Building a Gesture Control Robot

Dmytro Kavetskyy

Computer Science 207

Trevor Tomesh

December 6, 2019

Table of Contents

Introduction

The project done for the Computer Science 207 class is a gesture control robot-car. This was realized because of a very long and early passion for remote controlled cars. The car project is consisted of two wheels controlled by DC motors. It will move forward, backward, right or left depending on the position of another separate device that communicates with the car and sends information about the inclination it is in.

The concept is pretty straight forward. The communication between the car and the gesture device is possible because of bluetooth devices mounted on them. These bluetooth devices have been programed to send and receive information from the gyroscope, which detects the angle the gesture device is in. After the gyroscope gets the angle, it sends the information to a Nano V3 device which has a code uploaded that converts the information to clear instructions, which are sent to the bluetooth device that finally sends this information/instructions to the robot.

There have been quite a few changes to the project, mainly because of bugs, unexpected errors, and bad design. Also, a lack of documentation and help delayed many milestones for this project.

Inspirations

Custom controlled robot-cars are quite common in the electronic and Maker world. We see these kind of projects on TV, magazines, and social media. This project really brought our attention because of a very peculiar thing; the gesture control aspect of it. Most similar projects and robots are controlled by joysticks or other electronic devices. Gesture control gives it a very interesting and cool aspect to it that really separated it from other projects and creations.

The idea came when I was browsing through Instructables.com and came across Shubham Shinganapure's post "How to Make a Gesture Control Robot at Home" [1]. Shinganapure's publication was very helpful because he went into detail about how to configure and setup the two main devices part of the project.

The base and skeleton of my project was completely inspired his post. He showed a good a compact way of fitting the parts together so they would take as little space as possible, as well as using the least amount of extra cables or unnecessary components. Also, I found it really smart that Shinganapure used two wheels for the robotic car, instead of 4, like other posts I have seen in Instructables. This saved a lot of time and space.

Although Shinganapure listed many important hardware and electronic components involved in the making of the robotic car, he did not list all of them, such as the Arduino Motor Shield. This became a problem because I realized this after I had already ordered all of the components. I even already received some. A different shield had to be ordered without knowing if it would fit and work for the project. Other components, such as a button were not listed, while clearly being present in his video tutorial.

Designing Process

Although it may look complicated, the designing process was thequite simple. There are two main components in this project; they are the robotic car itself, and the gesture control device.

The construction of the base and frame for the robotic car was quite straightforward. I used a piece of plastic that came with the kit containing the wheels and DC motors with cables connected to them. Because the kit is designed to be used with four wheels, I had to cut it in half. As said, its material is plastic so the cut was not very clean. Refer to Appendix A for an image of the frame. Here, I also included the DC Motors with the wheels attached, since it is quick and simple process. Super glue was used to attach the materials observed in the image. Next, a battery was installed, so it can power the arduino through a barrel jack port. A 7.5V set of 5 AA batteries was used in this case. This can be seen in Appendix B.

After this, there is only the top base left to attach. I used hot glue to the top of the DC motors and placed another plastic board on top of it, as seen in Appendix C. This is done so the Arduino BOARD is attached on top of it, since there is no space in between the DC motors.

The second device, the gesture control device, was even simpler to design. A simple arduino pre-soldered breadboard was used, as seen in Appendix D. I did not follow Shinganapure's design of this device because I did not have a breadboard big enough to solder and fit all of the electronics that attach to it. Also, a pre-soldered breadboard made the building process very simple and forgiving to mistakes.

Building Process

There was much more complexity involved in the building process of the robotic car project. A lot of adjustments and modifications to the original project by Shinganapure were needed to make to make it work.

The initial idea was to follow Shinganapure's prototype for the setup of the Arduino and motor shield for the robotic car. However, this was not possible, so a lot of improvisation was required to make it work. As seen in Appendix D, a motor shield was not used for this project. Instead, I used a L293D Motor Driver to connect the two DC motors to the Arduino controller. This was performed in a pre-soldered breadboard, to make things easier. I followed a sketch by codebender_cc in his Instructables.com post "How to Use the L293D Motor Driver - Arduino Tutorial" [2]. The DC motors connected directly to the motor driver, which also connected to pins 2, 3, 4 and 5 in the Arduino.

After completing and setting up codebender_cc's sketch, I connected a bluetooth HC-05 device, as seen in Appendix E, to the same breadboard. The bluetooth was connected to the 11 and 10 pins, set as RX and TX pins later on in a program.

