Table of Contents:

Engineering:

Team Aries, 15342, started out in 2018 with a weak team structure and an unefficient system of doing things. Most of the members were starting their first year of FIRST robotics. Due to there hard work. Aries was able to advance on from League Tournament, but were unable to compete further because Super-Qualifiers was cancelled due to the weather.

This year, Aries has improved their structure, and system of work, and have moved their way up to become one of the best teams in Southern Oregon, winning the title of winning Alliance - First team picked, and winners of the Inspire Award. One of the main factors in their winning the Inspire Award was their outreach to Hucrest Elementary School (see page 27) where they teach Lego Robotics classes to thirty fifth graders every Tuesday. This has proven to be effective in spreading the word of FIRST to the community, for many of the fifth graders wish to join robotics this next coming year. Other forms of outreach that have been done are counseling a camp at the Boys and Girls Club and holding a booth at the Douglas County Fair.

The business section highlights our relationship with other teams in FIRST Tech Challenge and how our teams work. It includes demographics on the team including race and gender and talks about our sponsors.

The engineering section talks about the thought process behind our designs and how the robot is engineered. It also includes bits of programming that were fixed along with the progression of the robot.

Team Section

Brian: Co-captain and programmer

Brian is the co-captain and main programmer on Aries as well as being the web designer for Roseburg High's Robotics Club. Outside of Robotics, he is in Cross-Country, Tennis, Band, FBLA, and takes three honors classes and one AP class.

"Don't Be Bad," - Brian

Logan: Co-captain and Business

Logan is the other co-captain and is the business person on Aries. In his free time, he is in band, tennis, and FBLA. This is Logan's fourth year of robotics. He takes three honors classes and one AP class at RHS. His goal is to go to state.

"The future belongs to those who prepare for it today," - Malcolm X"

Leonhard: Programmer

Leonhard is our German exchange student. He plays trombone and is in National Honor Society. His goal is to learn how to engineer and program a robot. This is his first year of doing robotics.

"Cogito Ergo Sum," - René Descartes

Amy: Head Notebooker

Amy is one of the sophomores on Aries. Outside of Robotics, she is on the school newspaper and is in the school's film company. One of her goals is to learn how to program.

"Some Infinities Are Bigger Than Other Infinities,"
- John Green

Preston: Engineer

This is Preston's third year of Robotics. He is also in Key Club, Cross-Country, Track, and Band. His goal is to proficiently use Fusion 360 and AutoCad to help design the robot.

"In the middle of every difficulty lies opportunity," - Albert Einstein

Max: Engineer

Max, one of the newer additions to our team, is a freshman. He enjoys his honors math class and hopes to get sponsored by Little Caesar's Pizza.

Jesse: Designer

Jesse is a sophomore at RHS and this is his second year in Robotics. His goal this year is to get better at drafting and make more friends with those in robotics.

"Beneath this mask, there is more than flesh, beneath this mask there is an idea and ideas are bulletproof," - Alan Moore

Wayne: Engineer

Wayne is one of the engineers on our team. His hobbies include reading and his goal is to expand his knowledge in technology and robotics.

"Dreams tend to crush the dreamer, unless the dreamer sizes up to conquer them,"

-

Verelle: Coach

Verelle has a M.S. degree in Hearing and Speech Science from Arizona State University and a Doctor of Audiology degree from Salus University in Pennsylvania. Her hobbies include swimming and violin.

Ms. Knight: Coach

Ms. Knight earned her Bachelor's Degree in Teacher Education from Northwest Christian University and was a math teacher for seven years. She now works as a School to Career Transition Specialist with Douglas ESD.

Outreach

Outreach:

This year, our main focus for outreach has been to spread the message of FIRST through the children of our community. This has been done in three different ways.

1. Helping as counselors at a Lego Robotics Camp at the Boys and Girls Club
2. Holding a Lego Robotics Booth at the Douglas County Fair
3. "Teaching" Lego Robotics classes at Hucrest Elementary School

We, Team Aries, take the most pride in our work at Hucrest Elementary. Every Tuesday, we leave school an hour early to teach a fifth grade class programming, engineering, and team relationship skills.

How Does it Work?

- We visit a classroom of 30 students and split them up into groups of four or five.
- They are given a task of building and programming a sumo-bot that they will compete with after five weeks.
- Each team gets to compete with all the other teams in the room in a sumo-bot championship.

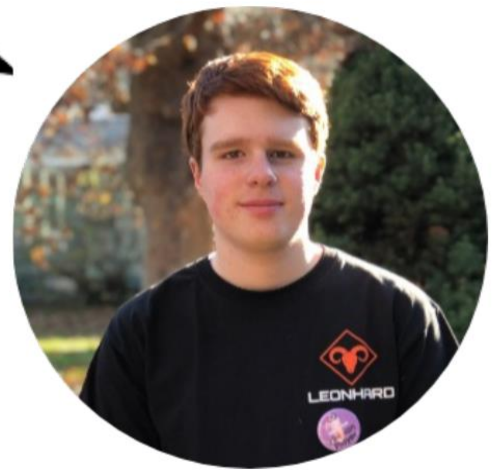
How is it Helpful to the Kids?

Our outreach to Hucrest Elementary School has shown to teach a lot through the course of the four weeks we would hold teach a classroom. The students would learn team relationship skills, critical thinking skills, designing, building and programming. So far, we have taught two classrooms and are working on the final fifth grade class at Hucrest Elementary before we move further.

"Witnessing teams plan and divide their teams into sections to efficiently build and program a fully functioning Lego robot."

"Seeing the young kids light up with smiles as we teach them about engineering."

"It is a lot of fun and successful. I think the Hucrest students learned a lot, like lego programming."



"Seeing them learn to work together to achieve a goal. It's a great experience and is something I hope to continue doing."

"They all worked very hard on their robot and tried their best. It was great being able to watch them and help with any troubles they have had."

"Helping 5th graders learn how to use the engineering process to design, build, and program a working sumo robot out of EV3 LEGO kits. I think that it has sparked an interest in STEM that will penetrate throughout Hucrest."

"Seeing them work together to accomplish one goal."

"When the children come together as a team and formulate ideas to overcome challenges that appear along the way ... I think what we do is good and beneficial for the students there."

15342 Aries

Roseburg High School

Business Plan

TABLE OF CONTENTS

Page Number

1.0 Executive Summary

1.1 FIRST® Description

FIRST® (For Inspiration and Recognition of Science and Technology) is an international foundation that promotes science and engineering through robotics. It was founded in 1989 by Dean Kamen and the now deceased Woodie Flowers, both brilliant people. The mission of FIRST® is to "inspire young people to be science and technology leaders and innovators, by engaging them in exciting mentor-based programs that build science, engineering, and technology skills, that inspire innovation, and that foster well-rounded life capabilities including self-confidence, communication, and leadership."

1.2 FIRST® Tech Challenge Overview

The *FIRST* Tech Challenge (FTC) is a robotics competition that provides students the opportunity to participate in a real world competitive STEM (Science, Technology, Engineering, and Math) team. Guided by adult Coaches and Mentors, students develop STEM skills and practice engineering principles (like keeping an engineering notebook), while realizing the value of hard work, innovation, and sharing ideas. The robot kit is reusable from year-to-year and can be programmed using a variety of languages, including OnBot Java, which is a spinoff of Java. Teams also must raise funds, design and market their team brand, and do community outreach for which they can win awards. Participants have access to tens of millions of dollars in college scholarships. Each season concludes with regional championship events and an exciting *FIRST* Championship.

2.0 Executive Summary

2.1 Team Purpose

The purpose of our team is to increase interest in technology in our town, and to be a place to help foster and grow STEM knowledge and skills across our high school population. We strive to encourage STEM fields and post-secondary education with giving basic knowledge, career explanations, and problem solving skills that correlate those jobs.

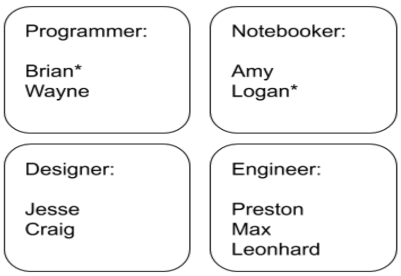
2.2 Team Motto

"Nobody who ever gave his [or her] best regretted it."

- *George Stanley Halas*

2.3 RHS Robotics Description

RHS Robotics is entering their fourth year as an organization this year. From 6 members and one FTC team, we have come a long way to three teams, over forty members, and being a three time state qualifier. Team Aries is a part of the three part conglomerate that makes up Roseburg High School Robotics.



2.4 Team Aries and Organizational Structure

Team Aries is a multi-faceted team. We are separated into four main parts on our team: Programmer, Engineer, Designer and Engineering Notebook. Although we do have some crossover of positions, we have only represented the primary roles. As you can see in the diagram on the right, we have a very functional team. The asterisks show the team captains, of which we have two.

2.5 Team Coaches and Mentors

Here on Aries we have two coaches and two mentors. Mrs. Knight and Mrs. Powell are the two coaches for our team. Mrs. Knight is a staff member at Roseburg High School working in the RHS Career Development Center and Mrs. Powell is a stay-at-home mom and the mother of Brian Powell on our team. As well as our two team coaches, we have two team mentors: Mr. Wier and Andrew Whightsil. The first mentor, Mr. Wier, is a physics teacher at Roseburg High School and he has been the executive authority of all Roseburg High School teams for his fourth year now. The second mentor, Andrew Whightsil, graduated from Roseburg High School last year and was the former president of Roseburg High School Robotics club. He is currently attending Umpqua Community College and planning to attend Oregon State University next year.

2.6 Team Relationships and Sponsors

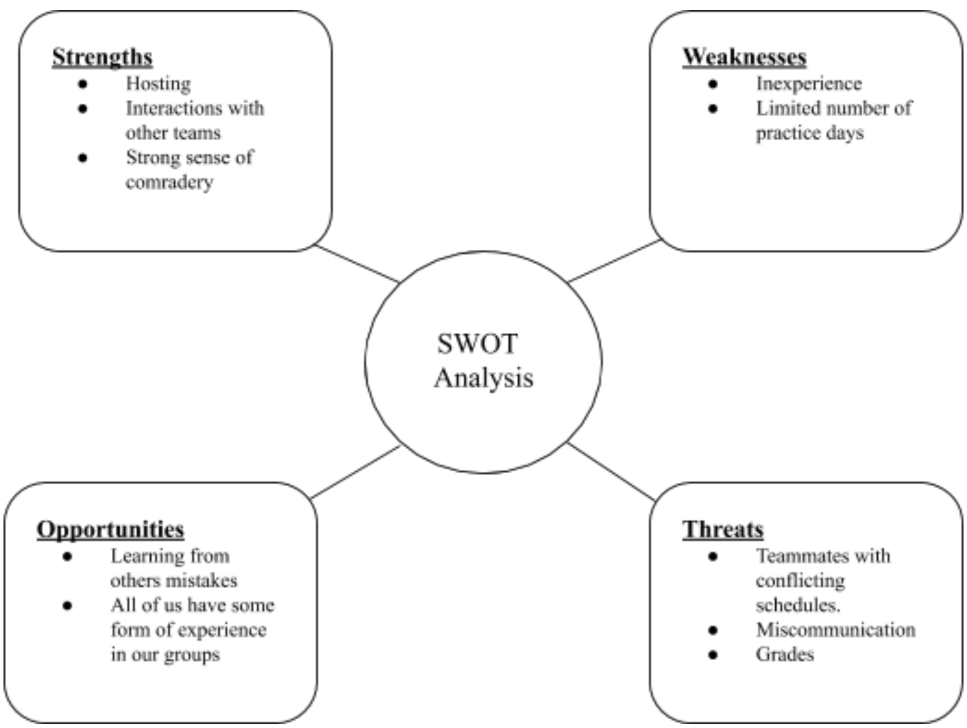
As part of a robotics conglomerate, we have many intramural connections. These teams are 15341 Taurus and 13189 Scorpio. Outside of our school however, we have befriended many teams like 7750 Mini Maniacs from Roseburg, 9567 Cybernetic Elks from Elkton, and 12132 Biobotz2.0 from Roseburg. Together we have had many practices with these teams at Roseburg High School and because of our partnerships we have been able to work to give an exchange of ideas to help better all teams involved.

