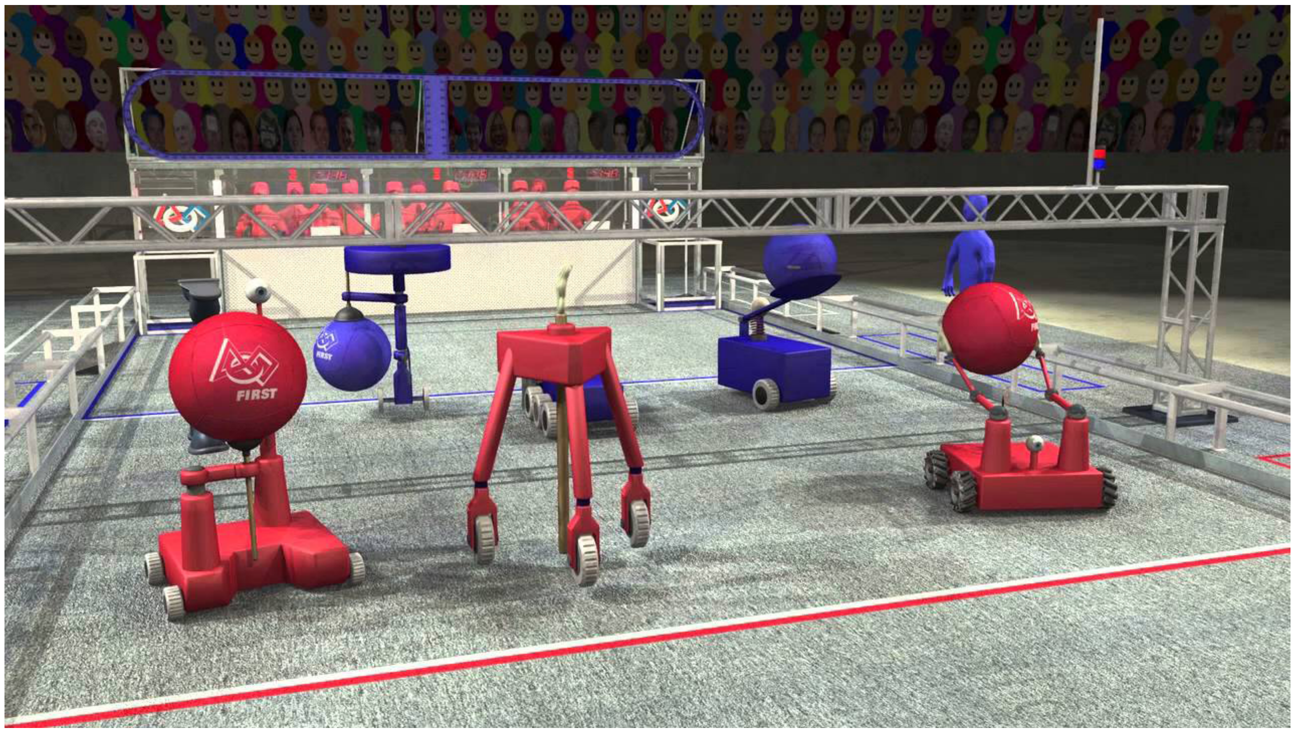
254

2014 Technical Binder

Table Of Contents

Kickoff Game Analysis: Goals, Priorities, and Requirements



Strategies and Goals

Cycle Configurations

We created a model of gameplay and scoring to determine other key requirements that would drive the design.

- The spreadsheet tells us what types of cycles would be the most efficient and reliable way to score points

- The user makes simple assumptions on the length of time per action and the probability of successfully completing that action in time

INPUTS	*Seconds to Completion*	*Probability of Completion*	*Point value*	*Cumulative Probability*	*Expected Value*
Time to score high	10	75%	10	75%	7.5
Time to hurdle	8	100%	10	100%	10
First assist	10	80%	10	60%	6
Second assist	10	75%	20	45%	9
Ball return time	10	100%	0	100%	0
Catch	15	10%	10	10%	1

RESULTING TELEOP SCORES	*Best Case score*	*P(Best Case)*	*Expected Value*	
One robot scoring	70.00	75%	52.50	Bad
One robot hurdles and scoring	100.00	75%	87.50	Great
One assist, then score	93.33	60%	63.00	Bad
One assist, hurdle, then score	110.53	60%	86.58	Great
One assist, hurdle, catch, then score	105.66	6%	64.72	Bad
Two assists, then score	140.00	45%	78.75	Okay
Two assists, hurdle, then score	145.83	45%	94.79	Great
Two assists, hurdle, catch, then score	133.33	5%	74.44	Okay

Autonomous Scoring

We estimated average teleop scores to be about 80 points. You can score a maximum of 75 points in atonomous. Thus, 1/2 of the total points come from autonomous which makes it the number 1 priority requirement overall.

- The only way to guarantee a maximum autonomous is to score all 3 balls yourself

This was the major driving force behind the entire design of the robot.

