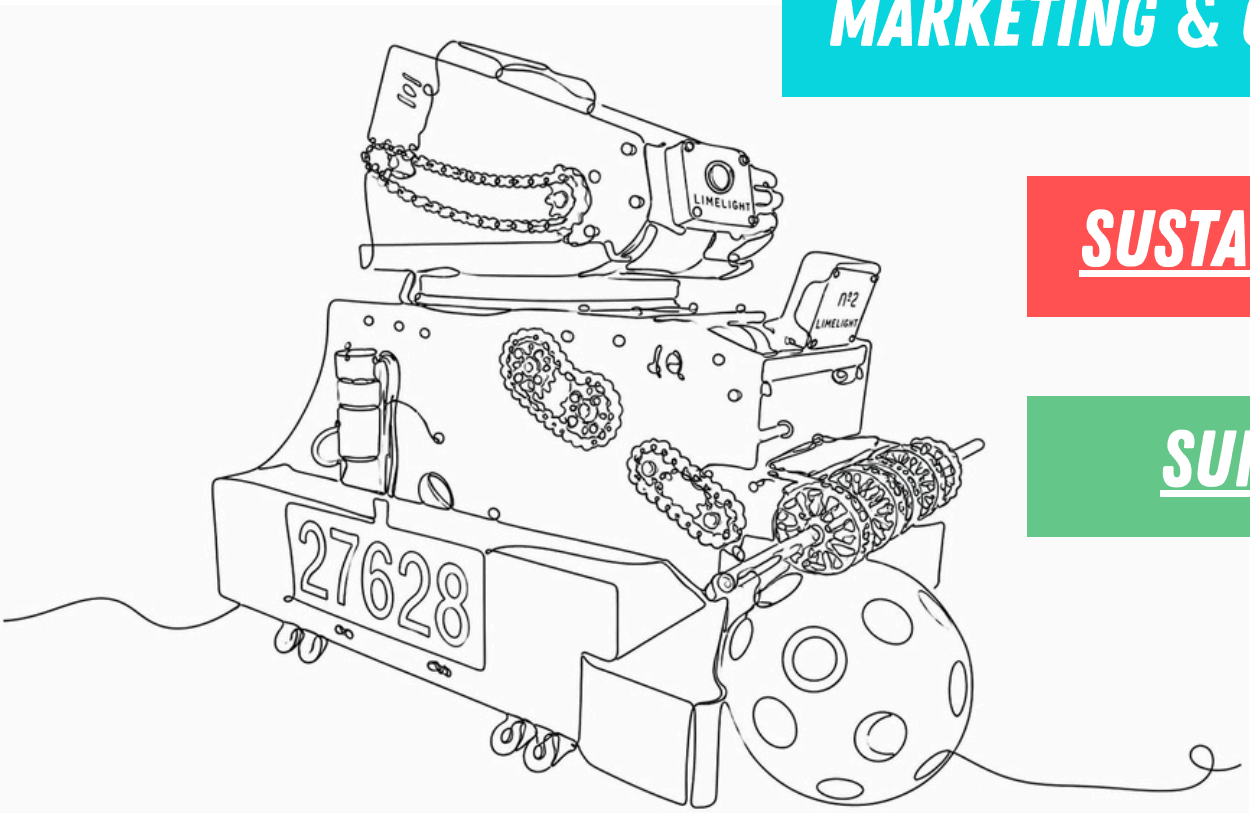
#27628 Beargineers

Engineering Portfolio

International School of **Amsterdam**

2025-2026, Decode

TABLE OF CONTENTS



TEAM

Operations:

Yasha (Grade 11) - supports all teams, organizes & delegates tasks. Designed the latest iteration of the robot as the Grade 12 students entered diploma exam season

Software:

- **Nina** (Grade 12, lead) - second year as software lead, programs with Python, Kotlin, HTML, PHP, SQL
- **Yalin** (Grade 10)
- **Nicolas** (Grade 11)

Hardware & Design:

- **Mirgali** (Grade 12, lead) - second year as hardware & design lead, mechanical and electrical focus
- **Josh** (Grade 12, lead) - second year as hardware & design lead, mechanical design and 3D design
- **Terentiy** (Grade 12)
- **Celine** (Grade 11)
- **Ishan** (Grade 11)

