Andria Grossenbaugh

Mikayla Kelley

The purpose of this project was to gain knowledge and understanding about GoPiGo robots and how one can use Python to create a working code that sends controls to the Raspberry Pi Robot to move in a path outlined for us in the directions of this project. The first part of this project was to use the basic commands already programmed into the GoPiGo program in order to make sure the team was comfortable doing simple commands and making sure that the robot moved in accordance to the command we typed into the program. The second part of this project was to look at a path, created by Professor Teate, and using Python, create a code that moves the robot in the way of the path outlined for each team.

The materials used for this project are:

1. Access to the robotics lab in HHS
2. JMU laptops that had the programs needed to create the code to run the robots such as Putty
3. The Raspberry Pi Robots
4. The path that each of our teams had to follow

Despite haven been given the materials, the path to follow, and the program software to write the code, during this project our team experienced multiple issues with the robot itself, the program installed on the school computers, and the security of the school wifi, which resulted in the acquiring of a router just for this project. The following issues range from very minor problems to problems that took more time to diagnose and fix before having the robot correctly function and run the desired path.

The first issue that arose dealt with remote connectivity and the James Madison University security problems when attempting to connect the GoPiGo program with the Raspberry Pi robots. The JMU network would not allow the program and robot to run on their connection. This was a major problem when first receiving the problem, but once the solution was discovered, the problem was an easy fix. This was a major problem at first because the teams working with the robots were unsure if they could work around the school’s security. Once the teams discovered that the robots could not work on the school’s network, it was a fairly easy solution to get another router not connected to the school’s wireless.

The step-by-step process to fixing this problem was fairly simple:

1. Discovered that the school wireless connection would not allow the use of GoPiGo programming and the use of Raspberry Pi robots
2. Get access to a different router that is not part of the school’s wireless connection
   1. In the case of this project, used an IKM-Thesis router
3. Set up the router for only the groups using the GoPiGo program and robots

We found that this was the easiest problem to fix because all it required the teams to do was get hold of a router that we could set up that only allowed the robots and the computers programming the robots onto the router’s wireless network. Once this issue was diagnosed and solved, the project could begin. Despite having other problems, this problem was the easiest to fix because it did not include actual programming or modifying the robot in anyway.

A second issue the teams encountered was with the new router that we got access to in order to allow our robots to connect with the computers that were writing the code (IKM-Thesis). This issue with this router is that the wireless on the computers would come up as limited which restricted the use of Internet on the computers. Another problem with the new router was that because three computer and three robots were using this one router, the computers and programs would run very slow. For this issue, it was necessary to do further research online to see what would make the router wireless connection faster because this knowledge was originally beyond my level. The following website was used to look up problems and solutions that were common with the use of routers:

<http://www.linksys.com/us/support-article?articleNum=136786>

This website was useful for suggestions about hot to get a stronger and continuous signal for the program and the robot. This website explained how the distance away from the router can be a source of the problem.

The step-by-step process on how we fixed the router problems was fairly simple:

1. Diagnose the problem of a weak signal from the router.
2. Relocate the router to the center of the room.
3. Make sure cabinets, walls, or any furniture was not blocking the router signal.
4. Move the router to a higher location for better signal
5. Have the computer closer to the router.

By moving the router free of obstructions and moving the computers closer to the router, the signal became stronger and the computers began to run faster. We thought the wireless signal was better, but the IP addresses became an issue when they kept changing every time the robot was turned off or the computer was turned off. This problem is discussed later in the report.

Another issue we encountered was that the batteries connected to the motor of the robot would die fairly quickly. This issue made programming and testing the code on our robot more difficult because the batteries had to be changed fairly often and we had to get batteries from Professor Teate so that we did not have to purchase our own batteries. In order to diagnose the problem, we would run the robot with basic codes and listen to the motor working and see if it sounded worn out. We could also diagnose the problem when we would run the code and the robot was on but would not run.

The step-by-step process on how we fixed the battery problem is fairly short since this was only a minor problem.

1. Once the problem was diagnosed, turn the robot off.
2. Disconnect the motor from the battery source
3. Gently slid the box containing the batteries out from under the plastic cover
4. Remove the 4 batteries and replace with new, rechargeable batteries or even attack a battery pack to the battery source which is connected to the motor
5. Turn the robot on and use until the batteries run out

Within our project we found that adding a battery pack to the batteries worked the best in fixing the problem of the batteries dying quickly instead of just changing out the batteries frequently. Also, we found that by only turning on the robot when coding and running the code saved power instead of keeping on the robot when not in use.

Another issue that we faced during this project that was harder to fix was that the motors connected to the wheels were uneven making the robot veered slightly left or right, which was also a problem because it never veered the same direction every time. This caused the path that we were trying to code more difficult to actual execute.

In order to diagnose and fix this problem we tried a few things.

Trouble-shooting without looking anything up:

* 1. Attempted to loosen the wheels slightly so that they were not bumping against anything on the actual robot, including the frame.
  2. Moving the robot from a carpet surface to a hard floor service to see if the material change of the surface helped the speed of the wheels
     1. We saw a alight improvement on the hardwood floor over the carpet because we felt the carpet fibers got tangled up in the wheels and was what could have caused the wheels to slow down and be different speeds.

