Technical Binder

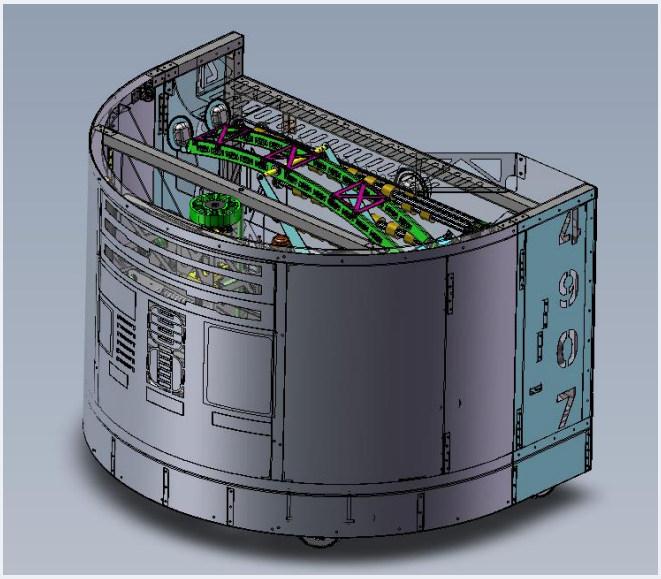
FRC 2020
Infinite Recharge

4907 ThunderStamps

INTRODUCTION

The 4907 robot design for the 2020 competition represents hundreds of professional design hours and countless small decisions that culminate in a single comprehensive design. Design is a process of maximizing the things we value (capabilities such as picking up power cells, shooting them accurately, moving quickly, and climbing) while minimizing resources like cost, manufacturing time, assembly time, and even design time.

Each decision affects various aspects of the robot in both positive and negative ways. For instance, adding strength in one area increases the weight which may require us to reduce functionality or strength in another area. This document attempts to explain the major components and some of the decisions that were made.

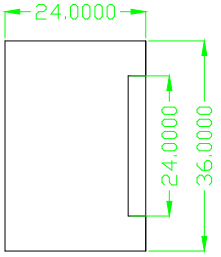


ROBOT SHAPE

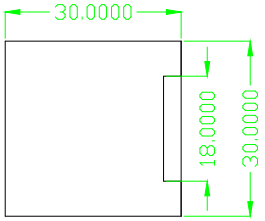
This year's robot sports a unique "D" shape, which came about based on a few factors. With the robot shape, we want to maximize internal area, but we have to stay within our 120" maximum allowable frame perimeter. Intaking balls from the floor was a "must-have" criteria, and we learned last year that we want as much "compliance" as possible. Compliance, in this sense, means how accurate you have to be when you're picking up balls. So, we want to maximize the intake width to make it easier for the driver to pick up balls.

Other criteria we have to consider is that we want to keep the "wheel-base" large, so we want the wheels far away from the center of the robot, which makes us more stable. We need to be able to fit through the trench run. We also need to consider how we get the robot through doorways at the competition, which are only guaranteed to be at least 32" wide.

Let's start by looking at how last year's shape, and the standard square robot shape stack up under these constraints:



Rectangle: 24" x 36"
Very close to last year's robot
Perimeter = 120"
Area = 864 sq. in. (worse than square bot)
Intake = 24" (good, roughly 3.5 balls)
Fits through trench run either way
Fits through venue doorways
Wheelbase and stability is OK, not amazing
Lots of corners for opponents to push on



Square
Side Length = 30"
Perimeter = 120"
Area = 900 sq. in.
Intake = 18" (poor, about 2.5 balls)
Fits through venue doors
Stable wheelbase
Lots of corners for opponents to push on

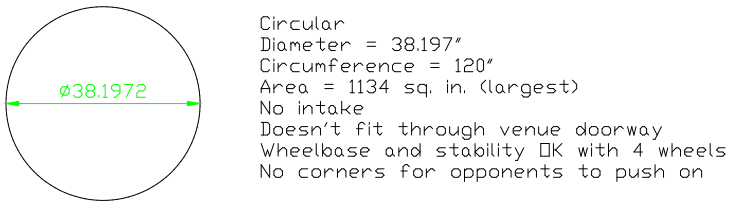
During the off-season, the mentors took on a project to see if we could build a round robot, and how you'd go about constructing it. There are actually some challenges to doing this, but we figured it out, and we had the students out before kickoff to explain how it could be done.

There are two reasons why a round robot is interesting, besides the novelty:

1. Effective defense bots (such as we were in 2019) would often come up and push on the corner of another robot while they were trying to place a game piece (or aim), so corners are a liability when you're being defended.

2. Round shapes have more internal area (for mounting stuff) than square or rectangular shapes.

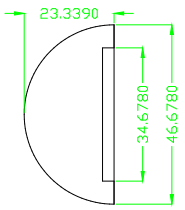
Let's look at a completely round robot with a 120" perimeter (the *Death Star* shape):



While it maximizes internal area and has no corners, a completely round shape has some issues with this year's game. We would have to intake balls over the bumper (all round profiles need to be completely protected by a bumper and can't have any cut-outs). Since we can only stick out 12" past the frame perimeter this year, that makes a fold-out intake particularly difficult. We considered it, but it's hard.

It also doesn't fit through doorways (we had considered this previously and the frame would have to be removable, or we'd have to turn the robot on its side, etc.). It's possible, but a challenge.

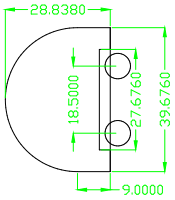
You really want a flat side to maximize your intake width, since you can only cut-out your bumper on a flat side. Let's take that to the extreme and make a semi-circle:



Semi-circle
Diameter = 46.678"
Perimeter = 120"
Area = 886 sq. in. (smallest)
Intake = 34.678" (largest, roughly 5 balls)
Only fits through trench run sideways
Fits through doors (sideways)
Not much stability in the short direction

This is a ridiculously wide intake, almost 5 balls wide. It has problems getting through the trench run if you want to go intake first to pickup balls in autonomous. It lacks stability in the shorter direction (it's tippy) and it has less internal area than a square robot.

After playing with some shapes a bit, we came up with a compromise, which is a "D" shaped robot:



Semi-circle with 9" rectangle on right
Diameter = 39.676"
Perimeter = 120"
Area = 975 sq. in. (more than a square bot)
Intake = 27.676" (very good, roughly 4 balls)
Fits through trench run either way
Fits through venue doorways
Wheelbase and stability is OK, not amazing
Only 2 corners for opponents to push on

This gives us a nice compromise. It has a wide intake, fits through the trench run comfortably and through doorways. It has significant internal area, and it has a decent stability with a nice wide wheelbase. It was further decided that if we oriented the shooter towards the top of the diagram, our round size would be facing the left (where the opponents are coming from) while we're aiming, which presents fewer corners to them, and keeps our intake side (which is more vulnerable) away from them when they're smashing into us.

WHY SWERVE?

There are only a handful of different drive systems seen in FRC: a few flavors of tank drive, mecanum drive, and swerve drive. Of these, swerve is the most advanced but requires the most technical ability, both mechanically and in programming.

With a tank drive, you can only move the robot forward and backward, and to turn you have to spin the wheels on one side slower or faster than the other side. This drive system is simple and powerful, but lacks the ability to "side shift" (also called "strafing").

Mecanum wheels give you the ability to side shift, which helps with lining up to a goal, but you lose a lot of pushing power.

Swerve drives use two motors for each corner: a drive motor to spin the wheel, and a steering motor which can steer each wheel independently. To go forward, we point all 4 wheels forward, and to go left, we can point all 4 wheels left. To turn, we can point all 4 wheels in a circle and turn on a point. Once you add some fancy math, you can turn to a different heading while driving the center of the robot in a straight line.