3.0 Sustainability

3.1 Team Mission

For the 2019-2020 season, we as a team have organized our goals for the season. They are as follows: Establish an elementary school FLL team, place in the top three teams for competition at the league meet, and bring on newer teammates to help carry the team forward after the older members graduate. This sustainability allows for us to be able to push forward the involvement of robotics and STEAM in our community which in turn allows for exponential growth and the inspiration of the next generation of robotics students.

3.2 SWOT Analysis



Strengths: This year, we've had many advantages including a supportive coach and mentor, but the larger ones are our experience. On our team alone, we have a combined 18 years of experience in robotics, with three people having four years of experience in the FIRST program. We also have had a lot of experience in FTC, and our experience will definitely allow for us to set up for more future success.

Opportunities: We have the opportunity to use our experience from our past years to better ourselves and fall into similar positions that we were in last year's competition such as not being able to get ample driving practice in. Another opportunity for us is that all of us we will be able to apply ourselves a little bit better to the situation.

Weaknesses: We have had a lot of newer kids on our team with new people constituting over 40% of our team. This could prove to be a hindrance by trying to get people to find their roles, and giving them all a type of "job exploration".

Threats: Despite all the positives that we have on our team, we certainly do have some threats on our team as well. Some of these threats have been realized, but thankfully so far this year, most have not. Some of our threats are teammates with conflicting schedules, which is a true thing with many members in our team being involved in sports and other on campus school activities. Another threat to us is the miscommunication that can happen to any team out there, due to absenteeism or some other form of truancy in robotics or responding to questions.

3.3 Team Goals

ACTIONS	STRATEGY
Expand the *FIRST* Program	• Reach out to Hucrest Elementary school by giving presentations to 5th graders about robotics. • Talk to Hucrest Elementary FLL Team and help show them to how to solve issues that they encounter.
Recruit new team members	• Reach out to teachers at RHS of fields that we specifically want (i.e. Journalism, Drafting, etc.)' • Hold interviews for interested applicants.
Develop Social Media Presence	• Further develop our YouTube channel, which has over 1900 views! • Help connect with teams across the nation about their robots.
Increase Team Funding	• Gain more team sponsors • Talk to local businesses about supporting our team • Plan fundraising events (i.e. Bottle & Can Drives, Dance, Car Washes, etc.)

3.4 Team Demographics

Team Information	
Team Year	2
Team Name	15342 Aries
Nickname	Aries
Location	Roseburg, OR
Locale	Roseburg High School, M - 211
Number of Students on Team	9
Number of Female Students on Team	1
Number of Male Students on Team	8
Number of High School Freshmen	1
Number of High School Sophomores	7
Number of High School Seniors	1
Percent of African-American/Black Students on Team	11%
Percent of Asians on Team	22%
Percent of White/Caucasian Students on Team	67%
International Students on Team	1
Team Motto	*"Nobody who ever gave his [or her] best regretted it."* • *George Stanley Halas*
Team Website	www.rhsrobotics.net
Other Media	YouTube: Aries Robotics

3.5 Summary of Team Growth

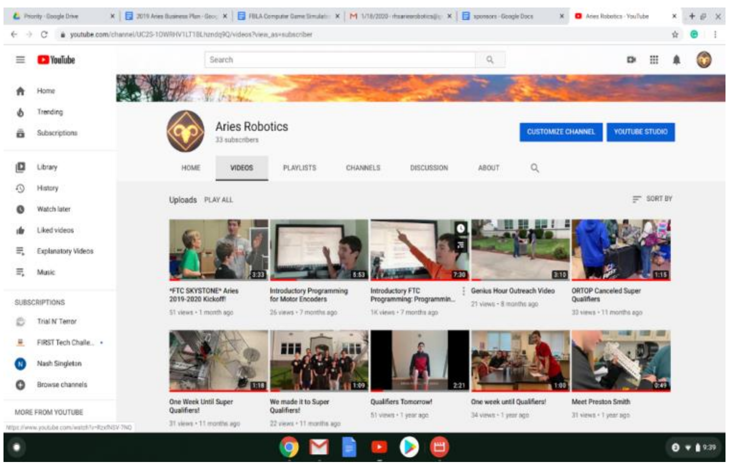
From last year, our team has stayed at the same number. However, we have grown a lot in younger aged students with only 1 student on our team being an upperclassmen. We have recruited a lot to help fill in the roles of the seniors and juniors from last year. We have been able to create a foreseeable future for our team because we have roles established from now until the time we graduate. We have an exciting path ahead of us, and we are ready for whatever it holds.

3.6 Summary of Future Team Plans

Aries has many goals this year, but one of the highlighted goals for us and our program has been the creation of Elementary FLL teams, and the fostering of young growth in FIRST and robotics. We've been able to see with our own eyes from the creation of the FLL and FTC teams at the local middle schools all the good that being involved in robotics early offers. Aries took charge of helping out Hucrest Elementary School with their robotics program, and we were able to create four teams and over sixty students our first year. Besides our first year with mentoring other teams, Aries has many other plans. We plan on reaching out to other FTC teams in our county about doing mini tournaments to help all of us get better. Aries has a goal of making state this year, and these practices will certainly help us get there.

3.8 Summary of Outreach

Our team has been fortunate enough to be able to work with our school district, and thanks to Roseburg Public Schools, we have been able to go to teach FIRST Lego League robotics at Hucrest Elementary School in our community. We have taught LEGO robotics to fifth graders giving them an explanation of what we do on a regular basis with the evaluation and implementation of the engineering process, and the process of building a robot and building upon your strengths and readjustment. We are hoping to continue this program for the future and continue our partnership with Umpqua STEAM Hub and Roseburg Public Schools. We have also built up a YouTube channel that has garnered over 2,000 views on our channel, and a robotics website with search engine impressions from over 10 countries. Also, we have had the privilege to be able to talk to professionals in engineering and designing from talking and presenting to designers at Con-Vey Keystone, programmers and website designers at Dell in Salt Lake City, get professional tours at Arenco, and also being able to communicate with one of the founding website programmers for Qualtrics, a unicorn website designing company. Through this we have been able to help establish connections with professional workplace jobs. Recently, we were also able to partner with Essential Craftsmen, a YouTube channel that has over 71.5 million views and over 700k subscribers. We are working on making a video with Essential Craftsmen right now comparing the similarities of FTC robotics and professional manufacturing and welding.



3.9 Sponsors and Business Partners

This year we emphasized a focus on sponsorship. We had the goal of getting everyone on our team to fundraise $300 by themselves by the end of the year. We do this so that we can be able to focus more on the robotics aspect of robotics, while not having to worry about money problems. Through this process, we exceeded our asking amount, raising over $12,000 alone this year, which helps cover expenses such as travel and housing for competitions like state. One of our business partners, Essential Craftsmen, is a well-known YouTube channel from Douglas County that has over 700,000 subscribers and over 71.5 million views. We are currently working on making a YouTube video with them. In this section we'd like to recognize our sponsors.

4.0 Lessons Learned This Year

4.1 Time Management

We have learned a lot this year in robotics, but one of the highlights that has come up has been the newfound improvement of time management skills on our team. We have been implementing new practices such as the use of apps like GroupMe and Remind with scheduling on the apps to make sure that we get tasks done at robotics in an orderly manner.

4.2 Comradery

In the words of John Steinbeck, "All of them had a restlessness in common." We as a team tried to create an environment where all of us could foster growth and discover newfound knowledge about the process of making a team that can work together in perfect sync. We have definitely come closer as mentors, friends, but more importantly we have become closer as a team through striving to make personal connections outside of the robotics lab.

4.3 Preparation

This year has led to a lot of timed decisions, and while we were happy with the times that we made them at, making those decisions can be difficult. We as a team can safely say that we are better able to evaluate our circumstances and conditions in which we are prepared to make a decision on the major and minor changes of our robot. We have become more thoughtful people because of our experiences this year.

5.0 Conclusion

5.1 Conclusion

This year has been a whirlwind. We have had so many wonderful and new experiences from being ranked second in the River league all season long, placing first at meet three, and being able to help innovate a new way to lead and inspire elementary students to see all the opportunities that FIRST and the STEM field can offer in their futures. We have also come closer as a team through utilizing communications technologies more efficiently, and by establishing a better use of time management. We have been working hard on our many media outlets, including our YouTube channel and our website. We have been able to help make new changes to the robotics program this year, and we have seen the rewards of our hard efforts by looking at the way we have had fifth graders get excited about getting to make a robot, or by watching newer competitors in FTC finally being able to grasp concepts that we taught them and then implementing them to help build their robot to become better. Needless to say, this year has been our best yet, and we're not planning on stopping here. We want to continue working on our robot and making the necessary improvements to make us a world class team.

5.2 Things for Next Year

While this season has been phenomenal, and our best one yet, it is important to note that we are continually looking to the future, and we would like to recognize some practices, actions, and changes that we have made this year that we have liked as a team, and will continue to include for the future. These practices include:

1. Getting involved with our school's exchange programs. We have learned so much from the exchange student on our team, Leonhard Grillmeyer, who's from Germany. He has been such a fun and interactive character to be around, and as a whole, our team agreed that we liked the idea of including more exchange students in our program and team for future years.
2. Taking a passive but engaging role in leadership. This year was interesting for us, because the original team captain decided to quit robotics, so we had to have a team vote to decide who the team captain would be. It came down to a tie with Brian and Logan both being selected. Throughout the course of the rest of the year, they both tried to keep their teammates involved while also doing the work themselves which helped influence a stronger work ethic this year.
3. Recruiting younger members. This year we got Max Polson on our team, and we have been so lucky for it. Max is extremely knowledgeable with engineering and math, and has been a major asset to both our head engineer Preston, and to the other engineers around him.
4. Continuing our YouTube channel. At the end of the 2018-2019 school year, Brian and Logan created some videos for a school project on robotics basics. They didn't get that many views until around October of 2019, and they have reached over 1,800 views on the team channel up this point.
5. Getting even more involved with Outreach. We have been extremely lucky to be able to go out and get involved with Hucrest Elementary School and help show them our robot and teach them how to build LEGO robots using Mindstorm parts which were graciously funded by our business partner Umpqua STEAMHub and FLL Guidelines as rules. We have enjoyed teaching these fifth graders, and we plan on doing this more next year, and possibly going to more schools to help run this program.

Engineering Content

Engineering progress

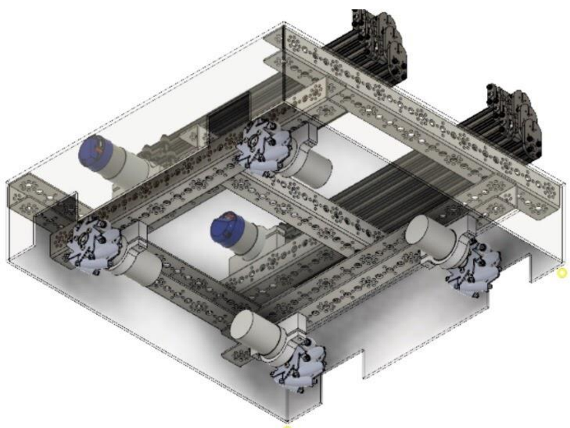
By the Aries engineering department:
Preston, Max, Wayne, Hayden, and Leonhard

(Also refer to the programming notebook by Brian for more details on other aspects of our robot.)

November 18th

The first step to coming up with a winning robot was making a good design.