Marketing:

- **Konrad** (Grade 12, lead) - second year as marketing lead
- **Tiffany** (Grade 10)
- **Helena** (Grade 10)
- **Freddy** (Grade 12)

Mentors:

- **Ford Watson** - mathematics teacher at ISA, supports the team with problem-solving strategies and logical reasoning
- [new] **Maxim Shafirov** ex-CEO of JetBrains. Contributed to the creation of Android Studio and the Kotlin programming language. Supports the team in programming

Strategy Fork

Optimize for **complete** feature set (scoring, pattern, parking), or focus on **one** particular aspect and hyper-optimize?

A win gives 3 RP, and completing even one additional field feature adds 1 more RP. That total of 4 RP is better than the 2 RP available in a loss. Pattern sorting takes time, which reduces scoring throughput. A full double-parking lift also adds weight and consumes match time. Since scoring is the required base layer anyway, we concluded that Decode is a race for throughput.

Race for Throughput
and any compromises are undesirable.

Compromising ranking points for game points turns out to actually not be a compromise. Watching top teams' performance as well as final RP thresholds announced for FIRST Championship confirmed **our choice for strategy**.

This strategy decision drove our robot design and iteration process.

During this season we went through three complete robot rebuilds. Each of the rebuilds was started when we realized that no incremental improvements can adequately improve our robot's scoring throughput.

1st gen robot: **ALPHA -** scrimmage

The design is taken from a publicly available design by team 23070 which was a quick way to start the season, having a starting point to test hardware and software

Pros

- Reliable artifact uninterrupted path
- Separate intake and feeder motors, makes sure that artifacts will not be fired when not needed
- Side panels made out of a strong material

Cons

- Hand cut panels lack precision, which makes shooter shoot to the side and have more friction
- Too much pressure on the artifact causes backspin
- For localization, motor encoders were used, which were not precise enough
- For pose correction, a webcam was used, which also lacked precision
- For intake and feeder low RPM motors were used, which had unnecessary torque and not enough speed
- Hard to change number plates which were secured with tape

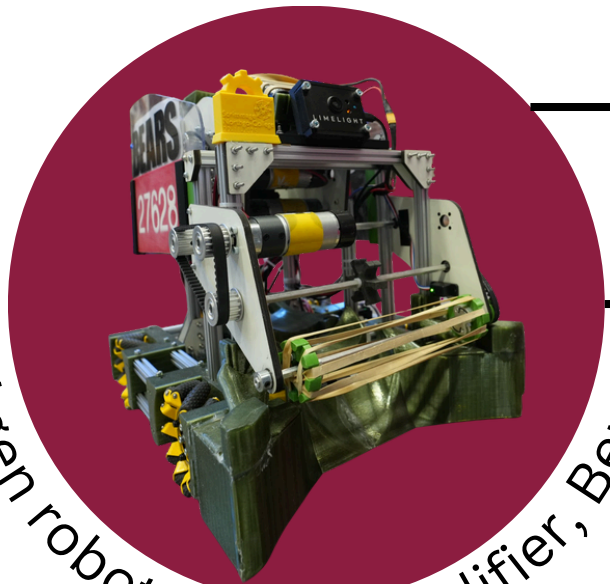
MECHANICAL DESIGN

Covers for team name and number

Transparent covers to cover up the messy electronics, provide a space for a slide-in mechanism number plate holder and team name for a more recognizable robot

Positioning

goBILDA Pinpoint localizer - used for position tracking
Limelight - used for position correction by tracking AprilTags



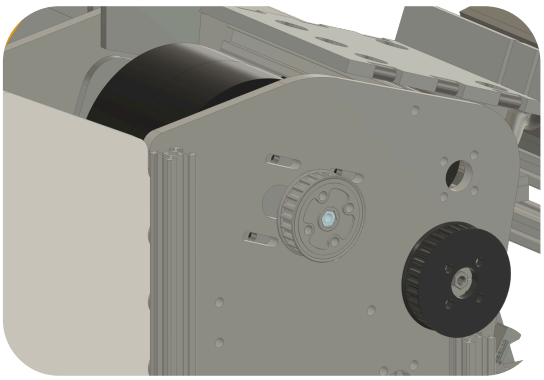
2nd gen robot: **BETA** - qualifier, Benelux

