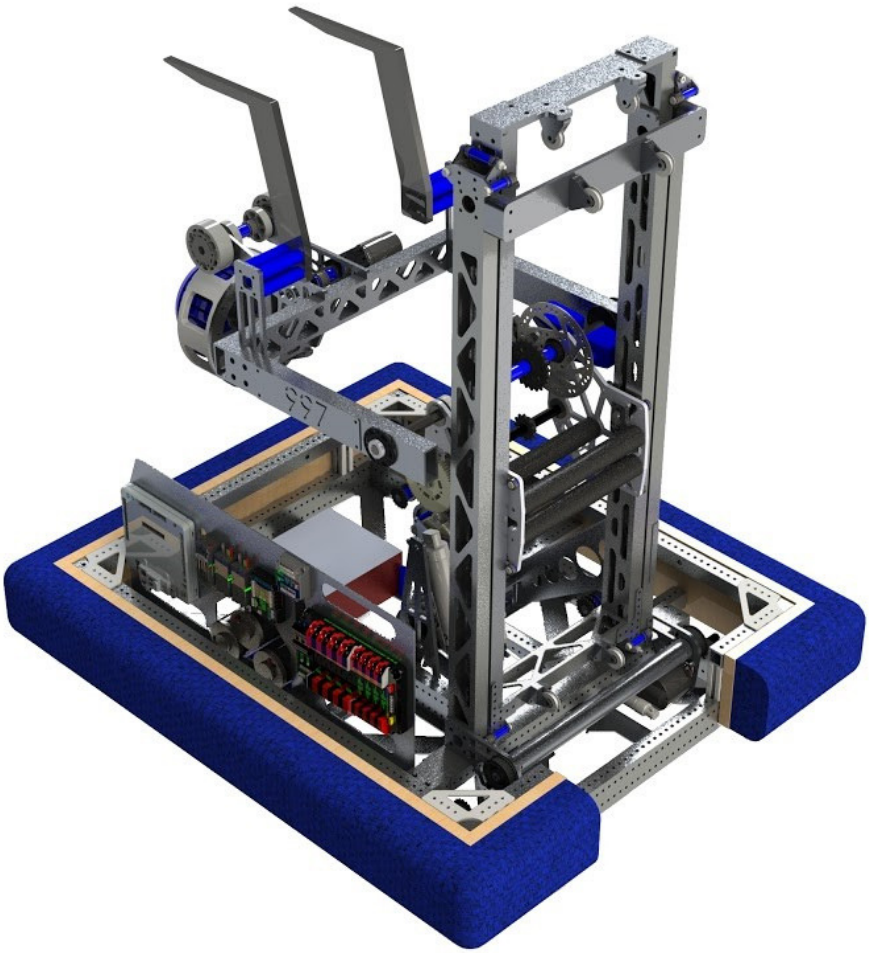
TEAM 997

SPARTAN ROBOTICS

OUR ROBOT ETHER



OVERVIEW

05

STRATEGY SELECTION

Dive into the four major strategies used by our team and how we collaborated and made a final decision.

10

DESIGN AND MANUFACTURING PROCESS

Explore the six main subsystems of our robot and how they were designed, revised, and finalized.

41

SUBTEAMS AND HIGHLIGHTS

Understand the way our sub-teams work together to create a professional and functional final product.

TABLE OF CONTENTS

TABLE OF CONTENTS

TEAM 997

OUR STRATEGY

In order to determine what strategy to use, we split up into multiple strategy groups. We created six student groups and a mentor group. Every student group was assigned a primary and secondary student leader. Each group brainstormed their own ideal robot strategy and alliance strategy, using mathematical and scenario-based analysis. Afterwards, we regrouped to present and condense the strategies. This allowed us to consider several different strategies before making a decision.

STRATEGY ALPHA

During the Sandstorm period, our robot will drive off of HAB level 2, place one hatch on the nearest cargo ship bay to hold in a piece of cargo, and collect the cargo from the adjacent cargo ship bay. After the Sandstorm, the robot would focus on completing a rocket by placing the hatches and filling the cargo on all levels. Toward the end of the match, the robot would return to the HAB Platform and climb to Level 2.

PROS

- Completes all main game tasks
- Potential to earn an additional ranking point by completing a Rocket on our own
- Generally self-reliant
- Lower climbing commitment
- More flexibility as an alliance member
- Lots of potential fallback options

CONS

- May forfeit HAB Docking ranking point
- Possibility to be generally mediocre
- Strategy leads itself to a complex robot
- Tall lifting mechanism could be a tipping hazard
- Repeatedly jumping off Level 2 during autonomous could cause damage to our robot

STRATEGY BRAVO

During the Sandstorm period, our robot will drive off of HAB level 2, place one hatch on the nearest cargo ship bay to hold in a piece of cargo, and collect the cargo from the adjacent cargo ship bay. After the Sandstorm, the robot would focus on completing a rocket by placing the hatches and filling the cargo on all levels. Toward the end of the match, the robot would return to the HAB Platform and climb to Level 2.

PROS

- Completes all main game tasks
- Potential to earn an additional ranking point by completing a Rocket on our own
- Generally self-reliant
- High likelihood of obtaining the HAB Docking ranking point
- More flexibility as an alliance member
- Lots of potential fallback options
- Level 3 climb is a differentiating factor

CONS

- Possibility to be generally mediocre
- Strategy leads itself to a complex robot
- Tall lifting mechanism could be a tipping hazard
- Completing every task could pose weight challenges
- High prototyping commitment for Lever 3 climb
- Repeatedly jumping off Level 2 during autonomous could cause damage to our robot

STRATEGY CHARLIE

During the Sandstorm period, our robot will drive off of HAB level 2, place one hatch on the nearest cargo ship bay to hold in a piece of cargo, and collect the cargo from the adjacent cargo ship bay. After the Sandstorm, the robot would focus on placing the hatch panel covers on the rocket and then the cargo ship. We can also defend our alliance during this time. Toward the end of the match, the robot would return to the HAB Platform and climb to Level 2.

PROS

- Can place a lot of hatches during a match
- Works well with robots that place Cargo
- Focuses on one action during the main gameplay (lets the team perfect it)
- Reduced complexity allows for a faster design and build process
- Better than focusing only on placing Cargo

CONS

- Relies on other robots to get a ranking point for the rocket or HAB Docking
- Potential to be redundant because there will likely be a lot of robots that focus on placing the Hatches
- Lower point cap
- Relies on alliance members to not put on null hatches in order to gain points from the Hatch
- If we hit the point cap and there is already a robot from our alliance focusing on defense, our robot could be stuck doing nothing
- Repeatedly jumping off Level 2 during autonomous could cause damage to our robot

STRATEGY DELTA

During the Sandstorm period, our robot will drive off of HAB level 2, place one hatch on the nearest cargo ship bay to hold in a piece of cargo, and collect the cargo from the adjacent cargo ship bay. After the Sandstorm, the robot would focus on placing the hatch panel covers on the rocket and then the cargo ship. Toward the end of the match, the robot would return to the HAB Platform and assist one of our alliance partners in climbing to Level 3.

PROS

- Can place a lot of hatches during a match
- Works well with robots that place Cargo
- Focuses on one action during the main gameplay (lets the team perfect it)
- Reduced complexity allows for a faster design and build process
- Better than focusing only on placing Cargo
- If the assist works, we are guaranteed a ranking point for HAB Docking
- Potential for 12 extra points
- Stands out for alliance selections because the Level 3 Assist is a differentiator
- Climbing could be a faster option

CONS

- Relies on other robots to get a ranking point for the rocket or HAB Docking
- Potential to be redundant because there will likely be a lot of robots that focus on placing the Hatches
- Lower point cap from Hatches
- Relies on alliance members to not put on null hatches in order to gain points from the Hatch
- Repeatedly jumping off Level 2 during autonomous could cause damage to our robot
- Potential to damage our robot if the Level 3 Assist fails
- Relies on alliance partners being willing to drive on us
- Could require our alliance partners to have a specific drivetrain or configuration
- We don't have as much control over what happens to getting to Level 3
- Less desirable than our robot going directly to Level 3
- More complex than Level 2 because it requires more integration of mechanism design into the robot
- We're not helpful if someone already has a Level 3 climb

MAKING A DECISION

STRATEGY ALPHA - THE FINAL DECISION

After outlining the pros and cons of our strategies as a team, we spent time discussing the different strategies, considering how the different pros, cons, and other attributes would affect our team's performance. Once our discussion stopped showing new ideas (about half an hour of discussion), we called a vote. Each team member was given votes relative to experience (i.e., 3rd build season = 3 votes). Our mentors each had one vote. Our team overwhelmingly voted for strategy alpha, with 54% of votes attributed to that approach.

