



Engineering Notebook



Aeolus

te@m GREECE

National Robotics Team



Eduact

Team Profile	1
Aeolus History & Logo	2
Aeolus	3
Simulator	4
Simulator & Community	5
Methodology	7
Strategy	8
Drivetrain	9
Shooter	10
Intake	13
Lift	15
Software and Automation	18
Mechanisms	19
• Movement	
• Facing a certain Direction	
• Localization	
• Facing the Sink	
• Lift	
• Lift Automation	
Control	23
Final Thoughts	24



tε@m GREECE
National Robotics Team

The Greek National Robotics Team consists of 50 high school students from all over Greece. The group is made up of talented individuals who have excelled in a wide variety of competitions in the field of robotics and STEM in Greece and Europe, like Mathematics Competitions, Physics Competitions, research and innovation competitions, art competitions, short film competitions and virtual entrepreneurship competitions. Most importantly, however, they have stood out for their love to explore new territories, research and promote the spirit of gracious professionalism and cooperation, while being passionate and having fun.

Our team has started its course in May 2018, as an initiative of the Educational - Non-profit Organization “eduACT - Action for Education” and has evolved through the years: new individuals are added annually since 2018. As a result, an educational continuum is created, which leads to the cultivation of the next STEM leaders of our community.

Te@m Greece is a living organism that is active throughout each academic year delivering STEM workshops and presentations, taking part in awareness actions, festivals and expos, contacting local leaders, contributing to our national educational system and spreading the message of FIRST throughout the country. Up to now, we have reached over 4.000 children of all ages and participated in over 200 events! Even the COVID-19 pandemic didn't manage to hold us down! We published online STEM activities, and we delivered distance learning lessons with the support of our national partner, our mentors and various supporting organizations.

The goal of this season is to build a united team that embodies the spirit of FIRST, expand our knowledge of robot design and engineering techniques, to secure the future of the team, keep on inspiring as many children as we can to do science, robotics, programming and engineering and, finally, to offer everyone opportunities in learning and self-development!



How our team worked:

In order for us all to learn and be able to apply our ideas and experiment, our team split into 3 subgroups to build three different robots taking advantage of materials left over from previous years.

The goal is to learn, not only to build a robot!

So, each team created a robot with mechanisms that could do all the missions of this year's playing field. The final robot would be a combination of the most effective mechanisms between all 3 robots, or a final and refined version of the robot that collected the most points.

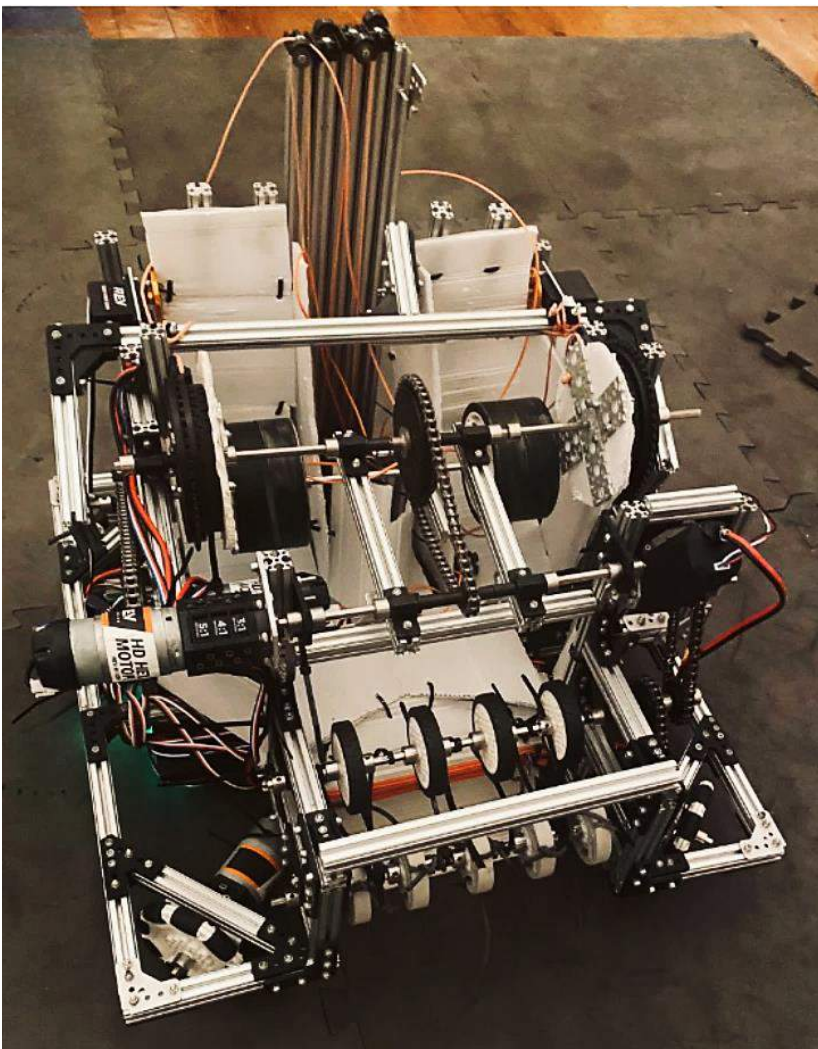
FGC is much more than a robotics competition! It enables young people from all over the world to communicate and learn about new cultures. From 2019, Team Greece decided to name its robots and projects after ancient Greek Mythology and History personalities. In this way we share elements from our culture with all the other teams.



In Greece, when the atmosphere is burdened, we tend to say
"if a wind blows, the sky will get clear".
 This expression was our inspiration for the robot's name.



In Greek mythology, Aeolus was the keeper of the winds and king of the island of Aeolia, one of the abrupt rocky Lipara islands close to Sicily. Diodorus describes him as a just and pious king, who taught sailing and navigation to his subjects, and divined the nature of the winds in a fire. Hence, he becomes described in much mythology as the ruler over the winds, and thus a god.



Mechanical

Angled Servos: The variation in launch angle, caused by two servos at the end of our shooters, allows our driver to shoot with focus either on distance or height.

Plastic Sheet placement: Through the placement of corrugated plastic sheets, we ensure no ball gets stuck inside of the connections between our intake and shooter.

Reel design: The design of our lifting reels allows for maximum torque thanks to the increase in reel diameter.

Programming

Localization: The robot is able to determine its position on the field, using the embedded motor drive encoders.

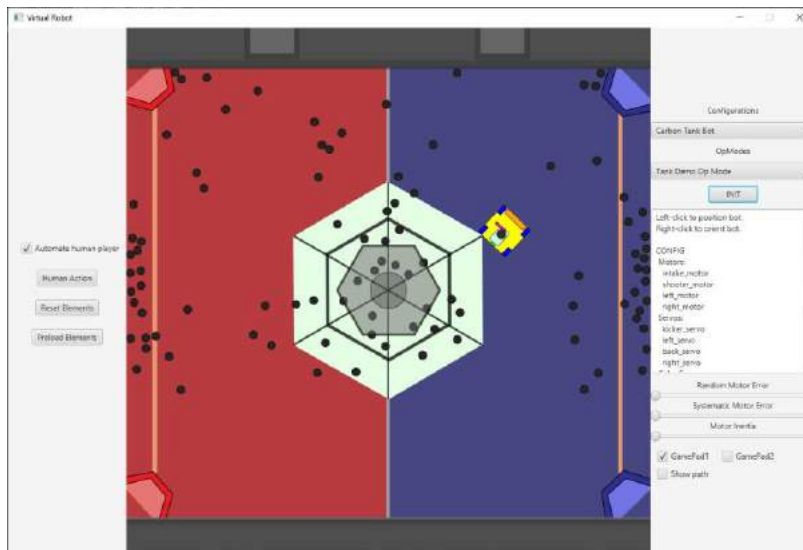
Facing the sink: The robot can turn towards the sink on the center of the field with the press of a button. This simplifies a lot the shooting process for the drivers.

Field Centric Movement: The robot moves relative to the field, which means that the movement is independent of the robot's heading ("Forward" always means "Forward" relative to the field etc).

A problem we face each year during our team preparation is that we usually cannot develop our robot's software as it needs continuous testing during CAD designing and building. After brainstorming, we figured out that we needed a tool to test our code without having a physical robot, so we started developing it based on the `virtual_robot` GitHub Repository.

The project is a JavaFX application developed using the (free) IntelliJ IDEA Community Edition IDE. It is a Carbon capture game adaptation, including the Carbon Capture game elements and two new robot configurations (Carbon Tank Bot and Carbon Omni Bot) with an intake and shooter mechanism, resembling the shooters showcased on the FGC site. There is a test opmode available for the tank robot configuration.

The Simulator contains all the game elements of the Carbon Capture Game, except the `STORAGE_LEVELS`, due to its 2D nature. This means that the player can collect the CARBON, shoot it in the SINK and place it in the compressor for the human player to throw, all complying with physics. Each robot can be thought of as the competition-allowed size.



