Roborodentia Robot
(Duct Tape Craze)
Senior Project, Spring 2017

Tarrant Starck
Advisor: Dr. John Seng
# Table of Contents

Introduction 3

Problem Statement 3

Software 5
- State Switching 6
- Line Following 7
- HaltAtWall 8
- TurnAround 8
- Turn 9
- Grab Rings 11
- Score 11

Hardware 12

Mechanical 14
- Body 15
- Claw Platform 16
- Sensors 17

Budget and Bill of Materials 18

Lessons Learned 20
- Start Early 20
- Team Management 20
- Line Following 22
- Turning Around with Only Two Wheels 22

Conclusion 23

Appendix: Code 23
Introduction

Roborodentia is an annual autonomous robotics competition held at Cal Poly in April. In 2017, Roborodentia was a head-to-head double elimination tournament with the winner being the robot that moves more rings onto the scoring pegs. For this year’s competition, I designed, built, programmed, and tested a robot.

Problem Statement

If robots wished to enter the competition, they have to meet certain specifications. During the 2017 Roborodentia Competition, the robot must:

1. Be Fully Autonomous and Self-Contained
2. Possess a 12” x 12” footprint or smaller at the start of the match.
3. Be no taller than 15” at the start of the match
4. Not Disassemble into Multiple Parts
5. Not use any RF wireless receivers/transmitters during the match
6. Not damage the course or the contest rings
7. Not modify the rings in any way
8. Intentionally Damage or Impede the Operation of an Opponent’s Robot
9. Not Fly for more than 20 seconds.

If any of these rules were not followed, the robot would be disqualified from the tournament.
In order to win a match, the robot must score more points than its’ opponent. It can do this by acquiring rings from the supply pegs and transporting them to either of the scoring pegs or the ring dump.

Each gives a different value for placing the rings on them:

- **Primary Pegs (#1)** : 4 points per ring
- **Secondary Pegs (#2)** : 6 points per ring
- **Ring Dump (#3)** : 2 points per ring

To further complicate things, there is a flag in the middle of the arena that either robot can raise. If the robot’s flag color is raised, any rings they score are worth an additional 50% of their original point value.

After the 3 minute match, the points are tallied and a winner is declared.
For further information visit:

https://drive.google.com/file/d/0B4AMo3oO1f0ycENZTnoyaUk2MTA/view

**Software**

Duct Tape Craze was programmed in C for the Arduino Mega 2560.
Duct Tape Craze has 2 Line Following Sensors mounted on its front and a button for when it hits a wall. By sensing when both sensors hit a line, the robot knows when to advance to the next state and to perform the next set of actions.

**State Switching**

The State method controls everything the robot does. It connects the rest of the methods together and tells the robot what to do and what is next. It has 11 different states which can be simplified down to 6 different methods. Each state is shown below.
Line Following

I would say that the Line Follow code was the most important section of code for Duct Tape Craze. Line Follow enables the robot to advance to the next state and positions the robot so that it is in a good position for that state. The two Line Following Sensors were used to detect the line so that the robot remained on said line. When the robot arrived at where it needed to go, both sensors would light up, telling the robot to start the next state.

```c
int follow(int currentState)
{
    int mid = middle_sen.read();
    int right = right_sen.read();
    int diff = mid - right;
    int out = currentState;
    if (diff < -100 && right > lineStandard)
    {
        motor(LMotor, FORWARD, turnSpeed * 2);
        motor(RMotor, BACKWARD, MotorSpeed);
    }
    else if (diff > 100 && mid > lineStandard)
    {
        motor(LMotor, FORWARD, MotorSpeed);
        motor(RMotor, BACKWARD, turnSpeed * 2);
    }
    else if ((right > lineStandard) && (mid > lineStandard))
    {
        motor(LMotor, BRAKE, 0);
        motor(RMotor, BRAKE, 0);
        out++;
    }
    else
    {
        motor(LMotor, FORWARD, MotorSpeed);
        motor(RMotor, BACKWARD, MotorSpeed);
    }
    return out;
}
```
HaltAtWall