DESIGN DECISIONS

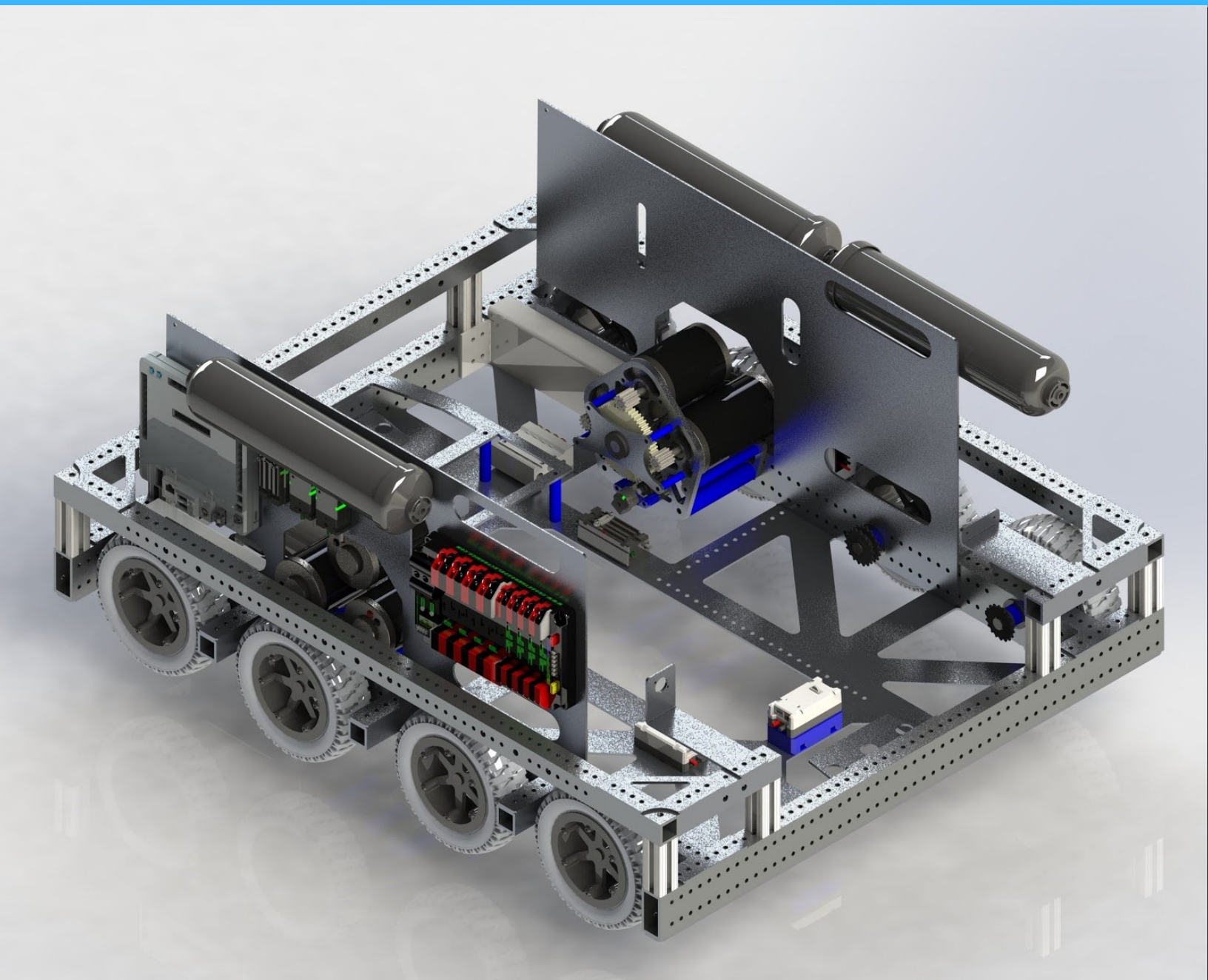
Several decisions were made in the process of designing our robot. In this section, the six main subsystems that make our robot function will be analyzed from a design and process aspect.

THE DESIGN PROCESS

We started our brainstorming process by developing lists of desired attributes for each mechanism to accomplish. We then created tables comparing each of our potential implementations with the attributes we developed. We used a 1, 0, -1 weighting system to approximate how well an implementation would realize its goals. This process was a crucial part of all of our subsystems, giving us a powerful tool to help determine what implementations were worth pursuing.

THE DRIVETRAIN

After developing our strategy on day one, we knew what the robot would do, and by the end of day two, we understood what mechanisms we would further develop. Once we had this information, technical team members who cared met up in our design space to discuss how we wanted to implement our drivetrain.



OUR OPTIONS

Required Attributes	Units	Minimum FN	Design Target	Tank Drive	Swerve	Drop Omni
Cycle Time/Speed	ft/s	7.5	15	1	1	1
Accel	ft/s^2	30	60	1	1	1
Exiting (time and height)	sec	5	2	1	-1	0
Lateral Resistance	lb	100	150	1	0	0
Pushing Force	lb	150	200	1	0	0
Ground Clearance	in	-	-	1	1	1
Fine alignment	yes/no	no	yes	-1	1	0
Total Score				**5**	**3**	**3**

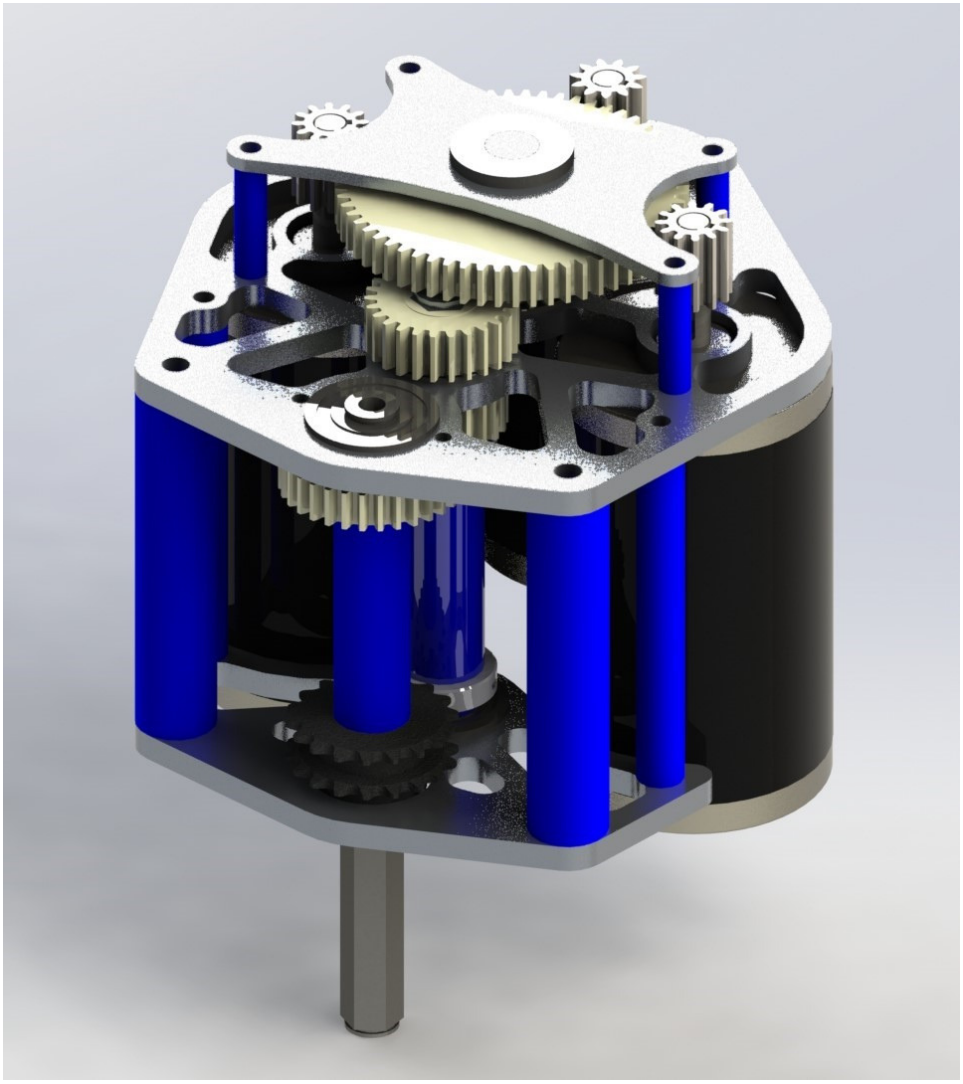
We considered three broad categories of drivetrains and determined which would best fit our needs. We realized quite quickly that only one of these categories was worth pursuing, and put the drivetrain on hold for a later discussion. At the end of our day two meeting, technical team members volunteered to outline specifics for a drivetrain.

Chassis

- Design Constraints
  - Wide as to not tip while climbing onto HAB
  - Sprint quickly to and from field elements
  - Drive on platform without high centering
  - Long wheel base to prevent tipping

- Chassis
  - 8 x 6in West Coast Products pneumatic wheels
  - Center 4 Wheels dropped 0.25in to reduce scrub
  - Center to center chain system to drive all wheels

THE GEARBOX



To understand the mathematics, reasoning, and revision of our gearbox, continue to the next page.

ITERATIONS

Iterated 1-Speed Drivetrain

	Free Speed (RPM)	Stall Torque (N*m)	Stall Current (Amp)	Free Current (Amp)	Speed Loss Constant	Drivetrain Efficiency
CIM	5330	2.41	131.00	2.70	81%	80%

# Gearboxes in Drivetrain	# Motors per Gearbox	Total Weight (lbs)	Weight on Driven Wheels	Wheel Dia. (in)	Wheel Coeff
2	3	154	100%	6	1.1

Driving Gear	Driven Gear	Drivetrain Free-Speed	Drivetrain Adjusted Speed	"Pushing" Current Draw per Motor
12	62	14.85 ft/s	12.03 ft/s	70.49 Amps
24	30	9.39 : 1	<-- Overall Gear Ratio	
22	32			
1	1			

ITERATIONS

Our robot was overweight, so we switched out a CIM for a MiniCIM to save approximately 1 lb. This led to a slightly different top speed of 14.16 ft/s

1-Speed Drivetrain

	Free Speed (RPM)	Stall Torque (N*m)	Stall Current (Amp)	Free Current (Amp)	Speed Loss Constant	Drivetrain Efficiency
Mini CIM	5840	1.41	89.00	3.00	81%	85%

# Gearboxes in Drivetrain	# Motors per Gearbox	Total Weight (lbs)	Weight on Driven Wheels	Wheel Dia. (in)	Wheel Coeff
2	3	154	100%	6	1.1

Driving Gear	Driven Gear	Drivetrain Free-Speed	Drivetrain Adjusted Speed	"Pushing" Current Draw per Motor
11	62	14.92 ft/s	12.08 ft/s	70.01 Amps
24	30	10.25 : 1	<-- Overall Gear Ratio	
22	32			
1	1			

THE ELEVATOR

CONSTRAINS AND SPECIFICATIONS

ELEVATOR

CONSTRAINTS

- Be light enough to drive at max height without tipping
- Reach the top of a rocket
- Fits under max robot height when a match begins

FRAME

- 63 in. max elevator height
- 2x1 construction
- .625 in. x .25 in. bearings

CONTINUOUS RIGGING

- 2 runs of .0625in kevlar rope rated for 500 lbs to pull up for redundancy
- 1 run of .0625in kevlar to pull elevator down
- 3D printed nylon pulleys and pulley blocks to reduce weight
- Keeps the second stage down allowing the arm to pass over the top of the elevator

ELEVATOR GEARBOX

CONSTRAINTS

- Able to stall motors at minimum voltage
- Be able to traverse entire 63 in. max height in under .75 seconds
- Low profile to allow arm on the back side
- Raise elevator to defined heights

