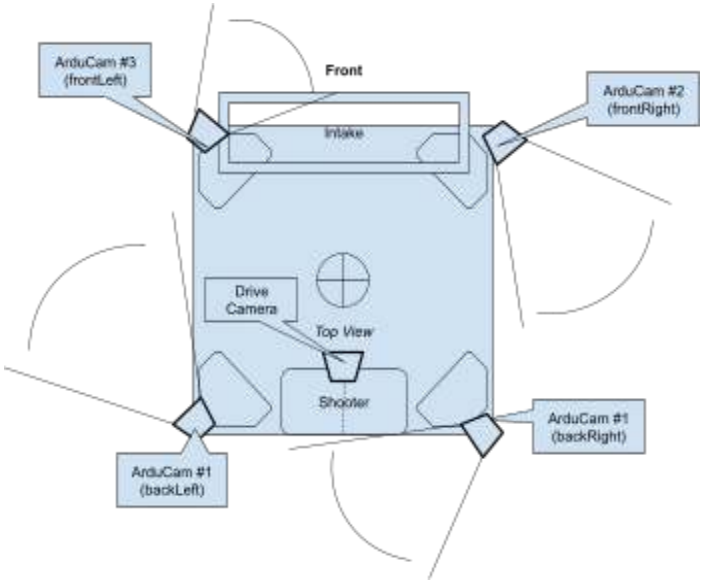
- 4 ArduCams
    - 2 connected to Orange Pi 5 Pro via USB

- 2 connected to Orange Pi 5 Plus via USB
- Custom-designed and 3D printed camera mounts

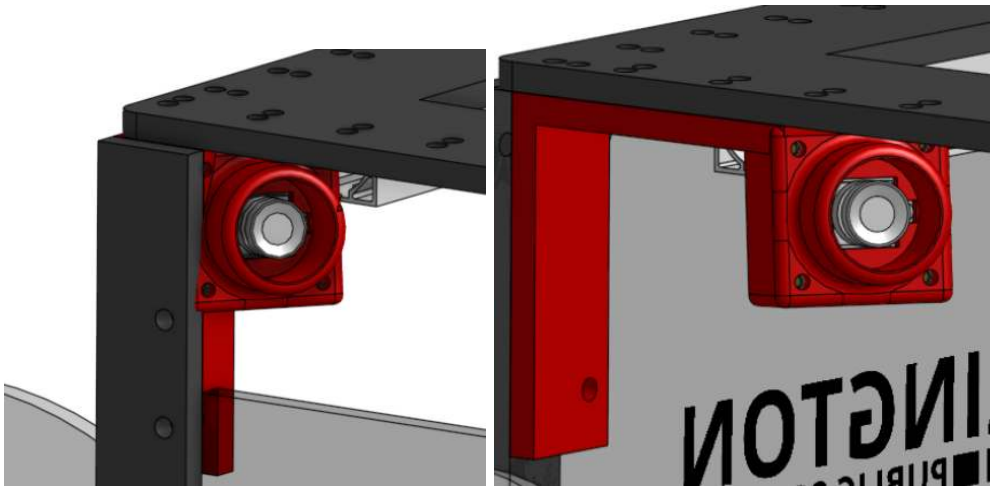
Connection Diagram

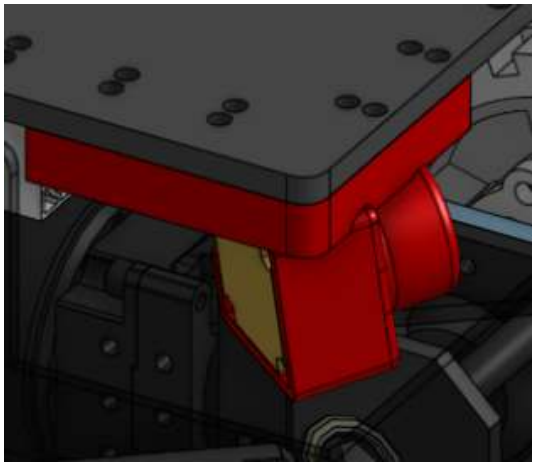
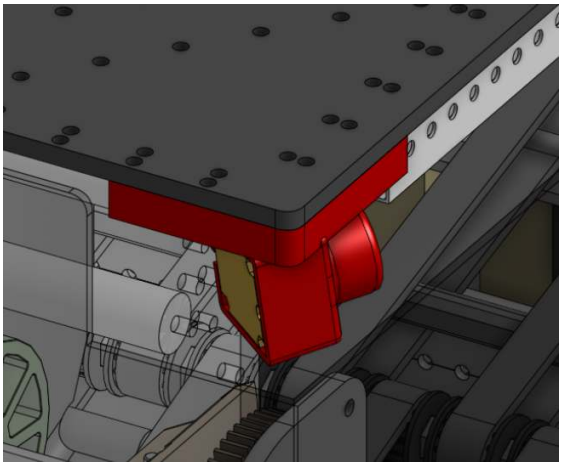