One way to control the robot is "robot-oriented" mode, which means you control it as if you're sitting in the robot, so forward on the joystick makes the robot move forward. This can be confusing when the robot is turned to a different orientation, including towards you. If you want to side-shift to your right, you need to learn that you have to move the stick left.

The more advanced way to control the robot is in "field-oriented" mode. To make this possible, the robot includes a special sensor called a "gyro". This measures acceleration, including the direction of down, but it can also measure heading. When the gyro starts up, it sets the current heading to zero, so as long as we know which way we're pointing on the field when we start up, we know which way we're pointing on the field the whole match. Using that sensor, when the driver says they want to move forward on the field, the robot uses the gyro heading to convert that into a robot-oriented heading, so if the robot is facing field-forward, it goes forward, and if it's facing back towards the driver, it knows it has to drive backwards instead.

Field-oriented swerve makes the robot much more intuitive to drive, so the driver can focus on the game, not the controls.

Our swerve drives from last year and this year are capable of a top speed of 120 inches per second (10 feet per second) and can accelerate and decelerate in excess of 1G, which means it can reach top speed in about 0.3 seconds. This places us in the medium-high range of FRC speeds, and top acceleration rates. In practice, acceleration is limited by how "tippy" we are. This year we have a low center of gravity, so we're not very tippy, so we can use most of our acceleration.

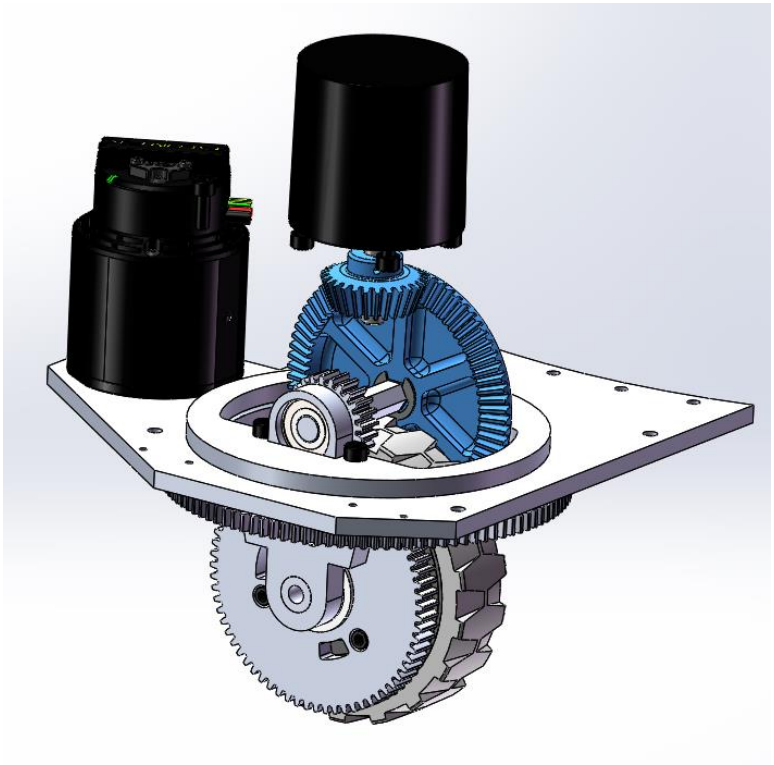
SWERVE DRIVE

This year saw a re-design of our swerve drive modules. Last year our swerve drive was a copy of the team 1533 design that was published online:

The top motor we had switched out to the new NEO with a 4:1 planetary gearbox, and the left motor we had swapped with a PG27 gearmotor with an integrated encoder. Otherwise in 2019 it was mostly identical. We still had the extra absolute angle sensor on the bottom to know what angle the module started at.

That design was light and elegant, but costly. The cost was around $500 US each, mostly of purchased components, meaning $2000 of our robot cost would be just in the wheel modules. This year we did some calculations in the off-season to see how much cost savings we could get if we eliminated the costly gearboxes and eliminated the purchased bevel gears.

In the 2020 robot, we changed everything except the NEO motor and the wheel. The motor on top is the drive motor (a NEO), and the motor on the left is the steering motor (the new Falcon 500). The blue bevel gears in this image are the core change in the design:

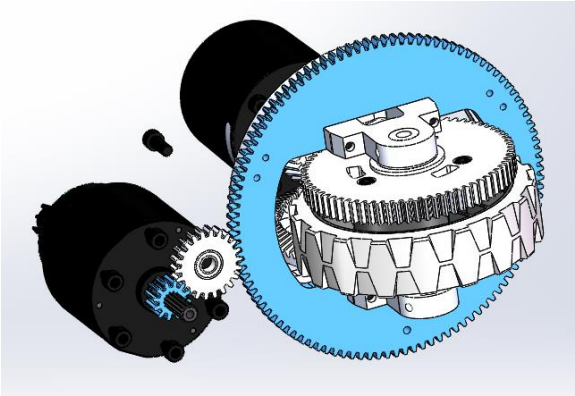


The old gear ratio from last year was a total of 8:1 (so 8 rotations of the motor gave us one rotation of the wheel) and we wanted to keep this the same. We also like the NEO for the drive motor because even though the Falcon 500 is more powerful, at lower speeds we're limited more by current, and the NEO can produce a little more torque per amp of current. Last year the 8:1 ratio was done with a combination of a 4:1 planetary gearbox on the NEO motor, a 1:1 purchased bevel gear set, and a 2:1 ratio in the "straight cut" (gray front) gears. This year we accomplished the same ratio in a combination of 76:27 in the bevel gears (about 2.8:1) and increasing the gear ratio in the straight cut gears to get the rest. This barely fit, and that was after we had to shave off the side of the wheel on a mill to get the large bevel gear to fit.

There were additional improvements: we were able to get the wheel precisely on the center of steering rotation (it was still slightly offset last year). There is also the problem that when the module steers, it turns the wheel because of the rotation around the small bevel. We have reduced this by a factor of 4 this year since the driving bevel is no longer after the 4:1 planetary gearbox.

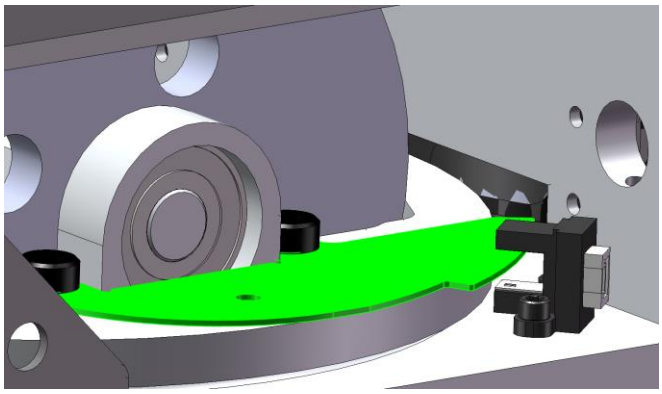
The small bevel gear that attaches to the NEO motor on the top, and all of the straight cut gears were all made on a Wire/EDM machine at ETBO over a weekend, and the large bevel gears were made on a 5-axis mill-turn machine at ETBO, but the teeth on that gear were cut on a Fanuc RoboDrill machine at ETBO. These were time-consuming to make, but the rules of the competition only require that you cost the raw material in this case, not the time that mentors spend doing it, which makes these gears much less "expensive" than purchased ones. Gears made like this are also noisier, so you'll notice our swerve system is louder than last year.

For steering, we replaced the PG27 gearmotor & integrated encoder with the new Falcon 500 motor with integrated Talon FX controller:



We changed the steering ratio on the bottom (blue gears) to a 108:12 tooth ratio, which gives us exactly 9:1, or 9 rotations of the steering motor to 1 rotation of the steering module. Note that the gear between the two blue gears is just an "idler" gear, and exists to allow the steering motor to be far enough away from the module to avoid hitting any rotating parts.

