Engineering Notebook

9447B

Team Number

Wild Cards

Team Name

Palmetto Scholars Academy

School

05/05/2022 Start Date

00/00/0000 End Date

1 of 1 Book #

v1.0.8.29.22

Table of Contents

TEAM BIOGRAPHY

Aidan Frye, Mohamed Seck, Duaa Jamaluddin, Timothy Gladden, Elliott McAninch (sitting)

*Left to right

TEAM BIOGRAPHY PT.2

Aidan Frye is the team's programmer and is currently a senior at Palmetto Scholars Academy. This is his fourth year in VEX.

Mohamed Seck is the team's scouter and online challenge manager and is a senior at Palmetto Scholars Academy. It is his second year in VEX.

Duaa Jamaluddin is the team's notebook writer and is currently a senior at Palmetto Scholars Academy. This is her fourth year in VEX.

Timothy Gladden is the team's prototype design, second builder, second notebook writer, and is a senior at Palmetto Scholars Academy. It is his fourth year in VEX.

Elliott McAninch is the team's main builder and driver and is a junior at Palmetto Scholars Academy. It is his fourth year in VEX.

GAME DESCRIPTION

SPIN UP

"Played on a 12' x 12' field with two Alliances - one red and one blue - composed of two teams each, compete in matches consisting of 15 second autonomous period, followed by a one minute and forty-fire second Driver Controlled Period."

- ***REC FOUNDATION***

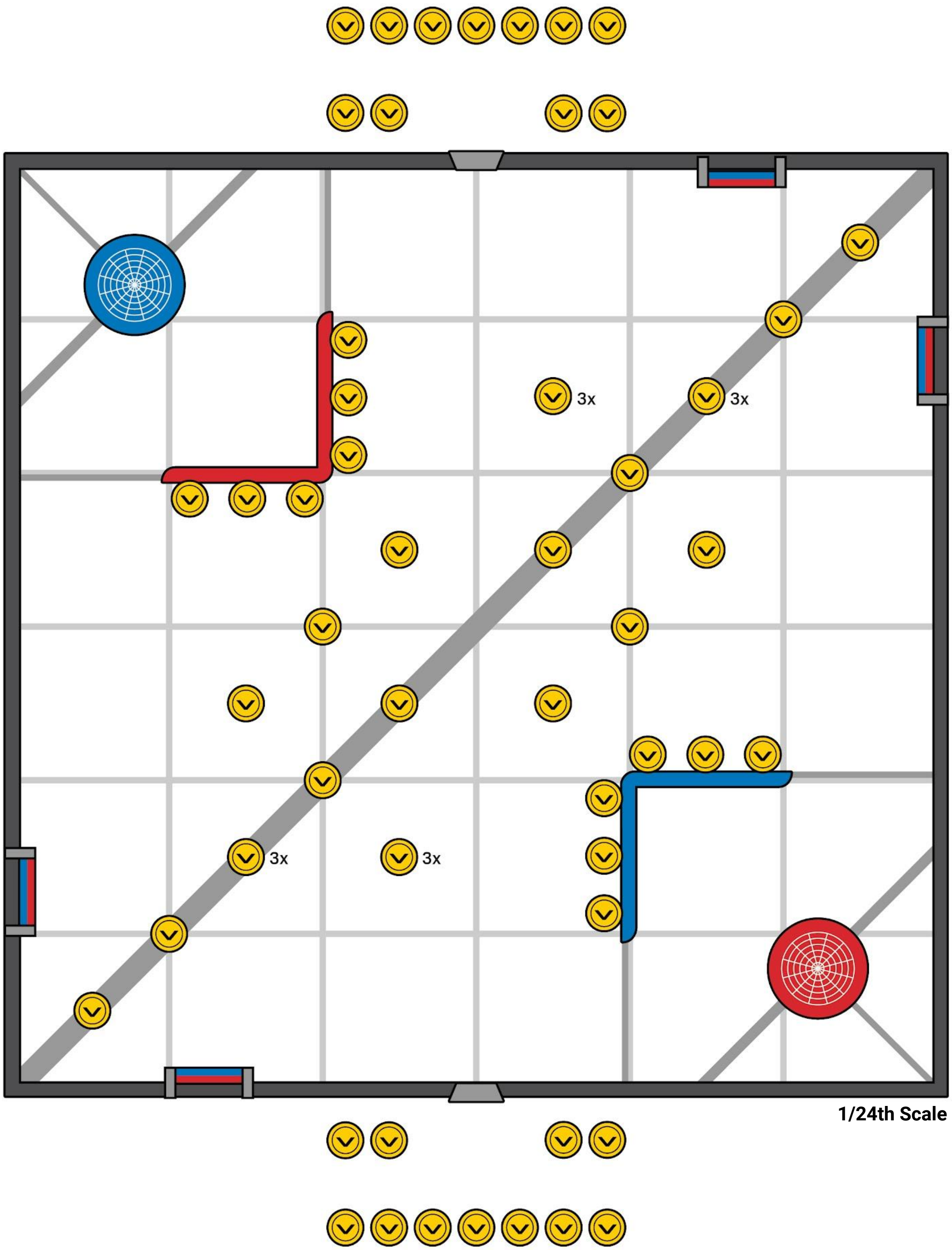
" There are sixty Discs and four rollers on a VCR Spin Up. Discs can be scored in the two High Goals, one per Alliance, at opposite corners of the field. "

- ***REC FOUNDATION***

SCORING:

Each disc scored in a high goal	5 POINTS
Each disc scored in a low goal	1 POINT
Each Owned Roller	10 POINTS
Each Covered Field Tile	3 POINTS
Winner of Autonomous Bonus	10 POINTS
Two Discs Scored in the Alliance - Colored Goal And two Rollers Scored in Autonomous	1 WIN POINT

GAME DESIGN: ***SPIN UP***



Project Game Field

Name Timothy Date 05/05/22 Page 4

- Field Introduction

<u>Dimensions:</u> 12’ x 12’ Square Field

60 Discs → 8 Pre-loads, 4 Per Alliance

14 Match Discs, 7 Per Alliance

Field is used for head to head alliance-based matches

The given dimensions and disc amount allow for alliances red & blue to go head to head and see which alliance can score within the 15 seconds auton period, AND within the minute and 45 second driver - controlled period.

- **Points are scored with discs, covering rollers, and covering field tiles.**
- **Auton win gained if and only if an Alliance owns two rollers and has scored twice in the high goals**
    - **Auton bonus given to whichever alliance has the most points after Auton - ends.**
- **Skills matches can be completed before, in-between and after matches to gain standing in a skills bracket and to test our robot.**

GAME RULES

Each Disc scored in a High Goals is worth ***5 points,*** However, Robots aiming for High Goal need to be accurate because underneath each high Goal is a ***1- point*** Low Goal for the Opposing Alliance

In Addition to discs, Robots can also spin the four rollers mounted to the field perimeter. If the area Inside of a roller's pointers only show one color, that is considered "Owned" by the Alliance with the corresponding color. Each owned roller is worth ***10 Points***

As the clock winds down, it's time for the Endgame. At the end of the match, Alliances will receive a **3 - Point** bonus for each tile covered by their robots . So, during the last 10 seconds of the match, there are no horizontal expansion limits

The Alliance that scores more points in the Autonomous period is awarded with ten bonus points, added to the final score at the end of the match. Each Alliance also has the opportunity to earn an Autonomous win point by scoring at least two discs in Alliance High Goals, and owning Both Rollers on their side of the field. This Bonus can be earned by both Alliances , regardless of who wins the Autonomous Bonus.

Summer Journals

Section 1

Thursday May 5th 2022

VEX Spin Up just got revealed, and our team is very excited for this game. The premise is there are many discs placed across the field, and two frisbee golf-like goals in the corners in which to score them. One part of the game we think is very interesting is each team has a low goal placed underneath the opposing team's high goal. If you miss the high goal, you score in the opponent's low goal. There are also rollers around the field, but the discs are the main way to score points in this game. We quickly identified 3 different designs based on our past experiences with robotics. One design is a catapult-type mechanism that will fling the discs into the high goal. Another is a flywheel shooter mechanism. The last is a manual conveyor belt that delivers the discs into the high goal.

Monday May 9th 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

The game manual was released this past Saturday during the HS worlds broadcast. Looking through the manual, there are some interesting rules, and the manual allowed us to better compare the viability of our designs. We identified the two most important rules to this game. In our opinion, these rules are the 3 discs possession limit, and the 2 inch circle limit on vertical expansion. A decision matrix comparing our designs based on these rules are below. The criteria we decided on was simplicity of design, efficiency of design, precision of disc travel to high goal and durability of design.

Monday May 9th 2022 (Pt.2)

Design Decision Matrix	Simplicity	Efficiency	Precision	Durability	Total
Catapult	7	5	3	4	19
Flywheel	3	10	6	7	26
Conveyer Belt	8	3	8	6	25

The flywheel design and the conveyor belt design are extremely close in total score based on our decision matrix, but in the end we decided to go with the flywheel due to the sheer efficiency it provides. With 38 discs on the field, efficiency will be incredibly important if we want to score high. After closer inspection, another major mark against the flywheel is that in order to reach the high goal, we would have to vertically expand, which would make it almost impossible to deliver the discs due to the 2 inch circular expansion limit.

Wednesday May 11th 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

After more discussion and brainstorming ideas for the flywheel mechanism, we feel confident in our decision due to the fact that the flywheel has the most room for improvements and small design and programming choices to give us an edge. Our programmer is getting extremely excited with the possibilities we have been brainstorming. Our biggest idea that we think will give us an extreme edge is automatic tracking of the high goal. If we make our flywheel mounted on an almost turntable-like mount, like a turret, then we can put a vision sensor on it. This will allow us to tell the vision sensor to keep the high goal in the middle of its camera. This will skyrocket our efficiency and precision, as we don’t have to line up the robot every time we want to take a shot. If we make it fast enough as well, we think we will be relatively immune to defense in the form of ramming, because the vision sensor will adjust for the sudden movement.

Wednesday May 11th 2022 (Pt.2)

High speed ramming tends to be the best form of defense in a shooting game like this, and if we become resistant to that, we think we can be an incredibly good team. Our programmer created flywheel auto targeting pseudocode, which is below. He says that the main issue in programming the function will be getting the finicky vision sensor to reliably recognize the high goal.

```
FOREVER

        IF high goal centered

                MOVE_TURRET nowhere

        ELSE IF high goal on right side of vision sensor
camera

                MOVE_TURRET right

        ELSE IF high goal on left side of vision sensor camera

                MOVE_TURRET left

        ELSE IF high goal not sensed by camera

                MOVE_TURRET to angle 0 (straight forward)
```

Friday May 13th 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

We have gotten started with our drivebase. We chose to start with an X-drive base because we think accurate movement will be incredibly important for this game. We have the basic shell of the X-drive, we need to work on adding motors and stabilizing the drive.