The gesture control was not as complex to build, as Shinganapure provided a very clear and concise sketch to follow, as seen in Appendix F. This device is formed by a Nano V3 device, an HC-05 Bluetooth device, and a Gyroscope component. In this case, soldering work was needed because the Nano V3 and Gyroscope units did not have contact with the holding pins that connected them to the pins from the breadboard. The end result can be seen in Appendix G. Not much else was needed or involved in the build of the remote control device.

User Manual for Robotic Car

The set up of the gesture control robotic car is easy to do. First, the two bluetooth devices must be set up individually. One of them as a slave device, and the other one as a master device. Please refer to Appendix H for the code and instructions to setup the two units. The instructions are found in “Readme.txt”.

Once the bluetooth devices are setup, it is time to configure the Nano V3 and Arduino microcontrollers. Download the program “RemoteControl.ino” located in Appendix H. Disconnect the bluetooth device from the remote control device, and connect the Nano V3 unit to the computer. Upload the code and disconnect the nano and plug the bluetooth back in the circuit setup.

Now it is time to set up the Arduino UNO. Once again, disconnect the bluetooth unit from its circuit, connect the Arduino to the computer, and upload the program “Gesture_controled_Robot__car_unit_.ino” found in Appendix H.

To play with the robotic car, simply connect the robotic car itself to a power source (through the Arduino UNO barrel jack and following the codebender_cc’s sketch), as well as the gesture control unit. Allow the bluetooth units to pair between each other, and begin to play by changing the angle of the gesture control unit. The robotic car should follow the direction of the angle you point the remote control unit.

The positive thing about the car having a bluetooth device is the opportunities it gives. The robotic car can also be controlled with the code it already has incorporated with a smartphone app, found in Appendix I. Simply download the app, turn on the bluetooth, connect to the HC-05 device with either “000” or “1234” as the password, and begin playing. The robotic car can be moved in any direction with the buttons in the app.

Setbacks and Failures

A lot of setbacks and difficulties appeared along the design and build of this project. First of all, I realized that the Arduino shield that I obtained did not work properly or had the necessary extensions to connect two DC motors, as well as the bluetooth module. As previously mentioned, the use of a motor driver was necessary to make it work. The setback of this is the amount of time it took me to figure out and discover that my motor shield was the wrong one. I did not understand why my DC motors were unresponsive after having already assembled and configured everything. I had to disassemble most of the robotic car and start over. Also, it is relevant to say that I did not come across codebender_cc's post right away. I had to watch a lot of videos and read a lot of documentation before to know what was wrong and going on before I came across his Instructables post.

While testing the robotic car, I realized that I was not using the right voltages; I was using 6V in total. The DC motors would spin really slow and inconsistent. I realized that the best practice would be to increase the voltage to the motor shield, so more power would come to the DC motors, and then separately power the Arduino UNO. Unfortunately, I found all of this out too late, so the only way to make it work was to use a greater voltage and power the arduino through a barrel jack, instead of through the breadboard, and then send power to the arduino this way.

I also faced a lot of problems with the gesture control device. When trying to upload the program to it, it would show many warnings, and it would not upload. I realized a few code bad practices in the program, and I fixed them. Then, the code would still not upload. After trying many different things and a lot of time, I realized that the Gyroscope and the Nano units did not make full contact with all of the pins that connected them to the breadboard. I soldered the pins that I needed for my sketch, and the code uploaded nicely with no longer any warnings or problems.

The pre-soldered breadboards were not perfect. A lot of cables were needed to replicate the sketches for the robotic car and the remote control unit. There were a lot of times of confusing, and it was frustrating when cables would come out of their pins, losing track of where they were located.

At the end, when testing the completed gesture control robotic car, the bluetooth devices did not pair between each other, so I reconfigured them following the instructions listed in Shinganapure's post two more times. They still did not connect between each other. Because of this, the interaction between the remote control and the robotic car was not possible. I came up with an idea. I decided to download a bluetooth RC app that could connect to a bluetooth device and send commands. As mentioned, Appendix I contains the app. I downloaded it, and connected it to the robotic car through its bluetooth unit. The app connected fine, but the robotic car would still not move no matter what I pressed in the app.

Milestones

Very few of the milestones were met. The only ones are the components delivering in reasonable time, and finishing the project. In between, there have been many setbacks and problems that delayed the milestones.

The project should have been quite function by November 27. The build process took way longer than expected, so this milestone was not met at all.

Also, the programming milestone was also a failure. The warnings and errors from the gesture control device delayed a lot of time, because of the research and all of the testing needed to make the program work.

Regardless of all of these failures, the project was finished in a timely manner.