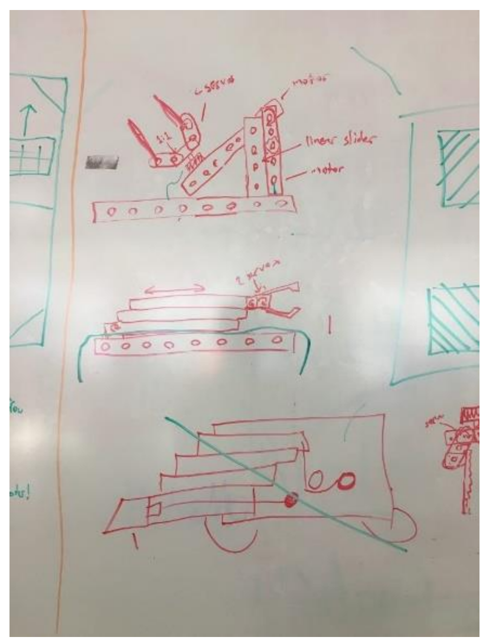
November 22nd



Chassis from last year

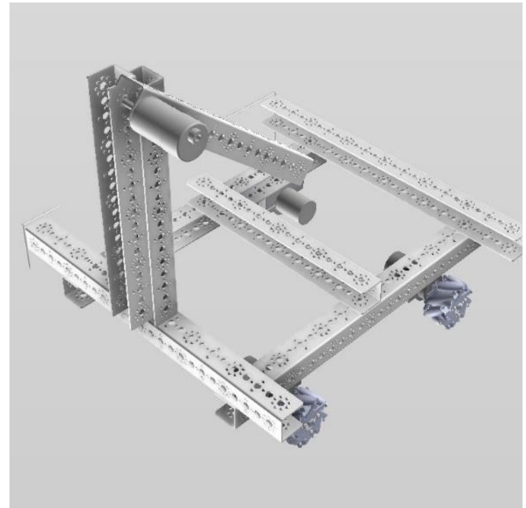
We had many meetings to discuss what each team member's ideas and goals were for the robot. The first decision made by the team was to keep the same chassis as last year, but take off the Lexan covering. The team decided this so that we could save time, because we felt that there wouldn't be a need this competition to have a different looking chassis. The chassis that was built last year had four mecanum wheels, connected by a basic tetrix frame. Each wheel was directly connected to 20:1 motors. This chassis seemed like it would work with any design we might come up with, and now we could focus on finding ways to properly design and build successful attachments to the robot for this competition.

Eventually the robot was narrowed down to three choices to vote on, and it looked like this:

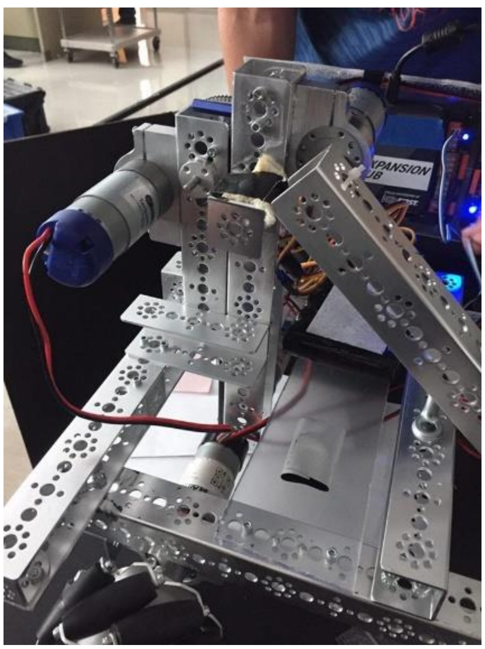


The team ended up voting on the robot with a vertical rack and pinion system and claw

This was robot version 1, as designed by Jesse:



The linear slider fit under 14 inches, but we realized after it was built that it would only get us 2 blocks high at max. This was a little disappointing, but we figured 2 would be fine for meet 0, when most other robots would be push-bots anyway.

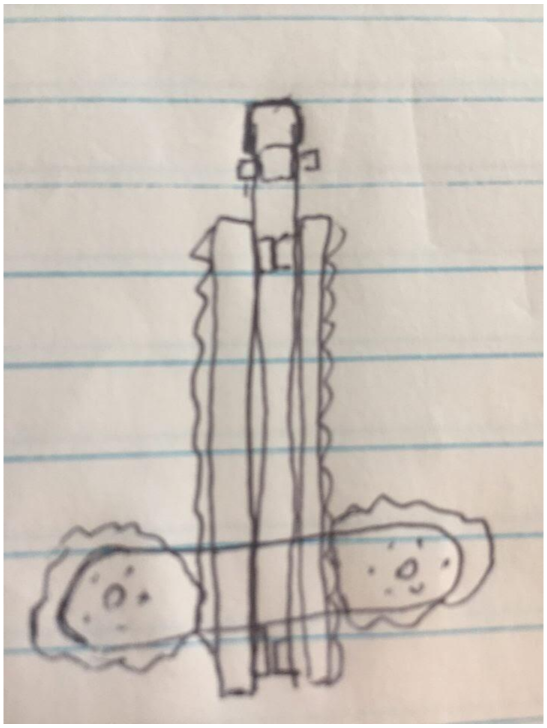


We had to start getting ready another design, though, if we wanted to have any success in later competitions. At meet 0 we got 3rd place, and did fairly well with V1, but noticed many problems that needed to be fixed, such as the complicated claw, the height of the linear slider, and other classic linear slider problems like the weak axles stripping inside the pinion gear.

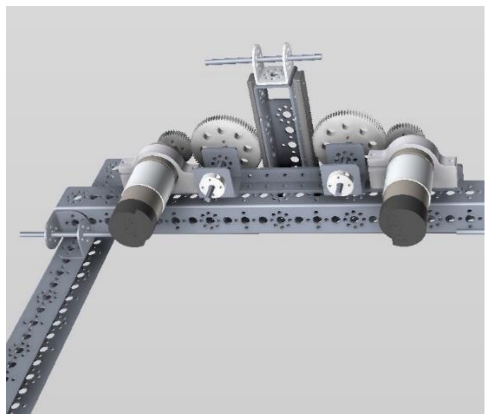
Directly after meet 0, the linear slider and the claw were taken off of the robot, and a design primarily made by Preston was presented to the team. It included an x-rail cascading lift system, and a simpler claw that only had one pivot instead of 2. As the x-rail was lifted up, it would also extend to get it higher and farther into the build plate. It would be lifted up by something near the middle of the robot, so that the leverage would make it easier to lift the x-rail. The team unanimously agreed to proceed with Preston's idea of V2, and Jesse proceeded to design V2 in Autodesk Fusion. In order to save time building and allow more time before meet 1, the engineers built V2 as it was being designed. The x-rail was added to the prebuilt chassis, and right away we realized there were things that had to be addressed before competing with V2.

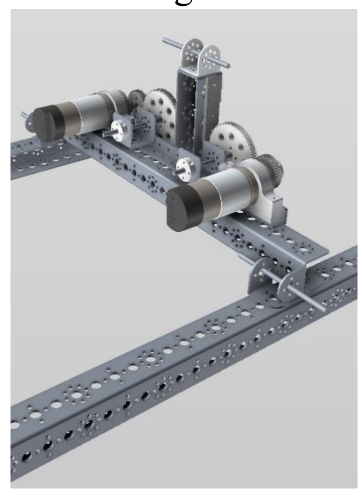
1. We had to find a way to easily, efficiently, and effectively move the x-rail up that also supported the weight of the leverage of 4 feet of metal.
2. The space to build a lift system was limited, since it had to fit under the 14 inch sky bridge
3. The entire system needed to be simple to make it easier for the driver, since we only use one driver

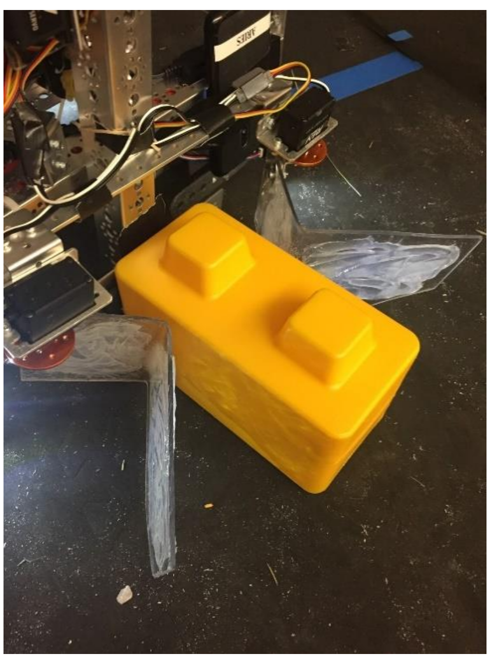
November 27th



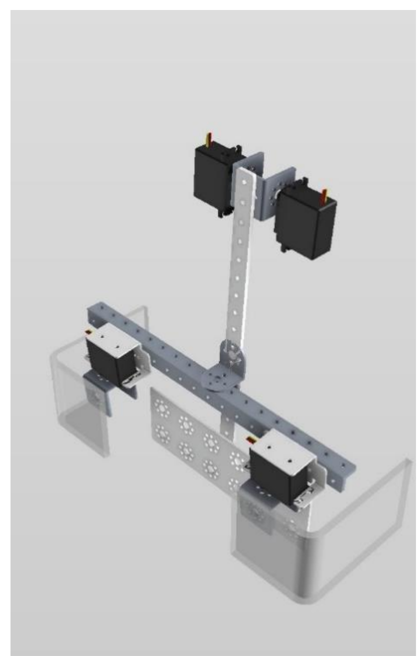
The next 2 weeks of robotics after school was spent trying to solve these problems, and eventually we put together a double rack and pinion system with two 60:1 motors geared from a medium tetrix gear to a small tetrix gear. This would slide along the bottom of the x-rail as it raised up, and when the x-rail was fully extended, had the ability to get tall enough to stack 5 blocks high.





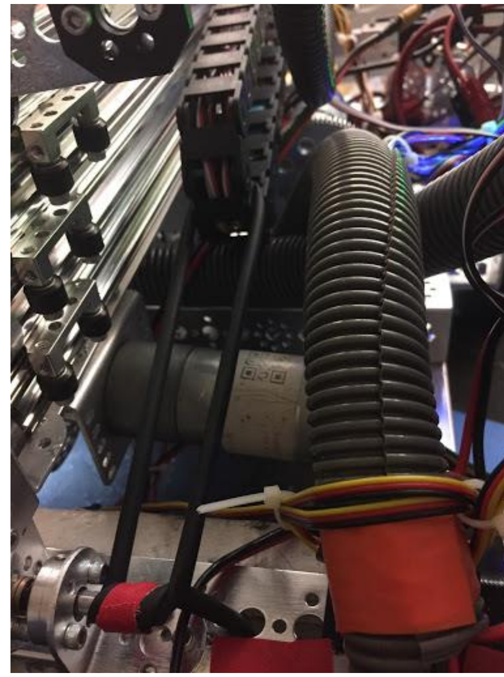


After we had solved these problems and made the arm raise, it was a fairly simple task to attach the claw/grabber. The claw was a fixed bar on the side that was attached with two REV Smart Servos, with two bars with Lexan attached with two Smart Servos to the fixed bar. We used foam from an old playing field mat for more grip on the blocks.



December 4th:

This is what the finished robot looked like for meet 1. It had the capability to stack 5 blocks high, and had a full autonomous. Knowing that we probably had the most advanced robot there, and that most teams in our league couldn't stack blocks yet, we had high hopes for meet 1. At meet 1 the highest we stacked was 2 blocks high, and we realized that we need to be much faster at intaking in order to succeed later on. We also needed to have a way to grab blocks off the wall. The next 3 team meetings were dedicated to discussing these problems and how we could make an improvement on the robot that wouldn't take a long time, but would still be effective. Some of the team, though, such as Preston and Jesse, didn't want to change the robot design at all right now, saying that we could improve more if we had more driver practice and better programming.



After about 3 meetings, the team decided on a simple addition of an orange flexible wheel, which was primarily an idea of Brian. Preston and Jesse were still opposed to the idea, but everyone else agreed with Brian and the engineers added the orange wheel on the clamp as follows:

December 6th

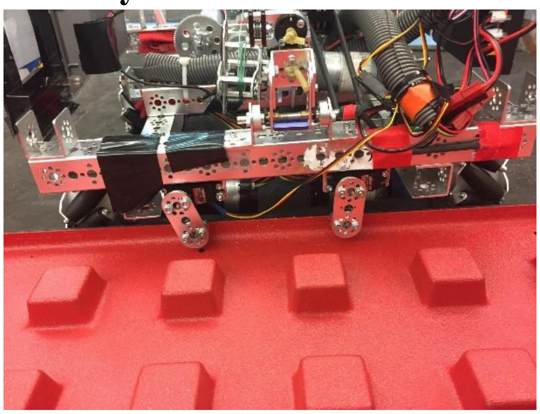
We built and tested the orange wheel claw design for about a week, and noticed that it added a lot of weight for the arm and made wiring complicated. When drivers were testing this claw, they noticed that it was very complicated to drive, and was not an improvement to the last simple claw design. We decided to take off this claw and we're back to the drawing board with a better claw design we needed to make.