Monday May 16th 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

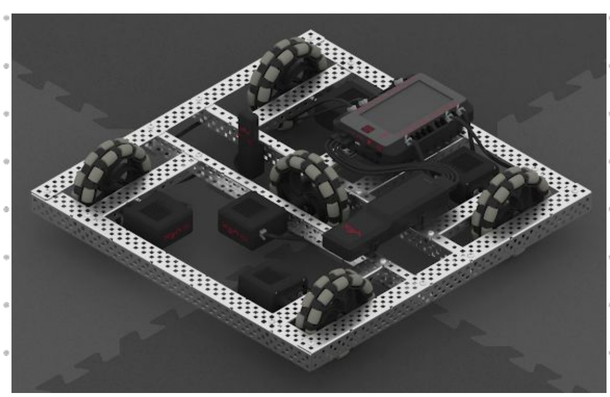
We have run into a roadblock with the X-drive base. One of the angels got messed up, and our builder does not know how to fix it. It took a while to perform an inventory of parts and find the ones needed for building the X-drive base. Our builder says he is going to try and fi it for a few more days, but because new parts may be needed, and we do not know if they are available,our drive base might be switched to a standard tank drive base due to time constraints.

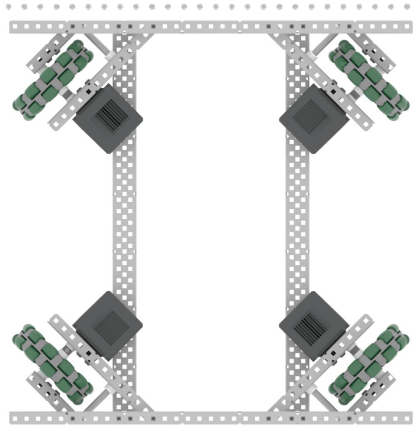
Wednesday May 18th 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

Our team has decided to make the jump from an X-drive base to a tank drive base. The maneuverability of the X-drive base is a major boost to robot performance, but we hope that because of our turret's auto-targeting, higher maneuverability isn't necessarily needed. In our opinion, drive speed is not going to be the limiting factor in overall speed for this game, it will be shooting speed, thanks to the 3 disc possession rule. Odds are, the robot will be able to intake discs faster than you can shoot them, so maximizing your shooting speed increases the overall speed of the robot. Below are CAD designs of our plans for this robot's drive base

Tank Drive Base





X-Drive Base

Friday May 20th 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

We have the drive base done, as the design is much simpler than an X-drive. We used many more screw joints this year than we did last year, due to them being much easier to work with and the fact that they give added stability to the joint. These screw joints also allowed us to implement free spinning tracking wheels to the robot in a unique and space-efficient way. Our programmer is going to have his work cut out for him this year with programming a PID/Odom movement system as well as auto-targeting with the vision sensor. The tracking wheels in the middle of the robot are also really nice because they will be locked omni wheels, doubling as ramming resistance. By locking the omni wheels, the smaller free spinning wheels around the big wheel no longer spin, which takes the strafing capability away from the wheel. This will make us resistant to ramming, and the people who ram us will not push us very far at all. This compared with the auto targeting flywheel should make us pushing and ramming defense all but useless against us. At least, that’s the hope we have for this design.

Sunday May 22nd 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

Our team decided to have a Sunday practice, but due to one of our members having covid, and two others not being able to come, it was more of a building session than a team meeting. In this building session we added polycarbonate backing for the discs to slide up on, made an intake and added the first chain links as well as added more stability to the design through supports.

Project Polycarbonate

Name Timothy

Date 05/22/22

Monday May 23rd 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

Today our team worked on connecting the intake mechanism through its entirety and created out flywheel. We found an online Desmos graphing resource that aided us in deciding on output angle for the flywheel, as well as the flywheel gear ratio and rpm. The desmos graph was a basic kinematic equation that calculated the rpm and angle needed to shoot a disc and well within the vertical bounds of the high goal. Using this resource, our team decided to mount the flywheel at a 20 degree angle, and we decided to use a 600 rpm motor geared 1:3. This allows us to have a maximum rpm of 1800, which is more than enough speed to be able to shoot from the match loader on the side of the field. We can then reduce the rpm of the flywheel based on the distance to the high goal, and this will be included in our auto tracking turret code. We also ensured that this angle and rpm was able to shoot from our opponent's low goal into our own low goal to further deny points to our opponent in the later stages of the match (0:30 - 0:15).

September - January
Section 2

September 2022

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

= Practice Days

Week 1 Practices

8/28/2022 - 9/3/2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

- Today was the first official meeting
- The focus of today was to establish a team practice schedule and create diagrams for future robot plans

Today was the first official day of robotics, while our team has been working for some summer, we finally have access to school materials and can start regularly scheduled practices. Our school is offering practices from Monday - Thursday from 3:30-5:30. However, our builder and programmer can only make it either Monday or Thursday, depending on the week and their job schedules.

Elliott started brainstorming ideas for a robot design on the whiteboard. Since we have most of the robot planned the sketches help to visualize the end product.

Project Whiteboard brainstorming
Name Timothy Date 08/29/22

Week 2 Practices 9/5/2022 - 9/10/2022

September 8, 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

Recap:

Started assembling the field today, however there was a five drill in the middle so practice was cancelled.

Week 3 Practices 9/12/22 - 9/17/22

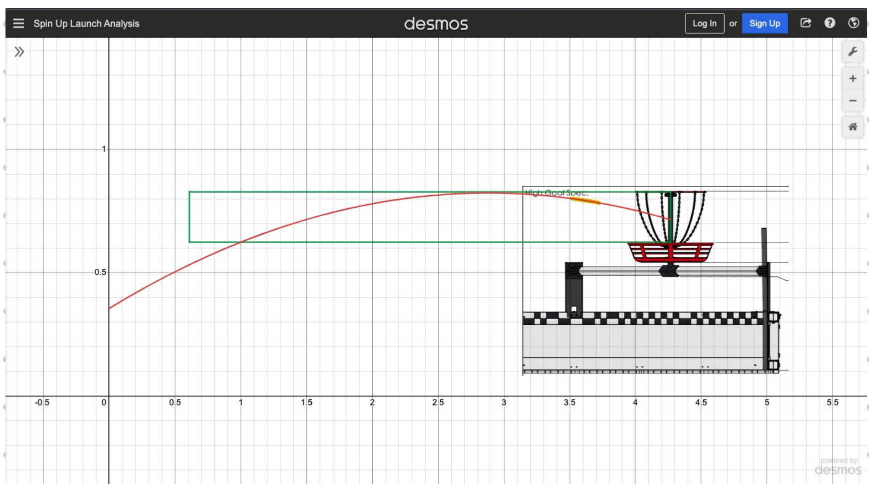
September 12, 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

The simple graph to view the kinematic model for a disc shot towards a high goal that we mentioned during our summer practices, we have used it again to work on the angle of the flywheel. This is a more explained and elaborated version of the previous use of it. The green box represents the zone at which the disc will hit the high goal. This graph has position on both axis. The Y axis is the vertical position and the X axis is the horizontal position. The upper and lower horizontal lines of the green box are placed at the highest and lowest height a disc can reach the high goal respectively. To match the specified tolerances, I placed these values at 26 inches and 24.5 inches. Drag forces are not considered. Note: All graphed dimensions are in meters. Please note the units in the descriptions of each variable below. The disc picture is set up to track the position graph and rotate according to its instantaneous slope if you were to animate the variable "d" This feature is purely cosmetic. I just wanted to animate the disc flying. The image for the disc can be disabled if you want to hide it.

Configurable Variables:
h: The initial launch height of the disc. (Inches)
v: The initial velocity of the disc (Meters per Second)
a: The launch angle of the disc above the horizontal.(Radians)
Max: The farthest distance from the goal that you want your robot to be able to shoot discs from. I placed this value at 14 feet to be able to shoot discs from the match load ramps. (Units: Inches)
Min: The closest distance from the goal that you want to be able to shoot discs from. Very similar to the Max parameter. (Units: Inches)

While Elliott was working on the angle of the flywheel to try to have it attached by next week. Timothy was working on a prototype for the string mechanism.



The string mechanism is designed to shoot out string from a spool at the end of the match (10-15 seconds) to gain extra points.

Timothy started off making this by connecting two beams with a pivot point. The purpose of the pivot point was to unfold to get as much surface area on the mat as possible. Our goal is to have one of each side of the robot. Since we have one month till out first competition on November 5th, we are trying different scoring methods to maximise out robot without out drive having to spend weeks on creating a driving plan.

Week 4 Practices:

September 22nd,, 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

Recap:

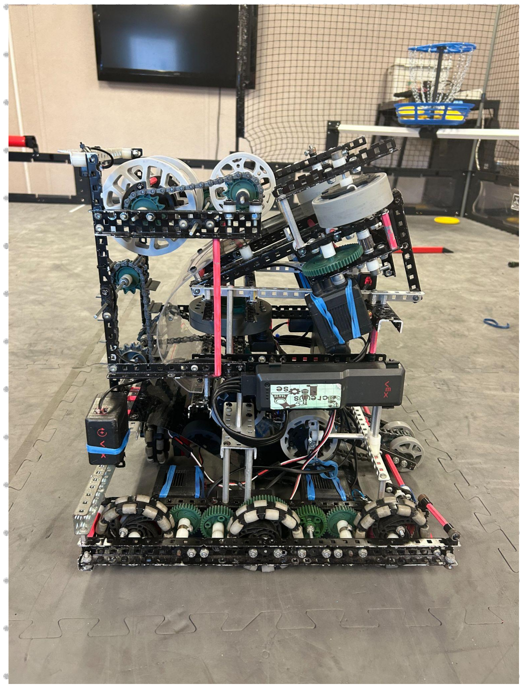
- Attach the turret (flywheel) today
- Finish the driving code

The desmos used to find the angle of the flywheel helped us determine the ideal angle to be 30 degrees.

Using that measurement, our builder, Elliott assembled the flywheel using geometry to create a right triangle.

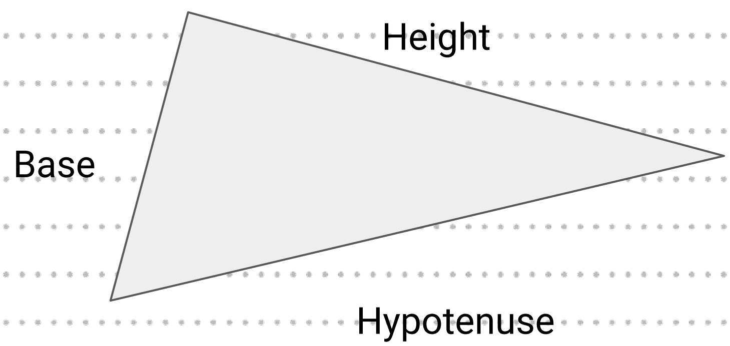
- Hypotenuse are the c-channels
- The height and the base are one-by-one L channels
- The flywheel is attached by a planetary gear that acts as a turntable
- Stand offs are used to connect to the robot.