One of the important aspects of this steering design is that the Falcon 500/Talon FX comes with an integrated absolute encoder. Specifically, it means that when the motor powers up, we know where it is within 1 rotation of the motor. We want to use this to know what angle we're at when we power up (we need a reference to "module forward"). However, since it's a 9:1 ratio, we've only narrowed it down to one of nine possible steering positions. To narrow it down, we added a "home" sensor to the swerve module:



The green plate is a piece of aluminum that sticks out into the path of the optical sensor on the right. When it breaks the light beam on the sensor, we know the wheel module is within 15 degrees of our zero position (a 30-degree window). Since one revolution of the motor is 1/9th of 360 steering degrees (360/9 = 40) then we now know we're definitely within one rotation of zero, so we can use the absolute encoder on the motor to know exactly where we're starting from.

To reference itself, when the robot turns on, if it sees that the beam is broken, then it uses the absolute encoder to set its starting steering angle. If it's not on, then it turns the steering motor until the beam is broken, and then it knows where it is.

During match play, our technician will setup the wheels so that they are close to module zero, so they will home instantly and the robot can start driving immediately after power-up. In case they don't get setup properly, or we're running in practice, the swerve modules will home themselves by moving the steering until the beam is broken, which takes a little longer.

The result? All of these changes mean that instead of costing $500 per module with the motors and controllers, the BOM cost of each module is now under $275 each. That saves us at least $900 total on our BOM, which takes a lot of pressure off the other components we choose to use.

Given the difficulty a team would have in cutting their own bevel gears (it takes specialty equipment that's accessible to 4907 thanks to our generous sponsor, ETBO), that also means this design is difficult for other teams to copy, so it can be a competitive advantage in future seasons even with the rule that we have to release the design to use it again. We're very happy with the result.

CONSTRUCTION

This year we decided to switch our construction methods.

Last year we constructed most of our structure out of a combination of CNC milled parts with some laser-cut plates, all bolted together. Note that ETBO's laser cutter was brand new last year, and the robot project was one of the first things we used it for. We were quite happy with the results.

Comparing laser-cut parts to CNC milled parts, laser-cutting has several advantages for us:

- It's much faster to setup and cut a part on the laser-cutter than on the CNC mill
- We have far more mentors trained on how to use the laser-cutter than CNC mills
- The laser-cutter is perfect for thin materials (which are lighter and less expensive)

However, our laser-cutter can only handle aluminum sheets up to 1/8" thickness and it cuts quite slowly at this thickness. For thicker material, CNC milling or Wire/EDM must be used.

To deal with this, we've designed many parts of our robot out of 1" x 1" square aluminum tubing attached to laser-cut brackets. Many of the brackets are cut out of thin material and bent at an angle on a metal brake:

To fasten the brackets to the 1” x 1” tubing, or to other laser-cut pieces, we are using pop-rivet fasteners:

Pop rivets are a permanent form of fastening which are also strong, inexpensive, and light weight. You use them by drilling a hole through both sheets of metal, inserting the rivet, and using a rivet gun to pull the shaft until causes the opposite side to expand enough to form a strong bond, and then the shaft breaks off (it “pops” when this happens, hence the name).

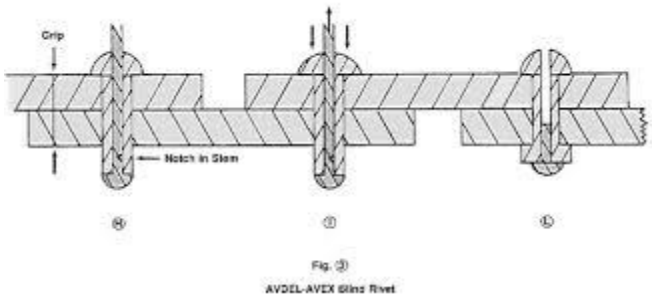


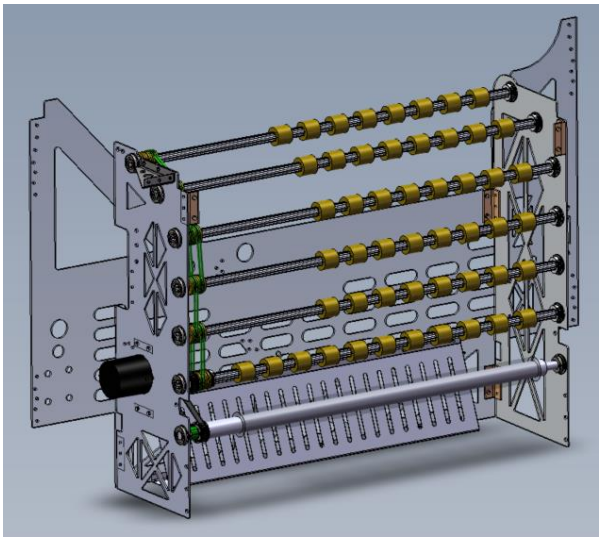
Fig. ③
AVDEL-AVEX Blind Rivet

Even though they are permanent, they can be removed by drilling them out with a drill bit relatively quickly. Still, there are parts which we want to remove regularly, such as the battery cover and some parts of the outside skin to allow us easy access to the robot’s internal mechanisms. For those cases we still want to use a bolt. Thick material can be threaded to accept a bolt directly, but thin material doesn’t have enough thickness and the bolt will strip the threads. For these cases we’re using rivet nuts:

Rivet nuts are installed like a rivet, but the inside of the rivet nut is threaded (like a nut) and there are many more threads so they are quite strong.

INTAKE

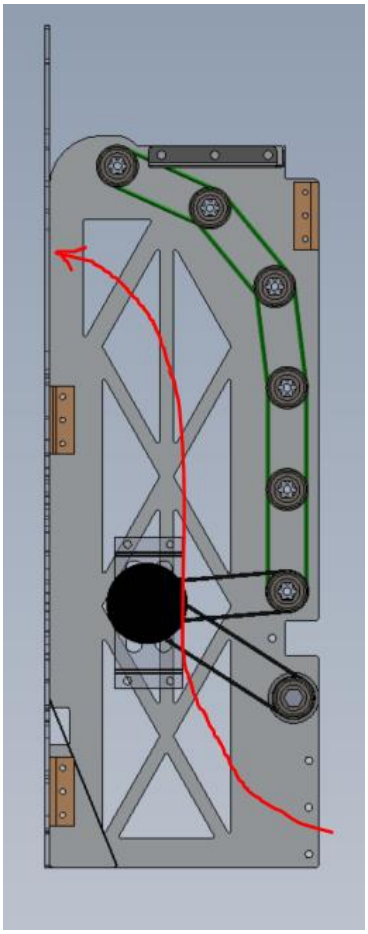
The ball intake was something we prototyped out of plywood and 2x4's quite early. Our goal was to have a wide intake, but to have it internal to the robot to avoid damage from other bots. The final intake design is completely within the frame perimeter, and it's 27" wide, so we're happy with the result:



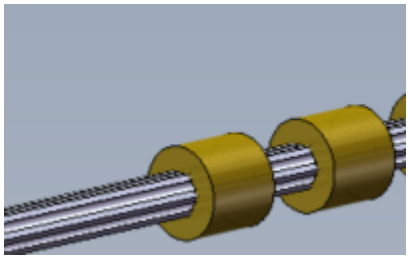
The bottom bar is the main intake roller. We originally designed it to be solid aluminum wrapped in a rubber hose to make it as strong as possible in case it was in a collision, but we changed the solid bar out to a hollow tube during construction due to weight concerns. Still, it's quite strong.

All of the rollers are continually rotating while intaking balls. The bottom roller squeezes the balls slightly against the diagonal slotted plate at the bottom behind it, and then the balls travel up the interior wall and then get rolled over the top behind the wall into the "hopper" area. A single NEO motor (black cylinder on the left) drives all the rollers.

