Yoink Info Packet

Table Of Contents

Overview

Robot Requirements

Subsystem Specifications

Overview

The Competition

Our team, Team FRC 6201 The Highlanders, designed and built a robot to compete in the 2023 FIRST Robotics Competition, CHARGED UP. From the game manual:

In CHARGED UP presented by Haas, two competing alliances are invited to process game pieces to bring energy to their community. Each alliance brings energy to their community by retrieving their game pieces from substations and scoring it into the grid. Human players provide the game pieces to the robots from the substations. In the final moments of each match, alliance robots race to dock or engage with their charge station!

Each match begins with a 15-second autonomous period, during which time alliance robots operate only on pre-programmed instructions to score points by:

- leaving their community,
- retrieving and scoring game pieces onto the grid,
- docking on or engaging with their charge station.

In the final 2 minutes and 15 seconds of the match, drivers take control of the robots and score points by:

- continuing to retrieve and score their game pieces onto the grid and
- docking on or engaging with their charge station.

The alliance with the highest score at the end of the match wins!

Our Goals

Our goals for this competition is that our robot will accomplish as many of these tasks as possible in order to obtain the highest score during a match. It will also be designed for safety, reliability, and operational ease.

Robot Requirements

Functional Requirements

These describe the functions which the robot must perform during operation on the playing field.

Name	Description
Human Control	Must be able to be remotely controlled by a human operator
Autonomous Control	Must be able to perform certain tasks without human operation
Drive	Must be able to drive around the playing field
Collect Game Pieces	Must be able to collect game pieces
Place Game Pieces on Nodes	Must be able to place game pieces on available nodes
Balance on Charging Station	Must be able to balance on Charging Station

Non-Functional Requirements

These describe the attributes of the robot both on and off the playing field.

Name	Description
Safety to Humans	Must be safe for humans to handle, operate, and be around
Safety to Environment	Must not be a hazard to other robots or the playing field
Reliability	Must be robust and reliable during operation, transportation, and storage
Wiring Organization	The internal wires must be well organized and routed effectively

Our goals for this competition is that our robot will accomplish as many of these tasks as possible in order to obtain the highest score during a match. It will also be designed for safety, reliability, and operational ease.

Constraints

These describe any constraints which must be observed when designing and constructing the robot.

Name	Description
Robot Height	May not be more than 72 in. (~183 cm) tall
Robot Frame Perimeter	May not be greater than 120 in. (~304 cm)
Robot Extension	May not extend more than 48 in. (~122 cm) beyond robot perimeter
Robot Weight	May not exceed 125 lbs. (~56 kg) excluding BUMPERS and BATTERY
Parts Cost	May not exceed $5000, or have any individual part more than $500
Damage Prevention	Must not have surface features or protrusions which could damage the arena or other robots
Hazardous Materials	Must not be construction from any hazardous materials (see game manual for full list)
Bumpers	Must use bumpers to protect outside corners of frame perimeter
Electronics Restrictions	May only use electronics permitted in the game manual

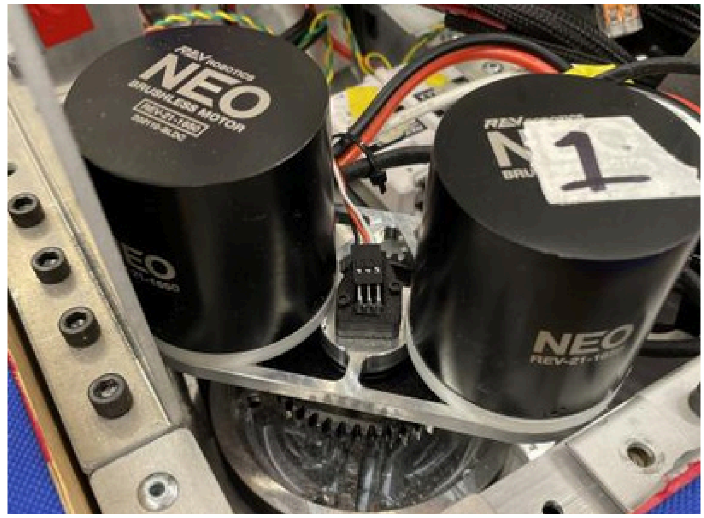
Subsystem Specifications

Our robot is made out of different subsystems. Each subsystem was designed to in order to try to get as many points as possible while being realistic for the team to build in turns of skill, time and budget.
The table below lists each subsystem and describes what functionality it provides.