Conclusion

I loved the challenge of designing and building this project. I am proud of what I accomplished in such a short time. Although it was very frustrating not having some of the right parts, that pushed me to think outside the box and find creative and new ways to solve problems, such as building a circuit without a motor shield. It is true that at some points I felt so hopeless that I literally wanted to drop the course, but I overcame those thoughts and began to think of ways to solve the problems I faced. For most cases, I did solve those problems and kept moving forward with the project.

For the most part, it was very fun and interesting to build this project. I always thought that building something like this was out of my reach; that I needed to be super intelligent to do something similar. I learned that this is not the case. There is information about how to do things everything. I just have to go, find it, and get to work.

This gave me a lot of confidence about future projects that I may have in mind. The Computer Science 207 class has opened new doors and visions for me. I feel more confident with my electronic and problem solving skills. I feel like this is a class everyone should take. Also, I really enjoyed the coding part for the project. I got to see programs working in real time, which is very motivating to keep learning and programming.

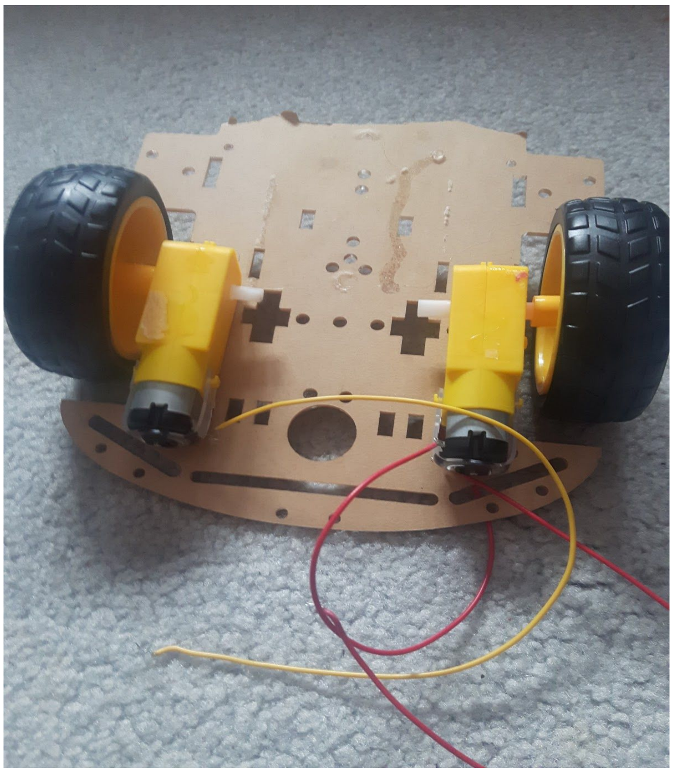
References

[1] Shinganapure, S. (October 7, 2019). How to Make a Gesture Control Robot at Home. Retrieved November 25, 2019, from https://www.hackster.io/shubhamsuresh/how-to-make-a-gesture-control-robot-at-home-a3f4a4

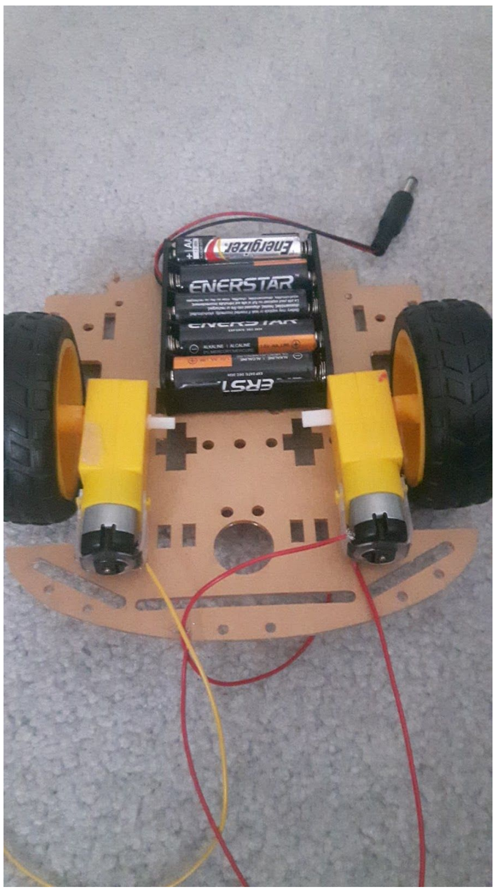
[2] codebender_cc. (n.d.). How to Use the L293D Motor Driver - Arduino Tutorial. Retrieved December 02, 2019, from https://www.instructables.com/id/How-to-use-the-L293D-Motor-Driver-Arduino-Tutorial/

