SoccerBot by MakeBlock Robot
Antone Evans and Zach Goins

The University of Central Arkansas Computer Science Department Class in Multimedia

***Abstract*—In this technical report, we use the mBot starter/orion board as a model for our Multimedia project. Firstly, we constructed the robot then we created a simple program where the robot turns and drives to ensure it worked. Next, we checked our ultrasonic senor and wrote a simple program to ensure the robot was working correctly. Our bot is call the SoccerBot because it's goal is the hit three soccer balls into the goal without the help of anything other than the program. The program is written in arduino mode using the mBlock IDE.**

***Index Terms*—mBlock, mBot, Multimedia, SoccerBot**

I. Introduction

mBot is a low cost, easy-to-run robot kit for kids to get hands-on experience about graphical programming, electronics, robotics. mBot is an all-in-one solution to enjoy the hands-on experience about programming, electronics, and robotics and it is all about fun and creativity. [1] It comes with various basic pre-assembled options, including obstacle-avoidance car, line-following vehicle, remote control car, and can be used in multiple games like balloon bursting game, soccer playing, sumo, and etc. The mechanical body of the mBot is compatible with Makeblock platform and most of Lego parts, while the electronic parts of it are developed based on the Arduino open source ecosystem. This ensures that the mBot is compatible to have infinite extensibility, using any mechanical parts and electronic modules you need to turn it. We accomplished writing our code by using the Ardunio mode in the MakeBlock v3 IDE. We used python to program our robot; however, you can use the drag and drop features to easily program the robot if program is not your strong suit. Using the IDE you can transfer data to the bot. We connect the mBot to our laptops using the USB driver. Once connected you should connect to the bot, update your firmware, after adding code you will need to upload your code by click Arduino mode. Ardunio mode will send the information to the robot. You will see this indicated by the lights on the bots board lightening up and your IDE will constantly output the status.

II. How to get started using mBot

- You should first unpack and assemble Your mBot. Carefully unpack all your mBot components and download the instructions, before putting it all together. The instructions normally come in the box with the robot. However, if they do not you can download them.
- Secondly, test your mBot with the remote control. Before you start graphical programming, you can play your mBot with three simple modes pre-uploaded on the device. Mode 1: Remote manual control - User can use buttons to control the direction and speed of mBot. Mode 2: Wall avoidance robot - A robot that can avoid walls and obstacles while moving. Mode 3: Line follower robot - Line follower is a robot that can follow a path. The path can be visible like a black line on a white surface (or vice-versa).
- Thirdly, install the graphical programming software, mBlock. mBlock is a graphical programming environment based on Scratch 2 Open Source Code that makes it easy to program Arduino projects. (Available for Windows and Mac).
- Next, configure mBot communication options. You can configure your mBot's communication options one of three ways (depending on your model): via USB cable, via Bluetooth or via 2.4G wireless (see videos below for more).
- Finally, start with some pre-made lessons to help you get started. There are some great lessons available to help you understand your mBot and how it works. We recommend you start learning graphical logical programming with Scratch 2.0 The Adventures of Mike, then learn how to use Scratch2.0 to interact with mBot with the book mBlock Kids maker rocks with robots. Scratch 2.0 The Adventures of Mike has nine chapters with different projects. Kids can explore the world of Scratch 2.0 by creating a lot of interactive stories, games and animation in the virtual world, aiming to help kids learn basic logical programming and know how the Scratch2.0 works. [2]

III. Our Program

Beginning Stages In the beginning stages of this program, none of us were familiar with mBot or robotics. Therefore, we started off by reading the user manual. We noticed that there was a robot with a flat surface at the front that would be useful when hitting a soccer ball. Therefore, we took apartment the robot and began the reassemble it with the flat front. This process was very easy because the steps were very simple. It took us a hour or two to finish this task. We than began by learn how to use the software of the robot. We wrote simple program such as moving forward for five seconds, reserving, moving left or right. This gave us the basic idea how to move forward when writing our final program. We then moved on to working with our **Using Ultrasonic Sensor**. This was the most important part of our project. The sensor would detect which how far the soccer ball was and pick one of the three

programs to run which would end with a successful goal for the soccerBot. The ultrasonic senor works by sensing how far an object is from it. It calculates the distances in CM. This tool can be very useful because you can upload it to may different commands. Such as if (US GREATER THAN 10) turn right else if (US GREATER THAN 10 US LESS THAN 5) move left else move forward.

