FTC engineering portfolio - 25905

Yellow IOT, Alkmaar

Mateo de Vos, Klaas van Loon.
Ties de Jongh, Melle Seghers and Ard Peeters.

Website: yellow-iot.nl
Email: team@yellow-iot.nl

FTC Season DECODE

Made on 5/12/2025 22:30.

Our graciously professional team:

(Image removed)

About us

We are a highly driven FTC robotics team from Junior IOT / Yellow IOT, a community that encourages makers to learn by building, experimenting, and engineering real solutions. Our team members come from different technical backgrounds, but we share one mindset: combining creativity with disciplined engineering. Through Junior IOT's workshops and open maker culture, we've gained hands-on experience in CAD, electronics, machining, programming, and iterative design.

For this season, we decided to push ourselves further by building a robot that reflects our ambition: mechanically robust, software-assisted, and packed with smart engineering. We value clear teamwork, problem-solving, and constant improvement. Whether we are designing parts in CAD, tuning PID loops, machining components, or assembling mechanisms, our goal is always the same - to build reliable systems through thoughtful engineering and consistent testing.

As Team Yellow IOT (Team 25905), we bring together curiosity, technical skill, and competitive spirit. We are proud of the robot we engineered this year, and even more proud of the knowledge, collaboration, and resilience we built as a team along the way.

Members

Mateo - Programming & CAD

Develops all major software systems, including turret auto-aim, PedroPathing, and flywheel PID. Also supports CAD design and makes a general design for Klaas to go on.

Klaas - CAD & Mechanical Assembly

Designs most custom robot components in CAD and helps assemble and refine the robot's mechanical systems.

Melle - Mechanical Assembly

Assembles some parts following the CAD design.

Ties - Programming & Assembly

Works on control logic, and helps assemble key mechanical subsystems to align code with hardware.

Ard - Assembly & Team Support

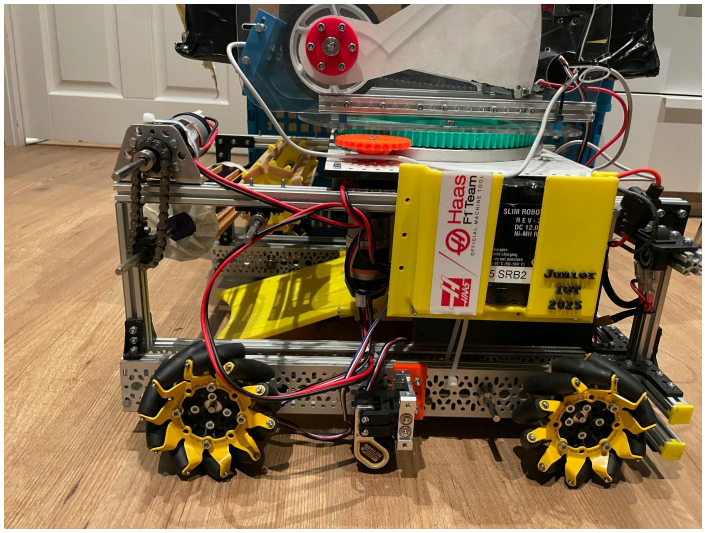
Ard supports the team by helping with general assembly tasks, he made a nice wooden box to easily transport the robot. He also makes sure the team has the food - needed to stay productive during long events.

(Image removed)

Robot overview

Drivetrain:

Mecanum with 104mm goBILDA wheels and 4x Rev robotics HD hex motors with planetary 5:1 and 4;1 gearboxes which equals to 20:1. mounted on the inside with some brackets to a c-channel chassis.