Requirements and Priorities

1) 3 Ball Autonomous

- Pick up and accurately (> 75%) shoot 3 balls in under 10 seconds into high goal
- Ground intake necessary
- Identify the Hot Goal and shoot all 3 within the 5 second hot goal window

2) Assist

- Receive and pass back and forth
- Should also be able to score in the low goal

3) Avoid Defense

- Quick drivetrain (faster than 15ft/s)
- Shoot from up high on the robot, over opponents
- Robust enough to not break if smashed into at full speed
- Minimize extensions of robot outside the bumper zone

4) Truss Shoot

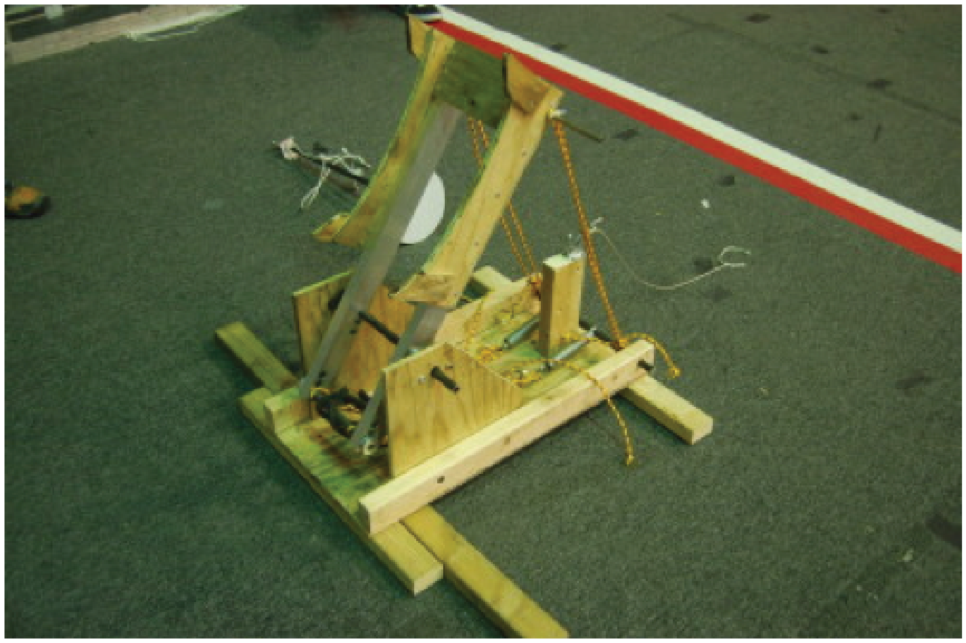
- With high goal shooting, truss shooting should be possible with little to no modifications

5) Other Considerations

- Catching is not included in the top 3 types of cycles, so it became the lowest priority
- Defense is also not a priority because it's more likely to win with efficient ofense than constant defense
- We wanted the ability to score points even if partners failed
- We wanted to be able to fulfill any part of a cycle

Brainstorming, Prototyping, and Design Selection

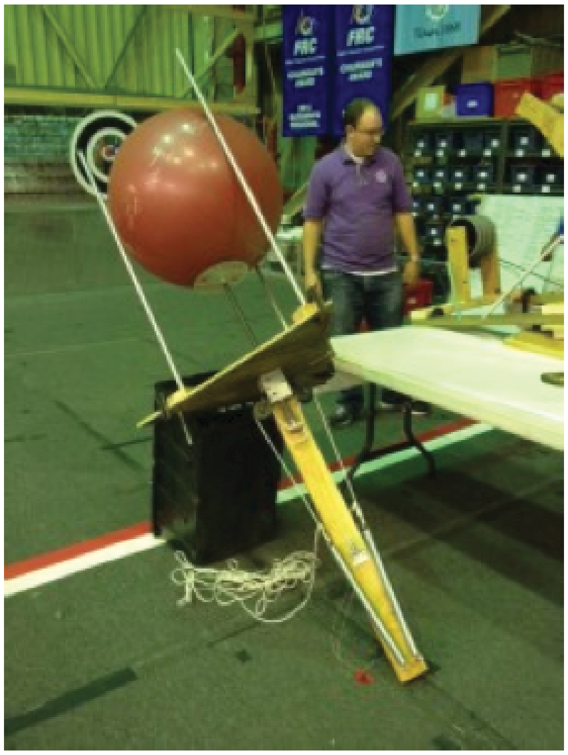
During brainstorming, we looked at past games and came up with three feasible options for shooting. We then prototyped each one of them and gathered data to determine which was best. The type of shooter we use would go on to determine the rest of the design. The top driving condition of the shooter was its ability to score three balls quickly in autonomous. Thus, the shooter we chose would need high shot consistency and a fast reset time.



Catapult

Built with surgical tubing and springs, using different release angles and cup shapes

Pros	Cons
• Low Center of Gravity • Easy to manufacture	• Trajectory of ball starts low to the ground, so it is easy for another to block the shot • Using stored energy requires a long reset time at least 2 second which is too long for a 3 ball auto • More difficult to manipulate and load 3 balls during auto. • Our prototype shots were less con-sistent not easy to settle the ball



Linear Puncher

Built with different kinds of springs, pulleys, and static guide rails

Pros	Cons
• High trajectory, can shoot over other robots	• More difficult to manipulate and load 3 balls during auto
• Shots were fairly consistent	• Time required to intake, aim, shoot, and reset takes too long (At least 2 seconds, which is too long for 3 ball auto in the hot goal)
• Works with over and under inflated balls	