The field can be thought of as the official field size. The field graphic (currently the Carbon Capture field) is obtained from a bitmap (.bmp) image. The colour sensor detects the field colour beneath the centre of the robot. The field graphic can be easily changed by providing a different .bmp picture in the `virtual_robot.config.Config` class. The .bmp image is the `freight_field648.bmp` file in the `virtual_robot.assets` folder. If a different .bmp image is used, it must be at least as wide and as tall as the field dimensions (currently 648 x 648 pixels to fit on the screen of most laptops). The Config class also allows selection between "real" hardware gamepads and a "virtual gamepad".

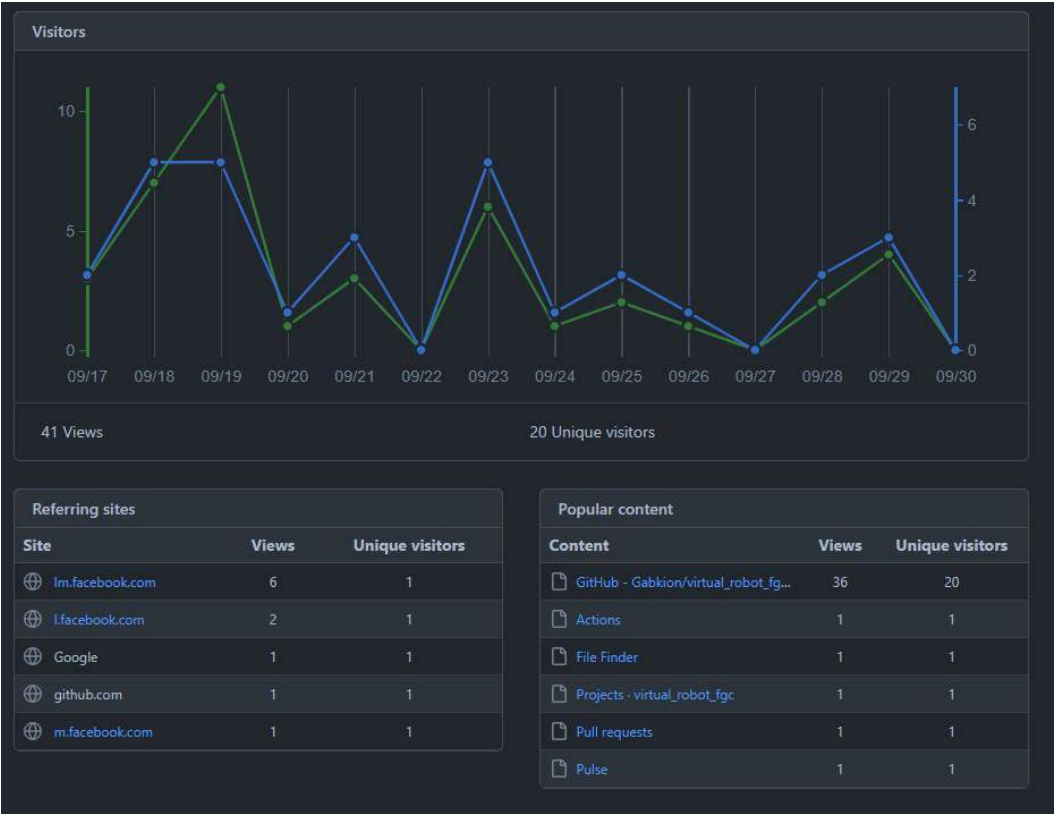
In addition to the robot configurations described above, there is an additional configuration called "ProgrammingBoard". It is a board with several hardware devices attached: DcMotor, Servo, Potentiometer, Touch Sensor, and a Color-Distance Sensor. It also has a BNO055 IMU. The board doesn't move around the field, but it can be rotated (to test the IMU) by dragging the board chassis. Also, an abridged approximation of the FTC SDK is provided.

The OpMode (and therefore LinearOpMode) class in the simulator provides access to:

- A Hardware Map object, which in turn provides access to the DCMotor objects, the BNO055 IMU sensor, distance sensors, the servo, and the colour sensor;
- Two Game Pads (actual hardware gamepads, though there is an option to use a "virtual gamepad" -- see Log of Changes below);
- A Telemetry object.

An approximation of the FTC SDK's ElapsedTime class is provided in the time package.

The project is available on GitHub for anyone to try. We have shared it through social media with all the other teams, receiving great feedback thus far.



About

A 2D robot simulator to help First Global Challenge teams write and test Java programs

Readme

5 stars

1 watching

121 forks

Special thanks to FTC Team Beta 8397 and their mentor Jim Kenny, that supported us and gave our team valuable advice concerning this year's challenge. Their help was invaluable and they were incredible!

"It was very interesting testing our programming skills on it and testing out what would work and what not!"
Team Palestine

"It was helpful testing the program since we did not receive the kit until very lately."
Team Ghana

"We used it, it was really nice! The robot construction was not done yet, so it enabled us go a step further!"
Team Australia



The use of Computer Aided Design played a vital role in our design methodology and the creation of our robot.

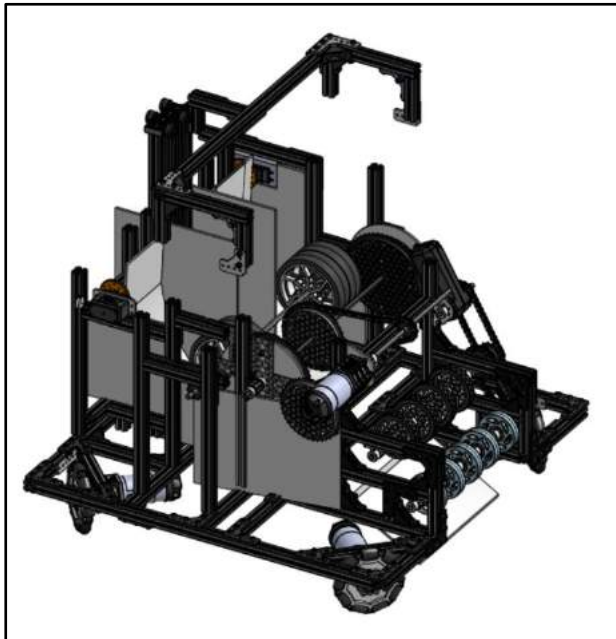
Because of our country's location we received our kits at the end of August.

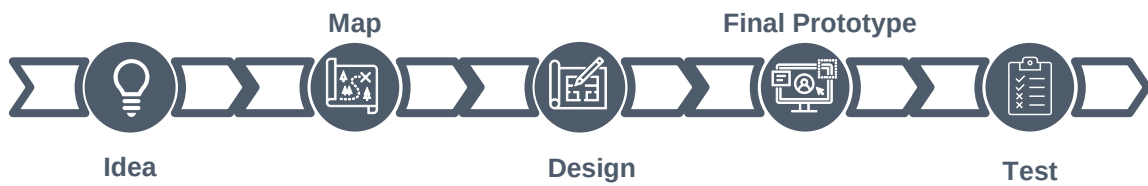
This large delay caused numerous problems for us as we would have been unable to begin creating our robot before the kit's arrival.



CAD allowed us to work from long distances, from anywhere with an internet connection, and ensured that our designs would be feasible before we attempted to create them with the limited materials we had in our possession.

Finally, thanks to the virtual design of our robot, we were given the opportunity to quickly prototype many different mechanisms, and, through analysing their different characteristics and traits, choose the one most efficient for its purpose.





Idea

To design our robot, we firstly brainstormed and thought of possible ways to tackle this year's challenge. At this stage, we recorded any idea a member would have; the crazier the better!

Map

After brainstorming, we designed our ideas on paper, in order to better visualize them. Then, we discussed on what could work or not, based on the advantages and disadvantages of each idea. At this stage, we eliminated the ideas that had the most disadvantages and progressed to the designing stage.

Design

We then designed the remaining ideas on CAD software. This process was valuable, as it revealed the mechanical flaws of some designs, and gave us a better understanding on how the final prototype should be. We continued redesigning the mechanisms until something worked.

Final Prototype – Test

We then manufactured the best designs and put them to the test. The one that worked best was kept as the final prototype and was refined even further.