December 12th - Wayne

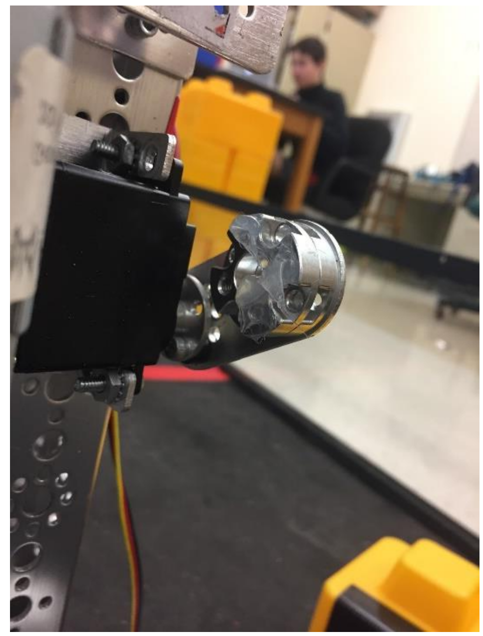
Due to the lack of efficiency of the foam on the claw, it was replaced with silicone for a better grip and to reduce the size of the clamping mechanism. Because the width of the claw was reduced, the Lexan claws were moved inward so that blocks could be gripped on to much better. Surgical tubing was installed to pull back the wire chain, because the engineering division of the team discovered that whenever the claw arm was extended, the wire chain would get caught, prohibiting further movement of the claw arm.

December 12 - Max
I'm the newest engineer, so I have been learning the basics of building and designing a robot. Since this is my first year, I have made a lot of mistakes that have led to learning experiences, such as fixing the claw alignment and then accidentally breaking it, which led to me learning about how to put together and build things so that they won't break as easily. Something else I have been doing recently is helping with the wiring, which is one our teams biggest engineering weaknesses.

January 10- Preston

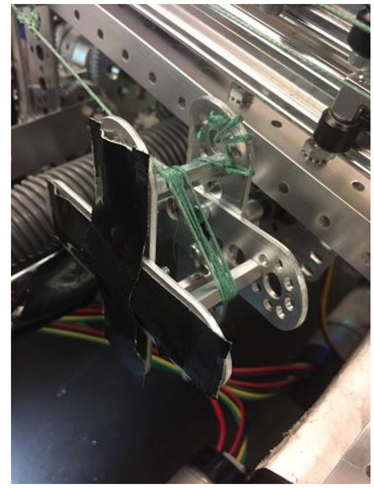


Today Brian, our head programmer, noticed that the block has been slipping from the claw a lot lately, and I thought it might be helped by adding more silicone, which would renew the claw's stickiness and ability to pick up blocks no matter how they were positioned. I added more silicone and it was able to pick up the blocks again. Another thing that happened in the engineering department was adding silicone to the back of the clips that move the build plate, which help autonomous because it makes it so that the build plate doesn't slide when the robot turns during autonomous, instead the build plate sticks to the same place on the clips for the entire turn until it unclips.

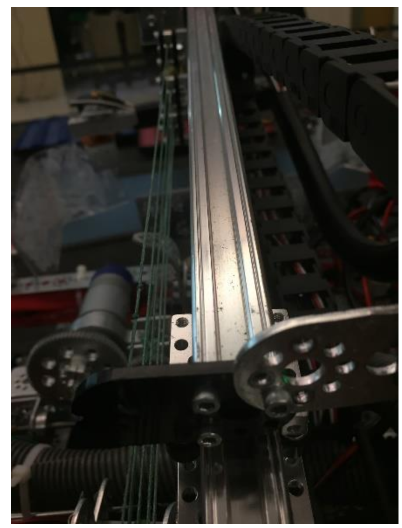


January 19 - Preston
Yesterday was league meet, and we exceeded all expectations by getting in the 1st place alliance, and earning the 1st place inspire award! Also, our robot had the highest scoring match, and the highest point average of 50.2. I am very happy with the progress we've made on the robot and am excited to see what is accomplished going on to super qualifiers and state. There are a few things engineering-wise, though, that need to be addressed after watching the robot this competition.

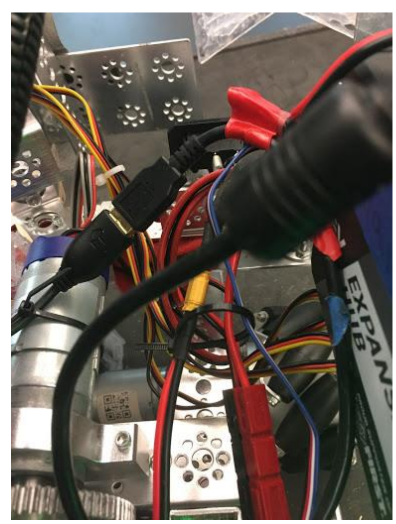
1. The spool needs to be redesigned and built to be more sturdy and smooth. During some of our matches the x-rail would extend, then the twine would get caught on the spool, and the x-rail wouldn't come back to its original position like it's supposed to. Our plan to improve this in the upcoming weeks is to 3D print a new spook that is smooth and custom fit to our robots specific design, or we can buy a spool to use for the x-rail. These ideas will greatly improve upon a spool I built out of tetrix pieces during the beginning of the season.



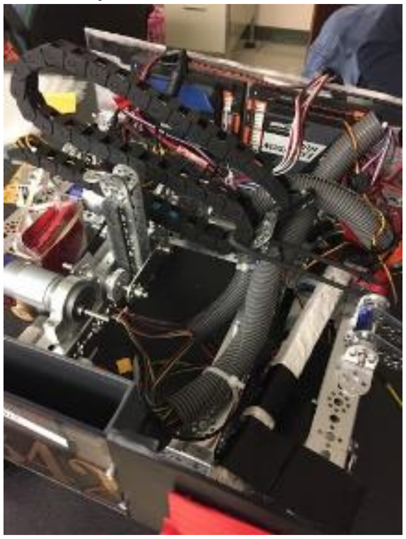
2. The x-rail needs to be taken apart, and all of the pieces inside need to be lubricated so that it can run more smoothly. During some of our matches the x-rail gets stuck, but in all matches it's clear that the x-rail is not going back down smoothly, as is apparent by the jerking as it unextends.



3. Wires need to be insulated and protected better. In one of our matches this competition, our robot was unable to compete because it got ESD, or electro-static-discharge, which is where the robot disconnects and can't function as soon as it touches another robot or a metal part is the field. This problem was experienced by TNT last year, and has also happened to Taurus this year, so we have had opportunities to learn about it and how to prevent it from happening.

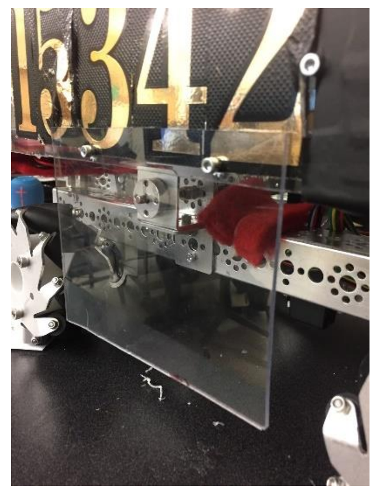


January 23 - Preston



Today, Logan and I started the process of taking apart the x-rail and finding out why it is not moving smoothly, and replacing the spool with one designed especially for our robot. We took off the x-rail and spool, making sure to keep all of the wires plugged in and trying not to mess up the wire chain at all. After taking it off of the robot, we dismantled the linear slider one section at a time, and found that 2 of the bearings were dented, so we got other bearings from another x-rail that we aren't using and replaced the dented ones. This was the only problem we found with the x-rail, and now we hope to design and 3-d print a new spool that will be smoother.

Another thing that we have done is add more Lexan underneath the Lexan we already have so that we can put our sponsors' logos on the robot for people to see during competition. We also started a 3D print of the spool that we found on thingiverse, but the print malfunctioned after everyone left, so on Monday we will be starting the print again.



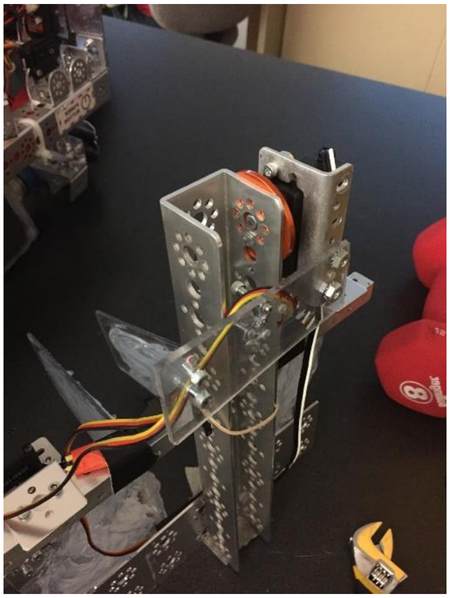
Friday January 24 - Wayne

Cut one of the Lexan wall extensions for sponsors to be put in. Also helped drill some holes in it. We did this to prevent items such as the game stones from being partially trapped under the robot.

Monday January 27 - Wayne

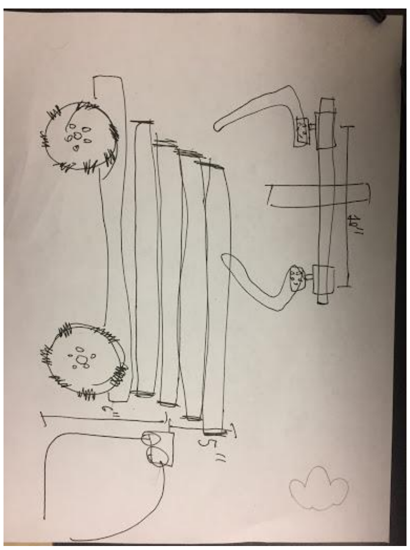
Helped Preston reattach some of the arm; moved the metal piece with a wire tube on top of it closer to the front, screwed on the arm and the part that holds it up on the rear end of the robot, and put a screw to attach the arm to the lifting mechanism. Moved the metal piece with the wire tube because it hindered the motor from being attached with the linear slider.

Monday February 3- Preston



Today we had a team meeting to discuss robot progress, team involvement, and sponsors. It was presumed from the meeting that our current version was not up to par with what we would be competing with at state, which is our goal for this season. We want to change the robot, but won't have time before the next competition, so we will plan on keeping the current robot, but with minor adjustments and improvements on stability and security, and then making prototypes and designs for changes we will do after this next competition. Our ideas for the new version includes making the x-rail fold up backward to a vertical position, the driver only has to focus on horizontal movement when stacking and hopefully making our robot faster. Today we are remaking the attachment from the claw to the x-rail, and securing the claw on better so it doesn't wobble as much.

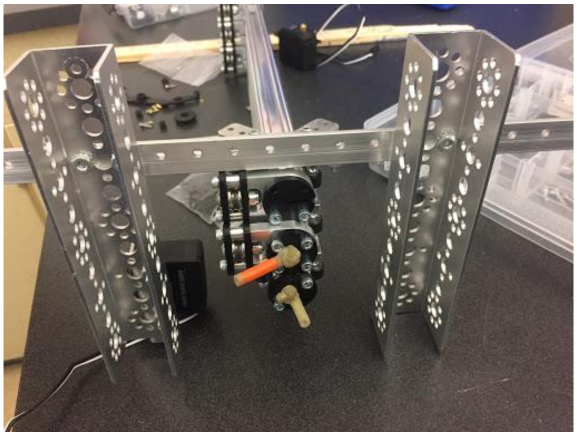
February 8 - Preston



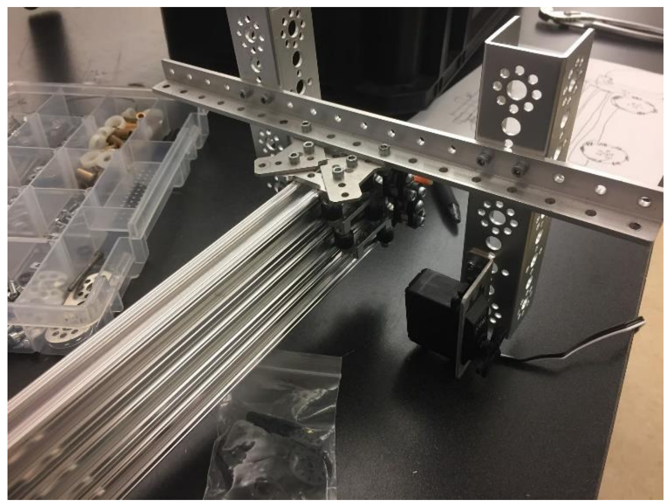
Today we started working on the prototype arm for our new design. We are making the prototype of where the claw attaches to the x-rail on a different x-rail that we're not currently using so that the rest of our team can use the fully constructed robot for programming and presenting to students for outreach. We also used the robot for the scrimmage today that we hosted for seven teams in SOAR that are advancing to super-qualifiers. Yesterday we were able to start building the prototype for attaching the claw onto a new slider system that brings the claw up when the arm will be vertical. Making this prototype involved taking measurements of the height of the x-rail and of how high the top of the claw has to be from the mat.