Week 4 Practices: PT 2

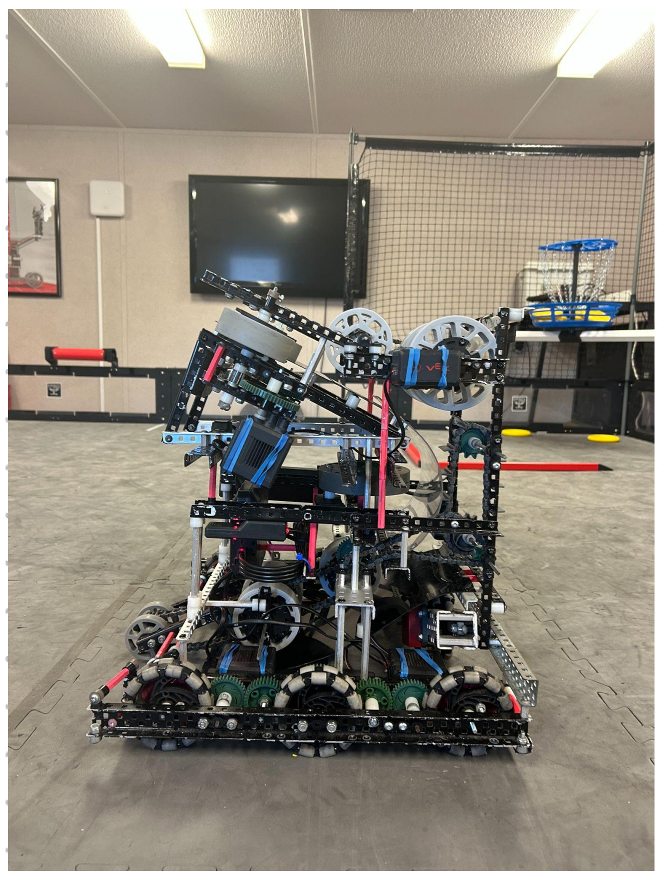


This picture of the flywheel is of after it was attached to the robot.

The c- channels on the bottom are the hypotenuse.



The base and the height are both made from L-channels. The flywheel is connected by standoff and has a planetary gear attached to it.



At the end of practice, this is what our robot looked like.

During the process of Elliott attaching the flywheel. Aidan worked on the Drive code that will be finished by next practice.

Week 5 Practices:

September 26nd, 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

Recap:

- Finish code
  - Driving code

```
void usercontrol(void) {
  Flywheels.stop(coast);
  Intakes.stop(coast);
  // User control code here, inside the loop
  while (1) {
    LeftDrive.spin(fwd, Controller_Ruh.Axis3.position() * 4.8, rpm);  //*4.8
    RightDrive.spin(fwd, Controller_Ruh.Axis2.position() * 4.8, rpm); //*4.8
```

```
  if (Controller_Ruh.ButtonA.pressing()) {
    waitUntil(not(Controller_Ruh.ButtonA.pressing()));
    while (flywheelMode == 0) {
      if (flywheelSpeed == 0) {
        LeftFlywheel.spin(fwd, 119, rpm);
        RightFlywheel.spin(fwd, 119, rpm);
        flywheelDesiredSpeed = 119;
        flywheelControl();
      } else if (flywheelSpeed == 1) {
        LeftFlywheel.spin(fwd, 150, rpm);
        RightFlywheel.spin(fwd, 150, rpm);
        flywheelDesiredSpeed = 150;
        flywheelControl();
      } else if (flywheelSpeed == 2) {
        LeftFlywheel.spin(fwd, 200, rpm);
        RightFlywheel.spin(fwd, 200, rpm);
        flywheelDesiredSpeed = 200;
        flywheelControl();
      }
 if (Controller_Ruh.ButtonA.pressing()) {
        waitUntil(not(Controller_Ruh.ButtonA.pressing()));
        flywheelMode = 1;
      }
    }
      if (flywheelMode == 1) {
      LeftFlywheel.stop(coast);
      RightFlywheel.stop(coast);
      flywheelMode = 0;
      flywheelDesiredSpeed = 0;
    }
  }

  if (Controller_Ruh.ButtonR1.pressing()) {
    TopIntake.spin(fwd, 200, rpm);
  } else if (Controller_Ruh.ButtonR2.pressing()) {
    TopIntake.spin(fwd, -200, rpm);
  }
```

```
if (Controller_Ruh.ButtonL1.pressing()) {
    Intake.spin(fwd, -600, rpm);
  } else if (Controller_Ruh.ButtonL2.pressing()) {
    Intake.spin(fwd, 600, rpm);
  }

  if (Controller_Ruh.ButtonUp.pressing()) {
    TopIntake.spin(fwd, 200, rpm);
    Intake.spin(fwd, 600, rpm);
  } else if (Controller_Ruh.ButtonDown.pressing()) {
    TopIntake.spin(fwd, -200, rpm);
    Intake.spin(fwd, -600, rpm);
  }

  if (Controller_Ruh.ButtonX.pressing()) {
    flywheelSpeed++;
    waitUntil(not(Controller_Ruh.ButtonX.pressing()));
    if (flywheelSpeed > 2) {
      flywheelSpeed = 2;
    }
    if (flywheelSpeed == 0) {
      Brain.Screen.setFillColor(red);
      Brain.Screen.drawRectangle(0, 0, 500, 500);
    } else if (flywheelSpeed == 1) {
      Brain.Screen.setFillColor(green);
      Brain.Screen.drawRectangle(0, 0, 500, 500);
    } else if (flywheelSpeed == 2) {
      Brain.Screen.setFillColor(blue);
      Brain.Screen.drawRectangle(0, 0, 500, 500);
    }
} else if (Controller_Ruh.ButtonB.pressing()) {
    flywheelSpeed--;
    waitUntil(not(Controller_Ruh.ButtonB.pressing()));
    if (flywheelSpeed < 0) {
      flywheelSpeed = 0;
    }
```

```
if (flywheelSpeed == 0) {
        Brain.Screen.setFillColor(red);
        Brain.Screen.drawRectangle(0, 0, 500, 500);
      } else if (flywheelSpeed == 1) {
        Brain.Screen.setFillColor(green);
        Brain.Screen.drawRectangle(0, 0, 500, 500);
      } else if (flywheelSpeed == 2) {
        Brain.Screen.setFillColor(blue);
        Brain.Screen.drawRectangle(0, 0, 500, 500);
      }
    }

    else if (not(Controller_Ruh.ButtonR1.pressing() ||
                 Controller_Ruh.ButtonR2.pressing() ||
                 Controller_Ruh.ButtonL1.pressing() ||
                 Controller_Ruh.ButtonL2.pressing() ||
                 Controller_Ruh.ButtonUp.pressing() ||
                 Controller_Ruh.ButtonDown.pressing())) {
      TopIntake.stop(coast);
      Intake.stop(coast);
    }
  }
}
```

October 2022

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

= Practice Days

Project October Calendar

Name Timothy Date 10/01/22

Week 6 Practices:

October 6th,, 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

There was a scheduled practice today, however due to certain circumstances, none of the team members were able to attend.

Week 7 Practices:

October 10th, 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

Recap:

- Autonomous Programming
- Code explanation

Autonomous Information:

This robot is the most programming-intensive robot we have made in our 4 years of doing robotics. To start, our design incorporates a turret, and the programming in the turret is more than we have attempted to code before. Our turret is powered by the same code we will use for autonomous. Due to time constraints, it is not entirely done yet, but the gist of the code is that it uses trigonometry to track the position of the robot. Every movement the robot makes can be described in terms of a line, from where it starts, to where it ends.

We can then take the distance of that line, along with the angle that the line is oriented at, (taken by the difference in initial and final angles found with the inertial sensor), and use component vector trigonometry to break down the resultant vector into two component vectors, which will be the x component and the y component. We can then add these components to our global x and y coordinates to get a form of absolute position tracking. This is obviously very useful for autonomous movement, and it also works to power our turret. If we know the position of the robot and the angle of the robot's orientation at all times, it is simple to tell the turret to always face the corner of the field with the corresponding goal. This way, through automation, we get a massive advantage over other robots in two major aspects of the competition, defense in driving, and programming skills. Defensive play will be abundant this year, due to its effectiveness. Against a normal design, the only thing a robot would have to do is to ram the opponent when they are lining up a shot. Our design design renders that largely ineffective, due to being able to shoot in a wide angle, and not just in a straight line in front of us. The turret also benefits us in programming skills, allowing for higher efficiency and speed because we do not need to line up every shot with our drivebase, we have a turret that does it for us. These our our final programming and design goals, although we have opted to not use the turret in this competition because it is not finalized yet, and we feel very confident without it.
The encoders we use for this odometry system are uniquely positioned on unpowered wheels on our drive train. The tracking wheels are mounted on an axle, but the normal axle hole is hollowed out, so the wheel does not spin with the rest of the drivetrain. The effect of this being that the wheel only moves when our robot moves, and not when our motors move. This way, say our robot is stuck on something or is up against a wall, if the robot drives forward it doesn't go anywhere. The encoder will not register movement. If we were to use the built in motor encoders, movement would be registered, which is not what is happening in reality.

We also have some form of flywheel speed control that we will use for this competition. We have buttons to raise and lower flywheel speed, giving us more range of control in our shots. We use the brain screen to keep track of this function. The flywheel starts on the lowest setting, denoted by the red brain screen. This is in reference to the red motor cartridge being 100 rpm. The other two settings, green and blue, (200 and 600 rpm respectively) represent the two higher flywheel settings. This way it is very easy to tell what speed our flywheel is currently running at in case we forget.

Aidan also plans to make a sort of countdown device to ensure there are no problems with the endgame expansion mechanism. Some teams have had problems with their driver accidently hitting a button for endgame expansion before the last 10 seconds, and he plans to fix that by implementing a countdown timer in the code. It is pretty simple, just a task running in the background that adds 1 to a counter every second. When the counter hits 95, (or 1:35), that will be the last ten seconds, and a variable will be updated allowing the endgame expansion to be used.

