Robotics Engineering Notebook

team name: When Robots Fly

team number: 1618 A

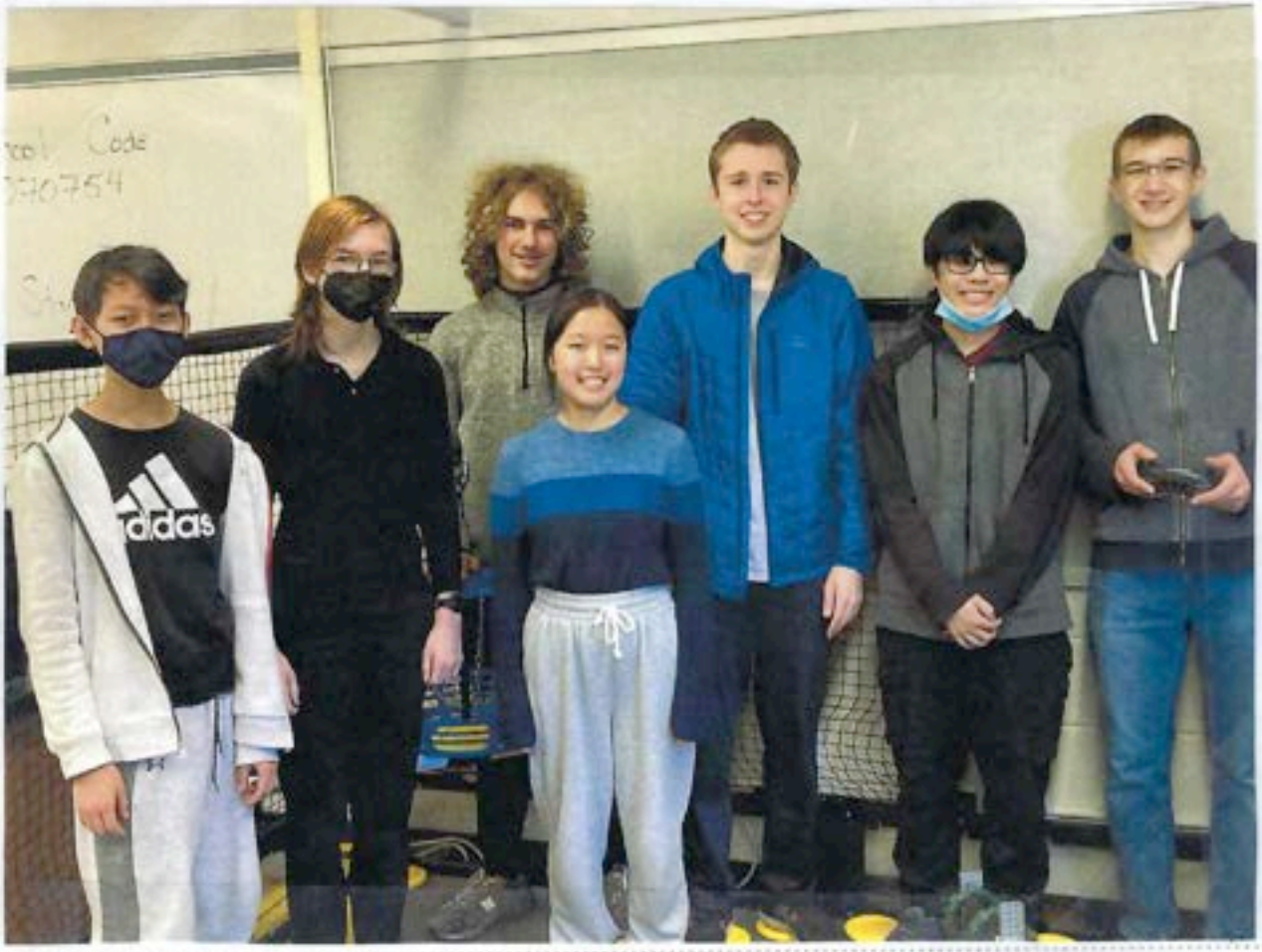
season: "Spin Up"

start date: 9/6/22

end date:

book number: 1 of:

Team Photo



Team Profile

We are 1618A, the robotics team at E.O Smith High School, located in Storrs, CT. Our team is currently made up of students from all grades 9-12. We have an open door policy during robotics meetings - anyone who's interested can come by to see what we're up to and help out if they wish. We also have a permanent group of members who show up to almost every meeting. These people make up our team: Sam, who mostly builds and designs; Riley, who mostly programs; Todd, who mostly designs and builds; Rain, who builds; Ian, who builds and drives; Sungjin, who builds; Anayi, who designs; Shaun, who builds; and Lucy, who designs and records in the engineering notebook. Each meeting brings different challenges and opportunities, so we're all very flexible in our roles and help each other out on what needs to get done.

My Projects

page	project	date
1	Brainstorming and organization (Robot design #1)	9/12/22
2	Brainstorming Robot designs #2 + #3	9/12/22
3	Brainstorming Robot Designs #4	9/12/22
4	Point priorities and Festival preparation	9/13/22
5	Roller designs	9/26/22
6	Festival reflection	9/26/22
7	Disk ~~Roller~~ Launcher designs + Roller prototype	9/27/22
8	Curved launcher design	10/3/22
9	Roller improvements	10/10/22
10	Mecanum wheels	10/18/22
11	Disk Intake	10/31/22
14	Disk launcher motor problem	11/5/22
15-16	Disk launcher redesign	11/21/22 - 11/28
17-18	Disk Intake modification	12/5/22
19-~~23~~	Disk Intake simplification	12/6/22
20-21	Robot construction	12/10/22
22-26	Code 1	12/12/22
27	Disk Intake Trouble shooting	12/13/22
28-30	QR Handbook	12/16/22
31	Tank Drive	12/17/22
32	MOSUK Tournament Reflection	12/19/22
33	Extender Design	1/3/23
34	Rail Placement Correction	1/7/23
35	Extender Attachment	1/7/23
36-38	Autonomous selection code	1/9/23
39	Pre-Tournament Testing	1/10/23

My Projects

page	project	date

9/12/22 Today we watched the official video on the rules of this year's competition, as well as reviewed the rules book. Then we had all members start brainstorming and designing initial ideas for our robot. We also had to set up our field, making sure all the pieces come together so that we will be able to use the field in the future to test our robot with.

Our town is having a local festival soon, and our team has a stall there to introduce our robots team to our community. We do this every year to hopefully inspire new people to have the same love for robotics that we do.

Design #1

This robot prioritizes scoring disks, it can grab discs and move them up a conveyor belt to score, with a motor at the top to launch the disk. -LL

project Brainstorming and organization Robot Design #1 designed by: Rain Zhang witnessed by: [signature] date: 9/12/22

9/12/22 (continued)

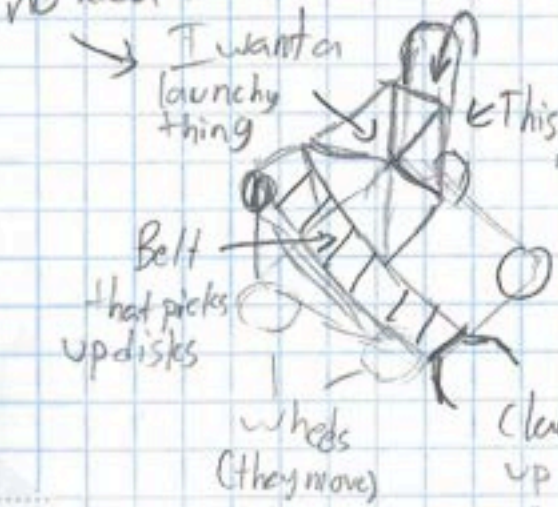
Design #2

Anayi

This is another conveyor belt design. From the video she saw that disks need to be carried from point A to point B, and thought the best way to do that was through a conveyor belt design. -LL

Design #3

This design also prioritizes scoring disks, with a claw to flip disks onto the conveyor belt, which may be tall enough to have the disks fall over the edge and into the goal. There would also be a structure on the side that would spin rollers. -LL



shaun

project Brainstorming Robot Designs #2 + #3 designed by: Anayi, Shaun witnessed by: Kayla

date: 9/12/22

9/12/22 Continued

Design #4

Todd's design includes a conveyor belt that would have a fly wheel at the top to launch disks into the goal.

Todd

There would also be a chain on the side to be extended in the last 10 seconds to get points for area covered. The arm in the front would turn the rollers.

After examing the roller part of the field, however, the roller proved much more difficult to turn than previously anticipated. The arm component will have to be carefully thought out. - LL

project Brainstorming Robot Designs #4 designed by: Todd Buch witnessed by: [signature]

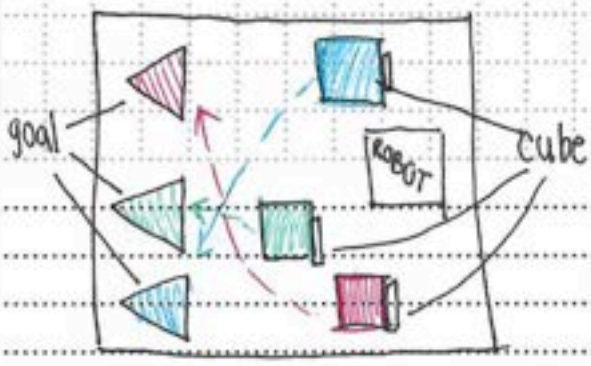
date: 9/12/22

9/13/22 Today we rewatched the official competition video. Then we created a spreadsheet to organize everyone's roles and brainstormed different ways to score points as well as estimated about how many points we thought we'd be able to score. We also spoke about our priorities and thought of possible strategies. We continued building the field and discussed the upcoming festival, deciding to introduce our robot by engaging our audience with a maze. We started work on the maze, planning and building its layout. - MS

ways to score points	Points per unit	※ units	Max Points	Realistic units	Realistic Points	Priority
scoring disks in basket	5	60	300	8	40	1
roller	10	4	40	2	20	3
area at end	3	36	108	7	21	2

We finished building a simple claw bot that we will let passerbys at the festival drive through a maze. There are always a lot of kids from the local elementary and middle schools who will hopefully take an interest in robotics, and our booth there might encourage that.

