The Ostrich

An Extensive Analysis

Date Submitted: 05/03/2017

Design Team 4
Section C02
RBE 2001 Final Project Robot
Worcester Polytechnic Institute

Members,
Signatures and
Contribution

Clayton “Rhododendron” Dembski		30%
Alex “Thunder-Mifflin “ Michelson		13%
Floris “Grashof?” van Rossum		30%
Dan “The Mighty Mango Muncher” Wensley		27%

Abstract

An analysis of Design Team 4's final robot for the RBE 2002: Unified Robotics class. The challenge, to locate a flame, report it's location and extinguish it and navigate a maze, was successfully completed. This report focuses on the design of the robot, the decisions made and conclusions that can be taken from those decisions. Additionally an extensive analysis of the sensors and the code which drove the robot to complete the firefighting challenge is performed, which mirrors the strong focus in the code.

Table of Contents

List of Figures:

1. Introduction

The use of robots has become increasingly prevalent in modern society. They can be used for work that is considered too extraneous or dangerous for the ordinary individual. It is in this challenge where a robot is built with that mindset as one of the most dangerous jobs a person can have is being a firefighter. If a robot could be made to do the work of a firefighter it has the potential to save many lives. For this reason, Professor Putnam tasked us with making a fire fighting robot. The robot must be able to locate a flame, report its location with respect to the robot's origin and extinguish it. To find the flame it has to navigate through a potentially complex maze, without touching anything. Secondary objectives include reporting the exact height of the flame and returning the robot to its home position. In this challenge there is a strong focus on sensor fusion, where many different sensors are utilized to complete the challenge. This challenge is completed by RBE 2002 students in every term. Usually most robots feature a fan as extinguishing tool, however the robot mentioned in this report featured an air cannon. This mechanism produced interesting and unexpected results. The team managed to successfully complete the challenge and take away many lessons for the future.

2. Methodology

Problem Statement & Initial Analysis

An analysis of the problem at hand is a crucial first step for any team that wishes to complete a challenge. A failure to complete this step, will result in problems at later stages. Therefore the first step our team completed is the analysis of the problem that was presented.

The final challenge for RBE 2002 is to navigate a 12 foot x 12 foot field. Most of the perimeter of the field is lined with walls. However, occasional gaps are possible, so proper precautions should be taken. A candle will be placed at a random location in the field. Additionally, wall segments can be placed to form a maze-like structure which increases pathing difficulty. Neither the walls nor the candle may be touched by robot at any time. Furthermore, it is expected that the robot will fit within a 12.5 inch cube. After the candle has been located, the robot will report the x and y location of the candle. As a secondary objective, a team could also opt to find the height. The robot should also show that a fire has been found and when it has been extinguished. The team robot must also use an Inertial Measurement Unit (IMU).

A team should analyze the problem as listed above, listing all the subtasks that must be completed. Then that team should devise a schedule which will guide them from start to finish. This schedule can be subject to change throughout the term as deadlines are met or delays occur. Our team met in the library to talk about the project and draw out the field and initial plans for the robot (*Figure 1)*. Some of the early design decisions that will be talked about later were also made during this session, although most concrete robot design decisions were made later on in the process. This session had the most impact on the team's understanding of the problem and how to best solve it. Therefore it had a great impact on the design decisions that were made later on.

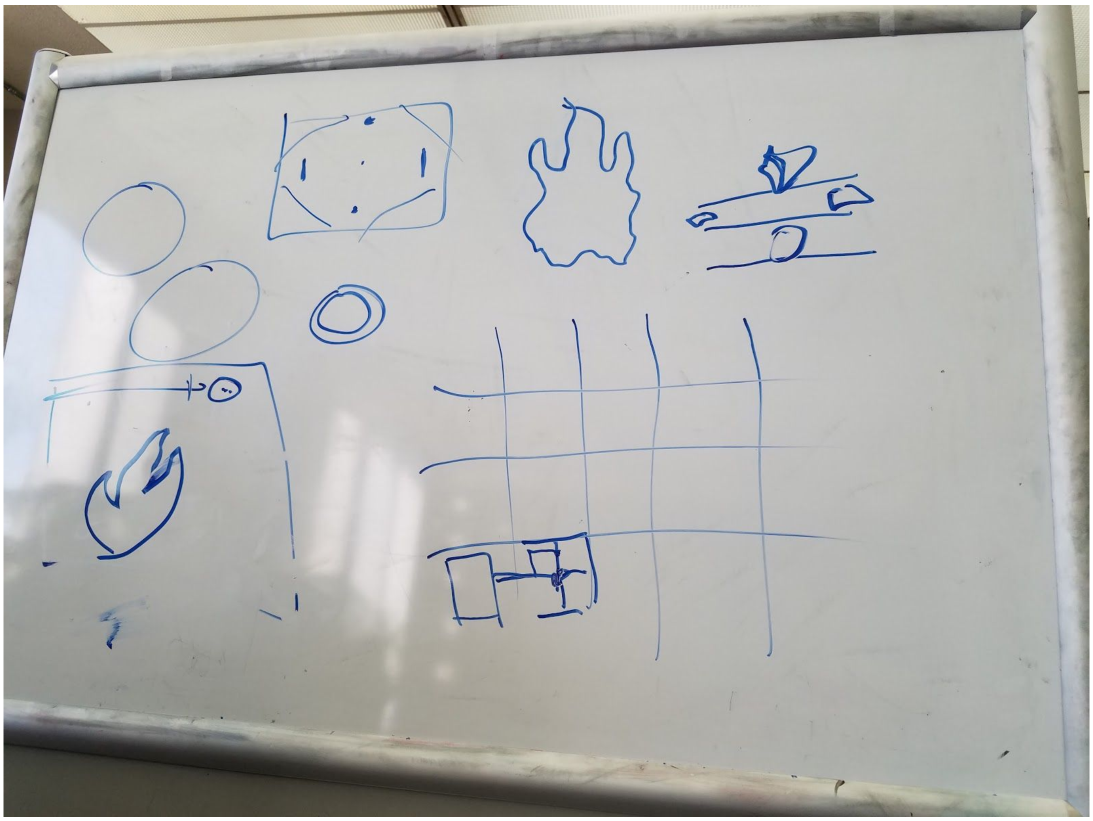
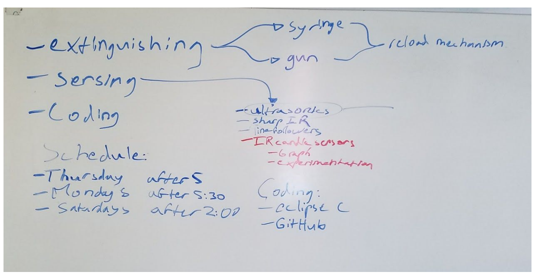


Figure 1: An initial drawing of the presented problem

One of the major breakthroughs our team had during this initial problem analysis was that the robot was not able to touch the walls. Although to a person who is familiar with the problem this might seem trivial, it is nonetheless an important discovery to make. Additionally we started brainstorming about how to best navigate the field. Whether we should do it in 12 inch segments, corresponding to the wall segments, then analyzing and then continuing, or perform calculations as we move. Our original perception of the board was that it would be split to be at perfect 90 degree angles. With this, we would split the map up into a wide grid, and keep track of position in a large array. We would thereby create a simplified version of SLAM. The team started discussing the technicalities of extinguishing a candle, and the difficulty such a problem would present. We looked at the options for putting out the fire: by air or by misting with water. Additionally, we discussed a basic schedule of events and tasks that had to be completed before the next meeting.