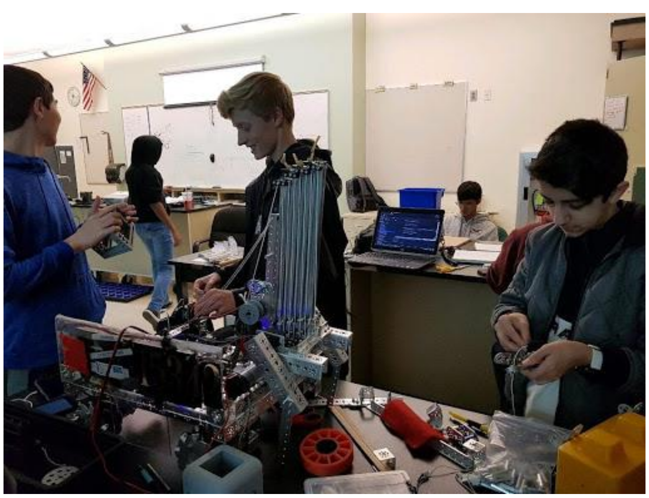
We also constructed the slider that will be attached to the claw and to the top rail of the x-rail. One of the main problems we encountered was the compatibility of our tetrix parts with the x-rail, which is based off of a tetrix MAX system. It is frustrating when the two systems don't fit together very well, but we managed to figure out a way to attach the tetrix channels using brackets and angles.

February 18 - Preston



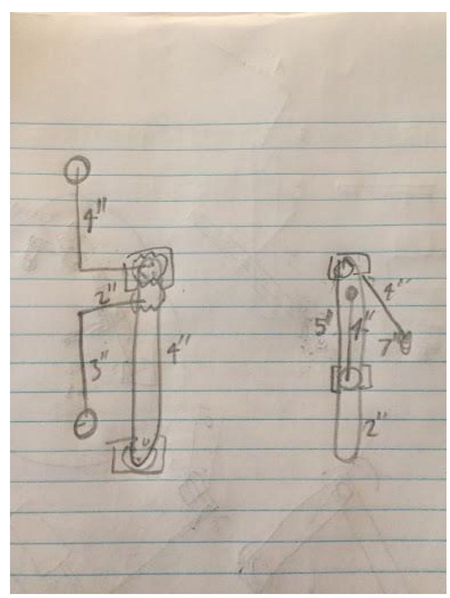
We had super-qualifiers last Saturday, and there were a few things to be learned from that. Our robot did very badly, placing us in 21st place before awards. The only way that we were able to get the inspire award was because of our judging and previous success at other meets. Part of our problem is that we had bad allies, but the other part is that our robot is slow and hard to control. With the x-rail having to go up and out, it makes it hard for the driver to stack because of how many variables are in play. Our claw and x-rail system also makes it hard to do very much in autonomous because of the size and speed. We have been prototyping and designing for version 4, which we hope will be done by two weeks from now so that it will be ready with a full autonomous for state. With version 4 we hope to solve the stacking problem by having the x-rail go fully vertical so that the driver can stack faster and easier, and we will be creating a claw specifically for autonomous that will clamp down on the blocks, lift them up, then let go when the robot gets to the build plate. We also have discussed moving the build plate clips so that they are on the same side of the robot as the autonomous claw so that we can drop the block and move the foundation at the same time.





February 21 – Preston

Today we invited Eli from Mini-Maniacs to help us with our new claw nubs of the block from the top, and we ended up getting a lot done. The claw is finished, we just need to attach it to

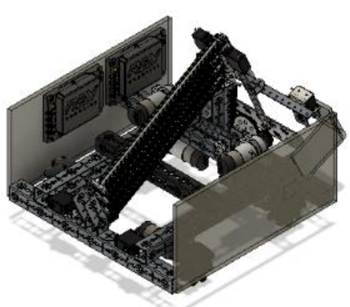


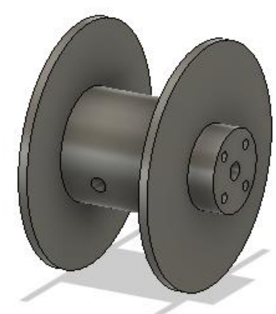
the arm somehow. I am also working on the autonomous claw, which is hard to design because it has to be able to fit inside our robot during teleop and also reach out far enough to grab the block easily in autonomous. I have made some measurements for two designs, and I think that 3 d printing the parts would be the best option because then we can make it just the right dimensions to be able to fit on the side of our robot.

February 23-Preston

Today we had a build day that I was not able to come to, but I have been informed by some other people on our team that we have encountered a couple problems already with the new design. One of them is the support of the arm when it is extended vertically with the 2 60:1 motors. The weight and stability is so hard and so much work for the motors that the motors are getting hot and draining the battery very fast. Our solution to this is to try and design a locking mechanism to lock the arm in place in order to help the motors keep it upright.

February 29- Preston

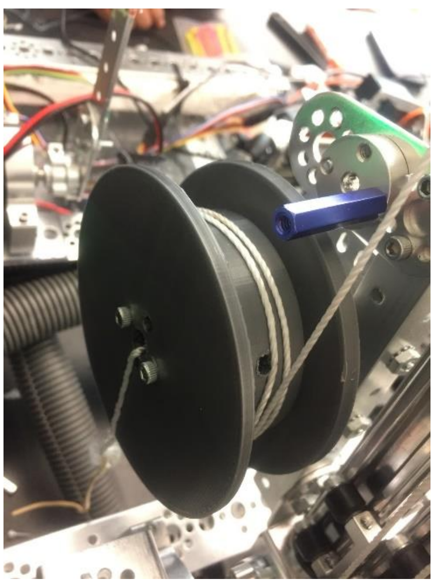


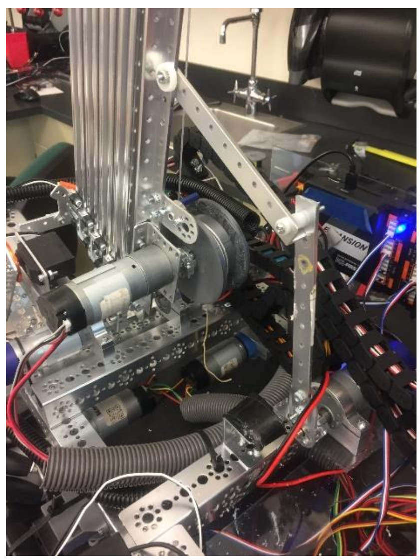


This week we've gotten a lot of work done on the robot because the goal is to get it done by the end of the week, which is today. We've made a way for the arm to lock in a vertical position thanks to help from someone from another team, and we've also been able to finish the autonomous arm. Mainly our focus now is on wiring and the tape measure idea. We are struggling with wiring because of limited space on the robot and limited supplies for wiring purposes. Next week we should hopefully be programming and having driver practices. Something else we have done is replaced the spool for the x-rail with a bigger one that we 3D printed we could extend the arm faster. Here are some pictures of our robots progress as of today:

March 7th – Max

Today we have encountered a few servo problems with the smart rev servo. Preston and I started to try and fix the smart rev servo's problem. At first, it was just the servo making different noises when it was turning or moving. As it turns out, the case was making noises when it was being set on at different heights from the gears. Brian told me about how there are two versions of the servo and how one is older than the other and how they use different types of gears than the old ones. I spent about fifteen minutes tinkering with the servo's gears and found out that no matter what, I can fix the servo because the walls of the case are too close to the gears causing them to ride up against the walls of the case. When the gears rub against the case, it will cause the servo to not work. As you can see in the second picture, we added a linkage to help stabilize the robot. When the motor rotates in a certain direction, the linkage will move, thus changing the angle of the linear slide, and hen securing itself into the robot, allowing for the arm to remain unmoved unless acted upon by the motor once again.





Programming Section

Documented Changes to Aries Programming Files

Brian Powell

The engineering process can be used in more ways than just physical design and engineering. This year, I decided to commit some time to documenting my engineering process of the main programming files. First, I will present the current files, and their version number. The number preceding the decimal represents major changes to the programming file, whereas, the number following the decimal displays the number of minor changes that have been made to the file. A major change by my definition would be a full redesign of the program, where the majority of the important code is entirely rewritten, and a minor change should be shown as a quick bug fix, small amount of code added on, or a small portion of code overwritten. After each version is presented, I will further extend to write down my ideas for improvement, with comments left in the code to explain why I think each idea is worth experimenting with.

Teleop 1.0 (October 6, 2019):

This is the original program from last year written to be compatible with mecanum wheels, allowing us to move the robot in all directions. The file has its flaws, for instance, the robot should be using logarithmic controls for a better "feel" of speed, the robot's rotation affects the translation of the robot with the left joystick, and the robot does not travel at full speed when moving diagonally or travels over its maximum speed. The robot has not been built yet, so this except from the program is based on the assumption that the robot will be built as it is designed:

```
//The amount of power in each wheel, set by the left and right sticks on the controller
leftBack.setPower((gamepad1.left_stick_y - gamepad1.left_stick_x - gamepad1.right_stick_x)*(-speedAjust/10));
rightBack.setPower((gamepad1.left_stick_y - gamepad1.left_stick_x + gamepad1.right_stick_x)*(-speedAjust/10));
leftFront.setPower((gamepad1.left_stick_y + gamepad1.left_stick_x - gamepad1.right_stick_x)*(-speedAjust/10));
rightFront.setPower((gamepad1.left_stick_y + gamepad1.left_stick_x + gamepad1.right_stick_x)*(-speedAjust/10));
```

Autonomous 1.0 (October 6th, 2019):

The original program is just a test file, it uses tensorflow and vuforia in order to find and move to the stones. The file uses telemetry to address how the robot should move to the stone, with instructions such as: move left, or move forward:

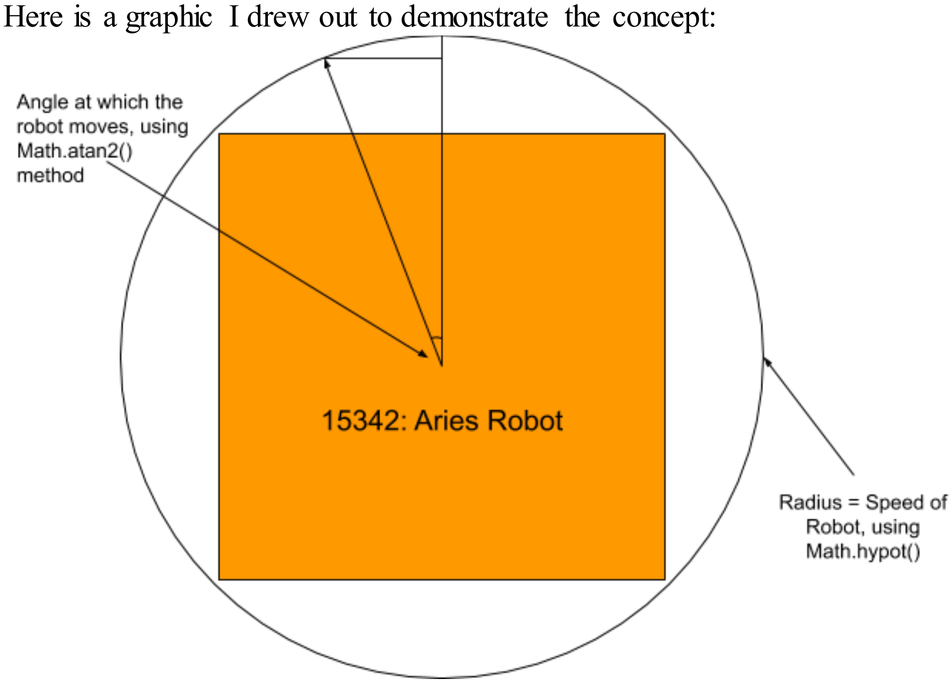
```
currentTime.reset();
if (opModeIsActive()) {
   List<Recognition> updatedRecognitions = tfod.getUpdatedRecognitions();
   if (updatedRecognitions != null) {
      telemetry.addData("# Object Detected", updatedRecognitions.size());
      // step through the list of recognitions and display boundary info.
      for (Recognition recognition : updatedRecognitions) {
         telemetry.addData("label", recognition.getLabel());
         telemetry.addData("left, top, right, bottom", "%.03f , %.03f , %.03f , %.03f", recognition.getLeft(), recognition.getTop(),
recognition.getRight(), recognition.getBottom());
         telemetry.addData("Go",((recognition.getLeft() > 0) ? "Move Right" : "Move Left") + ((recognition.getRight() - recognition.getLeft() <
900) ? "Move Forward" : "Move Backwards") + (recognition.getRight() - recognition.getLeft()));
      }
      telemetry.update();
   }
}
```

