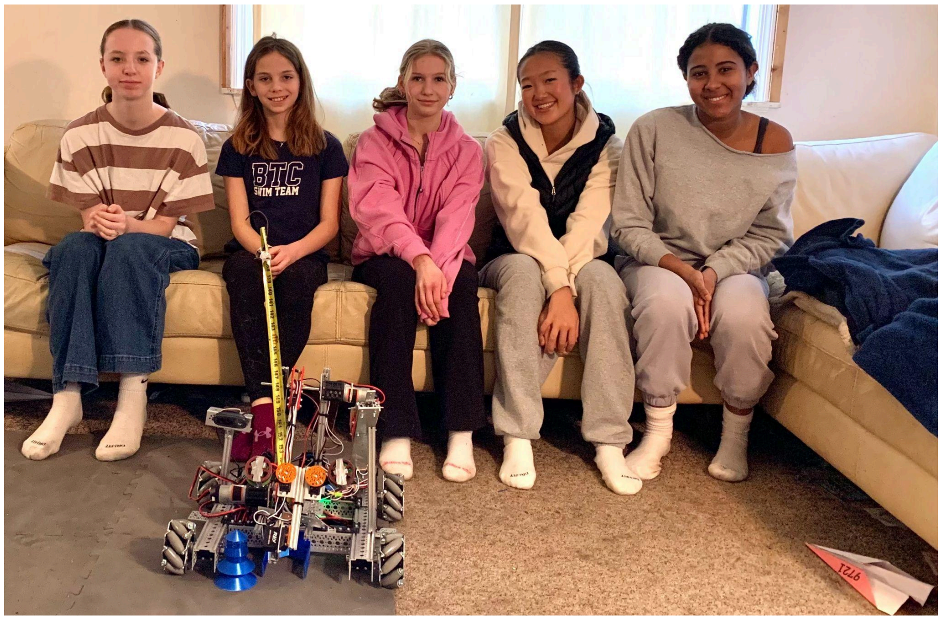
Green Mountain Gears 9721

South Burlington Vermont

TIPS(Team of Innovative Problem Solvers)



About the Team

The current team resulted from merging teams 9721 and 16295, which were both all girls teams. The teams merged to improve the quality of their experience.

We do not have strict roles on the team. Everyone does everything and whoever shows up works on the topic for that day. Everyone learns CAD, to build, programming, and operating the robot. We take turns driving the robot and operating the arm.

The teams have been attempting to build a "Robot Sisterhood" for the past five seasons. The "Robot Sisterhood" would be a coalition of FIRST teams, from Jr. FLL through FRC that meet in the same place so that the older teams can mentor the younger teams in real time while sharing equipment, mentors and experiences.

At one point, the"Robot Sisterhood" had three all girls teams but it has been a challenge to recruit and retrain girls in the post-pandemic space. The team hopes to make progress in this effort next year before Kayla, Sage and Yorda leave for college.

The team communicated with another all-girls team in North Carolina in an effort to promote regional all-girls teams but the last (2) emails to NC received no response.

Team 9721

- 2022-23 Power Play All Girls Team Launched mecum chassis with autonomous
- 2021-22 Freight Frenzy Complete Turn-over with transition to all-girls team.
- 2020-21 Ultimate Goal Dean's List Finalist, Luke Fitzgerald; 2nd Place Innovation and Connections Award.
- 2019-2020 City Shapers Shapers 1st Place Think Award, Participated as member of Semi-Final Alliance.
- 2018-19 Rover Ruckus Best of Vermont, Member of Semi-Final Alliance
- 2017-18-Relic Recovery Best of Vermont

Team 16295

- 2022-23 Competed in Power Play
- 2021-22 Freight Frenzy Team won the Connect Award-35+ pts.
- 2020-21 Ultimate Goal Team Participated in Virtual Competition-10 pts.
- 2019-20 Skystone Rookie season-5 pts.

