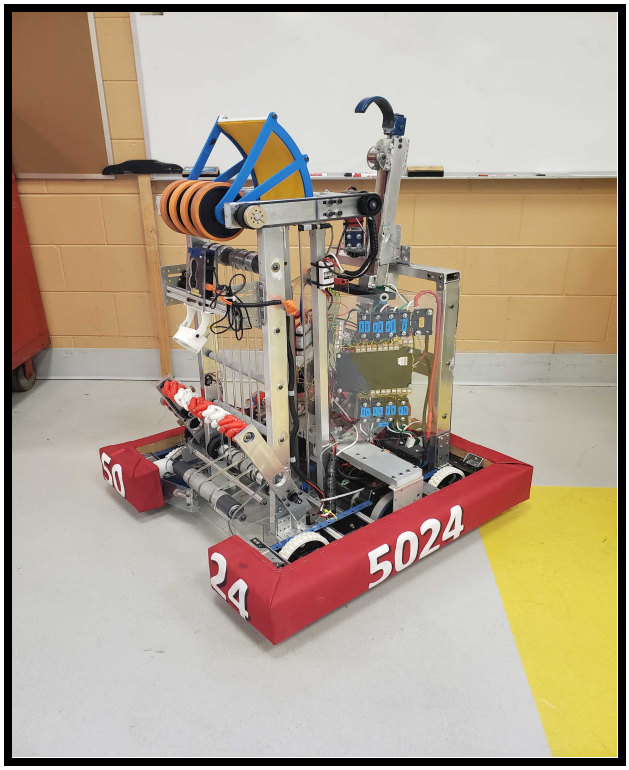
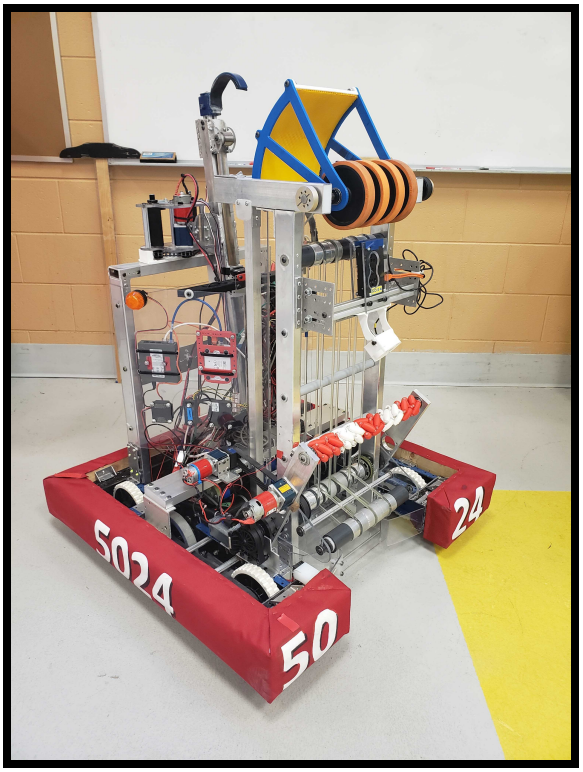
Raider Robotics
2020 Technical Binder

5024 Raider Robotics would like to present to you our 2020 Infinite Recharge robot: Darth Raider!





Strategic Analysis

FRC Team 5024 has developed an effective strategic thinking model that we follow after the initial game release. The following sections lay out the process which is taken the weekend of kickoff.

Kickoff Morning

As a team, we watched the game release video and field tour videos. With a rough idea of the game, we wrote out some general questions on bristol boards, and placed them around the room. Using post-it notes, team members wrote down answers to these questions and brainstormed ideas for strategy and robot design. Our goal was to efficiently introduce the game to the entire team to ensure everyone is comfortable with the game's objectives and rules.

Kickoff Afternoon

A group of students who choose to attend a secondary meeting on the afternoon of kickoff to review all notes and questions taken by the team earlier. These students also attempted to address key questions about the game. This was followed by a mediated group discussion about the different types of objectives, and what the team could potentially prioritize during the season. A pros vs. cons list was made for each objective. We worked to ensure that all team members had an understanding of the rules and scoring.

Sunday Strategy Meeting

On our Sunday meeting, we collected a group of students with exceptional knowledge of the game manual to have an in-depth discussion about robot design. We compiled a list of specifications, ideas, and priorities for the build season, including: abilities, size, speed, and game piece manipulation, all while keeping in mind game rules and ranking points.

In the week following kickoff, we continued working with our build and software teams to prototype, and establish our design for each mechanism.

Strategic Build Decisions

The following lays out our reasoning for making the following strategic design priority decisions. Several factors went into these decisions, such as resources, cost, time to achieve, as well as the potential Ontario teacher strike.

Drive Base

Decision- Modified tank drive with the back wheels closer together

Strategic Reasoning- A modified tank drive allows for a smaller turn radius, making it easier to move throughout the field.

Intake

Decision- Ground Pickup was made a heavily prioritized item this year

Strategic Reasoning- We decided that ground pickup must be a top priority in order to obtain balls from any area on the field which would give our team an advantage over other teams.

High Shooter

Decision- Was made a priority

Strategic Reasoning- Due to the potential of an oncoming teachers strike, we knew we had to prioritize our resources and efforts and pick between a low and high shooter. We believed we were able to create an efficient and consistent shooter if we allocated our resources into a high goal shooter in order to optimize points values, we decided to go with high.

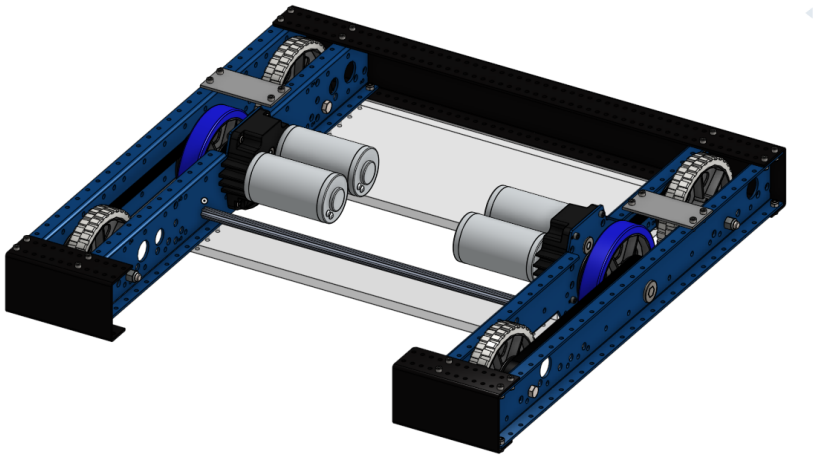
Control Panel

Decision- We decided to prioritize completing a control panel mechanism

Strategic Reasoning- In order to be a versatile robot in our alliance, and have the ability to manipulate this additional game piece, we choose to seek

Mechanical Systems

Drivetrain



Purpose- To make the robot mobile so we can move around the field in order to accomplish the objectives of the game.

