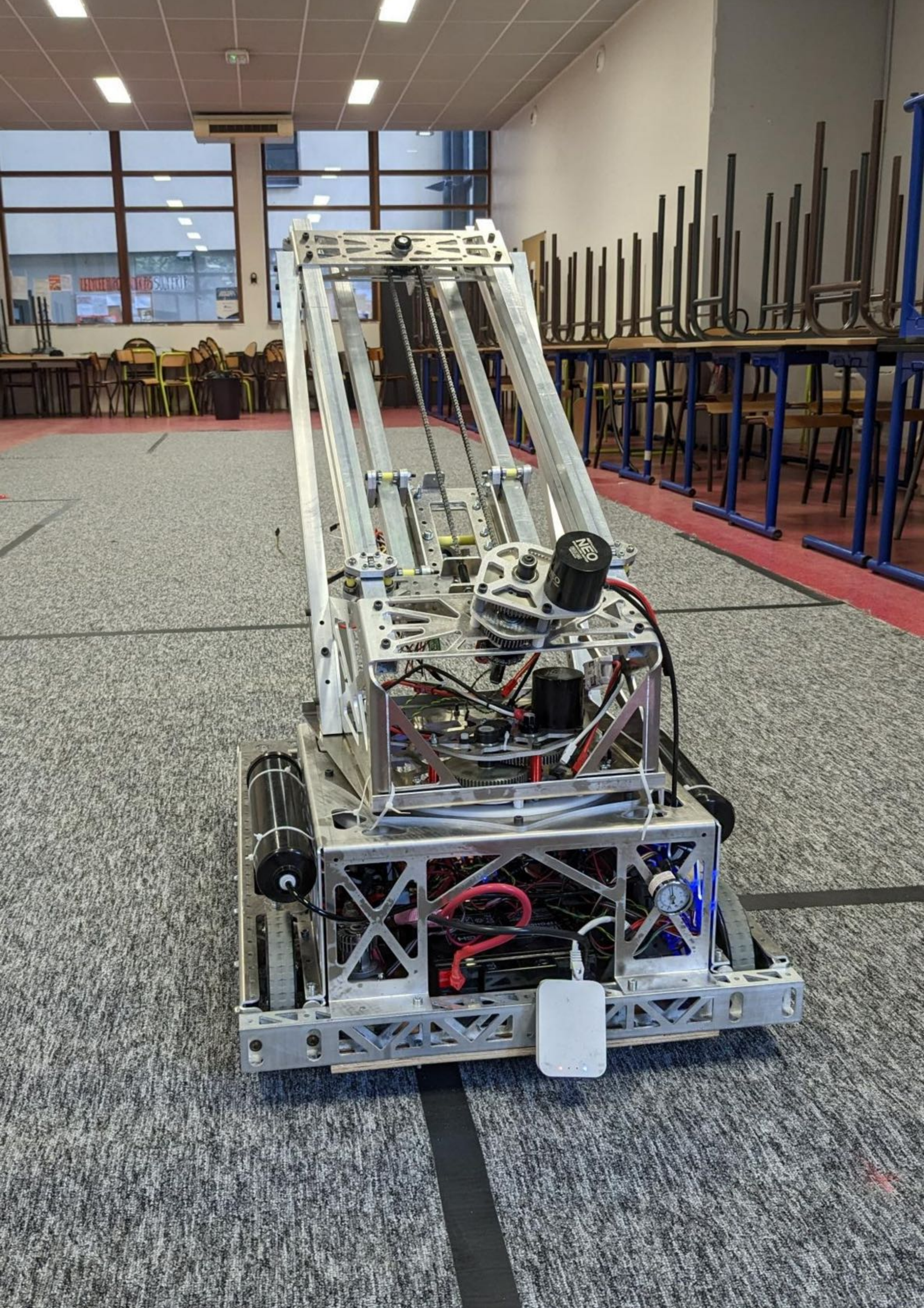
TECH BINDER

ROBO'LYON
The FIRST French Team


NEO

CONTENTS

ROBO'LYON The FIRST French Team

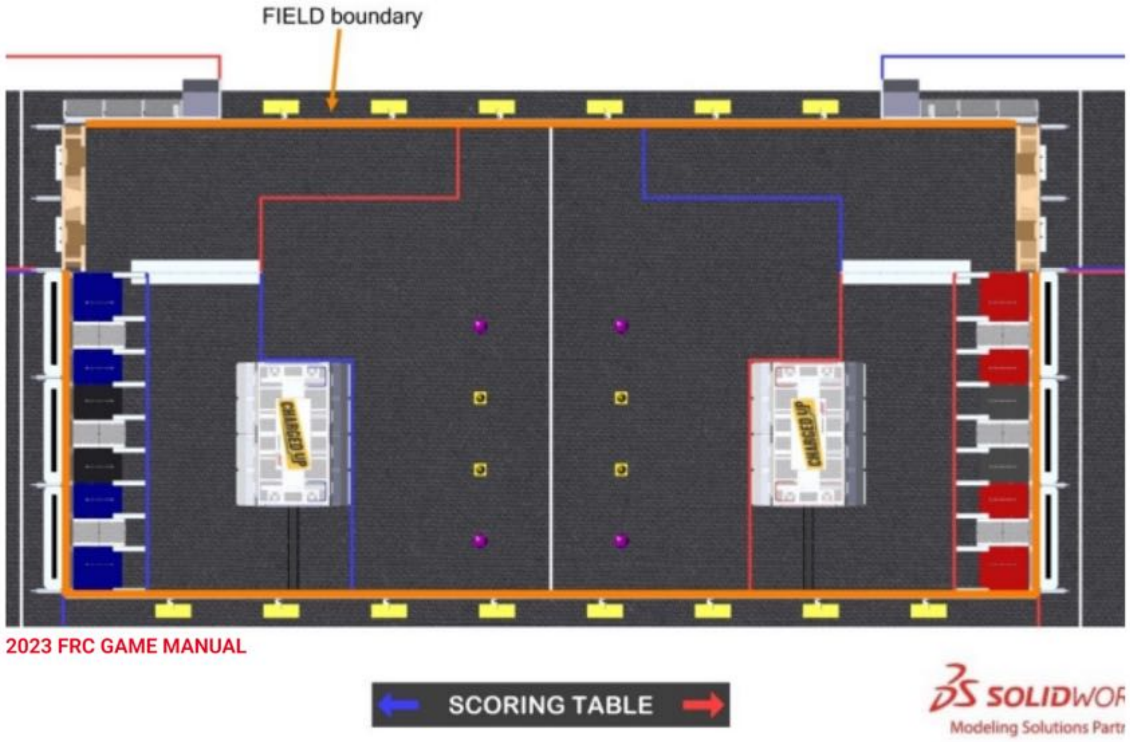
ROBO'LYON
The FIRST French Team

01. GAME

CHARGED UP

This season, each alliance has to bring energy to its community by dropping game pieces (cones and cubes) into the grid in the alliance's community zone. Cones and cubes can be retrieved from the field or from the sub-stations assigned to each alliance. At the end of the match, robots can dock (or engage) their charge station to earn additional points.

One match lasts 2:30. The first 15 seconds are called the Autonomous Period (AUTO) when robots operate without any input from the Drive Team. The remaining 2:15 are called the Teleoperated Period (TELEOP). During this time, Drivers operate robots remotely.



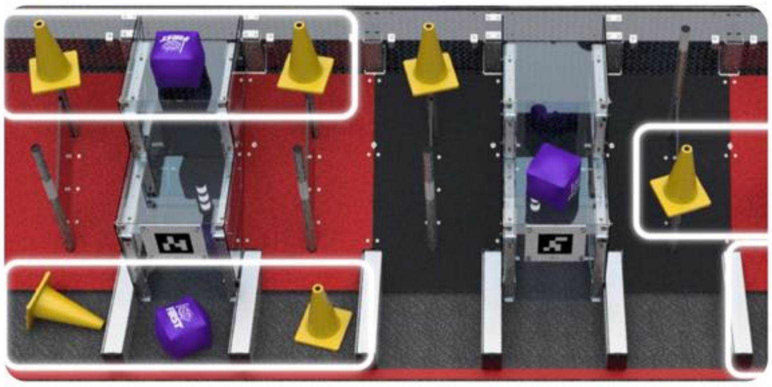
2023 FRC GAME MANUAL

SOLIDWORKS Modeling Solutions Partner

GAME

POINTS

Robot leaves completely its community	3
Game piece **SCORED** on a **BOTTOM** row	3
Game piece **SCORED** on a **MIDDLE** row	4
Game piece **SCORED** on a **TOP** row	6
One robot **DOCKED** and not engaged	8
One robot **DOCKED** and **ENGAGED**	12



2023 FRC GAME MANUAL

POINTS DURING AUTO

Game piece **SCORED** on a **BOTTOM** row	2
Game piece **SCORED** on a **MIDDLE** row	3
Game piece **SCORED** on a **TOP** row	5
LINK: game pieces scored on 3 adjacent nodes in a row	5
Each robot **DOCKED** and not engaged	6
Each robot **DOCKED** and **ENGAGED**	10
Each robot **PARKED** (completely contained within its community but not docked)	2

POINTS DURING TELEOP

SUSTAINABILITY BONUS: at least 5 links scored	1 RP
ACTIVATION BONUS: at least 26 charged station points earned	1 RP
TIE	1 RP
WIN	2 RP

RANKING POINTS (RP)

COOPERTITION BONUS: If at least 3 game pieces are scored on each alliance's co-op grid, both alliances only need to score 4 links to activate the SUSTAINABILITY BONUS.