Team Members

Cailin Fitzgerald

Hello, my name is Cailin! I am 12 years old, and have done 3 years of FTC, including this one. I love working with metal, and pushing the limits of my brain for creative and practical solutions to problems. As a part of an all-girls team, teamwork is a huge part of practice. I am the youngest, but we all treat each other as equals and collaborate to find the best solution to a problem.
I started playing the violin 8 years ago, when I was four, and I've been climbing and swimming my whole life, and have a blue belt in Aikido, which is a Japanese martial art.
When I grow up… I don't really know what I want to do, but maybe something involving animals.I am super excited to compete and see if all of our hard work has paid off.

Yorda Gebreselasie

Hello, my name is Yorda Gebreselasie, and this is my second year of FTC. I love working with math problems, and am curious to learn new things while exploring Robotics.

I am a junior in highschool at South Burlington High School, and have been playing sports since a young age. I have played soccer for the past 11 years, and have competed in varsity track throughout highschool. I have explored leadership roles being on Student council, and SJU (student justice union) at my school.

I plan to continue my education after highschool and aspire to become a Pediatrician. Being in FTC helps me reach this goal as I learn more about machine learning, and can help translate to the medical field, as machine learning can help identify issues with patients in hospitals.

Being on an all-girls team is important to me as we can show that girls can compete in the stem field, especially in Robotics where the competition is dominated by boys. I admire the dedication my teammates and I have put into this!

Kayla Kim (Captain, nominated for Dean's List Award)

Hello, my name is Kayla, and I am a 16 year old junior going into my fifth year of FTC. Outside of school, I play field hockey competitively all-year round on my school's varsity team, varsity track and field, and lacrosse. I value staying involved in our school by being a part of the student council and spelling club. My plans for post-graduation are working towards being a dermatologist. Incorporating robotics in my life consistently has been helpful in problem-solving and critical thinking at school, as well as translating to values that are important

to me. For example, being on an all-girls team and producing an environment that encourages optimal creativity for everyone on our team has been great.
I started playing the piano at four years old, and play the flute in my school's band.
I'm very excited for the progress we've made this year!

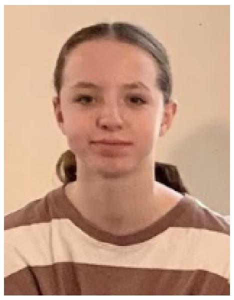
Sage Peterson(Captain, nominated for Dean's List Award)

Hi, my name is Sage, and this will be my fifth year of participating in FTC. I'm a junior at CVU, and along with robotics I play tennis all year round.

Participating in Robotics has enriched many areas of my life and shown me that any job can require machine learning and stem. I'd love to go to nursing school and become a psychiatric nurse practitioner, and I hope my involvement in FTC will help me reach this goal.

Participating in an all girls robotics team has allowed me to see how important it is to empower women in stem and I hope our team can encourage more young girls.

Abby Lynch



Hello, my name is Abby. I am 13 years old. I joined the robotics team just under a month ago and have loved making small contributions to the final project team. When I am not at robotics or school, you can find me in the studios at Vermont Ballet Theater School.

The team was coached by Paul Fitzgerald and run from his home. This is his 14th year with FIRST. He founded TIPS to support FIRST teams, which have included Jr.FLL, FLL, FTC and FRC teams. He aspires to build a "Robot Sisterhood" where there is a dedicated space for girls robotics teams to practice side by side, from k-12.
The team was mentored by two alumnus, Luke Fitzgerald and Andrew Kim. Luke is studying math and computer science at MIT and Andrew is studying Biology at Brown.

Financing the Team

The team was supported by grants from UVM to cover the registration costs for the Vermont Championship Event. ($275)

The team was supported by a roll-over balance from previous grants to the 501.c3 Teams of Innovative Problem Solvers (TIPSVT) to cover the registration fee for FIRST and to purchase the robot driver station. ($600 with shipping). Team also used grants to pay for the half-field, which was $500.

The team planned to reuse systems when possible and to utilize surplus materials from previous seasons. The team reused a chassis (frame, motors, wheels) from last year and an arm from two years ago. The team purchased no new materials for the robot this year. Many of these materials were purchased with VASE grants for team 9721 (2021) and 16295 in (2021 and 2020)

The team planned to design and print custom parts using a 3d printer and materials from a Vermont Academic of Science and Engineering (VASE 2022) grant from last year. There were no additional costs for these materias.

The participation cost for each team member was 0 because of the extensive re-use of materials.

