2019 Technical Documentation

Team 3082
"Chicken Bot Pie"
Minnetonka High School
Minnetonka, MN

TABLE OF CONTENTS

ROBOT OVERVIEW

Dimensions
- Weight: 123 lbs
- Width: 29.5”
- Length: 29.75”
- Height: 46.5”

Drivetrain
- Swerve
- 4” Traction Wheels
- Speed: 13 ft/s

Elevator
- 2 Stage Cascade
- Four Bar Arm
- 1 sec to max height

Intake
- Hatch Panels
- Cargo
- Cargo from floor
- Scores everywhere

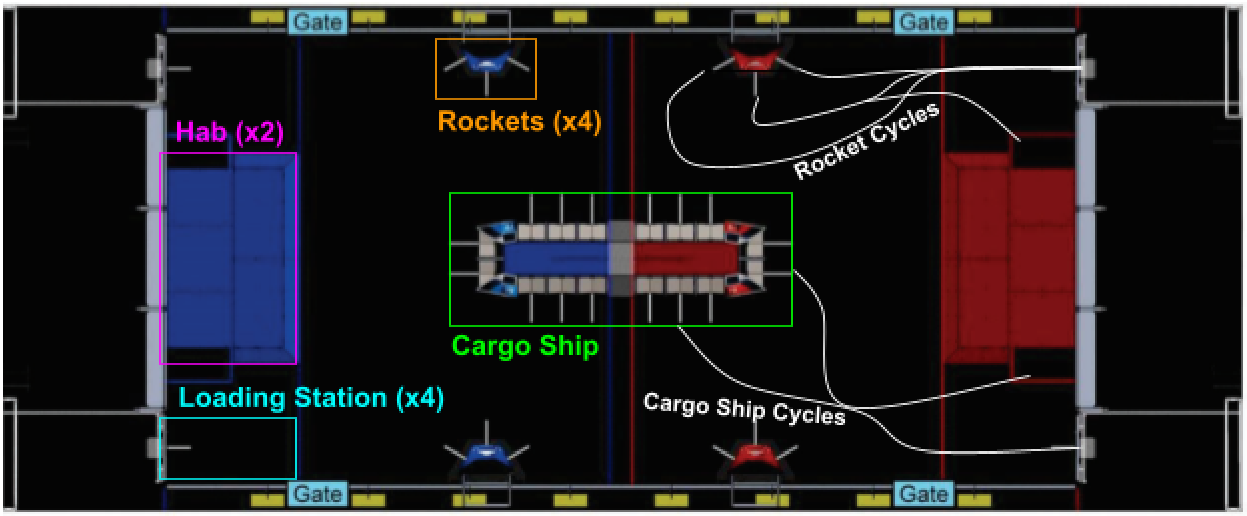
We are proud to present our 2019 robot: Aurora!

STRATEGY

- Destination: Deep Space
- Our Strategy
- Priority List and Design Goals

STRATEGY Destination: Deep Space

After reviewing the 2019 game, *Destination: Deep Space*, we identified a number of key tasks that robots would have to accomplish. These are:
Driving to navigate the field
Collecting CARGO and HATCH PANELS from the loading stations and floor
Scoring CARGO and HATCH PANELS into the CARGO SHIP (close to the ground) and ROCKET (various heights; up to 8ft)
Climbing the HAB PLATFORM to various heights (L1 - platform; L2 - 6” above platform; L3 - 19” above platform)



The 2019 Game Field

There are 20 total HATCHES on each half of the field. 8 of them are on the CARGO SHIP in the center of the field, and 12 are split equally between two ROCKETS on each side of field. A hatch panel (or a null panel) must be placed in each hatch before a CARGO can be scored in it.

In addition to the 2 ranking points (RPs) given for a win, there are two other available RPs. One is granted for completing one ROCKET, and one is granted for scoring at least 15 HAB climbing points.

STRATEGY Our Strategy

We set our goal this season to compete in the playoffs at both events that we attend (Great Northern Regional and North Star Regional).

It seemed feasible that one robot could control their destiny with an extremely reliable L3 climb, as it would almost guarantee the HAB ranking point every match (scoring 12 of the 15 required). Comparatively, the ROCKET ranking point seems more unreliable due to the high number of cycles required (12 total) and the susceptibility to defense.