Rationale- During our design process we decided that we wanted the robot to be back heavy in order to give the drivebase a back orientated pivot point. This gives the robot quicker and sharper turns. The high grip traction wheel in the centre allows for a higher friction rate with the carpet.

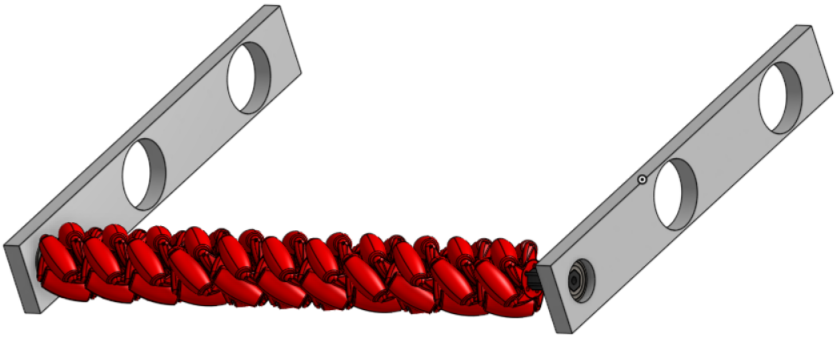
Description- The drivetrain type we chose was an adjusted C channel tank drive. The frame of the robot is 28”x 28.25”. There are 3 wheels on both sides however it relies more on the back 4 wheels as it is back heavy. There are four wheels making contact with the ground at all times. The back 2 wheels were placed closer together to allow for the robot to have a smaller turn radius. The wheels we used are standard Kit of Parts high grip traction 6-inch wheels in the centre and standard tread 6-inch wheels on either side. The C channel frame allows room in the front of the robot for the intake mechanism. The drive train also features a ⅛” centre drop.

Motor-The motors on the drivebase are two sets of CIM motors

Gears- A gear ratio of 8.45:1 which give the robot a speed of 2.7ft/sec

Sensors - In the drive train there are two optical encoders, these are used to count the rotations of the wheels.

Intake



Purpose- To roll the game piece from the floor into the hopper mechanism within the robot so we are able to perform a ground pick up.

Rationale- When deciding what we wanted to do for our intake we wanted to be able to have a grab and go intake, once we touched it we should be in control of it. Our arms are sitting on top of our bumpers so they can intake but still be protected from other robots. We chose the “mini-mecanum” wheels so that the ball would be directed into the center of our robot, they are lightweight and small rather than traditional mecanum wheels. While prototyping with lexsan arms we realized the lexsan arms were too fragile, so we switched to aluminum. While testing we saw that our intake arms extended too far out for them to consistently intake balls into our hopper mechanism so we shorten the length of our arms. There is a lift motor at the back which allows the intake to drop up and down to stay in our frame perimeter at the start of the match.

Description- Our Intake bar is a hex shaft that has mecanum wheels throughout it. Once the shaft is spun the balls that touch it will get spun into the middle of our intake due to the mecanums. The balls get funneled into our hopper mechanism.

Gears - The gears we use were the VEX 16 tooth sprockets the ratio is 1:1. The gearbox for our intake wheels is 12:1.

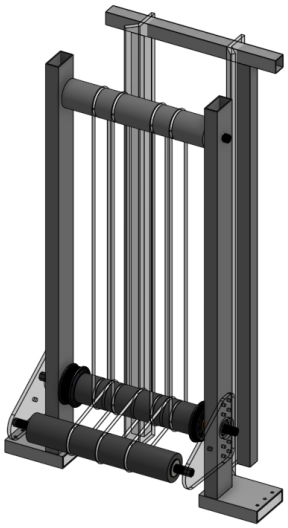
Motors - A AndyMark 775 RedLine Motor is used to spin the shaft. The motor used to lift the intake is a Bosch power window motor.

Purpose- The purpose of the intake is to be able to touch a ball and be in control of it. Moving it into the hopper mechanism

Wheels- There are two types of wheels on the intake on each side of the intake bar there are 10 3d printed Mecanum wheels (designed by FRC 125) directed towards the center of the bar and into the hopper mechanism.

Sensors- There are two limit switches within the intake, the purpose of stopping the arms so they don't damage other parts of the robot.

Hopper



Purpose- The purpose of the ball hopper is to make it so we can have multiple balls inside the robot ready to be pushed into the shooter mechanism. This is so we don't have to drive around to find another ball every time we shoot into the high goal.

Rationale- During our strategy meeting we came to the conclusion that we wanted the hopper to be able to hold the maximum allowed game pieces. In order to do this, we used tubing held by a roller to move all the balls simultaneously. The foam rollers used to move the balls from the intake into the hopper were used due to their friction. The foam material gave us enough friction to be able to properly grip the balls.

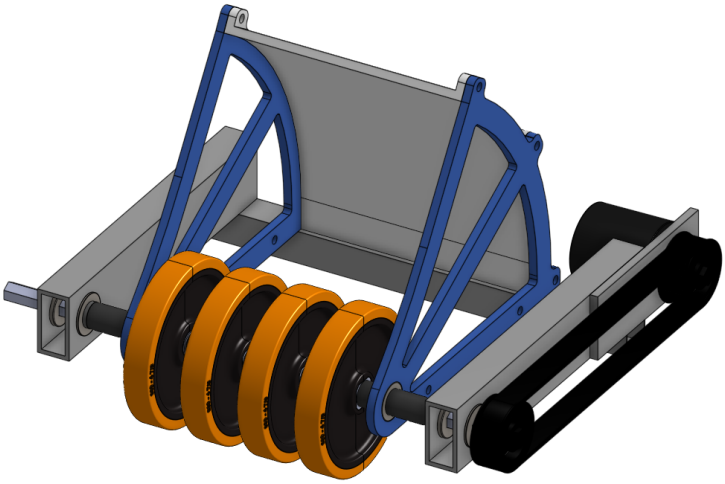
Description- The hopper is made up of 4 2x4 aluminum channels with enough space to hold a maximum of 5 balls. It uses polyurethane tubing that has been tied together tightly to use tension in order to compress the balls slightly, making the balls easier to roll. At the bottom, there are two foam rollers used to roll the balls within itself from the intake mechanism. The hopper uses 3 Line Break sensors to consistently know how many balls are in the hopper. The line break sensor at the bottom opening of the hopper and the top opening combined tells us how many balls are in the hopper at any given time. The line breaker in the centre of the hopper allows the balls to all have even spacing, which is crucial for accurate shooting.