Faster intake and feeder motors

Ensures fast pickup and fast feeding into the flywheel for fast shooting - 1080 RPM each

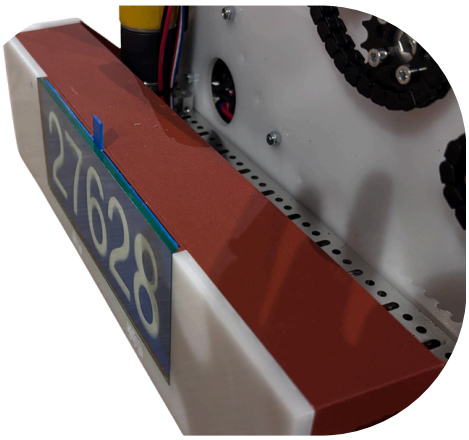
Adjustable flywheel position

Allows us to adjust flywheel compression to balance range and backspin



27628

MECHANICAL DESIGN



Sleek body design

Makes for a robot with no wires showing, and sides without any bumps, so when the robot comes into contact with another robot or the wall, it does not stick, and instead slides. It incorporates the number plate holder with no compromise to the flatness of the body.

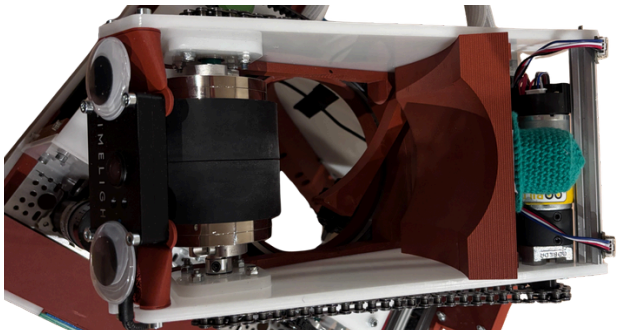
Turret

Allows the robot to shoot into the goal regardless of its heading. However, it complicates the design, reduces reliability, and requires a dedicated motor for turret movement.

3rd gen robot: **GAMMA** - the WORLDS

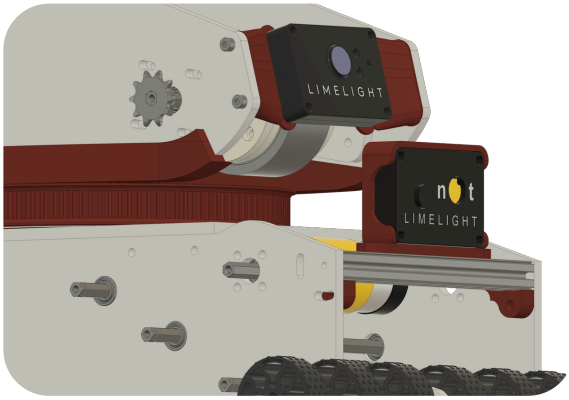
Single motor for intake and feeder

As one motor is required for the turret, the feeder motor was compromised. In order for the artifacts to not go into the flywheel before shooting is required, a latch was put in place for stopping the artifact, actuated by a servo.



Front facing camera

A Logitech camera in a custom Limelight-style housing makes the vision modules easily interchangeable allowing to choose the best tech for recognizing artifacts on the floor and automatically collecting them.



For **AUTO**, optimizing throughput required:

- maximum flexibility to match what our alliance partner can do
- dynamically calculated optimal pathing for the chosen AUTO sequence
- Reserving the last 0.5 seconds of AUTO to exit the launch area and secure a MOVEMENT RP.