Lift Pull

$5 \times 4 \times 3 \text{ gear box} = 60:1$
 $1:4 \text{ gear ratio}$
 $100 \rightarrow \frac{100}{4} = 25 \text{ rpm}$

$6000 \rightarrow \frac{6000}{60} = 100 \text{ rpm}$

$L = 2\pi r = 2\pi s = 11\pi \approx 34.5 \text{ cm}$
 $25 \text{ rpm} \times 34.5 \text{ cm} = 862.5 \text{ cm per min}$
 $\frac{862.5 \text{ cm}}{60 \text{ sec}} = 14.375 \text{ cm per second}$

Intake

3	1
0	2

Lift

$h = 171$

$4.5 - 3.5 = 1.0 \text{ cm of rope already extended}$
 $171 - 10 = 161 \text{ cm rope}$
 $\frac{161}{14.375} = 11.2 \text{ sec per retraction}$

CON. HUB

UP Right Servo 0

UP Left Servo 1

Down Right Servo 2

Down Left Servo 3

Lift Servo 4

EX. HUB

Intake 0

Shooter 1

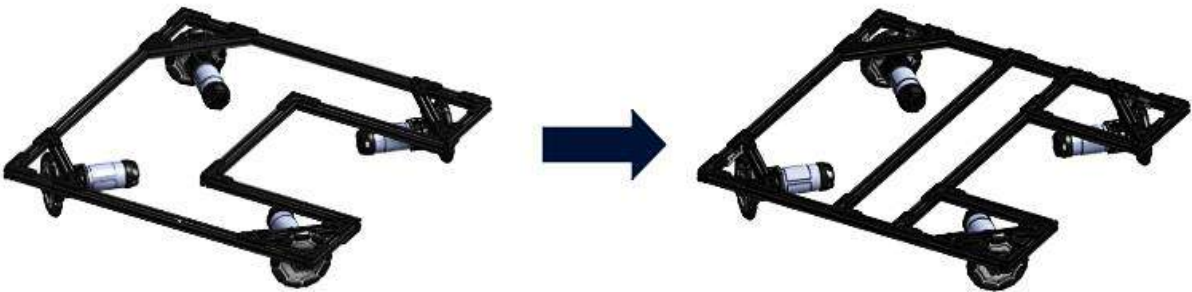
Lift 2

Reels 3

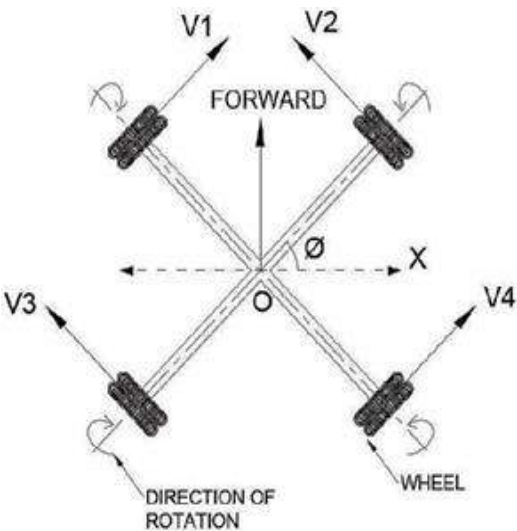
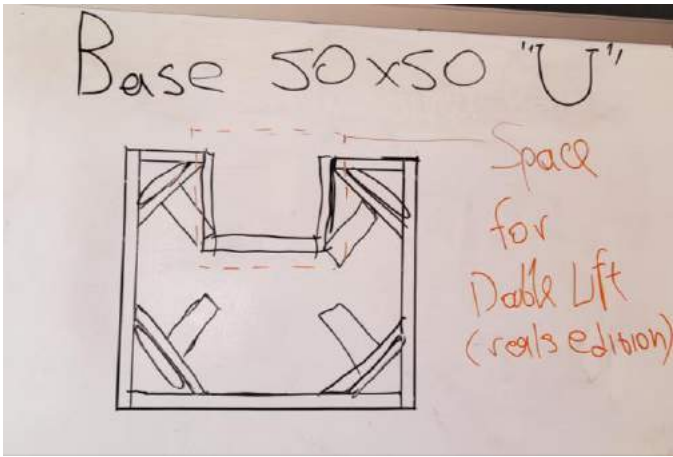
Our strategy aims to be efficient yet simple. When the game begins our robot will start by collecting as many balls as possible and storing them within itself. After the number of collected balls is adequate, the robot will activate its field-centricity program and after the firing flaps are adjusted and its shooter speeds up, it will begin firing. This process of ball collection, charging and firing is to be repeated until the time comes for the robots to begin hanging. Then, we will place the colour ball, the robot's lift will begin to rise, and the robot will start pulling itself up.



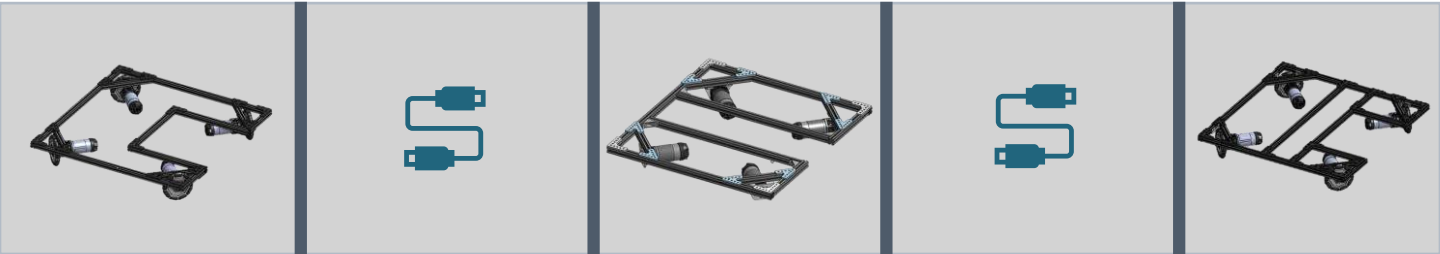
One of the first base designs we tried was a 35cm-by-35cm structure with omni wheels at a 45-degree angle to the sides (45°). Despite the strength and rigidity of the structure, it was too small for the mechanisms we intended to use. Therefore, we would have to put our mechanisms at the perimeter of our base, protruding outwards, and as a result they would be unstable and difficult to brace to the main structure. After we had agreed to make a robot 50cm-by-50cm, we thought of a U-shaped base. In essence, it would have a gap on one side so that the lift with the reels could fit. This solution was rejected because the other mechanisms did not fit since we had decided to put a double shooter.



We chose to implement an omni-wheel based drivetrain since our team has implemented it before and more importantly, it offers the driver the ability to move in any direction without changing the robot's heading, something vital for this year's challenge. Our drivetrain's dimensions are 50x50 and decided that these would be them, since we saw that this would work better for us including all the mechanisms that we wanted for our robot to have.



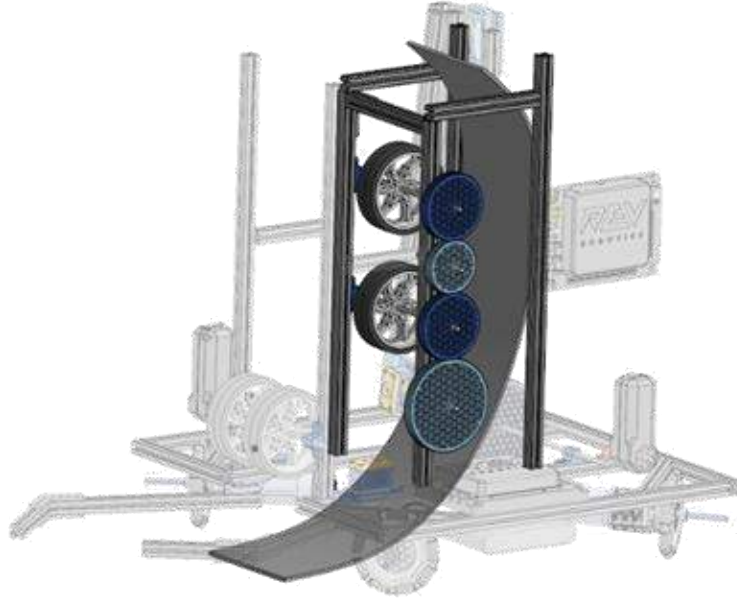
The team decided to use whiteboards to design and keep a digital archive of photos in their effort to reduce their environmental footprint as much as possible.



1st Idea:

The shooting mechanism created was not efficient enough and did not have enough power for the carbon to be shoot.

Also, the ideal angle of the plastic could not be defined so the carbon could not reach the proper height.

**2nd Idea:**

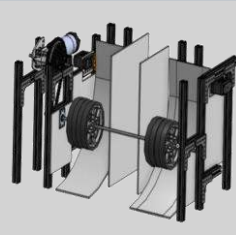
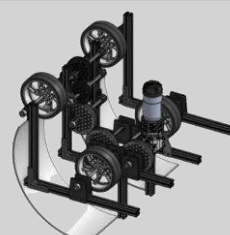
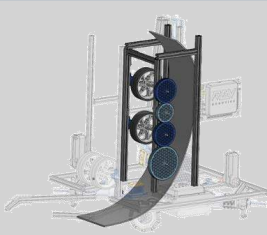
10

The shooter was a 2in1 mechanism along with the intake, so the motor used would lose a lot of power both collecting the carbon and then shooting it. Also, the ideal angle of the plastic could not be defined so the carbon could not reach the proper height.

Did you know..