Early Design Decisions

When first starting this robot there were a lot of decisions that had to be made. The first thing our group decided is that our robot had to move. There are several ways the drivetrain could have been designed, however it was quickly narrowed it down to two feasible options. The robot could either have a standard, 2 wheel drivetrain that turned by reversing motors whenever it needed to change directions or it could strafe with a holonomic drivetrain setup. By using the standard drivetrain, one would immensely simplify the drivetrain and therefore leave less room for error. On the other hand, if it strafed it could make sensing and extinguishing the candle easier and more reliable. In the end, it was decided that the standard drivetrain because the additional effort required to make a strafing drivetrain vastly outweighed the benefits it provided to the other systems. Most notably, as the walls were not at perfect 90 degree angles, we would still need to adjust angle and distance from the wall, making the holonomic near useless. The next decision that had to be made was how to make sure the robot could visit every part of the maze. There are a plethora of very interesting ways to accomplish this task however, such as randomized movement, but, once again, the simplest solution was implemented. The simple solution is this scenario was to wall follow around the maze. If the robot kept its left side always facing the wall, a sensor on the right side would eventually see the entire field. It is a simple, yet effective method and there was no suitable reason to use any other method. The last decision to be made regarding the robot's mobility was how to record its position. Initial thoughts on this were to use vex encoders to keep track of the position of each wheel while this value was checked with an accumulated position of the IMU. In addition, the IMU would be used for rotation at turns. After that, decisions had to be made about how the robot would sense and extinguish the candle. Our robot had to be able to sense and extinguish the fire at a wide range of heights. Therefore, the entire extinguishing method was put on a pivot, the extinguishing arm. While deciding how exactly the flame was going to be sensed we decided to also try to determine the height of the fire. In order to accomplish this the robot had to be very accurate with our flame sensor. In order, to be as accurate as possible, flame sensor was to be mounted as close to our extinguishing mechanism as possible. This lead us to one of the integral design decisions, the extinguishing mechanism. After some debate, the discussion was narrowed down to two options. The robot could use a fan or a concussion cannon. The fan was simpler and did not need to be as accurate; the cannon was a little more complicated and had a greater range. In the end, the cannon was decided upon because the rest of the robot had been built out of simple mechanisms and the cannon would be a manageable amount of extra work. A plunger method, similar to a nerf gun (*Figure 4)*, and balloon air cannon was presented for this cannon. Due to accuracy required by the cannon, it was decided that there would be an indent in the front of the robot a sensor could be mounted in order to definitely sense if the robot was at the candle. Once that was concluded, all the conceptual decisions needed for us to start CADing our robot had been made.



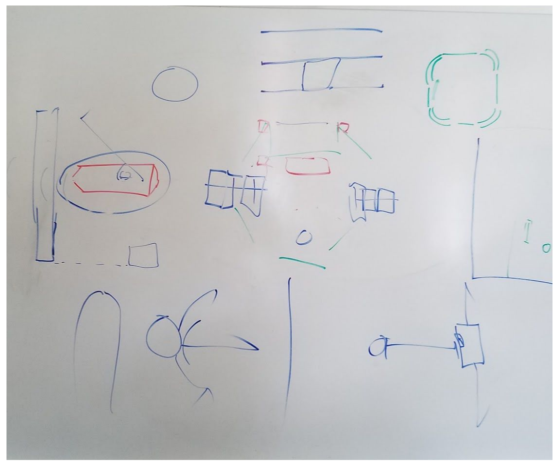


Figure 2 & 3: Whiteboard drawings of Early Design decisions

Figure 4: Nerf gun pullback inspiration

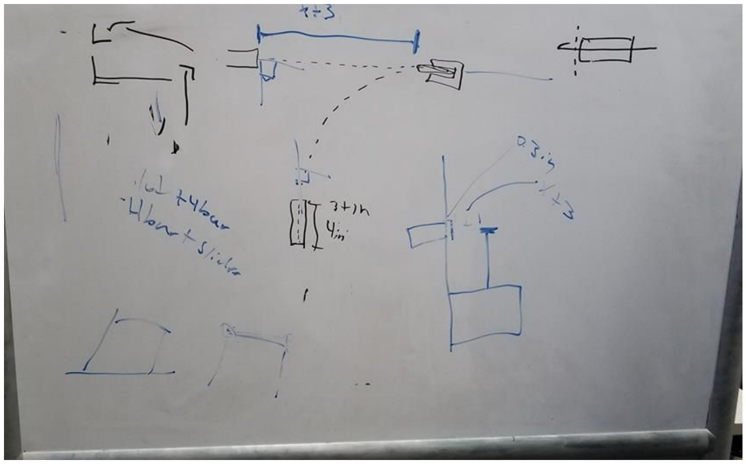


Figure 5: The notes of the first design session

Preliminary Design & Early Testing:

Creating a preliminary design and testing that design can prove to be an extremely valuable time investment. It allows a team to determine whether all of there early design choices are feasible and if there were any unforeseen difficulties to take into account. Testing proved incredibly important for our team in the design of the air cannon as it was a topic none of us had any design experience in. Additionally, we did not happen to have anyone with enough fluid mechanics experience to perform the necessary calculations required for designing the perfect air cannon. The importance of the testing of our preliminary design can be found in this subsection as well as the discussion later on in the paper.

The Drivetrain

The initial CAD of the drivetrain was created and laser cut out of 0.220 inch thick plywood. The purpose of creating this prototype was not only to be able to start coding, but also to test the feasibility of the drivetrain design that was decided upon earlier. Since the drivetrain proved sufficient and successful after testing, no changes were made, except for creating mounting holes for the superstructure, sensors, motor controllers and arduino.

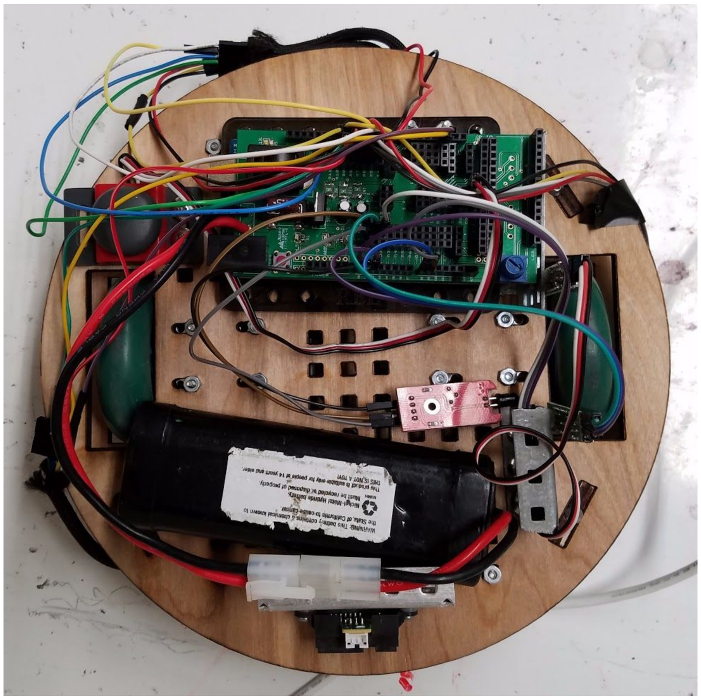


Figure 6: The first iteration of the drivetrain

The Cannon

Our team, as explained in the early design decisions, opted for an unconventional way to extinguish the candle, an air cannon. We made many different prototypes of the air cannon in order to figure out the best configuration for extinguishing. Below an initial prototype of the air cannon can be found. A pea shooter design seemed to work well, and thus multiple different versions of this design were used for experiments..

Figure 7: Air Cannon Prototypes

Multiple variables were changed in between the prototypes. The main changes were the length of the concussion chamber (X), the aperture of the opening at the front of the concussion chamber (Z) and the angle of the chamfer on the nose of the concussion chamber (Y) (*Figure 7*).

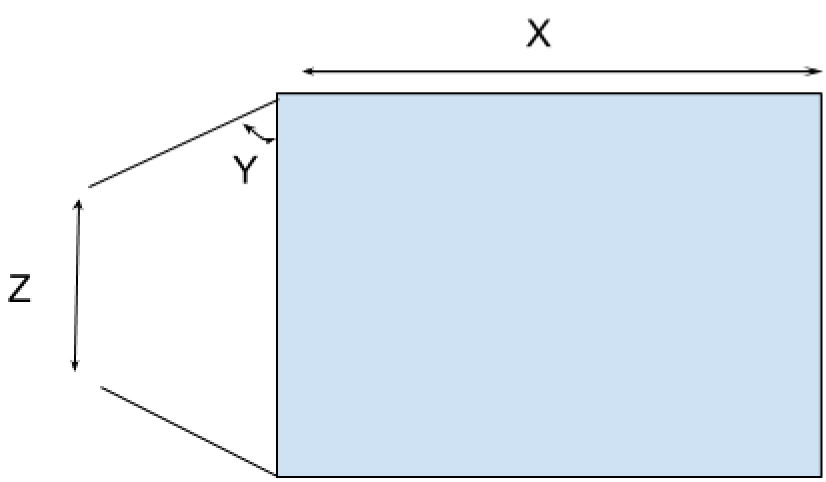


Figure 8: The variations in concussion chamber design

Through extensive testing many conclusions were derived, such as that a more narrow nose resulted in a smaller but more powerful shockwave. Because the shockwave was more powerful it could travel further and thus extinguish the candle at greater distances. However, because it proved extremely difficult to aim correctly by hand, we believed that it would be impossible to line up properly with the candle with the robot. Based on the significant testing, we found that a fairly large length (X) of our concussion chamber and a large aperture (Z) for a wider shockwave was preferable. In discussion a more extensive analysis of this topic and problems can be encountered as this testing was not as accurate as we initially thought and thus led to flawed design conclusions that had to be fixed later on.