POWERED BY 2 NEO BRUSHLESS DC MOTORS

- 2.5:1 gear reduction
- Final drum speed of 2,270 RPM
- Final lift speed of .64 seconds at a linear speed of 98.9 in/s
- Final current draw of 24.22 amps per motor

- SRX magnetic encoder on final drum
- Wired to two magnetic limit switches wired through a CANifier to detect when at top and bottom of travel

SPOOLS

- 1.25in OD .0625in aluminum drum with 3 cable runs
- Synchronized cable wrapping

OPTIONS

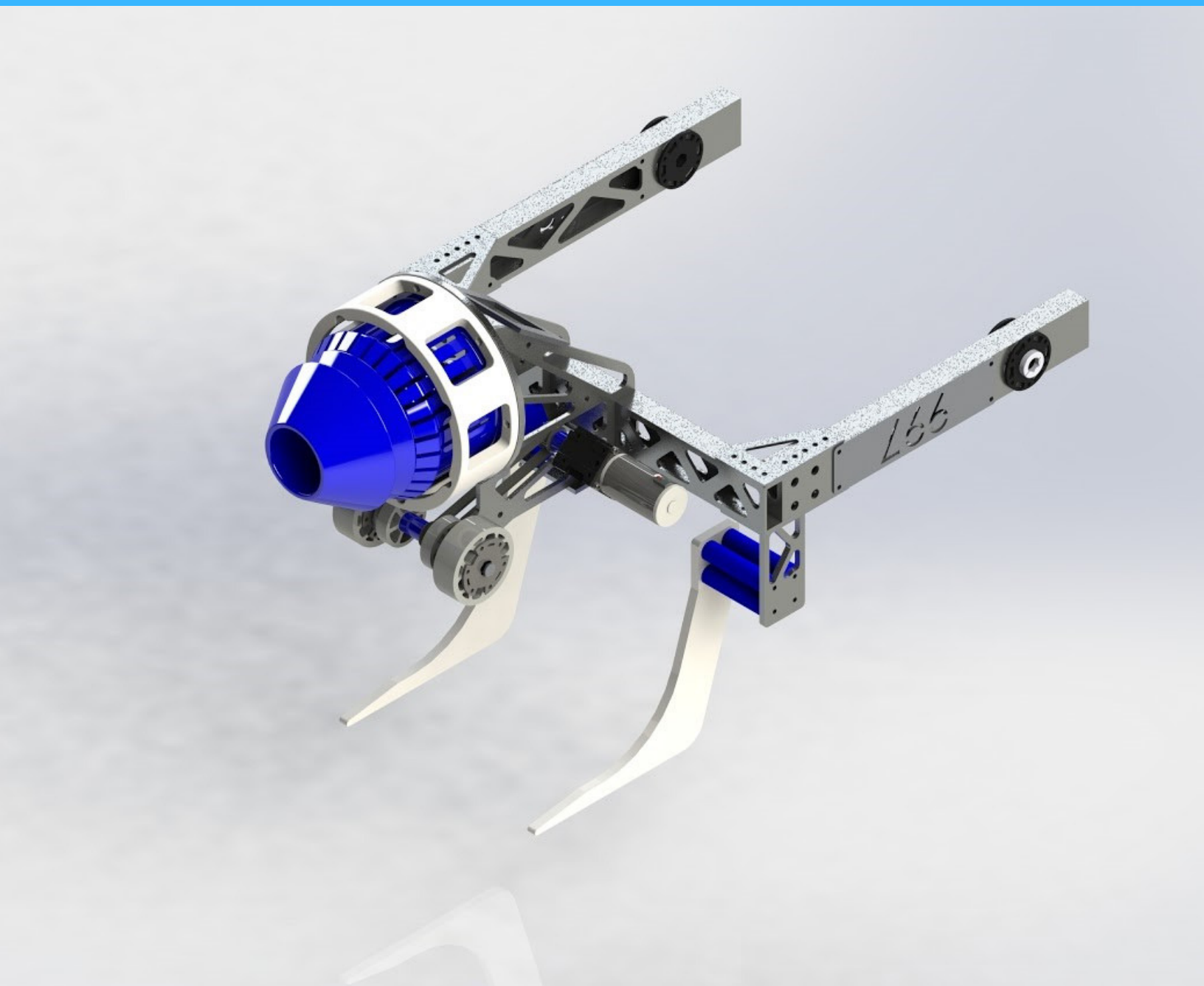
EXPERIENCE

We have experience from previous years with 2 stage elevators so this made it was an easy decision to use what we knew would work and that we could build. This elevator differed from previous ones as we used continuous rigging instead of cascade. This reduced the forces applied on the motors but increasing the time to lift, forcing a lower gear ratio on motors.

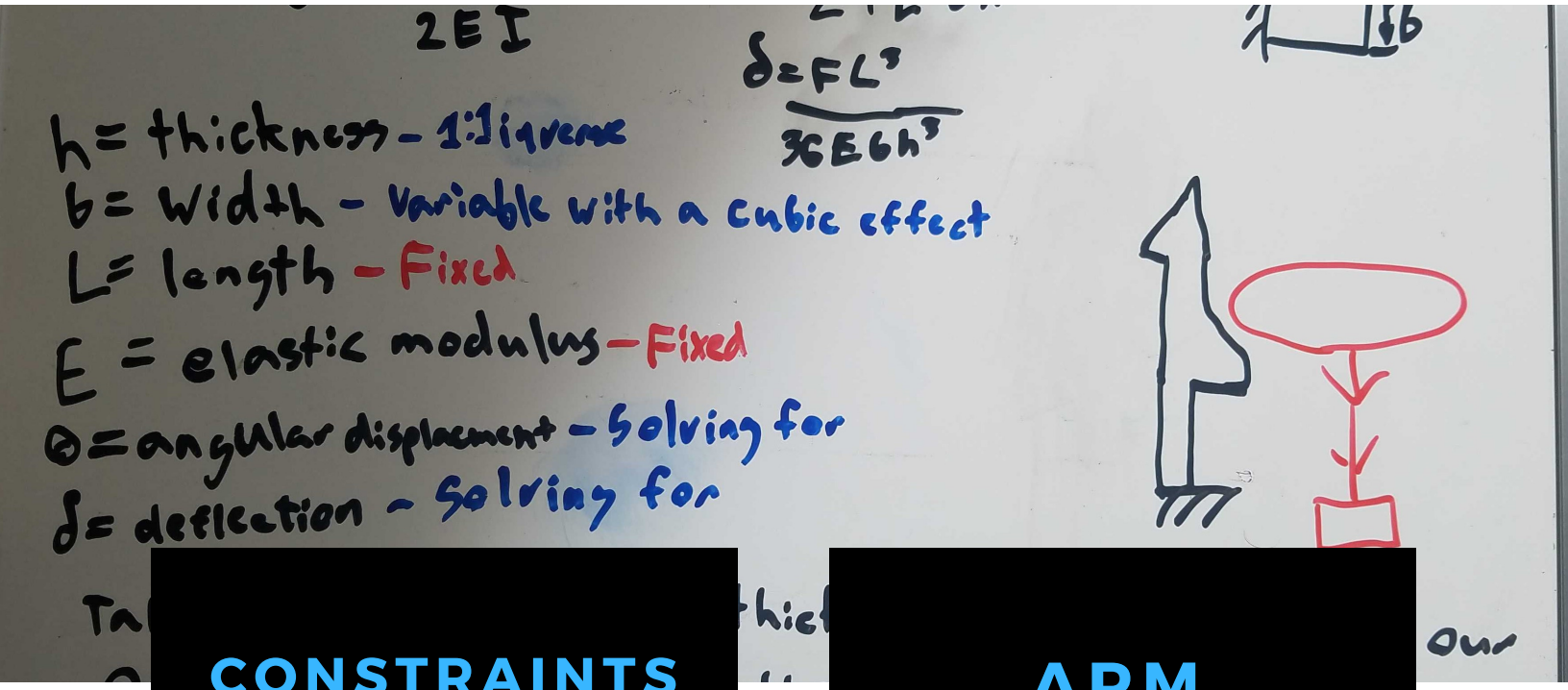
Required Attributes	2 Stage Lift	Telescoping Boom	Articulating Arm
Cycle Time/Speed	1	0	0
Maximum Lift Height	1	1	1
Ability to Raise to Unique Heights	1	1	0
Minimum Height	1	0	1
Max Load	1	0	-1
Maximum Stored Height	0	0	0
Ability to Know Position	1	1	-1
Maintainability	1	0	0
Manufactuability	1	0	1
Rigidity	1	1	0
Weight	0	0	0
Total Score	**9**	**4**	**1**

THE ARM

This major component enables our robot to pick up and maneuver with cargo and hatches safely and securely. To learn more, follow our design process on the next few pages.



CONSTRAINS AND SPECIFICATIONS



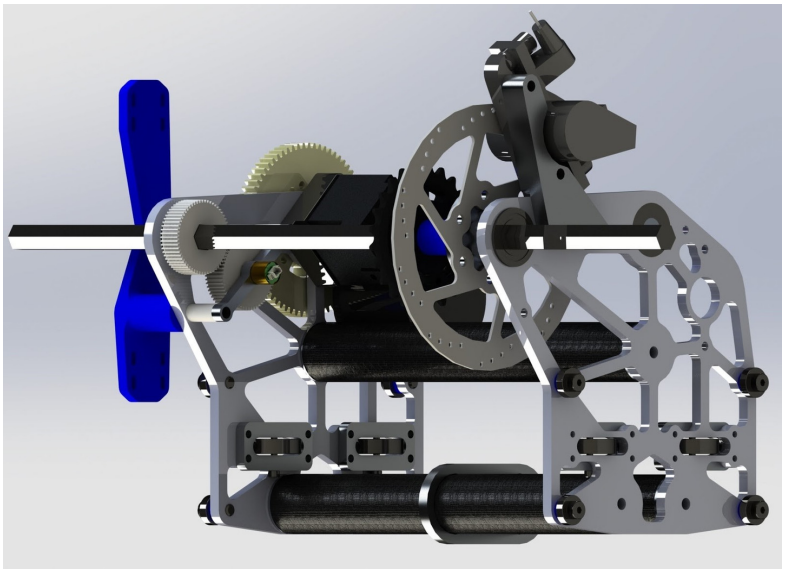
CONSTRAINTS

- 180 degrees of rotation
- Needs to start a match inside frame perimeter
- Needs to fit over the elevator
- Needs to hold in multiple positions
- Be able to go outside the frame perimeter on one side and stay inside on the other