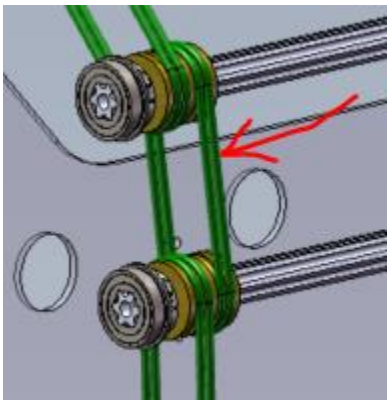
The direction of the balls is shown by the red arrow:



The upper rollers are constructed out of hexagonal “churro” rod, which is often used in FRC due to it being light weight and inexpensive. The downside of churros is that they’re not very strong. These rollers are inside the frame perimeter and protected by a lexan/aluminum skin over top, so they’re well protected from damage. The small yellow rollers on the churros are rubber cylinders that are purchasable in quantity for less than $5 US each, so we don’t have to count them on the cost of the robot.

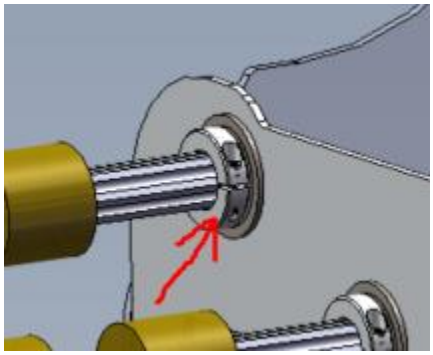


To drive the rollers, we have small round belts that connect each roller to the next one, so the single drive motor can drive all of them:

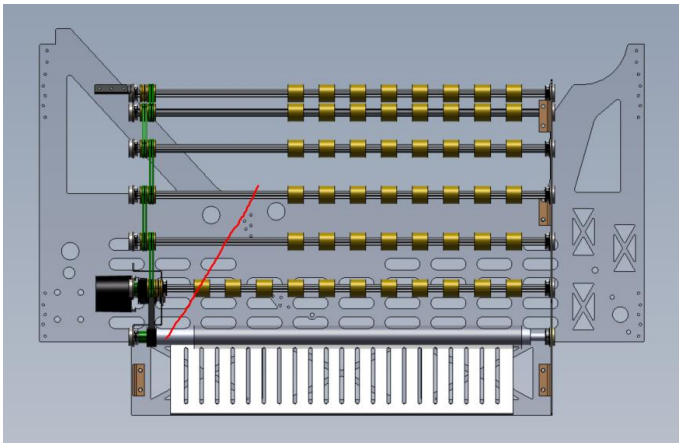


The pulleys that the belts ride on are the same rubber rollers with grooves cut into them using a grinding wheel.

We doubled them up in case one breaks during a match. The roller bars are fairly easy to remove for maintenance. Just loosen the shaft collars on either end, and you can slide the flanged bearing out of the hole in the plate and then take the roller out:



While we can intake balls along the entire width, the balls must fall into the right side of the hopper due to the shooter being at the left end of the hopper. In the final construction, there is a deflector plate that gets mounted in the left which diverts balls towards the right (see the red line below). The exact position and angle of this plate isn't designed and is set based on experimentation:



In testing with the robot stationary, we're spinning the intake roller at 1800 RPM (rotations per minute) which is 1800 / 60 = 30 rotations per second. The main intake bar has a rubber hose on the outside with a diameter of roughly 1.5". That means the outer perimeter of the roller is 3.14 x 1.5 = 4.725". If we're rotating that bar at 30 rotations per second, then the outer circumference of the rubber hose is moving at 4.725 x 30 = 141.75 inches per second.

If you recall, the robot has a top speed of around 120 inches per second. To get the ball to roll up off the floor, the outer circumference of that roller must be moving faster than the floor is moving past us underneath the robot, or else we'll tend to push the ball rather than pull it up the intake wall. While 141.75 inches per second is faster than the 120 inches per second top speed of the robot, this is something that needs testing in practice. If the balls don't pickup well when we're moving fast over the ball, we can use the robot forward speed signal to increase the intake roller speed when we're moving faster. This is relatively simple in software, and the NEO motor has a top speed closer to 5000 RPM, so we could easily double the intake speed when we're going faster without changing anything mechanically.

BALLS STICKING

One of the challenges of manipulating the yellow balls this year is that they tend to stick together and jam if you try to roll them together along a flat surface:

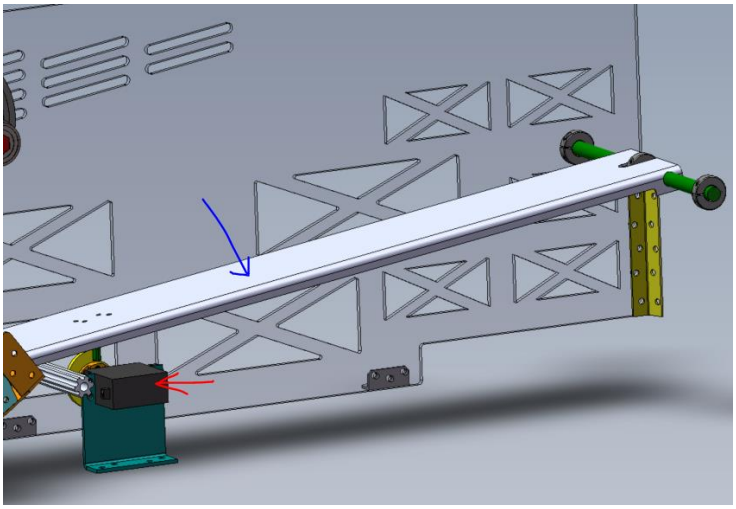
There are several known ways to solve this. In the picture above, it's likely they're running the belt faster than the intake roller, so the balls will stay apart. Another well known technique is to use two belts to convey the balls from opposite sides (sorry for the blurry YouTube picture):

In previous years this problem was solved by using multiple single rollers rather than a belt, and that's the method we're using (I apologize but I've since lost the reference). One thing we could also do to help separate balls is to stagger the pulley ratios between each roller and the next, so each successive roller moves slightly faster than the previous one. If necessary, we can also do that, but it seems to work well as-is.

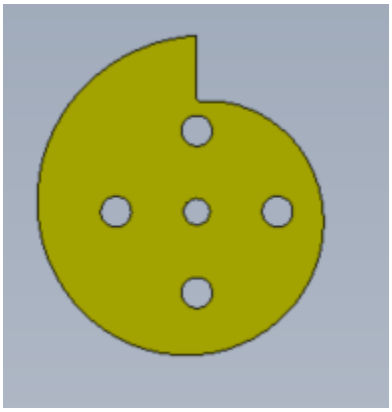
HOPPER

For ball storage and feeding balls towards the shooter, we selected a gravity-fed hopper design during prototyping. Just like in the intake, the balls will tend to stick together in the hopper and won't want to roll down to the shooter.

We prototyped a couple of different "anti-jam" ideas to separate the balls and keep them rolling, including an upward-running conveyor on the bottom, but we eventually settled on a simple solution of jostling the bottom of the hopper up and down, which seems to cause the balls to separate enough to keep rolling down towards the shooter mechanism. The hopper floor is indicated by the blue arrow:



The red arrow indicates a high torque R/C servo that rotates a disc with increasing radius. This pushes up on the hopper floor causing it to rise until it abruptly falls back down. The profile of the disc looks like this:



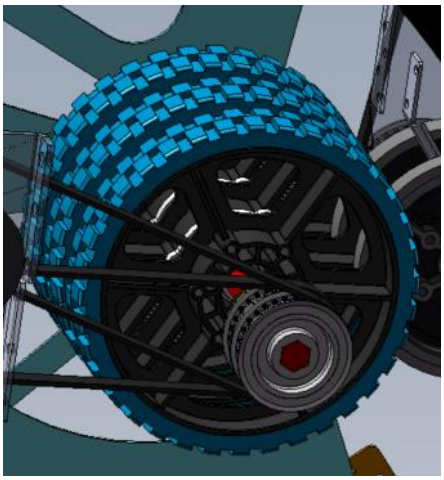
Most R/C servos can't rotate continuously in one direction but we purchased one that can. This may not be the final design and we can cut many different profiles to try to get the jostling action that we want.