The Pullback Mechanism

A great challenge that the team had identified early on is how to pull back the balloon on the air cannon. When tests were performed with the air cannon, it was always pulled back by hand. However, a robot would never be able to get as much grip as human fingers on the balloon. An experiment was setup to test a mounting system between the pullback mechanism and the balloon on the air cannon. The mounting system used rubber cement to mount a binder clip to the balloon. The binder clip would then be attached to the pullback mechanism. This proved successful initially, and it was incorporated into the final design. Later on, it was discovered that this mounting method was not as strong as it had initially appeared.

Figure 9: A binder clip

Although initially the plan was to create a design influenced by the nerf gun reload mechanism (*Figure 4)* this still proved difficult to conceptualize. Therefore, a design was made for the pullback mechanism consisting of a linear slider with a gear rack mounted on it. A slip gear, powered by a VEX 393 motor, was then attached to the top (*Figure 10)*. The linear slider had a rubber band attached which was mounted to a fixed position. The rubber band mimicked the balloon on the air cannon. The VEX motor was powered which caused the linear slider to pull back and then shoot forward at a high speed. Based on this test it was deemed that this version of the pullback mechanism was successful and so it was kept and implemented in the final design. This prototype was used as well to gain the specific distance measurements of the slip gear to the linear slider. This prototype still did not account for all dimensions necessary to design the final iteration of the air cannon and pullback mechanism.

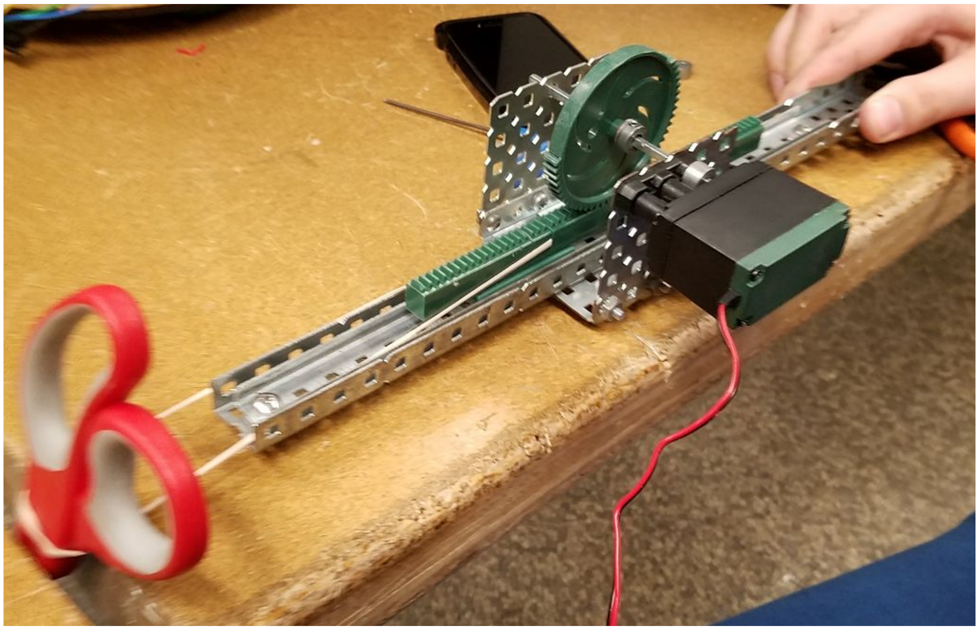


Figure 10: The pullback mechanism prototype

Final Design Decisions:

Below a picture, labeled *Figure 11,* can be found of the full CAD model of the robot. From this model, .dwg drawings were made from which the plywood drive base and acrylic robot were laser cut.

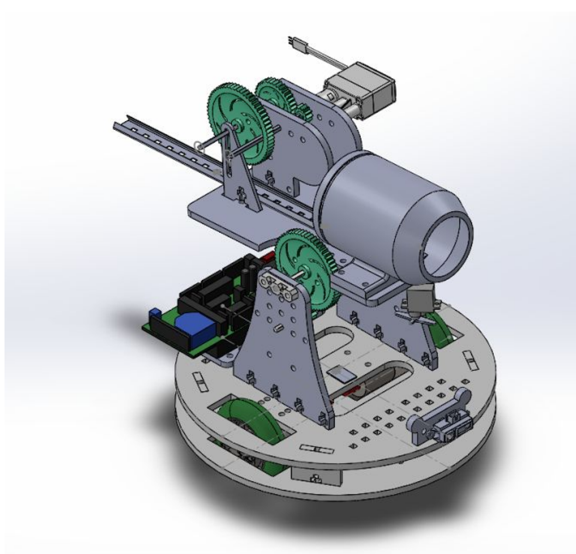


Figure 11: The final full robot CAD

Drivetrain Decisions:

The final design of the robot consisted of three layers.The bottom layer was the drive layer. This consisted of an acrylic circle housing a roller transfer ball and 2 wheels with 3D printed hubs, as well as all the electronics to drive the motors. These wheels were driven by pololu motors with built in quadrature encoders. Each pololu motor had a gear train of 171.79 internal rotations per individual motor rotation. Due to this, the encoders had an extremely high number of encoder ticks per revolution. Overall the drivetrain is the least complicated sub assembly of the robot.

Sensor Decisions:

The second layer was the sensor layer. This layer, built on the circular drivetrain, housed three infrared sensors (one on the front and two on the side), the IMU and the arduino. The final layer of the robot is the extinguishing layer. This layer has an arm that pivots around an axis located at the center of the robot. The arm is driven by a regular VEX motor with a 5:1 gear ratio. A potentiometer is attached to the axle with the smaller gear for a higher resolution of potentiometer readings. On top of this arm, is the extinguisher assembly.

At the front of the robot a mount was made for a sharp IR rangefinder, later this would be replaced by a IR + Ultrasonic rangefinder combo. On the left side of the robot there are two mounting holes for Sharp IR rangefinders. These will be used for wall following, and maintaining a proper distance from the wall

Extinguish Mechanism Decisions:

The extinguisher assembly (*Figure 12)* consists of two components, the air cannon and the pullback mechanism. The air cannon is powered by an elastic balloon. The air cannon will be 3D printed. The balloon is attached to a linear slide which is powered by a VEX 393 motor. The linear slide meshes with a slip gear, a 60 tooth gear that is chopped in half, to allow the mechanism to fire. Although in the CAD assembly uses a 36 tooth gear for a 3:1 gear ratio on the pullback mechanism, this proved to have insufficient torque. At the bottom of the arm assembly, under the air cannon there is a servo to which the flame sensor will be mounted. This allows the flame sensor to scan back and forth.