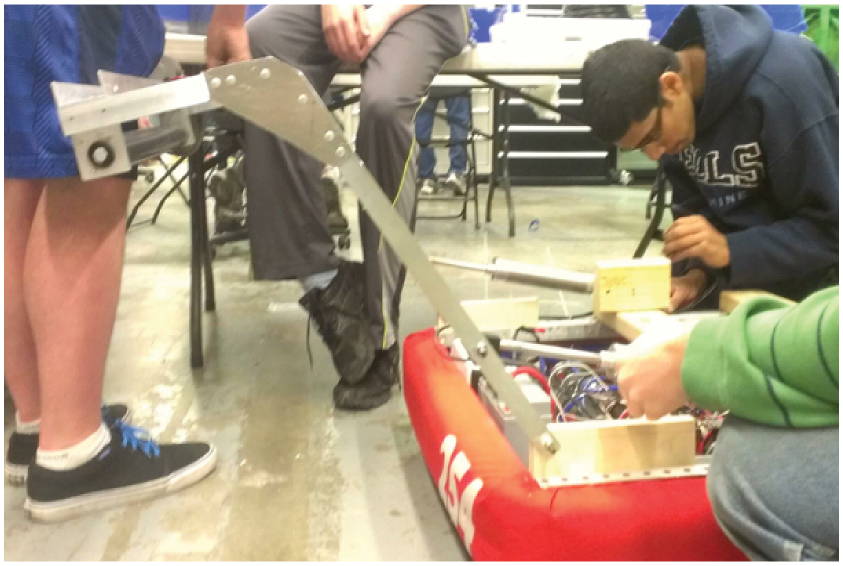
Flywheel

Built with different flywheel diameters and varying amounts of compression levels

- Best grip and size was 4″ neoprene rollers, given limited allocated volume
- Added extra weight to the roller to increase the moment of inertia

Pros	**Cons**
• Most consistent shot (With compression, was still accurate for both over and under inflated balls)	• Packaging
• High trajectory, can shoot over other robots	• 28″wide chassis with 2″ wide tubes on both sides give the ball (24″ diameter) only 24″ of room, making it hard for the ball to fit through.
• Fastest reset time between shots (Only about 1 second if flywheel is kept running)	• Requires more manufacturing precision to ensure consistent compression.
• Load from the bottom independent of intakes	
• Easiest to manipulate and load 3 balls during autonomous.	

Thus, the flywheel proved to be the best shooter for accomplishing our primary goal of quickly shooting 3 balls in autonomous mode.



Dual Intake

After selecting the flywheel shooter, the next step was determining how to manipulate 3 balls during autonomous mode. This led to the dual intake design.

Pros	Cons
• Can hold balls while driving	• More parts to manufacture, assemble, and possibly break
• Quicker to pickup balls in parallel rather than in series	
• Useful for driving and assists	

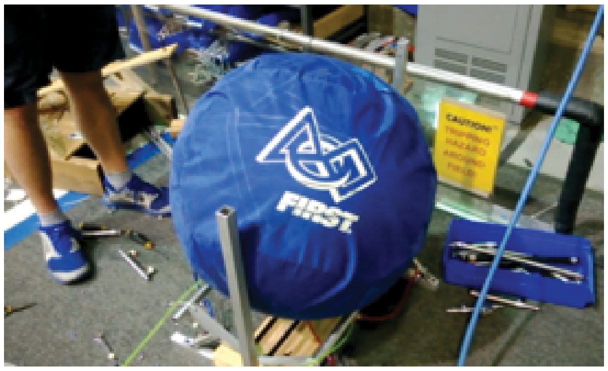
After selecting the flywheel shooter, the next step was determining how to manipulate 3 balls during autonomous mode. This led to the dual intake design.

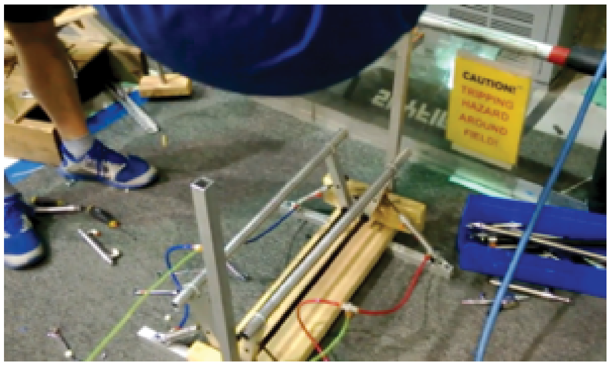
Intake Shape:

- Initially had 2 straight (flat) intakes
- Changed front intake to a bent angle that raises the intake to bring the ball into the robot
- Rear intake had to stay flat for packaging reason (to lie flush with the flywheel hood structure)

Ball is first pinned between roller and bumper:

- Raising the intake brings the ball into the robot
- Allows driver to navigate with the ball