This method does exactly like its’ name suggests, stops the robot at the wall. It is called in the Find Rings State and Score State. When the robot needs to pick up the rings or score, it must first locate and stop in front of the wall. This method does exactly that.

```c
void haltAtWall()
{
    delay(1000);
    int speed = MotorSpeed;
    motor(IMotor, FORWARD, speed);
    motor(RMotor, BACKWARD, speed);

    // while button not pressed
    while (digitalRead(FrontButton) == HIGH)
    {
        follow(0);
    }

    motor(IMotor, RELEASE, speed);
    motor(RMotor, RELEASE, speed);
}
```

TurnAround

Turning around was one of the hardest things to do with Duct Tape Craze. With only two motors powering the wheels and with only 1 set of Line Following Sensors, turning around was a challenge. The wheels needed to spin at different speeds to get the robot to turn properly so that the robot would be on the line when it finished. TurnAround starts the robot at the wall. It backs up till it hits the intersection and then turns around. It starts the turn ignoring the line sensors for a bit so that the sensors clear the line. Then the robot keeps turning till the left
sensor detects that it has hit the line, stopping the turn. Duct Tape Craze is now ready to begin line following.

```cpp
void turnAround()
{
    int mid = 0;
    int back = 0;

    while (!back)
    {
        Serial.println("back up");
        back = backUp();
    }
    Serial.println("stop");
    motor(IMotor, BRAKE, turnSpeed);
    motor(RMotor, BRAKE, turnSpeed);
    delay(500);

    Serial.println("start turning around");

    motor(IMotor, FORWARD, turnSpeed * 1.6);
    motor(RMotor, FORWARD, turnSpeed * 2);
    delay(1000);
    while (mid <= lineStandard)
    {
        Serial.print("turn around\n");
        mid = middle_sen.read();
    }
    motor(IMotor, RELEASE,0);
    motor(RMotor, RELEASE, 0);
    delay(400);
}
```

**Turn**

It may be asked why turnAround and turn are two separate methods. The reason for this is that the robot has to cross over a different number of lines in each case. During turnAround the sensors have to cross multiple lines before they are in the clear to detect the end of the turn.
Turn on the other hand, just needs the left sensor to cross the one line before it is ready to detect the end of its turn. Besides this small difference, the methods are pretty similar.
int turn(int currentState)
{
    int state = currentState;
    int motorSpeedLeft = turnSpeed;
    int motorSpeedRight = turnSpeed;
    Serial.println("turn method");
    delay(3000);
    int leftMotorDir = FORWARD;
    int rightMotorDir = BACKWARD;
    int sensorBorder = LSensor;
    int turnEnd = 0;

    if (state == TURN_RIGHT)
    {
        rightMotorDir = FORWARD;
        //motorSpeedRight = 0;
        sensorBorder = MSensor;
    }
    else if (state == TURN_LEFT)
    {
        leftMotorDir = BACKWARD;
        //motorSpeedLeft = 0;
        sensorBorder = RSensor;
    }
    else
    {
        // error
        return -2;
    }

    motor(LMotor, leftMotorDir, motorSpeedLeft * 1.6);
    motor(RMotor, rightMotorDir, motorSpeedRight * 2);
    delay(1500);

    while (turnEnd <= lineStandard)
    {
        Serial.print("turn\n");
        turnEnd = analogRead(sensorBorder);
    }
    motor(LMotor, RELEASE, 0);
    motor(RMotor, RELEASE, 0);

    return state + 1;
}
Grab Rings

This is the most simple method in the entire robot. After running haltAtWall, this method manipulates the claw so that it can grab the rings from the supply pegs.

```cpp
int grabRings()
{
    Serial.print("grab rings\n");
    motor(ClawMotor, BACKWARD, 150);
    delay(150);
    motor(ClawMotor, BRAKE, 0);
    return TURN_AROUND;
}
```

Score

This is another simple method very similar but slightly different than GrabRings. Just like in GrabRings Score starts of trying to find the wall using haltAtWall. After finding the wall, the robot backs up slightly to give the claw a bit of room. It then places the rings onto the scoring pegs using the claw method.
```cpp
int Score()
{
    Serial.print("score\n");
    motor(ClawMotor, BACKWARD, 150);
    delay(150);
    motor(ClawMotor, BRAKE, 0);
    haltAtWall();

    motor(LMotor, BACKWARD, 100);
    motor(RMotor, FORWARD, 100);

    delay(150);
    motor(LMotor, BRAKE, 0);
    motor(RMotor, BRAKE, 0);
    claw();
    claw();
    claw();
    return TURN_AROUND2;
}
```