SHOOTER

The core of our robot is the shooter, which shoots balls at the target. Our must-have goal was the ability to reliably shoot balls into the hexagonal (2-point) target from 40 feet, and it would be nice if our accuracy was good enough to get a significant number of balls into the 3-point target from 35 feet or further.

Most teams are either using a single or double flywheel shooter. We lean towards a single flywheel shooter for both accuracy and simplicity. The operation of the flywheel shooter is complicated, but let's take it one step at a time:

The core of our flywheel shooter is a spinning 6-inch flywheel made of blue traction wheels:



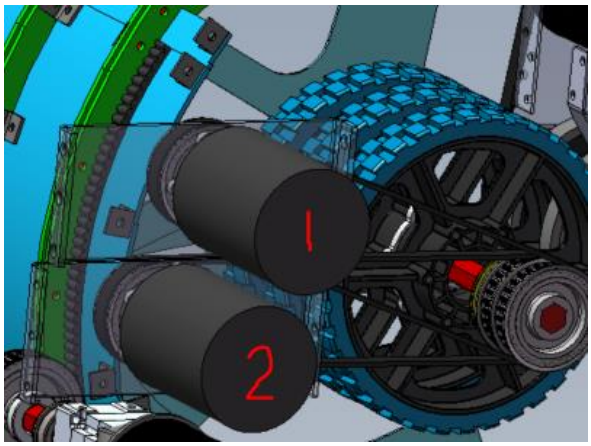
When shooting we accelerate this flywheel up to well over 5000 RPM. In testing, with one Falcon 500 motor we can reach 5300 RPM, and with two Falcon 500 motors we can reach over 5600 RPM. This is with both geared to 1:1 ratio. With two motors we can also spin up faster and recover the speed of the flywheel faster between shots.

However, we don't have enough channels on the power distribution panel (PDP) to support a second shooter motor unless we give up another motor somewhere. Most likely this would have to be the control panel spinner mechanism, which we will cover later.

With one Falcon 500 motor, we can spin up the flywheel in a little over 2 seconds, and we can shoot balls and hit the target from around 40 feet at 1 second intervals. This meets the original design requirement ("must-have" list). With two motors we can spin up in closer to 1 second and shoot at about 0.5 second intervals. Note that we can probably be spinning up the shooter while we're aiming, and 0.5 seconds per shot is likely faster than the hopper can reliably feed balls to the shooter, so it's not clear how much time we could save with two motors.

Using two motors does give us more launch velocity, which will give us more accuracy at longer range, but it's not clear yet how much we'll need that.

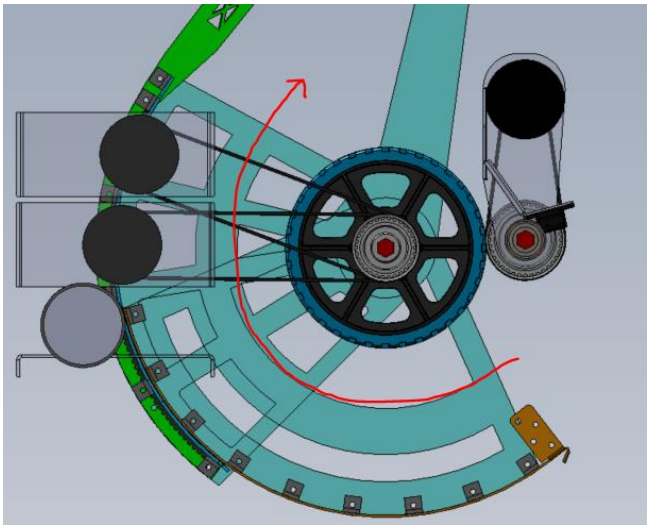
These are open questions that we've elected to defer due to lack of time, but we've left ourselves flexible by designing in room for two motors:



Update: the constructed flywheel can reach higher speeds than the prototyped one, so we are able to reach 5700 RPM (barely) with a single motor. Also, we ground off the tread on the blue traction wheels for 2 reasons:

- Reduce compression from 2" to about 1.75" which seems
- Avoid damage to the balls (the ball skin was becoming delaminated from the inner foam after many shots)

Balls are accelerated by the shooter flywheel spinning in the clockwise direction (red arrow):



The balls have a diameter of about 7" and the space between the flywheel and the outer hood is 5", which gives us 2" of "compression". The amount of force exerted by the ball against the flywheel increases with the amount of compression, but the amount of work it has to do to compress the ball also increases, which takes energy (i.e. speed) from the spinning flywheel, which will reduce the exit speed of the ball.

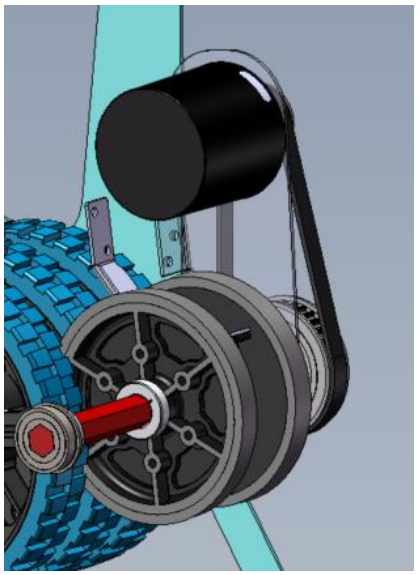
The balls are constructed of a spongy material and they're full of air, but they're covered by a rubber skin. The skin is airtight except for a small hole. The air can't come out of the ball instantaneously, but the ball moves through the shooter almost instantly. The flywheel is spinning around 5000 RPM, or 83 rotations per second so it spins the 1/3 of a rotation that the ball is in contact with the flywheel in about 1 / (83 x 3) = 0.004 or 4 milliseconds. Of course, that assumes instantaneous acceleration of the ball, which is completely false, but suffice it to say that the ball is in the shooter for a very short amount of time. There's almost no time for air to escape the ball, which means it returns to its round shape very quickly after exiting the shooter.

The other form of energy loss is drag (friction) between the ball and the outer hood. To combat this, we're coating the hood in a slippery type of plastic called UHMW, a type of polyethylene. This plastic is very slippery (similar to Teflon) but is more durable and it's often used in industry. For prototyping we purchased a roll of 1" wide UHMW adhesive tape from Amazon for $20, but for the actual robot we're using a wider roll of adhesive UHMW purchased from an industrial supplier.

So, the flywheel is doing a lot of work in a very short time: compressing the ball, accelerating it, and sliding the ball against the outside hood. All this work causes the flywheel to slow down. The motors will respond by increasing torque (rotational force) as soon as the flywheel starts to slow down, but even two motors aren't capable of providing enough power to the flywheel fast enough to keep it at full speed. The energy stored in the speed of the flywheel is most of the energy used during the shot. The motors then "recharge" the flywheel by speeding it back up between shots.

FEEDER

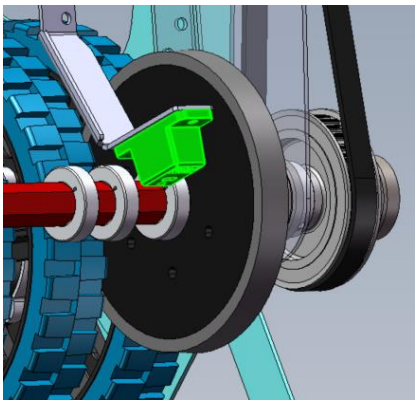
Balls in the hopper need to be "fed" into the shooter when we're ready to shoot a ball. This is the job of the feeder:



The feeder wheels rotate clockwise to push a ball from the hopper into the flywheel. (Note that these wheels are bigger than our final design.) The NEO motor (top) is initially at rest and rotates very quickly to roll the ball when the command to shoot a ball comes from the operator.