Week 8 Practices:

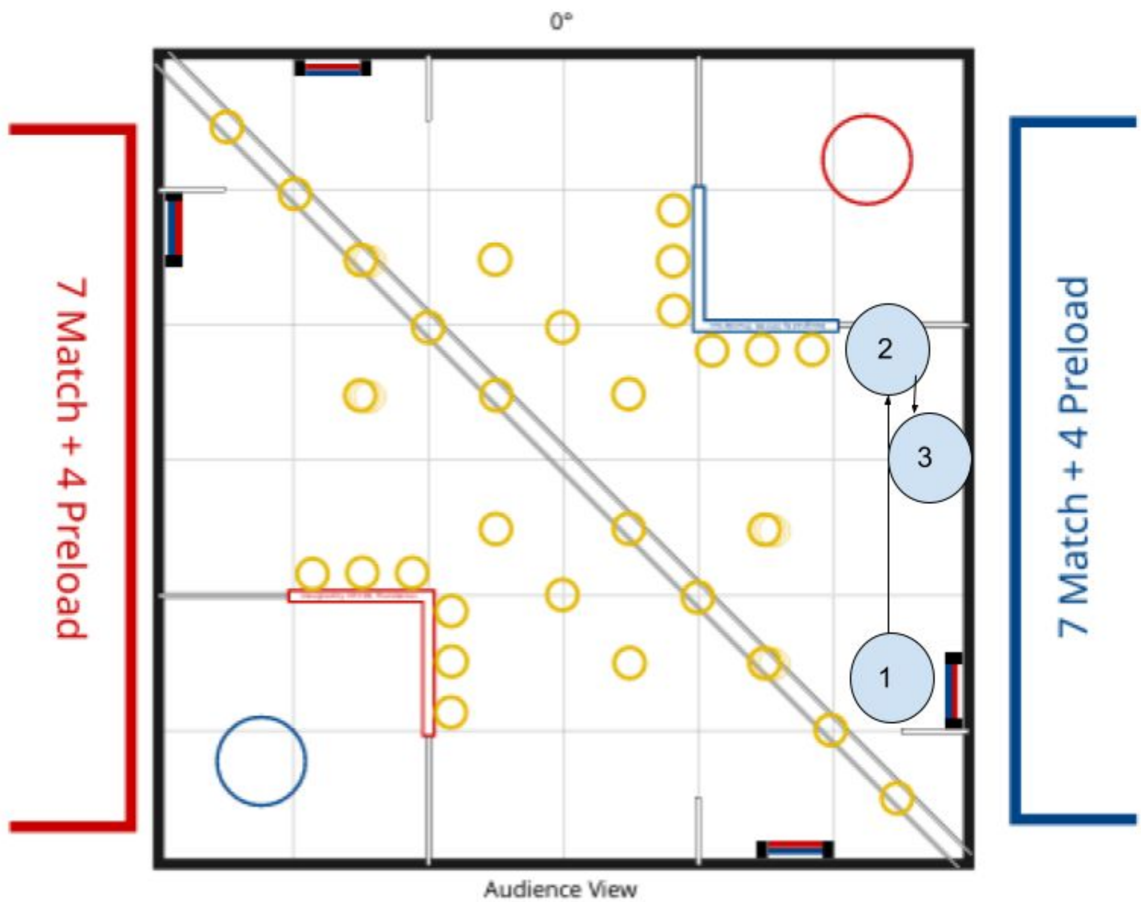
October 20th, 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

Recap:

- Programming / autonomous routes
- Autonomous code
- Autonomous explanations

As for our programming routes, we plan on having only one for in game autonomous as of now and one skills route. We were pressed for time in the programming aspect, so we looked for very fast autons that scored a decent amount of points. Our autonomous plans are described on the next page.

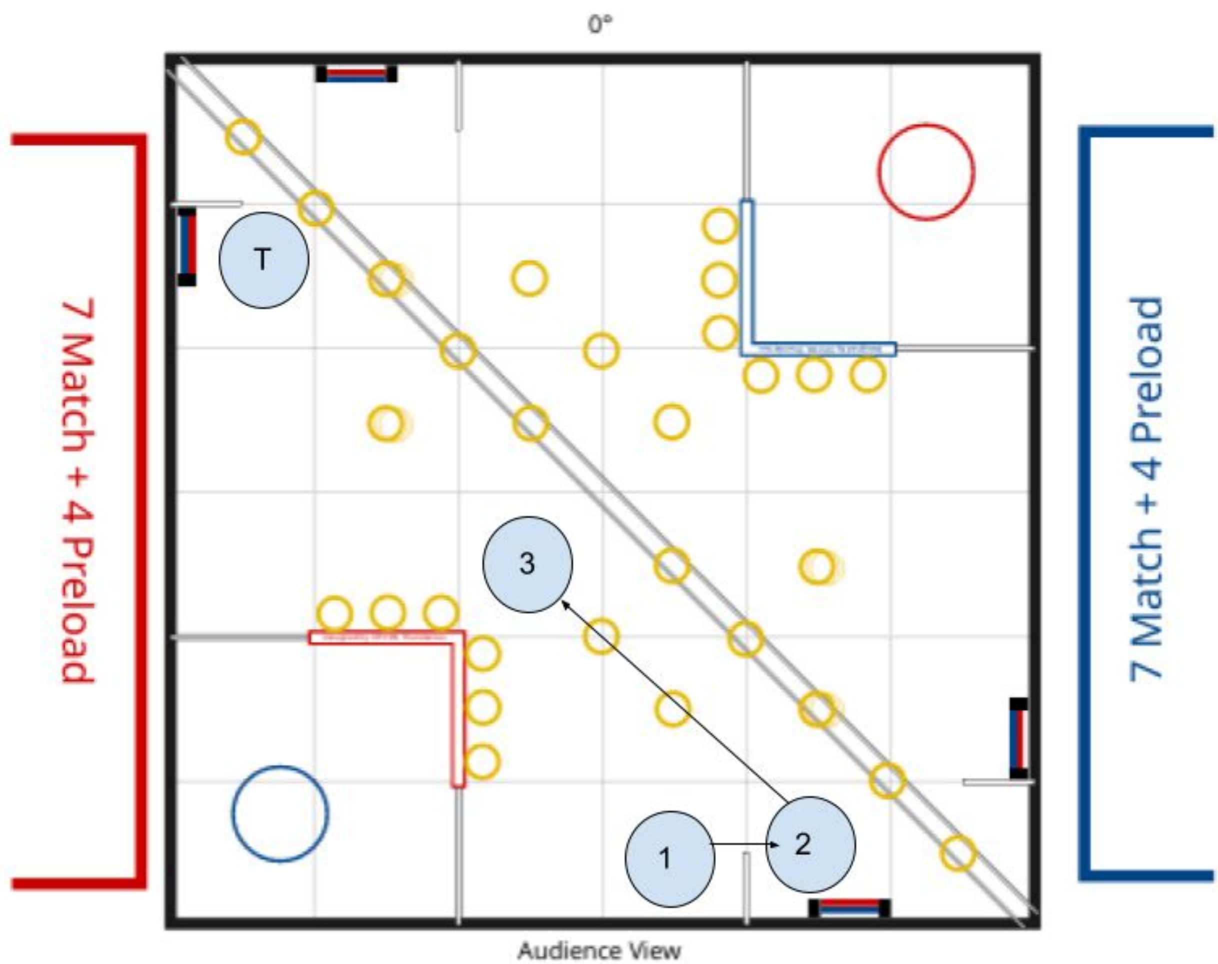


For our skills auton we went with a very simple plan. We start at figure 1. We spin the roller and go to figure 2, we shoot our two preloads and then go to figure 3, where the loader is located. We then use our 14 match load disks given to us in skills to load the robot. The robot then makes multiple trips from the loader to the goal, shooting each time. Assuming the robot makes 12 of its 16 shots and assuming no endgame expansion, this gives us a total of 74 programming skills points.

```
void autonomous(void) {
 //Hard roller side
 LeftRotation.setPosition(0, turns);
 driveBackward(6.5, 90);
 turnRight(3.1, 25);
 driveBackward(1.2, 40);
 wait(0.6, sec);
 DriveBase.spin(fwd, -10, rpm);
 TopIntake.spin(fwd, -50, rpm);
 wait(0.28, sec);
 TopIntake.stop(hold);
 DriveBase.stop(hold);
 RightDrive.spin(fwd, 80, rpm);
 LeftDrive.spin(fwd, 20, rpm);
 LeftRotation.setPosition(0, turns);
 waitUntil(LeftRotation.position(turns) * pi >
1.1);
 DriveBase.stop(hold);
 wait(0.4, sec);
 Intake.spin(fwd, 600, rpm);
 driveForward(7, 80);
 wait(0.3, sec);
 LeftDrive.spin(fwd, 30, rpm);
 waitUntil(Inertial.heading() > 45);
 DriveBase.stop(hold);
 wait(0.5, sec);
 driveForward(10, 80);
 Intake.stop(hold);
 wait(0.5, sec);
```

```
LeftDrive.spin(fwd, 30, rpm);
 RightDrive.spin(fwd, -30, rpm);
 Flywheels.spin(fwd, 155, rpm);
 waitUntil(Inertial.heading() > 131);
 DriveBase.stop(hold);
 TopIntake.spin(fwd, 200, rpm);
 wait(0.3, sec);
 Intake.spin(fwd, 200, rpm);
  //easy roller side
 driveBackward(0.2, 20);
 TopIntake.spin(fwd, -50, rpm);
 wait(0.3, sec);
 TopIntake.stop(hold);
 RightDrive.spin(fwd, 20, rpm);
 wait(0.8, sec);
 waitUntil(Inertial.heading(degrees) < 354);
 DriveBase.stop(hold);
 Flywheels.spin(fwd, 180, rpm);
 wait(3, sec);
 Intakes.spin(fwd, 60, rpm);
}
```

In-Game Autonomous:



For our ingame autonomous, we plan to hopefully get the auton win point, given our teammate's cooperation. We plan to start at figure 1, and then when the auton period starts, we go to figure 2 and spin that roller. We then go to figure 3 and shoot the three discs we pick up along the way towards the red goal when we stop.
This autonomous makes it as easy as possible on our teammates without actually doing the whole routine our self. The only thing our teammates have to do is back from their starting position (figure T) and spin the other roller. Given these, we get the autonomous win point.

Week 9 Practices:

October 25th, 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

Recap:
- Competition goals
- Programmer and Driver “Interviews”

Aidan’s favorite part of the robot is the intake section and turret. He believes that the turret is the defining section of the robot, and it has about 150 degrees of range. It is able to auto-lock onto the goals. An advantage that can not be matches by other robots at competitions.

Elliott believes that the most important part of our robot is the drive base which has 3 in line wheels on each side of the drive base and the middle wheel is free-spinning (it is driven by the other wheels)

Goals for this competition: This competition is mostly to see how we land on the “playing field.” However, we do hope to win.