Planning for the Season

1. Review the game objectives to identify potential robot behaviors. (September)
2. Set goals based on past performance, materials, and expanded skills. (October)
3. Identify specific robot behaviors(subsystems) to meet goals (October)
4. Research previous designs on robot behaviors.(October)
5. Prioritize subsystems to maximize points so that as time goes by, the robot earns more points.(October)
6. Brainstorm prototypes (November-December)
7. Prioritize testing prototypes based on difficulty (easiest first) and the availability of materials(built with what we have on hand) and reuse old designs when possible.(November-December)
8. Integrate new systems into existing systems.(November-December)
9. Revise and improve with testing. (January/February)

The game objectives for Centerstage are

1. to use TFOD to place pixels and park in Autonomous
2. place pixels in Driver-Controlled
3. Shoot a paper airplane and hang in the endgame.

Our goals were to autonomously detect a team-designed prop using TFOD(Tensor Flow Object Detection) , place a pixel and park. Place 5 pixels in the backstage for the driver-controlled period, and shoot our paper airplane and hang in the endgame.

In order to reach these goals the robot must be able to drive straight, turn and strafe. The robot must be able to grab a pixel, carry it and place it on the backstage. To shoot the airplane, the robot must be able to hold the plane, then *launch* it. To hang, the robot needs to grab the bar, pull itself up, and hold its position.

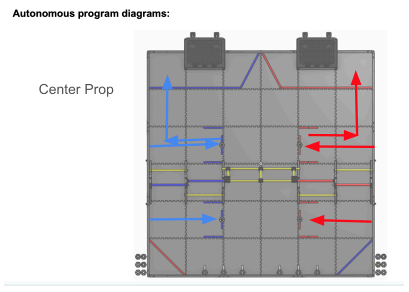
In our research phase, we looked at new build videos from goBilda, REV, and Andymark.

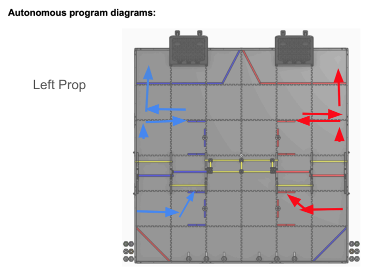
Our top priority was the drive system, because a robot that can't move, can't score. Hanging was the next priority, because it was simple, we had done it before in FLL, and it scored 20 points. The drone was the next priority, because it could score up to 30 points and it seemed easier than the pixels. After the drone was the arm and gripper, because we had built it before. We saved the machine learning until all of the other systems had been set, because we had to wait to make sure the camera wouldn't move to make room for other systems. The final thing we built was the pusher, which was 3d printed and relied upon the machine learning model. After all of the systems had been placed, we tested and revised the robot in its entirety.

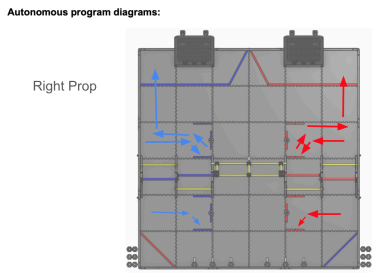
Cailin, Yorda and Kayla at FTC Scrimmage at SBHS. The chassis, the arm, the lifter and the drone launcher were all re-designed after the performance at the event.

The Robot

Our robot scores from 76-81 points and scores in all (3) phases of the competition.







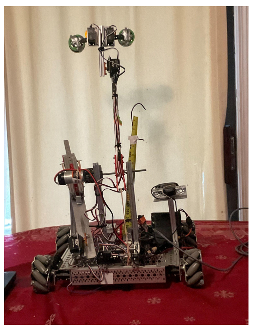
In autonomous, we can place the purple pixel, and if we are in the position closer to the backstage, park as well. We would like to have that position, but will defer it if our alliance team does all of this and uses the april tag to place the yellow pixel.