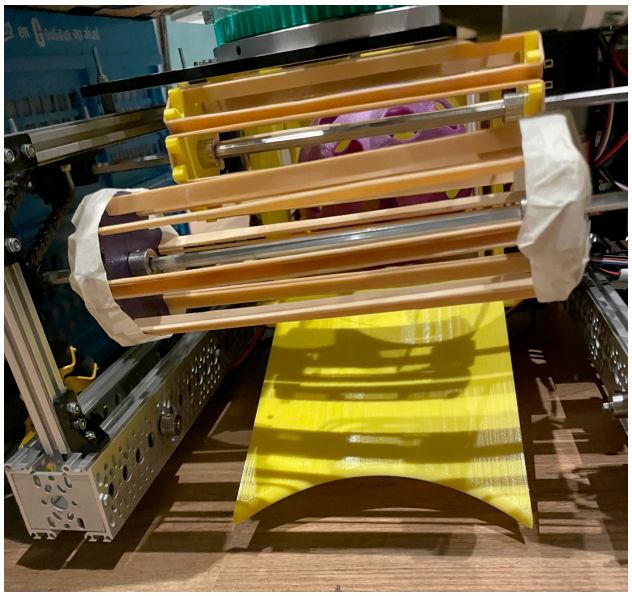


Intake:

Rubber band combine harvester style intake at 2k rpm with a Rev robotics HD hex motor 3:1. And two chains to connect the motor and two rubber band rollers. Our intake is designed based around the principle “touch it, own it” which means we only want to touch the artifacts in order to easily bring them into the robot.

Transfer:

A curved ramp with an additional rubber band combine to push the ball to a little box. There it will sit in place on a slope ready to shoot. When it's time to shoot, a servo arm with custom spoon attachment pushes the ball up into the flywheel.

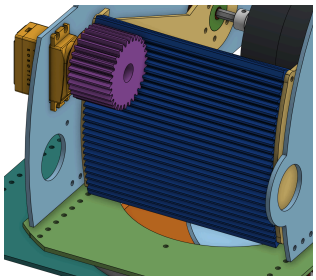


Turret:

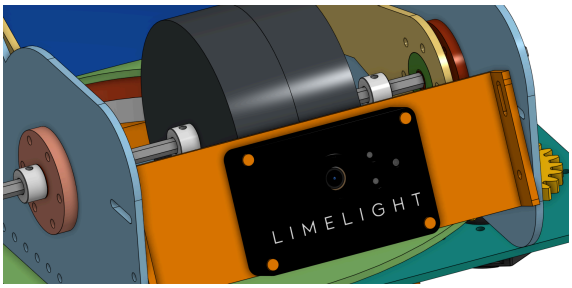
The shooting mechanism is mounted on a self aiming custom turret. It's made with a big 3d printed gear on top of a lazy susan bearing to hold the weight. The big gear is driven with a smaller gear mounted on another Rev robotics HD hex motor.

Shooting mechanism:

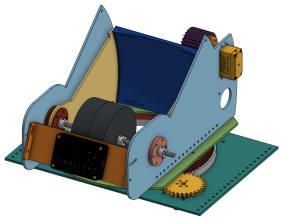
We shoot the artifacts with a custom made adjustable hooded shooter. It consists of 3 main parts:



1.



2.