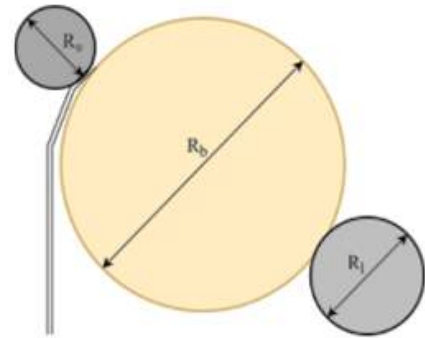
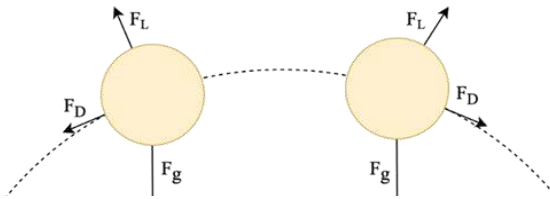
The shooter is loosely based on the ancient Greek technology of the crossbow. At first, the Greek catapult hurled arrows in great distances (crossbow), before spears and stones began being used. It was invented in Sicily in the late 390 B.C. , when the tyrant Dionysius the Elder of Syracuse called Greek mechanics to build a war machine for his campaign against Carthage. It was complete within an hour for the surprise attack of the Carthaginians on Syracuse. The enemy fled from there the arrows that attacked form every direction imaginable. The king of Sparta, Archidamus, was also left speechless at a demonstration of the catapult brought to him from Sicily, being literally decapitated. Given that in Ancient Greece motors did not exist, the crossbow used potential energy to shoot arrows instead of kinetic that is used by Aeolus.

[\[http://kotsanas.com/\]](http://kotsanas.com/)



3rd Idea:

The first idea we had concerning the shooter was with the two wheels in front so that it pulls in the ball and then the other two wheels in order to launch them by spinning in opposite directions. The gear ratio for the first pair of wheels was 2.7 intended to drive the ball and the second pair of wheels was 8.3 intended to launch them at high speed. We rejected this idea because the motor was slow. In addition, there was a difficulty in finding the inclination and also, we could not adjust it. We later realized that the second wheel was obsolete. The current design utilizes only one wheel spinning at approximately 6,000 rpm and a servo motor to modify the ball's angle of release.



Checking and adjusting the shooter

The workshop that hosts the team has a low ceiling. Until a suitable space was found, we made the necessary checks and adjustments on the pavement. Every time we lowered the robot, passers-by would slowly gather and ask what we were doing. Controlling the robot became an occasion to bring ordinary people closer to our team.



To precisely calculate the ball's trajectory, we needed to consider the three forces that acted upon the ball – lift, drag, gravity – and the ball's spin. For that we found a paper, a former FRC team developed, concerning flywheel-based ball shooters similar to our own.

After finding the desired ball trajectory we then calculated the initial speed, height and angle of release required for the ball to achieve said trajectory. We then modified our design to incorporate all those variables into our shooting mechanism.

$$\omega_{u_{max}} = \omega_{l_{max}} = C_r \cdot \frac{6380}{60} (2\pi)$$

$$v_t = \frac{\omega_u R_u + \omega_l R_l}{2}$$

$$s = \frac{\omega_b}{2\pi} = \frac{\omega_l R_l - \omega_u R_u}{2\pi R_b}$$

$$a_g = \frac{F_g}{m} = 9.8 \text{ m/s}^2$$

$$a_l = \frac{F_l}{m} = \frac{C_L \frac{4}{3} (4\pi^2 r^3 s \rho v)}{m}$$

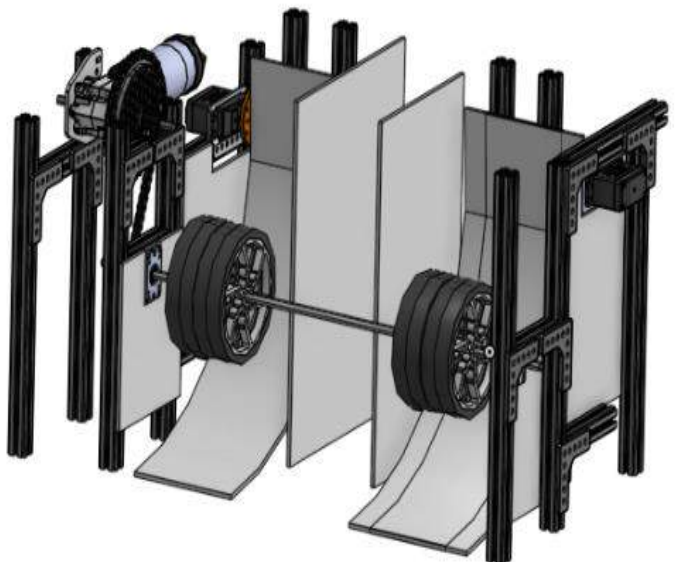
$$a_d = \frac{F_d}{m} = \frac{\frac{c}{2} A v^2 \rho}{m}$$

12

The next shooter we thought of making was a double shooter using an ultra planetary motor along with a 90degree in order to increase speed. The ball passed through the low wheel and when it found the other wheel it would give impetus and the ball passed between the two wheels. The gear ratio for the lower wheels was 6:1(so that he could pull in the balls and drive them at speed) and for the top wheels it was 3:1 (so that it makes curve).We rejected this idea because during the test the balls did not go far as we have hoped and did not get the necessary angle. In addition, the gears were gradually being destroyed from the high motor speed and friction.

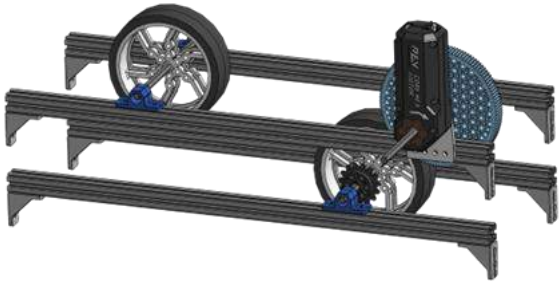
So, we decided that we should change it once more.

After this alteration and the use of a chain both upper wheels came out. The carbon passes through the wheels that are inserted in both shooters through the intake system and in contact with the corrugated plastic sheet they are launched. We positioned servo motors at the top of the corrugated plastic sheet for easy adjustment of the inclination and to assure that in any case the carbon would reach the sink. The gear ratio is 4:1.



1st Idea:

After brainstorming the team started building an intake. This intake system could only gather one ball at a time and its storage space could save only 2-3 balls. Though, it would spend too much time trying to gather the Carbon, and then shooting it. Also, there was too little friction between the polycarbonate plastic and the wheels resulting in the motor losing power.



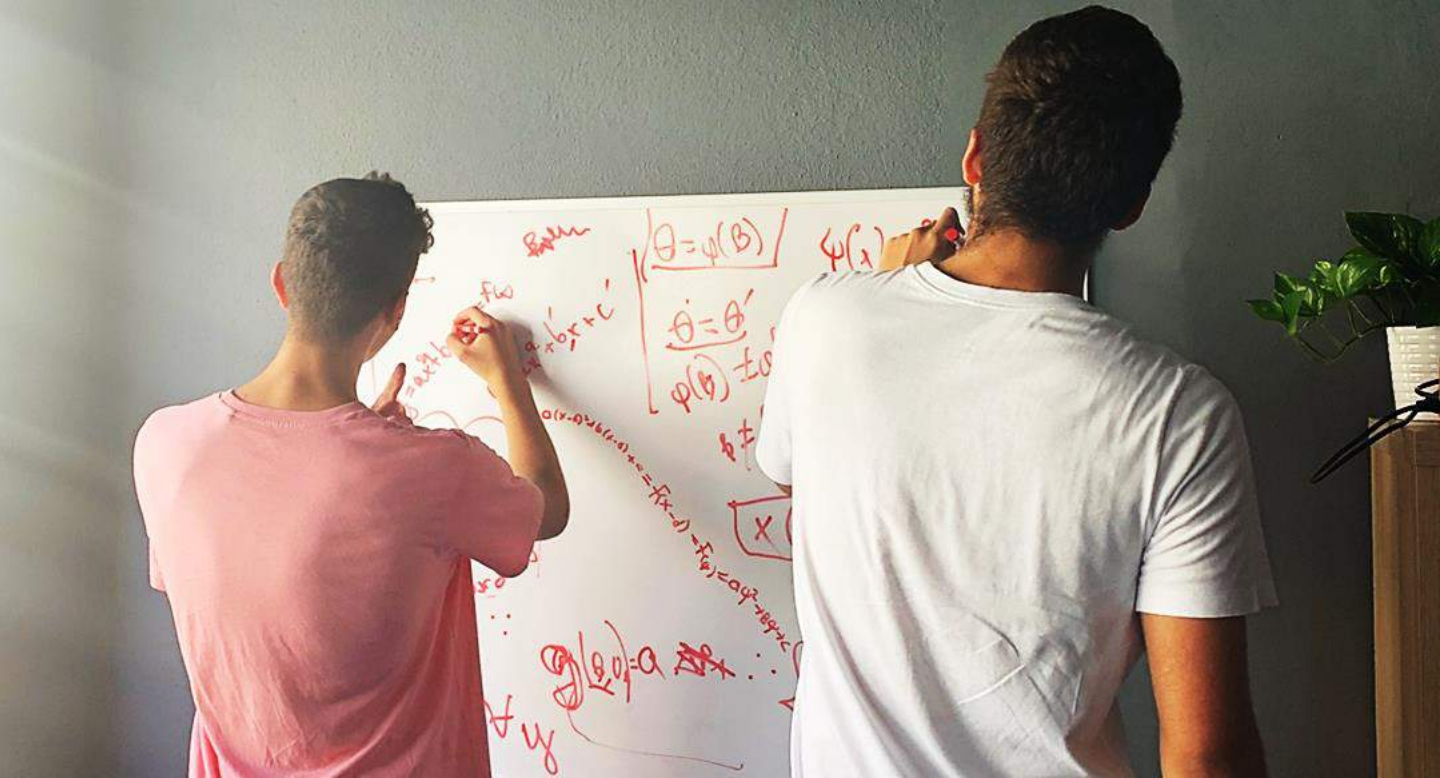
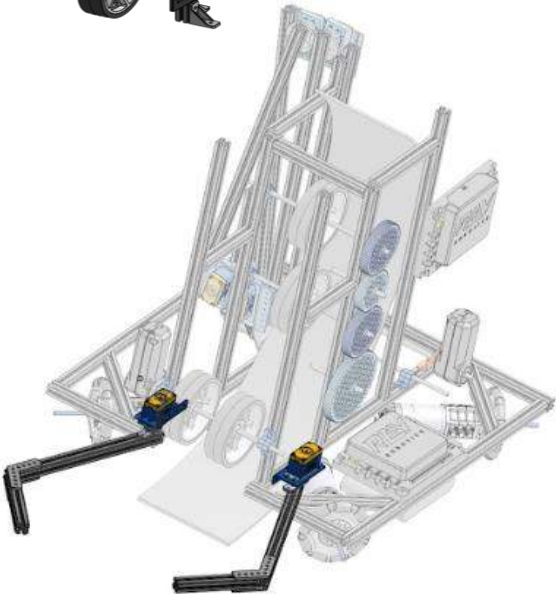
Did you know...