The ideal robot would be able to secure both additional ranking points reliably by filling (or almost filling) a ROCKET on its own and climbing to the L3 HAB. Knowing our abilities, resources, and constraints, we realized that we would likely have to pick either an L3 climber or a ROCKET scorer.

We expect this to be a relatively common alliance at MN events:
Captain Low cycler & L3 HAB (ranks high due to HAB ranking point)
First Pick High cycler (picked due to additional scoring potential)
Second Pick Additional low cycler & defense (fills defensive niche)

With this in mind, we decided to build our robot to be the most attractive first pick possible. We led our strategy towards scoring in the high levels of the ROCKET without an L3 climb.

If indeed L3 HAB climbers rank high, great! They probably would not want to pick another L3 HAB climber because there likely wouldn't be room for another one on the platform. That would make us an attractive pick with the ability to score in locations that they cannot. We could also be wrong -- ROCKET scorers could actually rank high. In that case, either we rank high ourselves, or we are still a solid pick to score on the other rocket. Therefore, this strategy aligned the best with our goal to play in playoffs at both Great Northern and North Star.

STRATEGY Priority List and Design Goals

Strategic Priority List

- Driving
    - Maneuver quickly and efficiently throughout the field
    - Correct spatial misalignment when collecting and scoring
    - Optimize gearing for sprint distances of 15-25 ft for rocket
- Collecting
    - Collect CARGO and HATCH PANELS from the loading stations
    - Collect CARGO from the floor
    - Retain ability to collect when not perfectly aligned
- Scoring
    - Score both game pieces at all heights
    - Precisely place game pieces without dropping them and wasting cycles
- Climbing
    - Reach HAB Level 1 every match

Associated Robot Design Goals

- Swerve Drivetrain
    - Outmaneuver defenders while retaining traction for pushing matches
    - Move in any direction if/when misaligned from scoring locations
    - Optimal ratio: 4 CIMs @ 6.51:1 on 4” wheels
- Manipulator / Wrist
    - Collect both from loading stations, CARGO from floor
    - Combined manipulator to collect both game pieces
    - Wrist to swap intake between game pieces
- Elevator / Arm
    - Two-stage elevator to reach top rocket level
    - Four-bar arm for extra extension while maintaining constant wrist orientation
- Passive Endgame
    - Ground clearance to avoid beaching

MECHANICAL

- Overview
- Prototyping and Iteration
- Drivetrain
- Elevator and Arm
- Wrist and Manipulator

MECHANICAL Overview

This year, we went for the most aggressive and complex robot design in our team's history. With a swerve drivetrain and a superstructure that integrates three degrees of freedom, we had to be more careful with overall system integration.

We used an effective blend of COTS and custom components. We were fortunate to have one of our sponsors, Abbott, fabricate our chassis rails using CNC equipment in their machine shop. In our shop, we were able to manufacture the other custom parts using a bandsaw, a manual mill, and hand tools. To facilitate the creation of custom parts, we started to make engineering drawings for the first time. Previously, we made parts directly from CAD models, but learned that engineering drawings are the standard method used in industry.

For the sake of rapid assembly, we used COTS components wherever possible. However, some FRC-specific COTS parts have recently seen significant price increases so we manufactured them on our own. For example, we cut gusset plates on a CNC plasma cutter we restored over the summer.