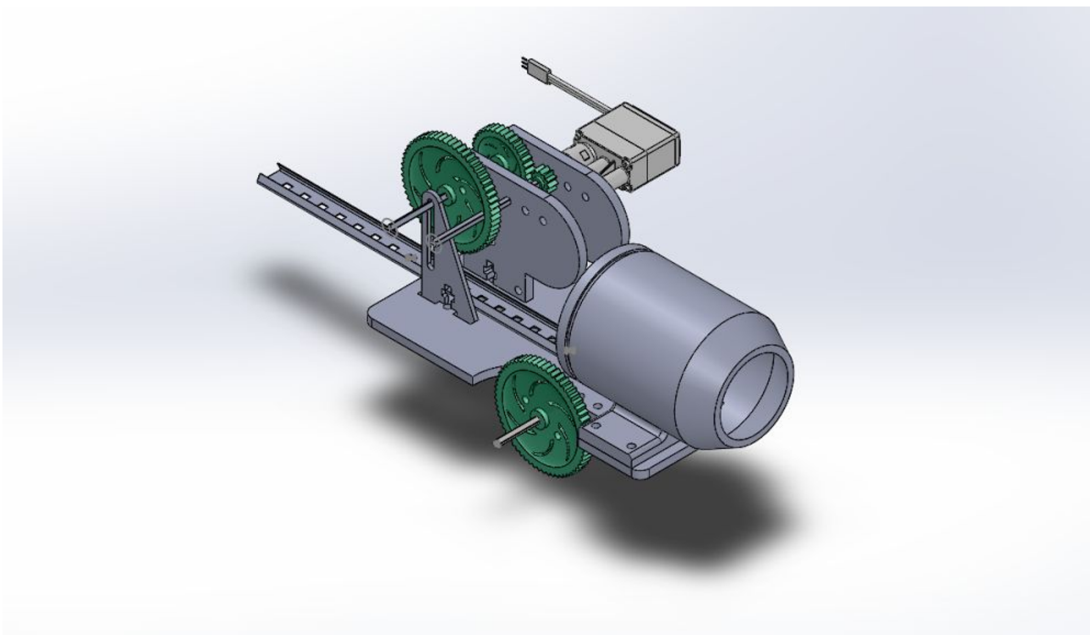


Figure 12: The extinguishing sub-assembly

Coding:

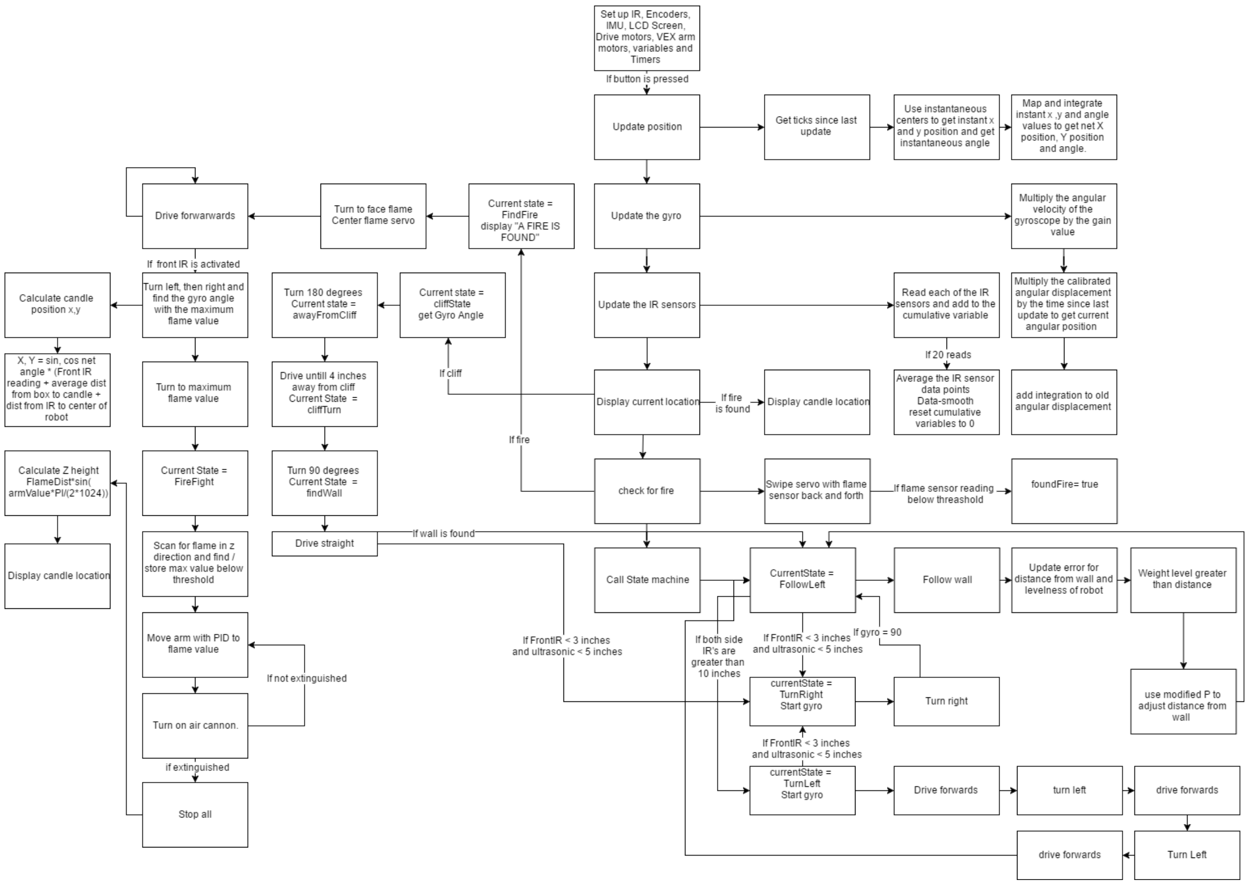


Figure 13: The Coding Flowchart (See attached for higher quality)

At the end of the design phase a basic structure for the code was developed. This structure was a state-machine code with some initial functions at the beginning of every loop. These initial functions consisted of updating the global position of the robot and updating the gyro in order to maintain accurate odometry. In addition during these initial functions the infrared sensors are updated in order to maintain proper wall following. The next of these functions checks if there is a cliff nearby and runs a cliff avoidance subroutine if there is. The last of these initial functions checks to see if the flame sensor sees a fire. Depending on whether it sees a fire or not it proceeds to one of the two main sections of the state machine.

If it doesn't see the fire it proceeds to the wall following section of the state machine. During this if it sees a wall to left and no wall in front, it will wall follow. If it doesn't see a wall to it's left it will turn left. Finally, if it sees a wall in front of it, it will turn right. While the robot is wall following it will continuously rotate the servo back and forth in search of the flame.

If the robot does see the flame, it transfers into firefighting mode. It will immediately stop wall follow and pause to show it has spotted the candle. Then it will turn towards the candle and start moving towards it. Once it gets a certain distance away it will make sure it is lined up with the candle. After that, it will scan the candle vertically, ascertain the height of the fire and extinguish. At the point the robot has accomplished its task and will stop moving.

5. Results

Overall Robot Performance

After the demo, our group was a little disappointed. We knew that our robot could have performed better, as it had done previously. For example, the flame sensor on the servo failed to detect the candle as the robot drove past. This had not happened previously and resulted in us having to readjust the robot to see the flame. It would have continued wall following, and inevitably seen the flame on the next pass, but that would have taken up a considerable amount of time out of the demo period.

The wall following code functioned fine, except when it encountered a specific complex wall section where a 90 degree left turn was made after which it had to straighten to perform a

180 degree turn. This occurred because of insufficient testing on the field, particularly the whiteboard surface. A tweaking of some motor speed values would have fixed this. The odometry/ position tracking code worked as expected however, when adding the final distance from the robot to the candle to the overall displacement of the robot, the x was added to the y and vice versa. This made our robot report the wrong x and y location of the candle, but it was still within the 8 inch diameter accuracy circle. These two problems were the biggest in the demonstration, both were very easy to fix. When they were fixed, a video was made and submitted.

The rest of the code worked as expected. The cliff avoidance system worked, although some final tweaking was performed for the video. The candle alignment code worked fine. The air cannon missed once but extinguished the candle on the next try.

Extinguisher

The extinguishing mechanism had to undergo a few changes to get to the final working form. First of all the mounting technique for the balloon to the linear slider with rubber cement was a failure. Instead, the balloon was twisted around a nut, tied up with a wire, which was then attached to the linear slide. The linear slide attached to the balloon added a great amount of mass to the firing mechanism, thus when the balloon pulled back a significant amount of mass had to be accelerated. Additionally friction caused an additional opposing force. This meant the air cannon was no longer powerful enough to extinguish the flame, even at close range. Through multiple changes, described in more detail in the discussion, such as adding rubber bands, a more powerful was produced shockwave to extinguish the candle successfully and reliably.