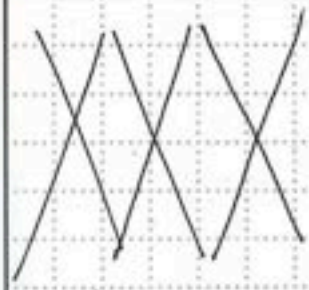
The game design will be a field using cubes of 3 different colors that will need to be moved into 3 goals. -LL

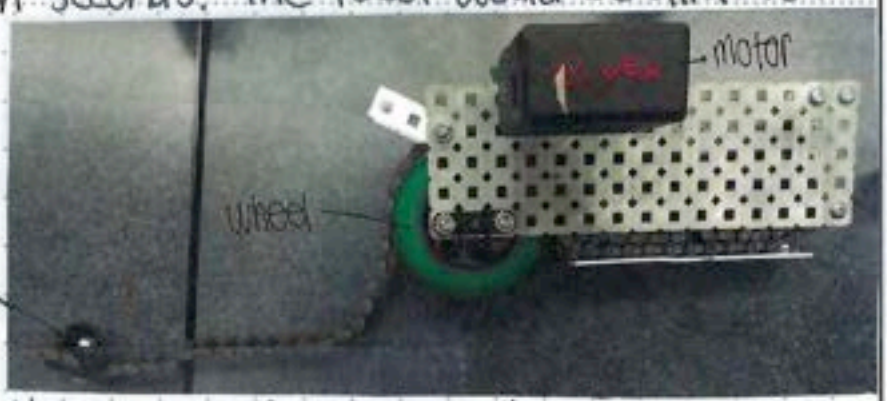


project Point Priorities and Festival preparation designed by: Miriam Shamshoh witnessed by: Riley

date: 9/13/22

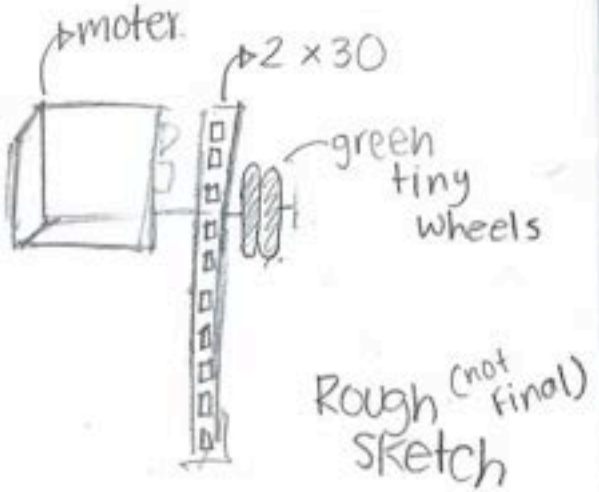
~~9/19/22~~ 9/19/22 we come up with an idea to eject a chain as a way to increase our robot's area at the last ten seconds. The robot would be attached to one end. -LL





9/26/22 We experimented with a design for a roller, to change the roller color. Our team has split into separate groups to design 1. scoring disks, 2. expanding to cover more area, and 3. changing the roller color. Our first design was to attach some mesh to a 4X13 piece of metal, hoping that the mesh would create enough friction to move the roller, which is a bit difficult to move.

Another design is to attach a motor and 2.75" diameter smooth wheels to a long bar of metal and use the wheels to turn the roller. We attempted to use an 84 tooth gear, but it required too much precision and was unreliable even when we manually held the design to try and turn the roller. -LL



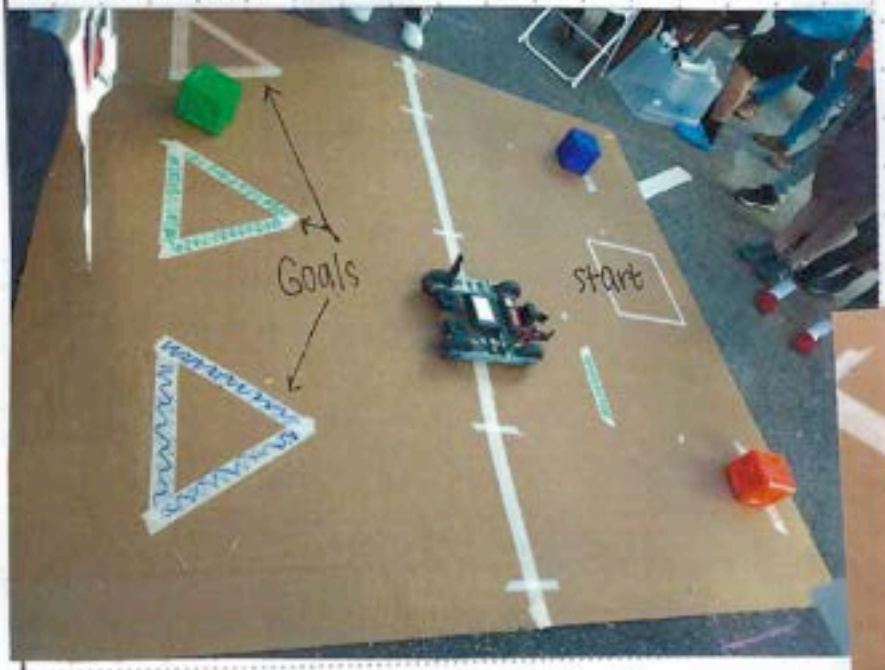
Rough (not final) sketch

project Roller designs designed by: witnessed by:

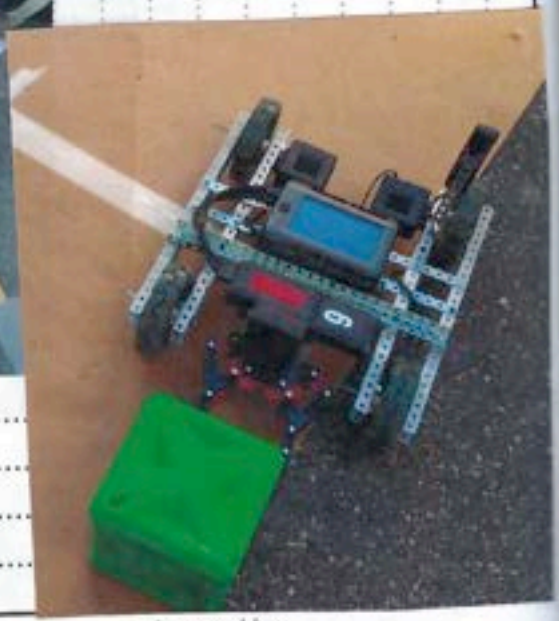
date: 9/26/22

9/26/22 Our experience at the community festival was a success. We set up a cardboard field with 3 different colored cubes and 3 goals. We invited people to drive a claw bot we put together for the festival. We had people of all ages attempt our challenge, from 4 years old to 54. It was an amazing experience being able to share our love of robotics with our community. We met quite a few middle schoolers who were eager and interested in robotics, which makes us hopeful that there will continue to be people who will want to participate in the future.

Next up we have another event to prepare for, a small school club festival. We're planning to do the same thing for this festival as we did with our community. It's important to us that we do outreach and spread our interest in robotics with other people. Hopefully we'll be able to foster a stronger connection between our robotics team and school. -LL



People were really invested in completing the challenge, whether it took 30 sec or 7 minutes! -LL

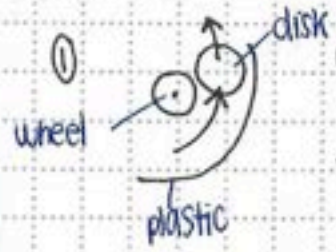


Even a simple claw bot is able to inspire awe and interest as some people drive a robot for the first time. -LL

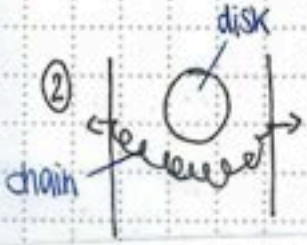
project Festival Reflection designed by: Lucy Liu witnessed by:

date:

9/27/22 we have 2 ideas for a disk launcher.

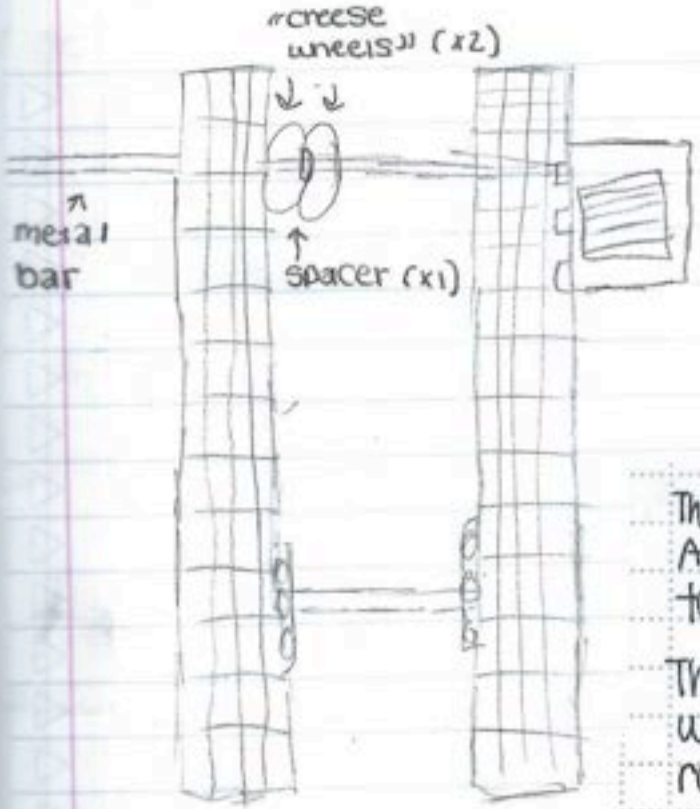


Riley is working on a design with a curved piece of plastic on the outside, and a wheel in the middle that will spin to launch the disk. -LL



Todd's design is to have a chain pull tight to launch a disk. -LL

ROLLER prototype



Rain
Kalyani
Shaun
Anayi

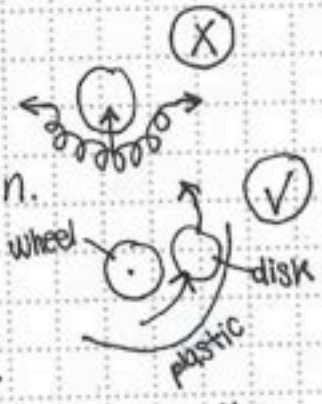
The roller design was revised. A gear and 2 wheels were removed to focus on power, not speed.

This design was tested, but did not work since when attached to the robot, the design was above the roller itself; no contact between roller and wheels. -LL

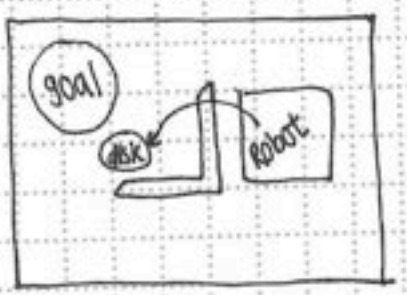
project Disk launcher designs + Roller prototype designed by: Riley, Todd, Rain, Kalyani, Shaun, Anayi witnessed by: Lucy Liu date: 9/27/22

10/3/22 We built the base today, making its dimensions as large as possible considering our robot can score more points for the number of tiles it can cover at the end. It's dimensions are 18 x 18 inches. -LL

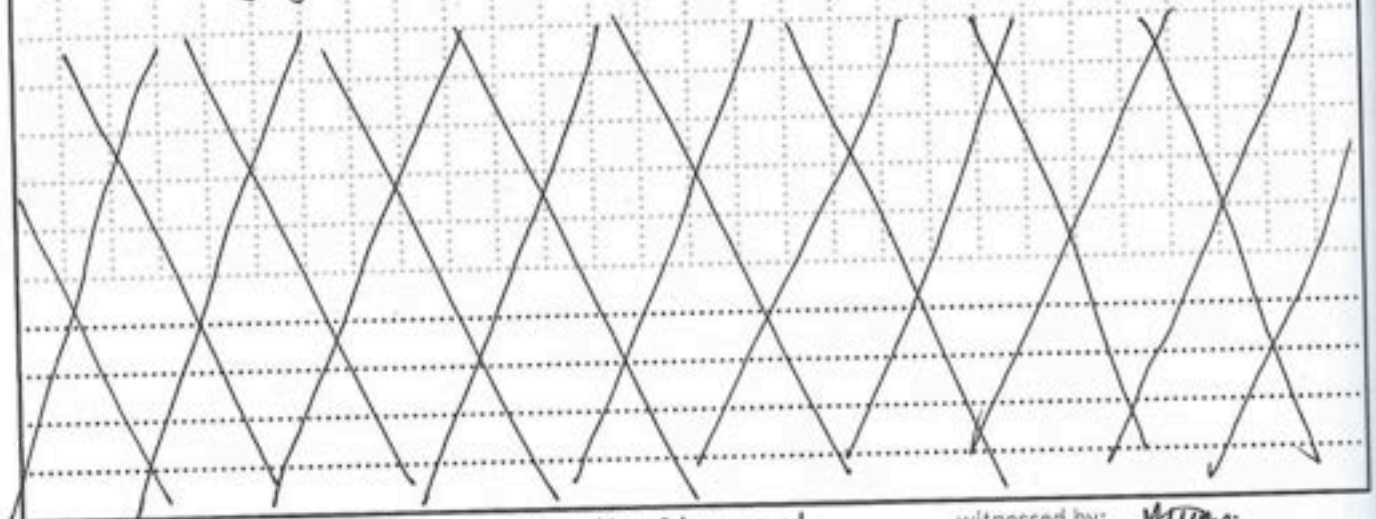
We are no longer considering the flinger design as it doesn't seem to be very effective or efficient. Instead, we'll focus on the curved design.