Appendix

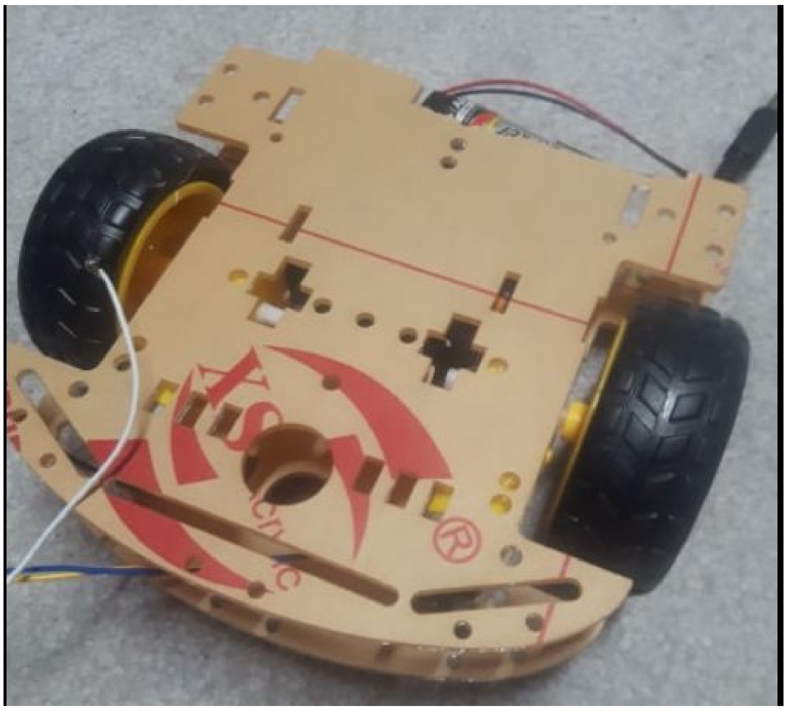
Appendix A: Frame Base and Wheel Setup



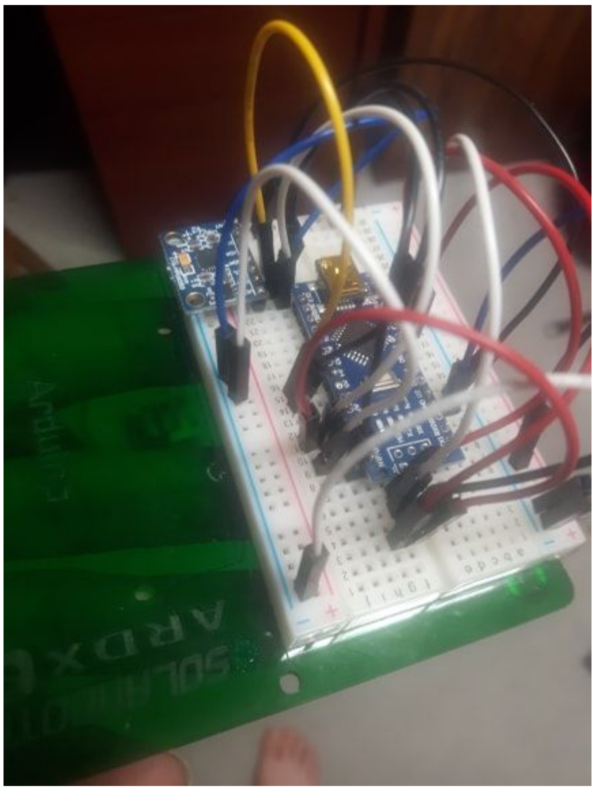
Appendix B: Robotic Car Frame with Power Supply