As soon as the ball contacts the flywheel, then the flywheel will pull the ball in very quickly, doing most of the work. Since the feeder will be rotating slowly relative to the flywheel, it's important that the feeder doesn't compress the ball too much, since this could allow air to escape, and will prevent the ball from returning to the full size when exiting the shooter, and may affect the accuracy of the shot.

There is a sensor located between the feeder rollers for detecting a ball (green):



The robot program will rotate the feeder until a ball is detected ready to be shot, and then will stop, holding the ball ready to feed.

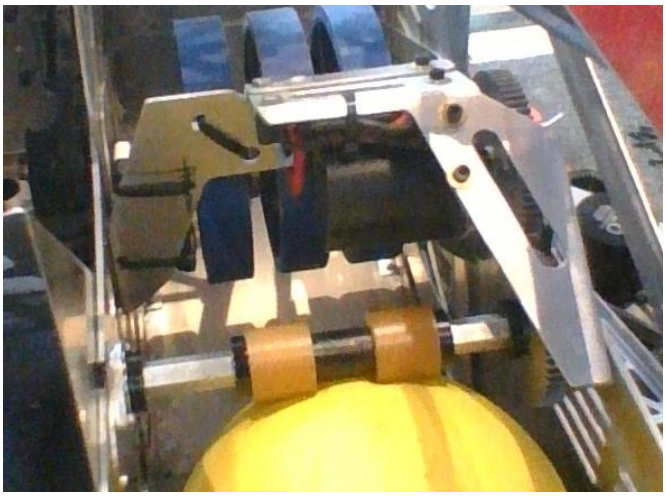
FEEDER VERSION 2

The originally designed feeder with the 4” wheels was creating too much compression on the ball. We want to limit the amount of compression as much as possible until the ball reaches the flywheel because we want to keep all the air in the ball to make the ball regain its shape quickly after exiting from the flywheel.

The original feeder also had the motor mounted up too high, and the ball exiting the flywheel was travelling at a lower trajectory that we originally thought, and it was hitting the motor on the way out, unless we were firing at a fairly high angle.

We designed and fabricated a completely new feeder with the following differences:

- Motor is mounted lower
- The intake wheel now uses the rubber rollers used on the intake
- Much less ball compression (almost none)
- Uses gears instead of belts so that we can get the motor and roller closer together
- The roller is much farther away from the flywheel, so it now releases the ball before the ball touches the flywheel, and this means the feeder and flywheel aren’t “fighting” each other anymore as they’re not both in contact with the ball at the same time

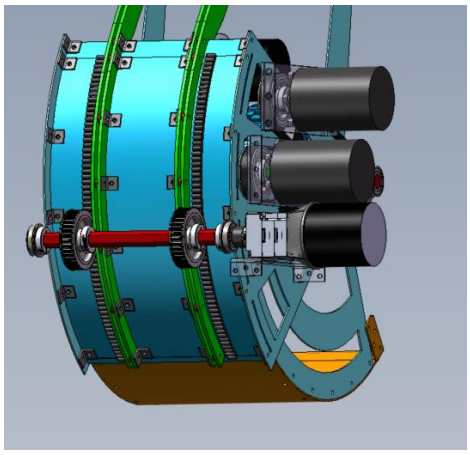


ANTI-JAM

The hopper floor shaker and feeder work together to prevent balls from jamming in the hopper. Both run until the feeder sensor senses a ball almost in position. Then the hopper floor shaker stops shaking, and the feeder motor moves the ball into a “ready” position where it’s ready to fire. It uses the feeder sensor to know where to stop the ball.

LAUNCH ANGLE

The hood is split into two parts: the fixed orange part at the bottom, and a movable blue part at the top:



We call this “articulating”. The black NEO motor connected to the red hex shaft in the picture above is the launch angle adjustment. The two pulleys on that shaft have teeth that mesh with a timing belt (a belt with teeth on it) that we connect to the back of the moving portion of the hood. When the NEO motor rotates, it causes the blue section of hood to rotate up and down, rotating around the flywheel axis. This allows us to change our launch angle from a shallow angle around 10 degrees from horizontal, up to a rather steep 60 degrees if we’re lobbing into the goal from close to the target.

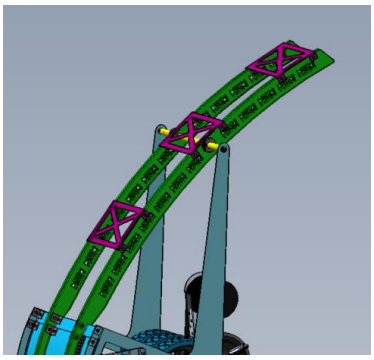
The NEO has a built-in position encoder, like all brushless DC motors, but we still need a fixed point of reference to know where we’re starting from. To accomplish this, we’ll include a home sensor (not shown). When the robot starts up, it’ll move the hood in the down direction until the home sensor turns on (or in the up direction if the sensor is on when it starts up, until the sensor turns off) and record the location where the sensor changes as the “home” position. The robot program will be pre-programmed to recognize that as a known launch angle.

There is a gearbox on the launch angle adjustment motor. We’re not certain what gear ratio to use here, so we’ll likely start at around 16:1 and adjust as necessary. More ratio means it’s easier for the motor to move the hood, but it can’t go as fast. There’s likely a range of ratios that will work, but some experimentation will let us find an optimal ratio.

Update: after initial testing, we switched from a 16:1 gearbox to a 63:1 gearbox because it was moving too fast, and since the NEO motor only measures 42 counts per motor revolution, we wanted more accuracy. Since it now takes almost 4 times as many motor rotations to move the hood the same angle, we now have better hood angle accuracy, which is important for the longer-range ball shooting.

FINGERS

The curved green bars in this picture are the “shooter fingers”:

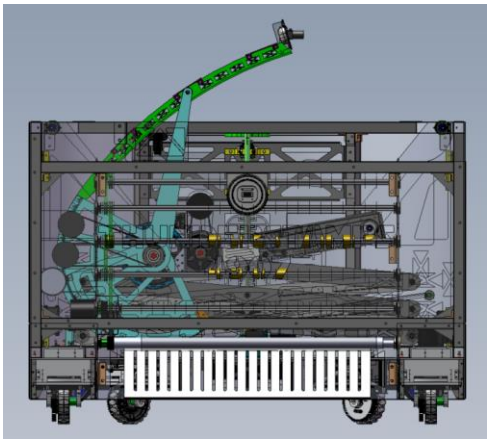


These fingers serve several purposes in addition to just looking cool. Mostly we need them for two reason: accuracy and height.

As the ball exits the shooter it will be returning to a round shape very quickly and, and we’re also not providing much left/right guidance within the shooter as it goes around the flywheel. The flywheel stage’s purpose is only to accelerate, not to stabilize. The fingers are there to stabilize the ball trajectory (path) after it leaves the shooter, mostly in the left/right direction, but also by imparting some more back-spin on the ball as it rolls along the fingers. Some back-spin is good because it imparts more left/right stability during flight, and it causes rebounds inside the target to send the ball down into the power port rather than back out of the hex (we hope).

Note that the hood and fingers have to move all the way to the forward (closed) position in order to be low enough to fit through the trench run.