We tested out the curved launcher design, but the wheel didn't spin fast enough to launch the disk far enough, to where we wanted it to be. To solve this problem, we will try to increase the gear ratio by attaching a 64 tooth gear and a 36 tooth gear to get our wheel to spin with greater speed. -LL



10/4/22 We tried out a pneumatic disk launcher. It worked, however only managing to score into the goal at an extremely close distance. There was also not enough pressure to be able to shoot many goals. -LL



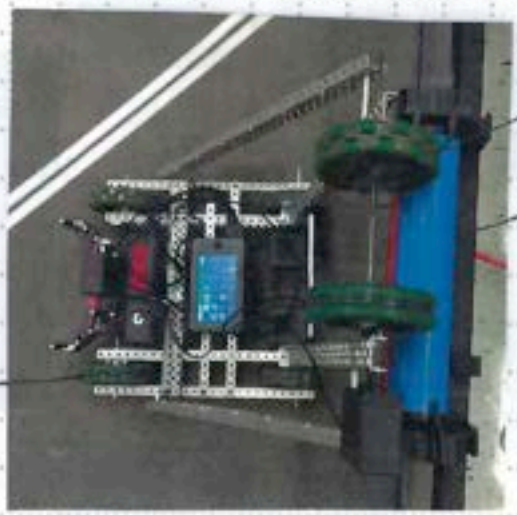
project Curved Launcher design designed by: Riley, Todd witnessed by: [signature]

date: 10/4/22

10/10/22 We've created an effective and working roller turner. It can turn the roller precisely and fast. The only concern is its size and how it'll fit on the robot and the launcher, but we'll need to finish the launcher and then re-evaluate. -LL

Side view

TOP view



We replaced the smooth green wheels with omni wheels to increase the reach and friction to be able to turn the roller more efficiently. The roller turner was also put at a tilt and wheels were positioned above the roller to better able turn it, as putting the wheels directly in front of the roller proved to have more difficulty when testing it. -LL

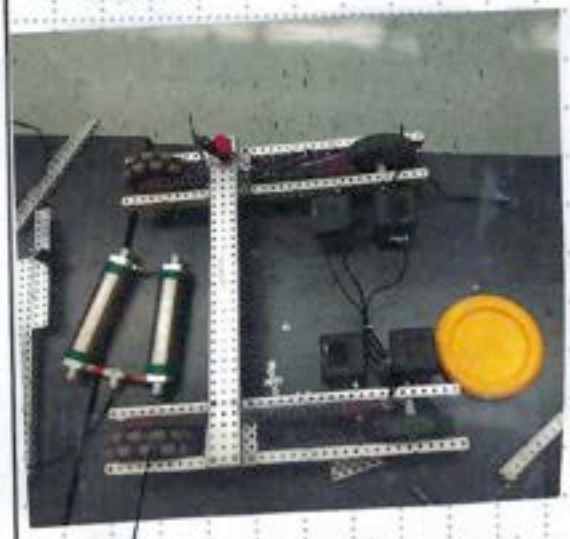
project Roller improvements designed by: Kalyani witnessed by: Lucy Li

date: 10/10/22

10/17/22 we've decided to use 4 mecanum wheels on our base. This will ~~else~~ allow for better positioning as our robot will be able to move sideways. Better positioning is essential when trying to shoot disks into the goals. Riley is working on coding the wheels. -LL

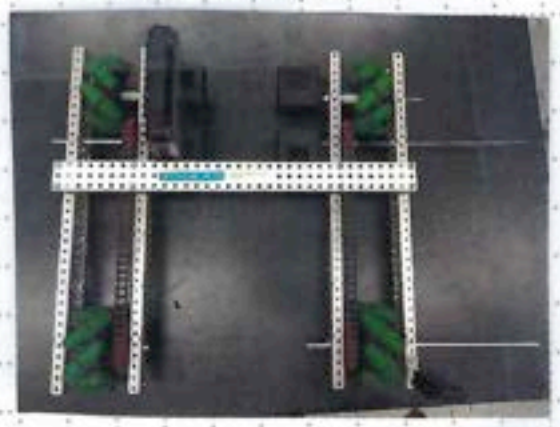
10/18/22 Today we attached the mecanum wheels to our base. It took much longer than expected since the wheels are new and we needed to insert axels through them. -LL

BEFORE

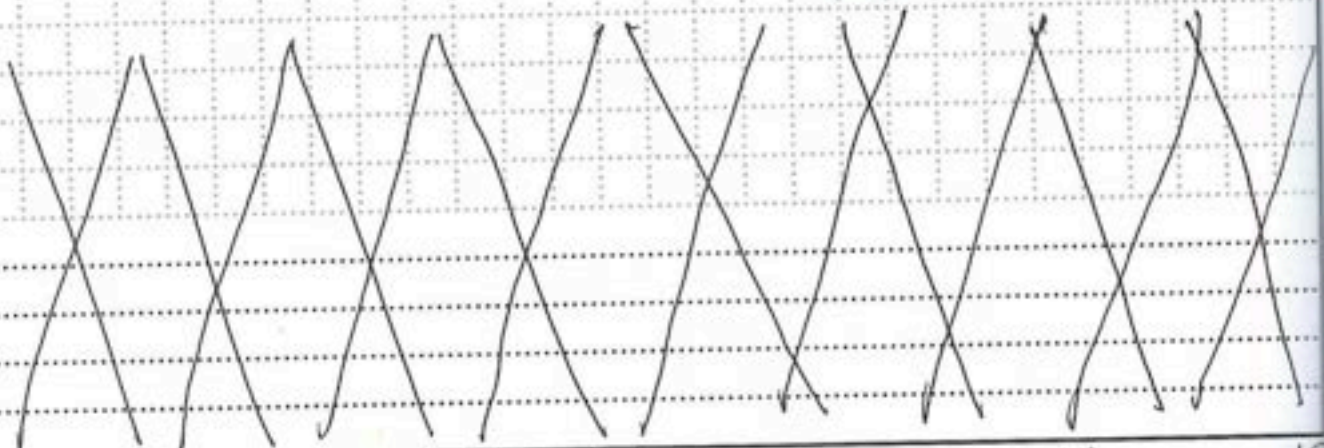


omni wheels

AFTER (mecanum wheels)



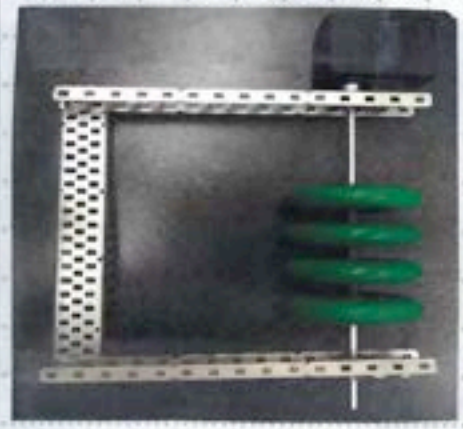
Front view ↓



project Mecanum wheels designed by: Riley witnessed by: Lucy Liu

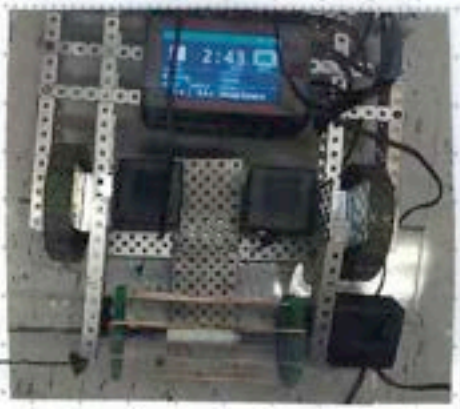
date: 10/18/22

10/25/22 We worked more on testing and constructing disk pickup/intakes today. Sungjin had a design with 4 smooth wheels connected through by a long axle with a motor attached on the end. Rain had a design where he had 2 24 tooth gears with 4 rubber bands pulled about 6 inches apart. Rain's design relies on the friction between the disk and rubber bands, while Sungjin's design relies on the friction between the wheels and disk. -LL



Sungjin's design

Rain's design



10/31/22 Todd worked on the launcher, and was able to make the launcher launch the disk fast enough to make it into the goal. Rain also continued working on his disk intake design and added a curved metal section for the disks to move up. However, after testing his design, he discovered that the disks would get stuck and couldn't be moved up. -LL

TOP VIEW

gears connected by chain

rubberbands

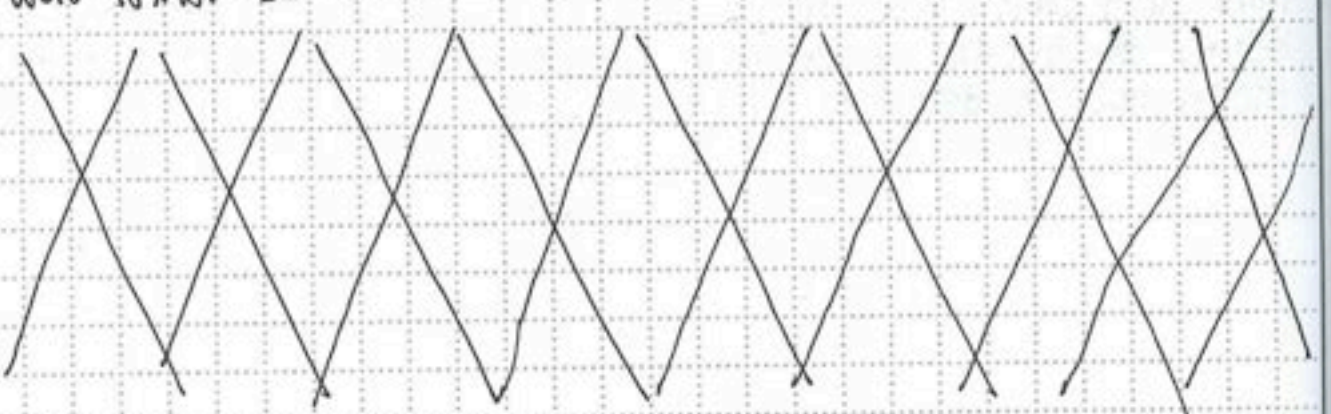
motor

SIDE VIEW

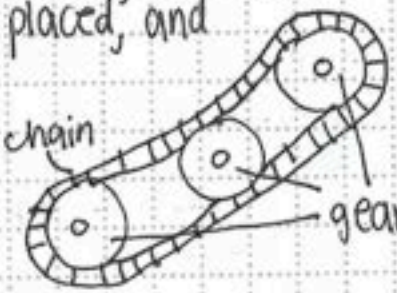
project disk intake designed by: Rain witnessed by: Lucy, Lil

date: 10/31/22

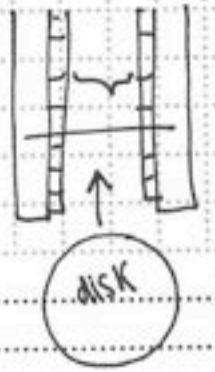
11/1 Rain decided to detatch part of the disk intake to bend the end of the curved metal ramp to allow for disks to be picked up. Riley worked on modifying the base to make sure the dimensions were 18 x 18. -LL