November 2022

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

= Practice Days

= Competition Days

Project November Calender

Name Timothy

Date 11/01/22

Week 10 Practices:

November 1st, 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

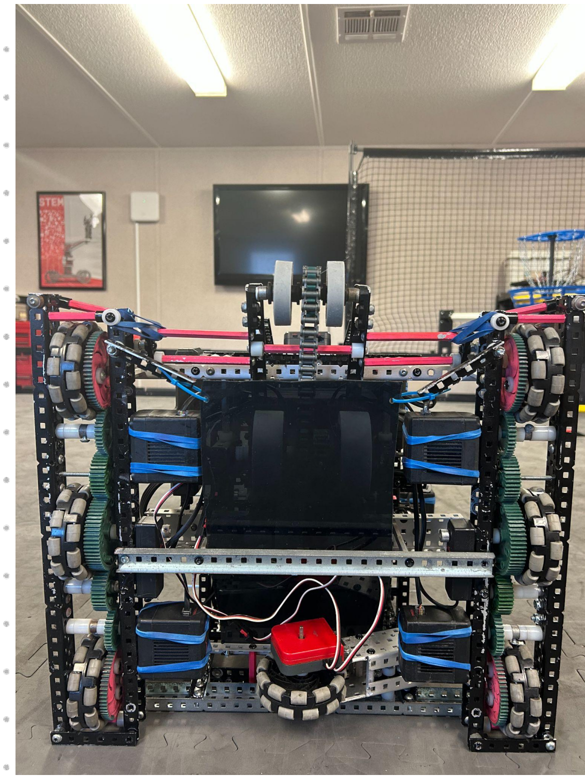
Recap:

- Take pictures of our robot for competition
- Create a scouting guideline for this Saturday
- Port and Control Drawing

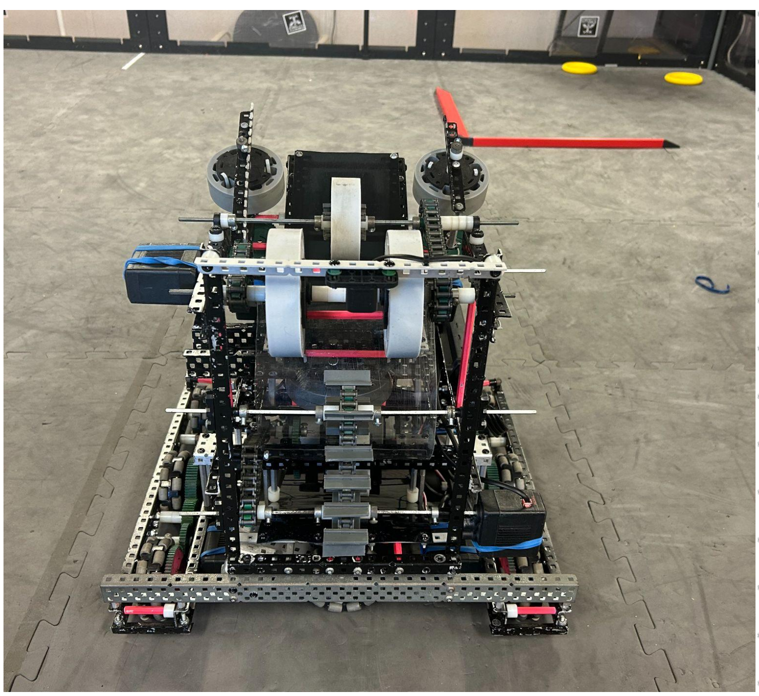
While we did talk about scouting, we decided not to do it before the competition this time, because there is no previous competition information to go off of for most teams.

Robot pictures are located on the next page.

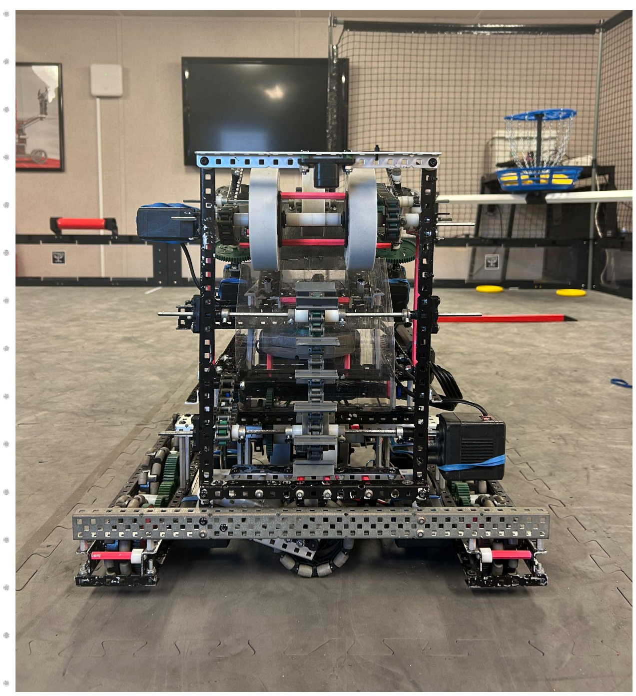
Front View



Drive Base



Top angle



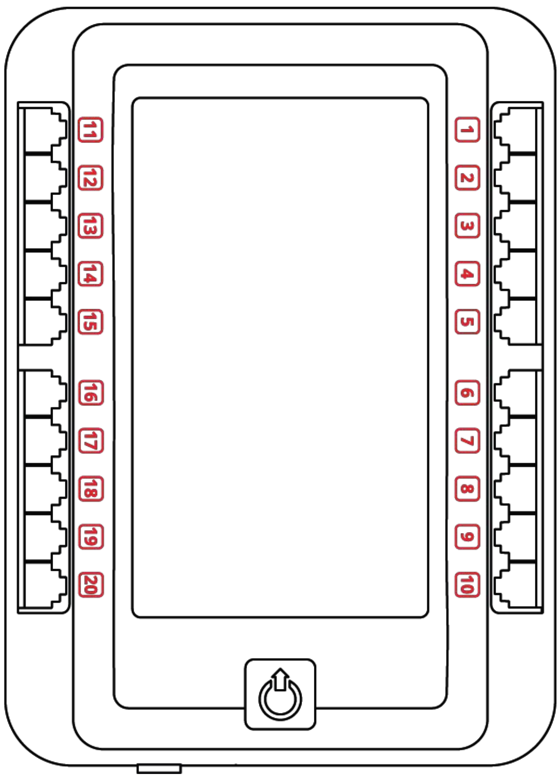
Back angle

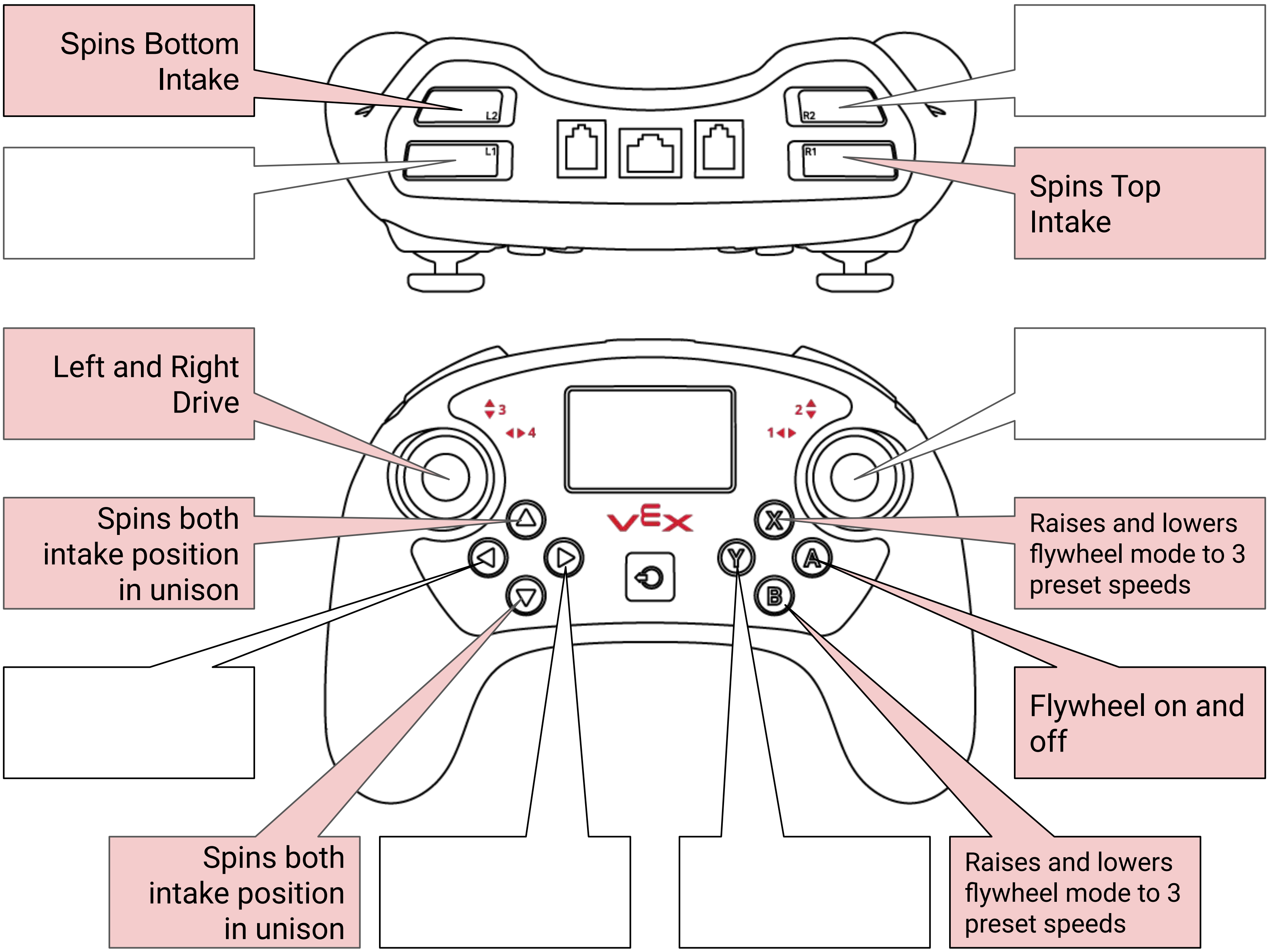
Project Robot photos

Name Timothy

Date

Top right (drive)	11	1	Bottom intake
	12	2	
Sensor - right rotation	13	3	
	14	4	Right flywheel
	15	5	Sensor - optical
	16	6	Sensor - inertial
	17	7	Left flywheel
Sensor - left rotation	18	8	
Back left	19	9	
Top left (drive)	20	10	Top intake




Spins Bottom Intake
L2
L1
R2
R1
Spins Top Intake
Left and Right Drive
3
4
2
1
Spins both intake position in unison
X
Y
A
B
Raises and lowers flywheel mode to 3 preset speeds
Flywheel on and off
Spins both intake position in unison
Raises and lowers flywheel mode to 3 preset speeds

November 5th, 2022

Tiger Throwdown Fall Qualifier:

Skills Rankings:

Rank	Team	Driver Attempts	Driver Highscore	Programming Attempts	Programming Highscore	Total Highscore
1	8926W	3	100	3	56	156
2	3796H	2	94	1	46	140
3	9447B	2	97	2	23	120
4	40994F	3	71	1	32	103
5	3796K	1	79	3	12	91
6	79267A	3	81	1	5	86
7	3796B	3	77	2	6	83
8	3796E	2	61	1	10	71
9	9880A	2	56	2	12	68
10	7432E	2	67	0	0	67
11	86400F	3	61	3	3	64
12	35016A	1	60	0	0	60
13	40994A	3	59	0	0	59
14	5278D	1	52	1	6	58
15	5278B	3	44	2	12	56

Our team’s ranking and qualifying:

Qualifier #6	**9447B**	42748B	76	35016X	35016A	66
Qualifier #21	9880C	8926W	182	3796E	**9447B**	63
Qualifier #26	3796D	**9447B**	34	86400F	35016C	79
Qualifier #45	7432B	42748A	51	3796C	**9447B**	84
Qualifier #58	**9447B**	42748C	114	44252A	44252N	42
Qualifier #68	40994B	3796K	82	**9447B**	97671B	58
QF #4-1	5278D	86400G	18	3796K	**9447B**	153
SF #2-1	3796B	8926W	102	3796K	**9447B**	34
R16 #8-1	3796K	**9447B**	123	98791B	35016Y	47

Rank	29
WP / AP / SP	6 / 20 / 314

Competition Recap:
- We made it to the semi finals
- Out qualifying ranking was 29
- Our skills placed us 3rd over

Goals:
- Change robot drive base for more durability and power
- Add mechanical deployment for end game expansion

Upcoming:
- Thanksgiving Break (November 21-25)
- Cat Midland Qualifier (December 3rd)

Week 11 Practice:

November 10th, 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

Timothy and Elliott worked on the main drive base as well as relocating the rotation seniors that are built into the drive base at the moment. Our team is going back to the system we used in previous years because of its simplicity and allowance for more room for gearing options in the drive base,

Moving the sensors allows us to have more driven wheels that prevent us from getting pushed around in the match (something we encountered in the last competition and are trying to combat).