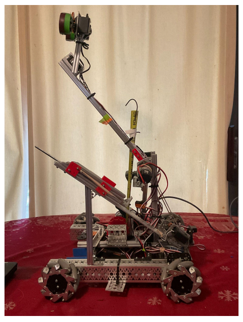
TeleOp

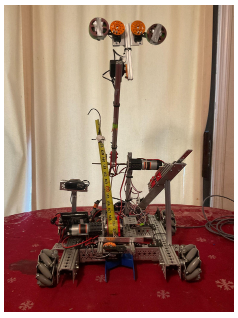
In Teleop, we can drive under the 14" bar and place 4-6 pixels, if they are on the same side as the platform and the pixels are on the same side. We do not want to go under the bars in Teleop because that would risk losing 30 points if we drop the paper airplane on accident.

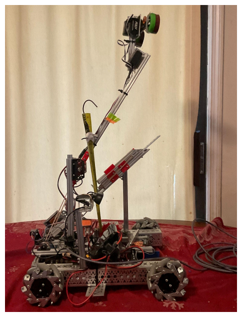
End Game

In Endgame, we can shoot our airplane to get 30 points, and hang on a truss. We cannot go under the bar after we shoot our plane because the launcher extends beyond 14 inches.



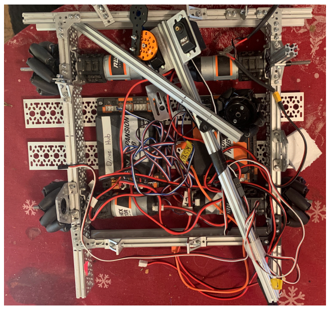






The chassis (Cailin 2/7/24)

The chassis shape is a H shape, which is symmetrical and allows for items to be collected and deposited by pushing. It can drive under the 14 inch bar while going backwards with the plane.



The current chassis replaced a previous prototype, which had 2 inch wheels and was made of all C channels, no U channels, and had motores in the middle, which filled up space for other things.

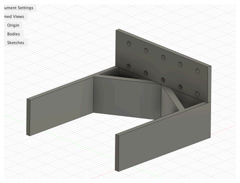
The chassis frame, and the mecanum wheels, contain 2 U and C channels from REV, which are compared to GoBilda's version, which is now the default because it uses its basic arm as a means to pull itself up.

The chassis is connected with REV 90 degree brackets on both the top and the bottom. The crossbars are Rev C channels, attached with L brackets on the top and bottom. Because if it were only on the top or bottom, the frame would flex.

The chassis frame is made from aluminum and weighs about 30 pounds.
It has 4 ultra planetary motors with a 4 : 1 and a 3 : 1 gear ratio, which allow it to be faster but with less torque than it the motors were not geared this way. The motors are attached via a bevel gear box, so that they can be stowed inside the channel of the robot, leaving more space open for the electronics. This also mounts the motors on two planes, instead of one, which make the motors more stable.

The motors are equipped with a 4" mecanum wheel, which is a heavy duty wheel bought by the team more than 5 years ago. The mecanum wheels have cylinders angled at a 45 degree angle, designed to strafe, and be mounted to the chassis while being parallel, instead of the Omni wheels, which have to be mounted at a 45 degree angle.

The wheels are angled to drive, strafe, and turn. The wheels are attached using Tetrix motor hubs, which have a much larger set screw when compared to rev collars. This allows a better connection to the axel, and allows the wheels to connect to the hub with a screw which is not possible with the rev hubs.

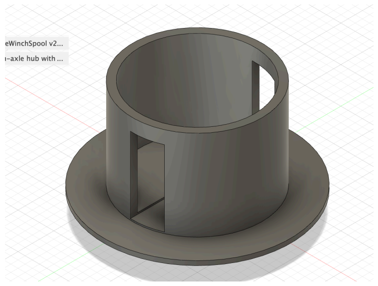


The chassis uses a custom designed and printed pusher to hold the purple pixel during autonomous. This is the third version. The first version couldn't strafe, and the second version mounted with one screw and broke in testing. The final version mounts with two screws.

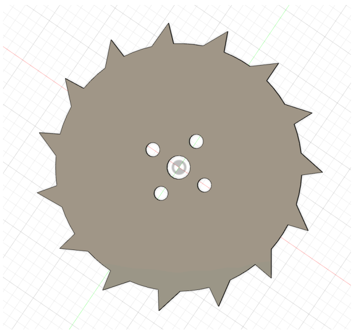
Lifter (Abby and Cailin 2/5/24)

The winch uses a Ultra Planetary motor with a 125:1 gear ratio. Having a 125:1 gear ratio is like a low gear on a bike going uphill, so it's easier to move but takes longer.