Loader

Required to feed ball from stationary into flywheel

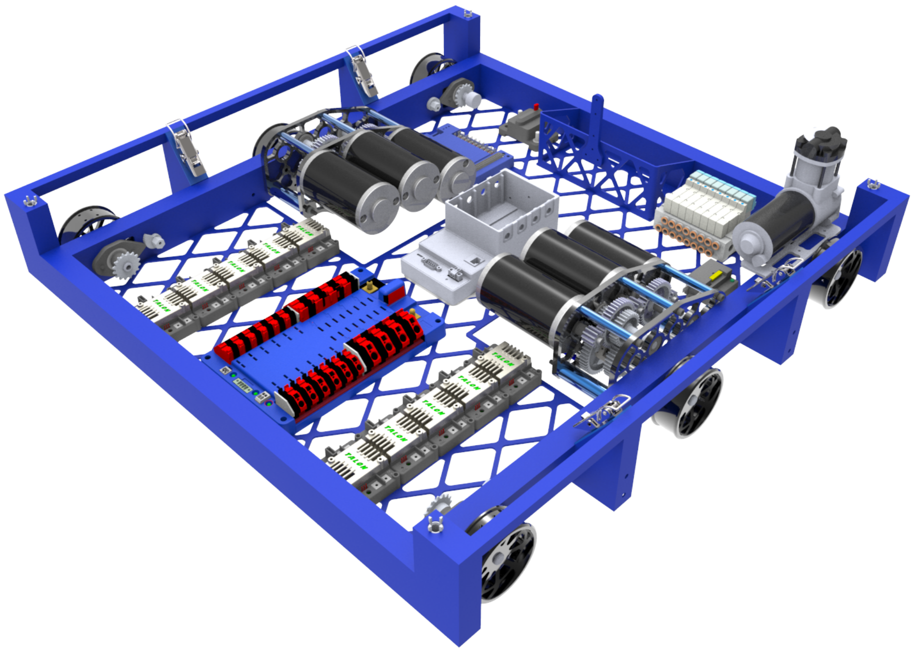
Loader Type

- Tried a linear elevator and a hinged pop-up mechanism to lift the ball
- Elevator did not settle the ball, ball shifted during launch, reducing accuracy.
- Reset time was longer, about 1.5 seconds

The hinged mechanism, nicknamed “Pinniped” after a seal, was superior in every way

- Automatically settled the ball, which meant shooting was more consistent
- By actuating only one side, balls can be pushed out the front or back for assists or scoring in the low goals.
- Faster to reset, helping with 3 ball auto (approximately 0.2 seconds cycle time)

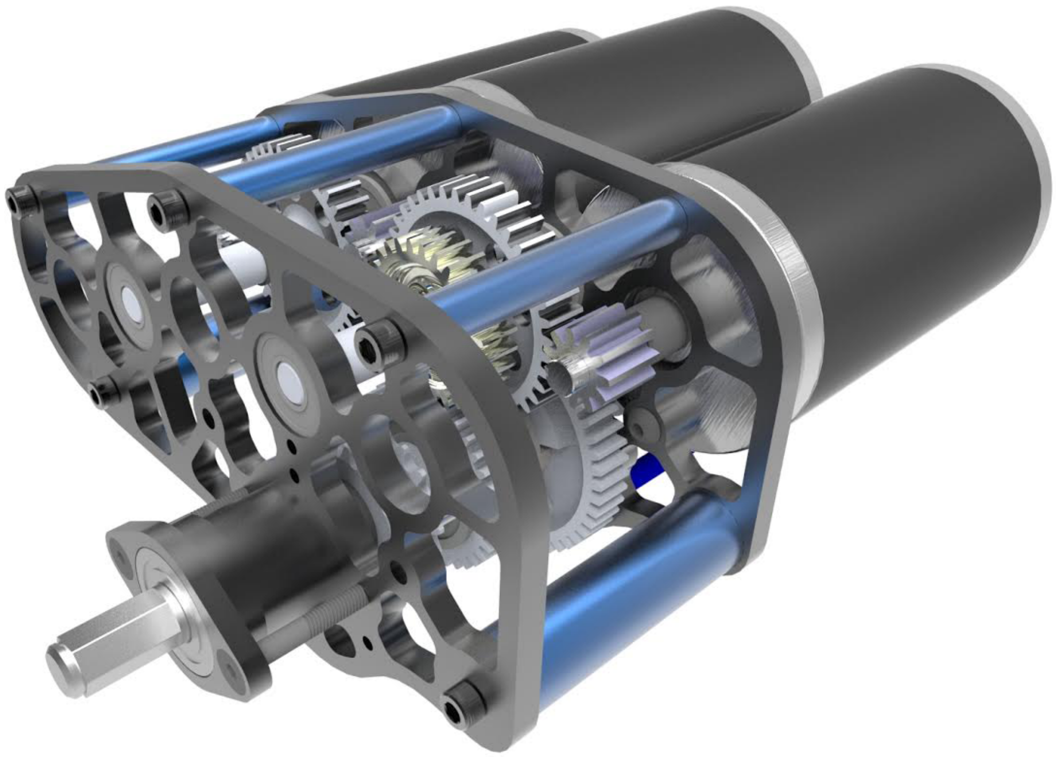
Final Mechanical Design and Technical Specifications



Drivebase

Used a similar drivebase to what we have been using for several years

- 6 Wheel, “West Coast Drive,” with a center drop of 5/32”
- Familiarity and ease of manufacturing allows us to focus on other subsystems
- Symetrical square chassis maximizes internal space to allow the symetrical ball room to fit within the sturcture of the robot
- Various components were optmized for strength and weight versus previous year’s designs
- Structure is 2” x 1” x 1/8” and 1/16” wall tubing



Drive Gearbox

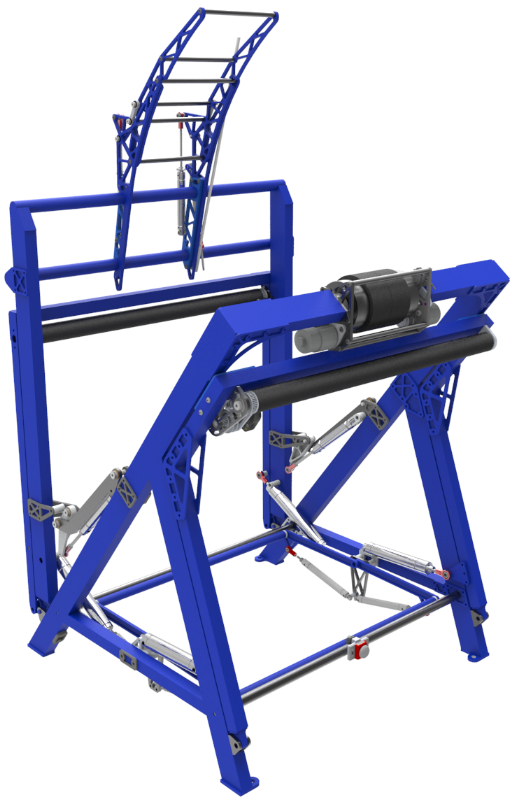
6 CIM Motors, can shift between 2 speeds with a 2-position pancake pneumatic cylinder

