Senior Design Mid-Semester Report

IEEE SoutheastCon Hardware Competition

Auburn University

March 20, 2019

Matthew Castleberry, Joe Hinely, Josh Jabłonowski, Nia Perkins

Project Overview

The purpose of this project is to create a robot to compete in the 2019 IEEE SoutheastCon held in Huntsville, AL. The purpose of the conference is to share and demonstrate technical advancements in engineering. We will be competing in the hardware competition against other universities from across the southeastern US.

The purpose of the robot is to move around a square carpeted area with a side length of 244 centimeters, and pick up small objects on the field and place them into one of four colored home bases based on the color of the object. The arena will be split into two zones. Zone one features four home bases corresponding to four different colors. The robot will start in one of the four home bases. Zone two is separated by a one-inch thick white line. Zone two is where twelve objects will be placed for the robot to pick up: eight two inch wooden cubes, and four 2.5 inch diameter pit balls. The arena will feature a wooden box in the middle that will serve as an obstacle as well as four LED lights situated at the edge of zone two on the one-inch line. The four home bases will be placed at the four corners of the arena in zone one. An arena was built in order to test the robot as shown in Figure 1. This allows testing of the robot in a controlled area leading up to the competition.

The robot has several constraints that limit the size of the robot, materials used in the construction that assure the safety of participants and other robots [1]. The robot's width and length must be within a nine by nine-inch area and a max height of eleven inches. The nine by nine area must include a bumper covering eighty percent of the perimeter. The bumper must be 1.5 inches from the ground and have a minimum vertical cover of one inch, and must not have a radius of curvature less than one centimeter. This is to prevent damage to the arena and other robots in the event of a collision. The robot can also raise a mechanical arm, with a max length of three inches, with a flag presenting the competing school's name.

The robot utilizes a Raspberry Pi chip to control the motors while using an attached camera to identify the color and shape of the objects on the field. A LIDAR is used to localize the robot on the arena to help avoid the four lights and the wooden box in the middle of the arena. The robot will move the objects to the appropriate home base by running over the objects, closing a small door on the front of the robot so the robot can go back without losing any objects, hold them under the robot and move towards the appropriate home base of the object.

Figure 1: Field Constructed in Broun Hall room 368

Technical Progress

Mechanical Updates

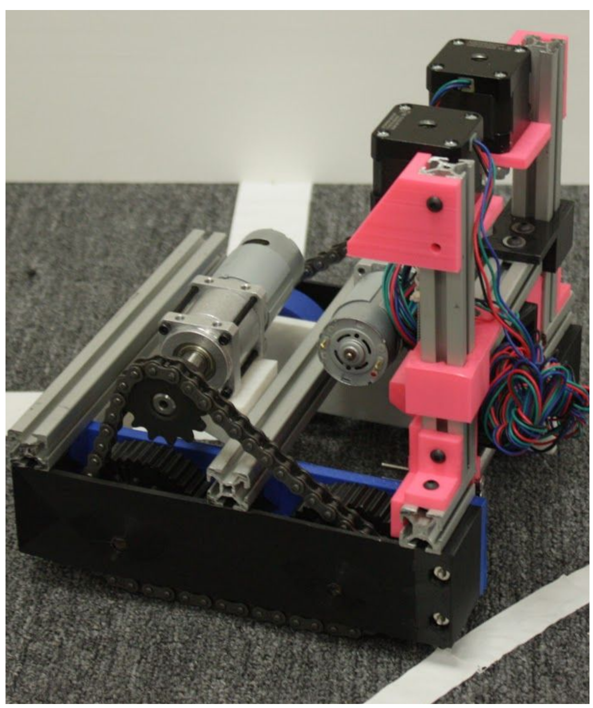

Figure 2: Initial Design Robot

During the testing of the robot, the stepper motors for the front gate were run using a basic program to make them turn. During this test it was discovered that the rubber tubing connecting the stepper motors to the threaded poles was not tight enough, allowing the motors to slip through the tubing. This, alongside the front gate making the robot front heavy lead to the decision to remove the stepper motors and front gate completely. The front gate can be seen in Figure 2 with the pink 3D printed material and the two servo motors on top. In its place, a servo motor will be mounted onto the side of the robot and attached to a paddle that will lift up to allow objects underneath the robot as shown in Figure 3.