ARM

- 15in length
- 16in wide to fit over 14in wide elevator

ARM GEARBOX/CARRIAGE



- Powered by 1 NEO brushless DC motor
- US digital MA3 absolute encoder
- Wired to 2 magnetic limit switches to register when the arm has reached horizontal on both sides

CONSTRAINS AND SPECIFICATIONS

Constraints

- Rotate 180 degrees in under 1 second
- Hold arm in locked positions
- Position arm at defined angles
- Light weight to enable fast raising and lowering of the elevator

CARRIAGE

- Sides made of 0.25in Plate
- 3 carbon fiber and aluminum spacers to create stability while also reducing weight

DISK BRAKE

- Cable actuated
- Piston at bottom of elevator to lower CG

CARGO MANIPULATOR

CONSTRAINTS AND SPECIFICATIONS

CONSTRAINTS

- Rapidly pickup and center a ball
- Grasp the ball as to not drop it when the elevator and arm are moving

BALL INTAKE

- 3/16in aluminum plate used to reduce weight but also keep rigidity
- Tines made of .375in polycarbonate plate so they could not break while hitting things but be stiff enough to secure the ball
- 1 BAG motor on a 10:1 reduction for a surface speed of 184.3 in/s
- 2 mecanum wheels vector the ball to the center of the intake

OPTIONS

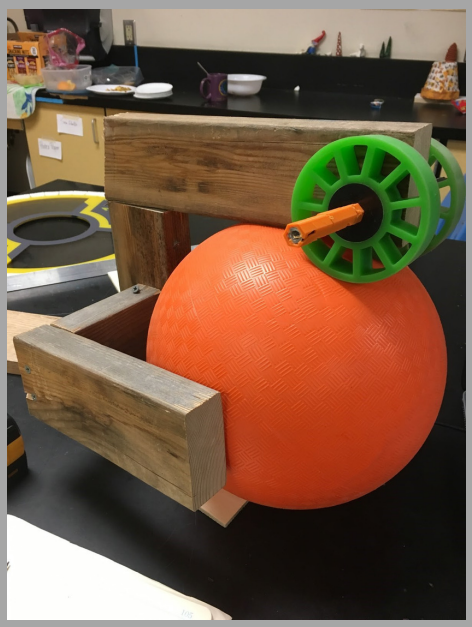
After we compiled this, we deliberated which mechanisms would be worth further research. We ended up prototyping a wheeled intake and a claw. While we felt the belted intake had potential, it was too similar to the wheeled intake to need a prototype.

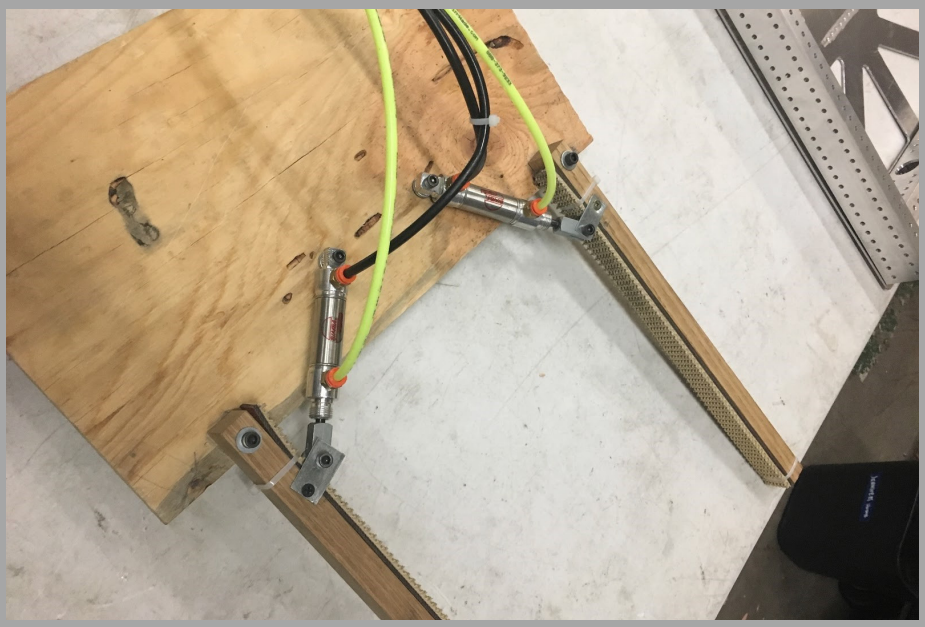
Required Attributes	Horizontal Claw	Vertical Claw	3-Pronged Claw	Wheeled Intake	Vacuum	Belted Intake
External Delivery	0	0	0	1	0	1
Collect from ground	0	0	0	1	0	1
Range of capture	-1	-1	-1	1	-1	1
Weight	0	0	-1	-1	0	-1
Time in	-1	-1	-1	0	-1	0
Time out	0	0	0	1	0	1
Complexity/MFG	1	1	0	0	0	-1
Secure possession	1	1	1	1	-1	0
Doesn't damage cargo	0	0	0	0	0	0
Robustness	0	0	0	1	1	1
Placement accuracy	-1	-1	-1	0	-1	0
Robust to ball variance	1	1	1	0	1	0
Total Score	**0**	**0**	**-2**	**5**	**-2**	**3**

PROTOTYPING

The claw prototype used pneumatics to actuate two pincers and grasp a ball. In the demonstration, it was not very effective at holding cargo - breaking our “you touch it you own it” philosophy - although we could have optimized this by tuning the pneumatic pressure.

The wheeled intake prototype used two compliant wheels to intake cargo. Wooden structures on the sides and bottom stabilized the cargo. It showed the ability to intake and eject cargo.





ITERATIONS

- Switched to mecanum wheels due to their ability to center balls.

KEY FEATURES

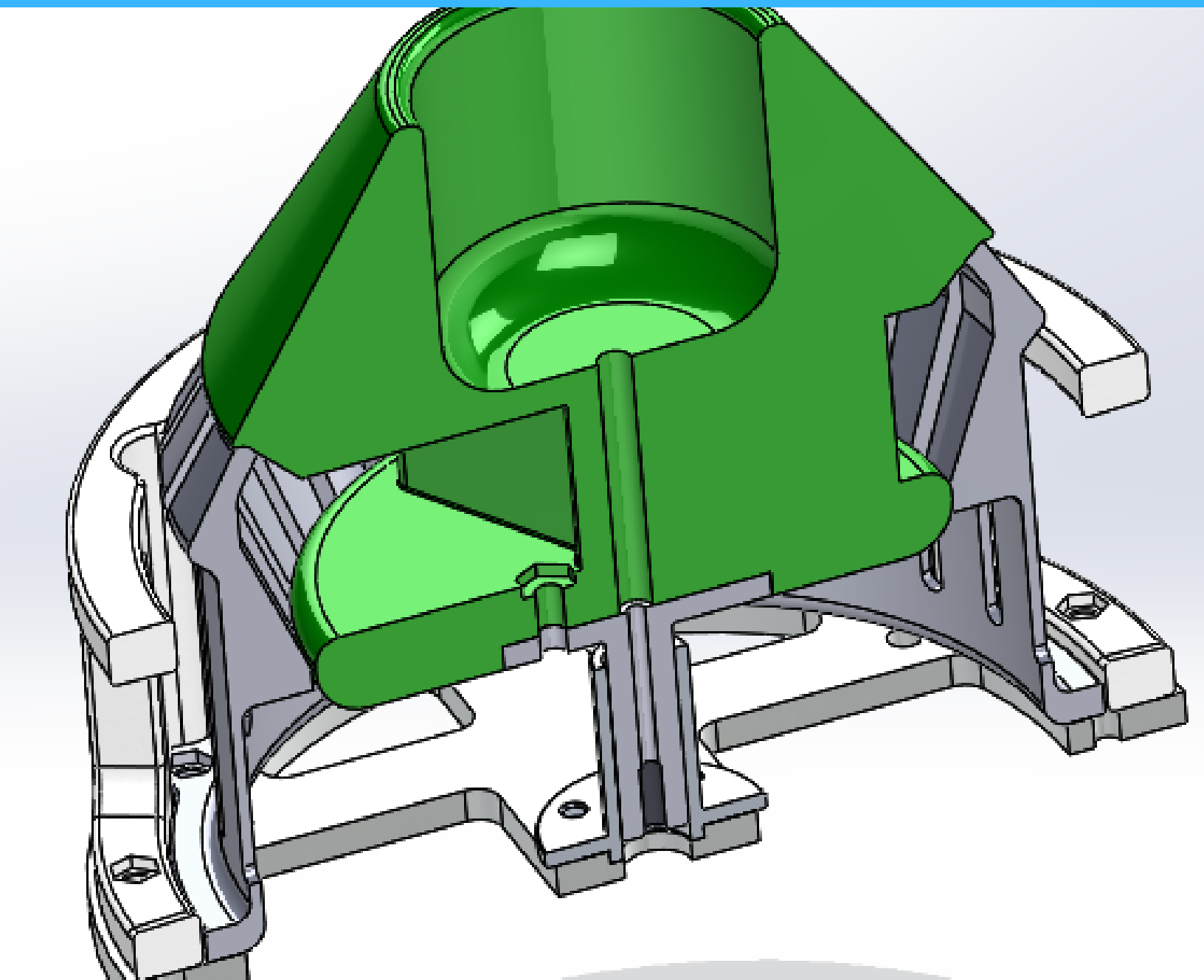
- Ability to pick up balls from the ground.
- Controls balls as soon as it touches them.
- Can launch balls several inches.

CHALLENGES

When collecting balls from off-center only one “tusk” supports them, which renders them vulnerable to being dislodged. Also, due to the same catterwonk force that makes the ball only sit on one tusk, the plates holding the tusks bent outwards.

THE HATCH MANIPULATOR

This major component enables our robot to pick up and maneuver with hatches safely and securely. The teeth and actuation enable us to pick up and drop off hatches. To learn more, follow our design process on the next few pages.