11/7 Rain attempted to connect the 3 gears together using a chain, but it wasn't working since the gears were not evenly placed, and the chain was too loose. we shortened the chain and it worked! Unfortunately, when we tested our design, we realized that the rubber bands did not create enough friction to pull the disk up. -LL



Shaun and Rain decided to try a new design, using tread chains to support the disk instead of rubber bands. But after only a few minutes we suddenly thought of the problem of how we'd get the disk into the tread chain system. -LL



The disk would not be able to enter, since the spacing inside/between them had to be exact to the disk. -LL

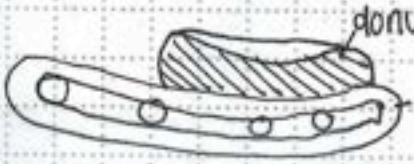
side view →

Top view →

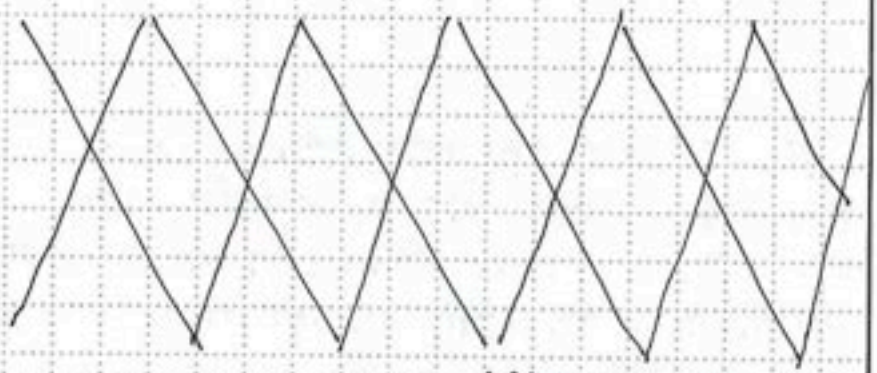
project Disk intake designed by: Rain + Shaun witnessed by: Lucy Liu

date: 11/7/22

11/7 (continued) I suddenly thought of a water ride of a donut moved along a path in a river by a conveyor belt. I suggested we instead place a conveyor belt in the middle of our metal structure. -LL



←Tread chain design, requires a little too much precision in terms of getting disks in the correct position in between. -LL

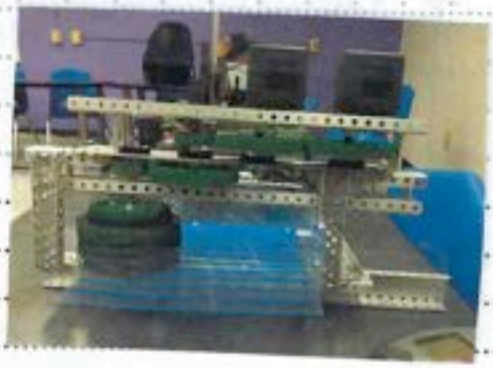


11/8 We worked more on our disk launcher design, increasing the gear ratio to increase the speed in order to launch the disks far enough. We calculated the speed at which the fly wheel spins. -LL

$$600\ \text{rpm} \cdot \underbrace{\left(\frac{84}{36}\right)^2}_{\text{gear ratio}} = \boxed{1900\ \text{rpm}} = 31.7\ \text{rot/s}$$

BEFORE

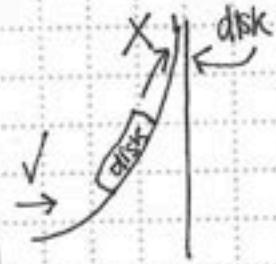
AFTER



project Disk Intake designed by: Rain + Shaun + Sungjin witnessed by: Lucy Liu date: 11/7/22

11/14 We encountered a problem with our disk launcher. One of the motors kept overheating, and the wheel was not getting up to speed, when it had previously been fine. We suspect that it's a problem with the overheating motor as the other motor is fine. We tested using a different motor, and it worked fine, however we need a turbo motor to replace the faulty turbo motor, which we don't have. -LL We also changed some of the gear locations to make sure the 2 motors are more evenly stressed. -LL

11/15 To solve the motor problem, we're switching the gearbox from the faulty motor into a working motor.
We noticed a new problem in the disk intake design as well. The dimensions at the bottom are wide enough to allow the disk to pass in, but the width at the top was blocking the disk from leaving. We adjusted our design to widen so that we can accomodate the disk size. -LL



our current design.
The top needs to be widened.

Riley started working on controls for the fly wheel as Ian attached a new working motor back onto the disk launcher. -LL

project Disk Launcher motor problem

designed by: Ian + Riley

witnessed by:

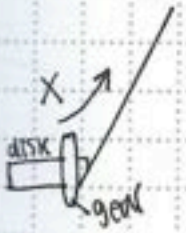
date: 11/5/22

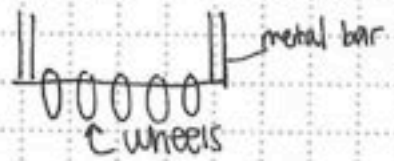
11/21 Unfortunately, even after attempting to fix the gear ratio, our launcher still cannot launch the disk effectively. Our design is already so bulky that increasing the disk ratio again wouldn't be worthwhile. With only 7 meetings left until ~~over~~ our tournament, we need to change our launcher design.

We will have 2 wheels on either side of the disk and one gear ratio.

We are also trying to figure out how to get the disks into our intake. Once they're inside, our design can smoothly transport them up, however it's getting them inside that's giving us some difficulty. We tried simply putting a thin bar with 2 gears on either side at the bottom, but the angle the gears make with the slope is too steep.

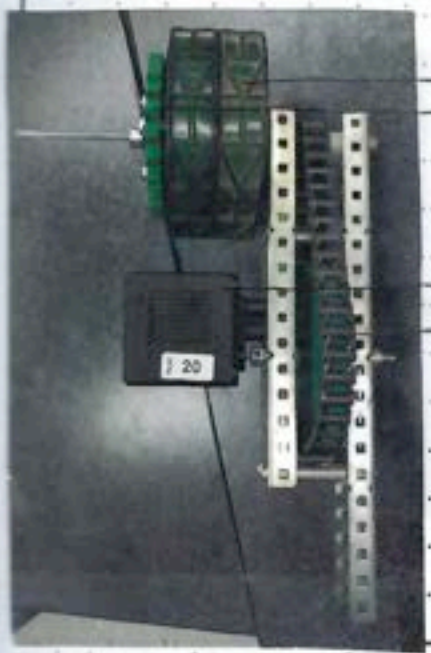
To make the disk entry more flexible, we plan on adding



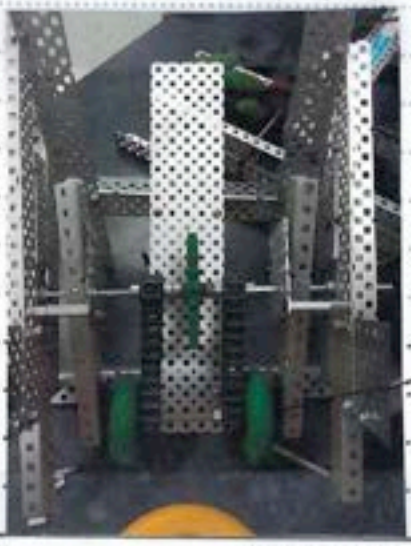


to the bottom that will be able to move slightly up and down to allow easier entry. -LL

1/2 of Launcher↓



Disk intake↓



project Disk Launcher Redesign designed by: Riley witnessed by: Lyay Liu

date: 11/21/22

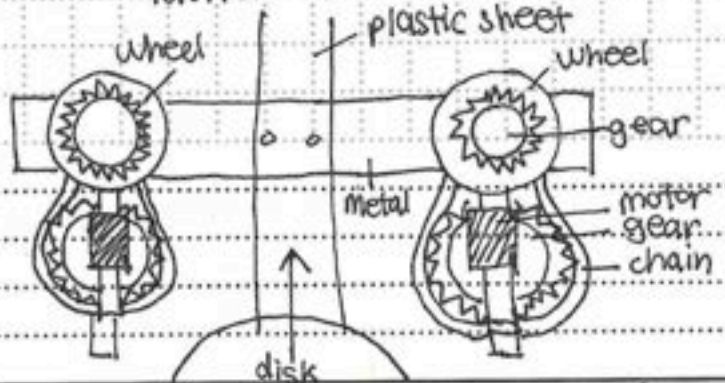
11/22 We modified our roller design to have it fit on the side of our robot, since previously it would have taken up the entire front or back of the robot and we want to conserve space to fit the intake and launcher. The design is now simpler, 2 metal bars on either side of 2 omni-wheels used to spin the roller. We tested this design and were successful.

Right now our priorities are finishing the disk intake and launcher, then successfully attaching them onto the base. -LL

FRONT VIEW

SIDE VIEW

11/28 we're almost done constructing the launcher. The disk should go in between the 2 wheeled disk wheels and be shot out on the thin plastic sheet. The disk seems to not have enough friction with the wheels to be readily shot out. -LL



project Disk Launcher Redesign designed by: Riley witnessed by:

date: 11/28/22

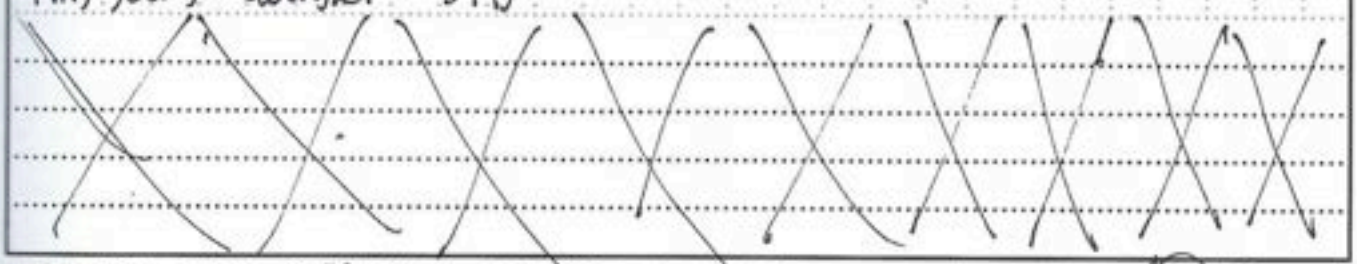
11/29 We added some mesh and rubber bands to the wheels to help the disk be shot out of our launcher.

12/05 We made some changes to the intake today. The main problem with the intake had been that there wasn't enough friction between the disc and the vanguard section of the intake, meaning discs were not getting a significant enough initial push to get them into the intake. If a disc was given a little nudge, the intake could do the rest — we needed more grip at the very beginning.

The solution for this came from something we'd built already: the flywheel. We knew that the launcher had sufficient grip on discs, so we took what we had in the launcher — two wheels, stacked, wrapped in mesh, the mesh being secured with rubber bands — and put it at the vanguard of the intake. We geared this vanguard wheel such that it spins four times as fast as the rest of the intake. This is working very well — the intake can pick up discs fairly easily.

We also gave the launcher its first proper test when it was firing at the goals from ground level, and it performed far better than expected. We were able to score goals from behind the autonomous line and were even able to hit above the goal (this is not due to inaccuracy; we hadn't attatched the launcher and so were holding it ourselves). However, the power was extremely inconsistent with some shots going halfway across the field and some only going a few inches. We added another layer of mesh on top of the first, theorizing that the thicker sections of mesh were responsible for the successful shots. This improved the situation, but not 100%. Tomorrow we will need to investigate this further and increase consistency.