The servo motor not only solves the connection problem of the stepper motors but is considerably easier to program. Removing the added weight from the stepper motors at the top of the gate also lowered the center of gravity. This allows the robot to make more sudden stops and maneuver easier around the obstacles on the field.

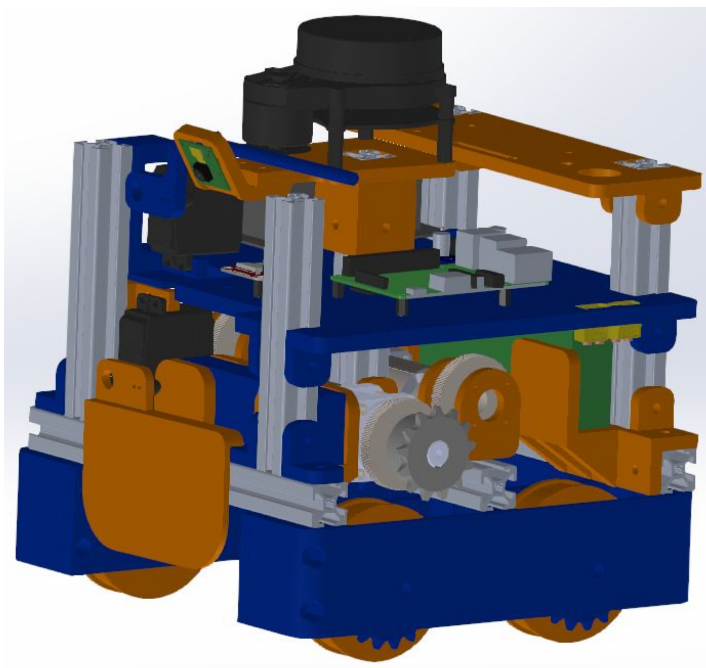

Figure 3: Final Robot Design with servo

Electrical Hardware Updates

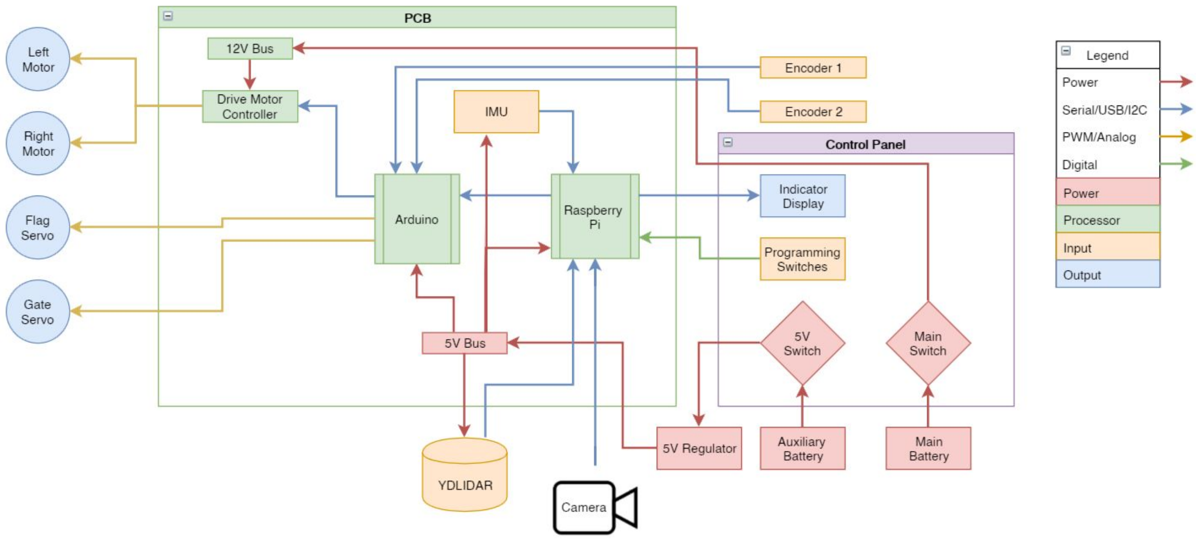

Figure 4: Electrical Block Diagram

Since the start of our robot build, the electrical hardware design has made significant changes. In Figure 4 above, an obvious change from our previous designs is the placement of our components and addition to a control panel for adjusting settings and controlling the robot without direct access to the Raspberry Pi UI.

We've added switches for both batteries so that they can be switched on individually and electrically separate the 12V and 5V rails. An LCD screen will be used for output, programmable