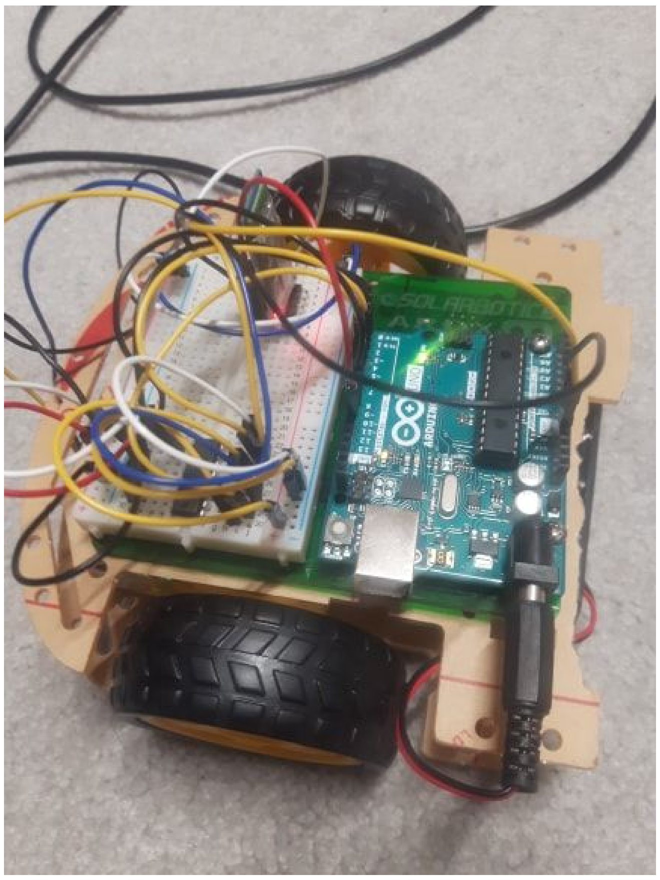
Appendix C: Robotic Car Frame and Base



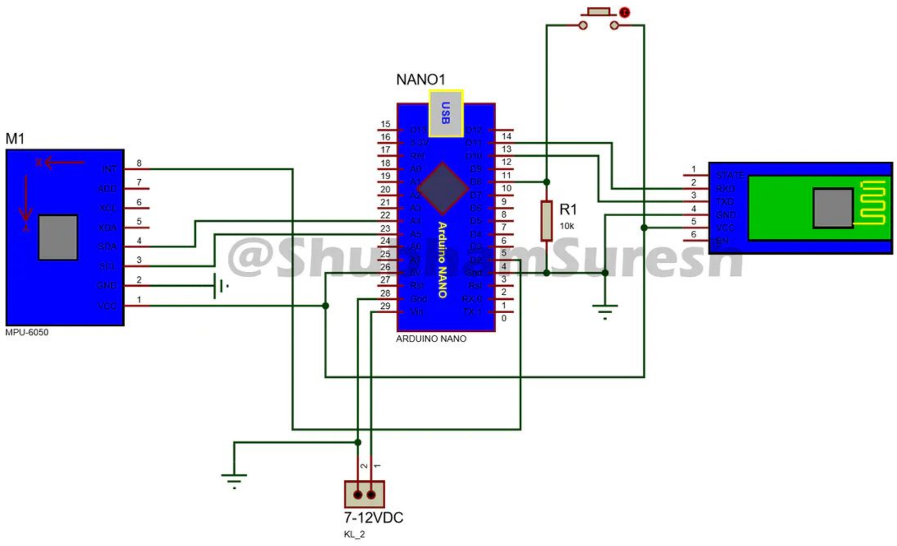
Appendix D: Pre-soldered breadboard



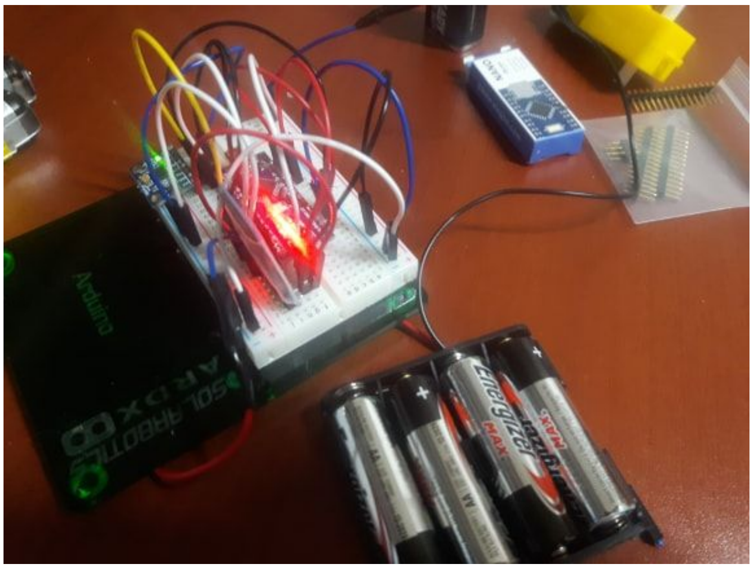
Appendix E: Robotic Car Setup



Appendix F: Sketch for Control Gesture Device



Appendix G: Remote Gesture Control Setup



Appendix H: HC-05 Setup

https://github.com/Dmytrocode/CS207_UofR/tree/master/Code

Appendix I: Smartphone RC App

App through Google Play Store found at

https://play.google.com/store/apps/details?id=braulio.calle.bluetoothRCcontroller&hl=en_CA