The concept behind the intake part of our robot can be considered the opposite of the hydro mill. The watermill is the first work-producing machine made by man using a natural, mild and renewable source of energy. With the force created by the flow of water and the help of the wheel, other tasks could be achieved. On the other hand, for our intake system, the motor provides the wheels with angular velocity that causes it them to rotate completing tasks, like the capture of balls.

[\[http://kotsanas.com/\]](http://kotsanas.com/)

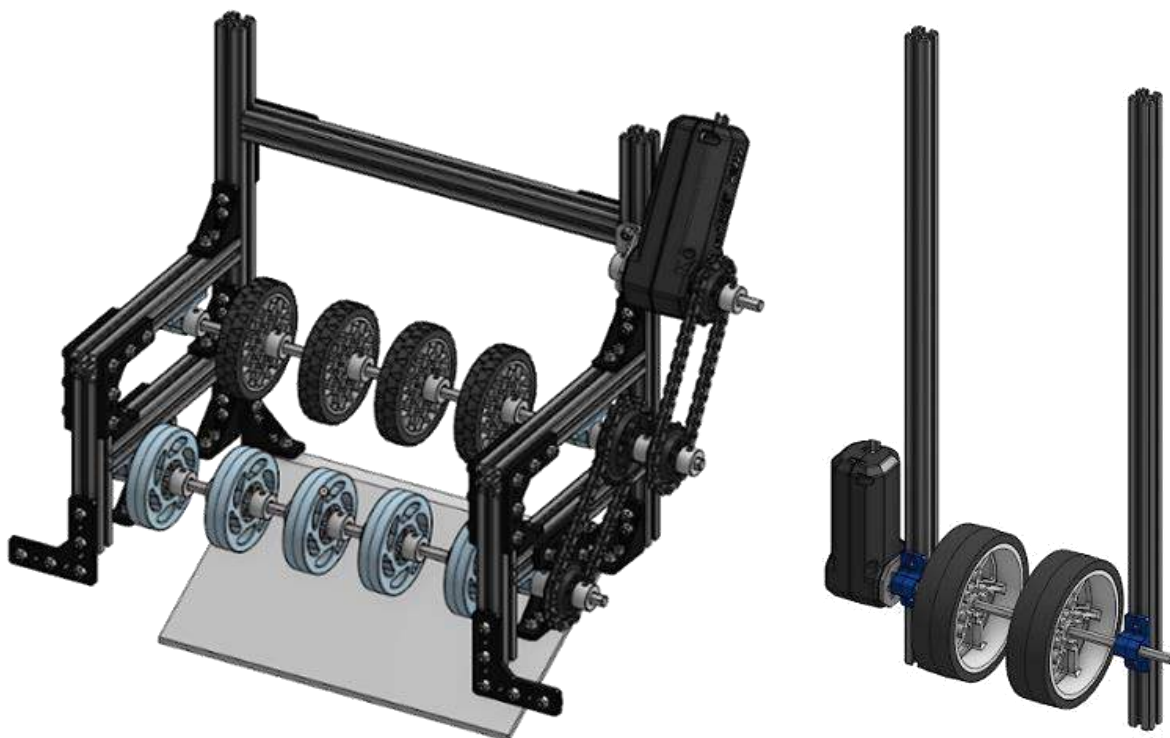
2nd Idea:

The intake would gather the carbon very slowly and only one ball at a time.



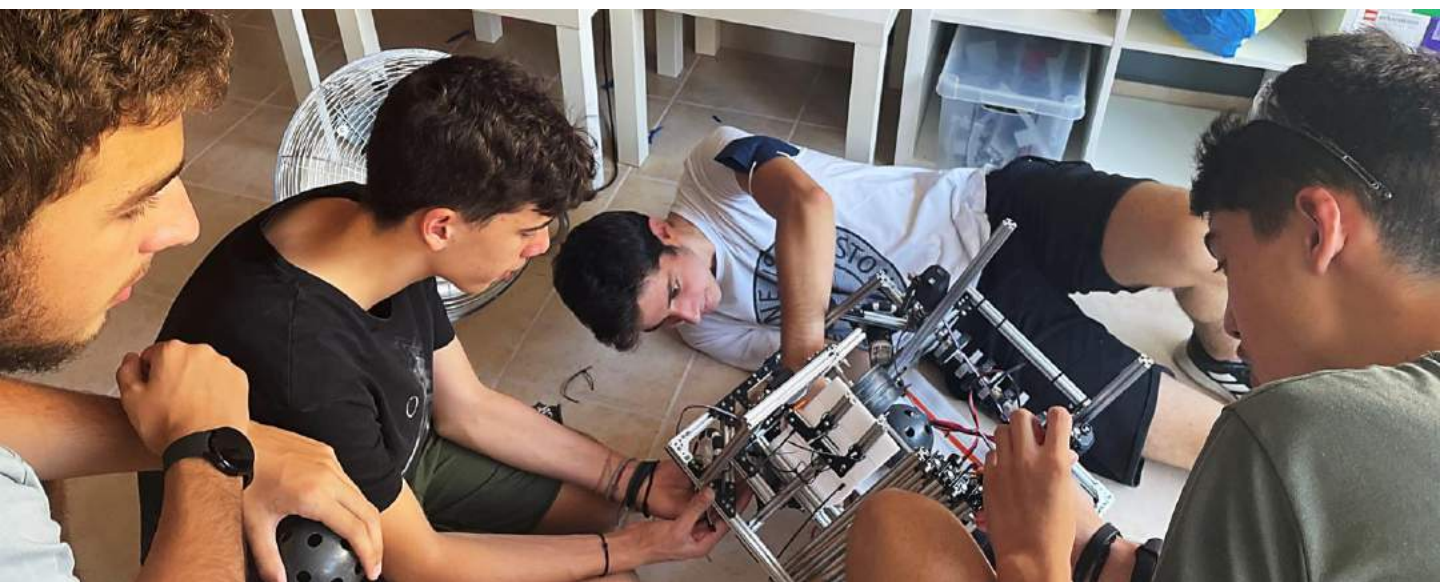
3rd Idea:

For our intake we decided to use the rubber wheels as they are easily compressible – unlike the carbon balls – and thus offer more friction. We also used zip ties to help lead the carbon balls to a polycarbonate ramp – inclining along both the x and y axes – which leads them to the shooting mechanism. A few changes were made considering the height of the upper wheels, adjusting how much friction they would have with the carbon. As far as the ramp is concerned, we changed its angle to find which one would work better for our intake system. Finally, we added a slot cover that offered extra grip to the carbon and helped them reach both of our shooters.



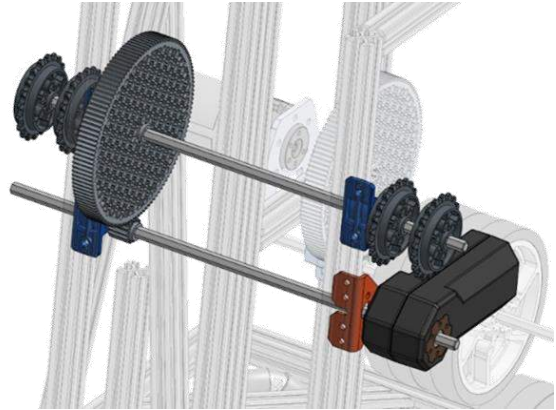
We decided to create a mechanism that will be used to stop the carbon from being directly shot when it is collected that will be placed between the area of the intake and the shooter.

More particularly this will consist of two servo motors, moving a plastic 90-degree bracket. This is a revolutionary feature of ours since it enables us to control when the carbon will be collected and then inserted to the shooter by moving the bracket downwards through the controller.