Improvement ideas (October 6th, 2019):

Idea 1: The teleop should be redesigned in order for it to fix the three flaws in the program, I addressed above. The program I have written below, after some research, uses trigonometry to answer two of the three problems, although, I think the rotation of the robot affecting the translation should be able to be added easily to this program, if it functions properly:

```
//Uses the hypothesis of left joystick and right joystick to calculate the magnitude (speed) of the robot
double radius = Math.hypot(gamepad1.left_stick_x, gamepad1.left_stick_y);


//Uses arc tangent (inverse tangent) to find how we need to rotate the power of the robot in order to move in the direction we want it to
double robotAngle = Math.atan2(gamepad1.left_stick_y, gamepad1.left_stick_x) - Math.PI / 4;


//Note: Math.pow(value, 2) may be a method for achieving logarithmic controls, but negative numbers will become positive!


//multiply the direction we need to travel in by the speed to create the power to each wheel, and also the amount that we are turning to get the power of the robot's wheels
leftFront.setPower(radius * Math.cos(robotAngle) + gamepad1.right_stick_x);
rightFront.setPower(radius * Math.sin(robotAngle) - gamepad1.right_stick_x);
leftBack.setPower(radius * Math.sin(robotAngle) + gamepad1.right_stick_x);
rightBack.setPower(radius * Math.cos(robotAngle) - gamepad1.right_stick_x);
```

Here is a graphic I drew out to demonstrate the concept:



Perhaps, it is also possible to store the current rotation of the robot and factor that into the equation in order to allow the driver to translate the robot and rotate, without affecting the translation of the robot.

Idea 2: It's time to add the motors into the autonomous program. Although the robot is not built yet, the programming should be ahead of the engineers so that the instant they are done building, I can begin testing. The program needs to be able to initialize all the motor variables, and there should be a motor encoder function if I need it. I do not think I will need the motor encoder function because my goal this year is to use sensors that compare our location on the field. If the wheels slip on the mat, the motor encoders will not know. But if the robot is running until it "sees" a color, it will reach that point no matter how long it takes, unless blocked by another object. To extend, the robot should be able to adapt to objects on the mat that have been moved intentionally or unintentionally, or another robot if it is in the way. This will make our robot more "intelligent" and allow us to work with any alliance. This may take a while to perfect, but it is important to begin on this project as soon as possible.

Improvement ideas (October 7th, 2019):

Addressing Idea 2: I wrote a draft of the motor encoder function for running to a certain position, it requires five inputs: The four motor powers, and the distance to be traveled by the motors. The function returns void, and regulates the entire distance traveled between the starting and ending point.

```
private void encoderDrive(double power1, double power2, double power3, double power4, double distance) {
  //RUN_TO_POSITION allows the robots motors to remain "busy" until they finish running to a certain point
  leftFront.setMode(DcMotor.RunMode.RUN_TO_POSITION);
  rightFront.setMode(DcMotor.RunMode.RUN_TO_POSITION);
  //etc...


  leftFront.setTargetPosition((int)Math.round(distance*power1) +
       leftFront.getCurrentPosition());


  rightFront.setTargetPosition((int)Math.round(distance*power2) +
       rightFront.getCurrentPosition());
  //etc...


  leftFront.setPower(power1);
  rightFront.setPower(power2);
  //etc...


  while(leftFront.isBusy() && leftBack.isBusy() && rightFront.isBusy() && rightBack.isBusy()) {
    //Wait for motors to reach their positions...
  }


  //motors have reached their positions, set their power to zero
  leftFront.setPower(0);
  rightFront.setPower(0);
  //etc...


  //run with encoder only uses the encoders to regulate the speed of the motors
  leftFront.setMode(DcMotor.RunMode.RUN_USING_ENCODER);
```

```
  rightFront.setMode(DcMotor.RunMode.RUN_USING_ENCODER);
  //etc…
}
```

Progress (October 9th, 2019):

Today, my main focus was to test all the code I wrote in the previous days and make any minor adjustments as needed. For the most part, I was constructing the teleop, but I did add the motor configuration and the first stage to the autonomous program. The teleop plan is to autotomize every preset value possible. For example, if an arm of the robot needs to be lifted to a set position, rather than having the driver move the arm upwards until they reach the position, they could just press a single button, and the whole arm would position itself exactly at that location. For the autonomous, the plan is to separate the 30s into different stages: "find" or move to and detect the first skystone block, "pickup" or pick up the stone, "move" or move to the building half of the field, "reposition" or move the building plate *if necessary*, "place" position stones and skystones on build plate, "return" or move back to the first half of the field, then repeat the steps until the last 5 seconds. The robot will then navigate under the alliance specific bridge in the "navigate" stage of the autonomous program. This program will vary based on location, of course, and will require a lot of sensors to navigate. After completing this strategy, I began to build the first stage of the autonomous, and wrote a function to move the robot, either based on encoders, or just on sensory information. These are functions `move` and `encoderDrive`, the first function moves the robot as the controller would, with three inputs, translate x, translate y, and rotate, the second is described above as an idea, and ran without any errors in the compiler. The second function runs based on the encoder position, which is more precise than time, but still not preferable to sensors such as tensorflow and vuforia, color, or proximity, which can all determine the robots location compared to the position of other objects. Encoders only allow us to see how many rotations the robot has traveled, which could vary based on the wheels slipping, collision with an unexpected object, or friction. The `move` function can be placed before a while loop that checks for a sensor, which stops the motors after they have reached the desired location. This is because the `move` function, by itself, runs infinitely at the set power until it is stopped.

Teleop 2.0 (October 9th, 2019):

I completed what I think will be a controller with logarithmic controls and finished perfecting the controls for the mecanum wheels. The reason I changed the version up a major version is because the majority of the previous code was rewritten in the process. Because the main ideas of the previous teleop were mostly removed with newer concepts, I decided this was a decent reason to increase the major version number. I created my own squaring function that maintains the negative or positive state for the number for the logarithmic controls:

```
//squares the power for logarithmic controls
private double logControl(double power) {
  return ((power < 0) ? -power * power : power * power);
}
```

The code below controls whether or not the driver wants to translate the robot relative to the field, or relative to the angle of the robot, theoretically. This code has been tested for errors in android studio, but has not been run on the robot yet.

```java
if(gamepad1.a) {
      if(mode.equals("rotate")) {
         mode = "translate";
      } else {
         mode = "rotate";
      }
   }
   if(mode.equals("rotate")) {
      //Uses the hypothesis of left joystick and right joystick to calculate the magnitude (speed) of the robot
      double radius = Math.hypot(gamepad1.left_stick_x, gamepad1.left_stick_y);



      //Uses arc tangent (inverse tangent) to find how we need to rotate the power of the robot in order to move in the direction we want it to
      double robotAngle = Math.atan2(gamepad1.left_stick_y, gamepad1.left_stick_x) - Math.PI / 4;



      //Note: Math.pow(value, 2) may be a method for achieving logarithmic controls, but negative numbers will become positive!



      //multiply the direction we need to travel in by the speed to create the power to each wheel, and also the amount that we are turning to get the
power of the robot's wheels
      leftFront.setPower(logControl(radius * Math.cos(robotAngle) + gamepad1.right_stick_x));
      rightFront.setPower(logControl(radius * Math.sin(robotAngle) - gamepad1.right_stick_x));
      leftBack.setPower(logControl(radius * Math.sin(robotAngle) + gamepad1.right_stick_x));
      rightBack.setPower(logControl(radius * Math.cos(robotAngle) - gamepad1.right_stick_x));
   } else {
      //Uses the hypothesis of left joystick and right joystick to calculate the magnitude (speed) of the robot
      double radius = Math.hypot(gamepad1.left_stick_x, gamepad1.left_stick_y);



      //Uses arc tangent (inverse tangent) to find how we need to rotate the power of the robot in order to move in the direction we want it to
      double robotAngle = Math.atan2(gamepad1.left_stick_y, gamepad1.left_stick_x) - Math.PI / 4;
      currentRotation += robotAngle;
      //Note: Math.pow(value, 2) may be a method for achieving logarithmic controls, but negative numbers will become positive!



      //multiply the direction we need to travel in by the speed to create the power to each wheel, and also the amount that we are turning to get the
power of the robot's wheels
      leftFront.setPower(logControl(radius * Math.cos(currentRotation) + gamepad1.right_stick_x));
      rightFront.setPower(logControl(radius * Math.sin(currentRotation) - gamepad1.right_stick_x));
      leftBack.setPower(logControl(radius * Math.sin(currentRotation) + gamepad1.right_stick_x));
      rightBack.setPower(logControl(radius * Math.cos(currentRotation) - gamepad1.right_stick_x));
   }
}
```

Ideas (October 10, 2019):

GitHub? It just occurred to me that we should be storing our code online, where it is safe from the restrictions of hardware and has plenty of storage to store previous code for future use, or to undo an action that is not productive, or worse. Whether or not the project should be public is a topic I should bring up during the next team meeting, tomorrow (build day from 9am - 4pm)! GitHub, or at least some type of git version control should be used in order to back up the code, and I already have a GitHub account setup for the robotics website which is being programmed by myself as well.

Problems (October 10th, 2019):

After testing a few programs on the phones, I noticed that the connection between the controllers and the phone is very weak, in fact, the controller must be held in an exact position in order to connect. The possible causes could be found in one or more of three places: the controller and controller wire, the adapter for the USB controller wire to the micro USB to connect to the phone, or the phone outlet that allows the adapter to attach to the phone. I have tried multiple different controllers, keeping the other two factors constant, with no different result. After that attempt, I tried several different adapters. Some of the adapters were a little more successful than the others, although the results varied only slightly. This would lead me to believe the cause is the phone outlet, but I also observed that the connection strength increases when the connection between the controller and adapter is loosened. This is a case I will hand over to one of the newer engineers in tomorrow's meeting for further investigation. It is important that this problem be taken seriously, because a failure in the remote connection during competition could result in devastating consequences. It may be necessary for us to purchase a new part, which could lead to a long wait for shipping and processing through our school, and should be solved as soon as possible.

Teleop 2.1 (October 11th, 2019):

Minor changes were made to switch the turning from positive to negative, and variables were moved so they could be local. The robot was built today, and our first test of teleop was mostly successful! However, flaws were found in the "translate" mode for teleop, and as of now, I am not sure of what to change to improve it. I know the problem is because the robot is not actually calculating its angle of rotation, but the angle of its traveling instead. However, I was not able to think of a method for how to calculate the distance rotated. Perhaps a gyro sensor would be able to tell us our rotation more accurately than attempting to calculate it.

Progress (October 11th, 2019):

I set up our code in github; it can be found here: https://github.com/TheAmazingBrianPowell/AriesTeamCode. As well as backing up and making our code open source, I also configured and updated android studio, so that it can test our tensorflow code correctly, and also so that it is compatible with the newest FTC app. I did no further autonomous programming today, although I did instruct one of newer team members, Leonhard, on programming in Java for FTC. The servo controls have been written out, but I have not been able to test them yet. I did test the driving for the teleop program today, and I made minor adjustments to fix small bugs in the controls. Meanwhile, the engineering team built nearly the whole robot, which means that driver tests can begin soon. This year, we will try to condense as much of the controls into one controller as possible, so we can just have one driver, and therefore reduce the amount of communication and coordination required between drivers.