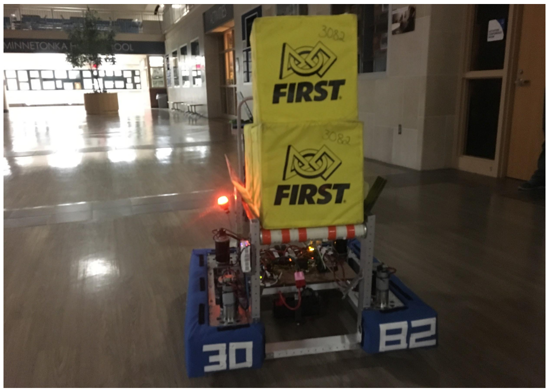
Mechanical by the Numbers

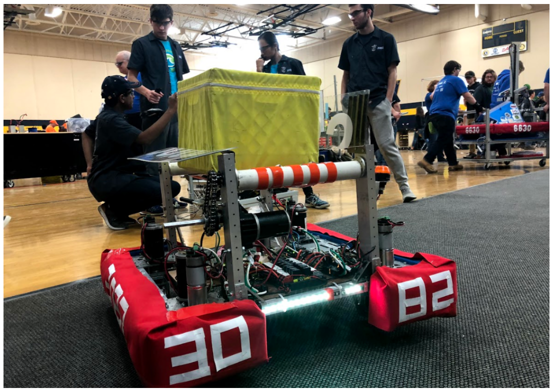
632 rivets

39.75 feet of aluminum tubing

2976 CAD components

MECHANICAL Prototyping and Iteration





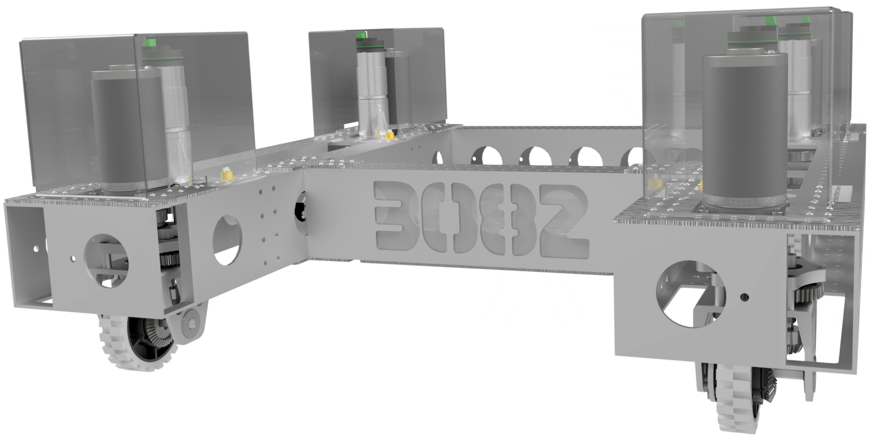
Off-Season 2018 Swerve Robot

Our journey with swerve started in the 2017 offseason. We didn't really know what we were doing at the time and had very unreliable software, so we didn't use it for the 2018 season. However, in the 2018 offseason we committed to building a full swerve robot. We took it to the Minne Mini offseason event and ranked dead last due to a really horrible Power Cube manipulator. We learned a lot from assembling this robot and applied those lessons to our 2019 robot.

Having a drivable chassis with a solid software base gave us the confidence to pursue swerve in-season. If your team is considering swerve drive, we highly recommend pursuing it in the offseason first as there is much lower risk.

Throughout the season, we ran through multiple iterations of key robot components. Two main examples are the manipulator and arm transmission. At our Week Zero competition, we learned that our "Hook Tape" Hatch Panel manipulator had issues collecting from the loading station. On top of this, the arm transmission mount severely bent. With no way to tension the chain, we were left unable to use the arm. We completely re-engineered the arm transmission so that it is attached more robustly and so that the chain can easily be tensioned with a Teflon idler.