1st Idea:

This lift worked with servo motors, so it was not stable and powerful enough, so the robot could not be lifted after the hook reached the proper height. Also, for its expansion only one servo at a time was used, so it was extremely slow.



Lift

2nd Idea:

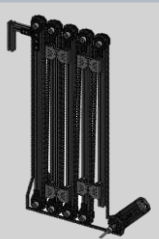
The lift was extremely slow and not powerful. When the robot would hang on the desired level and started climbing, it would come to pieces.



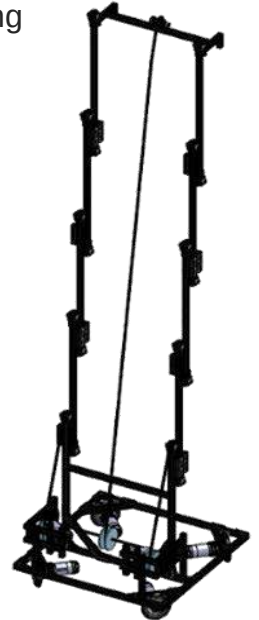
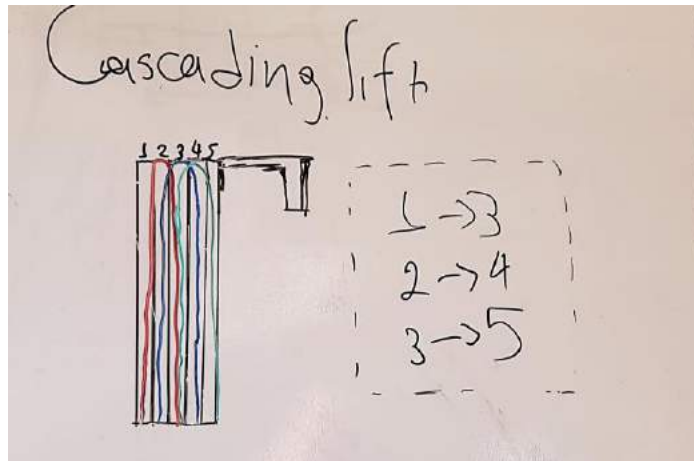
Did you Know..

The lift of our robot draws inspiration from the Dikolos lifting machine of Heron of Alexandria. It consisted of two long (slightly inclined from the vertical) beams and a horizontal U-shaped one. Lifting was achieved by using a “five-split” with three “pulleys” on the (upper) fixed “pulley” and two on the (lower) free one. The machine was usually mounted on pulleys for its horizontal movement. The pulling force of the load-lifting rope came from animals or a powerful hand-held ground winch. The pulling force in Aeolus is provided by a motor.

[\[http://kotsanas.com/\]](http://kotsanas.com/)

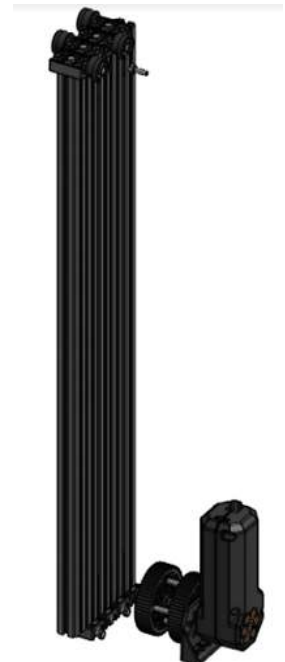


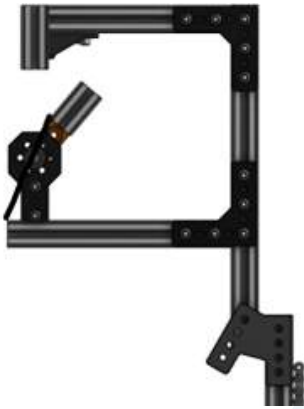
Before building our initial lift, our team brainstormed various lifts to be created. Among them there were scissor lifts, lifts with rotating joints, continuous lifts and cascade lifts.



Another idea was very similar with the previous chain lift design but has some important differences. First, we have a set of 2 lifts parallel to each other connected at the top with an extrusion. Another difference is that we used spools, one to pull the chain to extend the lift and one to pull the robot up and return the lift to the default position. A problem we experienced is that we were using the entirety of the chain provided in the kit, making us unable to use chain in the rest of the robot mechanisms. Finally, as the previous design, the Double-sided Sliders introduced flexing in the lift and combined with the chain tension caused it to lean forward.

After creating a continuous lift and doing several tests with a continuous, we saw that it was not stable and did not have enough power even though changes were made. Also, it was not fast enough so we minimized the possibilities of reaching the level we aimed at reaching, before our opponents. This lift consisted of five 42cm extrusions parallel to each other connected with Double-sided Sliders. The chain follows a standard continuous lift path and completes a loop under the lift. For the extension of the lift, we used a driver sprocket, connected to a Core Hex motor, and a free spinning sprocket to increase the tension and contact of the chain with the driver sprocket. The reason we scrapped this idea is because the connection of the 42cm extrusions with the Double-sided Sliders introduced flexing in the lift and combined with the chain tension caused it to lean forward. Finally, we experienced problems with the chain skipping the driver sprocket and occasionally the chain came in contact with itself adding more friction to the system.



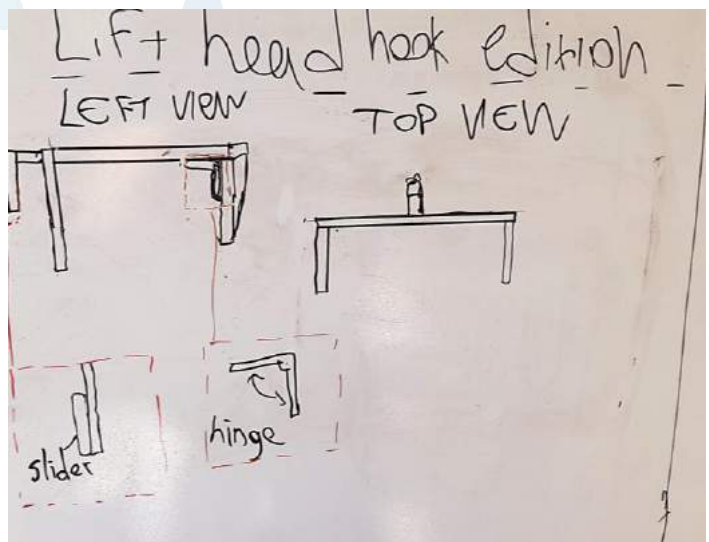
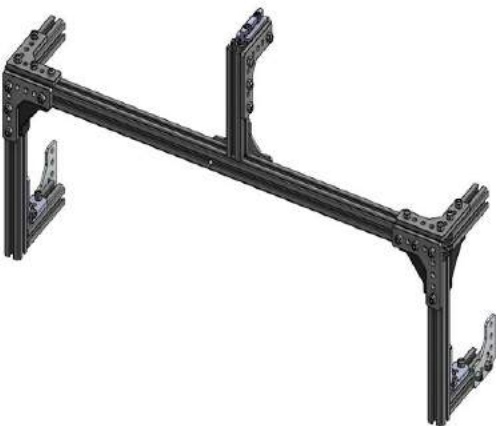


Additionally, we decided to lift the robot with a separate mechanism from the lift, thus we thought of a hook. The first hook we had created was a simple gripper. We used extrusions to make 2 parallel L shapes to grab the bars. The problem we faced was that it wouldn't grab the bar if they were at the same height and that it would occasionally slip and get off the bar. Before our final hook was created for safety that the mechanism that will come in contact with the hanger must lock. So, we created with the parts from rev a mechanism similar to a carabiner. Unfortunately, it did not fit properly to the hanger because of its length. Changing its length made it efficient enough and decided to keep it as our final version.



17

More specifically it has 2 clips that lock on the hanger. Furthermore, to be able to get out the lift from the hook we have put a slider. The moment that the lift leaves the hook on the hanger, the reels, which are attached to the hook with a rope, begin to pick up the rope. This has the effect of lifting the robot. After the hook's rope is extended and unfolded from a reel system, it is then wrapped again around it so that the robot starts climbing and reaching the ideal height.



This year, the programming department of the team worked towards creating a pleasant programming and controlling experience for the whole community. We made a 2D robot simulator and released it for everybody to try, and we also made sure that our robot would be as well automated as possible.



FIRST Global Challenge Team Greece



8 Αυγ • 🌐

◆ Team Greece presents to 🥁🥁🥁 the 2D Simulator for this year's playing field! It is a 2D simulator to help beginning Java... Δείτε περισσότερα



27

1 σχόλιο 2 κοινοποιήσεις



Μου αρέσει!



Σχολιάστε



Κοινοποίηση

Our robot consists of 4 main Mechanisms: the Chassis, the Intake, the Shooter and the Lift. All mechanisms were programmed to function automatically, with the capability of manual use if something goes wrong.

Every mechanism is represented by its own class, offering a modular coding experience. This way it is easier to write and debug code for each mechanism separately, and then combine them on the final OpMode File.