Encoders

The encoders worked as expected. With their incredibly high encoder tick per revolution count, the odometry was extremely accurate. The encoders were very reliable, definitely not the weakest link of the robot.

Infrared Sensor

The infrared sensors on the side of the robot worked as planned. Upon further experimentation it was found that the front sharp IR rangefinder did not have a range applicable to our circumstances. Therefore it was swapped for an IR with a better reading range. At two different distances the infrared sensor would give the same reading. This made it impossible for the robot to actually know how far it was from the wall. As a result, an ultrasonic was placed on the front of the robot. The ultrasonic does not have the resolution to be used on its own, however it does scale functionally so it can be used to determine which distance is correct for a given infrared reading. The IR sensors were reliably used in wall following code and flame alignment.

Flame Sensor

The flame sensor is extremely good at determining if there is a flame present within a specific cone. However, it cannot accurately read its proximity to the flame. With the addition of even some minor noise it became impossible to determine if the flame was 3in. or 13in. away. Another problem was that the cone did not seem to remain a consistent size at different distances. Therefore, the angle of the cone could not be used to determine the distance to the candle. On the other hand, the fire did always seem to be in the exact center of the flame sensors vision so the direction to the flame could be ascertained but not the distance.

Inertial Measurement Unit

The Inertial Measurement Unit (IMU) was found to be much less accurate than initially anticipated. Therefore it did not play the role in our robot we had intended for it. It was planned that the sensor was to be used for the distance tracking of the robot as it approached the flame. We quickly found, after the IMU lab, that it had considerable drift, even with the appropriate complementary filters and calibration. This meant that it could not be used as a distance tracking sensor to the degree of accuracy we had intended.

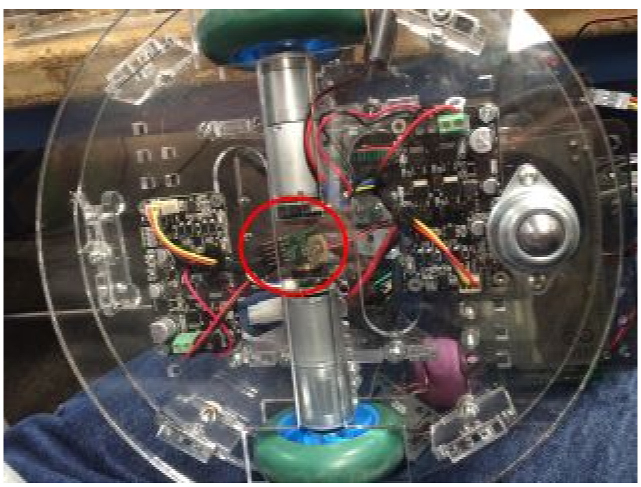


Figure 14: Location of the IMU on the drivetrain

Because using the IMU is required for this challenge, a new purpose was found for it. We used it instead to make accurate right angle turns when wall following. Additionally it proved crucial in lining up to the candle as it turned to a specific angle calculated by the flame servo sensor reading. More details on this can be found under Discussion and Extinguishing Code. The IMU gyroscope chip was lined up with the center turning point, there was even a dedicated laser cut hole in the center of the robot that lined up the chip with the turning center (*Figure 14)*. The drift in the IMU does not affect the gyroscope assisted turns because the gyro value is taken at the beginning of the maneuver and then read constantly until the final angle is reached. Because a turn never lasted longer than about 5 seconds the drift did not cause a significant error in the turn.

Code

The wall following code worked pretty much as expected except for two exceptions. The first one is when it encounters a cliff. Originally, no code was implemented for this scenario, however it quickly became apparent the this was a necessary precaution. Luckily, a line following sensor could be used for cliff detection and was easily attachable. In addition, several functions had already been written to do the operations required to avoid a cliff. Therefore, after a little testing a cliff avoidance protocol was added to the code.

The other scenario that made the wall following program fail was if the candle is directly in front of the robot as it finishes turning. If the flame sensor does not see the candle in time, the robot recognizes the candle as a wall and will begin wall following around it.

Odometry worked reasonably well. It would give very accurate numbers in most scenarios .However, in very specific circumstances, such as turning around one wheel it would provide very inaccurate information. Sometimes this distance could be as much as three times what it would supposed to be. This was later discovered due to some misunderstanding as to which encoder belonged on each side. The encoders had originally been programmed backwards.

Thus, the instantaneous turning center would be slightly off. This didn't matter when the instantaneous turning center was far away such as traveling in a straight line. However, in certain circumstances, like turning about one wheel, it could have a huge impact. Once this issue was resolved, the odometry became extremely accurate.

Searching for the candle was fairly successful except for two issues. The first is the issue mentioned in the wall following section above about seeing the candle as a wall instead of a flame. The second issue is that the height of the candle varies, yet the robot has no way to check for different heights. The arm of the robot must be set to approximately the height of the candle before the scenario begins or else the flame sensor might not see the candle.

Horizontal scanning was not originally anticipated, however due to issues with the flame sensor it became necessary in order to properly line up with the candle. Once it was implement though, it worked fairly successfully every time.

Vertical scanning worked well, provided that the robot was successfully lined up with the candle. Sometimes the robot did not extinguish the candle on it's first attempt, so an amendment to the code was added which made the robot try to extinguish the candle until it doesn't see the candle 5 times scanning in a row. This let the robot be sure that the candle was out.

6. Discussion

The Drivetrain:

Overall the drivetrain performed as expected. The Pololu motors were able to power the robot forward at a sufficiently slow and accurate speed to complete the challenge. A transfer ball at the rear of the robot allowed a wide base which supported the center of balance. The turning center was positioned in the center of the robot, this allowed for easier distance tracking calculations as well as a good reference point to position the IMU.

An analysis was performed on the drivetrain to find the linear speed of the drivetrain and the current through the motors under load:

$i_{stall} = 2.4A \quad \omega_{no\ load} = 34\ rpm \quad r_{wheel} = 1.375\ in \quad \tau_{stall} = 200\ oz * in = 12.5\ in * lb$

$i_{no\ load} = 250mA \quad W_{robot} = 5.5lb$

$Weight\ per\ wheel:\ 2.25lb$

$\tau_{wheel} = 2.25 * 1.375 = 3.0938\ in * lb$

$3.094 * (12.5/34) = 1.14\ rpm\ speed\ reduction$

$\omega_{load} = 32.9\ rpm$

Max Linear Speed:

$\omega = 34\ rpm = 34 * ((2\pi)/60) * r = 4.896\ in/s$

Max Linear Speed under load:

$\omega_{load} = 32.9 * ((2\pi)/60) * r = 4.732\ in/s$

Total Current under load:

$i_{stall} + i_{load} = i_{total} = 0.25 + (2.4/12.5) * 3.0938 = 0.844A\ per\ motor$

$48\ Counts\ per\ Revolution = 48 * 172 = 8256\ encoder\ counts\ per\ revolution!!!$

Additionally it mentions the extremely high encoder count per revolution. This explains why the obvious choice was made to favor the encoders for distance tracking over the IMU accelerometer.

Extinguishing Mechanism:

The extinguishing mechanism proved to be one of the most unpredictable assemblies on our robot. We had to make many small tweaks and discoveries in order to ensure that it worked reliably. We almost gave up on the air cannon and were very close to replacing it with a fan due to time pressure, and consistent failure to integrate the pullback mechanism with the air cannon.

The Air Cannon

The final design of the 3D printed air cannon (*Figure 16)* was based on very closely on the Mellow Yellow© can, as it proved to be the most successful at extinguishing the candle, or so we thought. When the pullback mechanism was fitted onto the balloon of the air cannon, after a lot of struggling, we found that the shockwave was not as powerful as we had observed in our prototypes. This can be attributed to many different factors. One is that we were able to pull back the balloon further with our fingers as compared to the motor. Additionally the friction of the linear slider assembly, and the weight, which causes inertial dampening, caused a significantly less powerful shockwave. This was mainly due to the reduction in speed of the balloon. Even at point blank candle range, where the candle is directly in front of the air cannon, it was unable to extinguish the candle. Therefore we adjusted the "nozzle" of the air cannon. During prototyping we had found that a more restrictive nozzle caused a more powerful shockwave. This is consistent with the Bernoulli Principle, where an increase in pressure caused by the reduction in concussion chamber diameter causes an increase in shock wave velocity. Initially we disregarded this design because we found that this type of cannon required a higher degree of alignment accuracy from the robot. Additionally rubber bands were added to add more force to propel the balloon forward.

