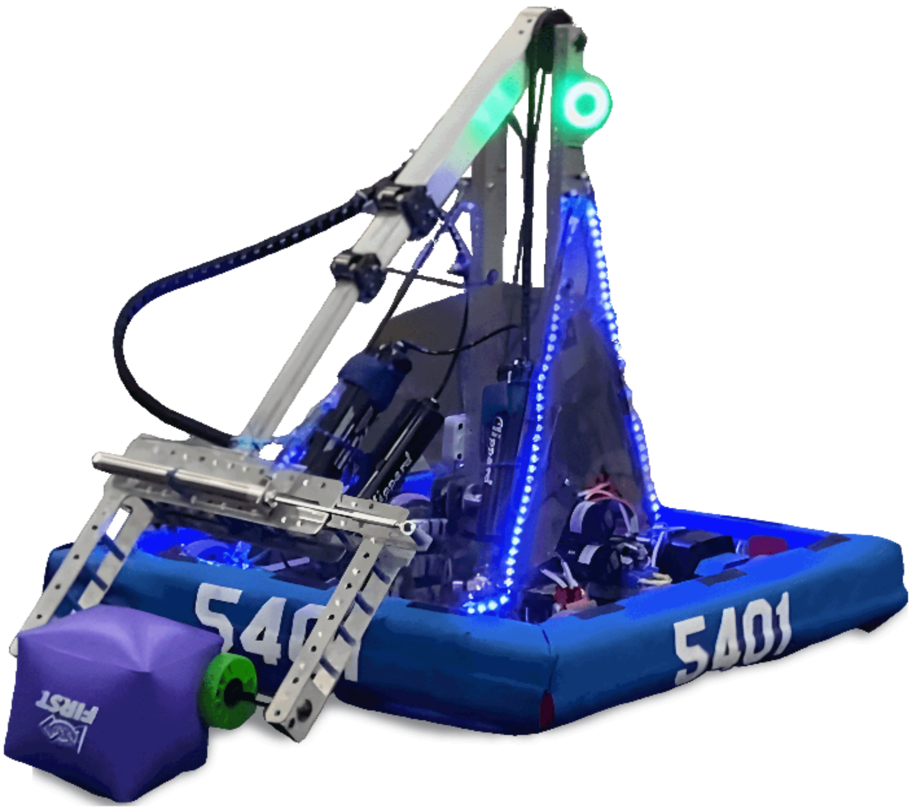
BUCKY



TEAM 5401

2023
TECH BINDER

Table of Contents

1. Introduction

Team 5401, the Bensalem High School Fightin' Robotic Owls, is ecstatic to be back and participating in 2023 Charged Up. While we've struggled in the recent past due to the pandemic, new processes and planning allowed our team to be more efficient than ever, despite an initial membership crisis. Last year, we had a Senior-heavy team, resulting in nearly half of the team graduating. After an extensive recruitment campaign, we welcomed an unprecedented 42 rookie members. We strive to create as many opportunities for STEM and Business education as possible, and with the massive inundation of rookies to the team, training them was no easy task. We overcame this challenge with dedication and thoughtful mentorship, and currently possess a super-motivated and capable rookie section. This year we welcome Bucky. Named after Marvel's Winter Soldier, known for his metal arm, our robot this year truly represents the growing strength and agility on Team 5401.

2. Kickoff Weekend

Team 5401's kickoff weekend started off on January 7th, when we all came to Bensalem High School, and gathered in the audion. We counted down the seconds, waiting for the official game video to release. As we watched the video, every individual started planning solutions to certain problems they witnessed in the announcement. Once the video was over, we broke off into multiple groups, made up of randomly selected individuals. This way we have the least bias and stay on task. Following breaking off into groups, we reconvene in the audion, to discuss further what the robot should do. After all groups presented their ideas, we took a break to think over each idea, and let the game really resonate with us. After the break, we reconvened for the last time to talk about what we should focus on in the manual, and then we disbanded. The next morning on January 8th, we came back in once again and then discussed strategy, and how to achieve it. We split back up into the same groups from the prior day, so it is more convenient to build off of what we had already talked about. After further discussion, and comparison to other groups' ideas, we began to make certain prototypes. This is so that we had a physical element to compare the ideas to, as results are always different in real life as opposed to theory.