Holonomic Chassis

Movement

As described in the Robot description, our robot's chassis consists of 4 wheels attached to motors, with a 45-degree angle. This enables the robot to move along 5 axes of freedom (4 translational, 1 directional).

In order for the robot to move, we must give the correct power to the motors, given by the next formulas:

$$flPower = (+x + y - turn)$$

$$frPower = (-x + y + turn)$$

$$blPower = (-x + y - turn)$$

$$brPower = (+x + y + turn)$$

x, y were are the values given from the Left Joystick, and turn is the x value of the Right Joystick.

(Note: Front Right and Back Right motors' directions are reversed)

This is because we want the motors to move in opposite directions to translate, and in the same directions to turn.

Facing a certain Direction

The robot's orientation is directly controlled by a variable called *target_angle*. This variable is subtracted in every loop tick from the current angle given by the IMU present in the robot. It is passed into the PID controller controlling the robot's angle. So, to face a specific direction, we simply need to set the Target Angle value equal to the direction we need to face (e.g., Forward: 0° deg, Left: 270° deg.)

Movement

The moving algorithm is improved even further, using a PID controller to counteract the excess turning movement caused by the robot's inertia. After the Driver has stopped turning the robot, the algorithm calculates whether the turn was greater than a turn tolerance. If yes, when enough time has passed from the turn, PID correction kicks in, and the robot corrects the excess turning. If not, it corrects it right away.



Localization

Localization

Another feature of our robot is knowing its position anytime on the field. It comes in handy for a variety of features. This means that we know the exact robot coordinates on the field any time during the game

First, let's define the local coordinate system: Yr is toward the front, and Xr is toward the left. And the robot has four wheels: back left (bl), front left (fl), front right (fr), and back right (br). And let's define the robot's heading: the heading is zero when the robot's y-axis (i.e., forward axis) is aligned with the field's y-axis. And the heading becomes increasingly positive (or in some cases, less negative) when it is turning to the right.

For a given brief iteration of your odometry cycle, for a given wheel, let w equal the distance that the surface of that wheel has moved as a result of rotation around its main axis. We are keeping track of ticks on each motor so that we know how many incremental ticks have happened since the previous iteration in a given iteration. Once we have calculated the incremental ticks, we save the current total ticks for the next iteration.

$$wbl = (back_left_incremental_ticks / ticks_per_rotation) * \pi * wheel_diameter;$$

20

The small increments of robot motion in its own coordinate system (deltaXr, deltaYr) will then be:

$$\begin{aligned} \mathit{deltaXr} &= -0.25 * \sqrt{2} * (wbl - wfl + wfr - wbr) \\ \mathit{deltaYr} &= 0.25 * \sqrt{2} * (wbl + wfl + wfr + wbr) \end{aligned}$$

We are also keeping track of the robot's heading, so we have the robot's heading (from the end of the previous iteration), and the new_heading (which we have obtained from the IMU during this iteration. Then the AVERAGE HEADING during the current iteration is:

$$\mathit{avg_heading} = \mathit{heading} + 0.5 * \mathit{norm}(\mathit{new_heading} - \mathit{heading})$$

We need to normalize the difference between new and old heading to the -pi to +pi radians range.

Now, the small increment of linear motion can be converted to the FIELD COORDINATE SYSTEM (deltaXf, deltaYf), as follows:

$$\begin{aligned} \mathit{deltaXf} &= \mathit{deltaXr} * \cos(\mathit{avg_heading}) - \mathit{deltaYr} * \sin(\mathit{avg_heading}) \\ \mathit{deltaYf} &= \mathit{deltaXr} * \sin(\mathit{avg_heading}) + \mathit{deltaYr} * \cos(\mathit{avg_heading}) \end{aligned}$$

So now we are ready to update the pose:

$$\begin{aligned} Xf &= Xf + \mathit{deltaXf} \\ Yf &= Yf + \mathit{deltaYf}; \\ \mathit{heading} &= \mathit{new_heading} \end{aligned}$$

Facing the Sink

Knowing the exact position of our robot enables us to face the Sink on the press of a button. We simply set the `target_angle` variable to the inverse tangent of our coordinates, plus an offset angle. If the robot is at the right side of the field, the offset is 90° , else it's 270° .

Facing the Sink

Shooter

Our Robot's shooter consists of one motor responsible for the actual shooting of the balls, as well as two servo mechanisms placed on the top part of the shooter's hood. These servos control the angle of the hood's exit-point, which basically is the angle of the ball's velocity.

Setting the Shooter's RPM

The motor's RPM is controlled by a PID controller. Firstly, we calculate the current RPM:

$$rpm = \Delta Rotations / \Delta Minutes.$$

Then, we calculate the error between the current RPM and the target RPM, and this is passed in the PID Controller. Then, the result of the PID is converted to Power:

$$power = resultRPM / maxRPM$$

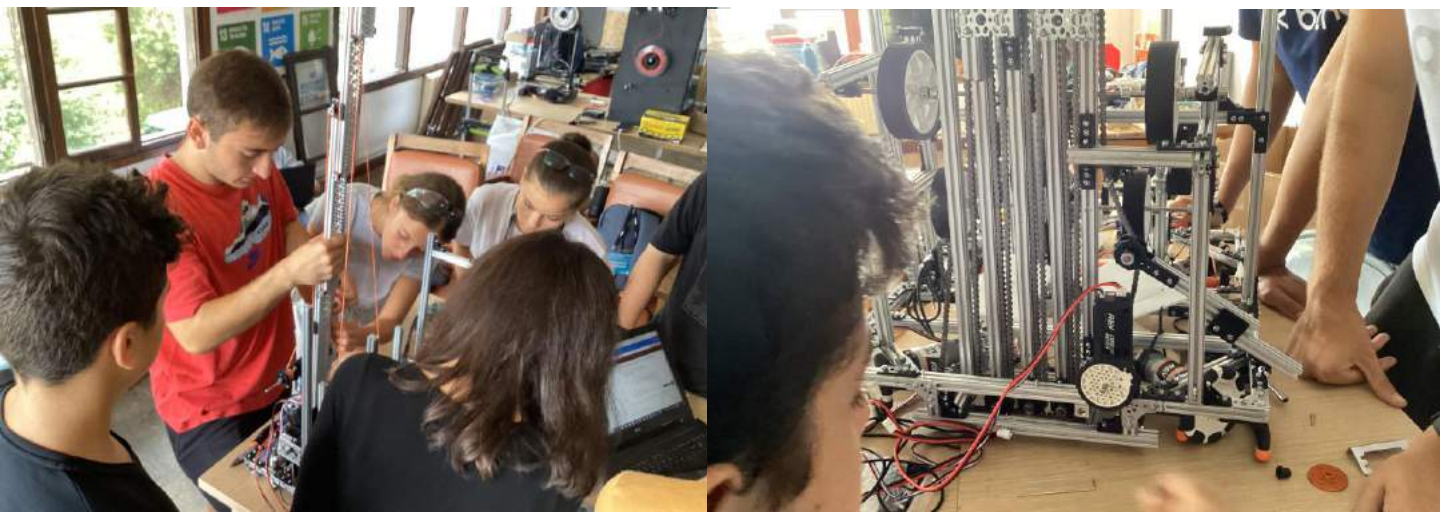
and it is passed to the motor.

Changing the Hood's angle

Based on our location on the field, the robot is able to adjust the hood's angle using the servos. If we want range, the servos will decrease the angle, for the ball to go further. If not, the angle will increase.

Lift

The robot's Lift consists of two main mechanisms: one that extends and leaves a hook on the `STORAGE_LEVEL`, and another that lifts up the robot from the hook.



The lift's behaviour is controlled by a Finite State Machine or FSM. It's a state machine, with a finite number of states. It can be in one state at a time and can transition to a different state once something happens.