To rebuild what we had from previous years we had to cut some very specific parts that will allow for tight clearance and saved weight.

Project Drivetrain tweaks
Name Timothy Date 11/10/22

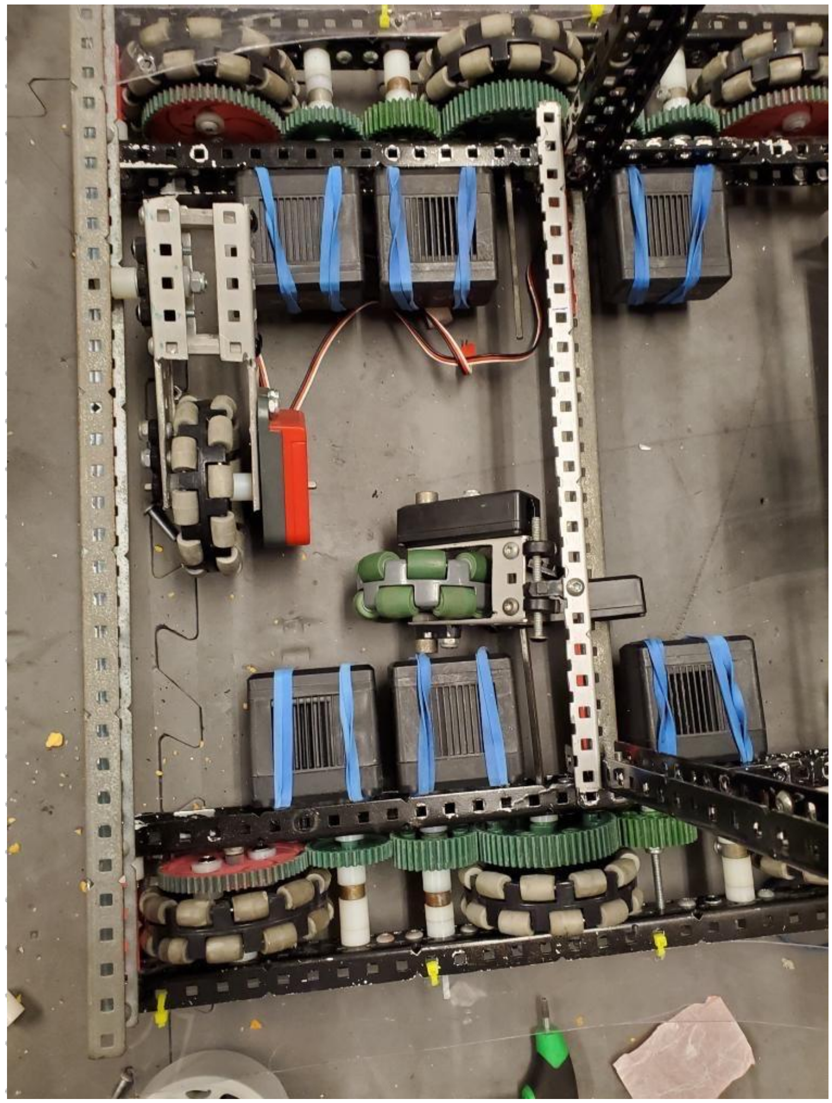
Week 12 Practice:

November 15th, 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

Today, Timothy and Elliot and we got a couple of things done. Elliott was working on the intake of the robot and the gearing for that. Timothy was working on both the rotational tracking and a string deployment mechanism. With the rotational tracking he simply added a wheel and sensor to the parts Ihemade last time and mounted it temporarily to the robot (they will be taken back off before the next competition). The reason he is building then now is so we can integrate them better into the our upcoming design that is going to have a fully functional turret. With the end game deployment, Elliott found a design online that Timothy will be basing our design off of. It's a very simple piston lever that is spring loaded and shoots a large nut with string attached. Each mechanism will use a different pneumatic cylinder so we can fire one at a time if needed.

*Photos of progress on the next page



December 2022

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

= Practice Days = Competition Days

Project December Calendar

Name Timothy Date 12/01/22

December 3rd, 2022

CATS Midland Qualifier HS

Skills Rankings:

Rank	Team	Driver Attempts	Driver Highscore	Programming Attempts	Programming Highscore	Total Highscore
1	8926W	2	137	2	83	220
2	3796B	3	108	1	9	117
3	3796A	3	98	3	18	116
4	7432E	1	74	0	0	74
5	7432B	2	65	0	0	65
6	97671A	3	51	1	10	61
7	44252A	2	60	0	0	60
8	9447B	1	59	0	0	59
9	97671C	3	57	0	0	57
10	3796C	3	52	0	0	52
11	97671B	3	50	1	0	50
12	3796D	2	49	0	0	49
13	44252N	1	49	0	0	49
14	916A	2	46	0	0	46
15	44252C	2	43	0	0	43
16	7432C	3	38	0	0	38
17	99322C	2	31	0	0	31

Our team’s ranking and qualifying:

Qualifier #5	3796C	916A	49	**9447B**	3796D	42
Qualifier #11	9623Y	**9447B**	48	9623S	97671B	52
Qualifier #13	7432C	97671A	39	3796A	**9447B**	102
Qualifier #18	99322B	**9447B**	51	7432B	44252A	98
Qualifier #27	97671C	3796B	134	**9447B**	1102X	65
Qualifier #32	**9447B**	44252N	55	7432B	44252C	82
R16 #2-1	7432B	**9447B**	64	9623Y	97671A	71

Rank	20
WP / AP / SP	2 / 30 / 300

Competition Recap:

Overall we placed 20th, losing 5 of out 6 matches.

Most of our matches were close and determined by a small mistake or problem. After this competition we have decided to change the robot to a catapult design, our builder is planning on creating CAD designs at the next practice.

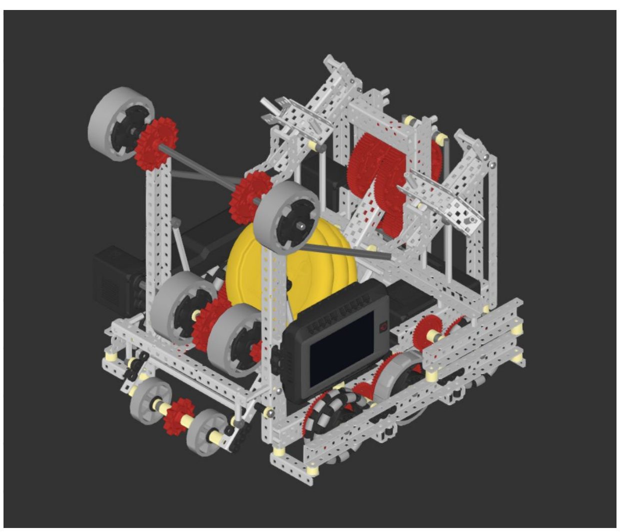
Week 13 Practice:

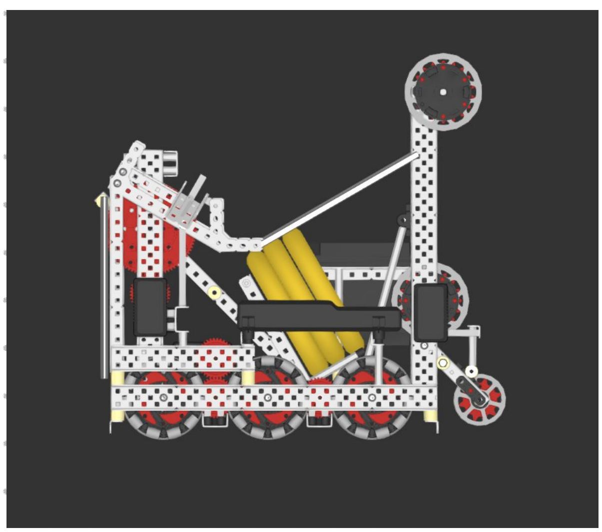
December 5th, 2022

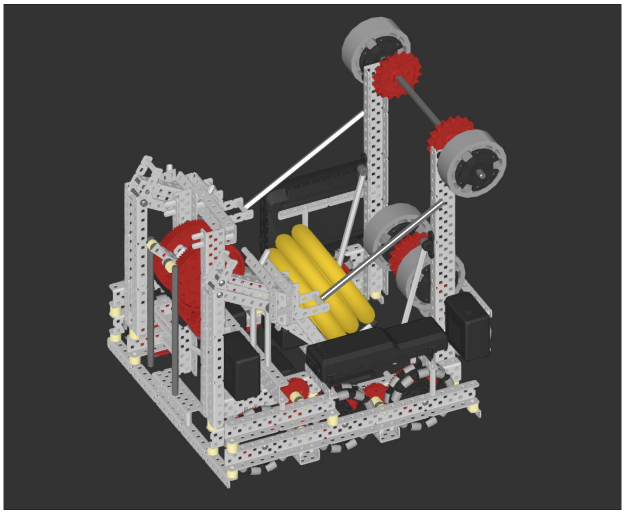
Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