Gears- There are three gears involved in the hopper mechanism, the gearbox which is a 57 sport gearbox with a 4:1 gear ratio, attached to the gearbox is a 167 tooth pulley, and a 42 tooth pulley gear that is attached to the the other end of the belt that spins the whole mechanism.

Motors- The motor that was used for the hopper mechanism is a 775 redline motor

Sensors- There are 4 sensors within the hopper, one magnetic encoder and three line break sensors. The magnetic encoder is to count the number of rotations the polyethylene belt travels. The line break sensor's purpose is to know how many balls we have within the hopper and to know when to start and stop the hopper from moving.

Shooter



Purpose- To fire game pieces into the high target from close and far ranges.

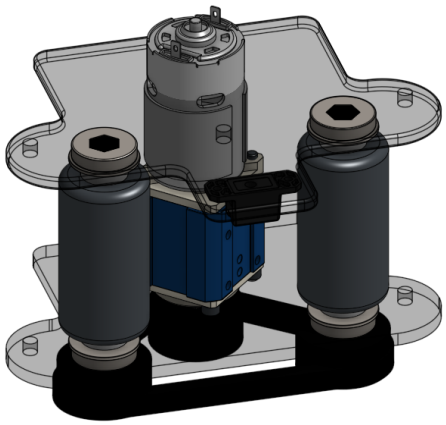
Rationale- The shooter is able to accurately score high goal shots from the far end of the trench, moreover the motor speed can be adjusted allowing the robot to score at closer distances. Additionally we opted to use a single axle to reduce the amount of overall motors on the robot, resulting in less power drawn from the battery. The research indicated that a fixed hood shooter was the most practicable option. Moreover, a hood would be optimal with a one motor shooter.

Description- The shooter consists of 4 wheels, a steel hex shaft axial and a fixed 30 degree angled hood. The benefit of the shooter mechanism is its efficient and consistent performance. The mechanism is able to fire 5 balls with a less than 1 second gap in between feeds and maintain a consistent angle and distance. The shooter is able to accurately score high goal shots from the far end of the trench, moreover the motor speed can be adjusted allowing the robot to score at closer distances. Overall the mechanism is efficient, consistent and extremely reliable.

Motor and Gears-The axle is powered by a neo motor with a gear reduction of 1:1.19 This is used for aiming at the ports, this will help for a more accurate aim.This motor was chosen because it has twice as much torque as well as a more RPM making it superior to the commonly used SIM motor.

Sensors- In the shooter there is one main sensor, the limelight. This sensor provides the robot with 3D pose calculations that are fed into the shooter velocity control loops and drivebase targeting systems.

Control Panel Spinner



Purpose- To spin the control panel in order to advance the levels in the game.

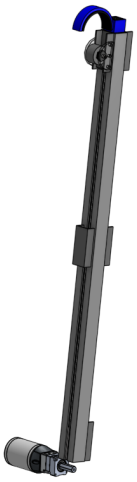
Rationale- We approached this task with one main goal in mind, that being to use one motor. We realized that for this to work, we needed to approach the panel from the side as opposed to having a wheel that goes on top of it. The shape and fixed placement of the control panel mechanism allows for it to always make contact with the control panel, even if we are slightly misaligned. The foam rollers used provided us the grip required to efficiently spin the control panel.

Description- Our spinner mechanism uses 2 rubber foam broom handle grip cut to 3.5” to spin the control panel.The rollers are mounted vertically with custom CNC cut ¼” lexan plates on the top and bottom. The shafts used are ¾” 3D printed hex centre pipe. The mechanism is mounted onto the robot’s subframe with a 3D printed 1 ¼” riser with bolts that go into the churros that connect the top and bottom plates of the mechanism. The actual spinning part that contacts the wheel are the foam broom handle grips overtop of the 3d printed sleeves that are on the output shafts.

Motor- For this mechanism we choose a 775 Pro Redline motor with a 16:1 gearbox, 1:1 gearing with 12 gears and #35 chain. We used the 775 redline motor to spin below 60 rpm while staying at a consistent speed to allow for an accurate automated performing of each spinning task.

Sensors- For each of the two rotating tasks, we use a Rev Robotics Color Sensor V3 to track each time a colour appears under the spinner. This lets our team know how many times the spinner has rotated without having to use encoders, so that friction is not an issue.

Climbing Mechanism



Purpose- To elevate the robot off the ground at the end of the game in order to obtain the extra end game points.

Rationale- We decided to set our climbing mechanism at a 10 degree angle to maximize the height that the climbing arm could extend while still staying within the horizontal frame perimeter. Our climbing mechanism was designed to deploy and retract quickly for a fast climb at the end of the match which is why we decided to use the vulcan spring. Additionally we used a mini CIM attached to a 100:1 planetary gearbox with a built in ratcheting system to hold the robot on the shield generator switch for the 5 second period before climbs are counted.

Description- The climbing subsystem's release mechanism is fully powered by a single Sh16p40 vulcan spring which is mounted atop the chute. For retraction, we use a mini CIM motor with two planetary gearboxes- including an engageable ratchet - giving us a gear ratio of 100:1. This is attached to the bottom of the shaft connecting to our climbing hook- a piece of ¼ inch steel pipe lined with rubber gasket that prevents swinging and sliding - and allows us to fully support our entire robot during the endgame. A magnadyne ALA50-DP linear actuator is used to pull a pin that stops the Vulcan spring from retracting and allows the climber to release. This design creates a fast acting climber that will take less than ten seconds to deploy. Our original plan was to use a pin locking mechanism to hold the arm in place but we switched to the ratchet to avoid strain on the pin and surrounding area.

Motor and Gears- Mini CIM with two ratcheted planetary gearboxes (gear ratio of 100:1)

Sensors - In the climber there are two hall effect sensors being used. These allow us to the approximate distance of the climber to the generator switch rung to ensure a more accurate climb. There is also a Microsoft lifecam HD-3000 camera located on the back of the shooter frame to aid the drivers in the positioning and accuracy of the endgame climb.

Control Systems

Overall Goals

Raider Robotics's Control Systems are built around a few principles that we use to guide our design decisions.

State Machines- A robot is a complex state machine which is made up of a collection of systems and subsystems working together to achieve desired actions. Raider Robotics designed our code by modeling the desired actions of any subsystem around a set of states that subsystem can be in and the transitions to move from state to state. Designing using states and transitions helps to focus coding and testing around expected actions and identifying unwanted or problematic actions.

Logging- All robot system data is automatically logged to a USB flash drive attached to the RoboRIO, and backed up to GitHub daily. This allows easy off-robot debugging of events during a match. This data can also be streamed real-time to any programmer's computer via TCP.

Simulation- All robot hardware is fully simulated. With the help of our simulation tools, all software is tested in full, in a simulated environment before being pushed to real hardware.