Since our robot fits under the trench run, it has to be less than 28 inches high (our robot is actually 27 inches high). If we’re trying to shoot over the control panel while staying inside the protected area of the trench run, or if we’re trying to shoot over a robot that’s defending us, it’s helpful to shoot balls from a higher point. The curve of the fingers allows the ball to go up higher before it leaves the control of the shooter, effectively raising the launch height without moving the heavy shooter flywheel up higher. We’re also mounting our vision camera and LIDAR (range finder) on the end of the fingers so they can see over the control panel or other obstacles.



FINGERS – TESTING

Using a slo-mo function on a smartphone, we were able to determine that the ball isn't rolling up the fingers as we'd like. We can see that it's in contact with the fingers at the beginning because we can see yellow through the holes in the fingers:

However, by the middle position, the ball has lost contact with the fingers:

It then comes close to contacting the fingers again at the end:

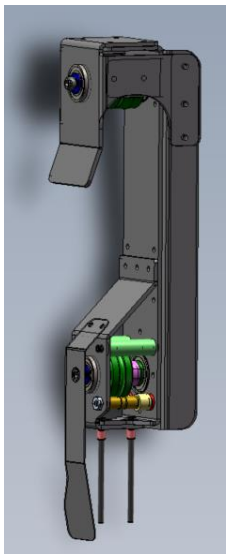
Zooming in, we can definitely see the ball between the fingers again at the end:

Our hypothesis is that after exiting the flywheel, the ball is decompressing and expanding. Since the top of the ball is behind held fixed by the fingers, the ball can only expand in the downward direction, which is altering the trajectory and causing the ball the "miss" the fingers on the way out of the shooter.

After the first competition in Durham, we will revisit this problem. As it is, it works well enough to hit the hex target most times from our desired range, and we don't want to do anything to make it worse.

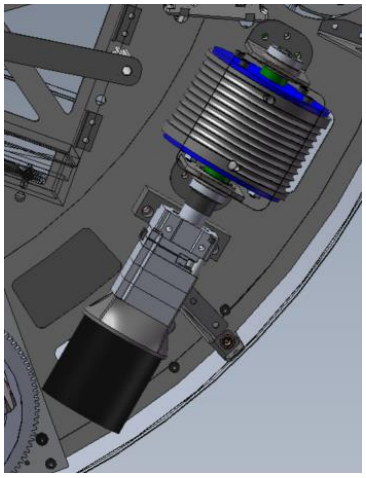
GRAPPLER

Our design for hooking on to the generator switch is a kind of hook we're calling a "grappler":



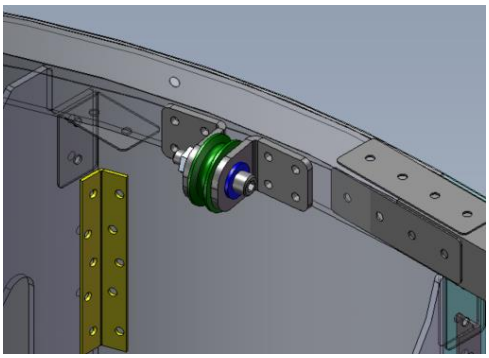
A scissor lift mechanism will be used to raise the grappler up to the height of the generator switch bar and hook on. Once attached to the generator switch bar, a length of paracord that loops over the bottom Coulson wheel (green) will be used to winch the robot up and suspect it from the switch.

There are two winches on the bottom of the robot that will reel in the paracord to raise the robot:



A NEO motor is shown, and we know from YouTube videos and last year's competition that two NEO motors are capable of lifting the robot weight. We've since adjusted the design to use two Falcon 500 motors, both to save weight and because the Falcon 500's are more powerful and have better current limiting.

The paracord runs up through slots cut in the base of the robot, and then around pulleys attached to the top frame of the robot:



It's important to keep the weight of the robot suspended from somewhere on the top of the robot (above the center of gravity) to prevent the robot from wanting to flip over.

The grappler, as designed, is about 2.2 pounds, which is fairly light but it's still a bit over-engineered. We're looking at taking weight out of it by making it slightly smaller, making it out of thinner aluminum, and designing some holes into it. Making it lighter will also reduce the size and strength of the scissor lift required.

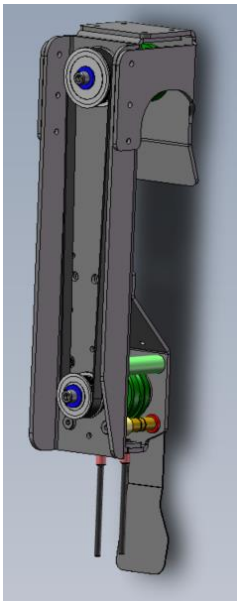
Update: we've re-designed the grappler to be smaller and lighter. It now weighs 1.6 lb, and fits comfortably inside the width of the scissor area, and can now stand straight up and down with the scissor in the down position.

SIDE SHIFT

Running both winches in the retract mode will raise the robot, and the gearboxes on each winch have a special ratchet mechanism in them that will prevent the robot from falling when the power turns off (it only allows the motors to reel in the paracord, and not let it out).

The ratcheting mechanism can temporarily be turned off to allow paracord to be spooled out again. We are planning to use small R/C servos to momentarily turn off the ratcheting mechanism.

To side shift, we'll take advantage of having two winch mechanisms. By spooling out one, and retracting paracord on the other, we can cause the paracord to turn the bottom wheel on the grappler:



This is connected via pulley mechanism on the back of the grappler so that it will spin the upper wheel, which is riding on top of the generator switch bar. That will cause the top wheel of the grappler to drive itself along the bar, allowing us to side shift in order to balance the generator switch for the extra 15 bonus points.

In prototyping, this worked as long as the robot stays attached to the grappler, which will keep the robot hanging relatively level underneath the grappler. If we don't leave the grappler attached to the robot, the robot will just want to rotate to one side in mid-air.

SCISSOR LIFT

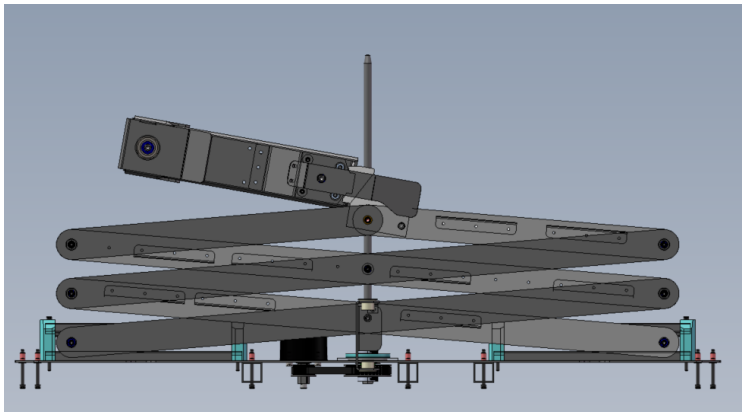
There is no doubt that scissor lifts are notoriously difficult mechanisms to get right, and we wouldn't choose such a mechanism without a reason.

The constraints of the competition mean that if you want to fit under the trench run, like us, then you're less than 28" tall, and you still need to have some ground clearance (let's say 2"). So, if you're 27" tall like us, and you have 2 inches of clearance, the tallest mechanism you can put in your robot is 25" tall. Let's say two feet to be conservative. Our grappler is about 12" tall and in order to get it to grab the bar at the tallest possible position, we need to get the top of the grappler up to about 7 feet high (or the bottom of it up to 6 feet high).

Due to our height restriction, if we use a standard multi-stage elevator design, then each stage is limited to somewhere less than 24" or two feet of lift. Now matter how you do the math, you need at least 3 stages to get up to your full height, and even that isn't very practical. You might be looking at a 4-stage elevator, which is very rare.

Needless to say, this would all be easier if we didn't fit through the trench run, but we want to fit through because we want to use the swerve wheel design that we're comfortable with from last year, and it won't go over the barriers without slowing down significantly. We also want to use the protected trench run area to protect ourselves from defenders, and be able to get all 5 balls from the trench in autonomous mode (ideally).