IV. Our Code

A. Hitting Ball Straight On

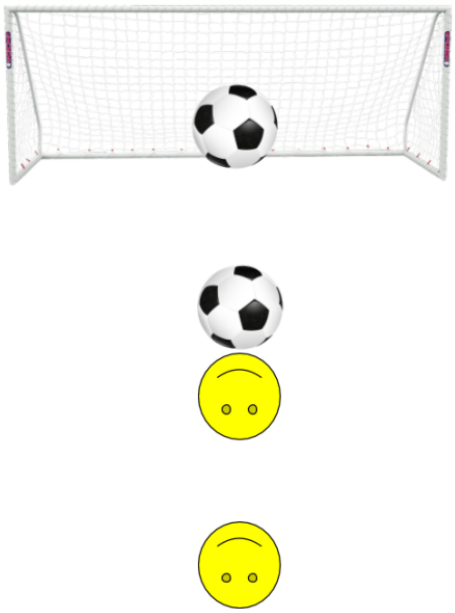

Fig. 1. This figure shows how the robot moves to hit the ball to the when the robot starts from the straight.

```
if((ultrasonic3.distanceCm()) GREATER THAN (50 ))
motor9.run(255);
motor10.run(255);
delay(3);
motor9.run(0);
motor10.run(0);
```

From the above code, we first check if the ultrasonic distance is greater than 50. If this case it true, this part of the code will run and the robot will hit the ball from a straight angel otherwise it will check a next case. The motor is the set to full speed of 255. As you can tell we have to set both motors, motor 9 is the right and motor 10 is the left. There is then a delay or wait for three seconds to allow the soccorBot to take the ball to the net. The motors are then set to 0 which stops the robot.

B. Hitting Ball from Left

```
if(((ultrasonic3.distanceCm()) GREATER THAN (40))
AND ((ultrasonic3.distanceCm()) LESS THAN (49)))
motor9.run(50);
motor10.run(0);
delay(1.5);
```

Fig. 2. This figure shows how the robot moves to hit the ball to the when the robot starts from the left.

```
motor9.run(50);
motor10.run(50);
delay(5);
motor9.run(0);
motor10.run(50);
delay(1.5);
motor9.run(255);
motor10.run(255);
delay(2);
motor9.run(0);
motor10.run(0);
```

From the above code, we first check if the ultrasonic distance is greater than 40 and less than 49. If this case it true, this part of the code will run and the robot will hit the ball from a left angel otherwise it will check a next case. The left motor is set to 50 and the right motor is set to 0 this will allow the soccerBot to turn left. There is a 1.5 second wait to allow the robot to turn. The bot then sets both wheels to 50 for 5 seconds to allow the robot to get to the middle of the field where the goal is. The robot then turns forward to face the goal. The the left motor is turn off and right motor is left at 50 this will allow the robot to turn forward. The robot then goes forward for 2 seconds at full speed. The soccerBot is then turned off.

C. Hitting Ball from Right

Fig. 3. This figure shows how the robot moves to hit the ball to the when the robot starts from the right.

```
if(((ultrasonic3.distanceCm()) GREATER THAN (30))
AND ((ultrasonic3.distanceCm()) LESS THAN (39)))
motor9.run(0);
motor10.run(50);
delay(1.5);
```

```
motor9.run(50);
motor10.run(50);
delay(5);
motor9.run(50);
motor10.run(0);
delay(1.5);
motor9.run(255);
motor10.run(255);
delay(2);
motor9.run(0);
motor10.run(0);
```

From the above code, we first check if the ultrasonic distance is greater than 30 and less than 39. If this case it true, this part of the code will run and the robot will hit the ball from a right angel otherwise it will check a next case. The right motor is set to 50 and the left motor is set to 0 this will allow the soccerBot to turn right. There is a 1.5 second wait to allow the robot to turn. The bot then sets both wheels to 50 for 5 seconds to allow the robot to get to the middle of the field where the goal is. The robot then turns forward to face the goal. The the right motor is turn off and right motor is right at 50 this will allow the robot to turn forward. The robot then goes forward for 2 seconds at full speed. The soccerBot is then turned off.