Control Loops- We make use of advanced control theory concepts to precisely and efficiently control all of the robot's systems.

Reusability- We have developed a robot-agnostic software library that contains many tools, controllers, and utilities to give programmers easy-to-use interfaces for advanced systems and control theory.

Robot Processors

This robot makes use of a network of devices to aggregate large arrays of data and control physical systems.

RoboRIO- The RoboRIO runs a custom real-time Linux distribution developed by National Instruments. It uses a custom Field Programmable Gate Array (FPGA) to instantly read sensor data into memory for ultra low-latency sensor inputs. It also handles data logging, coordinates and controls systems based on actions from the robot's operators and runs control loops with extremely precise timings.

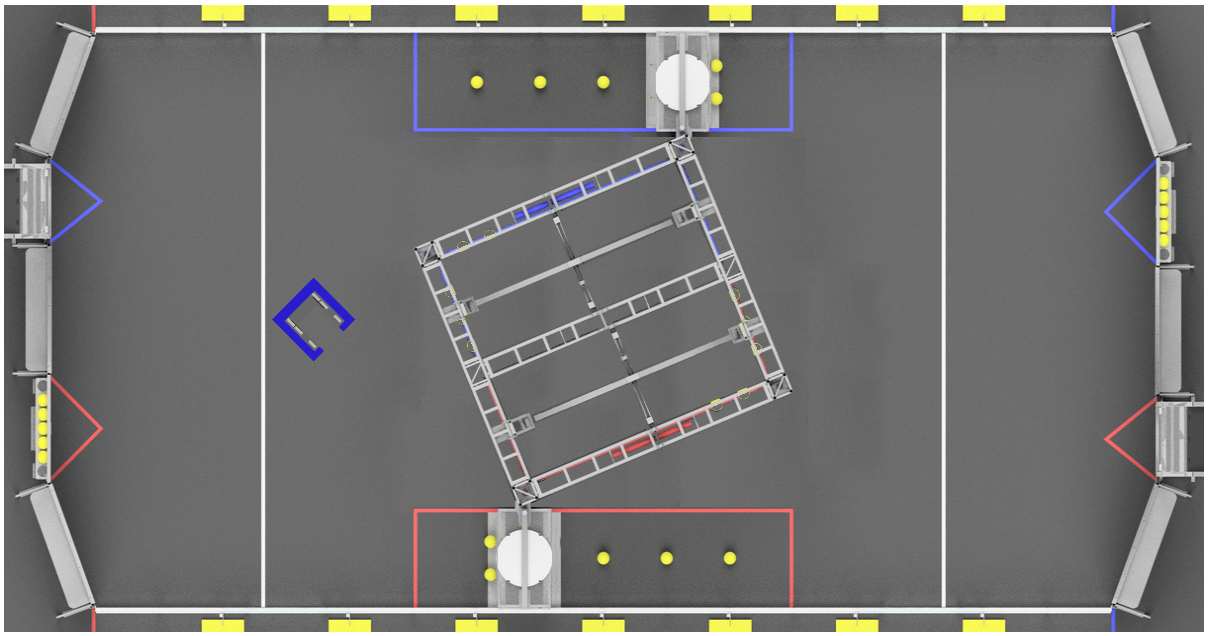
Limelight- The limelight handles all vision processing for the robot. It provides the robot with 3D pose calculations to feed into Shooter velocity control loops and drivebase targeting systems.It is used to re-calibrate the robot's software odometer periodically throughout the match

Laptop- The drivers laptop runs DriverStation software. It provides robot operators with a telemetry dashboard showing all important system info and camera feeds. This laptop is also used for recording a backup of all robot sensor data from the RoboRIO's network telemetry data stream for use in debugging and post-event data analysis.

Robot Simulation & Tooling

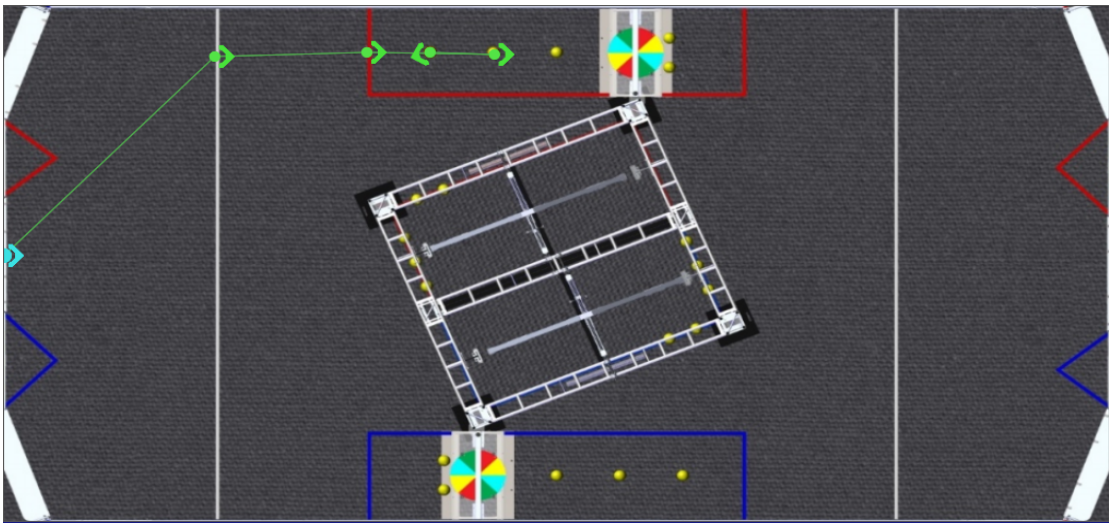
Our robot software can be packaged for both ARMv7 and x86_64 platforms. This allows us to run robot software locally on our development workstations for testing. To help with hardware testing, we have developed a few tools.

FieldSim- The FieldSim is a robot simulation tool developed in-house by Raider Robotics that allows programmers making Autonomous paths to quickly and safely test robot paths. The simulator can accurately display how the robot will perform in Autonomous. FieldSim works by reading odometry data that is produced by the simulated robot software and streamed through the robot's telemetry feed.



Point Planner- Point Planner is a tool designed by members of Raider Robotics. It is used to plan out paths and creates a blueprint for future autonomous paths. Point Planner provides programmers designing the Autonomous with points formatted in either .csv or .json. These exports provide information on the coordinates and rotation that the robot moves to in the field.

Using the exported file's data as reference, a programmer can design paths for the robot to follow in significantly less time and place the points more accurately. This allows for more time to be spent on refining and testing the path instead of figuring out point placement.



Drivebase Simulation- In order to simulate the robot's drivebase, we hook into the HALSIM software provided by WPILib and emulate the required sensors. Encoders are simulated with an RPM calculation passed through a simulated gearbox with some simple physics support to allow realistic momentum simulation based on the type of motors being used. Heading is calculated from a difference in simulated encoder velocities and a track width gain.