OVERVIEW AND OPTIONS

Our primary goals when developing a hatch manipulator were to make something robust and with low complexity. This meant focusing on simple (preferably passive) mechanisms. We wanted to place in all locations, which raised the value of systems which interface with the flat surface of the hatch or the hole in the middle. We also wanted to use the least number of moving components because they are the most likely to fail. Due to the simplicity, we wanted to keep, not many options for ground pick-up were considered.

OPTIONS

After compiling this as a team, we deliberated on which mechanisms to further look into. We ended up prototyping a CD holder. While the internal latch had potential, it was similar enough to the CD holder to not require a distinct prototype.

Required Attributes	CD Holder	Internal Hook	External Latch	Velcro	Vacuum	Internal Latch	Roller Claw
Collection height	1	1	0	1	1	1	0
Low deposit height	1	1	0	1	1	1	0
Middle deposit height	1	1	1	1	1	1	1
Top deposit height	1	1	1	1	1	1	1
Reach of pickup/placement	0	0	0	0	0	0	0
Horizontal tolerance	1	0	0	-1	-1	1	0
Secure possession	1	-1	0	-1	-1	1	0
Grips vertically	1	0	1	0	0	1	1
Placement force	1	1	1	1	1	1	1
Positive disengagement	1	0	1	1	1	1	1
Catches failed attempt	0	0	0	0	0	0	0
Size/weight	1	1	0	1	0	1	0
Ground collection	-1	-1	-1	1	1	-1	1
Complexity/MFG	1	1	0	1	-1	1	0
Release success rate	1	-1	1	0	1	1	0
Total Score	**11**	**3**	**5**	**7**	**5**	**11**	**6**

PROTOTYPING

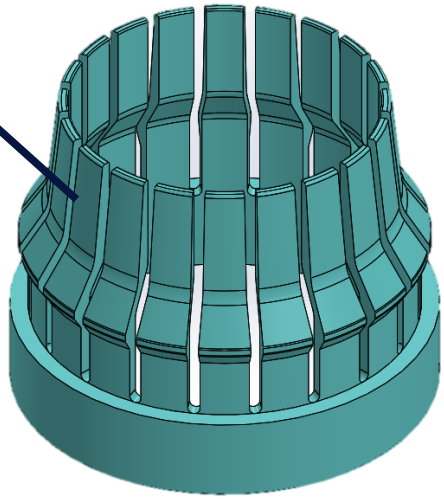
Our main inspiration for this prototype was a CD case. We quickly 3D-printed a proof-of-concept prototype, a circular piece of plastic with angled fingers that lock into the center of the hatch. We optimized various dimensions and investigated ways to place a hatch actively.

We had many iterations and lots of prototyping of this manipulator, because we had not seen many teams so anything like it before, and it is a crucial component to our strategy. Through our prototypes we learned two things. First, the importance of having a flexible/robust material for the fingers as their flexing is a key part of the design working. Second, the hatch manipulator needs a back plate. The design could have worked passively, however the performance was not up to our standards, so we started looking at actuated hatch manipulators

REVISION 1

DELRIN

THIS DELRIN ITERATION CAN BE EASILY MANUFACTURED ON A LATHE



Disadvantages
Cons

- Expensive
- Cost reduces ability to produce spares
- No actuation
- Driver cannot control the hatch manipulator individually (in this revelation)

During the early stages of the design process, our whole team brainstormed different ways to pick up hatches. This design was originally called the CD holder. After some discussion and prototyping, this model was chosen to be on our robot.

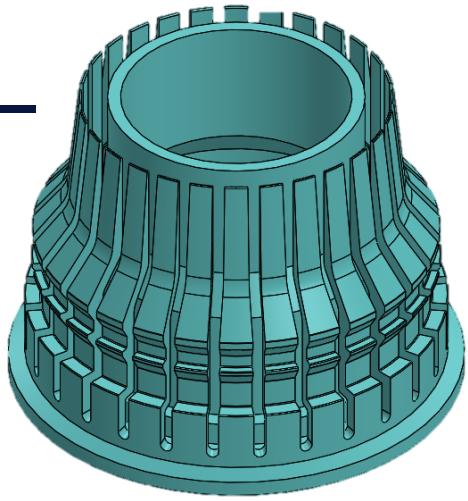
Advantages
Pros

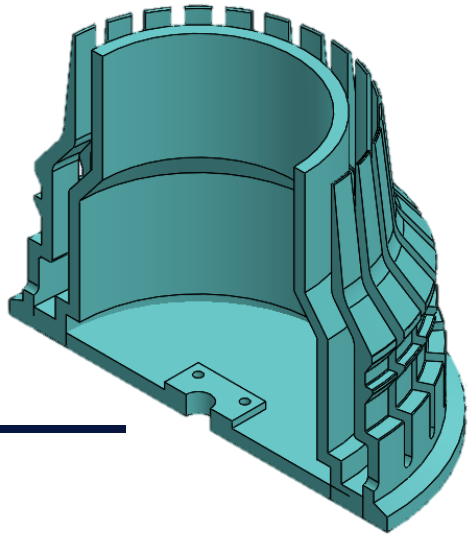
- Simple
- Reliable
- Light
- Delrin is near indestructible
- Easy to manufacture
- Can flex enough to fit the shape of hatches
- Finger-like structures provide flexibility
- Angles on fingers allow hatches to slip on
- Lower angles prevent hatch from accidentally falling off
- Larger base prevents hatches from falling backwards

REVISION 2

3D PRINTED

This version is 3D-printed and now has an internal ring to help support the teeth. During these initial stages, we designed the hatch manipulator to actuate externally by a pneumatic cylinder pulling from the bottom. Therefore, we added a location for pneumatic attachment at the bottom (shown in cross section view). The next version has more components relating to external actuation.





Disadvantages
Cons

- No longer machinable
- Slightly harder to make
- More complicated
- No longer durable

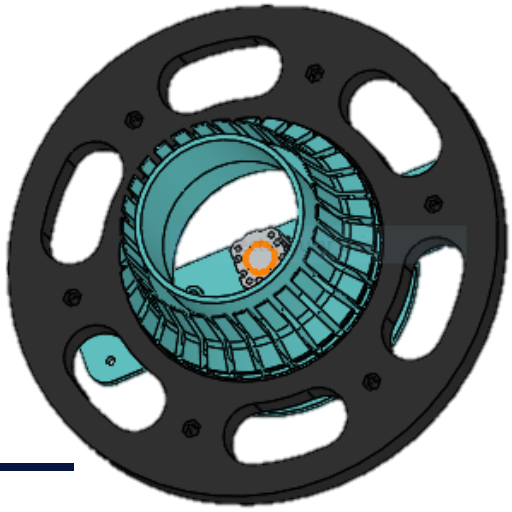
Advantages
Pros

- 3D-printable
- Easily replaceable
- Lightweight
- Has pneumatic attachment
- Internal finger support

REVISION 3

3D PRINTED

In the earlier versions, the hatch manipulator could attach to the hatch but was not able to push it off. Therefore, we integrated an external actuation ring (also 3D printed). This enabled the manipulator to push hatches off onto the Velcro strips adhered cargo ships and rockets. Without this addition to our design, the robot would not be able to leave hatches on Velcro strips. A new mounting system was also incorporated into the design so that the hatch manipulator could stay securely on the robot while being tested.



PROS

- Easy to mount
- Hatches can be pushed off onto Velcro strips of cargo ships and rockets
- Drivers can remotely control the movement of hatches individually

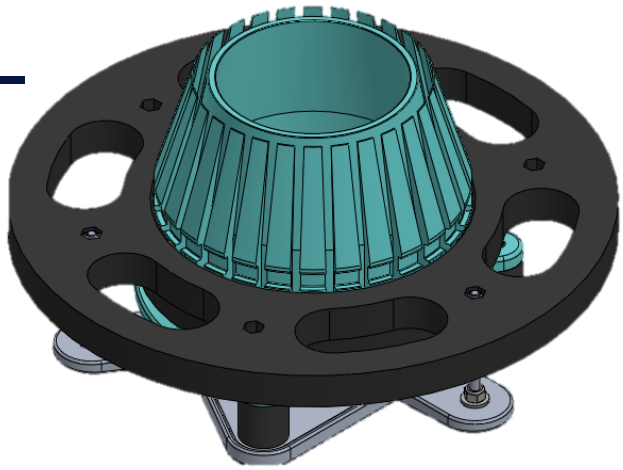
Cons

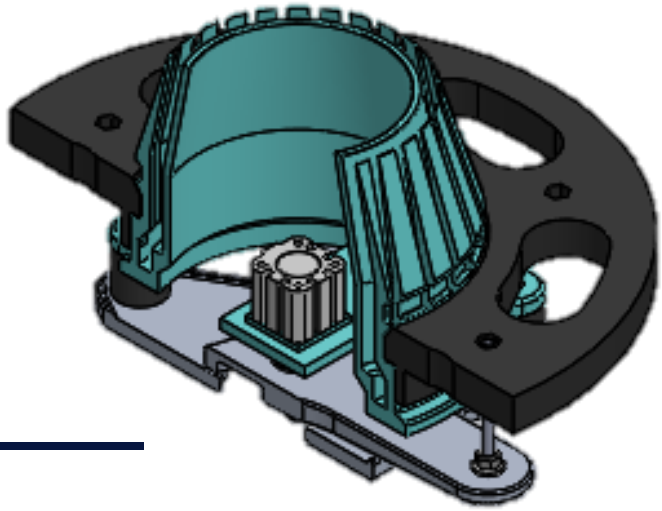
- Ring is fragile
- Printing is complex
- Bulkier
- Larger physical footprint
- Heavier
- Adds weight to arm

REVISION 4

3D PRINTED

In this revision, we added a star arm sub-assembly. Since we had not yet added the actuation mechanism to the model, we added a three tab "star arm" that enables the external actuator ring while being attached to the robot. In this revision, the actuation ring was given the ability to shift by means of pneumatics with more functionality.