via a DIP switch package immediately next to it. LED lights were initially considered, but it was decided that they would take up too much GPIO on the Pi vs. a serial connection with an LCD screen.

Some mechanical changes were also made to our design, requiring some of the electrical hardware to change. Specifically, the three stepper motor controllers to control the gate have been replaced with servos, so no motor controllers are necessary- and will reduce the size of our protoboard/PCB.

A PCB design was started to sit on top of the Raspberry Pi to help integrate all of our components onto one board, shown in Figure 5 However, during the design phase it was decided that changes may need to be made closer to the competition that would exceed the time required for lead time to acquire and design a new PCB. Thus, a custom protoboard will be constructed instead that will still sit on top of the Pi- shown in Figure 6.

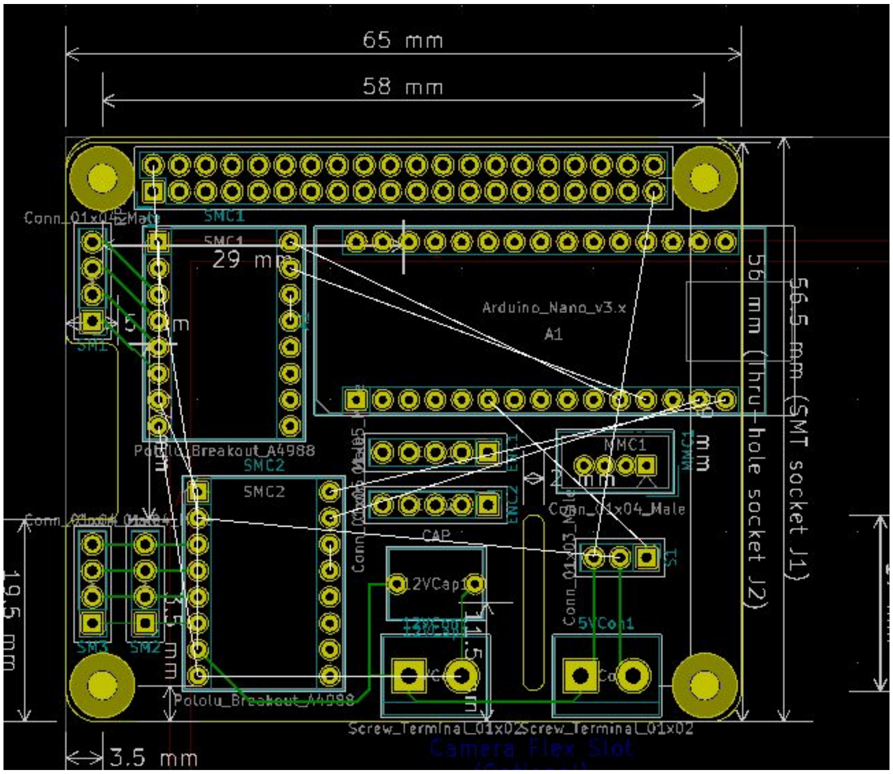

Figure 5: Pi HAT PCB Design (Unfinished)