V. Project Obstacles

Over the course of this project, we have had several issues that we had to either research or troubleshoot. As mentioned before, we had the difficulty of our hardware failing. The micro sonic sensor had worked up until the week before the report was due. Sadly, we were unable to get footage of the program working correctly before the hardware failure. Another obstacle that we had to face was that the mBot IDE had faulty arduino drivers for our motherboard. This caused a lot of confusion in the first week of this project because nothing would run and we couldn't save programs to the robot. The last obstacle that we had to endure was that the wheel on the robot would become very loose out of nowhere. This would cause the robot to veer off track. This made it very difficult testing our programs because we would first check the program on a failed test. After altering the program, we would realize that the wheel was loose ,and we would have to revert the changes to our program. Therefore, our final result does not reflect this work. Our initial goal was to write a single program that would be able to handle all three cases; however, do to hardware problems. Our final project could not handle all three cases. The ultrasonic sensor suddenly stopped working. It continued to read the same number for instance it would not change from 30cm no matter how close or far you would move it from an object. Therefore, from our code above you notice that our ultrasonic sensor handles the three if else cases; therefore, without it the soccerBot would be clueless on which angle to hit the ball. However, all three cases would work if the ultrasonic sensor was functional.

VI. Solution

Our solution to this program was to write the code separately on three different workspaces - straight, from the left angle and from the right angle. We then eliminated the ultrasonic senor if statement and allowed the program to run. This way is more hard coded and not dynamic at all; however, to show what we were attempting to do this would work. However, for demoing the robot and time limits, we made a simple program and would change the speed of the robot.

VII. Contributions

Our team was a well oiled machine and both players gave great contributions. Antone worked on the hardware, coming up with the logic for the program along with doing needed research. He also wrote our paper. Zach initial worked with the ultrasonic sensor and got it working, he came up with the steps for the program along with putting together our poster board. Together, we both worked on the programming of the robot, testing and debugging. We used agile mythologist when dealing with our project. We first meet to plan out what we'd be using the mBot to do in this project. Secondly, we used the in class plan to schedule our week to week task to keep us on track. Next, we started to design what we want our robot to do followed by developing the code. We tested a lot. We test our code after each new major change we created. This was helpful because none of us were expects at this task; therefore, we could quickly find bugs before moving code to the major project. We then evaluated each other. If we worked on the project while the other individual was not present we would then send our teammate the tested code. He would have the chance to give feedback or to correct mistakes. This would help with human and logically errors.

VIII. Future Work

If given more time and additional tools, we could extend this project. We could have multiple robots play soccer against each other. We could use the ultra sensors to allow for the robots to avoiding hitting each other and only trying to attack the ball. Another interesting program that can be written is the imagine the mBot as a car and use it parallel park or drive autonomously. For instance, set up the RGB LEDs to work as turn signals. In addition, to help keep mBot on its track, the line-following sensor can serve as a trigger for a car alarm. mCore's IR transmitter and receiver allow messages to be passed between mBots, making it possible to coordinate their motion and stop them from colliding. [3]

IX. Conclusions

In conclusion, we have learned so much about hardware. We've learned that suddenly parts go bad and it's nothing we can do to control it. Hardware is unpredictable. The project has given us the chance to see a little of Computer Engineering and the integration between software and hardware.

X. Acknowledgments

This research was supported by the University of Central Arkansas Multimedia Class in "Robotics" which gave us a greater understanding of how hardware works. Thank you to Dr. Sun for helping direct us with our bi-weekly schedule report, our labs sign in sheet and the honesty when accounting for our contribution. Many thanks to makerBlock and his industrial design team for the mBlock. The documentation and videos provided by makeBlock has been a source of endless help during this process. A special thank you to my teammate for being a great source of help. We worked together like a well oil machine and faced every obstacle together. We helped to uplift each others ideas and it was a great time working.

References

[1] Makeblock, "MAKER WORKS TECHNOLOGY INC."
[2] Logics Academy, "mBlock," April, 2017.
[3] Vernier, "Coding with mBot: Self-Driving Vehicles Module," March, 2018.