Disadvantages
Cons

- Many machined components
- Complicated
- The actuation mechanism is delicate

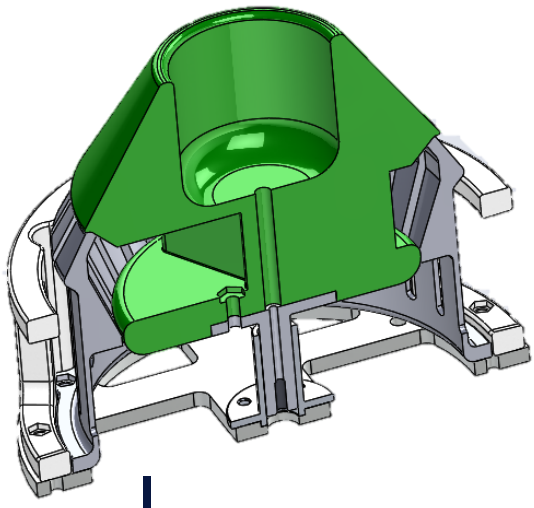
Advantages
Pros

- Now has star arm system to activate the ring
- Thus, the ring can serve its purpose of shifting to push off hatches

REVISION 5

3D PRINTED

Because of over-complication, fragility, weight concerns, and a large physical footprint, we redesigned the hatch manipulator to be internally actuated. This way, the internal actuation could connect almost directly to the pneumatic cylinders. Other than the complete redesign of the actuation mechanism, some small changes in the ramp and lead-in angle were made on the fingers.



PROS

- Less likely to break due to external forces
- Takes up less space
- Less complicated

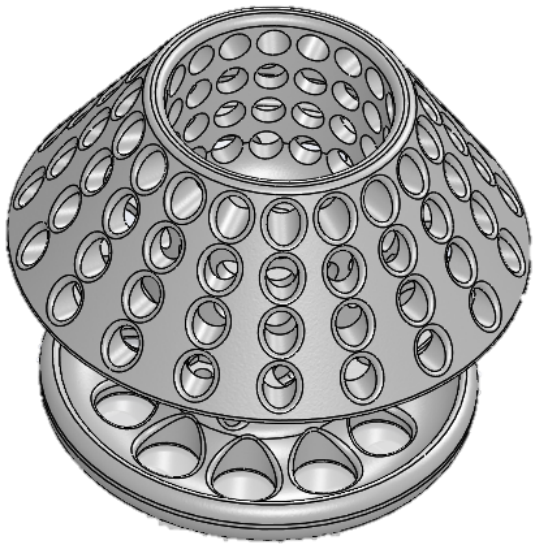
Cons

- Harder to assemble
- Internal components are difficult to install

REVISION 5A

3D PRINTED

Because the internal manipulator weighed almost 3.5 pounds, we lightened the manipulator. The actuator weighed 1.5 pounds, so we lightened it with holes.



PROS

- Less likely to break due to external forces
- Takes up less space
- Less complicated
- lighter

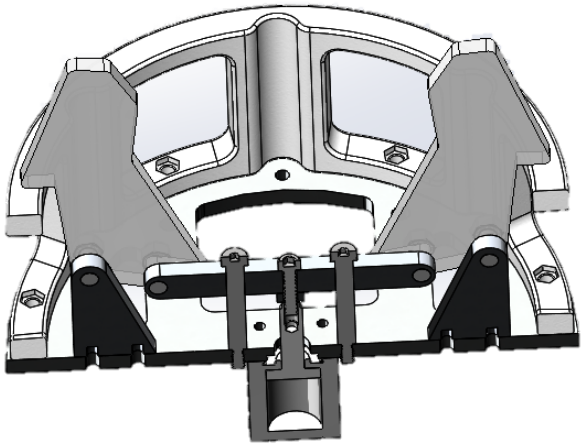
Cons

- Robot needs to me more accurately aligned in order to secure hatches

REVISION 6

BACKUP

Since several previous versions of the hatch manipulator had fell apart, this different design was created to ensure a safe solution. If revision 5a breaks or does not function properly, this completely different design is always there as a backup. In addition the main grasping components of this revision is made of Polycarbonate.



PROS

- More durable Polycarbonate parts
- Flexible components are not 3D printed, and are much less likely to shatter or break
- Less weight and protrusion
- Attaches to preexisting components
- Less skill necessary as a driver

Cons

- More small components

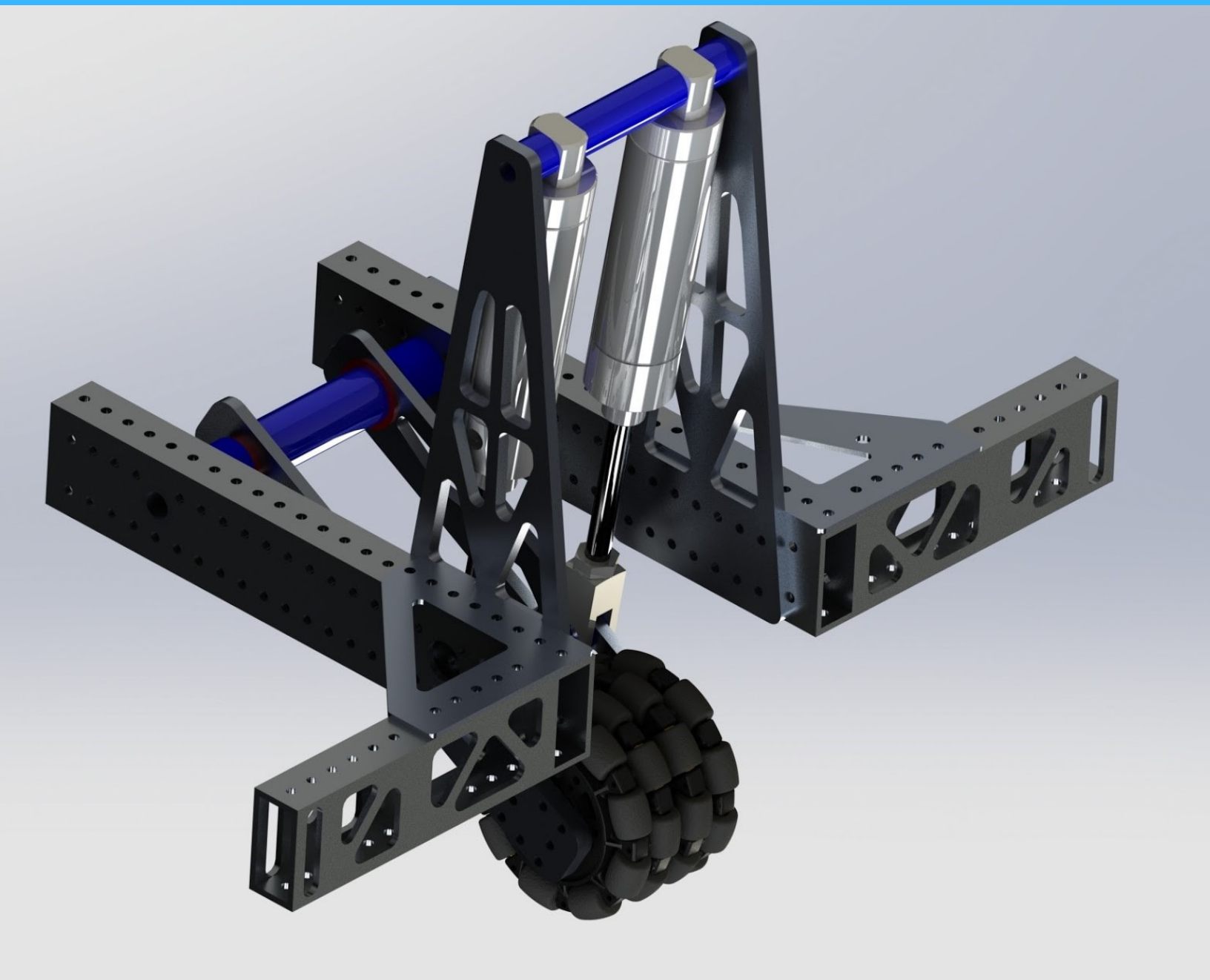
CHALLENGES

One of the first challenges we encountered in the design of the “CD Holder” hatch manipulator was the shape and angles on the outer teeth. Finding the right angle at which the teeth held the hatch securely required several tests. Another challenge we faced was the durability of the teeth. Due to the material and thickness of the teeth, they broke when fiddled with. Often, when the hatch was being moved, teeth broke off and several reprints were necessary.

The iterations broke before we could practice well enough with them. Because of the long printing times, spares could prove challenging to produce. We could not print the Nylon iteration ourselves, and instead, requested our sponsor, HP, to print it for us. The materials we used to make it more durable made the manipulator expensive.

HAB RETURN

According to our strategy, our robot had to climb onto level two of the habitat. Because we would not be trying to achieve the climb onto level three of the HAB, we could still incorporate our mechanism into the drivetrain and not have to make something separate.



OPTIONS AND PROTOTYPING

OPTIONS

After compiling this as a team, we deliberated on which mechanisms to further look into. We ended up prototyping single drop wheels ("landing gear.") While the wedge had potential, it was too high-risk for ramming into the driver station and damaging the sandstorm.

Required Attributes	Single Drop Wheels	Ramming Wedge	Tipping w/ High CoG	Massive Drop-Center	Articulating Frame	Dual Drop Wheels
Approaches from level 1	1	1	1	1	1	1
Risk of slamming into DS	0	-1	-1	0	0	1
Height	1	1	0	0	1	1
Robot width	1	1	1	1	1	1
Time	0	0	1	1	0	0
Weight	0	1	1	0	0	-1
Number of actuators	0	0	1	1	-1	-1
Complexity/MFG	0	1	1	1	0	-1
Stability	1	1	-1	0	1	1
Success rate	1	1	-1	0	1	1
Level 3 possibility	0	-1	1	-1	0	1
Total Score	**4**	**4**	**3**	**3**	**3**	**3**

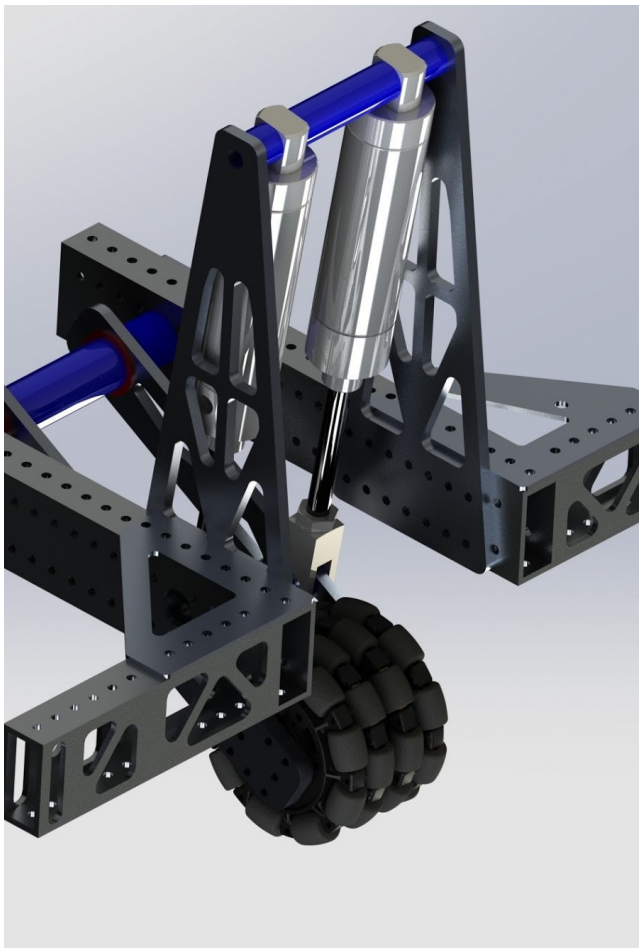
PROTOTYPING

The prototype consisted of two cylinders which extended wheels downwards, pushing the robot upwards and allowing it to climb onto the second level of the habitat. To prototype this, we made a drivetrain out of wood so we could test the strength of the mechanism and how it would interact to attempts to climb onto the habitat. Testing showed that the mechanism could work, it would just need fine-tuning.