MECHANICAL Drivetrain

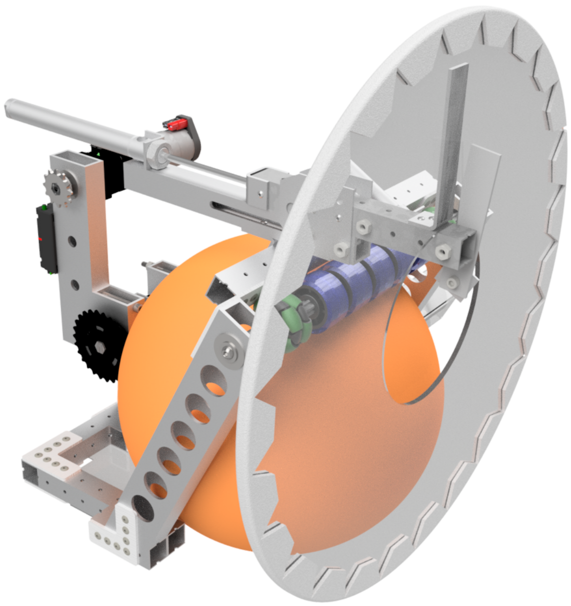


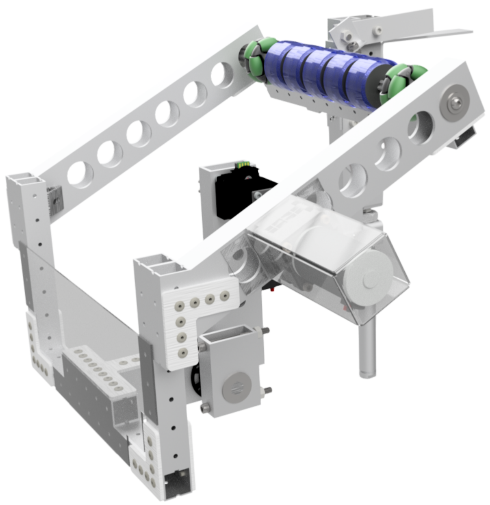
- Design Constraints
  - Maintain frame perimeter <120”
  - Maximize wheelbase size to improve stability
  - Avoid beaching while retaining a low center of gravity
- Frame
  - 4” x 2” x 0.125” Aluminum 6061 Tubing
  - Overall size: 29.50” L x 29.75” W
  - ~3.25” ground clearance
  - LED-illuminated team number for increased performance
- Drive
  - AndyMark Swerve & Steer Modules with added encoders
  - Custom main support plates to aid mounting into chassis
  - 13.9 ft/s top speed; 90 RPM steering speed
  - 4x CIMs @ 6.67:1 drive; 4x RS775s @ 59.3:1 steering

MECHANICAL Elevator and Arm

- Design Constraints
  - Stow within the 48” maximum starting height
  - Reach top level scoring locations in less than 1.5 seconds
  - Minimize backlash for more reliable control
- Elevator
  - Two stage cascaded elevator
  - 50 in travel, 120 in/s linear speed
  - Counterbalanced with 2x 20lb constant force springs
  - 2x NEO Brushless Motors @ 4.67:1
- Arm
  - Four bar arm preserves end effector orientation
  - Provide additional horizontal and vertical extension
  - Counterbalanced with 2x 20lb gas shocks
  - 1x 775pro @ 128.33:1