Robot Locomotion

During the match's Autonomous period and in Teleop during driver requested alignment and aiming actions, the robot's drivebase is able to very precisely move from position to position. This movement is performed by a variety of inverse kinematic controllers. The robot's most used controllers are High-Accuracy Real-Time Control and High-Speed Dynamic Path Navigation.

High-Accuracy Real-Time Control- For high-accuracy control at high speeds, we use a modified RAMSETE controller that works in the following way:

- Generates a desired path for the chassis to follow, given a start and desired end pose, along with information about the robot's chassis
- Splits this path up into frames, with each frame spanning a 20ms period in time
- Calculates desired track velocities for the differential drivebase to reach the current frame's desired state
- Determines the robot's positional error from its desired path, and feeds this back in to the controller

This feedback loop gives us the benefits of pre-planned path following, while allowing the robot to deviate from its path and correct in real time. As long as the odometer does not slip too far, the robot can be bumped during its autonomous pathing actions and correctly recover very quickly.

High-Speed Dynamic Path Navigation- For less accurate paths, or paths that require many interior waypoints and curves, we use a custom Pure Pursuit controller. This nonlinear controller allows us to reliably get the robot from point A to point B at very high speeds. Our Pure Pursuit controller, unlike the RAMSETE controller, does not pre-plan paths. This means that there is slight variance in the robot's path every time the controller is run, but the robot can react to very large disturbances such as being pushed, blocked, and getting caught on another robot.

Autonomous

How Autonomous was Designed- The Raider Robotics' Autonomous was designed with the goal of scoring the most points possible in the Autonomous period. These tasks were developed in coordination with the Raider Robotics' Strategy team to design the best autonomous possible based on the aforementioned goal.

How the Autonomous was Built- Every autonomous path is built by a group of "actions" (see Robot Locomotion above). These actions provide the robot with goal poses to reach through a mix of pre-planned, and real-time motion. The goal poses are plotted and tested during the build season. Each path interprets these goal poses to create a path for the robot to follow in real time, using various control systems. If the robot encounters a problem such as becoming removed from its set path, the robot will automatically adjust to continue towards the same goal position. This provides each path with more reliability in the situation where the robot is bumped by alliance members during the Autonomous phase.

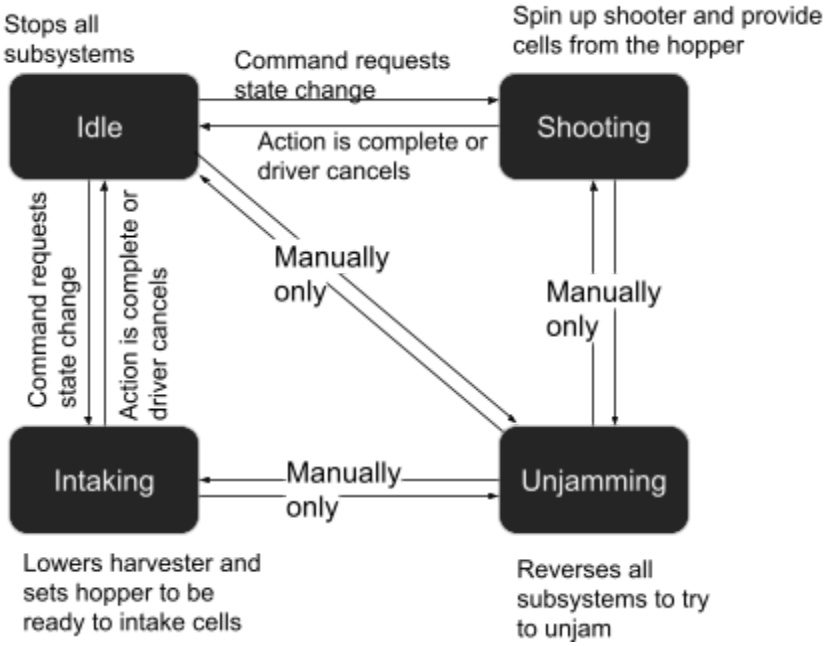
The Use of The Robot During Autonomous- *Darth Raider* has a variety of subsystems and sensors that are needed to perform a successful and effective Autonomous period, primarily the Intake, Hopper, Shooter, and drivebase mechanisms. The robot is able to control these mechanisms using commands that are created by each subsystem, specifically for use in the Autonomous phase. These commands allow a programmer to easily specify what the robot should do during the Autonomous period. If an autonomous path contains shooting, the robot is able to automatically lock on to the vision targets on the high goal, making scoring in autonomous extremely simplistic.

Handling Power Cells

The Superstructure encompases the Shooter, Hopper, and Intake subsystems. It manages higher level functionality such as “shoot 3 power cells”. The Superstructure is key to coordinating the actions of these three subsystems so that they work together seamlessly.

All three subsystems must be designed to allow the Superstructure to interface with them, and coordinate their actions. The most important part of the Superstructure is how the subsystems signal what they are doing. For example, when intaking the Hopper reports the amount of cells it owns. When the amount is large enough the Superstructure sees this, and stops the Hopper and Intake.

Design- Superstructure coordinates the three subsystems to work together using a state machine. Commands run during Autonomous or Teleop are able to request the state of the Superstructure to change. The Superstructure can also change its own state, for example, when the robot has shot all power cells, it will switch itself to idle.



Shooter

The Shooter subsystem receives power cells from the Hopper. It is responsible for launching the cells into the outer and inner power ports.

Components- The Shooter subsystem makes use of a REV Robotics NEO brushless motor and SPARK MAX motor controller to control a flywheel launcher. The launcher consists of a flywheel, as well as a hood with adjustable slats for calibrating the angle of launch. The Shooter relies on the Limelight camera for targeting information, as well as the Drivetrain and NavX gyroscope for target alignment.

Design- The Shooter can be in any one of five states:

IDLE:

- The flywheel velocity has been stopped.
- The Limelight is not being used.
- Subsystem is waiting until it is needed.

SPIN_UP:

- The flywheel's PID reference is set to the desired velocity, in RPM, calculated using the Limelight target information.
- The Limelight is used for reference determination.
- The flywheel spins up to the desired velocity, taking a minimum 0.3 seconds.

SPIN_DOWN:

- The flywheel spins down to idle over the course of 1.3 seconds.

HOLD:

- The flywheel maintains the desired velocity until further instructed, compensating for loss of speed from shots.

UNJAM:

- The flywheel reverses until the operator requests a stop.

Intake

The Intake subsystem takes power cells from the front of the robot, into the bottom of the Hopper.