Name	Description	Functionality
Swerve Drive & Frame	Swerve Drive allows the robot to move and turn in any direction while the frame of the robot is that which the rest of the robot is mounted.	Structure, Drive
Intake	Collects game pieces from the floor and launches them into the intook. It then ejects game pieces into hybrid goals to score.	Collect, Score
Arm	A four-bar linkage attached to the frame that reaches out from inside the robot to the shelf or nodes so that the gripper can access them.	Collect, Score
Gripper	A four-bar structure with rotating “fingertips” that is attached to the arm. Picks up game pieces from the shelf and places them on scoring nodes.	Collect, Score
Electronics	Battery, robot controller, motor control, and all other robot electronics.	Power, Control
Software	Programming for the robot for autonomous and manual functionality.	Control

Drive Train: Swerve

Swerve

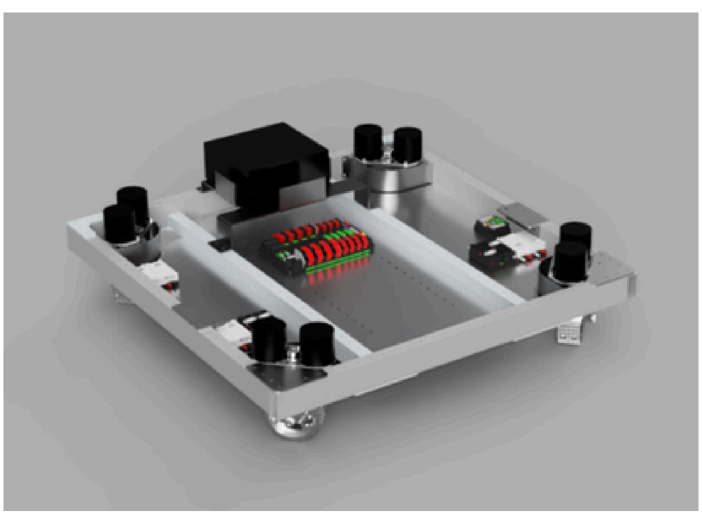


We use 4 SDS Mark 4 Modules with L1 ratio. There are NEO motors on both drive and angle controls. Additionally, we have Thriftybot Absolute Encoders to measure wheel heading. We mount these underneath a frame of 2x1 box tube.

Code

To control these modules we use a custom inverse kinematics function that takes in the desired translation and rotation speeds and splits those into translation and rotation vectors. Then it adds them, sending them to the wheels. We also run an optimization function on the wheel states to prevent them from spinning 180 degrees. Instead we can reverse the drive direction. We control each module using a PID loop on the angle of the wheel and for the drive speed. We utilize a PID loop around the robot's heading which helps to keep the robot straight and fights drift.

Our robot uses its onboard gyroscope, a NavX XMP, to keep track of its heading relative to a stored direction. We begin a match with the gyro aligned with the field to aid our autonomous, but during Teleop, the driver can reset it at any time, to aid in control. This understanding of the robot's heading on the field allows us to translate in straight lines, and rotate at the same time. This greatly increases the complexity of driving we can preform

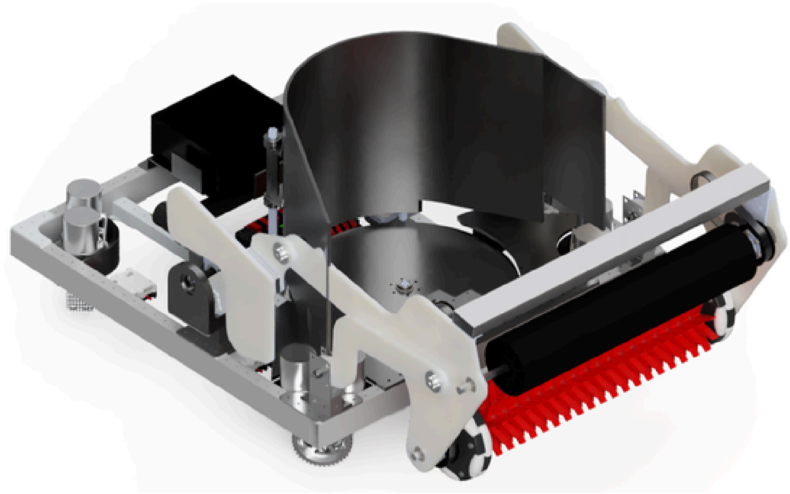


We also use an odometer on our robot, which keeps track of how far each wheel has gone, and in which direction they are facing. This allows us to estimate how far we have gone, and the current pose of the robot. We have a system to allow us to move the robot to any point in this coordinate system, which is implemented in our trajectory following, and AprilTag aligning code.