Additionally, we don't currently have a way of connecting the intake and launcher to each other and the robot, and the intake is a bit too big to fit on the robot at the moment. We have three in-school meetings and a six-hour Saturday session to fix these issues and make autonomous. This seems doable. — SPD

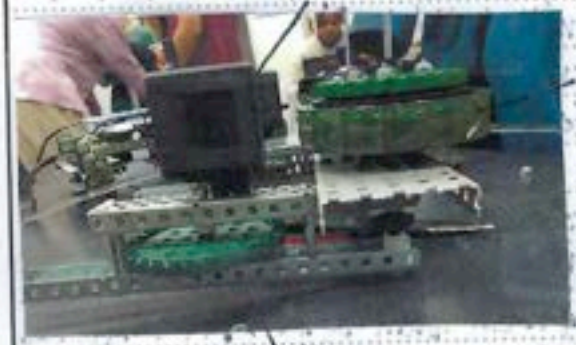


project Disk Intake Modification designed by: Sam P-D witnessed by: [signature]

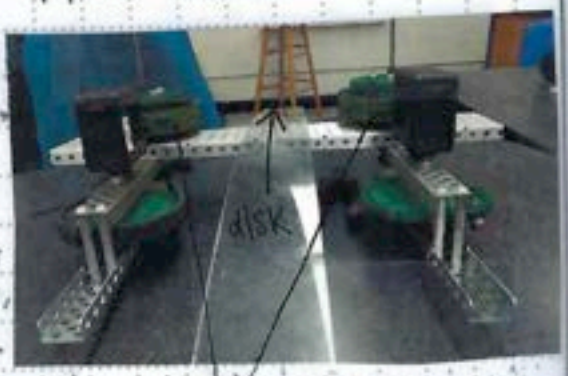
date: 12/5/22

12/05 (continued) Launcher:

side view ↓

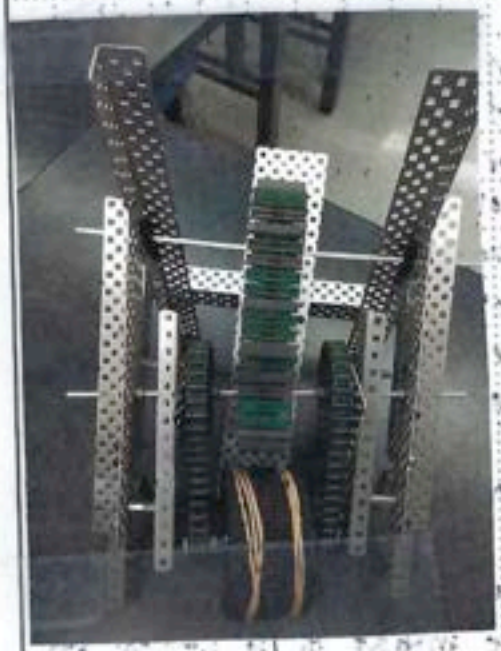


Front view ↓



Front view ↓

INTAKE



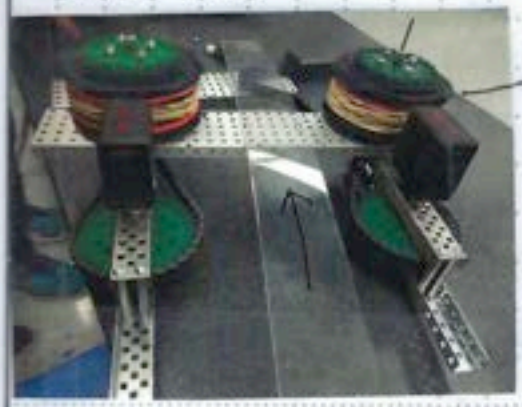
side view ↓

project DISK Intake Modification designed by: Sam P-D witnessed by:

date: 12/05/22

12/6 We added mesh and rubber bands to the wheels of the launcher for increased friction.

Front view



wheel with mesh + rubber band

Side view

We tested the launcher and the disk was able to be launched more effectively and smoothly.

We then examined the disk intake system and discovered the outer metal structure to still be too constrictive, and it didn't fit well on the base. The intake was also not designed with the launcher in mind, so we found it hard to figure out how to attach the 2 separate parts. We decided that instead of trying to attach 2 separate parts onto the robot, we would try to first attach the launcher and intake together and then attach them to the robot.

To do this, we had to remove a lot of the outer metal structure to fit and connect the launcher together. Next meeting we'll try and connect the launcher and intake together and attach it to the robot- LL

simplified ~~not~~ disk intake:

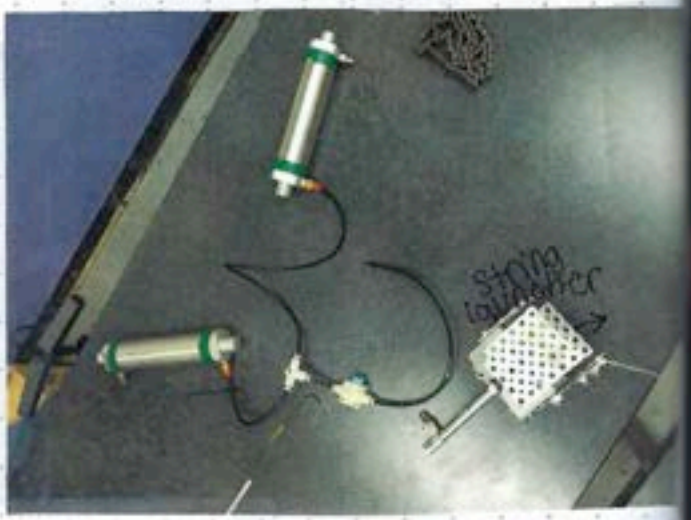
project Disk Intake ~~Redesign~~ Simplification designed by: Sam RD witnessed by: [signature]

date: 12/6/22

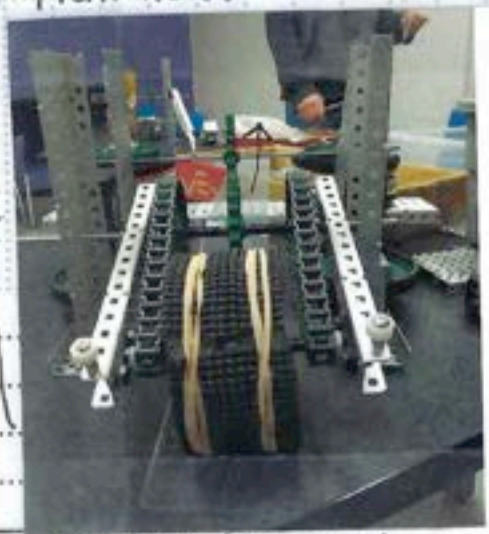
12/10/22 After removing the frame to our intake, we were able to attach it to our intake smoothly. by adding metal guards at the sides of the launcher.
we also worked on an extender design of using pneumatics to launch a string with a weight at the end, so we can get more points from having contact with more floor tiles in the last 10 seconds.
we changed the wheels on the roller to omni wheels + mesh + rubberband for better grip on the roller.
our launcher is angled at about 45° and works great, able to shoot disks from half/ middle of the field. our intake is still having trouble. -LL

Roller turner ↓

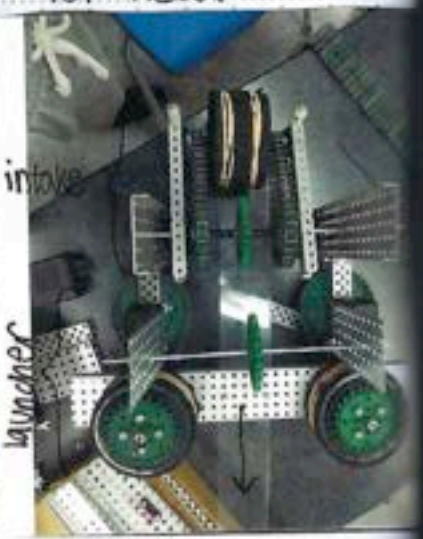
Extender using pneumatics ↓



Front view ↓



TOP view ↓

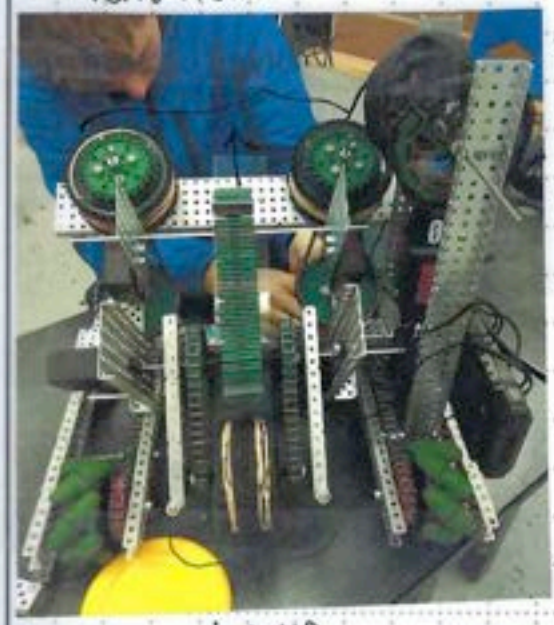


Connecting launcher and intake. The disk should travel along the plastic to the 2 spinning wheels of the launcher. → intake

project Robot construction designed by: Riley, Todd witnessed by:
date: 12/10/22

12/10/22 (continued)
We were able to attach our launcher + intake onto the robot.

Front view ↓

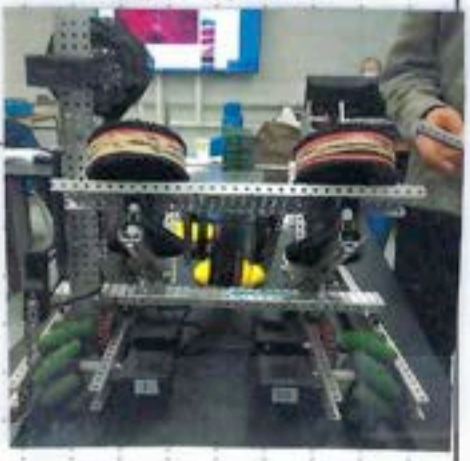


roller
turner

intake

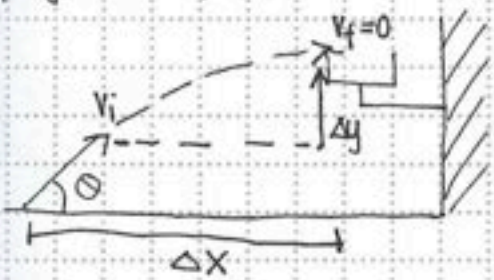
Back view ↓

launcher



Next meeting we're planning to attach the extender and pneumatic system underneath the launcher, as well as try to figure out how to get the intake working smoothly. -LL

12/12/22 To find how fast the flywheel needs to spin given the distance between the robot and the goal, we used a 2D physics projectile equation.