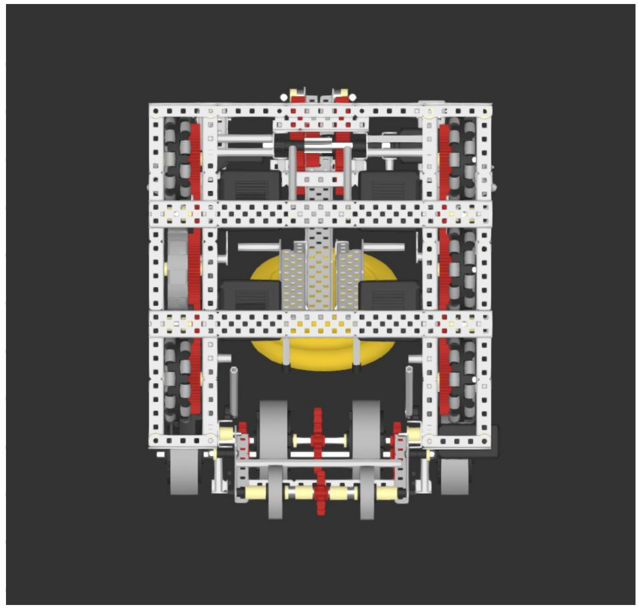
Goals:
- Drive Base CAD
- Catapult CAD

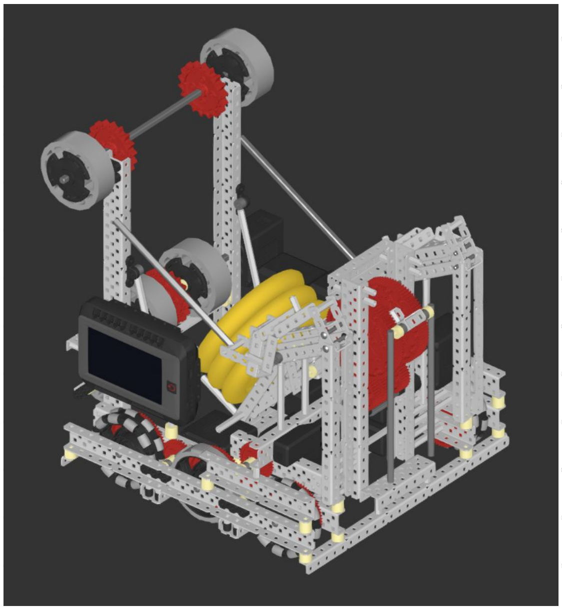
Elliott worked on making the new designs CAD today, to help start the process of rebuilding our robot.

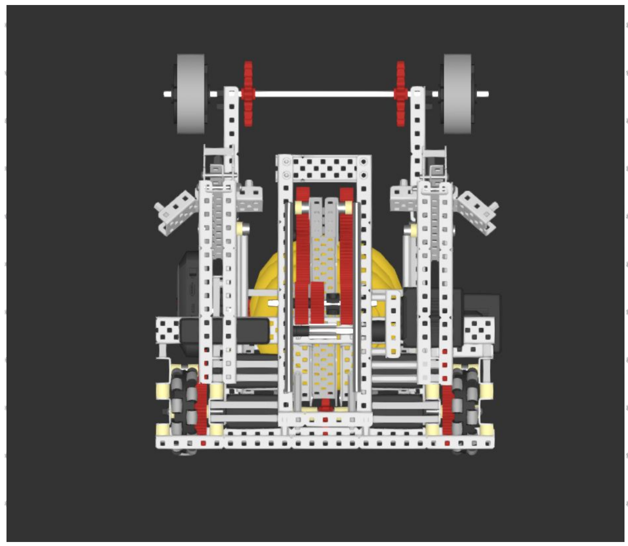


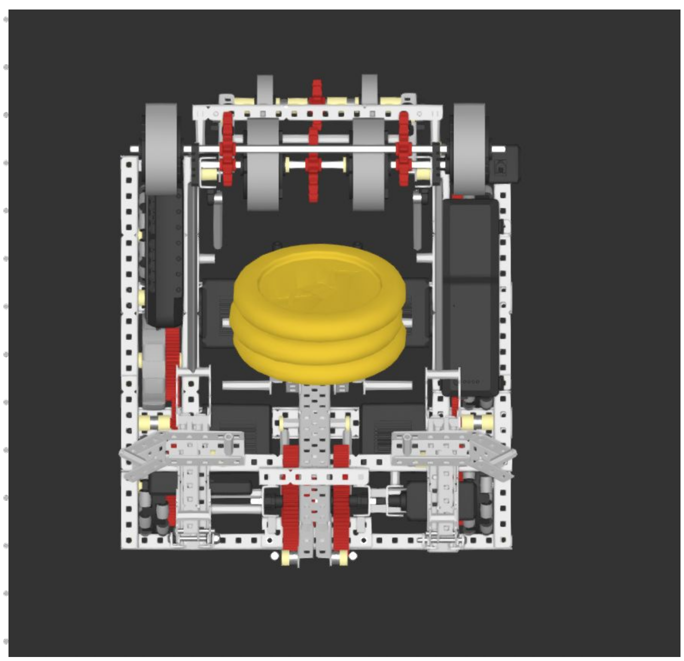


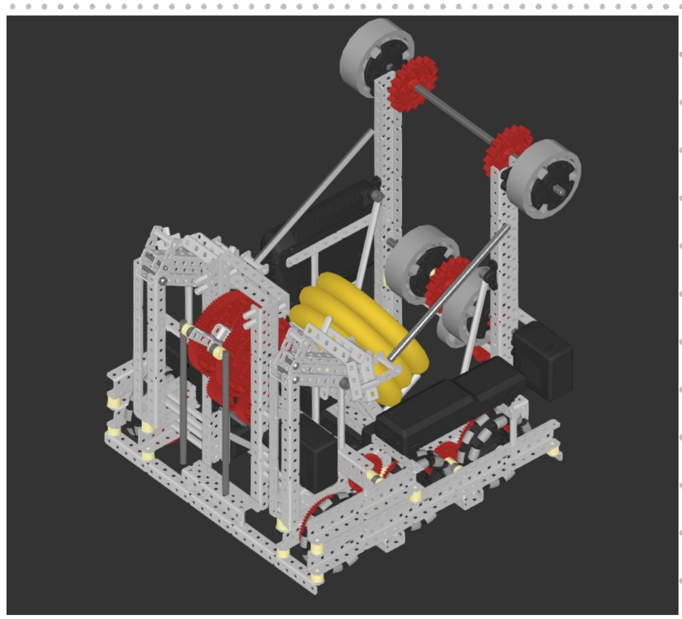












Week 14 Practice:

December 15th, 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

After examining another team's design online and the team's usage of custom pieces, Elliott decided to be more aggressive in our approach to scoring to make usage of our robots small frame.

One advantage is the fact that out robot's smaller frame allowed for easier maneuverability and ease of access to the high goals.

This ease of access to high goals makes it so that although we don't have extremely high-powered pistons compared to other teams, our scoring potential is still a threat to opposing alliances.

Week 15 Practice:

December 29th, 2022

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

With the amount of time remaining before the next competition, most of out time was committed to practicing our developed strategy to account for out lack of an extension.

Another focus was out autonomous and how well we could adjust out created autonomous based on changed to motor numbers and motor position relative to the robot.

A problem we had this week was adjusting motor functionality and finding the most optimal method to differentiate motors in out autonomous

- Solution:
    - On the following pages out created autonomous is showcased and shows the method of differentiations between motors.

Our programmer's code going into the competition will be placed in the following slides. Due to time constraints in the building of the robot, Aidan did not have as much time to program as he wanted to, but he was able to achieve a win point autonomous that should help us quite a bit in the course of the tournament.

Initializing motors and functions code:

VEXcode Pro V5 : 9447B_SpinUp_Com

main.cpp ● functionsAndMotos.h ●

```cpp
using namespace vex;

#include "math.h"

extern brain Brain;
vex::optical OpticalSensor = vex::optical(vex::PORT2);

vex::controller Controller_Ruh = vex::controller();
vex::motor TopLeft = vex::motor(vex::PORT5);
vex::motor TopRight = vex::motor(vex::PORT19, true);
vex::motor BackLeft = vex::motor(vex::PORT20);
vex::motor BackRight = vex::motor(vex::PORT18, true);
vex::motor TopBackLeft = vex::motor(vex::PORT4, true);
vex::motor TopBackRight = vex::motor(vex::PORT9);

vex::motor Catapult = vex::motor(vex::PORT10);
vex::motor Intake = vex::motor(vex::PORT1);

vex::motor_group LeftDrive = vex::motor_group(TopLeft, BackLeft, TopBackLeft);
vex::motor_group RightDrive = vex::motor_group(TopRight, BackRight, TopBackRight);
vex::motor_group DriveBase = vex::motor_group(TopRight, BackRight, TopLeft, BackLeft, TopBackRight, TopBackLeft);

vex::task timeCountTask;  //set up task

int times = 0;

 int timeCount ()
{
  times = 0;
  while(1)
  {
    wait(1, sec);
    times++;      //simple timer, every second add one to variable, which stands for seconds that have passed ingame
  }
}
```

Output (61) Problems (0) Terminal (0)

Driver Code:

VEXcode Pro V5 : 9447B_SpinUp_Com

main.cpp • functionsAndMotos.h •

```cpp
void usercontrol(void) {
  vex::task timeCountTask(timeCount); //keep track of driver time
  // User control code here, inside the loop
  while(1)
  {
    RightDrive.spin(fwd, Controller_Ruh.Axis3.position() * 1.50, rpm);  //tank control better than arcade
    LeftDrive.spin(fwd, Controller_Ruh.Axis2.position() * 1.50, rpm);

    if(Controller_Ruh.ButtonL2.pressing()) //intake
    {
      Intake.spin(fwd, 125, rpm);
    }
    else if(Controller_Ruh.ButtonL1.pressing()) //outtake
    {
      Intake.spin(fwd, -125, rpm);
    }
    else
    {
      Intake.stop(coast); //intake stop
    }

    if(Controller_Ruh.ButtonR2.pressing())  //cata shoot
    {
      Catapult.spin(fwd, 200, rpm);
    }
    else if(not(Controller_Ruh.ButtonR2.pressing()) && not(LimitSwitchA.pressing()))  //if not pressing button or cata not in place, spin cata
    {
      Catapult.spin(fwd, 200, rpm);
    }
    else if(not(Controller_Ruh.ButtonR2.pressing()) && (LimitSwitchA.pressing())) //if cata pressing limit switch, stop cata
    {
      wait(0.05, sec);
      Catapult.stop(coast);
    }
```

Output (61) Problems (0) Terminal (0)

Ln 147, Col 66

VEXcode Pro V5 : 9447B_SpinUp_Com

main.cpp • functionsAndMotos.h •

```cpp
      Intake.spin(fwd, 125, rpm);
    }
    else if(Controller_Ruh.ButtonL1.pressing()) //outtake
    {
      Intake.spin(fwd, -125, rpm);
    }
    else
    {
      Intake.stop(coast); //intake stop
    }

    if(Controller_Ruh.ButtonR2.pressing())  //cata shoot
    {
      Catapult.spin(fwd, 200, rpm);
    }
    else if(not(Controller_Ruh.ButtonR2.pressing()) && not(LimitSwitchA.pressing()))  //if not pressing button or cata not in place, spin cata
    {
      Catapult.spin(fwd, 200, rpm);
    }
    else if(not(Controller_Ruh.ButtonR2.pressing()) && (LimitSwitchA.pressing())) //if cata pressing limit switch, stop cata
    {
      wait(0.05, sec);
      Catapult.stop(coast);
    }
```


```cpp

     if(times < 94) //if seconds in driver control have passed 94 (1:34), allow for activation of endgame
     {
       if(Controller_Ruh.ButtonA.pressing())
       {
        Endgame.set(true);
       }
     }

  }
}
```