Camera Placement Diagram

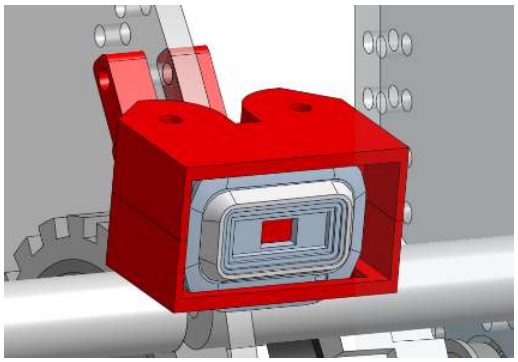
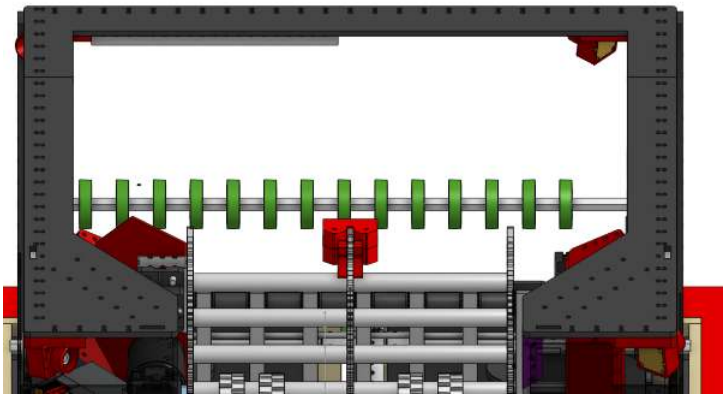


ArduCams



Driver Cam

Development Phases:

Camera Calibration

- This year, we vigorously tested our cameras and calibrated them to ensure they were optimal and could accurately measure the distance to the AprilTag.
- We completed camera calibrations for all four cameras on both robots and tested the strength of our calibration using the AprilTag on the TRENCH.
- As part of our testing, we had each camera face the AprilTag and see how far the camera thought it was away from the AprilTag on AdvantageScope versus the actual distance we measured manually using a rangefinder.

This year, DevilBotz 2876 refactored the vision processing. We wanted to develop and fix the problems we had last year.



Phase 1: 3D Pose Estimation

In this phase, we developed a new pipeline to validate the camera position provided by each camera. For a pose to be used in an estimated pose, one of the following must be true:

- A camera sees 1 AprilTag that is closer than 1.5 meters
- A camera sees the same AprilTag as another camera with the same timestamp

```
// Valid if: 1 tag within 1.5m, 2+ tags within max distance,
// or matching tag seen by another camera at same timestamp
if (singleTag && withinDistance) validPoseMeasurements.add(measurement);
else if (multiTag && withinDistance) validPoseMeasurements.add(measurement);
else if (getMatchingTagPoseMeasurement(cameraIndex, measurement).isPresent())
   validPoseMeasurements.add(measurement);
```

Phase 2: Automatic Robot Alignment using Vision Odometry

In this phase, the estimated pose is used to align the robot to the HUB. The camera provided an accurate pose, which was used to automatically rotate the robot with the HUB to determine the angle required for the robot to score.