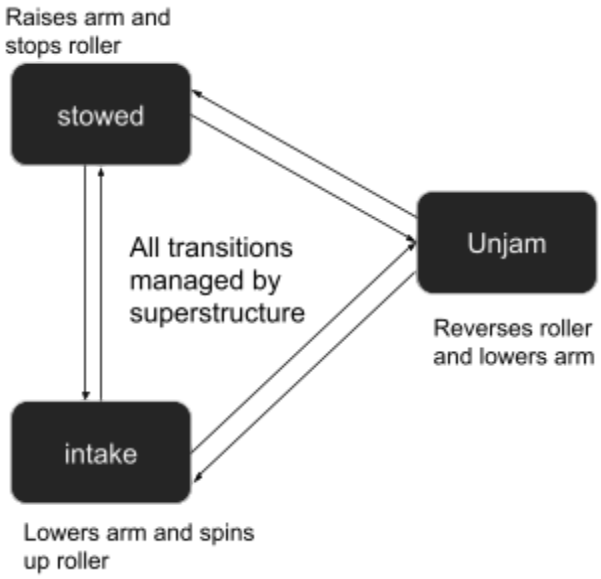
Components- The Intake consists of two motors - one to raise and lower the Intake arm, and one to move the Intake roller that pulls the power cells inwards. It also uses limit switches to determine when the Intake arm is at its top or bottom positions.

Design- The Intake may be in three states. It relies on the Superstructure to set these states.

INTAKE_CELLS
- Arms move to the bottom position and the rollers spin inwards

UNJAM_CELLS
- Rollers reverse to get any power cells out of the robot and so they do not all stay in the bottom of the robot.

STOW_ARM
- Rollers stop and the arm moves to the top position and will be actively controlled in order to stay in the top position.



Hopper

The Hopper subsystem receives power cells from the Intake, stores the power cells, then passes the cells to the Shooter when needed.

Components- The Hopper uses one motor to control the movement of the belt which moves power cells up and down.

To control the movement of the power cells, the Hopper uses 3 line break sensors:

- The first line break sensor is located at the bottom of the Hopper and indicates whether or not the Hopper should attempt to take a power cell in. This logic makes sure we aren't spinning the belt unnecessarily.
- The second line break sensor is located at the point where as a power cell goes past it. It indicates a power cell has moved far enough into the Hopper to have enough spacing between it and the next cell. The spacing is necessary to prevent the cells from binding together.
- The third line break sensor is located at the top of the Hopper and is used to determine when possession of a power cell has switched from the Hopper to the Shooter.

To enable precise movement of the cells the Hopper belt motor uses an encoder. The encoder lets the Hopper move cells an exact distance in order to maintain a consistent spacing between cells.

Design- The Hopper can be in any of the 4 states the Superstructure can be in:

INTAKING

- The Superstructure requests the Hopper to Intake a specific amount of cells (defaults to 5).
- The Hopper waits for the first line break sensor to trip, then moves the cell a predetermined distance using the encoder, and confirms the cell has gone far enough by using the second line break sensor.
- This process repeats until signaled by the Superstructure, or until the amount of cells has been taken in, reporting to the Superstructure that it is complete.

SHOOTING

- The Superstructure requests a certain amount of cells to be shot.
- The Hopper waits for the Shooter to spin up and report that it is ready to shoot.
- The Hopper supplies a cell, using the third line break sensor to tell when the cell has been passed off to the Shooter.
- If the Shooter is not ready then the Hopper will pause supplying cells until signaled by the Shooter.
- This state ends either through a signal from the Superstructure to cancel, or the Hopper can report that the desired amount of cells has been shot.

UNJAMMING

- The Hopper belt reverses upon request.
- Used to remove or attempt to unjam any balls within the Hopper.

IDLE

- The Hopper belt is stopped and the cells are kept in place inside the Hopper.

Control Panel Interaction

The Control Panel system is used to spin the field's control panel apparatus to one of two goals:

Rotational

- The field's control panel must be fully rotated at least 3 and no more than 5 times.
- This represents the transition into Stage 2 of the match

Positional

- The field's control panel must be rotated to a field defined color.
- This represents the transition into Stage 3 of the match

Components- The components that make up the Control Panel are a Rev Robotics ColorSensorV3, a single motor connected to two sprockets that rotate two felt covered PVC tubes, and a 3D printed mounting bracket.

Design- The subsystem is controlled by the operator.

IDLE

- Subsystem will always start in IDLE mode.
- During IDLE, the subsystem will loop each cycle to check if the ColorSensor's range detection is greater than 200 representing that the ColorSensor is over top of the field's control panel.
- The ColorSensor gives an arbitrary number from 0-2048 to show what the distance is. A larger number means closer to the sensor.

ROTATIONAL

- Represents the mode used in stage 2 of the match.
- The panel must rotate 360 degrees more than 3 but less than 5 times. To do this, an enum has been created to detect which color is under the sensor and expected color at the current cycle. If at the next cycle, the current color, is the same as the expected next color it will add one to a counter and continue again.
- Once the counter has reached 28 or greater, a flag will be set to true to tell the code that stage 2 has been completed. (28 gives some leeway in the count)

POSITIONAL

- Represents the mode is used to go to a specific color, defined by the FMS each match.
- The target color is sent via the GameSpecificMessage (GSM) which returns a single character.
- Once the subsystem has a GSM, the robot will convert that to a color object to match to.
- Once the color is matched, the subsystem will always be in the idle state.

ERROR

- Represents the case where the robot moves out of range of the panel or the operator locks the subsystem.
- All the values like the counter, current color and next color will stay the same the next time the subsystem is unlocked.

Endgame: The Climber

The Climber is used to hook onto the shield generator and pull the robot up.

Since our climber is a “one and done” this influenced the design of our climber, for us to climb we are using a hook on the top of our climber to help us “hook” on without having the risk of falling off the bar. Our goal is to climb in the last 10 seconds of Endgame to maximize our time scoring on the field. Having a “one and done” climber, comes with the risk of us not hooking on properly because mainly the drivers don’t have the best view of the rendezvous point. For that, we have a camera at the back of the Shooter facing up displayed on the driver station that will only be turned on during Endgame for our driver and operator to have a better chance of lining up and climbing without many problems and quicker.

Components-

Linear Actuator Pin

- Used to lock the Climber’s arm in a retracted position until needed for the climb
- To climb the pin will release and the Climber’s arm will fully eject.

Motor

- Used to retract the Climber’s arm.

Hall-effect Sensors

- Used to determine if the Climber has reached its goal position.
- There are three sensors:
    - Arm fully ejected.
    - Arm retracted to the point of a level climb.
    - Arm fully retracted. This is used as a guard to prevent the Cimber’s arm from pulling back too far into the frame.

Camera

- Used to help the drivers orient themselves during the climb.
- When the operator starts the climb, the camera will switch on
- The camera will switch off when the robot has completed its climb.