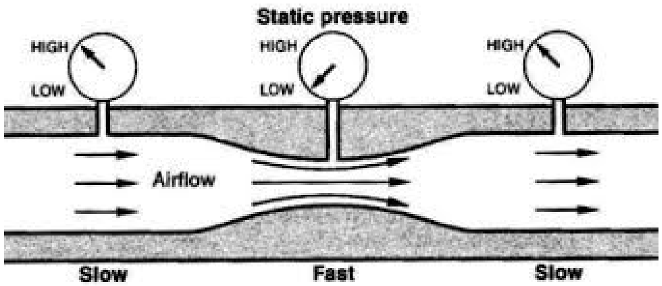


Figure 15: The Bernoulli Principle

The attachment (*Figure 17)* onto the air cannon, a plastic bottle nose, worked as expected and allowed the robot to extinguish the candle from an appropriate distance.


.

Figure 16: The 3D printed air cannon

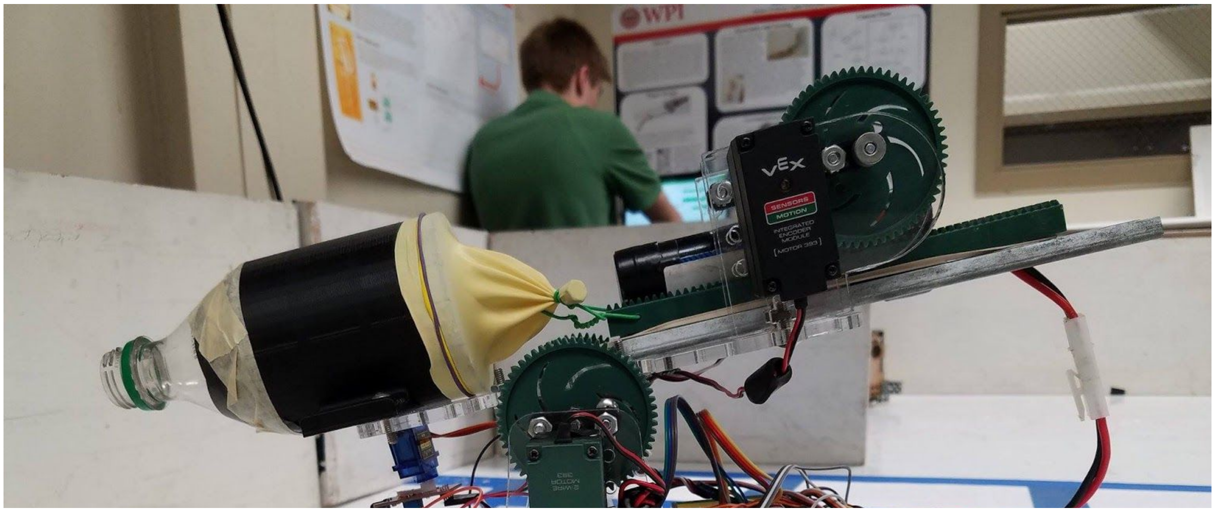


Figure 17: The final air cannon with a despairing RBE major in the background

The Pullback Mechanism

In order to fire the air cannon, a system had to be created to pull back and release the balloon, much like our fingers. The idea was to reduce as much friction as possible, so that, when let go, the balloon would act as if it was not inhibited by any external forces. To do this, we decided on a slip gear system. When the gear was rotated to a particular degree where the remaining teeth were removed, and there would be no surface to continue moving a mechanism. The mechanism would then be exposed directly to any outside force, of the rubber bands and balloon, without resistance. The pullback mechanism consisted of a linear slider attached to a slip gear. The slip gear had 180 degrees of teeth, allowing for a maximum drawback of 4.5 inches. To allow for a greater torque, the slip gear was attached to a gear train of 1:5. To allow for an ease of resistance on the balloon when firing, two rubber bands were mounted so that they too would be able to pull forwards the linear slider. This linear slider contraption was attached to a plate holding the air cannon. From here, the question became how to attach the pullback mechanism to the balloon. The first attempt was to use rubber cement to attach a binder clip to the launching mechanism. The launcher, as we found out soon afterwards, took ~7 pounds of force to pull back the firing mechanism. The radius of the slip gear was 1.3 inches, and the efficiency can be estimated on the low end as ~45 percent, due to the slight mobility of the acrylic in their respective slots. The 393 therefore experienced a torque of $\frac{7*1.3}{5*.45} = 4.04444444$, well within the 14 in-lbs available by the 393 motor. Due to this, the binder clip mechanism was not strong enough to hold the balloon. A feasible solution to fix the linear slider to the balloon was using a nut inside the balloon to create a pocket inside the balloon. The nut was covered in ripped up balloon to keep it from damaging anything due to the large amount of force exerted on the small area. Around this pocket we wrapped wax coated wire with a knot that would tighten when force was exerted on it. When wrapping the wire around the balloon, it was pulled taut so that a minimal amount of balloon surface area was used to create the pocket. This is because the more surface area that was wrapped up in the pocket would decrease the elasticity of the balloon as a whole. The wire was looped through a hole on the linear slider and a knot was tied. This is the mechanism that can be seen in the *Figure 17* above.

Sensor Analysis:

Sharp IR Sensor

For each of the Sharp Infrared devices and the flame sensor, two calibration curves were setup to calculate distance: a polynomial and an exponential trend. For each trend, an experiment was performed on the board to calibrate the distance. First, the robot was placed roughly 20 inches from either the flame, or the wall. Next, it would analogRead() from the sensor with a delay of roughly 1 millisecond, and collect 1000 points of data. This data would be averaged to yield a single data point. This was repeated another 60 times at the same data point, then, once again, averaged to yield the analog value for a particular distance. The robot was then moved forwards an inch. Using excel, these points were plotted, and a best fit, polynomial graph was created. These points were graphed a second time using mycurvefit.com [1] to allow for an exponential in the form ab^x. The reason for such extensive testing, especially for both the side IR's was the leveling of the robot. Two of the three IR sensors on the robot were placed on the left side, mounted to the side plates 1 inch apart from one another. In this way, a system of stereo depth perception was generated, the robot could then, not only find its distance from the wall, but, by comparing the difference in distances of the two Sharp IRs, find the angle of the overall robot. The robot could then compensate itself to normalize position from wall and the angle of the direction of the robot and the plane of the wall.

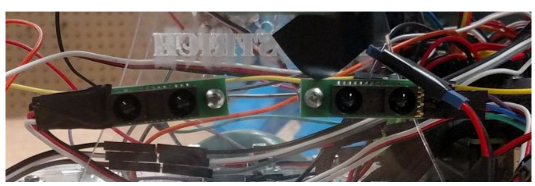


Figure 18:: Left and Right Side IRs

The two IRs on the side did not have the same curves as each other. This was most likely due to tiny imperfections in the resistances and capacitances within the sensor itself, yielding a different curve from one another. Distance values were used instead of the analog values as the distance was linear, while the analog values were not. If the distance values mismatched when level with the wall, the robot would overturn, to try to "level" itself so that each sensor matched distance. This would increasing the chance of running into a wall.

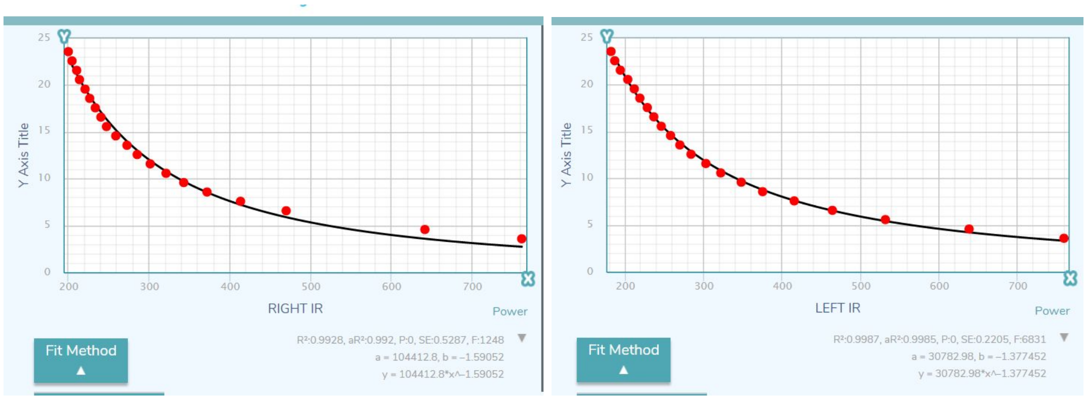


Figure 19: The calibration curves for the left wall following IR Sensors, Left and Right when facing the left side of the robot

The same calculation was done for the front IR sensor, as it needed to be accurate to calculate the distance to the flame box. As this was less dire, and the distance was closer, only the less accurate polynomial graph was generated.