## Hardware

The high level hardware design diagram is below.
Duct Tape Craze uses an Arduino as well as a Motor Driver Shield to control the robot. A picture of the Motor Driver Shield can be seen below.
The Motor Driver Shield is almost identical to the AdafruitMotorShield. For more information, go to https://playground.arduino.cc/Main/AdafruitMotorShield

**Mechanical**

Duct Tape Craze is a simple, somewhat functional robot with two motors for movement and one motor to control the claw. Below is the completed Duct Tape Craze.
Body

Duct Tape Craze uses a simple rectangular design to allow for as much work space as possible while still being easy to cut out. The base is ⅛ “ thick hardwood, allowing for easy assembly (aka. glueing) and a decent amount of strength. I decided to make the base quite large so that I could be flexible in the placement of wheels and sensors, while also giving it the strength to support the claw platform, which I wanted to be quite big. This made my robot quite heavy. If I were to design the robot again I would use hard plastic parts instead, which would provide similar strength while being a lot lighter.
Claw Platform

The Claw was difficult to design. I knew I would have to clear the wall but I didn’t know how I wanted to score. In the end I decided to go for a scoop/bucket method where I scoop up the rings and then drop them in the same manner.

As you can see from these views below, the claw platform extends quite far in front of the robot. I decided to do this because in the beginning I had no idea how I wanted to score. Having the claw platform so big gave me a lot of flexibility in placement of the claw.

Seen below are the front views of the robot.
Below are the top views of the robot.

Sensors

Duct Tape Craze has 2 analog sensors mounted below the claw platform. The line sensors are attached to a cardboard mount to get them closer to the line below. There is also a mechanical button located just above the line sensors. It is mounted directly on the hardboard claw platform with hot glue and duct tape. The sensors are 3/4ths of an inch apart, which likely caused problems in my line follow method. I should have spaced them closer together and used all them all instead.
Budget and Bill of Materials

There was no real budget for this robot, as I had no real experience buying robot parts. I also had no clue what I actually needed for the robot which lead to the purchase of extra parts that went unused. Also, since I was running late on the order of parts, many were ordered with rush delivery, bumping up the shipping cost. As such, the final cost of this robot is greater than it needs to be.