ACTIONS & CYCLES

	Scoring cube picked from ground on bottom row	Scoring cube picked from ground on middle row	Scoring cube picked from ground on top row	Scoring cube picked from sub-station on bottom row	Scoring cube picked from sub-station on middle row	Scoring cube picked from sub-station on top row	Scoring cone picked from ground on bottom row	Scoring cone picked from ground on middle row	Scoring cone picked from ground on top row	Scoring cone picked from sub-station on bottom row	Scoring cone picked from sub-station on middle row	Scoring cone picked from sub-station on top row	Engaging charge station in AUTO	Engaging charge station during End Game
Moving (forward/backward/left/right)	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Adjusting robot's position	X	X	X	X	X	X	X							
Climbing on the charge station													X	X
Balancing on the charge station													X	X
Descending from the charge station													X	
Picking a cube from the sub-station				X	X	X								
Picking a cube from the ground	X	X	X											
Picking a cone from the sub-station										X	X	X		
Picking a cone from the ground							X	X	X					
Straightening a cone up							X	X	X	X	X	X		
Dropping a cube on bottom row	X			X										
Dropping a cube on middle row		X			X									
Dropping a cube on top row			X			X								
Dropping a cone on bottom row							X			X				
Dropping a cone on middle row								X			X			
Dropping a cone on top row									X			X		

STRATEGY

To maximize ranking points in qualification matches, we must do the following:

- **SUSTAINABILITY BONUS** **If the coopertition bonus is activated the ranking point linked to the sustainability bonus becomes much easier to obtain as this drops the number of game pieces to place by three and thus the number of cycles by three.**

- **ACTIVATION BONUS** **Scoring charge station points can be done during the autonomous period. Having a robot docked or engaged during the autonomous period brings the ranking point threshold to 18 or 14. This means that there only needs to be two robots docked and engaged on the charge station during end game, since each docked and engaged robot brings 10 points. Making sure our robot can dock during the autonomous period is a must. If we do not manage to do so, we will need two more robots during the endgame and we must have a slim robot to enable other robots to dock and engage with us on the charge station.**

STRATEGY

To maximize points earned during matches, we need to optimize our way of interacting with the charge station and our way of scoring game pieces.
It is imperative to enable two other teams to dock and engage on the charge station during endgame. This means our robot has to be as slim as possible.
Since links increase the number of points significantly, it is crucial to minimize cycle time to get these game pieces. Most of the links need a combination of cubes and cones. This means our robot must be able to score cones as well as cubes reliably.
Defense will also be something to watch out for. This will happen during the cycles between the grid and the charging station on opposite sides of the field. Our strategy is to be powerful and heavy enough to protect ourselves against defending robots.
This analysis leads us to our Subsystem Strategy:
General:

1. Fast and heavy robot to minimize cycle time without being bothered by defense.
2. Precision and versatility to minimize scoring time.
3. Low center of gravity to prevent tipping during high acceleration maneuvers and collisions but also when docking and engaging on the charge station.

(See following pages for our strategies regarding each mechanism)

ROBO'LYON
The FIRST French Team

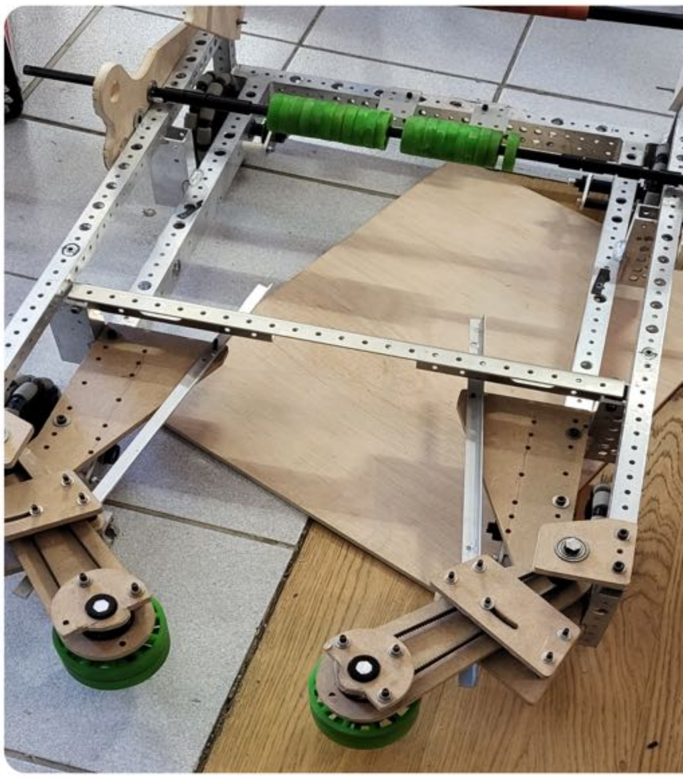
02. DESIGN

PROTO TYPING

When designing and building a robot, we work in an iterative way. We start by conceiving possible versions of mechanisms in CAD. Then, we build prototypes made of MDF or plywood to test the feasibility and efficiency of our ideas. After the testing phase we go back to CAD to improve our mechanisms. This back and forth process continues until we are totally satisfied with our robot's efficiency.
We use a **CNC** (Computer Numerical Control) to machine MDF and plywood ourselves for our prototypes. We also use it to machine some parts of our final mechanisms out of PTFE.
We also own a **3D printer** to make small elements for the robot, like struts.