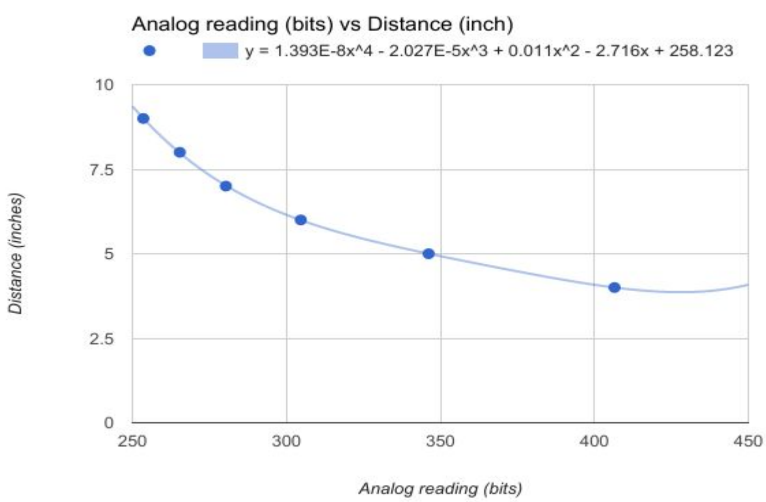


Figure 20: Calibration curve for Front IR sensor.

This front IR sensor became our first instance of direct sensor fusion. It was a separate SHARP Ir then the other two used previously. The sensor was, unfortunately, not a real function. If the IR was moved too far back, it would begin to read values that were given at a closer range, IE. 300

could correspond both to 6 inches and to 23 inches. This front sensor was implemented so that, when the distance was closer than ~3 inches, the robot would turn right. The problem with this, however, was that the robot would get false positives, and begin to turn right when the next wall was across the map. We decided, therefore, to analog read the ultrasonic sensor to find when the front IR was a useable function. At first, we wanted to use the ultrasonic to determine distance, as the value returned from it was a function, however, the particular one that we had could not detect and give an accurate direct analog reading within 5-6 inches from the wall, much further than we wanted to be before making our turns. We mounted the ultrasonic just above the IR on the front of the robot and had the IR activate when the ultrasonic, which did not spike values regularly, read an analog value of ~12, just over 7 inches from the wall. In this way we could get accurate readings for distance from the IR as we approached a wall to turn, and as we approached the candle to find the distance of the candle box from the center of the robot

Figure 21: Sharp IR(Bottom Left) with Pololu Ultrasonic (Top right)

Flame Sensor

Similarly to the Sharp IR rangefinders the team thought the distance to the flame could be found using the IR flame sensor. After finding the calibration curve, the discovery was made that this was not as feasible as initially thought.

A large amount of data was recorded by using an arduino program that recorded the analog signal of the flame sensor at a high frequency over a short period of time. This led to a huge excel document of raw data which had to be processed to provide a useful calibration curve for finding the distance to the flame. First the average was taken for the data measurements at each set distance. Now that the amount of data was reduced to two columns, a special website [1] was used to provide the equation. The website was chosen because it was able to provide a more accurate curve to the data. The type of curve chosen was Exponential: Proportional Rate of Growth Decrease, which follows the following format:

$y = y_0 - (V_0/K)(1 - e^{-Kx})$.

The trendline can be found below in *Figure 21*.

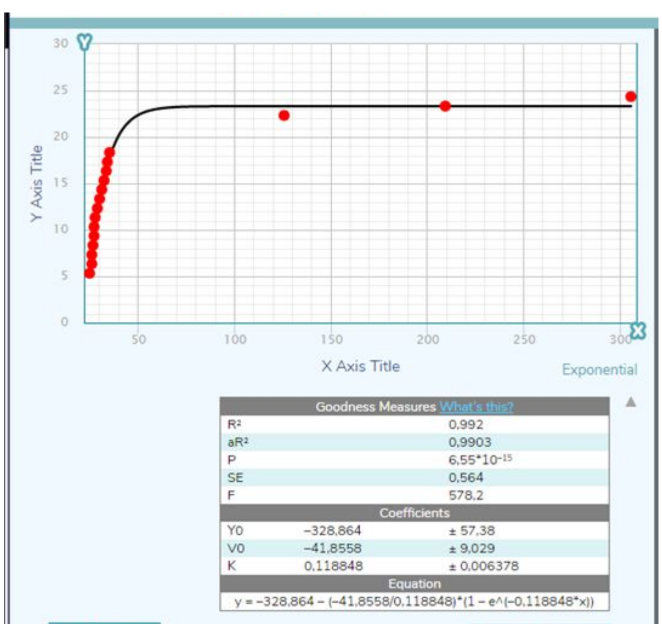


Figure 22: The Flame Sensor Calibration Curve

From this trendline the equation can be implemented in the code to provide the distance to the flame based on a reading. After observing the trendline it can be seen that finding the range to the candle with the flame sensor is unreliable. This is because the data follows a logarithmic growth pattern where initially a small change in distance will cause a significant change in reading. Therefore a reading at close proximity to the flame can be very inaccurate. A one bit change in the reading value can mean a huge change in distance to the flame. We briefly considered adding an operational amplifier to increase the range of bits for around 0 - 50 in the flame sensor reading, thus amplifying the reading at close proximity, however this would reduce the range we would have for the flame sensor.

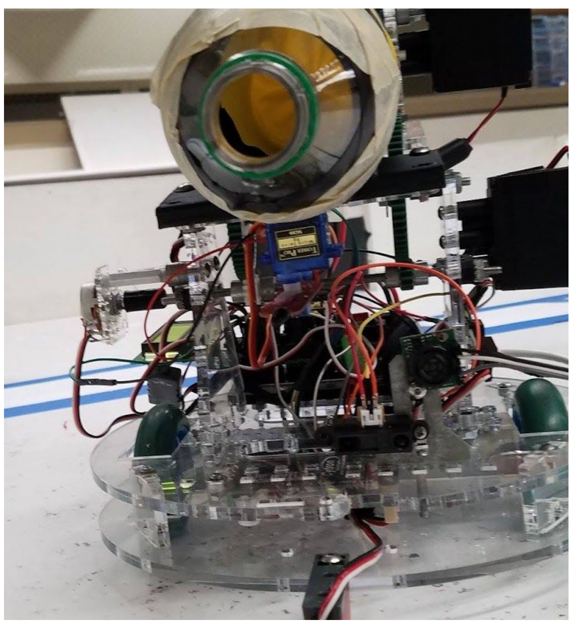


Figure 23: The location of the flame sensor on the robot

Code Analysis:

Wall Following:

The code for wall following was developed with the intent of solving two issues with wall following: keeping the robot parallel with the wall, holding the robot at an appropriate distance from the wall. To solve this problem, we created a control loop with two separate feedback variables. One variable took feedback from the difference between the left and right IR sensors, while the other took feedback from the normal distance from the wall. Both these feedbacks were made into separate errors and scaled by constants to drive the motors. The first of the two errors was the more difficult of the two to calculate. While calculating the difference between the two sensor distances would have been a reasonable way to find the error, this had its issues. The major issue was that the if the robot found itself with a large difference, the error would scale to late for the robot to compensate leaving it to run into a wall. The inverse was also an issue; when there were really small errors, the robot would compensate to much. Although adjusting the constant multiplier would have solved this problem, it would only help one of these two problems and make the other worse. To solve this a model was made that would scale non-linearly. This was done by calculating the average between the two sensors and comparing that to center point. In this case, the center point was 1.5 and the two sensors were treated as points at 1 and 2 respectively. In theory, when the sensors were the same, they should equal 1.5, but when they differed, they would diverge similarly to a second order polynomial. This gave the advantage of eliminating the two problems with just calculating the direct difference between the two sensors. The second error was much easier to calculate as it was just the desired distance from the wall minus the actual normal distance from the wall. Both errors were then scaled and used to drive the motors. This two error feedback system proved to be quite effective. The two errors counteract each other and balance the robot at the appropriate distance from the wall and keep it parallel.