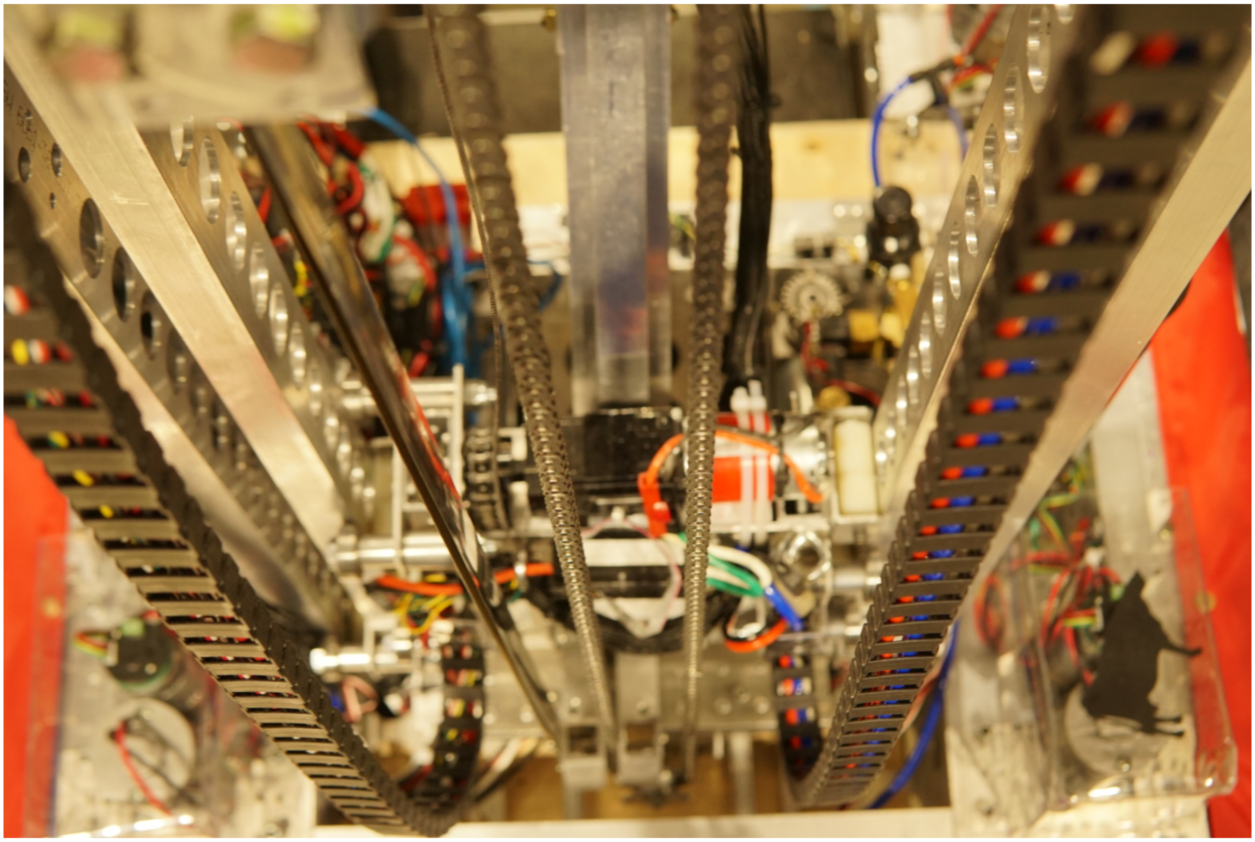
MECHANICAL Wrist and Manipulator





- Design Constraints
    - Stow within the frame perimeter
    - Collect CARGO and HATCH PANELS with the same mechanism
    - Secure game pieces with tolerance for misalignment
- Wrist
    - Rotates manipulator to collect and score both game pieces
    - 90° travel
    - 450 deg/s rotational speed
    - 1x 775pro @ 200:1
- Manipulator
    - 2” omni and traction wheels to center CARGO
    - 1x BAG Motor @ 7:1
    - Pneumatically actuated clamp to handle HATCH PANELS
    - Extends outside of frame perimeter to collect/score

ELECTRICAL



- Overview
- Schematics
- Component Distribution

ELECTRICAL Overview

This year, we went for a distributed electronics approach rather than a centralized one. Due to the spread-out nature of the swerve drivetrain and the superstructure, we placed electrical components, especially motor controllers, closer to their associated actuators.

This robot has the most actuators and sensors of any robot in our team's history. Part of this is the swerve drivetrain which comprises eight motors and twelve encoders. However, our superstructure with five motors and four encoders is also quite complex. We are also using two cameras for vision processing and a Pigeon IMU for robot heading correction.

We used two of the new NEO Brushless motors on our elevator, and we are pleased with the power to weight ratio and the built-in sensors. However, we did not opt to use them on the drivetrain due to them being brand new this year. We're excited to see the future of BLDC motor technology on FRC robots in the future!