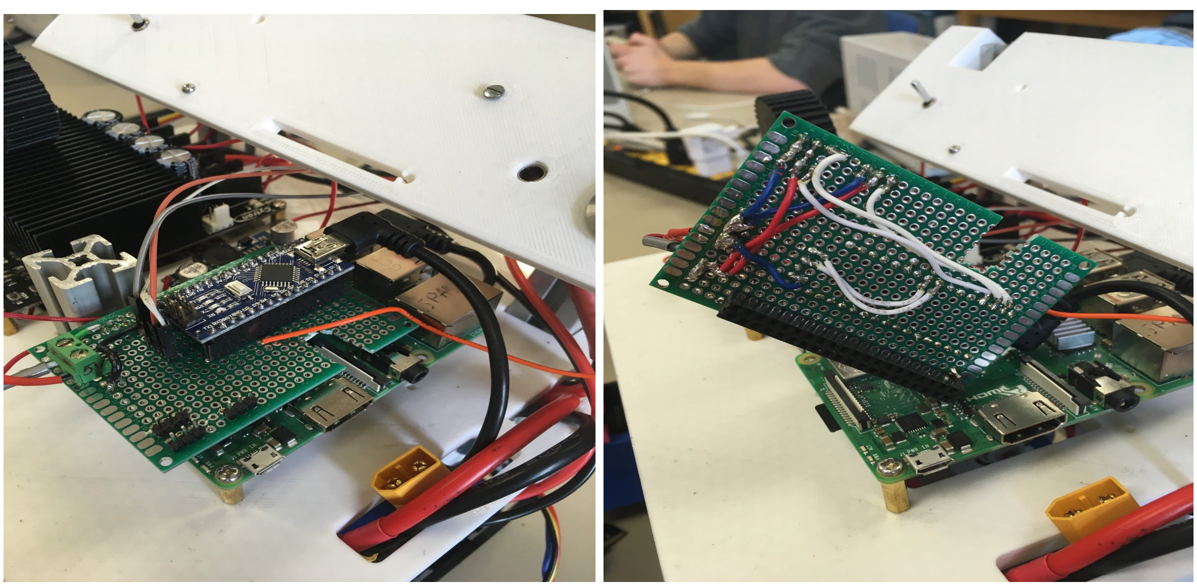

Figure 6: Pi HAT Current Protoboard

Extra sensors have also been added to our electrical design to account for ROS requiring more input for accurate navigation. The IMU function is described below in the next section, but it will be integrated on the protoboard with a serial connection to the Raspberry Pi. Encoders are also being added to the motors, but they will not fit on our current prototype and will need to be integrated to the second robot.

Software

Robot Operating System (ROS)

The need for a simplified approach for localization sparked the switch to use the Robot Operating System (ROS.) ROS greatly simplifies complex localization, mapping, and pathfinding algorithms; all of which are incredibly essential to this competition, although, heed was taken at first because of the vast learning curve coupled with ROS. A very layman's explanation of ROS is that sensor data is input nodes, which house all the vital information for the sensors such as programs, readme files, or launch files. Nodes can talk to each other and request information, process the information, and send to another node. A graphical explanation of how our sensor nodes will communicate is shown in Figure 7 below. We have four sensors; LIDAR, Camera, IMU, and encoders that create four ROS nodes. The LIDAR, IMU, and Encoder nodes all input into the localization node, this node coupled with the camera node create what we call the decision maker or essentially a path planner. The decision maker then outputs motor control commands to move the robot.