$\theta = 30°$ $\Delta y = 0.45m$

$V_x = \frac{\Delta x}{\Delta t}$ $V_{fy} = V_{iy} + 2a\Delta t$ $V_i = \sqrt{\frac{39.2\,\Delta x}{\cos 30}}$

$0 = V_i \sin\theta + 2a_y \Delta t$

$\Delta X = V_i \cos 30 \Delta t$ $0 = V_i \sin(30) + 2(-9.8)\Delta t$

$\Delta X = \frac{V_i^2 \cos 30}{39.2}$ $\frac{1}{2} V_i = 19.6\,\Delta t$

$\Delta t = V_i / 39.2$

project Robot Construction designed by: Riley, Todd witnessed by: Lucy, Lin

date: 12/10/22

12/12 (continued) Here is our code for the robot so far.

```
// ---- START VEXCODE CONFIGURED DEVICES ----
// Robot Configuration:
// [Name]               [Type]        [Port(s)]
// Drivetrain           drivetrain    1, 2, 9, 10, 20
// Controller1          controller
// Flywheel             motor_group   3, 4
// ---- END VEXCODE CONFIGURED DEVICES ----

#include "vex.h"

using namespace vex;
competition Competition;

//------------------------------------------------------------
// MISC. FUNCTIONS
//------------------------------------------------------------

/**
 * Starts spinning the flywheel given the specified
 * velocity percentage
 *
 * args:      pct (velocity percentage that the motors should spin at)
 * returns:   ---
 */
void startFlywheel(float pct) {
 Flywheel.setVelocity(pct, velocityUnits::pct);
 Flywheel.spinFor(1000, timeUnits::sec);
}
```

project Code 1 designed by: Riley F witnessed by:

date: 12/12/22

12/12 (continued)

```
/**
 * Stops the flywheel. BrakeType should be set to coast to
 * avoid damaging the flywheel motors
 *
 * args:       ---
 * returns:    ---
 */
void stopFlywheel() { Flywheel.stop(brakeType::coast); }

/**
 * Controls when the flywheel is turned on and turned off
 *
 * args:       ---
 * returns:    ---
 */
void flywheelControls() {
 if (Flywheel.isSpinning())
   stopFlywheel();
 else
   startFlywheel(100);
}

/**
 * Moves the roller forward when the up button is pressed
 * Moves the roller backward when the down button is pressed
 *
 * args:       pct (the velocity percentage that the roller will run at)
 * returns:    ---
 */
void moveRoller(float pct) {
 RollerMotor.setVelocity(pct, percentUnits::pct);
 if (Controller1.ButtonUp.pressing())
   RollerMotor.spin(directionType::fwd);
 else if (Controller1.ButtonDown.pressing())
   RollerMotor.spin(directionType::rev);
 else
   RollerMotor.stop(brakeType::coast);
}
```

project Code 1 designed by: Riley F witnessed by: [signature]

date: 12/12/22

12/12/22 (continued)

```
/**
 * Spins the intake when the left bumpers on the controller are pressed
 *
 * args:      pct (the velocity percentage that the intake will run at)
 *            brk (the mode that )
 * returns:   ---
 */
void spinIntake(float pct, brakeType brk) {
 IntakeMotor.setVelocity(pct, percentUnits::pct);
 if (Controller1.ButtonL1.pressing())
   IntakeMotor.spin(directionType::fwd);
 else if (Controller1.ButtonL2.pressing())
   IntakeMotor.spin(directionType::rev);
 else
   IntakeMotor.stop(brk);
}

/**
 * Moves the robot based on input from the controller joysticks
 */
void move() {
 // Get the joystick values
 int forwardBack = -Controller1.Axis3.position(vex::percent);
 int sideways = -Controller1.Axis1.position(vex::percent);
 int turning = Controller1.Axis4.position(vex::percent);

 // Turn the motors accordingly
 rightMotorA.spin(vex::forward, forwardBack - sideways - turning,
vex::percent);
 leftMotorA.spin(vex::forward, forwardBack + sideways + turning,
vex::percent);
 rightMotorB.spin(vex::forward, forwardBack + sideways - turning,
vex::percent);
 leftMotorB.spin(vex::forward, forwardBack - sideways + turning,
vex::percent);
}
```

project Code 1 designed by: Riley F. witnessed by: [signature]

date: 12/12/22

12/12/22 (continued)

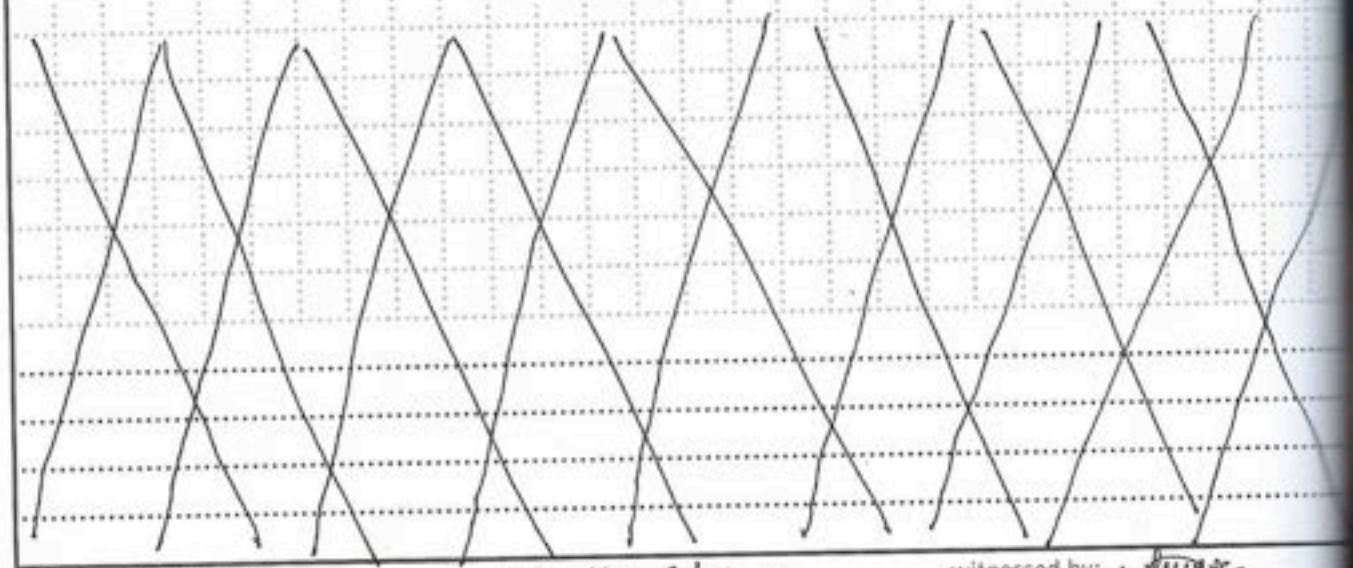
```
/**
 * Activates the piston when the correct controller button is pressed.
 * Piston should not be able to be activated before the 10 second mark.
 * Rumbles the controller and prints time to brain
*/
bool activated = false;
void extendPiston() {
  Controller1.Screen.print(Brain.Timer.value());
  if(Brain.Timer.value() >= 5) {
    if(!activated) {
      Controller1.rumble("...");
      activated = true;
    }
    if(Controller1.ButtonUp.pressing())
      piston.open();
  }
  Controller1.Screen.clearScreen();
}

//*********************************************************
// DRIVER CONTROL FUNCTIONS
//*********************************************************

void user_control() {
Brain.Timer.reset();
 // Start the flywheel at the beginning of the driver control period
startFlywheel(100);

// Main loop
while (true) {
  moveRoller(25);
  spinIntake(40, brakeType::hold);
  Controller1.ButtonA.pressed(flywheelControls);
  extendPiston();
  move();

  vex::task::sleep(1);
}
```

project Code 1 designed by: Riley F. witnessed by: [signature]

date: 12/12/22

12/12/22 (continued)

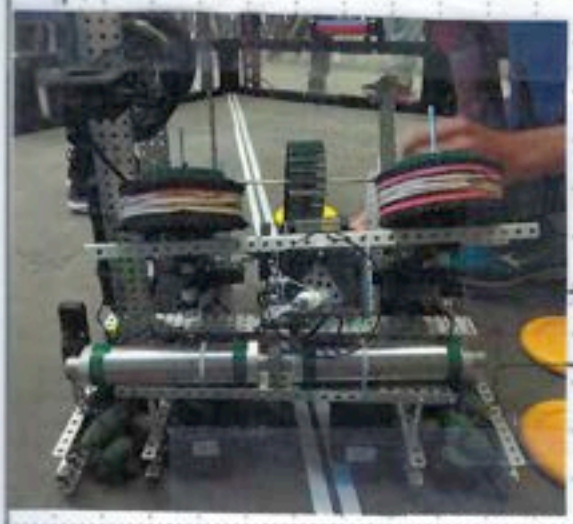
```
//----------------------------------------------------------
// AUTONOMOUS FUNCTIONS
//----------------------------------------------------------

void auto_1() {
 Drivetrain.setDriveVelocity(50, percentUnits::pct);
 Drivetrain.driveFor(directionType::fwd, 10, distanceUnits::in);
}

int autonomous_number = 0;
void auton() {
 printf("<<<AUTO>>>");
 Brain.Screen.print("<<<AUTO>>>");
 Controller1.Screen.print("<<<AUTO>>>");

 if (autonomous_number == 0)
   auto_1();
}

int main() {
 vexcodeInit(); // INITIALIZE OBJECTS: DO NOT TOUCH!!!
 Competition.autonomous(auton);
 Competition.drivercontrol(user_control);
}
```



project Code 1 designed by: Riley F witnessed by:

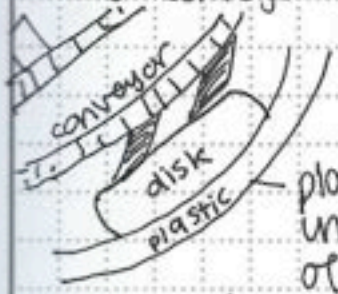
date: 12/12/22

12/13/22

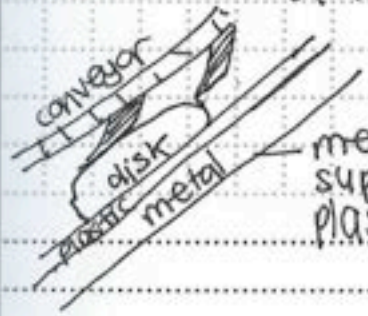


We were able to install the extender under the launcher, with the pneumatic air tanks lying horizontally across the base. String on a metal rod will be able to be launched out and allow our robot to score at least 3 additional tiles, a total of $3 \times 3 + 4 \times 3$ (↑ from robot occupied area) which equals 21 points from tile area. -LL

We noticed for our intake that the gap between the vanguard wheel and the conveyer belt was too big for the disk to be moved up towards the launcher. To fix this, we attempted to use bigger flaps. We were able to intake the disk by driving over it, but it got jammed halfway up the conveyor belt. We noticed that the disk pressed down against the plastic, creating greater difficulty for the disk to be moved. To fix this, we attempted to support the plastic and keep it straight by putting a thin sheet of metal underneath the plastic sheet. The disk no longer gets stuck, but the intake is still very slow. To solve this issue, we attempted to put guides to help the disk stay on track on the conveyor using 1 by pieces of metal as a sort of guard rail. -LL



plastic bending under the force of the disk.



metal to support the plastic.

project Disk Intake Troubleshooting designed by: Ian, Rain, Sam witnessed by: [signature]

date: 12/13/22

12/15/22 we noticed a problem with one of our launchers where one was spinning noticably slowly, slower than the other. we assumed that it was a problem with the motor, but after applying some WD-40 to the connections of the launcher wheel, it starts working fine. -LL

12/16/22 Each year, we like to make a kind of summary of the game manual for quick access during matches so we can advocate our position effectively should a dispute on the rules arise. This year, we were a little late starting on this summary, so it's not ruthlessly optimised for the most efficient use during a match, and due to the other commitments of some team members progress was rather slow meaning its now a bit out of date, but it should still be useful if called upon. - SPP

project QR Handbook designed by: Sam P-P witnessed by:

date: 12/16

Drive Team Quick Reference Handbook

Game manual version 2.2. Next scheduled minor update 6 Dec 2022; next scheduled major update 31 Jan 2023. Page numbers refer to those in the corners of pages.