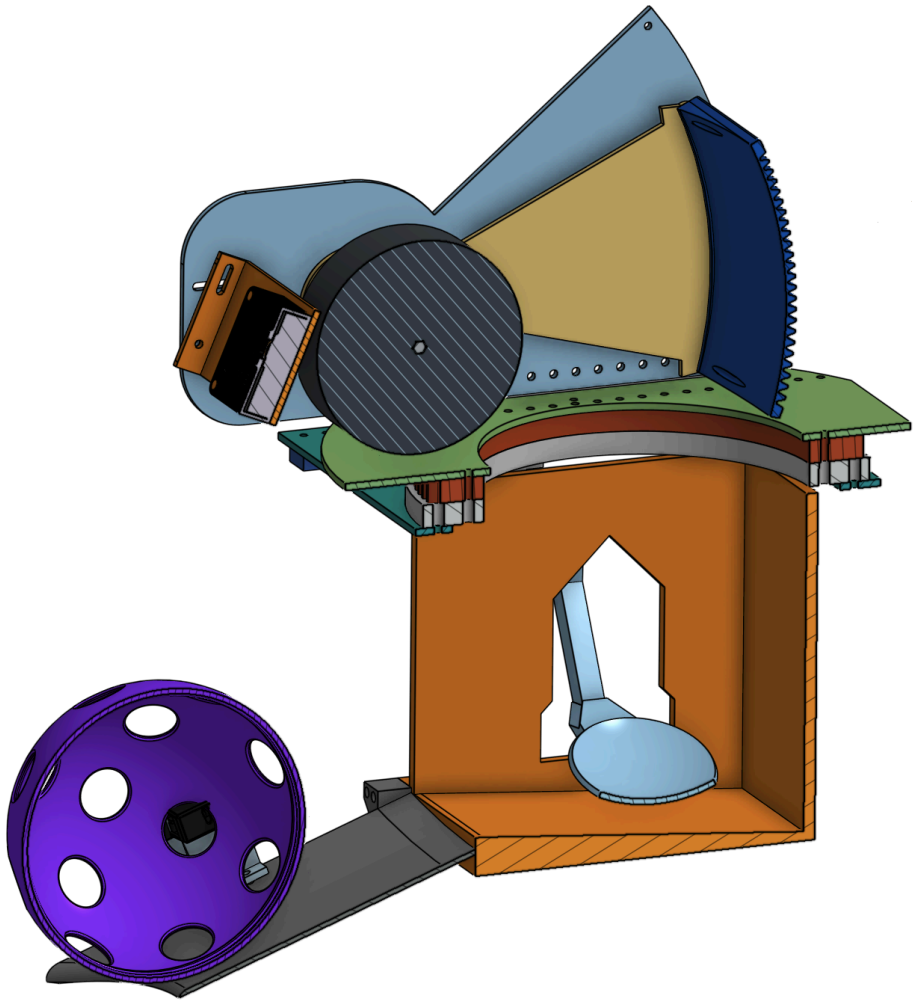


3.

1. A flywheel made from two rubber wheels and a custom steel weight we made on a lathe. The weight is for more inertia and consistent shooting rpm.
2. A 3d printed hood with gear profile on the back. This makes it possible to adjust the angle we shoot by changing the angle of our hood with a servo.
3. A limelight 3A mounted on the front to track april tags and estimate distance to the goal for auto aiming.
4. We use a “zombie axle,” where the same shaft supports the hood and drives the flywheel — a compact and efficient coaxial design.

Highlights

- ★ Our auto aiming is super fast because instead of only using the camera to track the april tag on the goal, we actually implemented IMU sensor fusion to "guess" where the goal will be when we go so fast the camera can't keep up. Then when we aren't moving as fast anymore the camera corrects the last error.
- ★ We use deadwheels with PedroPathing to easily create autonomous routines
- ★ Most of our robot is made from custom designed parts either laser cut, 3d printed, cnc-machined or turned on the lathe.
- ★ Our hooded shooter uses a clever “zombie axle,” letting the same shaft support the hood and drive the flywheel. It’s a compact, efficient setup that keeps everything aligned and reduces extra hardware.



Mechanical Design Process

Design Goals

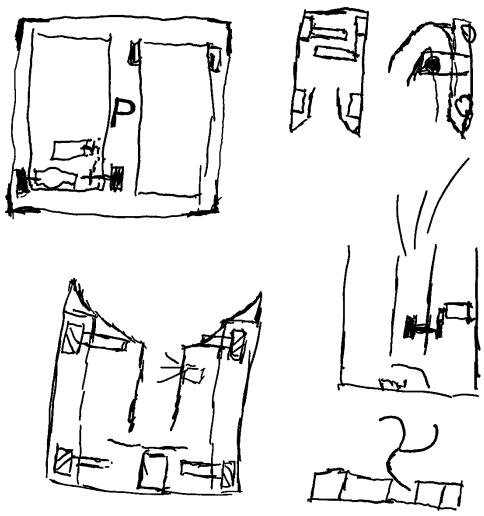
At the start of the season, our main objectives were:

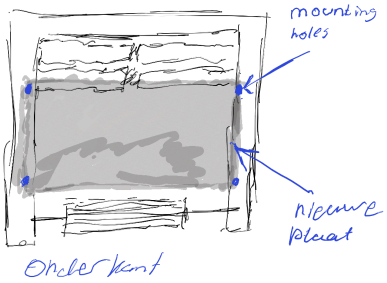
- Develop a self-aiming turret to score without rotating the robot
- Ensure smooth artifact flow from intake to shooter
- Trial and error a good touch and go intake
- Grow in autonomous performance