<table>
<thead>
<tr>
<th>Part</th>
<th>Vendor</th>
<th>Quantity</th>
<th>Price</th>
<th>Total</th>
<th>rush shipping</th>
</tr>
</thead>
<tbody>
<tr>
<td>Foamboard 30x40</td>
<td>Staples</td>
<td>1</td>
<td>7.89</td>
<td>7.89</td>
<td></td>
</tr>
<tr>
<td>Foam Display Board</td>
<td>Staples</td>
<td>1</td>
<td>15.79</td>
<td>15.79</td>
<td></td>
</tr>
<tr>
<td>Hot Glue Sticks</td>
<td>HomeDepot</td>
<td>1</td>
<td>5.27</td>
<td>5.27</td>
<td></td>
</tr>
<tr>
<td>Hardboard 2x4</td>
<td>HomeDepot</td>
<td>1</td>
<td>4.99</td>
<td>4.99</td>
<td></td>
</tr>
<tr>
<td>Duct Tape</td>
<td>HomeDepot</td>
<td>1</td>
<td>6.97</td>
<td>6.97</td>
<td></td>
</tr>
<tr>
<td>Wood Glue</td>
<td>HomeDepot</td>
<td>1</td>
<td>3.97</td>
<td>3.97</td>
<td></td>
</tr>
<tr>
<td>Electrical Tape</td>
<td>HomeDepot</td>
<td>2</td>
<td>0.99</td>
<td>1.98</td>
<td></td>
</tr>
<tr>
<td>AA Batteries</td>
<td>Cal Poly University Store</td>
<td>1</td>
<td>15.95</td>
<td>15.95</td>
<td></td>
</tr>
<tr>
<td>Metal 1/8 &quot; rod</td>
<td>Cal Poly University Store</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Hot Glue Gun</td>
<td>Cal Poly University Store</td>
<td>1</td>
<td>10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Hot Glue Sticks</td>
<td>Cal Poly University Store</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>AA Batteries</td>
<td>Radio Shack</td>
<td>1</td>
<td>16.19</td>
<td>16.19</td>
<td></td>
</tr>
<tr>
<td>Battery Holder</td>
<td>Radio Shack</td>
<td>1</td>
<td>1.56</td>
<td>1.56</td>
<td></td>
</tr>
<tr>
<td>Wire Kit</td>
<td>Radio Shack</td>
<td>1</td>
<td>3.64</td>
<td>3.64</td>
<td></td>
</tr>
<tr>
<td>USB 2.0 Cable</td>
<td>Radio Shack</td>
<td>1</td>
<td>15.19</td>
<td>15.19</td>
<td></td>
</tr>
<tr>
<td>SEEED Relay Shield</td>
<td>Radio Shack</td>
<td>1</td>
<td>10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Item</td>
<td>Supplier</td>
<td>Quantity</td>
<td>Unit Price</td>
<td>Total</td>
<td></td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-------------------</td>
<td>----------</td>
<td>------------</td>
<td>--------</td>
<td></td>
</tr>
<tr>
<td>Red Switch (button)</td>
<td>Radio Shack</td>
<td>1</td>
<td>1.36</td>
<td>1.36</td>
<td></td>
</tr>
<tr>
<td>Adaptaplug 0</td>
<td>Radio Shack</td>
<td>1</td>
<td>4.79</td>
<td>4.79</td>
<td></td>
</tr>
<tr>
<td>Replacement Adaptaplug Socket</td>
<td>Radio Shack</td>
<td>1</td>
<td>3.59</td>
<td>3.59</td>
<td></td>
</tr>
<tr>
<td>USB 2.0 Cable</td>
<td>Amazon</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td>2PCS 8mm Bore Inner Ball Mounted Pillow Block</td>
<td>Amazon</td>
<td>1</td>
<td>7.57</td>
<td>7.57</td>
<td>10</td>
</tr>
<tr>
<td>SunFounder L293D Motor Drive Shield</td>
<td>Amazon</td>
<td>1</td>
<td>10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>SunFounder Mega 2560 R3 ATmega2560-16AU Board</td>
<td>Amazon</td>
<td>1</td>
<td>17.99</td>
<td>17.99</td>
<td>18</td>
</tr>
<tr>
<td>SparkFun RedBot Sensor - Line Follower</td>
<td>Sparkfun</td>
<td>4</td>
<td>2.95</td>
<td>11.8</td>
<td></td>
</tr>
<tr>
<td>Jumper Wire - 0.1&quot;, 3-pin, 6&quot; (Black, Red, White)</td>
<td>Sparkfun</td>
<td>4</td>
<td>1.5</td>
<td>6</td>
<td>20</td>
</tr>
<tr>
<td>Pololu Universal Aluminum Mounting Hub for 6mm Shaft</td>
<td>Pololu</td>
<td>2</td>
<td>7.95</td>
<td>15.9</td>
<td></td>
</tr>
<tr>
<td>Pololu Stamped Aluminum L-Bracket Pair for 37D mm Metal Gearmotors</td>
<td>Pololu</td>
<td>2</td>
<td>7.95</td>
<td>15.9</td>
<td></td>
</tr>
<tr>
<td>Pololu Wheel 90×10mm Pair - Black</td>
<td>Pololu</td>
<td>2</td>
<td>9.95</td>
<td>19.9</td>
<td></td>
</tr>
<tr>
<td>8-AA Battery Holder, Back-to-Back</td>
<td>Pololu</td>
<td>1</td>
<td>1.95</td>
<td>1.95</td>
<td></td>
</tr>
<tr>
<td>USB Cable A to Micro-B, 6 ft</td>
<td>Pololu</td>
<td>1</td>
<td>2.49</td>
<td>2.49</td>
<td>16.45</td>
</tr>
<tr>
<td>Metal Gearmotor 37Dx52L mm with 64 CPR Encoder</td>
<td>Cal Poly Robotics Club</td>
<td>4</td>
<td>5</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td><strong>TOTAL:</strong></td>
<td></td>
<td></td>
<td><strong>277.63</strong></td>
<td><strong>79.45</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Actual Total</strong></td>
<td></td>
<td></td>
<td><strong>357.08</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Lessons Learned

Start Early

The most important thing that I learned over the course of building the Duct Tape Craze was how vital testing was. You have to test your designs, your parts, your code, and everything in between. All of this testing leads to reworking things which leads to more testing and so on. It takes a lot longer than you think to do all of this revision.