General

- Update information. —p. 4
- Vex Q&A responses are official rulings. —p. 5
- Use common sense. —G3, p. 25
- How violations work. —pp. 11-13
- Be safe; be nice. —S1, p. 23; G1, p. 24
- Match schedule can change. —p. 52; T9, p. 57

Drive Team Conduct

- Wear safety glasses when on the drive team. —S3, p. 23
- Phones must be in airplane mode. —G7a, p. 27
- You may not unplug your controller without permission. —G8, p. 28
- During the driver control period, if your robot has not moved at all during the match, you may touch your robot for certain troubleshooting reasons. —G9a, p. 28
- Do not break the plane, including when using the loaders. —G9b, p. 28
- Do not push on the perimeter in order to influence game objects. —G9c, p. 28
- Be careful about introducing discs on loaders. There should never be transitive person-robot contact, the discs should not be pushed from the loader with enough velocity to significantly change their horizontal position, and neither robot nor person should break the plane. —SG6, pp. 35-36
- If the string you use to expand is hard to see or not distinguishable as belonging to your alliance, the referees might miss it; tough. —R7(red box), p. 43
- Controllers may be plugged into power banks during matches. —R18d(i), p. 47
- No filling pneumatics on the field. —T6a, p. 56
- Robots must be placed on the field promptly. —T6b, p. 56
- Red alliance may place last; if a team repositions their robot, the other can reciprocate. —T8, p. 57

Gameplay

- Entanglement definition —p. 9
- Trapping definition —p. 11
- Tape and barrier are part of low goal —p. 15
- Possession definition —p. 17
- Scored in high goal definitions —SC2, p. 18

project QR handbook designed by: Sam PD witnessed by:

date: 12/16

- Scored in low goal definitions —SC3, p. 20
- Owned roller definitions —SC4, p. 21
- Covered field tile definitions —SC5, p. 22
- You can't touch anything outside the field perimeter during endgame —S2, p. 23
- An autonomous bonus-affecting violation during auton results in the bonus going to the other alliance (if both alliances do this, no auton bonus) —G11, p. 29
- Do not intentionally destroy other robots —G12, p. 29
- Some incidental damage is expected, especially during endgame —G12b, G12d, p. 29
- Reckless driving and design (small wheel base, for example) have consequences —G12c, p. 29
- You can't trap more than five seconds at a time, with at least five seconds between each trap, but traps that begin during the endgame are permissible—G15, p. 30/39, see also "trapping", p. 11/20
- Don't clamp onto the field —G16, p. 30
- Discs are extensions of the robot —G17, p. 30
- Any entanglement with the net results in at least a disablement. If a robot causes an opposing robot to become entangled with the net, both robots are disabled; this is done for safety, not penalty. —SG3, p. 32
- Do not expand beyond 18" by 18" before the endgame. If you do, immediately either fix it or remove yourself from gameplay by parking in an unused corner. —SG4, pp. 32-33
- Don't possess more than three discs. If you do by accident, immediately stop play and rectify. —SG7, p. 36
- Be careful when accessing discs on the line during autonomous. —SG8b, pp. 36-37
- Discs unintentionally leaving the field is fine; they'll be returned to the nearest location in the field by a referee. Do not do this repeatedly or intentionally. —SG9, p. 37
- You get one 3-minute time-out during the elimination rounds. It can be activated between matches by telling the head ref or event partner. —T17, p. 61

Violations

- The head ref's word is final. —T1, p. 54
- The head ref must say which rule has been violated. —T1b, p. 54
- If you wish to appeal a decision, you must stay in the alliance station until the head ref talks to you. —T3, p. 55
- Referees may not review photos or videos from the match when making decisions. —T3a, p. 55
- Possible criteria for replays —T7, pp. 56-57
- Offensive robots get the benefit of the doubt —G13, p. 29
- You can't force an opponent into a penalty —G14, p. 30
- Violation definitions —pp. 11-13

12/17/22 We had programmed our mechanum wheels so that the left joystick controlled linear movement — forward, back, left, right — and the right joystick controlled turning. Because our robot didn't come together until shortly before the tournament, our driver Ian didn't get the chance to try this control scheme until recently. When he did, it wasn't very intuitive for him; he thought tank drive would make more sense. Since this was at our final meeting before the tournament, I copied our program onto my personal computer before we left and made what I thought would implement what made most sense to Ian: traditional tank drive, which uses the front/back axes of both joysticks, plus side-to-side motion controlled by the left/right axes of either joystick (whichever is greater). This morning (the morning of the tournament), we downloaded this untested program, and for what must have been the first time in my robotics career, the solution worked perfectly on the first try. - SPP

project tank drive designed by: Sam PP witnessed by:

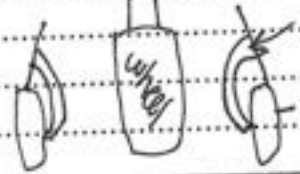
date: 12/17

12/19/22 Masuk Tournament Reflection:

We went to a tournament at Masuk in Monroe CT on December 17. We did alright. Unfortunately, due to all the time spent on our robot, our driver wasn't able to get in a lot of driving practice so he had to learn on the go. We were able to use a left side autonomous by our 4th match where our robot was positioned a distance away from the roller. We were able to successfully turn the roller our correct color. Overall, our robot was efficient at turning rollers, although our roller turner's screws weren't tight enough in one match and loosened, causing our robot to be unable to turn the roller that match. Our driver, Ian, was also able to improve quickly, manuevering our robot around to spin rollers and play defense against other teams. We discovered early on in our matches that our robot was prone to getting stuck on raised parts of the field and disks. After our robot experienced difficulties for the second time, we added a metal guard around our base to prevent disks from getting stuck. This solved our problem well and we didn't get stuck again.

We made it into the elimination rounds as the 13th seed we did our scouting well and even though our first choice alliance was taken, our second choice team was also really good. We played a good match but lost because both of us did not have extenders while the other team did. For our next tournament, we'll work on fixing our intake, which is too unreliable to use during a tournament, and adding an indexer to our ~~robots~~ robot so we can hold more disks.

In the intake area, we had also added slightly angled metal to direct the disks



project Masuk Tournament Reflection designed by: Lucy Liu witnessed by: Sungjin Park date: 12/19/22

12/20/22 We noticed that we needed smaller wheels on the roller tumer since the larger size of the wheel was interfering with our intake-launcher system. We replaced them with 15-tooth gears wrapped in tread, then mesh, then rubber bands. This works just as effectively when tested, at a smaller size.

Riley's working on an extender design, using pneumatics to launch a cord from a very compact structure, with a rubber band stretched to provide tension, and the piston responsible for releasing the tension.

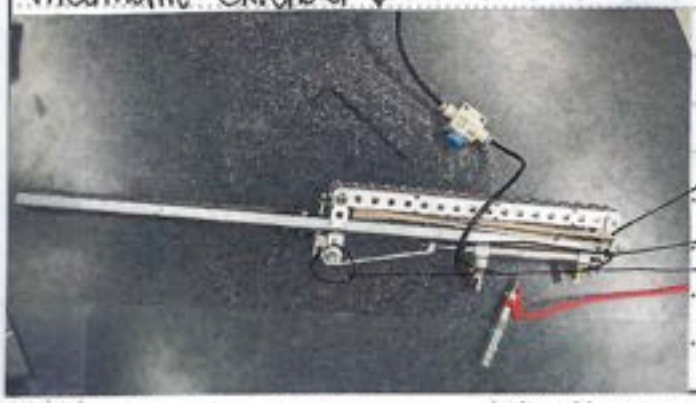
The intake-launcher connection was adjusted to be straighter to allow for a better path for the disk. -LL

1/3/23 Since we have limited time before our next tournament, we decided to prioritize making a functional intake launcher system over developing an intake indexer.

We're planning on making 2-3 pneumatic string launchers as an extender design.

We had some trouble attaching the intake-launcher onto our base. Ian suggested securing it with string, but I thought that would be unstable and had possible dangers of getting tangled. I suggested that we prop the intake-launcher to the right angle by using a piece of metal between the metal holding up the pneumatic tank and the launcher. -LL

pneumatic extender ↓



When the extender is activated, the pin is released and the string is launched. -LL

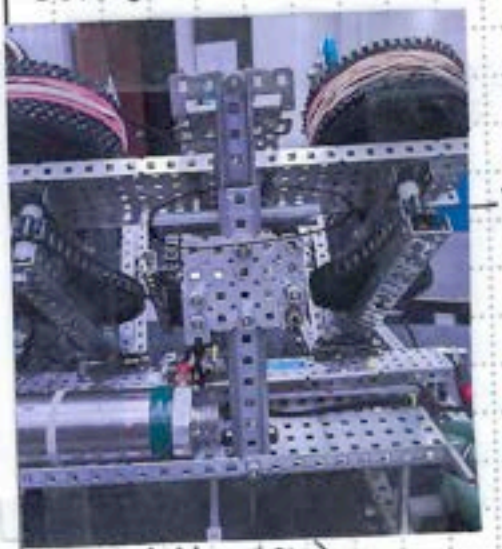
project extender design designed by: Riley F. witnessed by:

date: 1/3/23

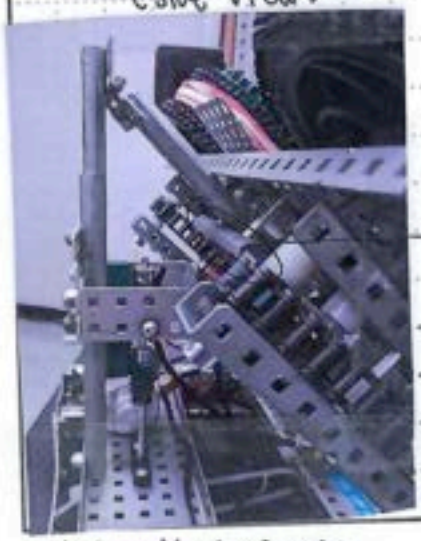
1/7/23 we built our ramp adjuster to connect the intake-launcher to our base. This ramp adjuster allows us to change the angle of our ramp. Unfortunately we noticed that we had attached the rails incorrectly ~~so~~ since they weren't sliding in the right way. We also finished building our second extender and attached our extenders to our robot: we attached them to the standoff are-by metal on the sides of our intake.