The goal is to be able to push a button, and then the bot will:

- Extend the lift to the specified level
- When extended, and enough time has passed start retracting

If the drivers press a specific button, we will stop executing the actions above as a failsafe - in case the robot is breaking somehow, and the drivers need to take manual control. All the while, the drivers should still be able to control our drivetrain so we can make adjustments.

```
public void setPower(double x, double y, double turn, double power, boolean debug){
    double flPower = (x + y - turn) * power;
    double frPower = (-x + y + turn) * power;
    double blPower = (-x + y - turn) * power;
    double brPower = (x + y + turn) * power;

    double maxPower = Math.max(1.0, abs(blPower));
    maxPower = Math.max(maxPower, abs(flPower));
    maxPower = Math.max(maxPower, abs(frPower));
    maxPower = Math.max(maxPower, abs(brPower));

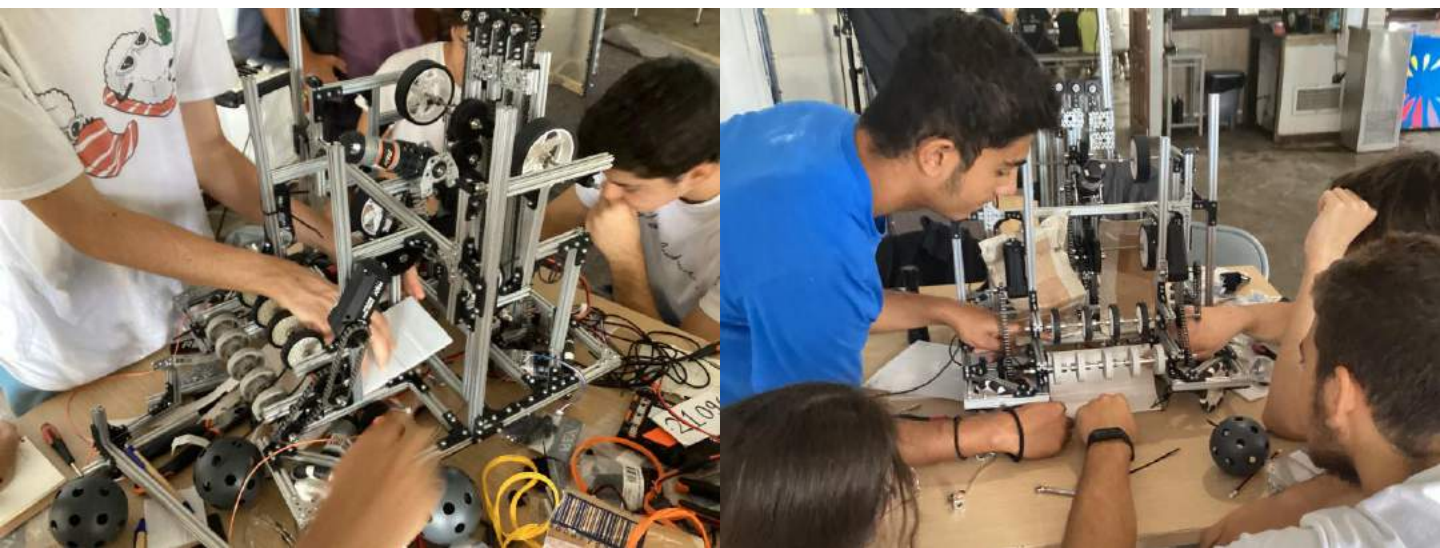
    if (maxPower > 1) {
        flPower = flPower / maxPower;
        frPower = frPower / maxPower;
        blPower = blPower / maxPower;
        brPower = brPower / maxPower;
    }

    this.frontLeft.setPower(flPower);
    this.frontRight.setPower(frPower);
    this.backLeft.setPower(blPower);
    this.backRight.setPower(brPower);

    if (debug) {...}
}
```

Intake

Programming the robot's intake was relatively simple, as it consists of only one motor, that either can be on or off. So, the intake class contains 3 methods (intake_on, intake_off, intake_reverse) that set the according power to the motors. These methods are accessed by a Finite State Machine.



For this year, our aim was to have the best controls possible, to make it easier for the drivers.

Controller Class

We made a Controller class, as well as Controller.Button and Controller.Joystick classes, representing the controller we plugged into the Driver Station. Each button can be used in 4 different ways:

- Normally,
- As a toggle,
- Raise 'True' on press,
- Raise 'True' on release

This way a single button can be purposed to do a plethora of actions, and mainly due to the toggle function, we need fewer buttons per action.

The Joystick class to invert and/or shift the values of the Joysticks based on a given angle. The shifting algorithm is as follows:

$$x_{shift} = x * \sin(\theta)$$

$$y_{shift} = y * \cos(\theta)$$

Where θ corresponds to the current heading of the robot.

Field Centric Movement

Using this shifting algorithm the robot can move independently from its orientation, relative to the field. This means that the joystick movements correspond to the direction we want the robot to move, based in the field.



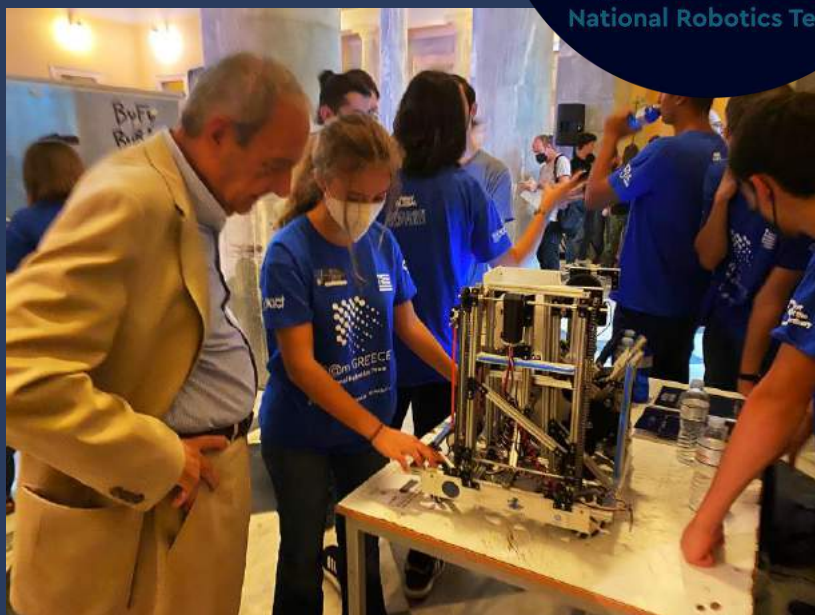
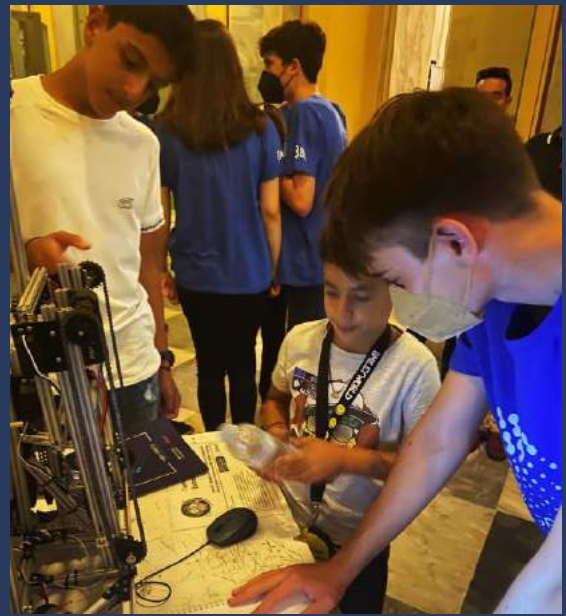
Preparation for this year's FIRST Global Challenge, Carbon Capture, led our team to improve as a whole. We got to research many different topics, expand our knowledge and raise even more our awareness of ourselves and our communities.

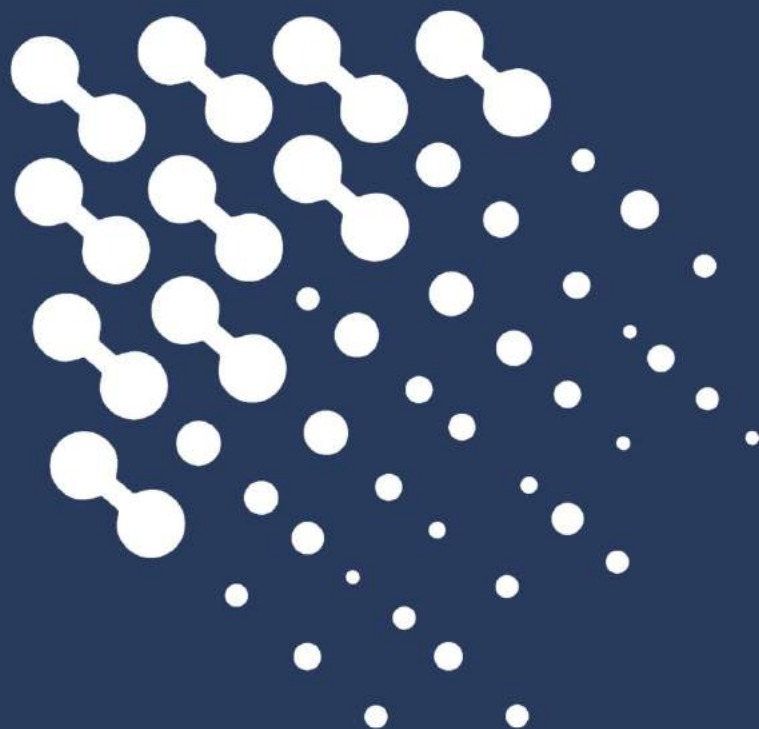
The competition also gave us the opportunity to meet many different people from countries all around the world, which, despite numerous cultural and religious differences, shared common goals, aspirations, and interests with us. We were all given the privilege of experiencing other countries' cultures without having to travel across the world to do so.

Our team has come to the realization that its main purpose is not solely to compete, but to emphasize on engaging more people in STEM throughout our communities. This year's theme made us aware of how crucial it is for us to conserve our planet's resources and valuable ecosystems.

Having come together in our team we formed a lot of new friendships through collaboration which we hope to expand on in the coming years. We are most excited to be able to represent our country in FIRST GLOBAL Challenge Robotics Competition and even more excited to continue working and spreading the message of STEM in the coming years!







te@m GREECE

National Robotics Team



Eduact