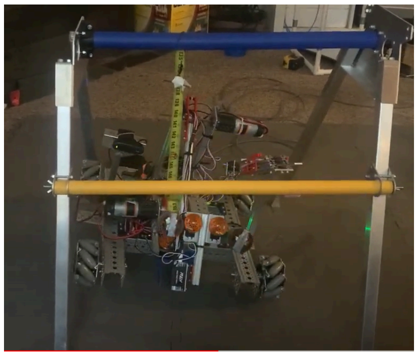
The hub sends power from the motor through the wire which spins the axle. The axle causes the spool to either wind or unwind a recycled tape measure that has a coat hanger hook attached to it. An earlier prototype used a motor and a swing arm to set the hook, but that was too complicated, didn't work at the scrimmage and was replaced with the tape measure. The first tape measure worked at first then broke after extensive testing, so we are now using a new stronger tape measure.



The spool was 3d printed with the cover having a shark tooth design intended to force the robot to stay suspended in the air by using a distance sensor to sense when the robot was high enough to clear the ground, and then triggering a servo to send a rod into the way of the cover, which solved the problem with the motor not being strong enough to hold its position on it's own. This was the second prototype which had a larger spool for more mechanical advantage

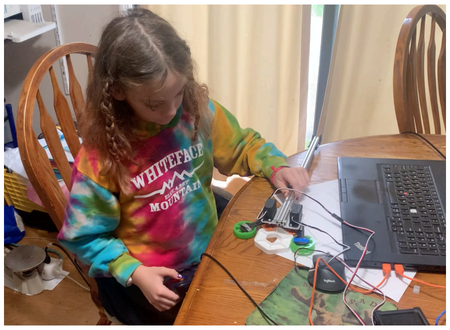


After we tried it, however, we didn't know when the brake fired, so we didn't know when to stop winching up which resulted in broken teeth, so we coded the winch to stop at the same time that the brake triggered. Later in the year, however, we discovered that we didn't need the brake because we coded the motor to resist motion. The first prototype was completely circular and would not catch on the brake, and the second had too many small teeth so it wouldn't catch on the brake either. The third was in the goldilocks zone.



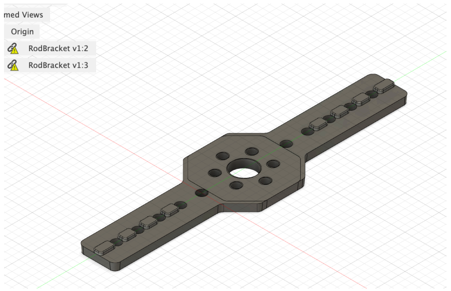
This shows the robot hanging, but the battery had to be moved as a counter balance.
When the robot is balanced, it doesn't have to go as high.

Gripper/Arm (Kayla Kim 2/5/24)



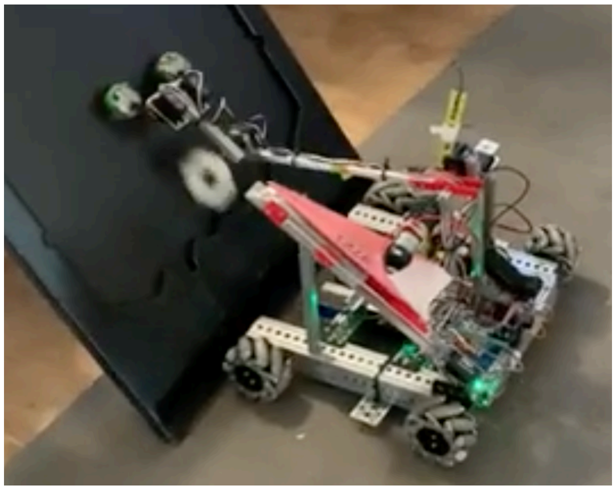
To design the gripper/arm, designs from previous designs were used as an initial base, which had 3 5:1 planetary gears on an ultra planetary motor. Turning the robot around to drop off pixels was inconvenient, so the gripper/arm was designed with an axle in the middle of the robot so the arm could flip around instead.