The rotation was completed with a PID loop implemented and tuned to reach the desired robot yaw as fast and reliable as possible.

```
public static double getHubScoreRobotRotation(double x, double y) {
   double leg1 = DynamicLocation.HUB.getY() - y;
   double leg2 = DynamicLocation.HUB.getX() - x;
   return Math.atan(leg1 / leg2) + Math.PI;
 }
```

Phase 3: Drivetrain Odometry Updates using 3D AprilTags

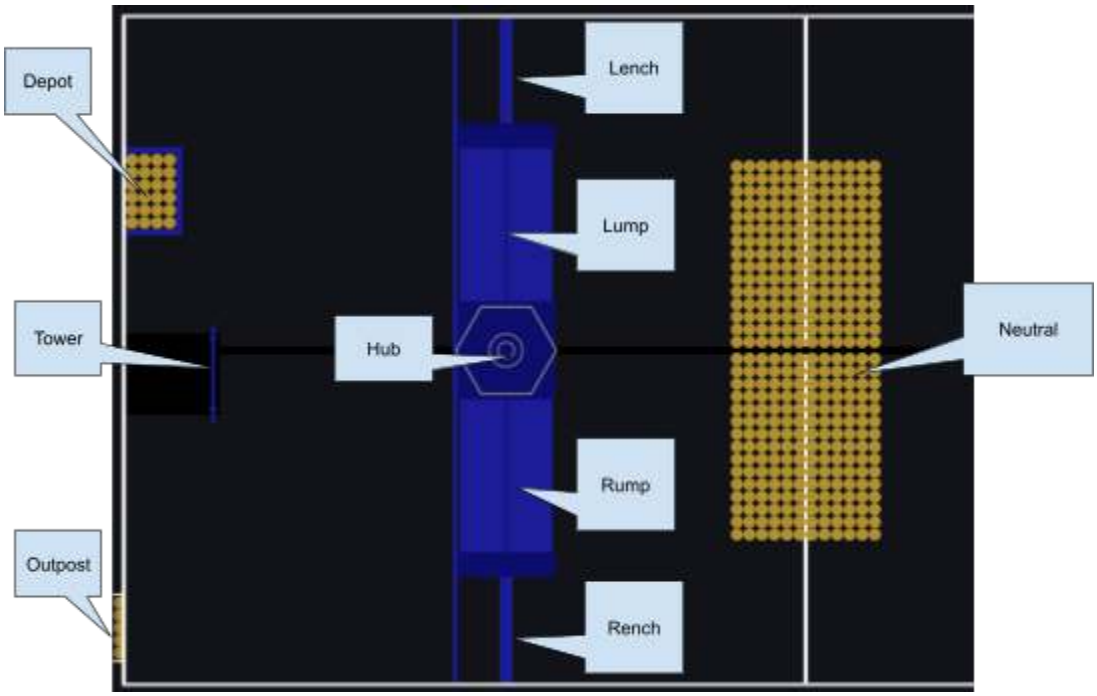
This phase requires at least the following known steps to achieve:

- Perform camera calibration using a grid pattern to enable “3D” AprilTag tracking
- Verify the reliability of the AprilTag-based odometry
- Feed the vision-based odometry estimates to the drivetrain

Autonomous

PathPlanner was used exclusively to develop the autonomous routines for REBUILT 2026.

Field Position Naming Key



In order for the strategy and programming teams to communicate clearly and concisely regarding desired autonomous routines, we developed a naming scheme for the various locations on the field

Routine Reliability and Adaptability

To build autonomous routines that are reliable while being able to adapt to each match strategy, we prioritized building upon one autonomous routine and moving it across different parts of the field.