The one thing a scissor lift does it convert horizontal distance into vertical distance, because each stage of the scissor lift lies down flat:

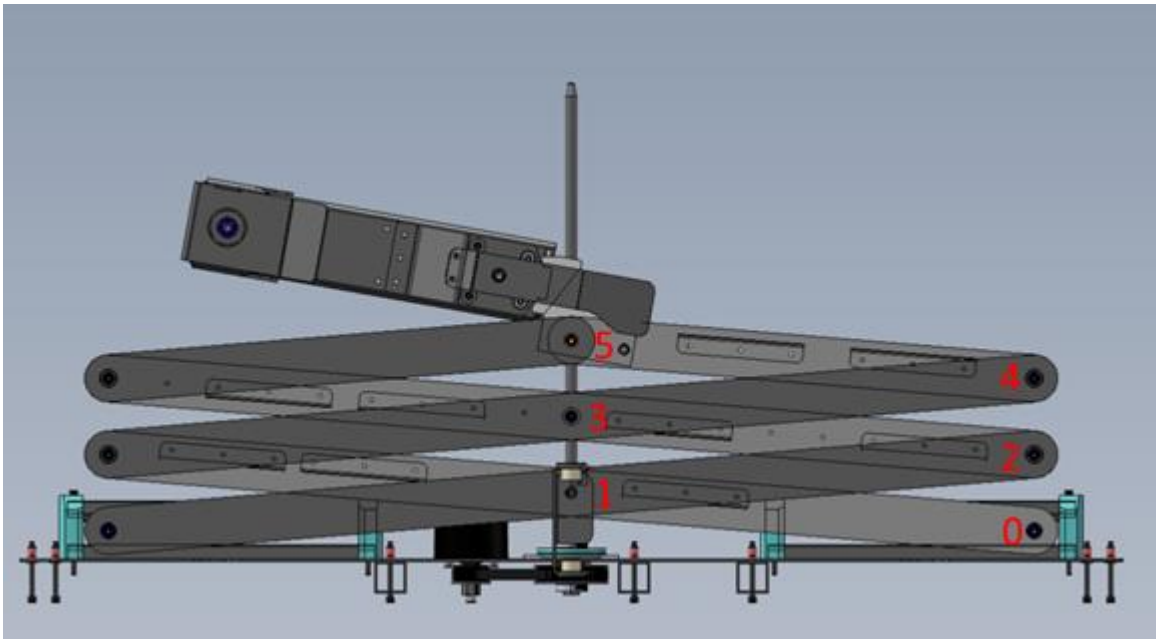


As you can see, the scissor lift folds down very flat, convenient for our short robot, and yet it can still reach up quite high, easily high enough to reach the highest point on the generator bar when fully extended.

It works by the motor (black NEO shown inside the scissors) turning a pulley on the bottom that drives a threaded rod in the center. The threaded rod goes through a nut that's only connected at the lowest cross joint in the middle.

When you turn the threaded rod, it will push the lowest center pivot up by 1/16 of an inch for each rotation of the threaded rod, so if you turn it 16 times, that point will go up by 1 inch.

The scissor effectively multiplies that distance at each pivot point, like this (see the red numbers):



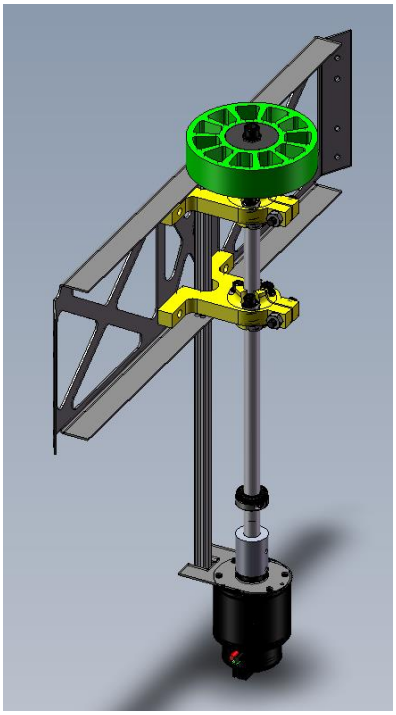
So, if you raise the pivot labelled “1” by 10 inches, then the top pivot labelled “5” will go up by 50 inches, for 5 times that distance. You don’t get this for free, however: the weight you’re lifting at the top requires 5 times as much force when you lift it from the lowest pivot point. So, if the grappler weighs 2 pounds, it will be as if the motor is lifting 10 pounds at the first pivot point. That just means you need 5 times as much torque, or rotational force, from your motor.

In testing, the NEO motor had no problem lifting this weight. This is not surprising since we’ve seen videos where teams are using 2 NEO motors to lift their whole robot. The robots are around 150 pounds each, so each of those 2 NEO motors is lifting 75 pounds relatively quickly. That means one NEO should be able to lift 10, 15, or even 20 pounds of weight on top of a 5-to-1 scissor lift like this with no problem. In engineering we often use “rules of thumb” like this to figure out if something is feasible before we invest a lot of time in designing it.

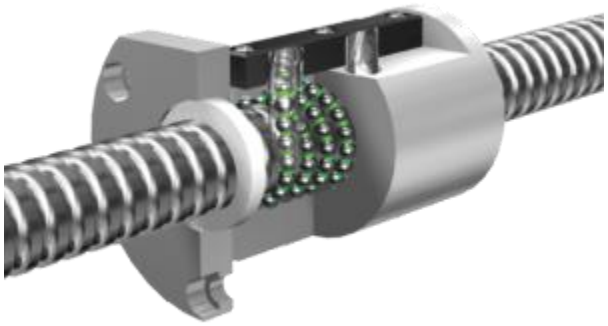
Update: After building the scissor lift it was too “floppy”. This was resolved partly by building a smaller grappler, but also by taking the “arms” of the scissor lift and using a shop press and a small die that we fabricated to form ridges down the length of the arms. This stiffens the arms in the flat direction and does it without adding any weight.

CONTROL PANEL SPINNER

As mentioned earlier, our two big constraints on this robot are weight and a limited number of motor channels on the PDP. I think our control panel spinner mechanism is a really creative way to work within those constraints, as it's light, and only uses one motor to perform both the up/down motion to deploy and retract the wheel, and uses the same motor to spin the control panel spinner wheel:



The core of this mechanism is something called a "threadless ballscrew." A regular ballscrew is a round shaft with grooves cut in it in a spiral pattern, and it has a carriage that also has grooves on the inside with ball bearings that roll through those grooves. A ballscrew is shown in this cut-away image:



As you turn the shaft of a ballscrew, the carriage will move along the shaft. However, ballscrews are both expensive and they have a limited range of travel, so you have to stop turning the shaft before the carriage reaches the end of the grooved rod.

In a threadless ballscrew, the shaft is a simple round shaft with no grooves cut into it. The carriage has radial bearings mounted on an angle around the shaft and there's adjustable pressure that pushes the bearings against the shaft:

When you turn the shaft in a threadless ballscrew, the bearings will roll against the shaft, but they're mounted on an angle, so they'll exert a force on the carriage pushing it along the shaft. Most interestingly, if the carriage reaches the end of the shaft, and we block it from travelling further, the bearings will just slip against the surface of the shaft, so the shaft can keep spinning but the carriage will stop moving.

Our mechanism uses this to get two motions from a single motor. The motor and wheel are connected to the ends of the round shaft. When they start to spin, the shaft (and wheel and motor) are pulled upwards by the two yellow threadless ballscrew carriages.

Eventually, after travelling up by 6 inches, the shaft collar (and thrust bearing) indicated by the red arrow will hit the hard stop (indicated by the blue arrow). At that point, the motor continues rotating the shaft, so the green wheel keeps rotating, but the shaft, motor, and wheel stay at the same height. To lower the mechanism, we run the motor in the reverse direction.