Intake

Intake Rollers

Our robot utilizes an intake comprised of a set of two wheel rollers to intake cubes from the floor. During our brainstorming and prototyping phase, we tested many different variations and configurations to determine what the best dimensions and parts would be. Once we had our final design, we moved to CAD to design and manufacture the final parts. We then assembled the final components and began testing.



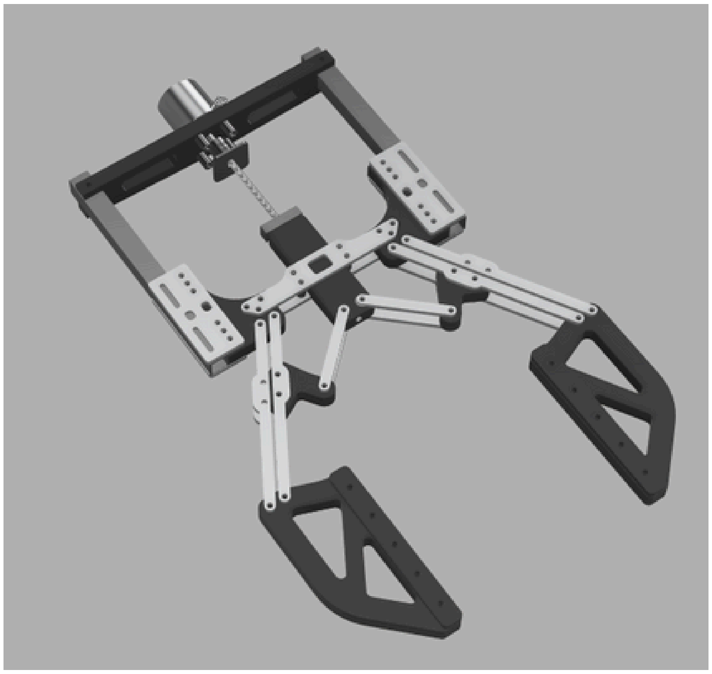
Our intake uses two rollers of Andymark compliant wheels mounted on custom designed and waterjet ABS sheets. The intake is stored upright and inside our robot at the start of a match, and is then deployed to rest low along the floor allowing for optimal intaking capabilities. Our intake's deployment system also allows the intake to be retracted during the match to allow for easier aligning to the nodes.

Intake Code

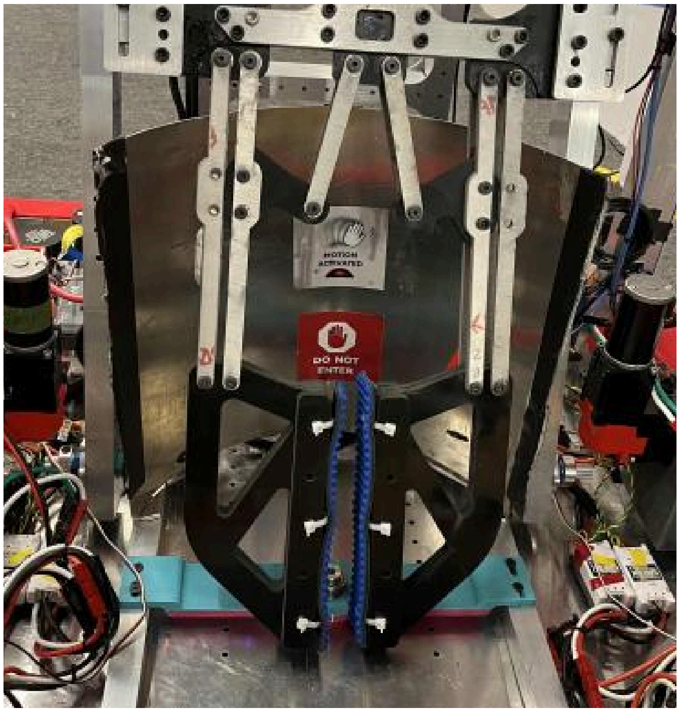
The intake is controlled using a variety of buttons for the driver, which control the speed of the intake, and alignment mechanisms. We also use a PID loop for smooth deployment and retraction.