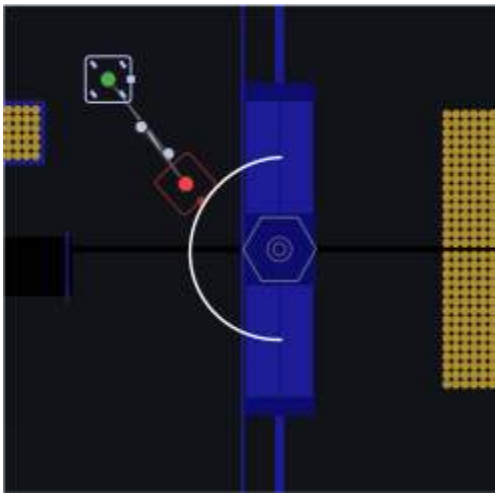
Main Auto: Score 8

Place the robot anywhere on the ALLIANCE side. It finds the closest point exactly *r* meters from the HUB along the robot-HUB line, drives there, and scores 8 FUEL.

$$D = \left(\frac{r}{\sqrt{1+m^2}} + H.x, \frac{mr}{\sqrt{1+m^2}} + H.y \right)$$

D represents the shortest pose to a set radius away from the HUB

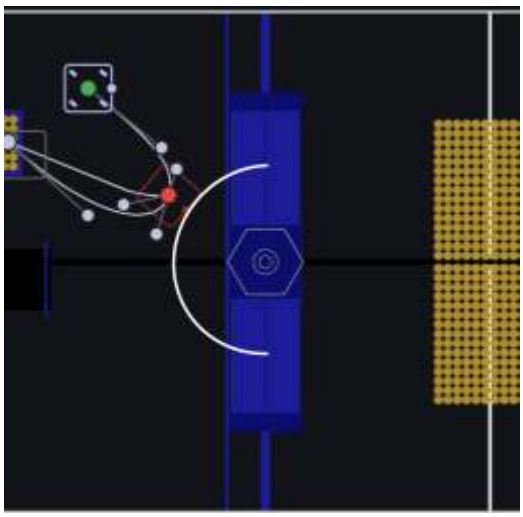
```
// Flip pose for Red alliance
if (alliance.get() == Alliance.Red) drivePose = FlippingUtil.flipFieldPose(drivePose);
// Find two candidate points at radius r along the robot-HUB line
if (drivePose.getX() - HUB.getX() == 0) {
    x1 = drivePose.getX();  y1 = HUB.getY() + radius;
    x2 = drivePose.getX();  y2 = HUB.getY() - radius;
} else {
    double slope = (drivePose.getY() - HUB.getY()) / (drivePose.getX() - HUB.getX());
    x1 = (radius / Math.sqrt(1 + Math.pow(slope, 2))) + HUB.getX();
    x2 = (radius / -Math.sqrt(1 + Math.pow(slope, 2))) + HUB.getX();
    y1 = ((radius * slope) / Math.sqrt(1 + Math.pow(slope, 2))) + HUB.getY();
    y2 = ((radius * slope) / -Math.sqrt(1 + Math.pow(slope, 2))) + HUB.getY();
}
// Return the closer point, facing the HUB
return dist1 < dist2
    ? new Pose2d(x1, y1, getHubScoreRotation(x1, y1))
    : new Pose2d(x2, y2, getHubScoreRotation(x2, y2));
```



An example auto is shown above of the robot going to the closest point on a set radius away from the HUB, and scoring 8 FUELS. The white line represents the set distance to the HUB

Auto: Score 8 + DEPOT

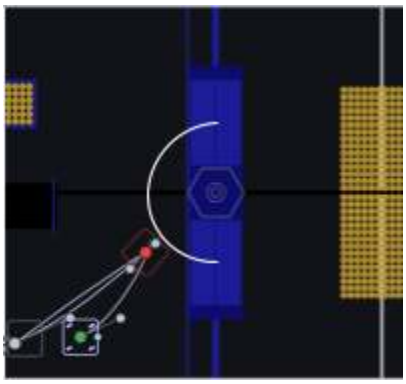
Place the robot anywhere on the ALLIANCE side. It will drive to the closest point that is a set distance from HUB and score 8 FUELS. Then it drives to the DEPOT to intake and scores the FUELS intaked.



An example auto is shown above of the robot scoring the pre-loaded 8 FUELS, intaking FUELS from the DEPOT, and scoring them

Auto: Score 8 + OUTPOST

Place the robot anywhere on the ALLIANCE side. It will drive to the closest point that is a set distance from HUB and score 8 FUELS. Then it drives to the OUTPOST to intake and score the FUELS intaked.



An example auto is shown above of the robot scoring the pre-loaded 8 FUELS, intaking FUELS from the OUTPOST, and scoring them