We tested out our intake and it finally works, able to smoothly pick up disks while driving around the field. our aim still needs practice, however. -LL

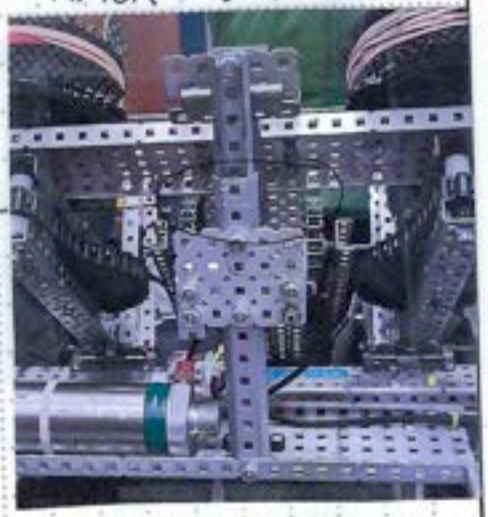
Before (Front view)



(side view)



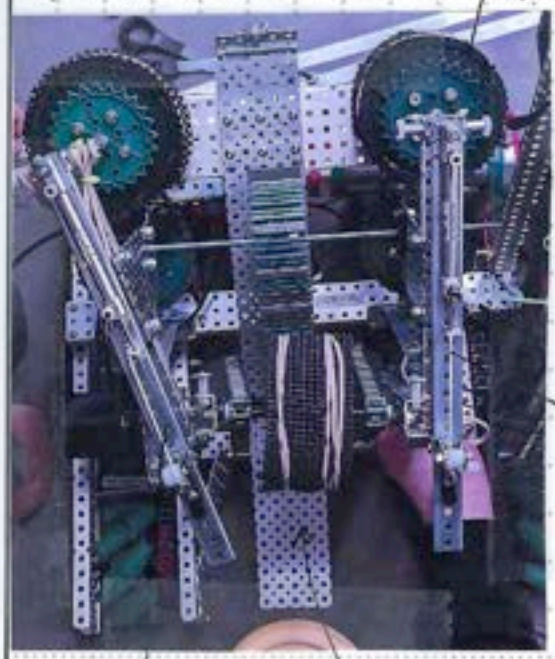
After (front view)



(side view)

project Rail placement correction designed by: IAA witnessed by: [signature] date: 1/7/23

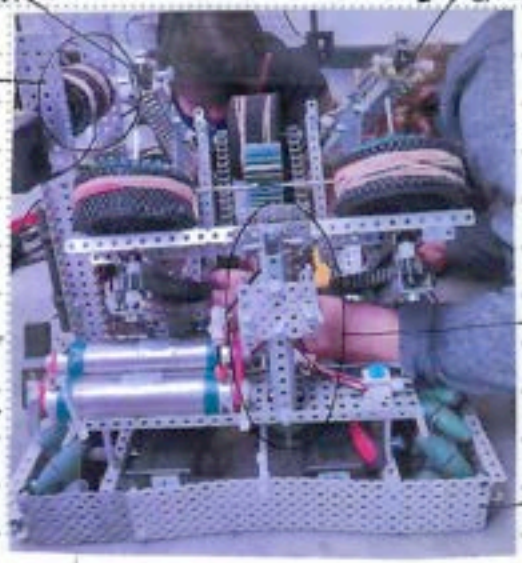
1/7/23 (continued)



one of our extenders is placed at an angle, while the other is positioned straight.

Next meeting we'll practice loading / reloading the string, as well as confirm the length each should be at to get us the maximum points but avoid getting disqualified. -LL

extender [our current design] extender



project Extender Attachment designed by: Riley witnessed by:

date: 1/7/23

1/9/23 Sam worked on code for Autonomous selection. -IL

```
int screenTop = 0; // The y-value in pixels of where the next button should
   be placed. Each time a button is placed, this value is incremented to
   correspond with the bottom of that button.
int buttons[3][5]; // [xpos, ypos, width, height] Stores information about
   the buttons on the screen
int buttonArrayIndex = 0; // The index at which the next element added to
   buttons should be placed
int buttonPadding = 20;
int autoNumber = 0; // The index of the currently selected autonomous.
   Incremented on button press.
int numberOfAutonomi = 2; // Total number of autonomous programs.
int driverNumber = 0; // The index of the currently selected drive program.
   Incremented on button press.
int numberOfDrivers = 2; // Total number of drive programs.

void makeButton(std::string text, int buttonX, int buttonY) {
      int textWidth = Brain.Screen.getStringWidth(text.c_str());
      int textHeight = Brain.Screen.getStringHeight(text.c_str());
      Brain.Screen.setFillColor( black );
      Brain.Screen.setPenColor( white );
      // Draw text and a rectangle around it.
      Brain.Screen.drawRectangle(buttonX, buttonY + screenTop, textWidth +
         buttonPadding * 2, textHeight + buttonPadding * 2);
      Brain.Screen.printAt(buttonX + buttonPadding, buttonY + buttonPadding +
         textHeight * .75 + screenTop, true, "%s", text.c_str());
      // Add information about the button to the array for reference by the
         function pressedCallback
      buttons[buttonArrayIndex][0] = buttonX;
      buttons[buttonArrayIndex][1] = buttonY + screenTop;
      buttons[buttonArrayIndex][2] = textWidth + buttonPadding * 2;
      buttons[buttonArrayIndex][3] = textHeight + buttonPadding * 2;
      screenTop += ButtonY + textHeight + buttonPadding * 2; // Add the
         height of the button to screenTop.
      buttonArrayIndex++;
}
```

project Autonomous selection Code designed by: Sam P-D witnessed by: [signature] date: 1/9/23

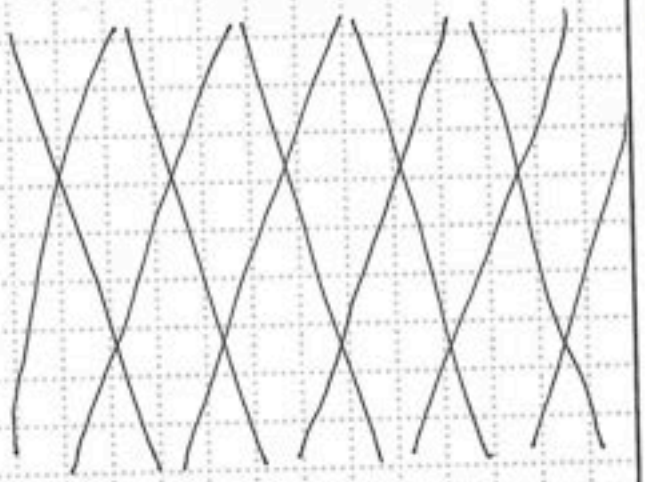
1/9/23 (continued)

```
void makeStats() {
      Brain.Screen.setCursor(1, 26);
      Brain.Screen.print("Auto: ");
      switch(autoNumber){
            case 0:
                  Brain.Screen.print("no auto");
                  break;
            case 1:
                  Brain.Screen.print("right side");
                  Break;
      }
      Brain.Screen.print("                              "); // Ensures the text
         previously in this location is fully covered
      Brain.Screen.setCursor(2, 26);
      Brain.Screen.print("Drive: ");
      switch(driverNumber){
            case 0:
                  Brain.Screen.print("Ian tank drive");
                  break;
            case 1:
                  Brain.Screen.print("Arcade drive");
      }
      Brain.Screen.print("                              ");
}

void pressedCallback() {
      if (buttons[0][0] < Brain.Screen.xPosition() &&
         Brain.Screen.xPosition() < buttons[0][0] + buttons[0][2] &&
         buttons[0][1] < Brain.Screen.yPosition() && Brain.Screen.yPosition()
         < buttons[0][1] + buttons[0][3]) {
            autoNumber++;
            if (autoNumber == numberOfAutonomi) {
                  autoNumber = 0;
            }
            makeStats();
      }
      if (buttons[1][0] < Brain.Screen.xPosition() &&
         Brain.Screen.xPosition() < buttons[1][0] + buttons[1][2] &&
         buttons[1][1] < Brain.Screen.yPosition() && Brain.Screen.yPosition()
         < buttons[1][1] + buttons[1][3]) {
            driverNumber++;
            if (driverNumber == numberOfDrivers) {
                  driverNumber = 0;
            }
            makeStats();
```

1/9/23 (continued)

```
void pre_auton(void) {
      vexcodeInit(); // Initializing Robot Configuration. DO NOT REMOVE!
      // Brain.Screen.print("preauto");
      Brain.Screen.setPenColor(white);
      Brain.Screen.drawLine(240, 0, 240, 272);
      makeButton("Cycle autonomi", 10, 10);
      makeButton("Cycle driver", 10, 10);
      makeStats();
      Brain.Screen.pressed(pressedCallback);
}
```

Autonomous Selection



we also tried to construct 2 string storage containers to keep the string on the robot.

our driver also had the opprotunity to practice driving. We're able to shoot disks into the high goal but our consistency could be improved.

we encountered the problem of one wheel of our launcher spinning slower than the other again. It's the same wheel that was having problems as last time. we used WD-40 but the problem persisted. upon closer examination, we noticed that one of the cords (attached to the motor attached to the roller) was in the way of the wheel which was rubbing against it. we moved the cord and the problem was fixed. -LL

project Autonomous Selection Code designed by: Sam witnessed by:

date: 1/9/23

1/9/23 we noticed another issue: when our string launched, since our launcher is always on, the string would get tangled in the spinning wheels of our launcher. Upon closer examination, we realized it was just a manual error of touching the string during testing, something we want encounter during an actual tournament. our goal for next meeting is to have the 2 string storage containers securely attached to our robot in a way that doesn't disrupt it. -LL

1/10/23 during testing today, we discovered that the ramp/intake dug into the ground, preventing our robot from moving backwards. We adjusted the intake ramp and this problem was solved.

We have the 2 string storages designed resembling a metal rectangular bin with an open top, attached using zip ties to the front bar containing our pneumatic air tanks. We were able to attach the string storages and load the string.

We tested our autonomous and realized ~~#~~ that we need more grip on the roller turner wheels since we changed the size. We decided to add another wheel, increasing from 2 to 3. -LL

1/13/23 Our expansion mechanisms (we call them "harpoons") use pneumatics, ~~[illegible]~~ when continue to mystify us. At the end of Tuesday's meeting, we discovered a hugely problematic quirk in the way the pneumatic solenoids respond to inputs: it seems that when the robot is turned off, the solenoids switch states, causing our harpoons to fire. This is a major problem because an accidental firing of a harpoon (particularly if the robot is not on the field) could injure someone. We immediately stopped all other work on the robot to look for a solution. Scouring the Vex forums revealed some people with similar problems but no workable solutions. After trying myriad permutations of pneumatics →

project Pre-Tournament Testing designed by: Riley witnessed by: [signature]

date: 1/10/23

1/13/23 (continued):
code, we ~~determined~~ ran out of ideas on how to fix the root of the problem. However, Riley come up with an idea that could circumvent the problem: two solenoids essentially running in series. Since the default state of a solenoid seems to be opposite when the robot is running from when it isn't, we could have one solenoid that's open when the robot is on and closed when it's off feeding into another that's closed when the robot is on and open when it's off. This would essentially create a pneumatic 'and' gate in which one input is always off. When we wanted to fire the harpoons, we could simply open the solenoid that is closed when the robot is powered on. At a supplemental meeting today that we scheduled for the purpose of solving this problem, we implemented Riley's solution. It was looking promising, but the solenoid we added stopped responding to inputs. We troubleshot for quite a while but eventually ran out of time and new things to try. Before we left, though, we did some testing so that we're now more sure of the pneumatics' behavior (one of the problems was uncertainty in how they would act). We now know that when the program starts, the pistons will retract then extend in rapid succession, but ~~not~~ never with enough force to pull the pin and launch the harpoons. When the robot is turned off, the harpoons will fire if they haven't already. ~~Unfortunately~~ Armed with this knowledge, I made this safety checklist to ensure nothing goes wrong. -SPD

project Pneumatics troubleshooting + checklist designed by: Riley + Sam witnessed by:
date: 1/13/23

Preflight safety checklist

1618A; from 13 January 2023

In queue

- ☐ Check in with queuemasters
- ☐ Affix proper license plate
- ☐ Check robot battery
- ☐ Check controller battery
- ☐ Ensure all screws are fully tightened
- ☐ Tighten threaded collars on harpoons
- ☐ Check harpoons' rope coils
- ☐ Don safety goggles
- ☐ Pressurise air tanks to no more than 100 psi
- ☐ Cock harpoon guns using ramrod

On field

- ☐ Place robot on field; turn on robot
- ☐ Plug controller into field; turn on controller
- ☐ Run program
- ☐ Select autonomous and driver program
- ☐ Load harpoons

After match

- ☐ Remove harpoons from guns if unfired
- ☐ Secure harpoons to prevent from swinging
- ☐ Stop program
- ☐ Turn off robot
- ☐ Be in possession of:
  - ☐ Robot
  - ☐ Controller
  - ☐ Ram rod
  - ☐ Bicycle pump

project ..... designed by: ..... witnessed by: .....

date: .....