To address **flexibility,** we came up with an interpretable pseudo-code to configure a specific AUTO sequence. The driver can choose from a preconfigured list during INIT phase or create ad-hoc sequence using a gamepad controller.

```
# Programs ============================================================
# B or F - Switch to shooting from Back or Front. Also, initial pose
# P - Push alliance partner out of launch zone (if they don't have AUTO)
# 1, 2, 3 - collect from the spike, where #1 is the closest to the Back
# 0 - collect from the human loading zone (farthest from the goal)
# 4 - open and collect directly from the ramp
# / - Shoot now. By default it shoots before collecting more if loaded
# Human readable program name after ':'
ProgramCatalogue=BPF2R31:Push bot,
                 F2R31:North,
                 B10000:Last spike and box cycling,
                 F2413:All spikes + ramp
```

While **optimizing pathing** we faced multiple issues. Here just a couple of them:

- Depending on collection order shortest path to the launch area may disturb artifacts, that we haven't collected yet. We solved this by maintaining the list of remaining artifact positions and corresponding **"no-fly" zones**.
- While collecting from the ramp a timeout-based completion criterion sometimes led to collecting 4 artifacts. Also, timeout-based criteria for shooting completion didn't work well with small shooting time inconsistencies. As a solution we installed 2 REV distance sensors to **count artifacts in and out**. Unfortunately this had very significant negative impact on our loop time.
- Later on we replaced one of the sensors with a goBILDA sensor, which has a reading time ~10x shorter and used motor current draw instead of a second sensor.

```
override suspend fun DecodeRobot.autoProgram() {
    // Addressing MOVEMENT RP. Master coroutine cancels program 0.5 seconds
    // before timeout and drives to a final position
    cancelWhen( condition = { opMode.elapsedTime.seconds() > 29.5}) {
        interpretProgram(program)
    }

    driveTo( target = Locations.OPEN_RAMP_APPROACH)
}
```

SOFTWARE & CONTROLS

For **TELEOP,** optimizing throughput and achieving shorter loops requires as many driver automations as possible protecting the driver's cognitive capacity for high-level decisions during the match.

Here's the list of automations available to the driver during TELEOP:

- Indicate **how many artifacts** the robot currently controls using LEDs. Shut off the intake system after collecting 3 artifacts.
- Go to the near or far **shooting zone** at driver's discretion via shortest route and immediately start shooting
- Naturally, the shooting process itself is **automated** even though it requires precise orchestration of different mechanisms.
- Actively **hold** shooting position/heading to counter opponent alliance's defense play
- Go to the ramp lever and **open the ramp** and optionally collect artifacts rolling down, automatically. Thus, one of the fastest loops involving collecting from the ramp and shooting from the close zone can be achieved by pressing just two buttons on a gamepad without having to actually drive a robot.
- Go to the **parking** position.
- Last but not least, **visually identify** an artifact using a webcam and drive directly towards it and automatically collect.

SOFTWARE & CONTROLS

Solving those during teleop required a software system capable of **dynamically adjusting** to an environment rather than just following an optimized plan, which is often what many teams do during the AUTO.

To solve this, we have developed our **own library** based on the Kotlin programming language with heavy use of **coroutines**, which allows for easy composition of asynchronous code.

An additional layer of software complexity for the team to solve was a requirement to support at least two subsequent iterations of our robot design.

To solve this, we **decoupled** the robot "interface" required for Decode season algorithms from robot implementation. This also allowed for **unit testing** to a certain extent. Furthermore we came up with all configuration parameters being stored in a per-robot **config.properties** file, which can be easily updated to a running robot via an HTTP API (similar to FTC Dashboard and Panels but doesn't require copying and pasting through a user interface).

All of the BearPlatform library code as well as code for all our robot design iteration is open source and is available on GitHub for everyone to use. After season completion we **plan** to:

- Thoroughly review the APIs to make sure it can be conveniently used by anyone, not just us
- Update documentation to reflect the latest code version
- Add more unit tests
- Add more examples
- Set up build and version deployment process
- Set up a documentation website
- Write blog and Reddit posts for community awareness
- Assemble Kotlin and coroutines learning resources so that more team members could easily contribute

MARKETING & OUTREACH

School lobby demonstration

To reach students who would not normally encounter robotics, we built a full FTC field in the school lobby and demonstrated our robot throughout the day, where students from multiple grade levels were able to observe matches, ask questions, interact with team members, and drive the robot.