Design- System States are used to determine the state of the Climber. There are four states the robot will cycle through:

SERVICE

- Dued when the robot is being worked on in the pits.
- The lift-motor is disabled, the climb camera is enabled, and the release pin is locked.

LOCKED

- Locked is used when the robot is on the field before Endgame.
- The lift-motor is disabled, the climb camera is disabled, and the release pin is locked.

DEPLOYING

- Deploying is used when the robot is on the field during Endgame.
- The lift-motor is enabled, the climb camera is enabled, and the release pin is unlocked.

RETRACTING

- Retracting is used when the robot is climbing.
- The Climber is reading the desired destination sent by the operator, the climb camera is enabled, and the release pin is unlocked.
- The climber will stop at the desired destination.

Climb positions are used to determine where the climber is located. There are four positions the robot will cycle through:

Current

- Current is the current position of the Climb arm.

Extended

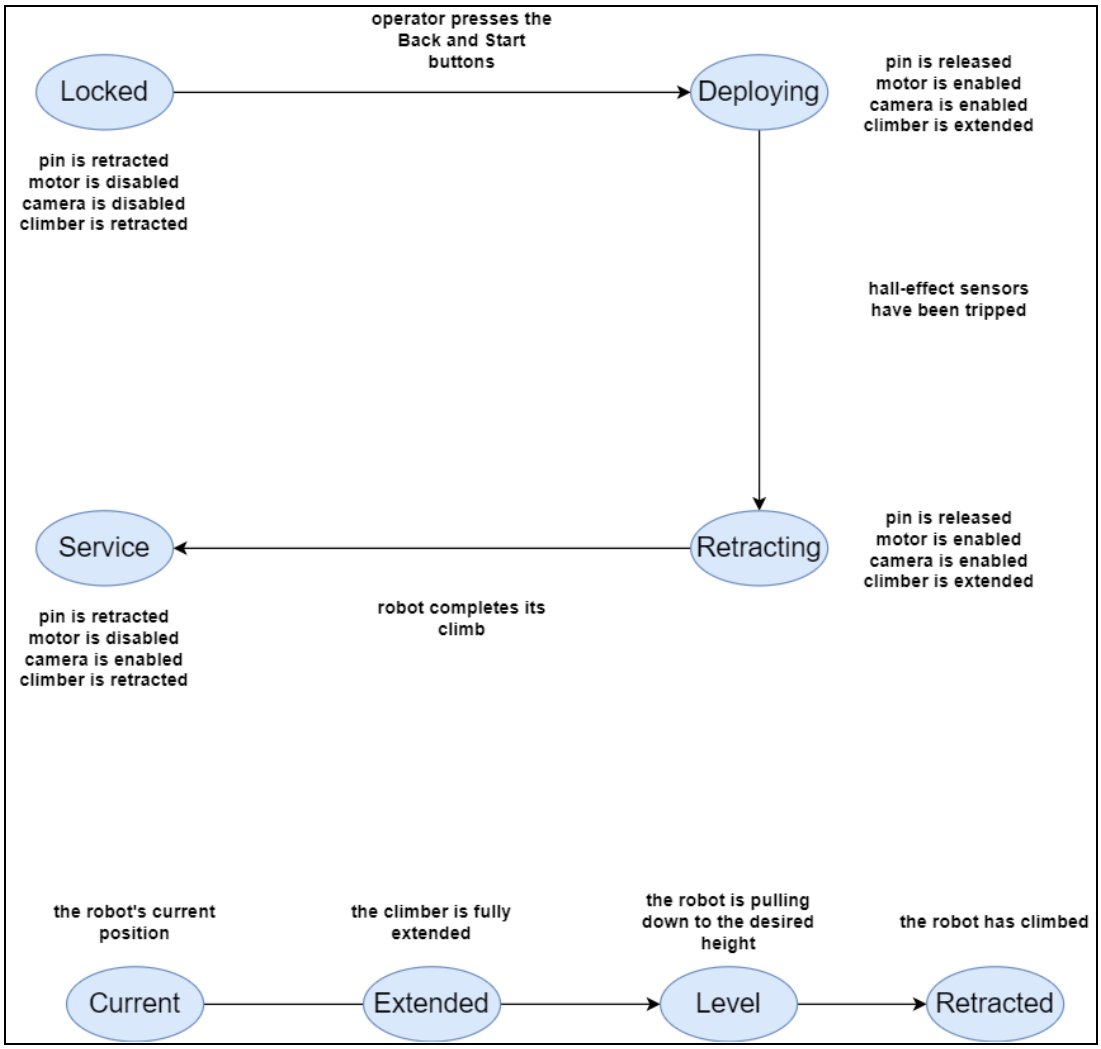
- The maximum height of the Climb arm. The climber will reach this position when it is released by the pin.

Level

- Level position of the Climb arm represents a point where the arm is able to hook on to the Power Station bar when it is in a Level position

Retracted

- Retracted position represents the point where the Climb arm has pulled the robot off the ground.
- Climb arm should not pass beyond the Retracted position as it will pull the arm too far into its frame potentially causing physical damage.


operator presses the Back and Start buttons
Locked
Deploying
pin is released
motor is enabled
camera is enabled
climber is extended
pin is retracted
motor is disabled
camera is disabled
climber is retracted
hall-effect sensors have been tripped
Service
Retracting
pin is released
motor is enabled
camera is enabled
climber is extended
pin is retracted
motor is disabled
camera is enabled
climber is retracted
robot completes its climb
the robot's current position
the climber is fully extended
the robot is pulling down to the desired height
the robot has climbed
Current
Extended
Level
Retracted