The main arm is made of a 15mm extrusion that is connected to the axle using Tetrix motor hubs. The team chose these hubs over REV hubs because of their larger set screw.



Custom brackets were designed using Fusion 360 to connect the arm to the axle hubs to keep the axles in place. The design was easy to create by copying and rotating a REV step drawing and merging it with the original. The brackets have two sides, allowing elastics to be attached to the back, providing support for the arm throughout its range of motion. The elastics were a new addition to the design this year. The brackets broke when testing a lift of the swinging door. The brackets were re-designed and made to be thicker and therefore stronger.

The arm was cut to size with a miter saw so that it could reach forward, inside the H shape at the front of the robot. When two REV servos (the grippers) were mounted to the top of the REV extrusion, they made a weird angle with the floor, which made it difficult to grab pixels. Another servo (the wrist) was added so the grabber could be rotated parallel with the floor, creating an angle that made it more effective for grabbing pixels. The grabber uses two andy mark friction wheels to hold pixels in place.



The moving wrist was added after a previous prototype failed at the scrimmage.

The grabber arm has an algorithm for operation.

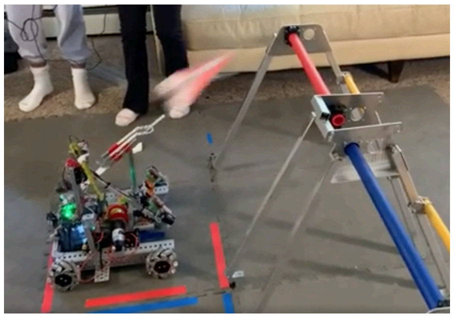
1. position robot
2. flip wrist down
3. grab pixel
4. flip wrist back
5. drive to placement
6. lift arm
7. flip wrist
8. drop pixel
9. flip wrist back

Repeat

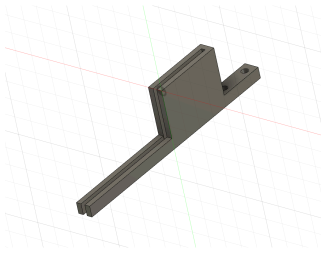
After trying different methods, this is the algorithm that worked best for the team.

Verbal communication between the driver and arm operator is essential to our team, consisting of speaking to one another positively and continuously, as a means to provide feedback and make adjustments accordingly. Driving the robot under pressure and stress necessitates compliments and support in order to maintain a positive attitude and have a good time.

Drone Launcher (Sage Peterson 2/5/24)

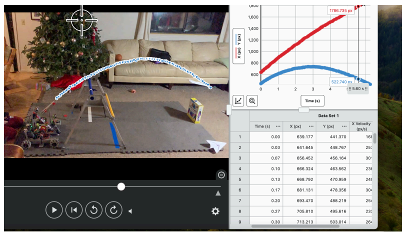


The design for the drone shooter was completely customized and fabricated by the team. We mounted the shooter to our robot by using a single post, which was mounted to a flat piece of aluminum that we cut with a miller saw and drilled with a drill press. We then took two REV extrusions to create the sides of the shooter and a single piece of aluminum was placed into the extrusion as a slide. The shooter is set at an angle that was made by bending a very old rev corner bracket. These brackets are no longer made as they were considered too weak.
The shooter is powered by elastics which are pulled back and released by a servo that is remotely triggered. When the shooter was originally designed 50% of the time the plane had not flown correctly.



We fixed this by designing a 3d printed cradle which holds the plane safely during the operation of the trigger. With the current drone shooter the plane flies from the second or third tile and easily passes the playing field.

The current design of the drone shooter has been modified several times and began with a spinning ball shooter which failed as it was too uncontrollable and violent. After we redesigned the drone shooter, the planes still had a 20% chance of crashing into the support structure, so we designed the cradle. This is the third prototype of the cradle. The first prototype had one hole for a screw, and was too brittle, while the second prototype had two screws, but was just a rectangle with a slit, like the first one. The third had two prongs sticking out to guide the cradle into the bars.