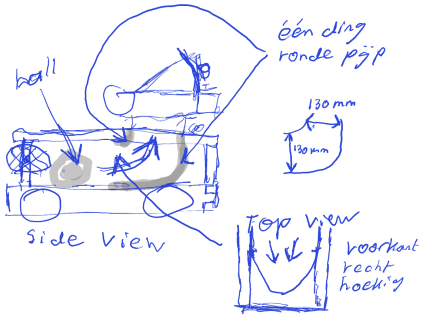
Brainstorming

We began by analysing the game and identifying the most valuable scoring actions. We quickly saw two possible strategies:

- Fast shooting with some sort of turret or auto aim to effortlessly score a lot of balls.
- Slower scoring with an indexing system built into the robot and a less accurate shooter focussing on pattern points. We thought about which was the best option and made a list of Pro's and Con's for both designs and agreed that option one would be the best.



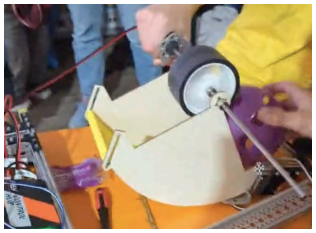
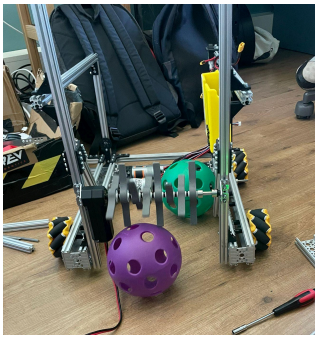
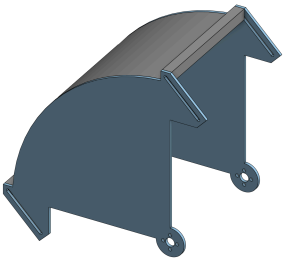




Prototyping

Early prototypes included:

- A laser cut wooden prototype of the hooded shooter to test compression and range.
- A simple flipper based intake

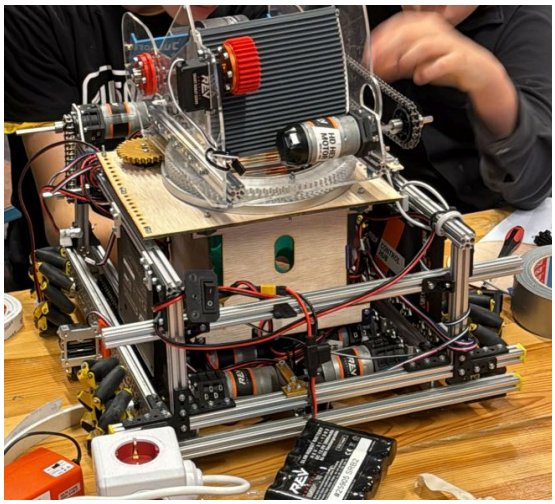




1. 2. 3.

These prototypes revealed important insights:

- Higher flywheel inertia was necessary for avoiding jams and consistent shooting
- The turret needed a stronger base to prevent wobble and flex
- The intake didn't follow the "touch it, own it" principle, so we redesigned it to be a dual set of combine intakes with thick rubber bands.

  These discoveries guided the first real version of the robot.



Iteration & Improvement

We went through multiple different designs:

1. Shooter Improvements

- The initial flywheel lost too much speed when shooting artifacts. This also helps with avoiding jams from low inertia causing artifacts to get stuck in the hood
- **Fix:** We machined a custom weight on the lathe to increase rotational inertia.
  Result: More consistent shot velocity, less effect from battery voltage drop and fewer jams.