Problems and Solutions (October 17th, 2019):

Recently, my main goal in robotics has been fine tuning the teleop program so the drivers and engineers can test all the functions of the robot. I noticed that, despite the logarithmic controls, the robot was still incredibly difficult to control with fine adjustments. Because of this, I think I will rewrite a similar program as last year's which has a top speed control in order to scale the speeds to the desired range the driver wants. During the few autonomous tests I was able to complete, I was able to determine several flaws in the program. Mainly, the robot was getting stuck in a loop, causing the robot to find the skystone, but continue to plow right through it, an error we would not want to replicate at competition. There were several other loop problems that I managed to fix, but I have not yet determined the cause of the bug in the find stage of autonomous. I also noticed that the color sensor, which the engineers mounted onto the bottom of our robot, was not responding, and its light remained off. The tensorflow had some problems detecting all the blocks on the field, especially the further the robot was from the blocks, this may mean that we will have to have our robot move in towards the blocks and then translate sideways to detect all the block's position, an idea I think is very inefficient. Another concerning problem was related to toggling servo positions with buttons. Toggling is a very important function because it prevents the robot from interpreting one button press as hundreds of button presses. However, in order to achieve this, I was forced to use a while loop, that could potentially stop all other functions on the robot if the button is held down. It seems like I could use an if statement as a substitute for the while statement in this situation.

Solutions (October 17th 2019):

It just occurred to me why the while loop for toggling controls works, and how to rewrite it as a more efficient program. The problem is that the code measures a change in the state of the button, which compares its previous state to the current one, this happens twice: when the driver initially presses the button, and when the driver releases the button. Because this happens twice, the program just reverses what it did on the initial press of the button when the button is released, which is generally fast enough that the driver sees nothing happen at all. To fix this, I simply made sure I was only detecting when the previous state of the button was not pressed, and the current state is pressed, therefore we are detecting only the initial press of the button.

Teleop 2.3 (October 24. 2019):

Minor changes have occurred to teleop. I completed the controls for the intake system so the drivers can begin driving the robot. Unfortunately I was set back one day because I did not back up a file recently. On the other hand, it was fortunate that the majority of the file was saved on the internet because I did remember most of the changes I had made. In addition to making software minor adjustments, I also completed several hardware adjustments to the REV smart servos on the robot. We needed the smart servos to be set to noncontinuous rotation so that the driver did not have to worry about the exact amount of time that was needed to press down the claw release buttons. The two servos for the claw, or clamp, I managed to set successfully, while the smart servo that rotated the claw was not responding to the programing device. I will continue to work on getting the rotation servo to switch to a noncontinuous rotation servo so that the wires will not get tangled in the servo if it rotates.

Autonomous 1.3 (October 24, 2019):

After finishing the teleop, my main task has been completing the autonomous so that the robot is able to pick up the bricks using tensorflow. With each run, my program is getting closer, and closer stopping at the block, but there are some major setbacks. First of all, the phone needs to be angled just right, for the tensorflow to get the most "descriptive" view of the block. By "descriptive", I mean the view with the most points along different planes on the brick that result from viewing the block along an angle. This allows the program to view more of the block. Also, the tensorflow program does not always sense the blocks if they are too far away. This can sometimes be fixed by adjusting the lighting, or moving the robot a little. I will need to make sure in the code that if the robot does not sense any bricks, it moves forward to get a better view. There are more problems with the program sensing when to stop moving. The robot aligns itself perfectly in the x direction parallel to the brick, but has trouble knowing when it has gotten closer to the brick. I think this may have been caused by the movement in the sideways and forward directions at the same time rather than moving each one at a time.

Outreach (October 24th, 2019):

Although I am only documenting this today, there have been several instances where Aries has helped Fremont Middle School in programming. Yesterday they were at our club to observe and build their robot with us, and in the beginning of the season I also went to Fremont to set up their phones and instruct them on using Java with the robot. During the summer, I helped with a Lego robotics camp at the boys and girls club in Roseburg as well.

Autonomous 2.0 (October 30th):

In this time, I have been working on an effective method for finding and tracking the skystones and stones, and I have completely redesigned the code. My original idea for locating the blocks was entirely about using tensorflow in order to detect the blocks and their locations relative to our robot. Unfortunately, tensorflow was not accurate enough for our needs, and could not always "see" any of the blocks from the distance of the quarry to the wall. In other cases, tensorflow has assumed incorrect x and y values for the blocks, and even once thought that three bricks were one. To compromise, I devised a system using vuforia image recognition combined with a distance/color sensor in order to drive to the blocks, then move sideways in a scanning manner in order to detect the skystone. The distance sensor stops the robot from running into the blocks, where as vuforia looks for the images on the skystones. The color sensor had problems detecting differences between the skystone and the stone because of the dramatic lighting differences in different locations. This would prove disastrous at different meets with different lights. I completed the code, and it worked perfectly! Now I only need to pick up the block and move it after locating it. Here is a portion of the code which runs during the runOpMode():

```
//release the claw
clamp1.setPosition(0.5);
clamp2.setPosition(0.1);


//move until the robot is close to the blocks
move(0,-0.3,0);
while(opModeIsActive() && (sensorDistance.getDistance(DistanceUnit.CM) >= 40 ||
Double.isNaN(sensorDistance.getDistance(DistanceUnit.CM))));
move(0,0,0);


//move backwards at 0.2 power for 1000 encoder units
encoderDrive(0.2,0.2,0.2,0.2, 1000);


//move until the skystone is visible to the phone's camera
move(-0.1,0,0);
while(opModeIsActive() && !targetVisible) {
  targetVisible = false;
```

```
  for (VuforiaTrackable trackable : allTrackables) {
    if(((VuforiaTrackableDefaultListener)trackable.getListener()).isVisible()) {
      targetVisible = true;
      OpenGLMatrix robotLocationTransform = ((VuforiaTrackableDefaultListener)trackable.getListener()).getUpdatedRobotLocation();
      if(robotLocationTransform != null) {
        lastLocation = robotLocationTransform;
      }
      break;
    }
  }
}
move(0,0,0);


//Drive forwards for 1000 encoder units at 0.2 power
encoderDrive(-0.2,-0.2,-0.2,-0.2,1000);


//rotate the arm for intake
bigRotate.setPower(1);
sleep(1000);
bigRotate.setPower(0);


//rotate the claw
rotate.setPosition(1);
sleep(1000);
rotate.setPosition(0.5);


//clamp the claw onto the brick
clamp1.setPosition(-1);
clamp2.setPosition(0.5);
```

This code is very close to being able to pick up the block. The program definitely can consistently stop at the skystone. I believe I will be able to finish the autonomous before the meet with the four hour build day tomorrow.

Meet 0 Notes (November 2nd, 2019):

Aries placed 3rd in the meet today, with a highest score of 42. I did not decide to run the autonomous that used the sensors and found skystones because it earned us no points. Instead, I created a new autonomous and programmed it during the meet to move the build plate to score and then navigate to the line. The program was successful in navigating almost every time, and most of the time got the build plate, but sometimes just missed it by a few millimeters. I believe we did well for our first meet, but there is a lot to improve on in both engineering and programming. The team has decided to remove our current intake system and replace it with an x-rail system to stack roughly 6 blocks high, so I will have to wait for about a week and a half to program autonomous again. In the meantime, I will work on creating a smoother easier-to-work-with autonomous.

Teleop 3.0 (November 7th, 2019):

The engineering team has been working hard to complete the new robot design. Today was a no school day, and so our team gained access to the robotics room to work for five hours. In the time, the engineers completed better wiring, a pushbot idea that locks and aligns the brick to the robot, an x-rail that extends out, and can be moved up and down, and the beginnings of a clamp to lock on to the stones. Because of the significant changes to the robot, the programming also changed a major version with the robots increase in version. In programming, I added programming for the x-rail extension, and the rotating of the x-rail. I wrote the controls to use encoders so the drivers do not have to worry about extending past the physical limit of the x-rail and breaking the string, or the limit on the linear slider that rotates the x-rail. I also programmed a zero position for the intake that when a button is pressed, the intake returns to the initial position based on encoders. I did not complete more today because I was mostly working as an engineer on wire management and the stone pushbot and alignment idea. Here is an excerpt of the code that manages the x-rail controls:

```
//controls x-rail extension
if(gamepad1.left_bumper && slide.getCurrentPosition() < 0) {
  if(slide.getMode() != DcMotor.RunMode.RUN_USING_ENCODER) {
     slide.setMode(DcMotor.RunMode.RUN_USING_ENCODER);
  }
  slide.setPower(1);
}
else if(gamepad1.right_bumper && slide.getCurrentPosition() > -8800) {
  if(slide.getMode() != DcMotor.RunMode.RUN_USING_ENCODER) {
     slide.setMode(DcMotor.RunMode.RUN_USING_ENCODER);
  }
  slide.setPower(-1);
}
else if(((!gamepad1.left_bumper && !gamepad1.right_bumper) || (slide.getCurrentPosition() > 0 || slide.getCurrentPosition() < -8000)) &&
slide.getMode() == DcMotor.RunMode.RUN_USING_ENCODER) {
  slide.setPower(0);
}
if(!slide.isBusy() && slide.getMode() == DcMotor.RunMode.RUN_TO_POSITION) {
  slide.setPower(0);
  slide.setMode(DcMotor.RunMode.RUN_USING_ENCODER);
}
if(gamepad1.y) {
  slide.setTargetPosition(0);
  slide.setPower(1);
  slide.setMode(DcMotor.RunMode.RUN_TO_POSITION);
}
```

Teleop (November 14th, 2019):

With the final touches being put on the robot, I have been completing the teleop programs for the best control of the driver. I added the control of our new claw and programmed the REV Smart Servos to be noncontinuous rotation servos for the claw. The claw consists of four REV Smart Servos, two are used to clamp onto the block, and two are used to rotate the claw. Because I was unable to make the rotation servos to rotate for more than 180 degrees, I had to make those servos run on time and continuous rotation, rather than two locations, but they do not need to be very accurate anyways. Also, when I was working on wire organization, I fixed a lot of the wiring, but in the process of that, I mixed up all the motors for the mecanum wheels. In order to reorder the motors and their encoders, I had to test the motors with programming to make sure their encoders were on the same port. I fixed the motor alignment in the teleop, but the autonomous drive functions caused the robot to spin instead of moving forward. The problem might have been caused by the encoders and incorrectly calculating the amount of distance the motors need to move forward. In addition to this, I also created another drive based on sensor function that moves the robot on an angle. It's an overload of a previous function I made; it does the same thing, but takes different parameters.

Teleop 2.2 (November 29, 2019):

After working on the problem for a long time, I was finally able to accomplish one of my goals from the beginning of the season which was to translate the robot according to the driver's view of forward. I did it by using the gyro sensors in the robot's expansion hubs to determine the robot's angle of rotation in radians and add that to the rotation in which it is driving in. Now the robot can turn and drive at the same time, and the turning of the robot doesn't affect the direction it's traveling in. The code for that appears like this:

```
//gets the angle being returned by the gyro sensor
Orientation angles = imu.getAngularOrientation(AxesReference.INTRINSIC, AxesOrder.ZYX, AngleUnit.RADIANS);
Orientation angles2 = imu2.getAngularOrientation(AxesReference.INTRINSIC, AxesOrder.ZYX, AngleUnit.RADIANS);


//"zeros" the rotation to make the current rotation forward
if(gamepad1.a && !prevA) {
   realign = -(angles.firstAngle + Math.PI + angles2.firstAngle + Math.PI) / 2;
}
prevA = gamepad1.a;


//finds the angle that the robot needs to move in
robotAngle = Math.atan2(gamepad1.left_stick_y, gamepad1.left_stick_x) + (angles.firstAngle + Math.PI + angles2.firstAngle + Math.PI) / 2 -
Math.PI / 4 + realign;


//finds the percent of power to each wheel and multiplies it by the speed
leftFront.setPower(speed * Math.sin(robotAngle) - gamepad1.right_stick_x);
rightFront.setPower(speed * Math.cos(robotAngle) + gamepad1.right_stick_x);
leftBack.setPower(speed * Math.cos(robotAngle) - gamepad1.right_stick_x);
rightBack.setPower(speed * Math.sin(robotAngle) + gamepad1.right_stick_x);
```

Teleop 2.4 (December 6, 2019):

After a lot of testing, the drivers decided that the original form of the drive system was actually less complicated to use, so I switched back to the original code for the drivers, but I also improved the code to work with our new robot design (the lexan clamp that can grab blocks at any angle). I changed the rack and pinion sliders to move faster so the drivers could move the blocks up and down easier.