High Speed

- 4.166/1 Overall Reduction
- 19.6 ft/s
- 1284 RPM at output shaft

Low Speed

- 10.66/1 Overall Reduction
- 7.66 ft/s
- 502 RPM at output shaft



Superstructure

Flywheel design and location drove the superstructure shape

- Allows room for a ball to come in and not touch the flywheel until ready to shoot while staying within volume requirements

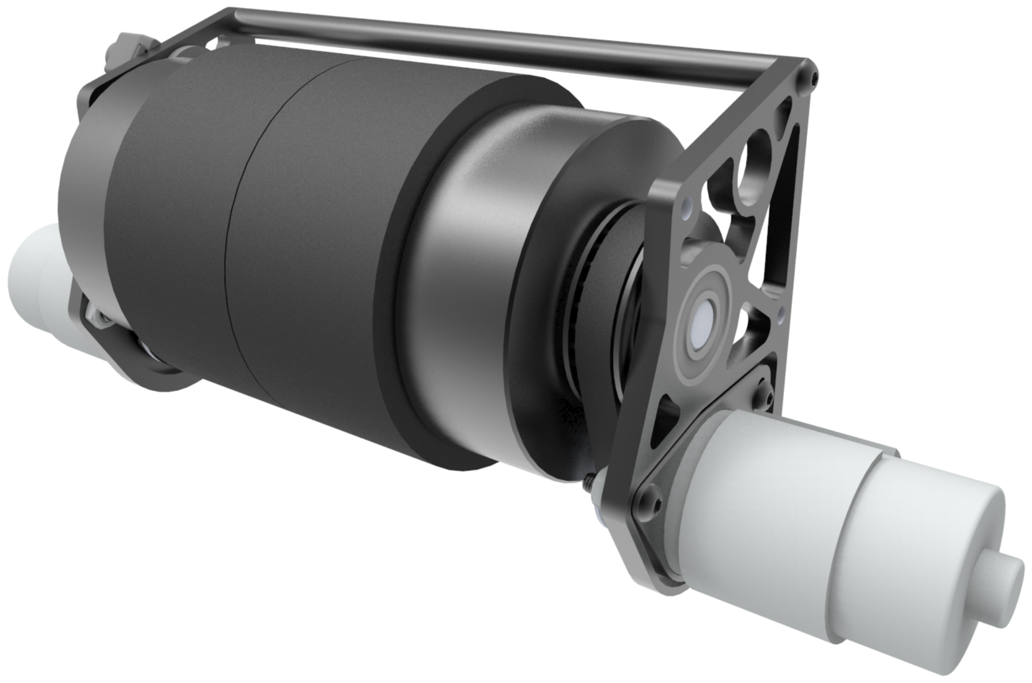
Triangular base directly supports flywheel and pinniped

- Also includes pivot points for front and rear intakes

- Modular design allows for complete removal of superstructure with all components attached

While not a priority, a pneumatic cylinder was included to open up the back wall of the shooter/hood for catching

2″ x 1″ and 1″ x 1″ 1/16″ wall tubing



Flywheel

Powered by two Banebots 775 motors with 5mm HTD timing belt

- Belts are quieter, simpler, and lighter than gears

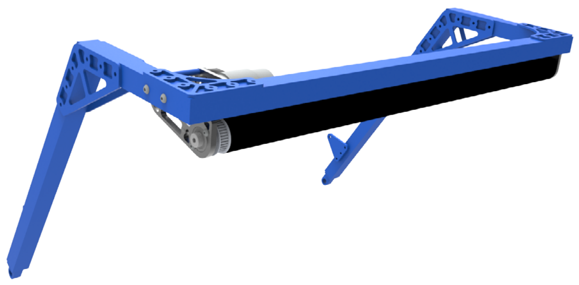
Shooter wheels are two 4′′ diameter, 2″ wide Neoprene wheels

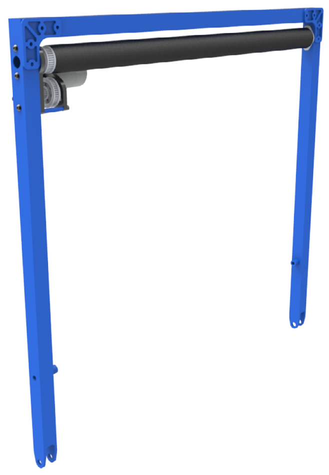
3.5′′ diameter Steel wheels were added to increase the system's moment of inertia. They were lightened only in the middle to reduce weight and have a minimal impact on the MOI.

Uses a Retroreflective Sensor that detects retroreflective tape to calculate RPM

- Allows us to specify different RPMs for different shots

The Banebots motor is mounted on sliding plate that allows us to have the option to quickly adjust the belt tension.