Gripper

Hand



The gripper on our robot is a very important mechanism because it allows us to pick up game pieces from the shelf and strategically place them on the right node. Our gripper is primarily made out of aluminum and ABS plastic as well as neoprene strips. The gripper takes linear motion from the lead screw and turns it into grip strength. We had multiple options for powering the gripper, but we chose the lead screw option because we thought it was more reliable than pneumatics or a winch. The lead screw is very simple as it takes energy from a NEO motor and converts it into linear motion. Lastly, we used a universal joint to connect the motor to the lead screw without unwanted forces. The "fingers" of the gripper are very important for placing the cones on the nodes. The fingers, made grippy by neoprene strips, are applied on a trapezoidal hand. This year, the theme of our robot is the 4-bar mechanism, which shows up in almost every system. The gripper uses two mirrored 4-bar mechanisms in order to keep the fingers parallel and ensure that they don't grab the cones at an angle.

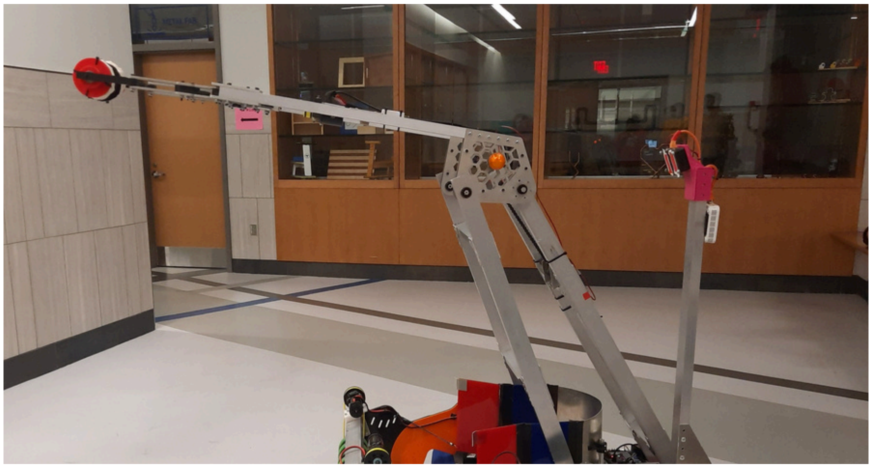


Gripper Code

We use motor current to control the force on the game piece. We control this using an extended SparkMAX, to set a current limit, and command the motor to full throttle.

Arm

During the early stages of the build season our team designed a number of different concepts for how we could place the cones and cubes onto the nodes. We considered factors such as how many joints would our arm have and what materials it would be made of. We split apart into small groups of students how would each explore a concept, then we reconvened to decide on a version to pursue. As a team we came together and decided on a four-bar based design for our arm. This arm choice had the benefits of being mechanically simple with only one degree of freedom. While considering the simplicity of this option we also found that it was one of the simplest arms to drive, as the motor would only have to be wired around one joint for the optimal ratios. The arm was designed in CAD to minimize weight while maximizing its strength. To accomplish this we designed the arm to be made out of aluminum 2x1 box tube, which gives us plenty of rigidity while maintaining a lightweight construction. Furthermore, the chain drive for the arm allows us to achieve an additional 2:1 gear ratio. A mix of custom water-jet parts, COTS brackets and stock, and rivets allows us to achieve precise geometry for the four bar linkage.