3. Robot Design and Control

3.1 Drive Train

3.1.1 Initial Design and Prototyping

After some deliberation about Swerve Vs. West Coast Drive (WCD), we decided to use WCD as it would not negatively affect our cycle times when compared to swerve as we also have the ability to flip the entire arm 180°. We initially wanted to use four standard wheels and 2 omni wheels to assist in turning, but when comparing wheels, we found that a set of six 6” wheels in addition to a slightly lifted bumper worked most efficiently when attempting to climb and balance on the Charge Station, as ground clearance to drive up the steep angle of the Charge Station at a high speed was optimal. In addition, the omni wheels affected our ability to precisely turn the robot in order to place the game pieces. We decided to keep the Neo motors from last year as they still worked well in this competition. Additionally, we updated our frame perimeter from last year, with this one coming in at 29” x 30”. Everything is held in-place by ⅛” walled 1”x2” aluminum box-tube frame rails riveted with VexPro L-Gussets.

3.1.2 Final Design

While our design is constantly evolving with consistent testing, we have mostly stayed true to the Initial design concept. Our West Coast Drive drivetrain propelled by 6 Neo Brushless Motors attached to Gearboxes proved effective regarding this year's design challenges.

3.1.3 Control System

This uses Spark maxes that are directly connected to the motors as well as the encoders connected to the sparks as well. We are able to run a standard tank drive and use those controls.

3.2 Arm

3.2.1 Initial Design and Prototyping

Our main goal with the Arm was to be able to comfortably get the claw to a scoring node on the field without reaching outside of the frame perimeter. To do this, we knew that we had to use a multi-stage retractable arm, as we also needed to fit within the frame perimeter set for our starting configuration. We used a variable sketch in Onshape to act somewhat as a Geometric Testing Environment. After sketching a few designs, it became evident that a 33" upright combined with a 45.5 extending & rotating arm was most desirable to meet our goals.

3.2.2 Final Design

When finalizing our design, we decided to use 1"x2" aluminum box tube as the base of our uprights, attaching them via 4 VersaFrame T-Gussets and 2 custom-built L-Gussets to attach our uprights to our drivebase's center frame rail, another 1"x2" aluminum box tube with angled cuts to increase ease of assembly. For our arm's rotation, we decided to link a chain from top to bottom, driven by a Neo-1650 Brushless Motor mounted on the side of the upright, and driven by a set of steel hex shafts. We decided to use steel as the aluminum was not strong enough to handle the massive amount of torque applied to them via our chain driven sprockets. . The telescoping portion of our arm is made up of three spring-loaded box tubes, tethered back for variable control. This tether is connected to a spool fabricated out of a solid block of aluminum, further attached to the motor plates and powered using another NEO-1650. Holding the three box-tubes together.

3.2.3 Control System

Powered by 3 NEO-1650 Motors, and with braking power provided by one Andymark locking friction brake, our arm has the ability to fully rotate over the robot, driven by a chain & sprocket at the top of the uprights, otherwise known as the "shoulder". The arm is made up of three stages made of box tube, held out with constant force springs attached via ThriftyBot arm blocks. The end of the arm is tethered to a spool powered by another NEO-1650, allowing the arm to retract and extend as needed.

3.3 Claw

3.3.1 Initial Design and Prototyping

The claw proved to be the most difficult design to zero in on, and rightfully so, as it has the potential to make or break our robot's effectiveness. We started prototyping many different designs during and right after kickoff, but the one that emerged most successful was a claw using passively-auto orienting grippers, and horizontally compressing L-Brackets.

3.3.2 Final Design

Our Current claw design retained most of the aspects present in our prototype. The claw is all held in a c-channel which is attached to the arm. The c-channel has two pneumatic pistons attached on both the top and the bottom. The pinions on the pistons are attached to the outer and inner L-brackets that we call "fingers". On the final design, we used a finite element analysis tool called SimScale to virtually test the strength of these parts while optimizing weight, as the claw's mass was proportionally related to the arm's torque, & moment of inertia, something we struggled to reduce in order to increase the robot's speed and handling. These fingers glide along the inside of the c-channel, where they are held in by vertical cylindrical spacers. The far corner of both fingers has a piece of aluminum box-tube sandwiched between each finger's upper and lower plates. This box-tube has a grippy wheel and axle attached through bearings, providing a full range of passive rotational reorientation so that the game-piece may be accurately placed on its scoring node.