Output (61) Problems (0) Terminal (0)

Ln 147, Col 66

Autonomous In-game:

VEXcode Pro V5 : 9447B_SpinUp_Com

9447B_SpinUp_Com

main.cpp ● | functionsAndMotos.h

```cpp
void autonomous(void) {
  Catapult.stop(hold); //added layer of protection for cata slipping
  DriveBase.spin(fwd, -20, rpm);
  Intake.spin(fwd, -100, rpm); //back into roller and spin it
  wait(0.3, sec);
  Intake.stop(hold);
  DriveBase.stop(hold);
  LeftDrive.spin(fwd, 45, rpm); //drive off of roller
  RightDrive.spin(fwd, 45, rpm);
  wait(0.5, sec);
  LeftDrive.stop();
  RightDrive.spin(fwd, 50, rpm);
  waitUntil(Inertial.heading() > 38); //spin until driving diagonally along auton line
  DriveBase.stop(hold);
  wait(0.3, sec);
  TopLeft.resetPosition();
  DriveBase.spin(fwd, 60, rpm);
  wait(0.3, sec);
  DriveBase.spin(fwd, 100, rpm);
  waitUntil(TopLeft.position(turns) > 3.5); //drive to middle of field, slow speed at start to prevent acceleration jerking
  DriveBase.stop(hold);
  wait(0.2, sec);
  RightDrive.spin(fwd, -40, rpm);
  LeftDrive.spin(fwd, 40, rpm);
  waitUntil(Inertial.heading() > 200);
  wait(0.1, sec);
  waitUntil(Inertial.heading() < 322); //turn to face opposing goal
  DriveBase.stop(hold);
  wait(0.2, sec);
  DriveBase.stop(hold);
  Catapult.spin(fwd, 200, rpm); //shoot catapult
  wait(0.6, sec);
  while(not(LimitSwitchA.pressing()))
    {
      Catapult.spin(fwd, 200, rpm); //reset catapult to load position
```

Output (39) | Problems (0) | Terminal (0)

VEXcode Pro V5 : 9447B_SpinUp_Com

9447B_SpinUp_Com

main.cpp ● | functionsAndMotos.h

```cpp
  Catapult.spin(fwd, 200, rpm); //shoot catapult
  wait(0.6, sec);
  while(not(LimitSwitchA.pressing()))
    {
      Catapult.spin(fwd, 200, rpm); //reset catapult to load position
    }
  wait(0.05, sec);
  Catapult.stop(coast); //rubber band boosting is off, no need for cata to be set to hold now
  Intake.spin(fwd, 200, rpm); //spin intake to pick up discs
  DriveBase.spin(fwd, -40, rpm);
  TopLeft.resetPosition();
  waitUntil(TopLeft.position(turns) < -0.36); //back up to align with discs
  RightDrive.spin(fwd, -40, rpm);
  LeftDrive.spin(fwd, 40, rpm);
  waitUntil(Inertial.heading() < 225);  //turn to roller
  DriveBase.stop(hold);
  wait(0.4, sec);
  TopLeft.resetPosition();
  DriveBase.spin(fwd, -20, rpm);
  wait(0.4, sec);
  DriveBase.spin(fwd, -70, rpm);  //drive to roller
  RightDrive.spin(fwd, -60, rpm);
  wait(0.5, sec);
  DriveBase.spin(fwd, -70, rpm);
  waitUntil(TopLeft.position(turns) < -3);
  RightDrive.stop(hold);
  LeftDrive.spin(fwd, -60, rpm);
  waitUntil(Inertial.heading() > 256);  //turn to roller
  DriveBase.stop(hold);
  TopLeft.resetPosition();
  DriveBase.spin(fwd, -40, rpm);
  Intake.spin(fwd, -100, rpm);
  waitUntil(TopLeft.position(turns) < -0.3);  //back into roller
  wait(0.3, sec);
  DriveBase.stop(coast);  //stop drive base
  Intake.stop(hold);
```

Output (39) | Problems (0) | Terminal (0)

Autonomous In-game (PT 2):

VEXcode Pro V5 : 9447B_SpinUp_Com

9447B_SpinUp_Com

main.cpp ● | functionsAndMotos.h

```
    }
    wait(0.05, sec);
    Catapult.stop(coast); //rubber band boosting is off, no need for cata to be set to hold now
    Intake.spin(fwd, 200, rpm); //spin intake to pick up discs
    DriveBase.spin(fwd, -40, rpm);
    TopLeft.resetPosition();
    waitUntil(TopLeft.position(turns) < -0.36); //back up to align with discs
    RightDrive.spin(fwd, -40, rpm);
    LeftDrive.spin(fwd, 40, rpm);
    waitUntil(Inertial.heading() < 225);  //turn to roller
    DriveBase.stop(hold);
    wait(0.4, sec);
    TopLeft.resetPosition();
    DriveBase.spin(fwd, -20, rpm);
    wait(0.4, sec);
    DriveBase.spin(fwd, -70, rpm);  //drive to roller
    RightDrive.spin(fwd, -60, rpm);
    wait(0.5, sec);
    DriveBase.spin(fwd, -70, rpm);
    waitUntil(TopLeft.position(turns) < -3);
    RightDrive.stop(hold);
    LeftDrive.spin(fwd, -60, rpm);
    waitUntil(Inertial.heading() > 256);  //turn to roller
    DriveBase.stop(hold);
    TopLeft.resetPosition();
    DriveBase.spin(fwd, -40, rpm);
    Intake.spin(fwd, -100, rpm);
    waitUntil(TopLeft.position(turns) < -0.3);  //back into roller
    wait(0.3, sec);
    DriveBase.stop(coast);  //stop drive base
    Intake.stop(hold);
    DriveBase.spin(fwd, 20, rpm);
    wait(0.5, sec);
    DriveBase.stop(hold); //back off of roller
                          //end of ingame auton solo awp
    }
```

Output (39) | Problems (0) | Terminal (0)

Search

2023 JANUARY

SUNDAY	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY	SATURDAY
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

= Practice Days

= Competition Days

Week 16 Practice:

January 5th, 2023

Attendance:	Here:	Not Here:
Elliott		
Aidan		
Duaa		
Mohamed		
Timothy		

We as a team talked about upcoming competitions (Great Wolf Lodge and Clemson) and how we want to prep for these competitions and what we wanted to complete before attended them.

We want to have our expansion done by Great Wolf Lodge.

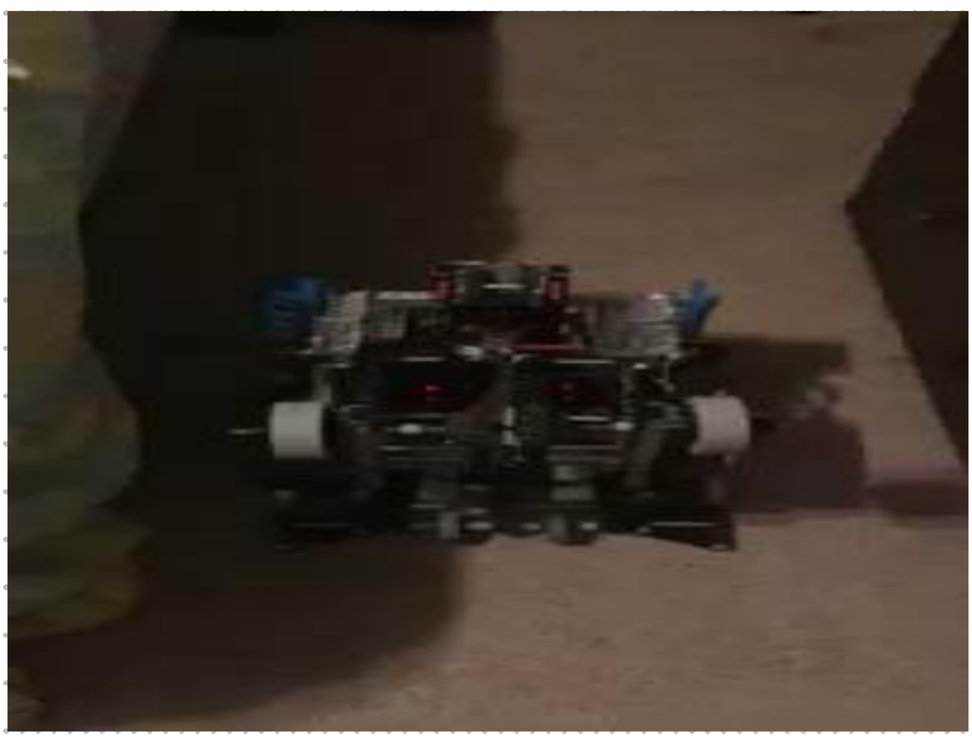
Currently, Aidan is working on the P controller. The Code and the video demonstration are located the next slide.

Elliott worked on driving the robot today, and practicing this aim while using the new catapult design. A demonstration is located in two slides.

P Controller Programming and Video:

```
void pDrive(double driveDistance)
{
  maxSpeed = 0;
  error = 10000; //set to 10000 to enter the loop
  while(error > 0.1 || error < -0.1)  //while distance to travel is greater then 0.1 run the pid loop
  {
    maxSpeed = maxSpeed + 10; //slew code to reduce acceleration jerking to left or right
    error = (driveDistance - TopLeft.position(turns)); //sets error to end point minus distance traveled
    motorPower = error * kP;  //p controller: sets drive speed proportional to distance left to travel
    if(motorPower > maxSpeed)
    {
      motorPower = maxSpeed;  //slew code continuation
    }
    DriveBase.spin(fwd, motorPower, rpm); //spin motors
    Brain.Screen.printAt(20, 20, "error: %f", error);
    Brain.Screen.printAt(20, 60, "motorPower: %f", motorPower); //display values for tuning and troubleshooting
    wait(0.08, sec);
  }
  DriveBase.stop(hold); //stop drive when done
}
```

Demonstration of the P controller and the code going with it. The P controller is a subset of PID controller, in which the P stands for proportional. This code sets the power of the motors proportional to distance still needed to travel, allowing for fast acceleration and deceleration while still making very accurate movements.



These two videos showcase the three main aspects of a catapult that drew us to use this design over a flywheel, its efficiency in firing discs, its consistency in grouping the discs, and the impressive overfill capacity of the catapult design. Since the catapult fires upwards instead of shooting at the goal, the catapult is able to fill the goal with many more discs than a flywheel is able to.