To make matters worse was the fact that I didn’t have a lot of background in robotics or even model building. I had no clue how to acquire parts, build, or even test the robot before I started this senior project. I also didn’t know any of the little things to make building models/robots easier, like what type of glue works best or how this robot part works better in this situation. I had to try and do research to figure everything out. What I couldn’t find I had to try and see if I could figure something out.

All of this lead to me not finishing in time for the Roborodentia Competition this year. While I didn’t meet the deadline I did eventually finish my robot. With what I have learned in the process of building it, I could probably rebuild it from scratch in half the time.

Team Management

One of the biggest issues that came up during this project was the fact that it used to be a group project. Both of us had no idea what we were getting into but we decided to do Roborodentia as
our senior projects. I knew him from another class and we worked well enough together so I had no issue grouping up with him. However, over the course of this project my partner proved to be a terrible teammate.

One of the issues I had with him was how group meetings were constantly delayed, ranging from an hour to a couple of days. To make things worse is that he would often ask to delay the meeting last minute. A couple of times I was walking over to the meeting and he would shoot me a text asking to delay the meeting, this being about five minutes before. It was ridiculous. He just didn’t seem to be considerate of my time.

Another issue was the fact that he never seemed to do much work. During each of the group meetings it seemed like I was the leader. I would bring up which design or parts we might use and he would go along with it, never showing up with his own ideas. While this could be ok, it turned out that all of the parts we selected to use with the robot were parts I found while researching online. This made it look like to me that he wasn’t doing any research into parts for the robot.

To make matters worse was when we split up the code for the robot. I asked him to write Arduino code for the scoring methods while I would handle line following and grabbing the rings, as well as a couple other things. In about three to four weeks he still had nothing to show for it. Arduino code isn’t hard and there is a lot of example code online. It should have taken him a couple of hours at most to get the code into a testable state. This was a big issue. I was testing and figuring out my side of the code with the expectation that he would finish his side. He never delivered as our senior project advisor split us up on my request before he could finish anything.
Line Following

I also made a stupid mistake in my line follow when I was early on in testing. The loop my line follow function was in had a three second delay in it which completely broke it. The line follow method is not complicated but it took me quite a while to finish it, partly because of this error. The other issue I had with my line follow method was that I was too light on my adjustments to keep the robot on the line. I eventually amped up the wheel speed which made line following a lot easier. To make my line follow method even more precise, having 2 line sensor arrays on the front and back of the robot would be ideal.

Turning Around with Only Two Wheels

Getting the Duct Tape Craze to turn around was one of the hardest things to do for this project. It was constant testing and adjustment only to find out it didn’t quite work out. I spent many hours playing around with the numbers of how fast I was turning the wheels to get the robot to turn around consistently enough that it was on the line for it to line follow. To make things worse was the fact that the wheels would change their speed at different battery levels due to changes in voltage. My wheel speeds right now work decently well at different levels of battery that my turnAround method works relatively well. If I was to remake the robot, I would have four motors and sensors on the front and back of the robot. This would make it easier to turn around on a spot as well as seeing where exactly the robot is in relation to the line.
Conclusion

The Duct Tape Craze ended up being a robot as barely working as its name suggests. When I started this project I had no idea what I was getting into. Trying to research and test everything by myself was a challenge, requiring a lot more time than I was expecting. If I were to make a new robot I feel like I could do it better and a lot quicker than before.

While I did miss the competition deadline, I did end up learning a lot about what robots can do and how they do it. I needed to learn about hardware and model building. I had to learn how to solder and measure parts. I learned how to write Arduino code. I learned the difficulties in testing a robot, constantly playing with the numbers in order to make the turns correctly. I chose this project in order to learn about robots, and I definitely did so.

Appendix: Code

https://github.com/tar9code/seniorProject/blob/master/senior_project.ino