3.3.3 Control System

The claw's control system is relatively simple, benefitting ease of maintenance during the competition. This was important as the claw is the most likely system to be damaged. For the claw to open and close, we use two spring-loaded pneumatic pistons, one placed above, and one placed below the C-channel of the claw. To automatically orient the piece using gravity, we are using free-rotating wheels attached to both fingers, which allow the game piece's center of gravity to fall to its lowest point, which also happens to optimally position the game-piece for placement on a scoring node.

4. Software and Controls

4.1 FRC Architecture and Object Oriented Programming

Team 5401, like many other teams, primarily utilizes the Java programming language. Java provides us the ability to match objects found in the real world, with objects directly embedded into our code.

This year, Team 5401 jumped from the old to the new FRC Command System. Code is split into Subsystems and Commands. Subsystems mimic the physical parts of the robot, while commands execute actions that the physical robot can perform. The Robot and RobotContainer classes serve to loop the code of our robot utilizing the Scheduler class built into the FRC Architecture. With these loops, we are able to map a button to a given command which can be executed in teleop. Autonomous code, which runs without human input, was also made compliant with the new FRC Command System, by changing CommandGroups into their respective Parallel and Sequential command groups to allow us to chain together commands in a sequence, or run them side-by-side

4.2 Collaborative Work Environment

The software team for Team 5401 have a weekly meeting to discuss goals for that week. Additionally, tasks for any given day are written on a whiteboard. Meetings are often organized to discuss the general logic implementation of each individual Subsystem. Outside of meetings, programmers may work individually or in a team depending on the code or robot part that is being tested.

The following apps are used to communicate and organize the software team:

- Slack: This application is mainly used for communication, such as writing down goals and reminders. Slack also notifies us when changes are made to the code.
- GitHub: This application is used to store the team's code online as a code repository. As a result, any computer can pull the code from GitHub and edit it. In addition, GitHub offers the ability to "branch" code which creates a copy of the current working code. The existence of the copy allows experimental code to be created and tested without overwriting the current code that works. If the

experimental code fails, the branch can be abandoned without modifying the working code. If the experimental code works, the code can then be merged into working code.

- Google Drive is utilized to share other files that are not code. Code documentation and block diagrams are created and kept in Google Slides. Mechanical and software teams share a Google sheet detailing all the PWM, sensor, and pneumatic solenoid channels for the robot. Controller mappings are documented and stored on Google Drive as well.

4.3 Implementing the Programming Process

Identifying the Problem

- In the beginning of the build season, the software team has a meeting with the design team. The design team gives the software team a brief explanation of each subsystem, or group of mechanical parts that work together on the robot and its corresponding movements.

Designing/Drafting the Solution

- The software team begins to draw Block Diagrams of each subsystem. The Block Diagrams are very similar to a typical IPO chart: the Block Diagram displays the possible control inputs, the necessary processing, and the actions of the robot as a result.
- The software team then converts the Block Diagrams into pseudocode. Pseudocode is not “real” code and consists of comments strung together in order to provide an outline for the real robot code. Pseudocode also helps work out logic and structure for the future code.
- The software team then makes Controller Maps to better understand what Xbox Controller buttons should activate what command of the robot

Writing the Program

- Branches for each Subsystem are made off the main code/branch. Each programmer is assigned a Subsystem to write by converting pseudocode to actual code.

- During this step, programmers will often talk with the members of the design team. The goal is to discuss the specific components and movements of each Subsystem.

Testing and Revising

- After the practice robot is built, the software team tests each Subsystem on the practice, one Subsystem at a time. If the Subsystem works, the branch for that Subsystem is then merged into the main code/branch. Discrepancies between the practice robot and competition robot are noted and updates for the competition robot are made.