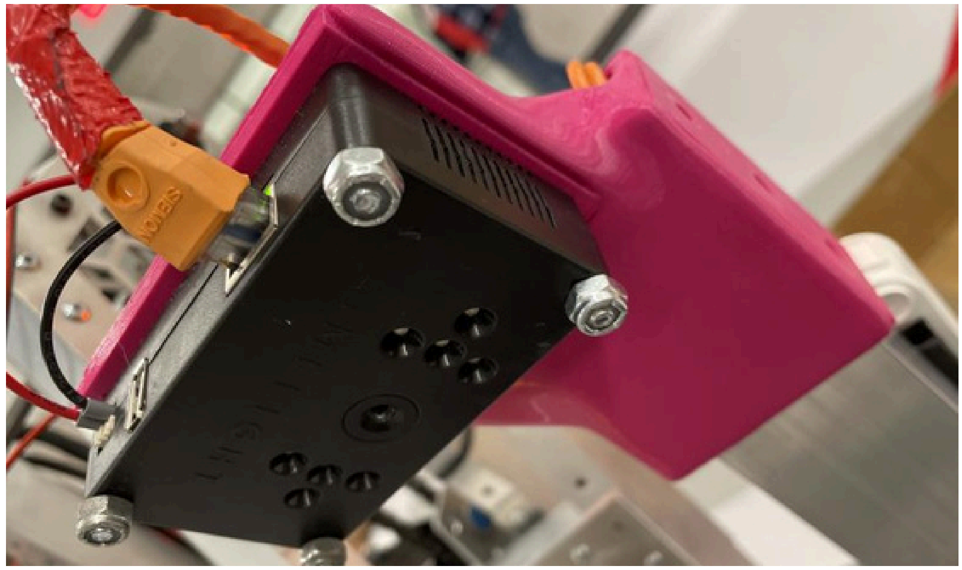


Arm Code

We utilize a PID loop to control the arm, but we use a custom wrapper class to enable us to log the setpoint data.

Programming

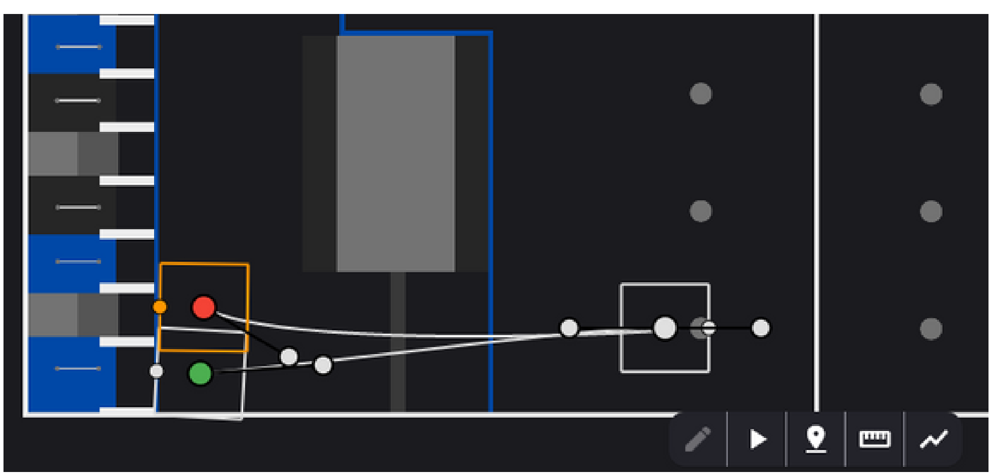
Autonomous: Path Following



In autonomous, the robot follows smooth paths across the field. We generate these using a program called Pathplanner. We use an exported CSV file of points, and our own custom following algorithm, to drive to a series of points on that path. This allows us to get a smooth following of the curves. The robot uses its encoders on the wheels to update an odometer, which lets us estimate position and controls autonomous movements. Using an autonomous relying on various paths which are easy to create means that we can have multiple autos depending on the layout of the field and what position we are in. This makes us more flexible during auto, allowing for different movements based on our alliance-mates and their plans for during auto.

Autonomous- AprilTags Alignment

Our robot utilizes a LimeLight camera to estimate the tag's position relative to the robot and then uses its gyroscope and some trigonometry to convert that position into the corresponding position in the robot's odometer relative coordinates. This enables us to command the robot to move to that point or a point based on it.



Thank You To Our Sponsors!

Team Contact Information

If you have any questions about any of our subsystems whether it be a question about the planning, the CAD, the assembly or code then please feel free to contact us using our contacts below.

We would love to help and answer any questions!

Social Media Contacts

Website: team6201.com
Email: frc6201@gmail.com
Facebook: FRC Team 6201
Twitter: @frc6201
Instagram: @frc6201