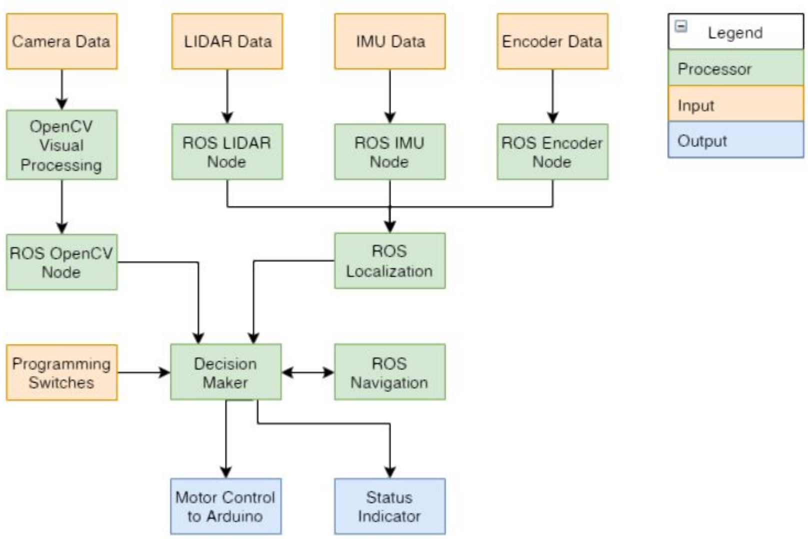

Figure 7: ROS Node Diagram

LIDAR and Localization

The LIDAR sensor is one of the main components of localization; this sensor acts sort of like a GPS for the robot. The YDLIDAR has a 360-degree scanning range with a 10-meter range. As stated previously and shown in Figure 7 the LIDAR will be a node, and this will be one of three inputs into the localization node. ROS greatly simplifies the localization process as it already has the algorithms needed, one being AMCL. Adaptive Monte Carlo localization is the process of tracking the pose of a robot against a known map, which is our case. AMCL takes in a map, LIDAR scan, and transform messages, and outputs pose estimates. As stated one of the first things we need to implement the AMCL algorithm is a known map, which is the competition field. We created a jpeg image which was feed thru a python program that was launched within ROS, and this gave us our map. However errors are being given when adding the LIDAR scan data to the map, we are currently debugging this error the next step will be to apply algorithms.

Inertial Measurement Unit (IMU)

An inertial measurement unit (IMU) is a sensor that can measure nine degrees of freedom (9DOF). It has a combination of three three-axis sensors: gyroscope, accelerometer, and magnetometer. These sensors measure gravitational tilt, translational acceleration, and magnetic fields respectively. The IMU we have chosen has a well supported ROS library. It will be used as an input for the localization node. We originally did not recognize it as a required sensor but after further researching ROS, we are now implementing it. The demo program provided online is able to display the sensor orientation on a display. We verified this program worked with our hardware. We also verified that we can receive the raw sensor messages where needed in ROS.

Rotary Encoder

In order to have more precise measurements in movement, we will use two capacitive encoders attached to the motors. The encoders will measure and count the turns of the motors, allowing us to know how far the robot will travel with one motor turn. Using this alongside the localization and object detection, we will be able to move the robot the exact distance towards any detected object to ensure the robot picks up the object. This will also prevent the robot from pushing any object into a position that would make it difficult to pick up.

Visual Detection

We are using the Raspberry Pi Camera to detect the debris objects. By using the open-source computer vision library, the Pi can extrapolate the debris's pixel location, pixel size, the angle from center, color, type, and approximate distance from the robot. It uses this data to make decisions on where the robot should go. For E-Day, we demonstrated that the robot can turn towards an object, drive until the object is right in front of it and then turn and find a different object. The program worked really well. The biggest issue is that the camera was pointed too far up and needed to be mounted at a lower angle. The only other issue was that it had some minor false detections but those should easily be easy to eliminate with more accurate tuning.

The next step is integrating visual detection with ROS. A custom message group has been created to pass the object information to the main group but needs to be tested. Once this is completed, it can be integrated with the localization in order to construct the competition algorithms.