- Other testing branches are made for specific purposes on a case-by-case basis. For example, testing Shuffleboard, Vision, and Autonomous for the first time typically requires its own branch before it can be integrated into the finished product

4.4 Teaching New Programmers

Every year, several rookie programmers join the team with little to no experience in writing code for the robot. This year, rather than place responsibility into the senior members of the team, the Programming Lead assumed the responsibility of teaching any and all incoming members

Unlike in previous years, a lesson plan was developed for the purpose of teaching incoming (and returning) members. The lesson plans varied from a wide-array of disciplines, with rookies becoming well versed in basic Java, FRC Control Systems, and fundamental software (like VSCode & Github) before being sorted into the “clique” of their choosing. Programmers were able to choose between learning more advanced Java with Robot Code, to exciting Mobile App Architecture with Android Studio, to even advanced vision processing with Python and C++.

4.5 Human Machine Interface

The team utilizes two Xbox controllers to run our robot this year. This is due to many being familiar with its overall layout and design. This also allows for customization of the buttons and joysticks.

Unlike in previous years, the method in which Xbox controllers were implemented into the code has changed. The new FRC Command System created a way of assigning commands to buttons directly, and allowing you to bind commands directly to the axes. As a result it allowed the team to use much simpler, compact, modular, and easier to read code this year

4.6 Scouting App

- The "FROScoutingApp" was developed using Flutter. The app is optimized to run on an Amazon Fire 7 Tablet running Android 5.1 (API 22).
- This application can be used to scout any game of any year, as it is entirely customizable, allowing for users to set up and create the layout and scoring guidelines they find to be the most crucial that year. A file containing the layout of inputs can be exported from the app and imported on other devices.
- The data from scouting is outputted into a csv file which can be easily imported into sheets to allow for fast and easy data collection.

4.7 Vision Processing and Camera Streaming

The vision subsystem in programming has gained multiple significant changes this year. We have incorporated the use of programs such as PhotonVision to help working on AprilTag recognition this year. This will allow the robot to have a faster and more precise reaction to the various changes on the field that may occur during this dynamic competition.

5. Manufacturing

The STEM program at Bensalem High School received a new technical education facility as part of the renovations that took place in 2017. Thanks to our new space, all manufacturing of Bucky was done in-house in the Technical Education Shop of Bensalem High School, adjacent to our robotics lab, by our student-led manufacturing team consisting of 12 team members. This year, we further strengthened our manufacturing ability by further training rookies on our CNC machines,, which was used to produce many complex parts on the robot, making up most of the claw, and cutting other specifically complex shapes such as our tinted polycarbonate sponsor plates and . Additionally, we trained ~5 rookies on how to use a modern milling machine with digital measurement and calculation abilities, as well as a set of lathes. To further advance our goal of educating the next generation, we integrated multiple 3D printers into our production pipeline. Using a Makerbot Replicator+, an Ender 3D Pro, and a Markforged Onyx (a material made of nylon embedded with strands of carbon fiber) Printer, we were able to fabricate extraordinarily complex parts out of strong and lightweight materials.

6. Practice Field

After Kick-Off Weekend, we started building our field with wooden elements to assist prototyping of robot assemblies, as well as give the drive team a practice field on which to hone their skills. The assembly of the practice field this year was an incredible opportunity to get as many of our rookie students involved as possible with one of our rookie and intermediate members leading everyone, and noticeably gaining effective leadership experience due to it. Since we have such a large team this year, we struggled to find things for them to do, part of their build season experience was assembling the field. This helped introduce them to basic engineering concepts to prepare them going into assembly, CAD, or manufacturing. They were the ones that found, printed, and read the measurements on the FIRST released field elements, an experience that helped them become familiar with the field on a much deeper level.

7. Conclusion

With our team of ~72 members, and a broader student leadership team, Team 5401 has created a robot that brings the team a sense of pride and achievement. From day one of kickoff weekend, our goals were simple. To build a robot that achieves our goals, is expertly engineered to exceed our quality standards, and is a robot we are proud to show off at competitions. With the right course corrections, Bucky came together quite efficiently while affording each and every one of our team members invaluable STEM education and training. This year truly charged up our team with motivation and technological advancements that will only continue to snowball in the future with our now seasoned rookie-heavy team. We hope Bucky's flair and performance has you marveling at the growth our team has experienced as much as we are.