[picture of ball jammed (faked)]

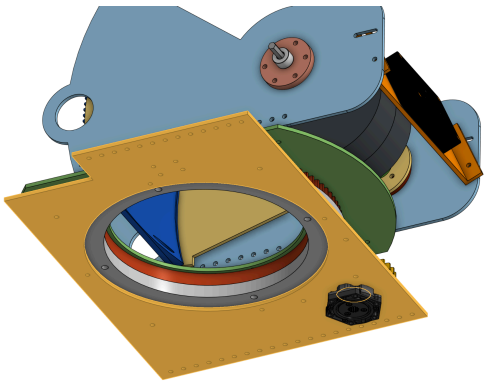
2. Hood Angle Adjustment

- We needed an adjustable trajectory for shooting in the goal from anywhere in the field and made a gearing on the inside of the hood but didn't have enough torque.
- **Fix:** Designed a hood with a gear profile on the back, allowing servo-controlled angle adjustment with massive torque.
- **Result:** Variable shot arc and more flexible auto-scoring.

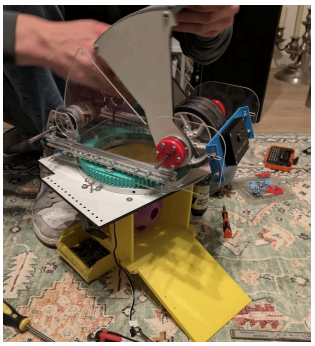
3. Turret Stability

- Early turret versions were mounted on a laser cut wooden board, which had a large amount of flex.
- **Fix:** Make the robot mount out of compact panel
- **Result:** Increased stiffness and improved accuracy

1.



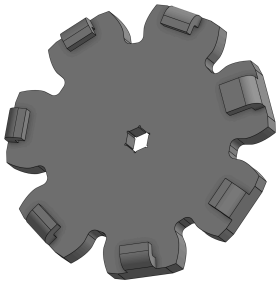
2.



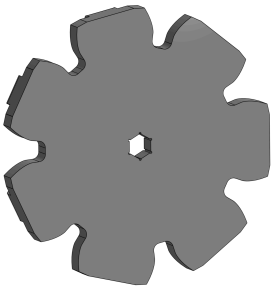
3.

4. Intake Reliability

- The flipper based intake wasn't reliable when taking in artifacts, requiring the artifacts to be pushed against the wall. That also pushed against our principle of "touch it, own it".
- **Fix:** Changed from one flipper intake to two combine style intakes which went above a ramp.
- **Result:** More reliable artifact intake and accomplishes "touch it, own it".



1.



2.

3.

Final Mechanical System

The result is a robot with:

- Stable artifact scoring loop
- Fast dual combine intake
- Smooth transfer system
- Amazingly speedy self-aiming turret
- An adjustable-hood flywheel shooter

This mechanical foundation enables consistent scoring and reliable automatic aiming.

Programming Overview

Software Philosophy

Our programming approach focuses on reliability, modularity, and assisted scoring. We designed our code so that our drivers can focus on movement while the robot handles aiming, distance estimation, and shot consistency automatically.

Programming Environment

- Language: Java
- IDE: Android Studio
- Control System: REV Control & Expansion Hub + Driver Hub
- Vision: Limelight 3A (AprilTag detection + distance estimation)
- Sensors:
    - 2× goBILDA 4-bar odometry pods
    - Built-in Control Hub IMU for heading
- Motion Planning: *PedroPathing*

Software systems

1. **Auto aiming turret system**

   Our most advanced feature is the self-aiming turret.

   - Reads robot orientation from the IMU
   - Gets target yaw & distance from the Limelight AprilTag pipeline
   - Calculates required turret angle and flywheel RPM
   - Turret angle is set using a PID loop for smooth rotation

   This allows the robot to aim independently of drivetrain movement.

2. **PedroPathing Motion System**

   Our autonomous pathing uses the PedroPathing library for smooth, predictable robot motion.

   - Uses 2 odometry pods (left and right) for accurate XY tracking
   - IMU provides heading and drift correction
   - Path following is consistent even without AprilTag localization
   - Designed routes include leave and automatic intake

   This creates fast and reliable autonomous routines with minimal tuning.

3. Custom Flywheel PID Controller

- The flywheel uses a fully custom PID loop, tuned for fast stabilization and minimal overshoot.
- Maintains consistent velocity under battery sag
- Reduces battery voltage drops
- Works with both manual RPM settings and auto-aim lookup-table values
- Ensures each shot has the same speed for accuracy
- Drivers can switch between manual RPM mode and auto RPM mode depending on strategy.