Budget

The budget for this project has remained the same- however the first report did not include parts that were needed for the second phase of the project and the second robot. The items bought for that are in the Table 1.

Description	Cost	Quantity	Line Price
2020 tubing 20mmX20mm	$0.16	96	$15.36
100x Frame screws	$9.90	1	$9.90
flanged bearings for axles	$3.97	8	$31.76
Shaft collars for axles	$3.89	8	$31.12
Axle	$7.72	1	$7.72
Sprocket	$13.85	2	$27.70
Motor	$7.25	2	$14.50
Gearbox for motors	$45.90	2	$91.80
Motor mounts	$6.50	2	$13.00
Dual DC Motor Controller 5-36V	$63.39	1	$63.39
LIDAR	$99.00	1	$99.00
Capacitive Encoder	$21.86	2	$43.72
Servo	$17.89	2	$35.78
Pi Cam Cable	$5.43	1	$5.43
LCD	$9.79	2	$19.58
PLA pro orange	$22.99	1	$22.99
PLA pro blue	$21.99	1	$21.99
			$554.74

Table 1: Second Robot Costs

As to be expected with a project that is subject to change as better parts and ideas become available, the actual cost of development is different from that of the estimated budget for the original parts. The current actual cost of parts ordered to date is lined out below in Table 2.

Part	Quantity	Price/per	Total
2020 Extrusion	196	$0.21	$41.16
Stepper Motor Controllers	5	$8.99	$44.95
Raspberry Pi 3	2	$48.99	$97.98
Lithium Motor Grease	1	$8.76	$8.76
Rubber Bands	1	$5.49	$5.49
DC Motor Controller	2	$63.39	$126.78
OpenMV Cam	1	$65.00	$65.00
32GB microSD	2	$8.15	$16.30
Triple Axis Gyro	1	$14.95	$14.95
Pi Cam Cable	2	$5.43	$10.86
Encoders	4	$21.86	$87.44
Neoprene Rubber Strip	1	$0.96	$0.96
Lipo 11.1V 5000mAH	1	$39.99	$39.99
Lipo 7.4V 2200mAH	1	$18.99	$18.99
Stepper Motors	3	$12.99	$38.97
USB Cables	1	$6.99	$6.99
Epoxy Paste	1	$8.79	$8.79
2020 Frame Screws	2	$9.90	$19.80
Flanged bearings for axles	16	$3.97	$63.52
Shaft collars for axles	16	$3.89	$62.24
Axle	2	$7.72	$15.44
Sprocket	4	$13.85	$55.40
Motor	4	$7.25	$29.00
Gearbox	4	$45.90	$183.60
Motor Mounts	6.5	$4.00	$26.00
LIDAR	2	$99.00	$198.00
Servos	2	$17.89	$35.78
LCD	1	$9.79	$9.79
PLA Orange	1	$22.99	$22.99
PLA Blue	1	$21.99	$21.99

2020 Nuts	50	$0.21	$10.50
Various Screws	1	$17.92	$17.92
Nema Screws	1	$4.18	$4.18
Hex Nuts	2	$2.14	$4.28
Lead Screws	1	$5.14	$5.14
Lead Screw Tubing	1	$7.00	$7.00
Gearbox mount accessories	2	$4.50	$9.00
Chain	1	$16.88	$16.88
Smaller Stepper Motors	2	$10.98	$21.96
Chain tools	1	$21.80	$21.80
		ACTUAL COST	$1,496.57

Table 2: Total Robot Costs to Date

This is technically $300 over budget for just the parts we initially expected to buy, but we anticipated spending a bit more during development until we determined the parts we wanted to use as we experimented. We found this amount reasonable considering the progress we have made.

Distribution of Tasks

The work done during the Spring 2019 semester can be split into four different task groups. Since there are seven total members between senior design and SPARC, it works well to have two members devoted to each task group. Matthew and William are focused on OpenCV and visual processing. Nia and Noah are focused on ROS and localization. Josh and Joe are focused on electrical hardware. Alex and Josh are focused on mechanical hardware.