Autonomous Plans (December 9th, 2019):

I decided that the most efficient method of autonomous is to start in the loading zone and end in the building zone so that we only have to make one trip under the bridge. The goal is to use two color sensors to determine the difference between two blocks reflective values in order to find out which is the skystone.

Meet Notes (December 14th, 2019):

Although it was evident that our autonomous was capable of earning far more than any other team in our league, our autonomous performance was very inconsistent. Our maximum autonomous score was about 25 points (we moved a skystone under the bridge, relocated the foundation, and navigated), but in practice we scored higher. For autonomous, we used only three main sensors, two color distance sensors on the front of the robot for comparing the blocks to find the skystone, and a color distance on the bottom to find the line under the skybridge.

Autonomous Improvement Ideas (December 15th, 2019):

After discussing the problems with autonomous, I determined the main cause of inconsistency was the drifting of the robot causing a small difference in the angle of travel. My solution is to use the gyro sensors inside the expansion hubs to correct this mistake. I also plan on going back to using vuforia for sensing the skystone, but this time without scanning. I only want to check if two of the blocks are skystones because this will give me the location of both skystones fast, easily, and consistently.

Autonomous 2.5 (January 4th, 2019):

I have been experimenting with the gyro sensors at the last two build days over winter break, and during this time, I was able to complete a new encoder function that uses both the encoders and the gyro sensor to correct any unwanted rotation. The function will line up the robot perfectly to a certain angle. Because I built this function as an overload to the previous encoder function, it was extremely easy to transfer to using this function for almost every movement in the autonomous. I also switched to using vuforia to test the blocks rather than two color sensors. The concept is relatively similar to before because it checks two blocks then determines the third block as the default if the first two are not the skystone. The program should finish with about two seconds to spare now. Here is an example of my gyro sensor and encoder drive function:

```
while(opModeIsActive() && leftFront.isBusy() && leftBack.isBusy() && rightFront.isBusy() && rightBack.isBusy()) {
  //gets the angle of the robot from the two gyro sensors and averages them for more precision
  double angle = imu.getAngularOrientation(AxesReference.INTRINSIC, AxesOrder.ZYX, AngleUnit.RADIANS).firstAngle;
  double angle2 = imu2.getAngularOrientation(AxesReference.INTRINSIC, AxesOrder.ZYX, AngleUnit.RADIANS).firstAngle;


  //adjust the power based on how far the robot is from the runAngle which is the desired angle of travel
  leftFront.setPower(-power1 - ((angle + angle2) / 2 - runAngle) / Math.PI);
  leftBack.setPower(-power3 - ((angle + angle2) / 2 - runAngle) / Math.PI);
  rightFront.setPower(-power2 + ((angle + angle2) / 2 - runAngle) / Math.PI);
  rightBack.setPower(-power4 + ((angle + angle2) / 2 - runAngle) / Math.PI);
}
```

Autonomous (January 17th, 2019):

I completed our autonomous program for the tournament which is tomorrow. The program seems to be very consistent which was my goal for this next meet. The autonomous program scores us about 30 points and uses three sensors: gyro sensor, color/distance sensor, and vuforia. I created a separate program for each side of the foundation, and I program to set up our robot so that it fits into the 18 by 18 by 18 inch box. We added silicon onto our hooks that move the foundation so that the foundation no longer slides out of the robot's control.

League Meet Notes (January 18th, 2019):

Our meet on Saturday went very well for autonomous with an average autonomous score of 22 and 1 third. We consistently hit our max points in autonomous, 29 and our minimum autonomous score was 11. The reasons for the inconsistency in autonomous were fixed so that it ran successfully in the last few matches. We ended up being the winning alliance partner, and won the Inspire Award, so I think we performed very well, however, there are still some touch ups that need to be done.

Improvement Ideas (January 19th, 2019):

The engineers pointed out that when our driver switches directions or goes from a velocity of 0 to full speed, the robot is very difficult to control and can move in a jerky way due to braking. The idea is to write a new function where the controls affect the speed, but the robot accelerates or decelerates to reach that velocity. I believe this would also be extremely helpful during autonomous because when our robot travels too fast and the robot brakes, it rotates because of unbalanced weighting. I was also watching the driver practice earlier and realized that because of our very original claw idea with the L shaped claws, our robot cannot stack blocks next to any other blocks because when the block is released, it knocks over the blocks next to that tower. If I can get the current position of the servos, and only release them by enough to drop that block, our driver will not have to worry about that problem anymore and we will be able to stack much faster. Another problem we have is that our robot does not allow our alliance partner to also navigate while we run our program. This is because we run over every location on the entire playing field except between the two queries which is where our alliance partner sets up. Unfortunately, I think we can safely say that no one is going to create a program to go around the queries to go navigate to the line during autonomous, so we need to make sure there is a location on the wall that our robot does not hit on our side. This way our alliance can just run a program that goes forward until it hits the line.

Progress (January 24th, 2019):

I managed to create a more object oriented program for autonomous and teleop. I created an abstract class named Bendy which is extended by our OpMode classes. Bendy.java extends from linear OpMode and declares runOpMode() as an abstract function. This allows for a very simple, readable code layout for both teleOp and autonomous. Our actual code now is simplified down to only 87 lines of code:

```java
package org.firstinspires.ftc.teamcode;
import com.qualcomm.robotcore.eventloop.opmode.Autonomous;
@Autonomous
public class BestAutonomous extends Bendy {
  @Override
  public void runOpMode() {
    int skyPosition = 0;
    //sets up the robot
    setUp();
    telemetry.addData("status", "callibrating gyrosensor");
    telemetry.update();
    gyroCalibrate();
    telemetry.addData("status", "initiallizing vuforia");
    telemetry.update();
    vuforiaInit();
    telemetry.addData("status", "ready");
    telemetry.update();
```

```
        waitForStart();

        drive(0,-1,0,700);
        gyroAlign(0);
        drive(1,0,0,1400);
        sleep(500);
        if(!vuforiaSkystone()) {
            drive(0,1,0,500);
            skyPosition = 1;
            sleep(500);
            if(!vuforiaSkystone()) {
                drive(0,1,0,500);
                skyPosition = 2;
            }
        }



        gyroAlign(0);



        drive(-1,0,0,400);
        setPosition("rotate", 0);
        setPosition("rotate2", 1);
        sleep(200);
        clamp();
        move("lift2", -1, 50);
        move("lift", 1, 50);
        encoderStop("lift");
        encoderStop("lift2");
        sleep(1800);
        setPosition("rotate", 0.5);
        setPosition("rotate2", 0.5);




        // etc... more code...



    }
}
```

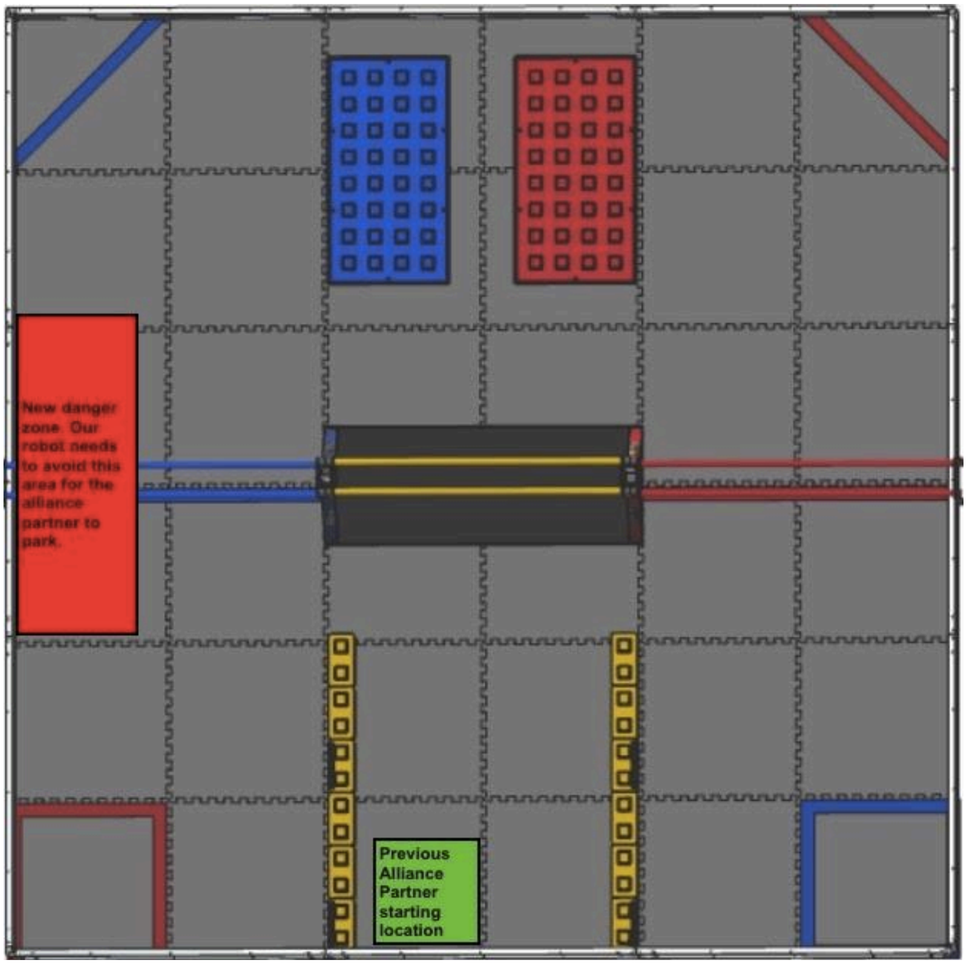
Our Bendy.java file is 320 lines of code now, so our functions actually take up more space than the autonomous program itself here is a portion of Bendy.java:

```
@Disabled
public abstract class Bendy extends LinearOpMode {
    private static DcMotor[] motors = new DcMotor[7];
    private static Servo[] servos = new Servo[5];
    private static String[] names = {"a", "b", "x", "y"};
    private static boolean[] prevButtons = {false, false, false, false};
    private static BNO055IMU imu;
    private static boolean clamped = false;
    private static final VuforiaLocalizer.CameraDirection  CAMERA_CHOICE = BACK;
    private static final boolean PHONE_IS_PORTRAIT = false ;
    private static final float mmPerInch  = 25.4f;
```

```java
    private static final float stoneZ = 2.00f * mmPerInch;
    private VuforiaLocalizer vuforia = null;
    private VuforiaTrackable skystone;
    ColorSensor groundColor;



    @Override
    public abstract void runOpMode();



    void setUp() {
        if(!isStopRequested()) {
            // code etc...
        }



    void drive(double angle, double power) {
        if(!isStopRequested()) {
            motors[0].setPower(power * Math.sin(Math.toRadians(angle)));
            motors[1].setPower(power * Math.cos(Math.toRadians(angle)));
            motors[2].setPower(power * Math.sin(Math.toRadians(angle)));
            motors[3].setPower(power * Math.cos(Math.toRadians(angle)));
        }
    }



    // an overload function
    void drive(double y, double x, double turn) {
        if(!isStopRequested()) {
            double speed = Math.hypot(y, x);
            double robotAngle = Math.atan2(y, x) - Math.PI / 4;



            motors[0].setPower(speed * Math.sin(robotAngle) - turn);
            motors[1].setPower(speed * Math.cos(robotAngle) + turn);
            motors[2].setPower(speed * Math.sin(robotAngle) + turn);
            motors[3].setPower(speed * Math.cos(robotAngle) - turn);
        }
    }
}
```

Problems (January 28th, 2019):

I just got word that we are no longer allowed to set up our alliance partner between the two quarries which is good news and bad news at the same time. It will be helpful because it will force me to build an autonomous where we avoid a spot under the field for our alliance partner, preferably next to the line so they can park and gain an extra five points. The problem is that I need to redo the entire autonomous program now so that it is more precise than before. This is going to make it difficult to complete the newer autonomous version before super qualifiers, and I may just have to continue to build off the program I currently have rather than create entirely new functions from scratch. I will plan on adding the second skystone for the state competition, if we advance.



Autonomous 2.6 (February 9th, 2019):

I have been instructing Wayne on how to build the autonomous program based on the functions I created so that when I am gone or busy, he can help program as well. There haven't been very many changes to this current autonomous, except for the slight changes in the path that the robot takes. However, I am creating a program for our robot at state, if we advance. Our engineering and design team is working with great persistence to complete a whole new design for state before super qualifiers. This design has the same drive system, but a faster and more efficient intake and outtake system. While they are working on this, I have been working on the program that goes along with that robot.