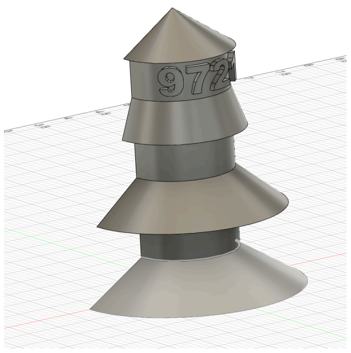


Data Set 1

	Time (s)	X (px)	Y (px)	X Velocity (px/s)
1	0.00	639.177	441.370	168
2	0.03	641.645	448.767	257
3	0.07	656.452	456.164	301
4	0.10	666.324	463.562	236
5	0.13	668.792	470.959	249
6	0.17	681.131	478.356	304
7	0.20	693.470	488.219	254
8	0.27	705.810	495.616	233
9	0.30	713.213	503.014	264

The design of the drone shooter allows the plane to fly instead of following a simple parabolic trajectory. This is proved by the asymmetrical flight that the plane takes:

Machine Learning(Yorda 2/4/24)



The team's object was designed to have radial symmetry and to have a tree-like shape. The team's object was also specifically designed with dimensions that correspond to our team's number, 9721. We created this object in the red, and in the color blue.

To create a dataset, we took multiple videos under different conditions so the robot would be able to detect the object, no matter the circumstances. These different conditions include;

- The prop placed at 5 positions on each line (start, halfway to half, half, halfway to the end, and the end)
- At each position, there are 4 rotations (front, left, back, right)
    - Images with or without a prop in different lighting conditions

To capture video, we mounted the camera on the robot and plugged in an HDMI cord into a laptop. The placement of the camera on our robot was influenced by the placement of the chassis, the winch, gripper/arm, and the drone shooter.

We took multiple videos of the prop while the robot strafed in different directions, and would cut the video when the prop moved out of frame. We took multiple videos of the robot strafing under the previously stated circumstances such as prop position, prop rotation, and lighting.

We uploaded these videos to **ftc-ml**. We individually uploaded each video and using the machine learning tool, we added labels to a specific set of data. We named our prop "AK", and highlighted the object with an added box around the prop. Using *FIRST* machine learning, we went through multiple frames, making sure our designated prop was highlighted with an "AK" box, and the robot was properly identifying the prop.

Once we successfully did the red prop, we applied the same principles when doing the blue prop. We decided to train the red and blue model individually, due to the event that the model training either succeeded or did not go well, so we could apply or revise our approach for the blue model.

- Collected Video
- Uploaded
- Added Box around the prop
- Data Set
- Built Model
- Downloaded Model

Programming(Cailin 2/7/24)

The general process for programming is to develop code to operate a subsystem in isolation using onBot java. We chose this because it was the simplest, also because we had used it before, and it is similar to AP computer science A.

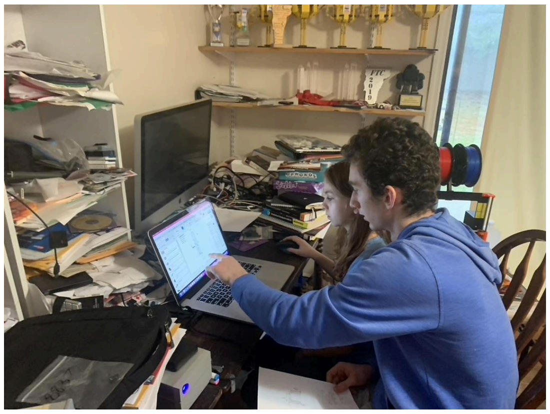
This makes it easy to test the code with as few potential issues as possible. When the sub-system is working correctly, the code is integrated into the code of the main opMode at the same time the systems are integrated. We add comments describing the date, person, and purpose of the code.