ITERATIONS

- We tried to add more pistons in the back. One of them broke and we scrapped the idea.
- Used Omni-Wheel as it would allow for better maneuvering when actuated.
- Decided on where to put the pistons after lots of math.

CHALLENGES



At first did not seem to be able to extended the pistons fully without outside help. Two solutions were developed for this, adding springs to the piston and moving the arm and robot itself around to get the center of gravity into a better position. Moving of the arm to the inside of the robot, driving the robot to the edge of the habitat coupled with adding spring allowed the landing gear to be actuated fully every single time.

CONTROLS SUBTEAM

This year the controls subteam learned how to use and implemented many new sensors we had never used before.

We used three IR (infrared) sensors to tell A) when to retract our landing gear and B) to tell how far we are from the rocket and cargo ships (one on each side of the robot). These were chosen for their abilities to sense short distances precisely, which was exactly what we needed for these two tasks, especially for the landing gear which would hold us back if we didn't retract at precisely the right time.

We also are using several simple magnetic reed switches we have used in the past to prevent our elevator from exceeding its bounds. These were chosen for their simplicity and the fact they didn't need touch to be activated. Despite their simplicity they are absolutely essential for the elevator and arm's well being.

Along with that there are two sets of three photo-diffuse sensors that we are using to detect and follow the white gaffers tape lines on the field. Basically each set is in a line and when the right sensor sees white the robot turns left and vice versa.

We use CTRE mag encoders integrated with the Talon SRX motor controllers for the drive train. The addition of the NEO motors for the arm and elevator brought new challenges with sensing integration. The new Spark Max motor controllers do not yet allow external encoder inputs, so the mag encoder for the elevator and the absolute encoder on the arm are wired into a CANifier to send signals back to the RoboRIO. Breakout boards for limit switches were plugged directly into the Spark Max motor controllers. The same CANifiers are also used to control the LED ring lights for the camera, and send the ball intake switch signal to the RoboRIO.

As well as sensors we used a launchpad to make our custom driver station to reduce the information overload and complexity for the secondary driver.

The vision system uses the Microsoft 3000 HD camera and green LED ring light on a servo controlled turret, with signal processing through a Raspberry Pi.

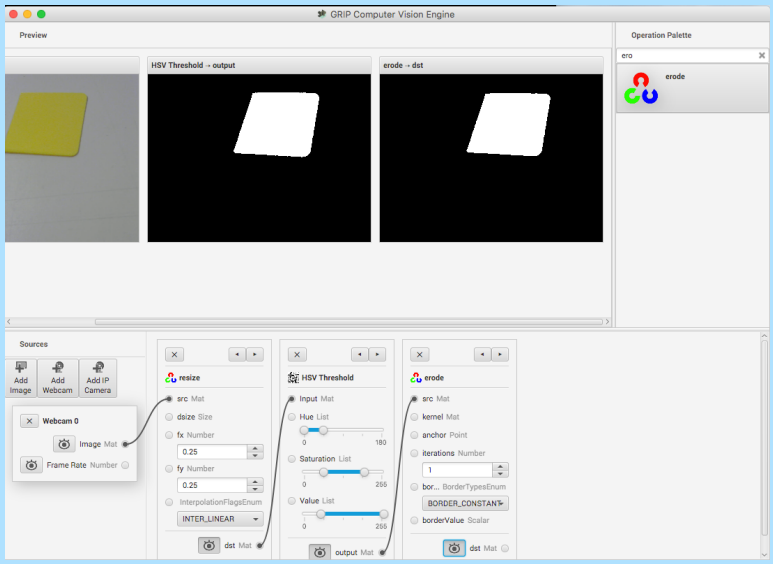
SOFTWARE SUBTEAM

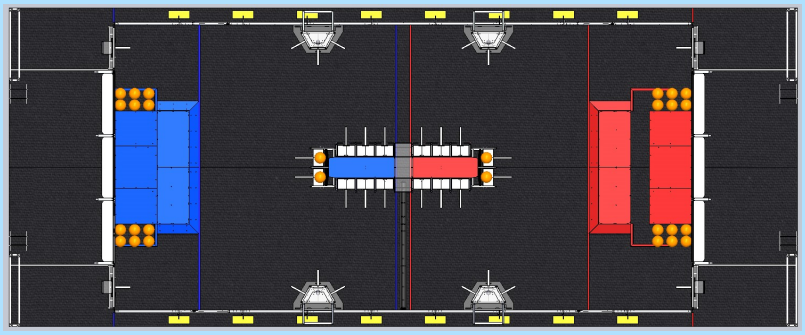
Deep Space has presented 997's software team with challenges that we have never faced before. As such (and certainly more so than any other year), we have utilized our resources to design, develop, and test creative and complex solutions to these challenges.

Albeit autonomous function during sandstorm is not required in this year's game, our team came to the decision that developing an arsenal of autonomous approaches was more reliable than driver operated control during this period. With a plethora of sensors, vision algorithms, and complex path generation, we have enough tools up our sleeve to rise to any challenge sandstorm might present.

The first addition to the collection was line following. This year, we decided to utilize two sets of three photo-diffuse sensors, with each telling us whether or not it sees a line (under the left, center, or right sensor). Using this data, we determine a correction factor, which is then fed directly into the drivetrain. Having a set of these sensors on both sides of the robot allows us to follow lines front and back. Along with these, we also use ultrasonic and infrared sensors to determine the distance between the robot and the game component and gauge when our routines are complete. Using all of these sensors, we can collect a pool of data that aid us in precisely aligning to our targets on the field.

Another field of data collection that our software team delved into this year was that of vision guidance. We have two cameras, two pan-tilt gimbels, and a Raspberry Pi. The cameras, along with a green ring light, feed visual data to the Raspberry Pi. The Pi then uses GRIP pregenerated pipelines to find vision targets. This data is streamed on an http server, which is displayed directly on the Driver Station. The driver(s) can then send customized commands to the RIO to either track the target, drive to the target, or choose a new target that is also seen by the camera.





With over a year of testing and development, the third and final method we use is that of motion profiling. Combining the new WPILib tool “PathWeaver” and JacisNonsense’s “PathFinder”, we are capable of creating and executing any path within a short amount of time. These tools allow us to shave off vital seconds of our routine runtimes, aid us in proper field orientation, and follow meticulously crafted paths in a precise and accurate way.

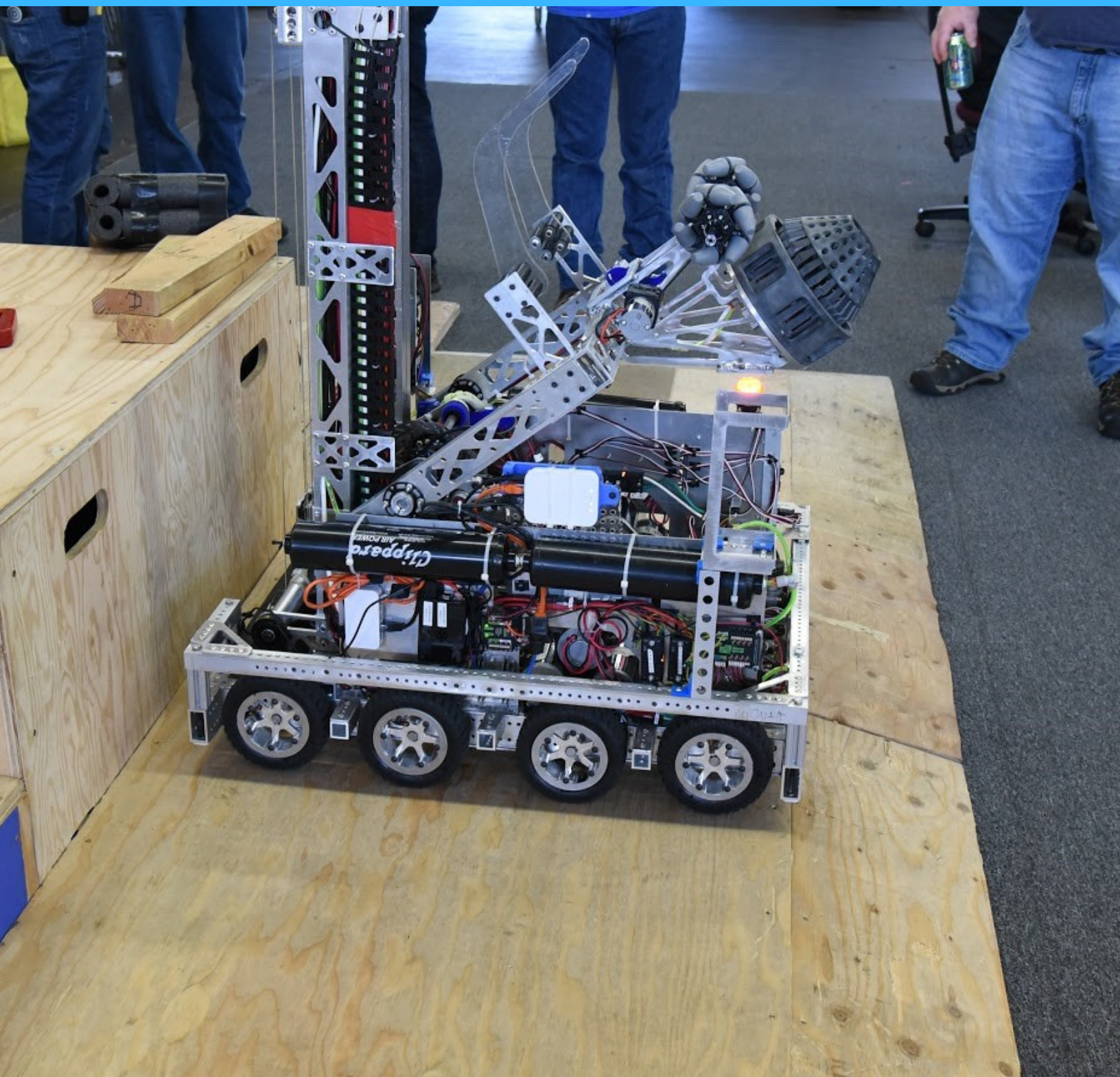
The diligent work done by the software team is certainly not limited to the sandstorm period. As we venture into the Teleop and endgame periods, we have developed several mechanisms to abstract away trivial actions and allow the drivers to focus on the real game.