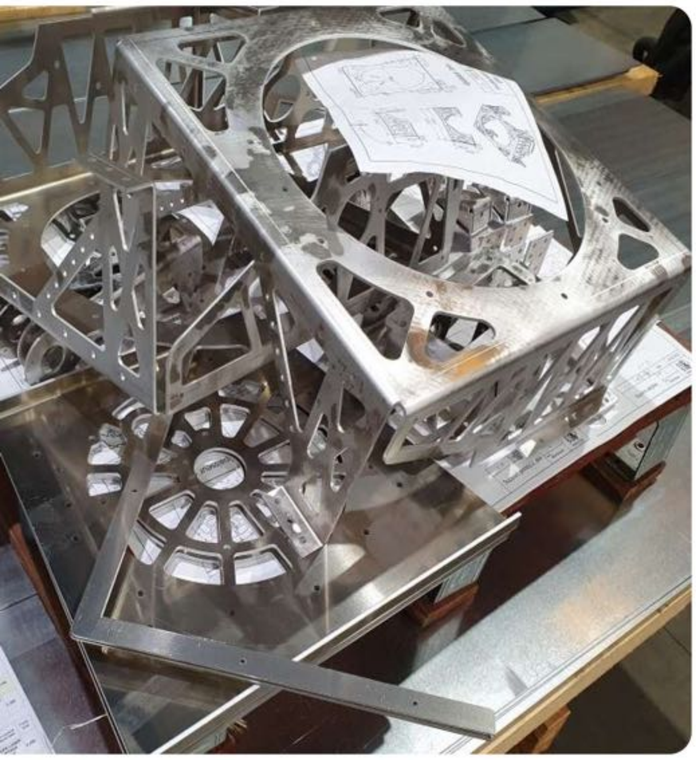
MATERIALS

MDF : Medium-density fibreboard is principally made of wood fibres. We can machine it ourselves using our CNC. Machining MDF is easier than plywood but it is less sturdy. We do not use it to build the final robot but it is very convenient to use on prototypes.



PLYWOOD : Plywood is composed of thin layers of wood. It does not break easily so we use it on prototypes that need to be very sturdy. We can machine it with our CNC but it is more difficult than MDF. It often ends up wrapped or the CNC can't manage to pierce it completely. We don't use it to build the final robot either.

PTFE : Polytetrafluoroethylene is a synthetic material with lots of properties. It is light and sturdy (almost as sturdy as aluminum) and bends before breaking. It is smooth which can be very interesting. We can also machine it using our CNC. It means we can build some pieces used on the final robot ourselves without any help from a partner.



ALUMINUM : It is the main material used on the final robot. We use it for plates and folded sheets. It is less sturdy than steel but lighter. We also need help from our sponsors to machine it.

STEEL : It is the sturdiest material we get to use on the final robot but it is also the heaviest. We only use it on parts that need to be very sturdy (e.g. the gears used in our 3 Falcon motors gearboxes). We can't machine it and need help from our sponsors.

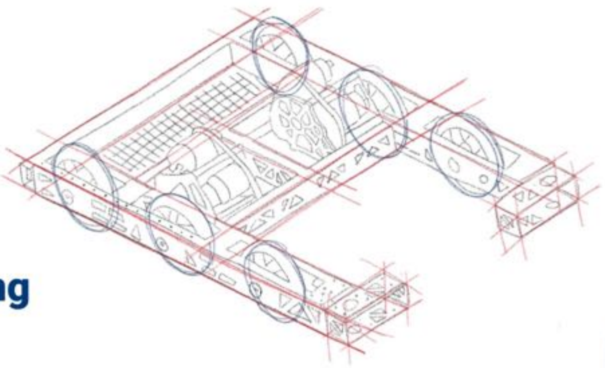
ROBO'LYON
The FIRST French Team

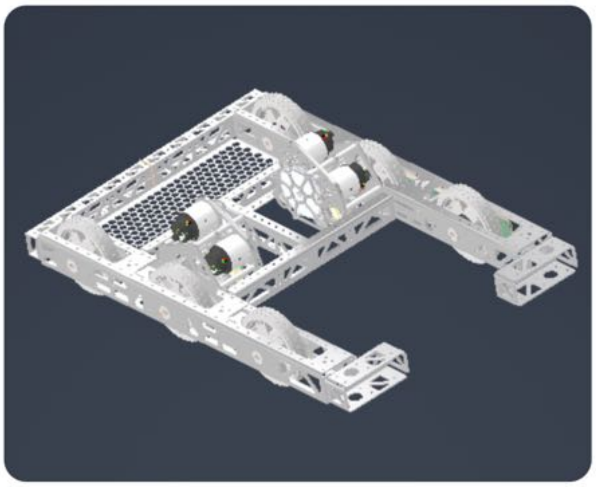
03. MECHANISMS

DRIVETRAIN

GOALS

- Powerful and sturdy drivetrain
- Fast movements
- Quick assembling and disassembling