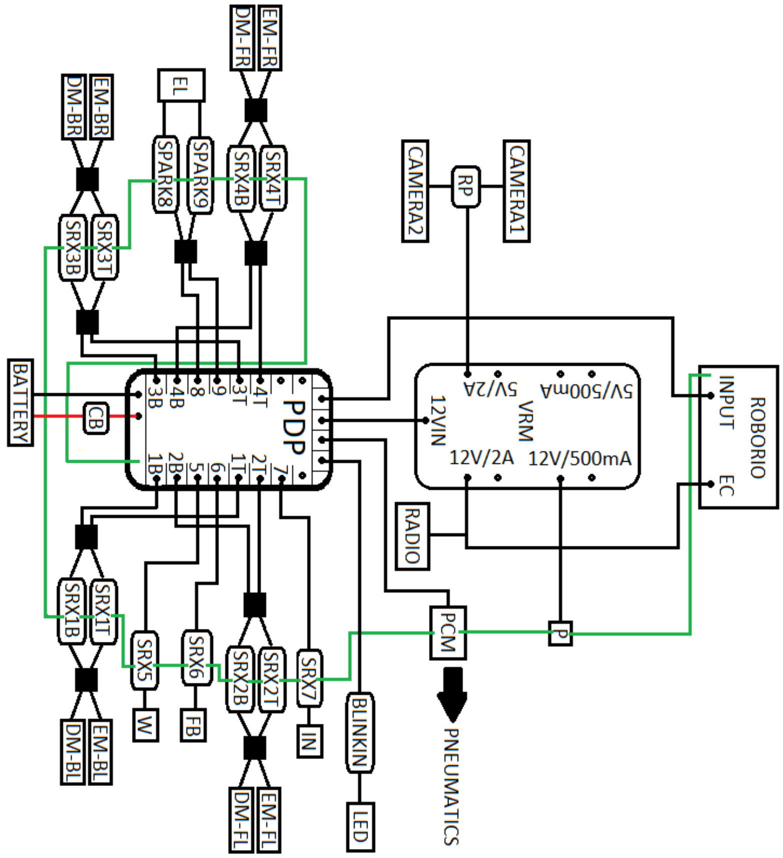
Electrical by the Numbers

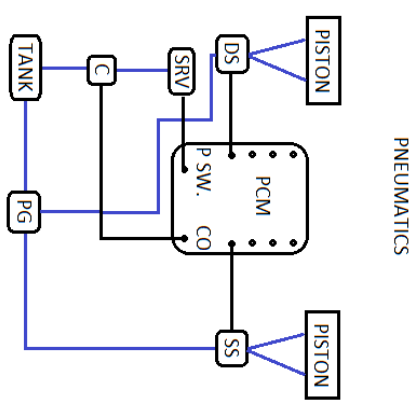
17 actuators

60+ feet of wire

135 LEDs

ELECTRICAL Schematic





KEY:

- EC Ethernet Cable
- P Pigeon
- EM Exterior Module
- DM Drive Module
- R Right
- L Left
- F Front
- B Back
- CB Circuit Breaker
- SS Single Solenoid
- DS Double Solenoid
- SRV Safety Release Valve
- C Compressor
- PG Pressure Gages
- CO COMPRESSOR OUT
- P SW. PRESSURE SW.
- RP Raspberry Pi
- Tube
- CAN
- Anderson Powerpole

ELECTRICAL Component Distribution

- Battery
  - Placed centered and low to optimize weight distribution
- PDP
  - Centralized vertical PDP placement to minimize wire lengths to high-power components
- 2 Cameras
  - Elevator fixed frame
    - Primarily used for driving during Sandstorm
  - Intake
    - For vision processing and driver alignment
- Swerve Controllers
  - 4 Talon SRXs for Steering and 4 Victor SPXs for Drive
  - Located near the modules to minimize encoder cable length
  - Hidden in the chassis to protect them during matches
- Elevator Controllers
  - 2 Spark MAX driving NEO Brushless motors
  - Mounted on the bellypan near the brushless motor gearbox
- Arm Controller
  - Talon SRX
  - Mounted on elevator carriage to minimize sensor cable length
- Wrist Controller
  - Talon SRX
  - Mounted at the end of the arm to minimize sensor cable length
- Pneumatics
  - Centralized Pneumatics Board
    - PCM, Compressor, Pressure Switch, Regulators, Gauges, Storage Tank, Release Valve.
  - Solenoids
    - Located on intake to minimize post-solenoid tubing, which minimizes air loss due to solenoid venting

SOFTWARE

- Overview
- Swerve Control
- Superstructure Control
- Vision and Driver Assist
- Driver Station

SOFTWARE Overview

Our main robot software is written in Java, and our vision software is written in Python (runs on a Raspberry Pi 3). When writing the software, we always make sure to try and take the most efficient approach to our problems. Keeping with the theme on the rest of the robot, this year was our most ambitious year ever for software. With a complicated drivetrain, a complicated superstructure, and more sensors than ever, we needed to have a solid software backing to make it controllable.

Code structure wise, we create static management classes for the smaller individual parts of our robot and bigger static management classes for dealing with the calculations of the repetitive parts of our robot. By using a mix of Functional programming and Object-oriented programming, we never have to deal with the hassle of large variable names. At the same time, we can keep our more complex structures kept to their respective management classes and away from our main class. For example, our four swerve modules and our multi-actuator superstructure are each separated.