One way in which we did this is with the ButtonBox. The ButtonBox is a custom built control system built and programmed by our software team, with the primary goal of reducing control complexity (pertaining to the elevator and camera). It utilizes a Teensy development board for its sheer simplicity to develop on and a high enough pin count to accommodate for our button and LED needs. The first 6 buttons (on the left side) are elevator height configuration buttons. This set specifies whether the driver aims for using the front or back side of the robot, whether the game piece is cargo or hatch, and whether we are heading to the cargo ship or rocket ship. The other three buttons are to determine scoring height if we are heading to the rocket. Along with the configuration buttons, there are 3 action buttons (below the first six)—activate, cancel, and intake. There are also 6 vision buttons (on the right side): A, X, and Y are for specifying which vision target to lock on to, the ‘center’ button to focus the camera on the desired target if it isn’t already, and the last two to manually pan the camera left and right. As a precaution, we also have a backup controller with matching controls. This is by far the most intricate control system our software team has ever developed.

```
switch (buttonBox.getScoringDirectionState()) {
 case Front:
  switch (buttonBox.getScoringArtifactState()) {
    case Ball:
      switch (buttonBox.getScoringDestinationState()) {
        case Rocket:
          switch (buttonBox.getPositionState()) {
            case High:
              setpoint = new ElevatorArmSetpoint(RobotMap.ElevatorHeights.elevatorFrontTopCargoHeight, 45);
              break;
            case Medium:
              setpoint = new ElevatorArmSetpoint(RobotMap.ElevatorHeights.elevatorFrontMiddleCargoHeight, 0);
              break;
            case Low:
              setpoint = new ElevatorArmSetpoint(RobotMap.ElevatorHeights.elevatorFrontBottomCargoHeight, 0);
              break;
            case None:
              break;
          }
          break;
```

A small sample of the deeply nested logic resulting from the vast amount of potential setpoints.

HIGHLIGHTS



MANUFACTURING EFFICIENCY

OVERVIEW

Throughout the season, we considered how we could reduce our manufacturing time by creating simpler, manufacturable designs. It's essential to our schedule to make sure we design parts that can make on manual machinery and CNC, as the single CNC our team can use is a major bottleneck. This year, we've done several things to improve our processes:

TOLERANCING AND A QUALITY CONTROL PROCEDURE

- After we design every part, we apply tolerances to the drawing before it goes to manufacturing. We have a set of standard tolerances we derive from ANSI and ISO standards (but not exactly because of our unique needs). Examples of these include +/- 0.005" for hole positions, or +0.000/-0.0015 for a press-fit (such as a bearing).
- Once the machinist has manufactured the part, they will take the part to the head machinist (the mechanical student lead) or a team captain for a quality control check. If the part passes the checks, it's placed in a bin labeled by subsystem to make sure we can keep track of which components are complete.

FASTENER STANDARDIZATION

- We have standardized over 75% of our fasteners to 10-32 hardware and 3/16" rivets, which allows us to drill one hole size to accommodate both. #10 is also convenient because it is the only bolt size where a snug fit is a #10 drill size, which makes the value convenient to remember for our machinists.
- Other standard fasteners we use are ¼-20, 8-32, and 6-32. We use these in structural applications and for mounting custom components such as the control system.

CNC OPTIMIZATION

- For CNC machining, we use CAM (Computer Aided Manufacturing) software to read our CAD files and generate conversational g-code that the CNC machine can interpret to make cuts. The CAM software we use is the HSMWorks plugin for SolidWorks, which lets us generate tool paths on part files within our robot. If we have a specialty part we need made, we also have MasterCAM available for specialty tool paths.
- We design our components on the robot for efficient manufacturing. For example, the rails on our drivetrain use standard #10 holes to minimize tool changes and speed up the manufacturing process. We do most of our milling with a ⅜" flat end mill to lower the amount of tool changes we have to do.

- This year, we reduced cycle time on our gearboxes from ~90 minutes from first cut to finish to ~45 minutes. This significant improvement came from more advanced knowledge of tool load and optimizing tool paths to reduce post-processing. Some things we did to optimize the manufacturing process include:
    - Switched our drilling operations to CNC peck drill operations. This increases tool life and nearly eliminates burrs on the bottom of holes. The extra time spent on the CNC (roughly 20% longer for each hole) was traded off for reduced post-processing time, we can remove burrs with 2 minutes on the belt sander.
    - Increased usage of Adaptive Clearing tool paths over legacy pocketing. This also increases tool life, reduces risk of tool breakage by evenly distributing the tool load, and allows us to machine faster. Previously, each gearbox would be pocketed with a ⅜" end mill with 0.1" radial steps and 0.05" depth of cut at 19ipm (inches per minute). However, with adaptive tool paths, that same pocket can be performed with a 0.075" stepover (slightly reduced), but with a full depth of cut. This means more material is being removed at a time, but since the tool is being evenly loaded, we can also push the machine faster and bring our feedrate to 34ipm. This alone resulted in a 10% reduction at least in cycle time per plate.
    - Efficient workholding. For our gearbox plates, we drill and tap a plate with the same bolt pattern as the gearbox, which allows us to set a work zero once on the fixture plate and then perform all the operations with the other plates bolted to the top. This means that only one zero is needed for the entire run of gearbox plates, which reduces cycle time by up to 10%.

TOOL LIBRARY

- This year, we created a tool library for our CAM software, which includes accurate feeds and speeds for all our cutters. This speeds up the CAM process significantly, as the integrations specialist can simply select a tool from our continuously updated tool library (as tools wear, break, and get ordered the library gets updated). This tool has a number in the library which corresponds to a tool in real life. When the machinist goes to make the part, they get the list of tool numbers from the setup sheet, set those tools up, and then run the operations necessary.

SPECIALISTS

OVERVIEW

After last year's build season, one of our team members stepped up, asking to create a new sub-team, called Integrations. This member noted how their approach to integrations could aid the team's workflow, both in the heat of crucial work and during more casual meetings. Our leadership felt that this was a brilliant idea, but we agreed that (at least in its current state), it was not a significant enough task to create a sub-team. However, it wasn't a duty we felt fit into any of our existing sub-teams.

We concluded that creating a new protocol to handle these tasks would be a valuable choice. Since then, we've implemented a 'specialists' system. A team member who wants to focus on a particular part of our team and develop a system for better implementing their ideas can apply to become a specialist. This empowers our team members to pursue their passions while supporting the team. While we haven't used this system in a prior build season, we've seen its value and are confident in its ability to enable both our team members and our full team to experience greater success, both during competition and while skill-building at our lab.

INTEGRATIONS

Integrations Specialist is a position created to help smooth over transitions between steps of the robot design and building process by aiding in communication.

The main issue we faced in previous build seasons was a lack of communication between the Design and Mechanical sub-teams. With Design sub-team members busy focusing on finishing the robot by their deadline and Mechanical sub-team members waiting for part drawings or GCODE, there was no in-between that could focus on creating part drawings or automated tool paths using CAM software to run on our CNC mills. To bridge this gap, Integrations focused on creating part drawings and automated tool paths during meetings. As a result, Mechanical could machine earlier to match our more aggressive deadlines. It also made sure that parts were designed for the ease of manufacturing.

To keep track of parts, we use a system called the Manufacture Parts List. It notes the CAD, CAM, Drawing, Manufacturing, and Assembly statuses of all parts. It also records the COTS parts our robot needs to use. By implementing this system, we have been able to reduce the number of parts that we lose or accidentally make too many of. It helps facilitate communication by making statuses readily available to all.

Integrations is also responsible for maintaining inventory and ordering COTS parts. This year, an Inventory System was made. It was regularly updated whenever parts were taken out or added to the Inventory through a Google Form accessible to the whole team. This took stress off the Design Sub-team, allowing them to focus on designing while someone else ordered the parts. In previous years, we have had issues where the parts needed were ordered too late, holding up the assembly process. By pulling this task away from Design, we decreased wait time and were able to assemble the robot sooner.

PROTOTYPING

The prototyping specialist was a role created to improve and manage our team's prototyping process. To begin, a set of procedures was documented in order for everyone to understand what was trying to be achieved.

The first of three documents outlined how our team was going to move through the prototyping process during week one. We were going to start by selecting from a pre-developed list of protolords who would lead proto-teams. This was then going to be followed by assigning team members to proto-teams. The document then outlined how each proto-team should continue to work throughout the first prototyping phase. The second document outlined the methods which protoleads would use to develop useful and conclusive prototypes. The third outlined the steps that the prototyping specialist and related team members would expect to use during build season to ensure smooth assignment of duties to team members.
This whole process worked fairly well. The prototyping specialist was able to manage all the prototyping groups, keep them organized, make sure everything was documented, and then keep working on a few of the prototypes that needed more development. In comparison to past years, there were significantly more useful conclusions gained from the prototyping process, and this helped to streamline the initial work of the design subteam.