FEATURES

- Use of folded sheet metal instead of welded tubes
- Gearboxes: 3 Falcon motors per box
- Reduction of 12.6:1 in low gear
- Reduction of 8:1 in high gear

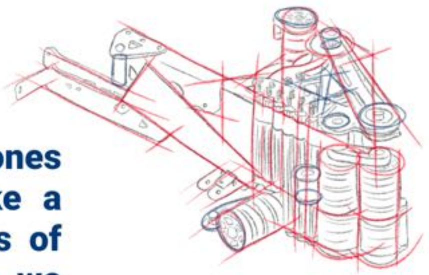
BENEFITS

Important torque and speed allow the robot to cross the field very quickly.

DRAWBACKS

Less maneuverability than a swerve drivetrain.

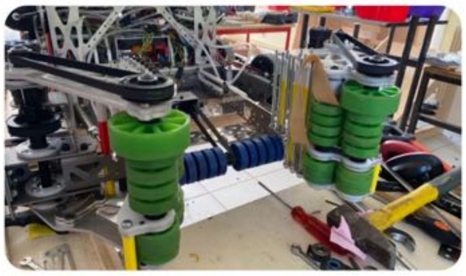
INTAKE



GOALS

In this year's game, it is possible to pick up cubes and cones from the ground. the initial issue was therefore to make a versatile intake capable of being effective with both types of objects. However, after different versions of the mechanism, we decided to focus on picking up cubes. this year, the intake also had to respond to a second problem, namely bringing the cubes to a place on the robot so that they could be manipulated by the gripper.

FEATURES



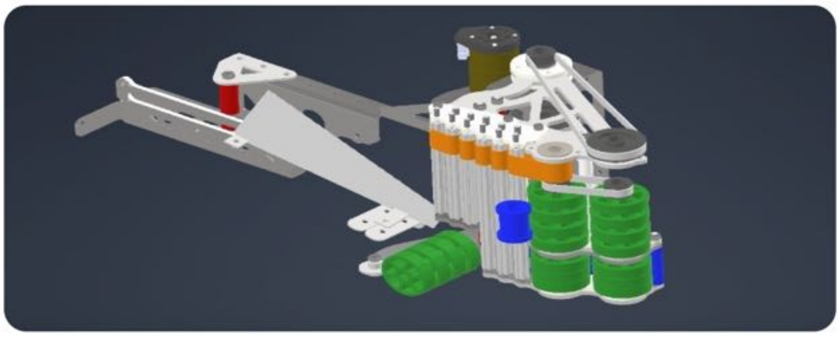
- Composed on each side of 2 vertical rows of wheels driven by 2 motors 775 pro with a reduction of 10.
  - A row of vertical axes driven by belts to better support the cube inside the robot.
  - Each arm of the intake can be retracted inside the robot thanks to 2 jacks with a 100mm stroke.
- Inclined plate with 2 horizontal rows of wheel serving as a conveyor driven by a CIM with a reduction of 15.
  - Connected to the arms of the intake so that it retracts inside the robot at the same time as the intake.

BENEFITS

It is a robust and compact mechanism which is very effective in picking up cubes on the ground.

DRAWBACKS

Mechanism with many parts that is therefore difficult to assemble and disassemble and is very heavy. It also can't grab cones on the ground.

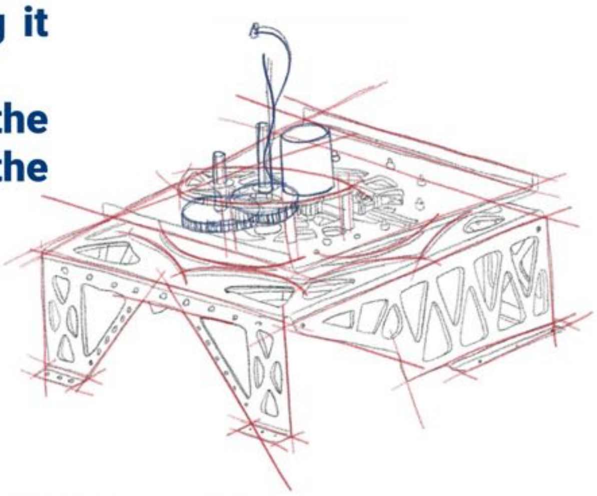


MECHANISMS

TURRET

GOALS

- Supporting the robot's upper part – elevator and gripper – and making it turn 360 degrees.
- Keeping an easy access to the gearboxes positioned under the turret.