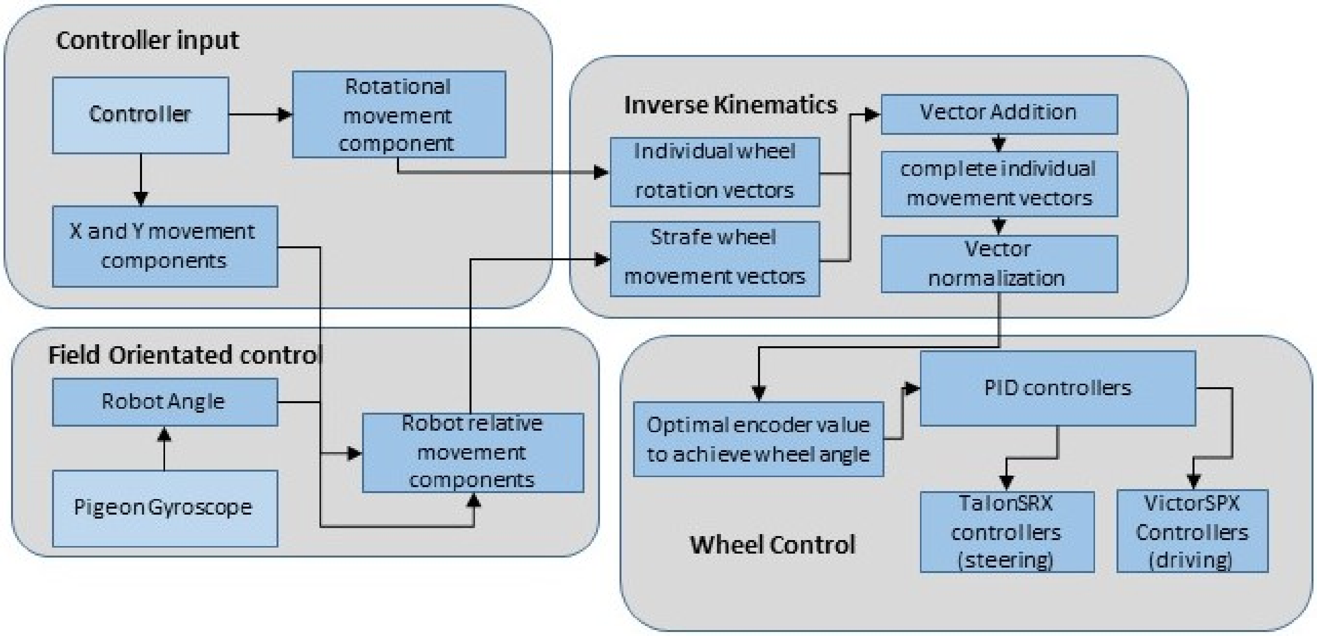
Software by the Numbers

4217+ lines of code

24 PID control loops

31 buttons

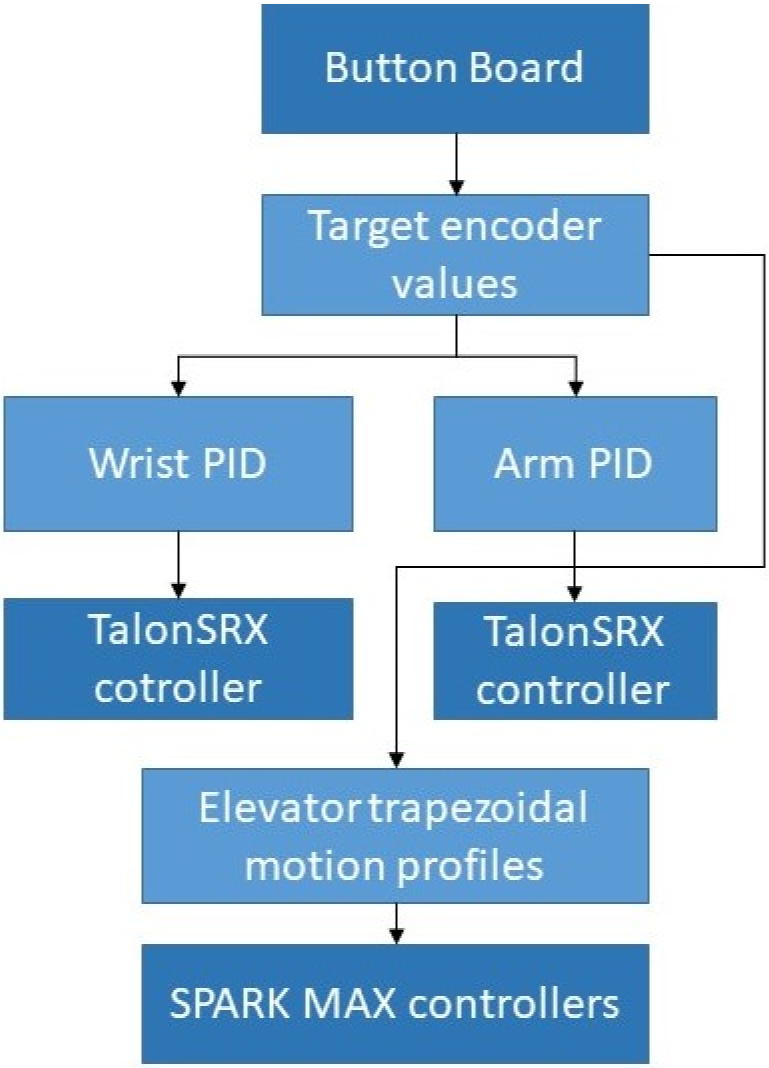
SOFTWARE Swerve Control



Our swerve drivetrain this year needed a lot of software work to get it to drive at all.