Intakes

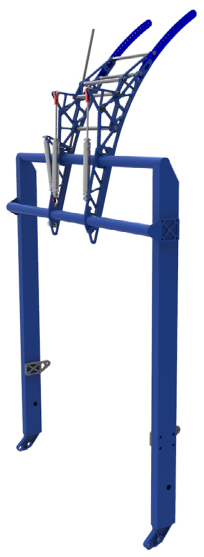
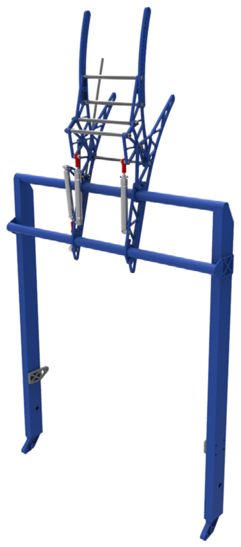
Front and rear intake have the same roller and gearbox

Pneumatic Cylinders

- A pair of cylinders for the front intake are mounted to superstructure
- A pair of cylinders for the rear intake are mounted to hood support structure

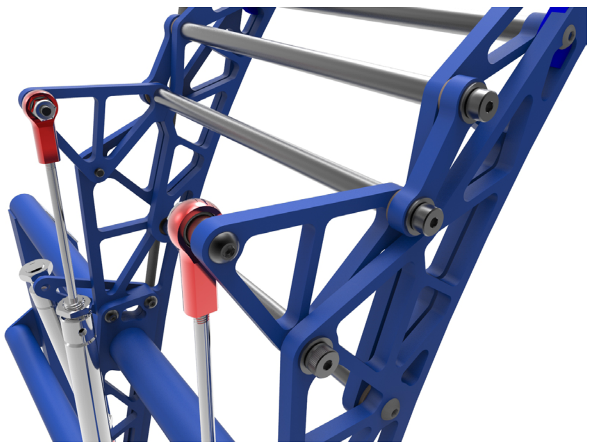
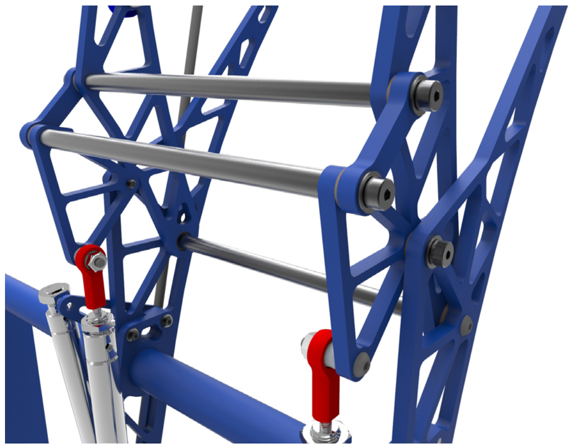
Technical Specifications

- Powered by one Banebots 775 motor with timing belts, the 775 was chosen over the 550 due to its larger thermal mass, which can more easily disipate heat when the intake roller is stalled and carrying a ball
- 7.5:1 reduction was optimized such that surface speed of the roller was slightly faster than the top drive speed so that a ball can still be acquired while driving away from it.

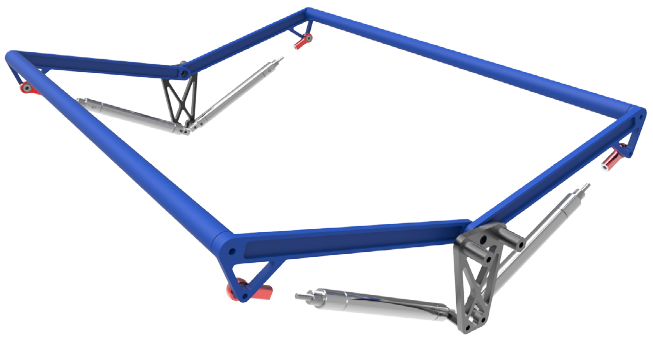



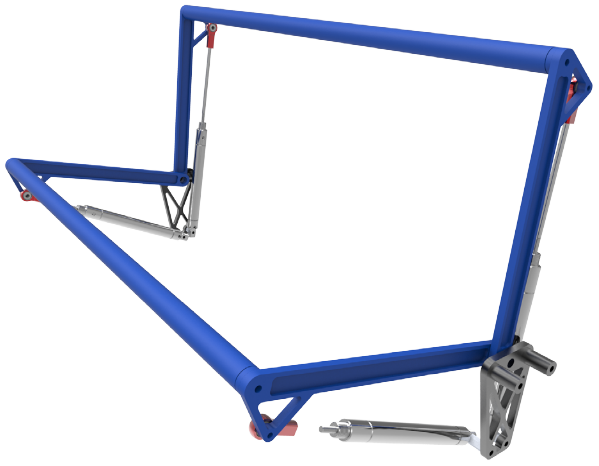
Adjustable Hood

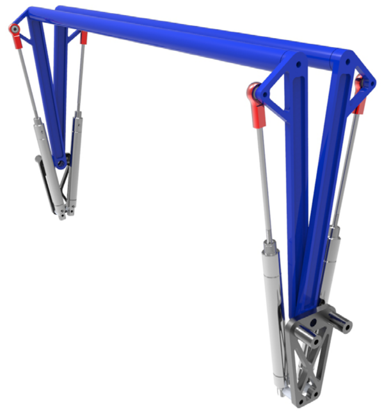
The hood uses an over-center linkage powered by a pair of cylinders to actuate between a steep and shallow angle to allow it to lock in the shallow angle. Locking the angle ensures the ball will not force the hood open and create an inaccurate shot.

The steep angle is used while close to the goals and truss shots. The shallow angle is used other shots from farther away or while moving.