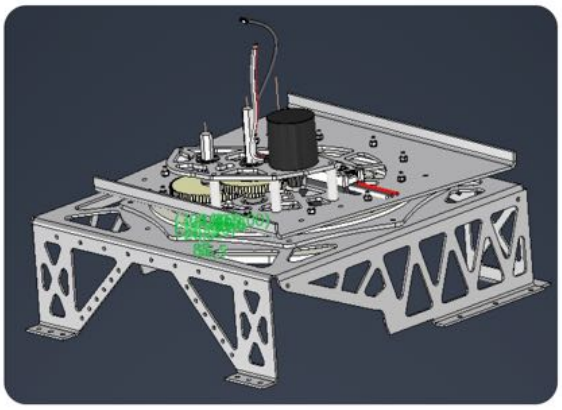
FEATURES

Fixed part:

- Attached to the drivetrain
- Folded sheet metal structure supporting everything
- Support plate for bearings guiding rotation
- PTFE gear – 54 teeth module 5 millimeters

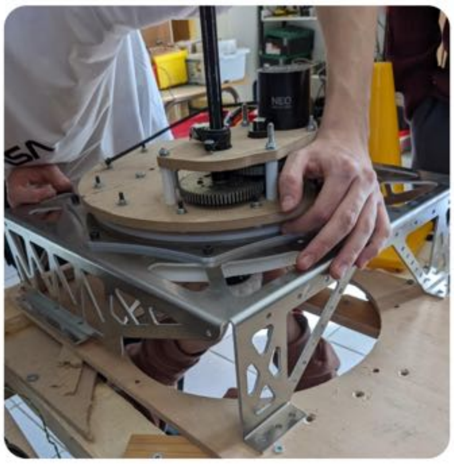
Movable part:

- NEO motor gearbox: reduction of 95
- PTFE cogwheel – 14 teeth module 5 millimeters
- Bearings guiding the motion
- Plate linking the turret with the elevator



BENEFITS

Robot can set game pieces 360 degree round.



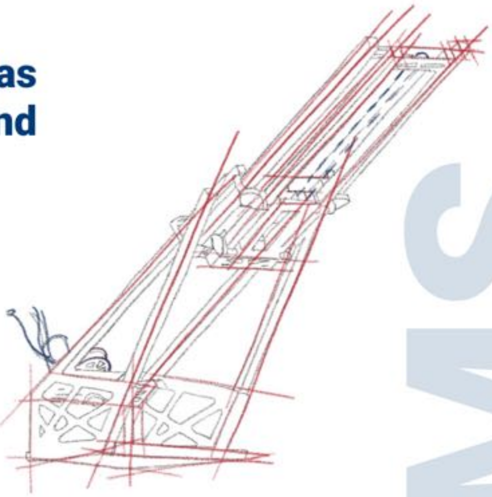
DRAWBACKS

Can make the robot fall over if the turret is moving too quickly.

ELEVATOR

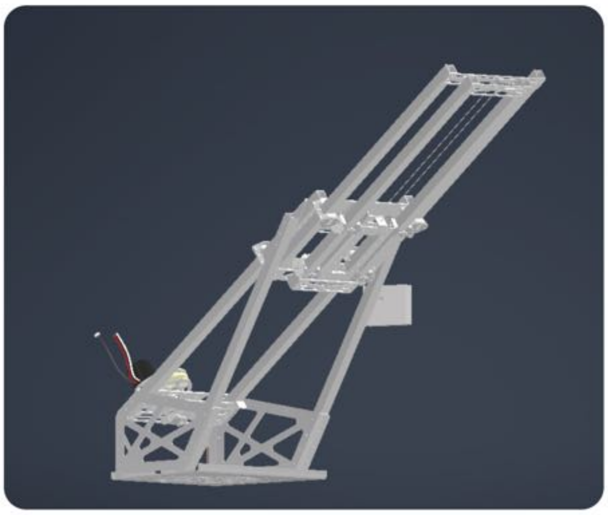
GOALS

- Allowing the gripper to move up and down (as well as forward) to pick up game pieces and set them down in the grid no matter the row.



FEATURES

- "Diagonal" elevator to be able to move up and forward at the same time
- NEO motor gearbox: reduction of 12
- Extension with waterfall effect with a chain linked to the first level and a cable to bring it up



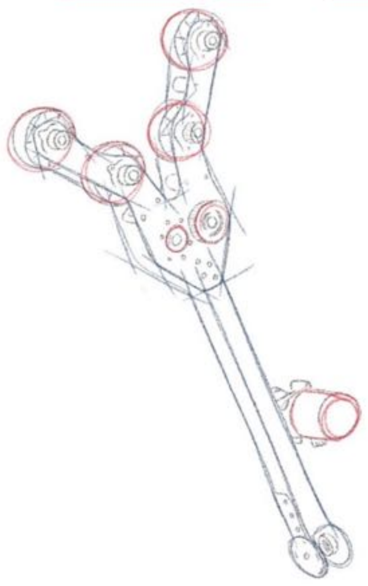
BENEFITS

The waterfall effect allows the weight and the current needed to be evenly distributed.