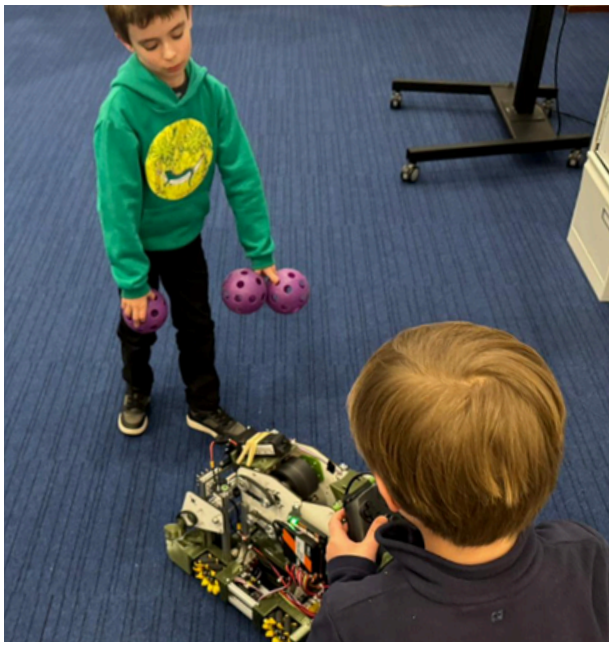
We found that hosting the event in a public space made engineering approachable and sparked interest beyond our existing club members.

School-wide STEM promotion

We shared videos of our robot running and competing on school displays and announcements, which explained FTC and highlighted how students could join our team and engineering club, increasing awareness and lowering the barrier of entry for new members. When we have our robot demonstrations, we also explain our engineering decisions, such as why we chose certain mechanisms, as well as how sensors improve consistency.

Outreach to younger students

We also introduce robotics to even younger students through simple robot demonstrations during classes. By focusing on visual explanations and basic concepts, we make robotics engaging and understandable, helping develop interest in STEM.



SUSTAINABILITY

Our goal is to ensure that our engineering club remains sustainable, inclusive, and that it will continuously improve beyond a single season. We focus on long-term growth by developing members' skills and strengthening our presence within our school community.

Team Development & Knowledge Transfer

To make sure that future teams will not have to start new seasons from scratch, we aim to maintain clear documentation of designs, code, and lessons learned to support future teams, as well as involve younger members directly in building, programming, and testing to ensure hands-on learning. We also encourage mentorship within the team by pairing more experienced members with newer students, which also strengthens collaboration within the club.

Recruitment & Growth

Our plan is to continue hosting school-wide demonstrations and outreach events to attract new members, as most of our team is made up of seniors at the moment. This involves using visible projects and competition footage to spark interest in engineering and robotics, and lowering the barrier of entry by continuing to welcome students with no prior experience, providing them with structured learning sessions.

Education & Skill Building

For the next season, we plan to organize regular club meetings focused on teaching core engineering, programming, and design principles, and introduce CAD, manufacturing, and software concepts early on for members to build strong technical foundations. With this, we aim to help more team members emphasize design and problem-solving over simply building a robot.

Budgeting & Continuity

Currently, our budget is mostly provided by our school with some contribution from parents. While this does cover our needs, we feel that it is very important to have more diverse funding to support a bigger team, multiple robot experiments, and upcoming FTC hardware platform upgrades. We plan to leverage our success this season to attract more sponsors and have already secured one commercial company to cover our FIRST Championship travel expenses.

SUMMARY

Strategy
We concluded that Decode rewards throughput above all else, so we optimized for scoring speed and avoided compromises that reduced cycle efficiency.

Engineering
A strategy-driven engineering process led to three full robot rebuilds, modular software architecture that could support multiple robot iterations, a flexible autonomous system, and TELEOP automations that reduced driver cognitive load and improved consistency.

More than Robots
Most importantly, we learned a lot, and we expanded robotics at ISA through outreach events, stellar competition results, rookie team development, and long-term knowledge transfer to support future Beargineers.

This season, Beargineers did not just build a better robot — we built a stronger engineering process, a stronger team, and a stronger local FTC community.

And we had a lot of fun!
Thank you for your attention

#27628
Beargineers
International School of **Amsterdam**

https://beargineers.nl