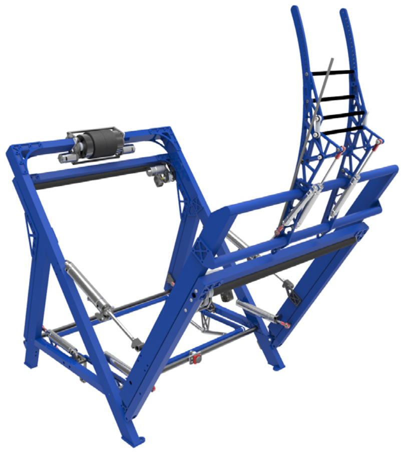


Pinniped

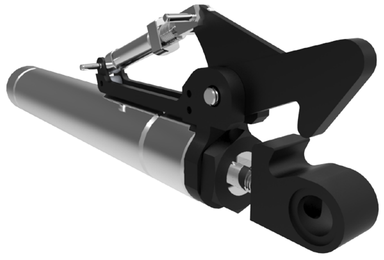
Controlled by two pairs of pneumatic cylinders

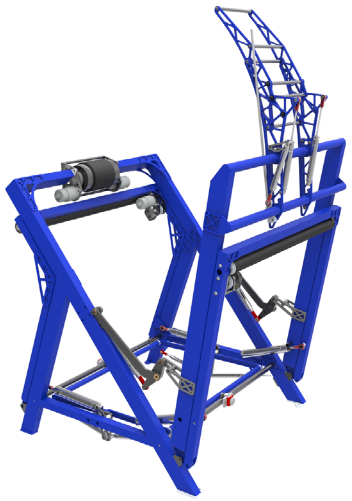
- Front and back sides individually controlled
- One side can be actuated at a time to push a ball out through the front or rear of the robot, allowing the robot to easily pass the ball for assists or score in the low goal.

Front side goes a bit faster than back one to push ball towards back of hood

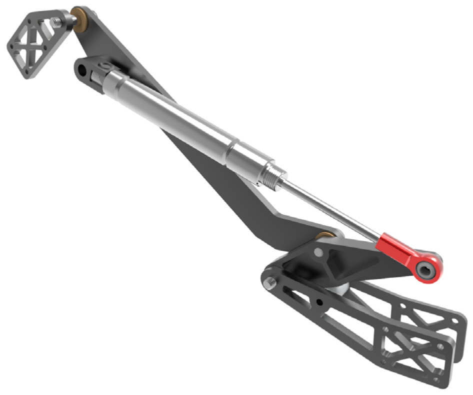


Pre-SVR





Post-SVR



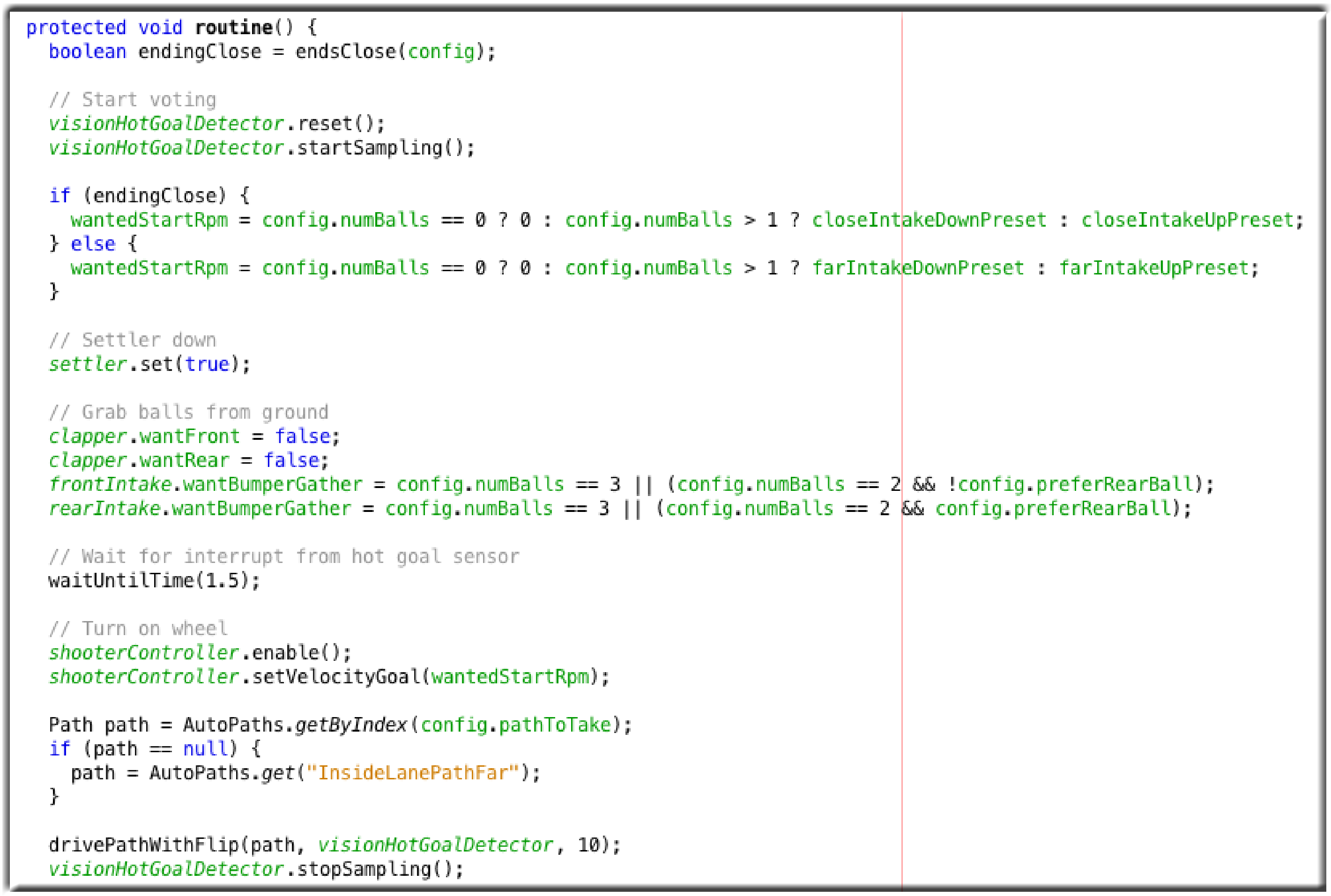
Catcher

A pair of pneumatic cylinders push open the hood support structure, allowing a ball to easily fit between the flywheel and hood for inbounding from the human player through the top of the robot.