DRAWBACKS

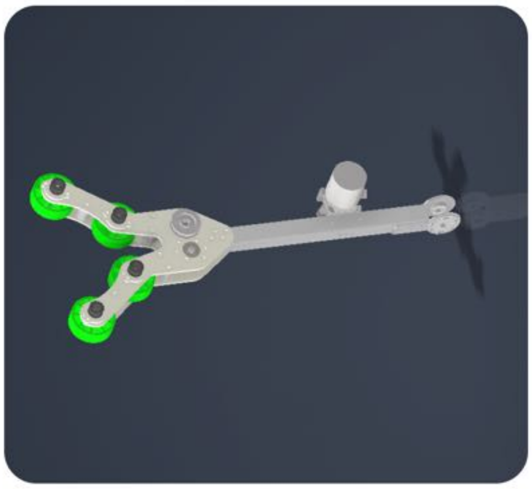
Both levels of the elevator need to move simultaneously.

GRIPPER



GOALS

We aim to catch game pieces stocked in the robot or on the field and to be efficient for both cubes and cones. The mechanism designed for the Trois – Rivières regional, which consisted of 2 fingers that opened or closed thanks to cylinders of 100mm course and 16mm diameter, was completely redesigned for the championship.



FEATURES

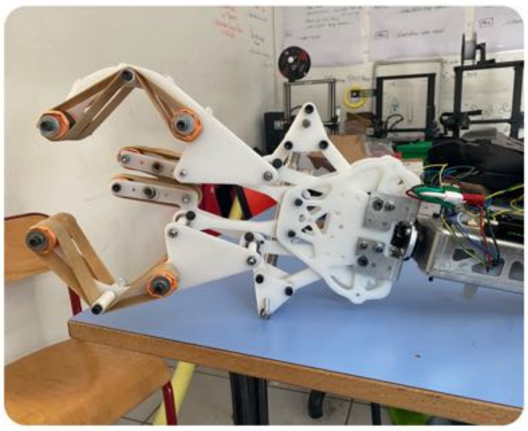
- Use of 3 inches green wheels that catch and compress the game's pieces from 3 to 4cm.
- Motorized by a NEO coupled to a planetary with a reduction of 5.
  - Transmission thanks to GT2 24T pulleys and 100T belts
- Design of bearing blocks fixed on the plates to prevent them from coming out.

BENEFITS

System that allows the gripper to move forward and close at the same time, and allows us to catch the object even from a distance.

DRAWBACKS

It was difficult to decide how and where we should place the jack because it takes up space and it needs to be firmly hooked. We opted to fix it towards the end of the clamp, to optimize the space it takes, even if it weighs down the jack.



ROBO'LYON
The FIRST French Team

04. PROGRAMING

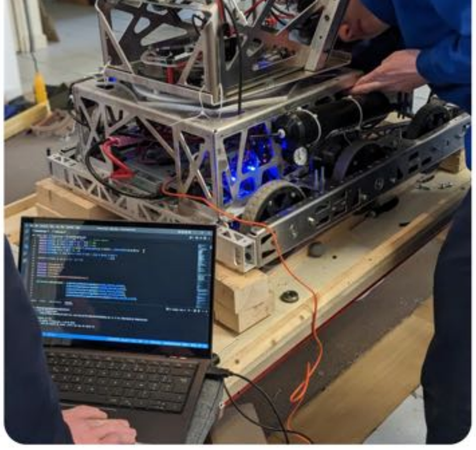
CODING

Like every year since 2017, we've coded our robot in C++. To deal with the code writing clarity and readable issue that we have been facing for years, we have organized it this year in command-based programming in order to allow better accessibility and organization. It also allows us to individually modify each mechanism without affecting the robot code. Thus, it enables a better adaptability in the code and therefore a more practical side in the improvement of the mechanisms. It is an object-oriented programming. About the drivetrain, we use of several encoders for the drivetrain to count the distance.

To ease the piloting of the robot and make its manoeuvrability more fluid on the playground, we have developed an automatic two speed gearboxes, with an autonomous changing between a high-torque low gear and a higher gear. For this we have thought about 10 rules for automatic gear changing.

First of all, in any case, the change of speed is impossible if a change has already taken place recently or if the robot is turning. Then, the passage to a higher speed can only take place if the robot does not slow down, if it accelerates, if the joysticks are activated at more than 80% and if it is above 70% of the maximum speed. For the downshift, the robot must be below 60% of its maximum speed, the joysticks must be less than 60%, it can only be triggered if the robot is at reduced speed and if it decelerates.

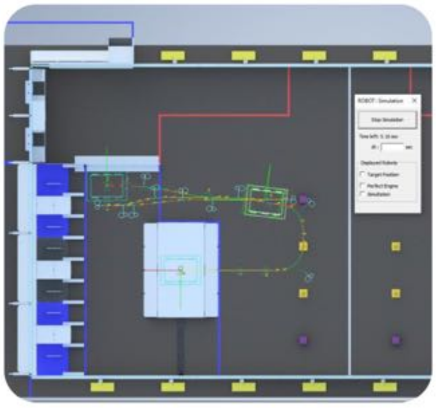
This year, we also used PID controller to control our turret. It can thus be ordered to turn with precision in a certain angle, which enables to avoid wasting time and to gain precision during matches.