Looking up solutions on forums for GoPiGo robots and seeing if anyone else had a similar problem:

1. <http://www.dexterindustries.com/GoPiGo/learning/python-programming-for-the-raspberry-pi-gopigo/add-trim-to-the-motors/>
   1. We learned from this website about the trim function in the Firmware of the robot which allows you to increase/decrease the speed of a wheel, one at a time.
   2. To start, disconnect the motors
   3. Have to update the robot to v1.3
   4. Go into the firmware using the sudo command
   5. Update the GoPiGo library
   6. Run the basic test of the commands in python file
   7. Check the firmware version first
   8. Check the trim value using the tr command
   9. A trim value of 0 will make the wheels the same speed

After looking at this website we gained knowledge about the wheel speeds and how to control and adjust them, but since we were low on time, Professor Teate took control of our robots and made sure the trim values were 0 and that the wheels were properly moving at the same speed.

After knowing that other people experimenting with these robots had the same problem as us, we found a GoPiGo forum that had people talking about the power issues and the wheel speed so we read through the comments and posts in hopes to finalize the wheel speed and the power issues for the batteries and the usage of the robot.

Forum website:

<http://www.dexterindustries.com/forum/?topic=power-supply-issues>

One of the long-standing problems we encountered was the changing of the IP addresses and attempting to make them static instead of trying to find which IP addresses were online and which computer connected to which robot. Our team, as well as others, when we first got the robot had to search the computers and type in different IP addresses into Putty until the basic test python code ran the certain robot we wanted it to control.

To diagnose and continue working with the robots:

1. The first day with the robots with typed in the IP addresses on the robot itself into the Putty to see if that robot was connected
2. If the computer network was timed-out and or the GoPiGo username did not pop up within 10 seconds, we would try a new number
3. Tried numbers 1-10 in order to find which one was online and connected to which robot
   * 1. Had to type in 168.192.1.\_\_\_\_ to try a bunch of numbers

Attempting to find a way to make static IP addresses on the computer:

1. First, we turned to Google and searched how to create a static IP address
   1. <http://www.howtogeek.com/howto/19249/how-to-assign-a-static-ip-address-in-xp-vista-or-windows-7/>
   2. From this website we learned about the cmd command into the computer
   3. Then we learned about the “ipconfig” command in order to show the IP address for that network
   4. Realizing that the computer was online and so was the robot, figured out using the “ping command” within the cmd command we knew which ever number we typed either connected or didn’t.
   5. We realized this was a problem and brought it to Professor Teate’s attention

The next few times of working with the robots we would have to keep typing and guessing numbers until we found the robot that the computer was connected to on the network. Professor Teate understood that this was time consuming and difficult, so he decided to work on the static IP addresses outside of class. Within a few days he had told us that he got the IP addresses to stay constant but only when we were logged on, he informed us that once we logged off we would have to find the numbers and addresses again. This gave us a lot of difficulty, but near the end of the work days for the project, Kadar was able to get us all together and tell us that he was able to get the IP addresses to stay the say so that when we worked on the project outside of class, we wouldn’t have to keep trying a bunch of different numbers.

This website our group looked at was about the problem with the changing IP addresses and the cause of the IP addresses to change:

<http://whatismyipaddress.com/keeps-changing>

We learned from this website that the reason IP addresses are likely to change frequently is:

* There are so many computers and people using computers at the same time all over the world that it is often expensive to keep the numbers the same so they change.
* Most commonly the IP address changes when you change over routers.

When completing this project, there were multiple problems that were faced. In addition to the endless worry about the IP addresses, the forward command veering slightly to either side, IOErrors that appeared out of nowhere on previously working code, we came together as three groups outside of class and worked on these robots until the day there were due. Despite the frustration of having multiple problems with the robots themselves, we learned that starter kits and first makes of anything, including robots, have some kinks that need to be worked out. We are so lucky to have had the help from other groups, Professor Teate, and the teacher assistant Erin go through these problems as well and learn more about the robots alongside of us.

When working on this project, we worked both in-class and outside of class. When first given this project, we had about 2 weeks to complete the project, but them with the difficulties that the robots gave us, Professor Teate extended the due date because the robots wer not functioning for the first few days we got the project. In total preparation for the presentation, we would argue that in class we had about 4 classes to work on the robots, which meant that we did most of the work on the robots outside of class. In total time, we would say that we spent about 12 hours outside of class on this project including writing the report, filming the movements of the robot’s commands, and making the robots actual work and connect to the network that worked with the computers.

The hardest part of this project was fighting the frustration. Just when we thought we got something fixed or corrected, something else would happen or cause an error. A majority of our time was spent diagnosing the problems we had, researching online, in forums, and by tri-and-error, and changing our codes in order to try to make the computers communicate with the robot and have the robot move in the path we wanted it to. Working alongside of other teams was really beneficial so that we were able to bounce ideas off each other about how to try and fix certain problems or throw out ideas about how we could alter our codes so that it would follow Professor Teate’s given path.

When given this project, the goal was to have the user enter one command that would run the entire path of the robot, This plan originally allowed the user to not have to manually enter the code to make the path themselves. Due to problems encountered throughout the project, including the actual manufacturing problems in the robots, in order to have a presentation we had to film manual codes for the robot that showed the different functions that the robot would have done if the problems weren’t happening. We had the robot go forward, backward, turn left, turn right, and blink the two LED lights attached to the back of the robot.

For future teams working on this kind of problem, or even with the GoPiGo robots, we recommend doing research prior to trying to code them so that if a problem pops up they will have a basic knowledge of what to try to fix. We also recommend that the companies and people working on these robots work with the circuit board and the mechanics of the robot to make sure they actually run smoothly. Lastly, we recommend before writing your own code to test the basic robot controls to make sure those are properly working.

Lastly, despite the frustration and problems that these robots had, we had a great time working outside of class with other teams and figuring out different things to do with them. We both learned so much from this project.