Appendix

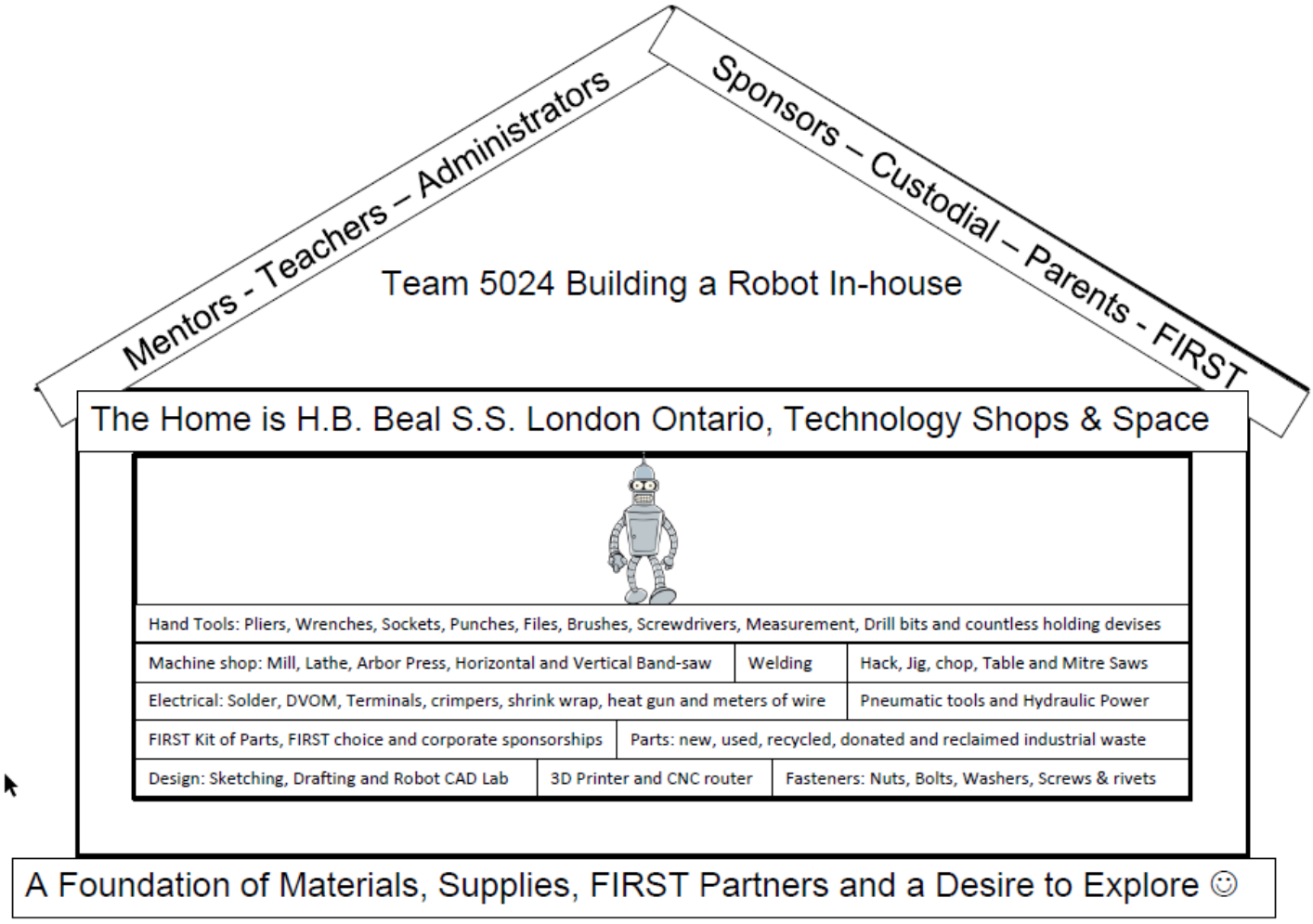


Figure 1. Our robot is built completely at the school. Above is a list of the available tools, parts, and equipment

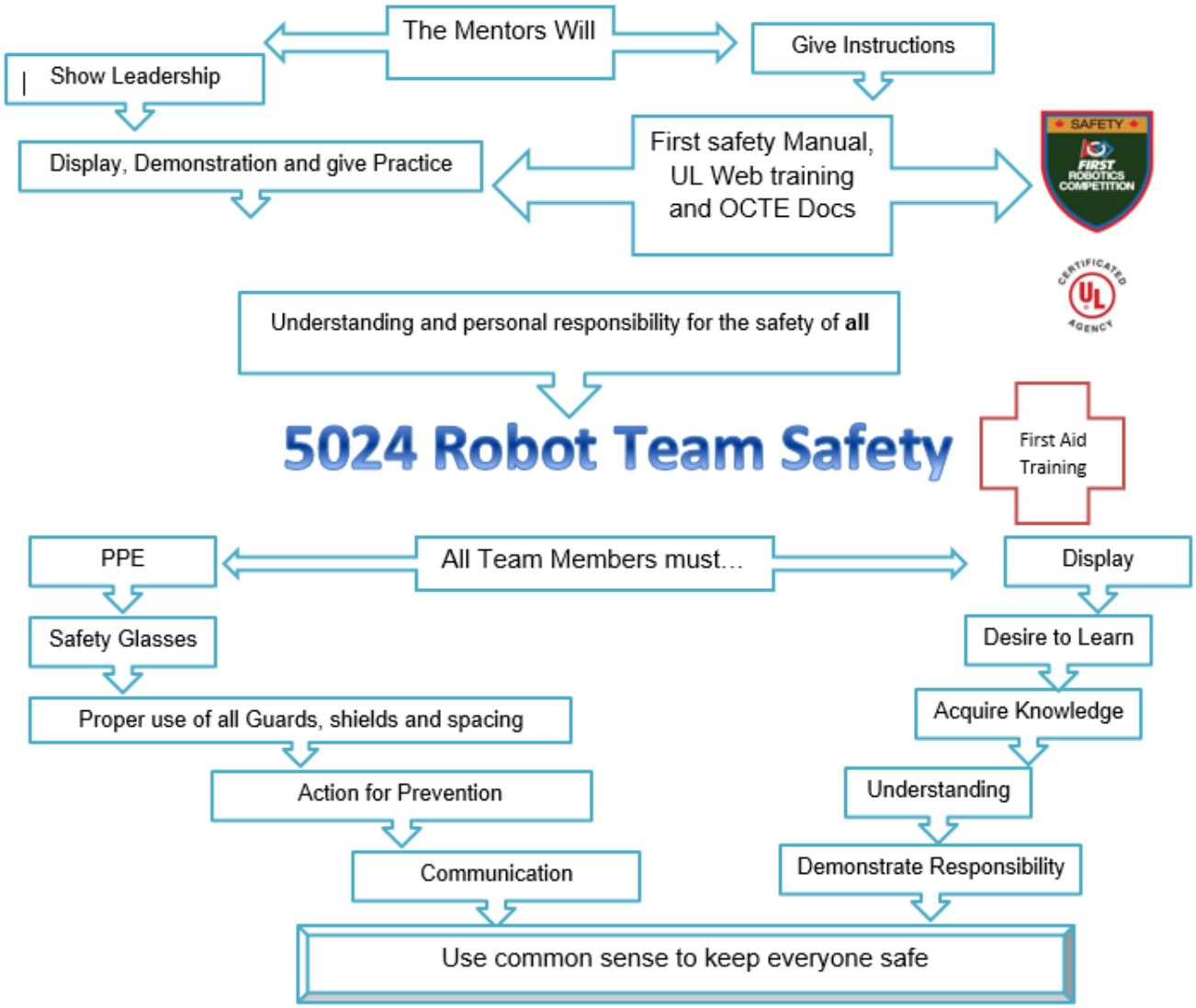


Figure 2. All students within our mechanical team must be trained in safety procedures. The flow chart above illustrates expectation of all members on the team.