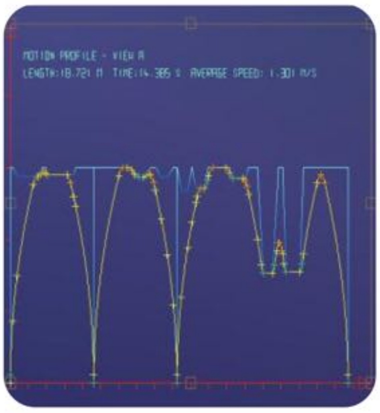
PROGRAMMING

AUTONOMOUS PERIOD

For 3 years, the programming team has overseen developing a software and a library facilitating the coding of the autonomous period. We normally used the “path planner” to encode the autonomous period but we wanted to go further. Indeed, this software is based on splines, but we wanted better control with a path 100% operable by the robot. For this, we draw a path on the playground that the robot will follow during the autonomous period. This path is composed of 3 types of primitives: segments, circular arcs and clothoids. These clothoids are essential to allow a linear change of speed between segments and circular arcs, thus ensuring that the robot is able to follow the path.

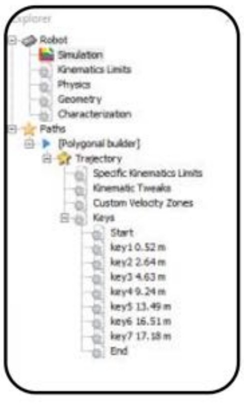


To this we have integrated a speed graph which allows the software to calculate the theoretical maximum speed of the robot on the whole path. However, this graph considers the real acceleration of the robot, which we calculated during the characterization period. To do this, we made the robot perform numerous round trips to obtain data on acceleration, maximum speed, etc. A different code listed these data and calculated different coefficients to numerically model the motors. Thanks to this, we can convert all our speed and acceleration data into voltage to transmit it to the robot. We also modeled a simulation robot with this characterization data to ensure that the robot can actually follow the path. From then on, we can modify the speed or acceleration data to ensure that the robot correctly follows the trajectories indicated.



```
MOTION PROFILE - VIEW A
LENGTH:18.721 M  TIME:14.385 S  AVERAGE SPEED: 1.301 M/S
```

Finally, in this path, we can integrate keys allowing to give orders to the robot. These commands can be simple, like stop, make a U-turn, reverse motors, or can send messages to the code for the activation of the different mechanisms of the robot. This software, very adaptable, allows for each season to load the different playgrounds and the different commands so that it can henceforth guarantee an effective autonomous period every year.



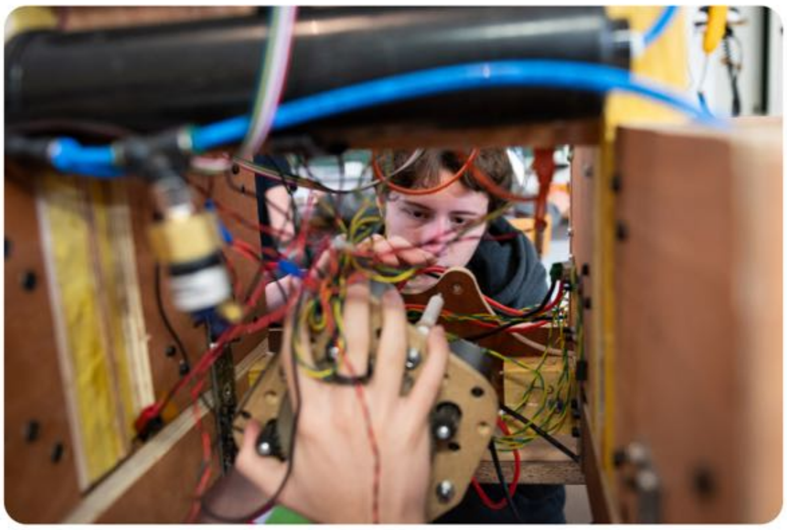
```
Explorer
Robot
    Simulation
    Kinematics Limits
    Physics
    Geometry
    Characterization
Paths
    [Polygonal builder]
        Trajectory
            Specific Kinematics Limits
            Kinematic Tweaks
            Custom Velocity Zones
            Keys
                Start
                key1 0.52 m
                key2 2.64 m
                key3 4.63 m
                key4 9.24 m
                key5 13.49 m
                key6 16.51 m
                key7 17.18 m
                End
```

VISUAL PROCESSING

The objective was to save the pilot time by automatically aligning with the elements of the playground.

Use of the Photonvision library for image processing

Use of the library Network table to communicate with the robo rio



The program is operating with a raspberry pi 3b+ to improve performances. It is divided in different stages:

1. Converting the image from RGB to HSV
2. Filtering each pixel according to its color
3. Image processing with canny filter
4. Contour's detection
5. Contour's filtering (according to their properties)
6. Contour's coupling, creating a new target
7. Tracking the optimal trajectory to reach the detected target thanks to the use of odometry and motion control
8. Aim the turret, control the hood angle, and adjust the flywheel speed

55
53