Position Tracking:

Accurately handling the position of the robot proved to be a challenge. Using methods like gyro headings and distance vectors from encoders accumulate too much error while using strictly encoders could lead to error from wheel slip and uneven terrain. Luckily, with the two driven wheel design, and the overall flat terrain, using just the encoders was a viable option. To tackle the problem, we used two things: a global set of variables to keep track of angle and position and a set of instantaneous variables contained within a class. This let us keep the constant track of where we are, and grab instant changes relative to the robot's current orientation and sample that at whatever control frequency necessary.

In order to do the actual position tracking with encoders, one method more commonly used is to add instantaneous displacement vectors using the global angle. This method is reasonably effective; however, fails to account for small, immediate differences in wheel rotations, and leads to the same error accumulation that gyros experience through integration. To solve this, we devised a different method. The method created used instantaneous sin() approximations to determine the angle change, and an algorithm based on instantaneous centers to find x and y displacement. These values are calculated in a separate class, and then added to the global variables using a coordinate transform due to the local variables being calculated base on the robot's current heading and not original coordinate system.

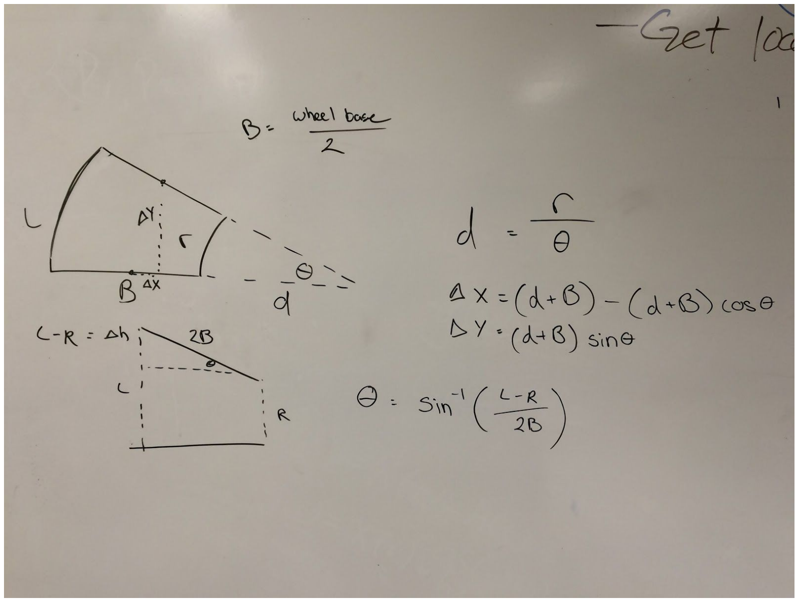


Figure 24:Odometry Calculation Visualization

The sin() approximation was done by finding the difference between the two wheels distances

$$diff \ = \ d_L \ - \ d_R$$

Then by finding the arcsine of the difference, there is an approximate angle change

$$angle \ = \ arcsin(diff)$$

To find the position, things were a little more complicated. First, a radius was found to the instantaneous center based on the rotation of the robot.

$$d \ = \frac{d_R}{ang}$$

Then using geometry, delta x and y can be found as shown through the diagram above.

$$\Delta\, x \ = \ (d + b) \ \ - \ (d +) \ cos(angle)$$

$$\Delta\, y \ = \ (d + b) \ sin(angle)$$

This method calculates the x and y displacements based on full wheel rotations. This leads to a far more accurate measurement than displacement vectors and proved to be very successful.

Extinguishing Code:

Throughout the program in the `loop()` function a method called `flameScan()` is called consistently. This causes the flame sensor, which is mounted on a small RC airplane servo, to move back and forth on an xy plane. The cone turned out to be adequate to detect the flame at a range wide range of possible heights of the candle. Once the flame sensor on the servo detected a flame it would exit wall following and move on to a new state called `FindFlame` which approached the flame. After `FindFlame` was completed, which meant the robot was lined up with the flame, the state machine switched to `FireFight` which calls functions in order to extinguish the flame.

Approaching the flame:

Once the robot has entered the `FindFlame` state it means that it has detected a flame. The first thing the robot does is stop all motion, including the flame sensor servo. The angle between the center of the robot and the flame sensor servo is calculated. Then, using the gyroscope, the robot turns to face the flame. The robot then drives forward in order to close the distance between the robot and the candle. In order to prevent the inevitable collision of the robot and the candle, an IR distance function is used which returns a boolean value based on whether the distance is more or less than an input distance. Once the robot detects that it is close to the candle it moves on to line up properly with the candle.

The air cannon design of the robot meant that the robot had to be properly aligned with the candle. If there was a fan on the robot, a “spray and pray” type of code could be implemented. But because we could only fire the cannon once every 5 seconds, that could prove to be difficult. Although an initial alignment was made earlier to drive toward the candle. It is not sufficient to extinguish the candle. Therefore the flame sensor on the servo performs one final scan back and forth, finds the max value from the flame sensor and records the servo angle at that point. The angle between that servo angle and the center of the robot is then calculated again, like in the code above, and the robot adjusts again. Now that the robot is properly aligned with the candle. The Z-alignment and extinguishing code can begin in the `FireFight` state.

Extinguishing the flame:

Once the robot is lined up with the candle, it proceeds to scan vertically for the flame. The entire arm shifts up until it reaches the top of its range. During this time the flame sensor is continuously checking for a flame. If it reads a value lower than the minimum value it currently has stored, it will store the new value and its location on the potentiometer. Once the arm reaches the top, it will check if the minimum value is lower than the threshold to determine that it saw a flame. If the minimum is higher than the flame threshold it will go back to scanning. However, if the minimum is lower than the flame threshold it will attempt to point the cannon at the flame. In order to do this, it uses PID control to move the arm 200 ticks lower than where it saw the flame. This adjustment is because the flame sensor points at a slightly different area then the cannon. Once the cannon is aimed the slip gear is rotated approximately one rotation and fires the cannon. After the flame has been extinguished, the potentiometer is converted into a Z value which is displayed on the LCD screen along with a signal that the candle is extinguished.

7. Conclusion

RBE2002 provided a suitable challenge in a small package: with an enclosed area, locate a fire and put it out. The numerous sub challenges, avoid the wall, keep track of the net position and net angle, accurately put out the fire and attempt to return to the original position, set a higher precedence for sleek robot performance with integrated sensor analysis. To avoid the wall, 2 distance sensors had to be integrated with a bell curved P control, to level the robot and bring it close to the wall. In addition, the IMU was used to accurately turn any specified degrees. To keep track of position, and return home we used the method of instantaneous centers to gain the current position. This task especially brought to light the difficulty and reward of the mathematics associated with robotics, and gave us an introduction to the world of position mapping. To put out the candle, we ambitiously decided to use an air cannon, a stretch that ended up working better than we could have ever hoped. In this challenge, there needed to be a direct precision of the robot in comparison to 1001 and 2001. In previous classes, if the robot was unable to complete a task, say it was slightly too high to pick up the fuel rods, we could have the robot shake and jiggle to approximate all errors. In 2002, if the robot approximated a wall, the robot easily could run into said wall, and lose its position, as the wheels would begin to slip. This course served as an introduction to the mathematical and programming challenges presented at the peak of robotics, and gave us all a sense of the sensor synergy needed in the real world.

8. Bibliography

[1]https://mycurvefit.com/
[2]https://en.wikipedia.org/wiki/PID_controller
[3]https://www.arduino.cc/en/Reference/HomePage
[4]https://canvas.wpi.edu/courses/3400/pages/lab-materials-and-final-project
[5]https://canvas.wpi.edu/courses/3400/pages/lecture-materials
[6]https://www.pololu.com/product/2826

9. Appendices

See Attached file for view of master flowchart

This page was intentionally left blank.