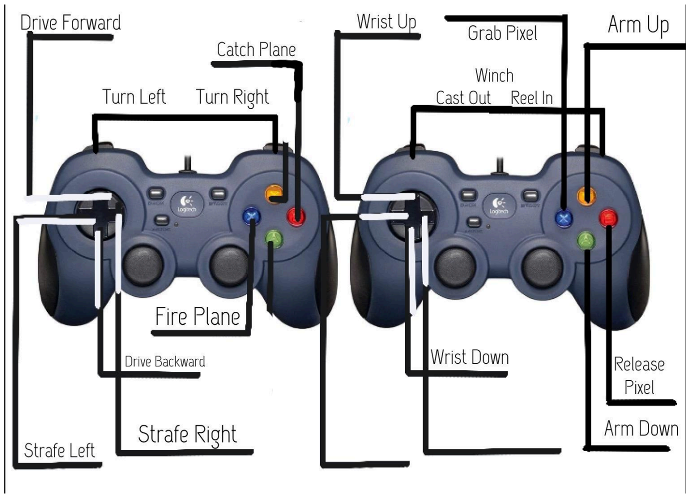
The team attempts to match human intuition with the operation of the robot so that when people are stressed, they can still operate the robot effectively. During testing, we modify the mapping of programming to the controller and tune parameters like the motor power.

When possible, code is organized into methods and re-used. This robot uses the following methods:

Name	Purpose
driveAxialForward(double power)	Used in teleop to drive forward
driveAxialBackwards(double power)	Used in teleop to drive backward
driveRotateLeft(double power)	Used in teleop to rotate left
driveRotateRight(double power)	Used in teleop to rotate right
driveStrafeLeft(double power)	Used in teleop to strafe left
driveStrafeRight(double power)	Used in teleop to strafe right
resetMotors()	Resets Motor encoders
absAverageChassisEncoder()	Finds the average of the absolute value of the motor encoders
strafeLeftIMU(int distance, double power)	Strafes left with IMU
strafeRightIMU(int distance, double power)	Strafes Right with IMU
driveAllStop()	Stops all driving motors
	]
driveRobot(double power)	Drives the robot
Autonomous Methods	
winchTrigger()	Triggers the winch brake
int convertInchesClicks(double targetDistance)	Converts Inches to Clicks
driveEncoderForward(double distance, double power)	Drives forward by tracking clicks
driveEncoderBackwards(double distance, double power)	Drives Backward by tracking clicks
initTFOD()	Prepares Tensor Flow Object Detection
telemetryTFOD()	Describes TFOD
string getPropPosition() //	Finds Prop

The robot uses (4) autonomous opModes and a single teleOp. The teleOp was developed first, because we wanted to be able to manually test everything before automating it. Once we had a system working well in teleOp, we would convert to autonomous as needed. The four autonomous methods align with the starting positions in each of the two alliances. We only park in one of the two positions in each alliance because driving under the truss in autonomous risks losing the drone, which is our highest scoring element.





This table shows how the robot makes corrections while strafing.
The robot is proportional to both the power and error, this is important because if the robot goes faster it will cover more incorrect ground.

Power	Tuning Parameter	Angle to Correct	Adjustment	FR	BR
0.1	0.001666666667	-22	-0.03666666667	0.1366666667	0.06333333333
0.2	0.003333333333	-22	-0.07333333333	0.2733333333	0.1266666667
0.3	0.005	-22	-0.11	0.41	0.19
0.6	0.01	-22	-0.22	0.82	0.38

```
public void strafeLeftIMU(int distance, double power)
{
        double tuning_param = power/60.0;
        angles = imu.getAngularOrientation(AxesReference.INTRINSIC, AxesOrder.ZYX,
AngleUnit.DEGREES);
        String initialHeading = formatAngle(angles.angleUnit, angles.firstAngle);
        double initial_heading = Double.parseDouble(initialHeading);
        while (elapsedClicks < clicks)
        {
        elapsedClicks=absAverageChassisEncoder();
        angles = imu.getAngularOrientation(AxesReference.INTRINSIC, AxesOrder.ZYX,
AngleUnit.DEGREES);
        String currentHeading = formatAngle(angles.angleUnit, angles.firstAngle);
        double current_heading = Double.parseDouble(currentHeading);
        telemetry.addData("Current heading", current_heading);
        double power_adjustment = tuning_param * (current_heading-initial_heading);
        telemetry.addData("Power Adjustment", power_adjustment);
        telemetry.addData("Elapsed CLicks", elapsedClicks);
        driveFR.setPower(power - power_adjustment);
        driveBR.setPower(power + power_adjustment);
        driveFL.setPower(power);
        driveBL.setPower(power);
        telemetry.update();
        }
      driveAllStop();
}
```