Old designs featured a locking cylinder mechanism (top right) that opened the hood structure to create an area for catching (top left). However, the catcher was not found to be useful during real gameplay and was never used.

For this reason, we iterated the old design to utilize only one piston on each side tha toggled an overcenter latch (bottom right) to open the hood slightly (bottom left). This uses less air than the locking pistons and is going to be more useful during matches.

Control Software and Autonomous Strategy



```java
protected void routine() {
  boolean endingClose = endsClose(config);

  // Start voting
  visionHotGoalDetector.reset();
  visionHotGoalDetector.startSampling();

  if (endingClose) {
    wantedStartRpm = config.numBalls == 0 ? 0 : config.numBalls > 1 ? closeIntakeDownPreset : closeIntakeUpPreset;
  } else {
    wantedStartRpm = config.numBalls == 0 ? 0 : config.numBalls > 1 ? farIntakeDownPreset : farIntakeUpPreset;
  }

  // Settler down
  settler.set(true);

  // Grab balls from ground
  clapper.wantFront = false;
  clapper.wantRear = false;
  frontIntake.wantBumperGather = config.numBalls == 3 || (config.numBalls == 2 && !config.preferRearBall);
  rearIntake.wantBumperGather = config.numBalls == 3 || (config.numBalls == 2 && config.preferRearBall);

  // Wait for interrupt from hot goal sensor
  waitUntilTime(1.5);

  // Turn on wheel
  shooterController.enable();
  shooterController.setVelocityGoal(wantedStartRpm);

  Path path = AutoPaths.getByIndex(config.pathToTake);
  if (path == null) {
    path = AutoPaths.get("InsideLanePathFar");
  }

  drivePathWithFlip(path, visionHotGoalDetector, 10);
  visionHotGoalDetector.stopSampling();
```

Prototyping

We bootstrapped last year's practice robot (without superstructure) to start writing new drive code immediately.

Our team was able to prototype the intake and shooter on top of an old practice robot.

Field Navigation Controller

Due to the fact that a goalie robot could block our shot, we wanted to have autonomous modes that could shoot all three balls from a variety of spots on the field.

These paths needed to be easy to implement and read so we could iterate quickly. To accomplish this, we built a system that takes a series of waypoints in the terms (X, Y, Heading) and generates smooth paths for the robot to drive.

We wanted the robot to have the ability to make smooth arcing turns in autonomous mode, so we used a spline interpolation algorithm to build the paths using the given waypoints.

Each path generates a list of expected position, velocity, acceleration, and overall robot heading values for each wheel a 10ms interval. When we want the robot to drive the path, we "press play" and the robot takes off.

Autonomous

The driver inputs some parameters for what the autonomous mode should do: how many balls to start with, which path to drive, how far from wall to end, etc.

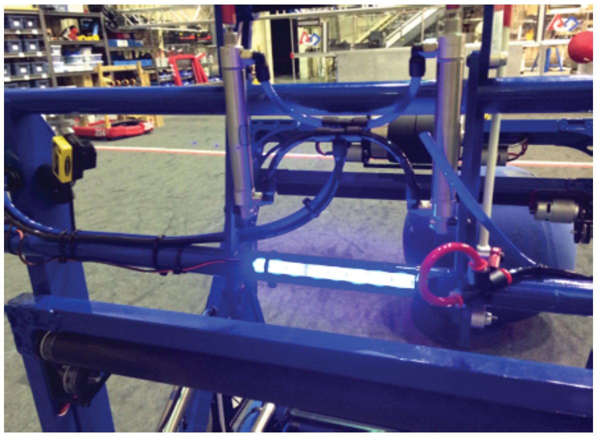
- From the parameters we load the correct path and turn on 2, 11, or 0 intakes.
- Our autonomous starts by determining which goal is hot using reflective sensors from the Banner.
- The robot then drives the path input by the operator.
- The robot shoots all three balls.

Driver Control software

Using a Vex bump sensor, we created a set of buttons to automatically intake the ball onto the pinniped.

To control the flywheel speed, we first calculate the current velocity of the wheel by measuring the time between rising edges of a signal provideed by a retroreflective sensor aimed at the wheel. We use this velocity measurement to feed a control loop which adaptively gives more or less power to the wheel to spin it to our goal RPM.

LED strips on the back of the hood indicate when flywheel reaches optimal speed, allow ing the driver to accurately perform a running shot.



Compiling

Our coding team developed a program to fine tune robot constants without having to recompile and deploy new code each time, allowing for rapid iteration during testing.

They also created a separate library for computing driving trajectories for autonomous that would generate a Java file for the robot to use, allowing us to test the trajectory code without putting it on the robot.

254
254

254

Bellarmine College Preparatory
254

254
254

Bellarmine College Preparatory