4. Intake & Transfer

To reduce mistakes during matches:

- Intake is activated with a toggle to remove overhead for drivers
- The flywheel motor turns off before the transfer servo actuates, to save strain on the battery because of stall current
- Logic prevents firing when the RPM is below the target

6. TeleOp Driver Enhancements

The robot includes several quality-of-life improvements:

- Slow-mode / fine-control for alignment
- “Auto-aim” button: turret snaps to calculated angle
- Safety limits to prevent turret cables from over-rotating
- Controller buzzes when the flywheel target speed is reached

7. Autonomous

Our autonomous routines use:

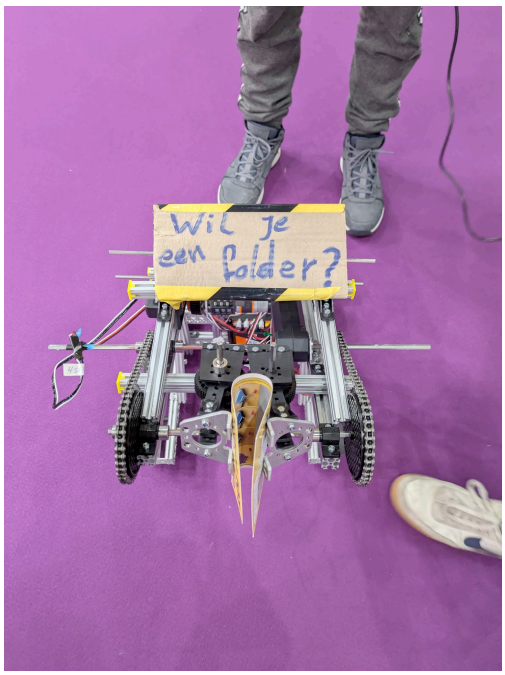
- PedroPathing position control
- IMU-based heading correction
- Timed intake control
- Automated shooting using distance → RPM → hood lookup values

No AprilTag localization is used - autonomous runs purely on odometry + IMU.

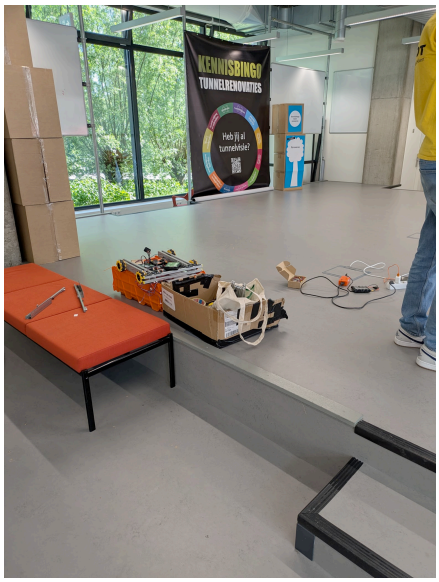
Outreach

We went to

- **NOT (Nederlandse Onderwijs Tentoonstelling, Dutch Education Exhibition)**

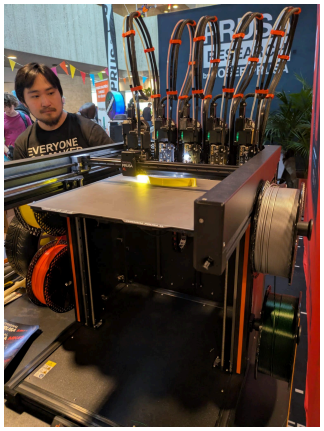


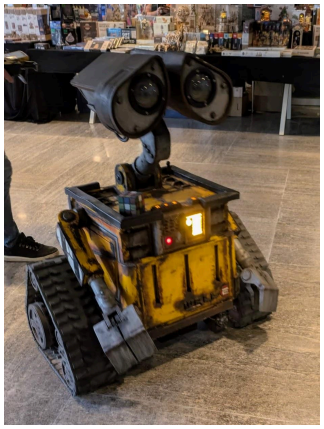
- **Maker days Delft**



- **European Premier Event**

- **Maker days Eindhoven**





- **Scrimmage Noordwijk**