We experimented with an alternate control system that called turntable control that allows drivers to directly control the heading of the robot instead of controlling the rotational velocity of the robot. This system would have greatly simplified our autonomous code and would have made robot movement more accurate. Unfortunately, due to the removal of encoders on our drive wheels we cannot use turntable control this year.

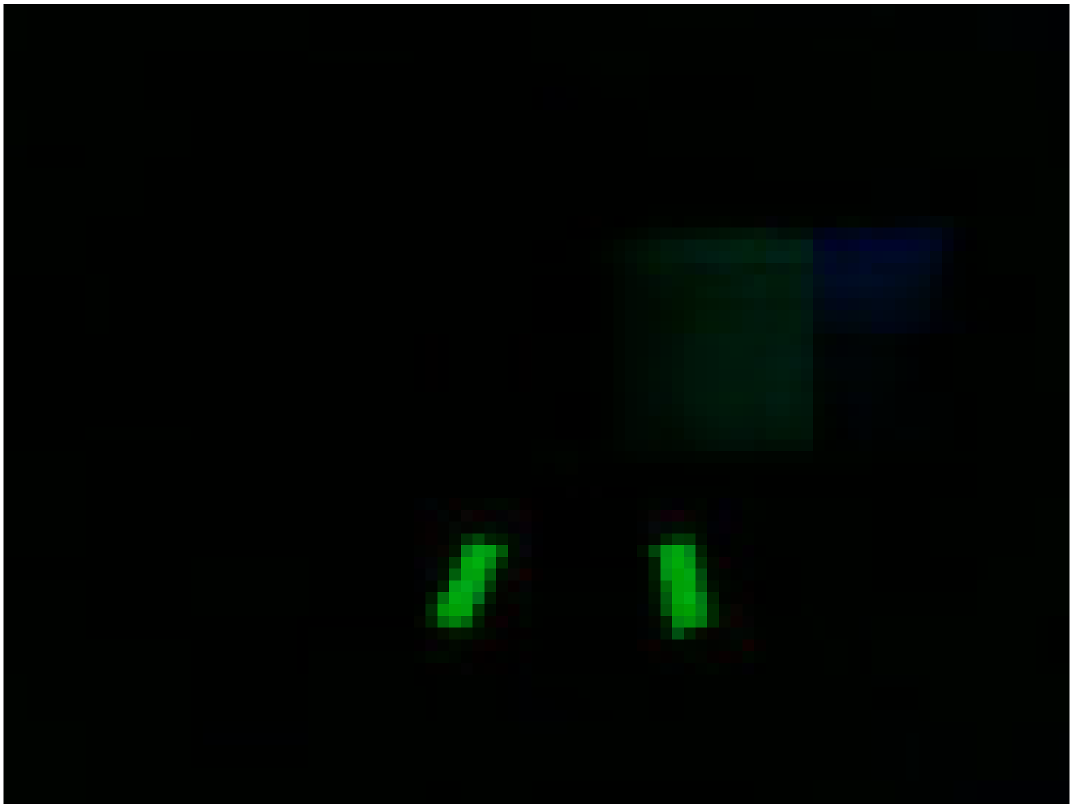
SOFTWARE Superstructure Control



Our intake (not pictured above) uses a VictorSPX to spin the wheels forward or backward, depending on the button pressed. The hatch panel intake is also controlled by buttons that fire solenoids to extend the pistons.

SOFTWARE Vision and Driver Assist

We use a Raspberry Pi 3 coprocessor to run our vision processing software. It collects data from a Microsoft LifeCam HD3000 camera and interprets the image using OpenCV to find the location of the retro-reflective tape on the scoring locations and feeder stations. We then send this information to the roboRIO using Network Tables, which is then interpreted to align the robot to the vision targets.



SOFTWARE Driver Station

With so many different actuators, we realized that it would be important to simplify the operator controls so that there are various preset positions for the different scoring locations. We could have done this on an off-the-shelf controller or joystick, but that would have led to having buttons all over the place. As a result, we built a custom button panel with buttons that are spatially located in a more intuitive way. It is powered by a Cypress PSoC microcontroller board.