Timeline

Table 3 lists the tasks to be completed before the competition. Figure 8 summarizes the table in the form of a Gantt chart. Referencing the Gantt chart we are in cycle 2 of 3 and about a week behind the projected schedule. We plan to have the individual tasks complete by Wednesday, March 27th. The remaining week and a half before the competition will be used for integration and testing. The third phase will take place after the competition and will be focused on preparing for the final presentation.

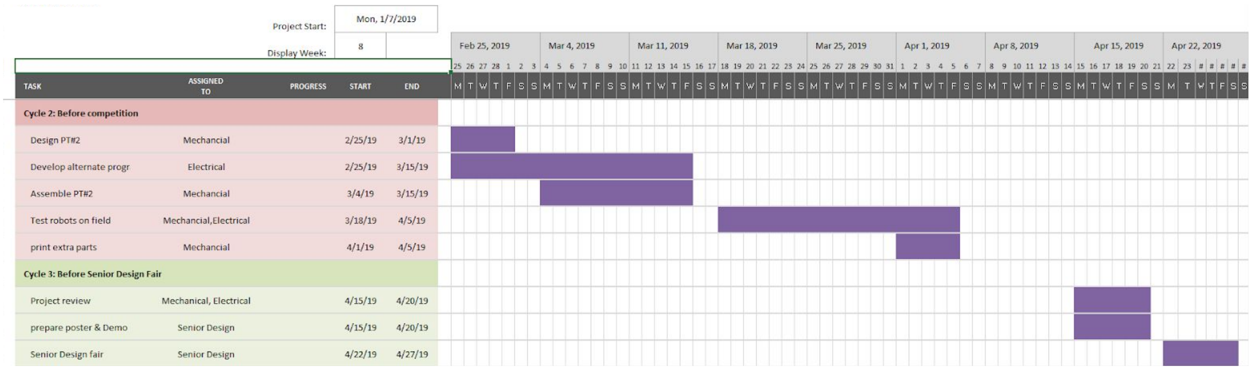

Figure 8: Gantt Chart of Cycles 2 and 3

Task Name	Version*	Category	Deadline	Assignee
Construct 9”x9”x11” interior sizing box	2	Mechanical	April 1st	Alex
Mounting hole improvements on Electronics plate	2	Mechanical	March 22nd	Josh
Fabricate, 3D Print, Assemble, and Wire New Robot	2	Mechanical/Electrical	March 27th	Full Team
Main and Auxilary Battery Voltage Detection	2	Electrical/ Software	April 1st	Joe
Wire Encoders	2	Electrical	March 27th	Joe
Design and Build Control Panel	1 and 2	Electrical	March 27th	Josh
Integrate Serial Control in ROS	1	Software	March 27th	William
ROS Localization	1	Software	March 27th	Nia/Noah
Capture ROS bag	1	Software	March 27th	Nia/Noah
Integrate Visual Detection in ROS	1	Software	March 25th	Matthew
Integrate Encoders in ROS	1	Software	April 1st	Nia/Noah
Test and Tune Full Competiton Algorithm (With 1 robot)	1	Software	April 5th	Full Team
Test and Tune Full Competiton Algorithm (With 2 robots on the field)	1 and 2	Software	April 12th	Full Team

Table 3: List of Tasks to be Completed

Conclusion

Overall, we are constructing a robot to autonomously move small blocks and balls into coordinated colored corners. The technical deadline for this project is April 12th. The operating technical budget is $715 for version 1 of the robot and $555 for the remaining parts for the second version.

References

[1] "2019 SoutheastCon Hardware Rules v1.3." IEEE, Huntsville, 18-Aug-2018. "http://sites.ieee.org/southeastcon2019/files/2019/02/2019-SoutheastCon-HW-Rules_v1_3-1.pdf"