CSC 615 Spring 2021

Final Project Documentation

Team Autonomous

Team Members:

Name	ID	Email	GitHUb Username
Pramod Khatri	920831584	pkhatri1@mail.sfsu.edu	pramodkhatri10
Bikram Tamang	920465296	btamang@mail.sfsu.edu	btamang9295
Bhupendra Chaudhary	920113386	bchaudhary@mail.sfsu.edu	cbhupendra7
Gerardo Ochoa	918631875	gochoa@mail.sfsu.edu	shadow6188

Project GitHub URL:

https://github.com/CSC615-Spring2021/csc615-group-term-project-pramodkhatri10

GitHub primary username: pramodkhatri10

Task Description:

In a team of four students, we were given a project to build our own physical self-driving robotic car from scratch. Our car would detect lanes, navigate autonomously on lanes and steer accordingly . The car would recognize obstacles and within 20 centimeter from obstacles the car would stop and go around the obstacle and go back to the lane and finally complete the lane course designed. We will be using C to program in on the raspberry pi for our project.

Building the Robot:

It was important to set up both the software and hardware correctly in building the project.Below is the details on building the car with pictures:

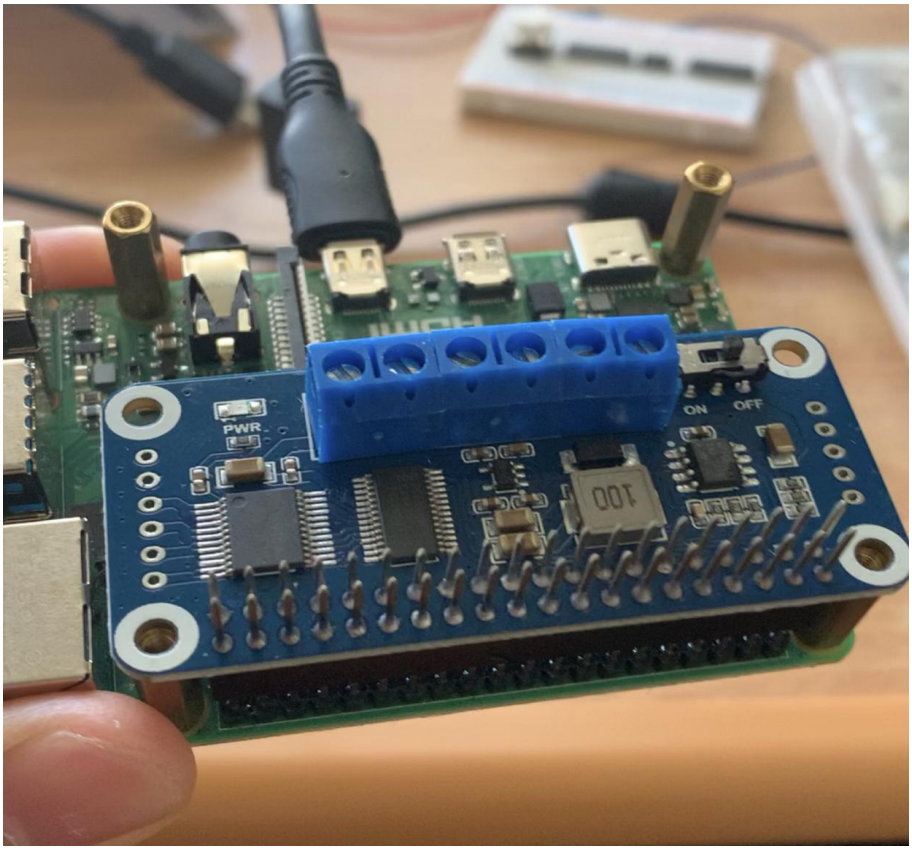


Fig: Motor Driver Hat mounted on top of raspberry pi 4

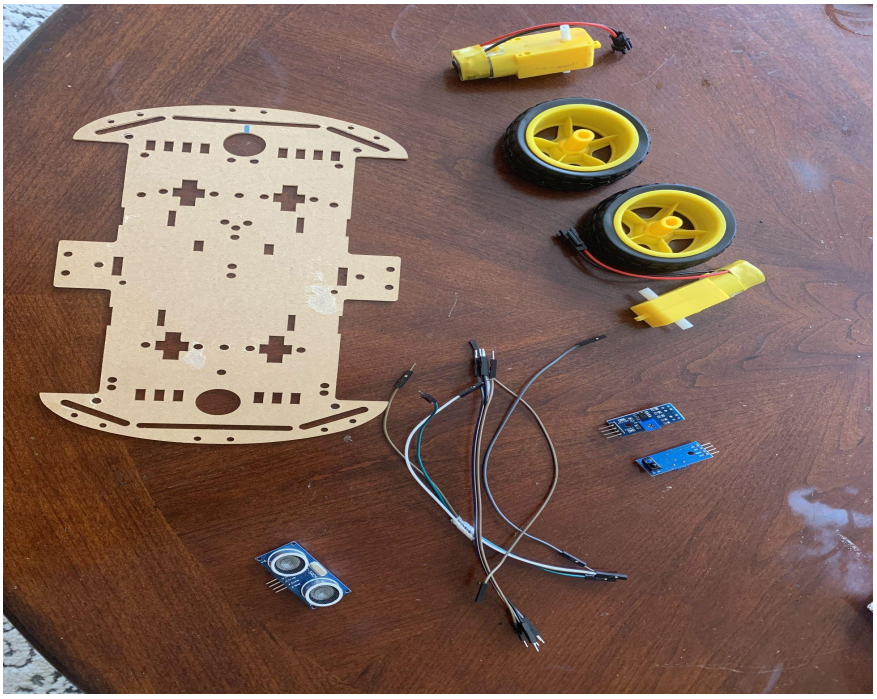


Fig: Car chassis frame with some wires, motors wheels and sensors before mounting them together

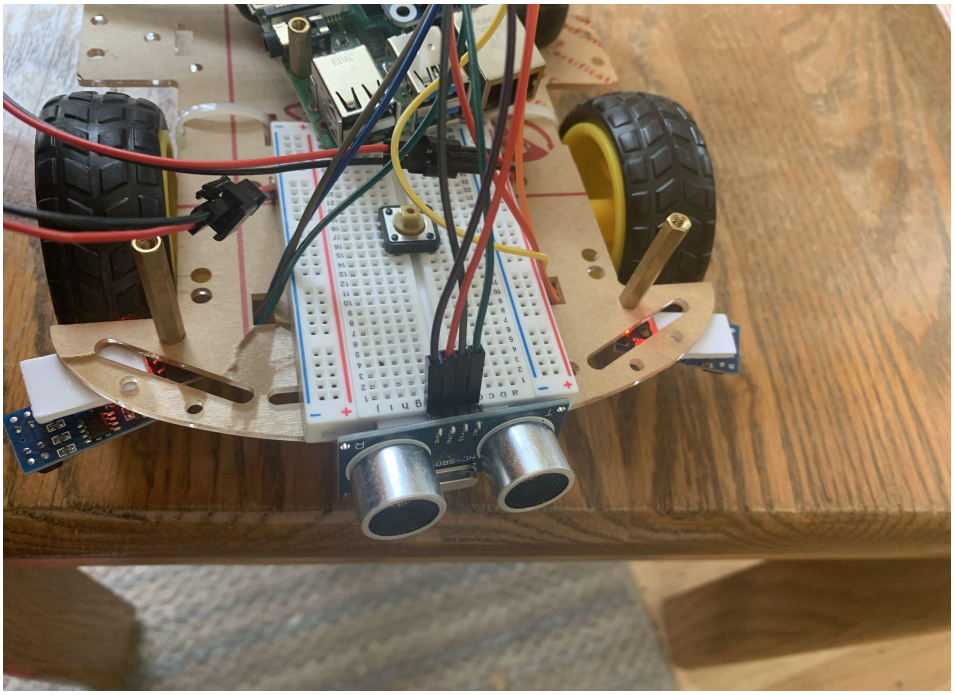


Fig: Motors and wheel mounted with car frame. Lane sensors and echo sensor attached and setup in the pi

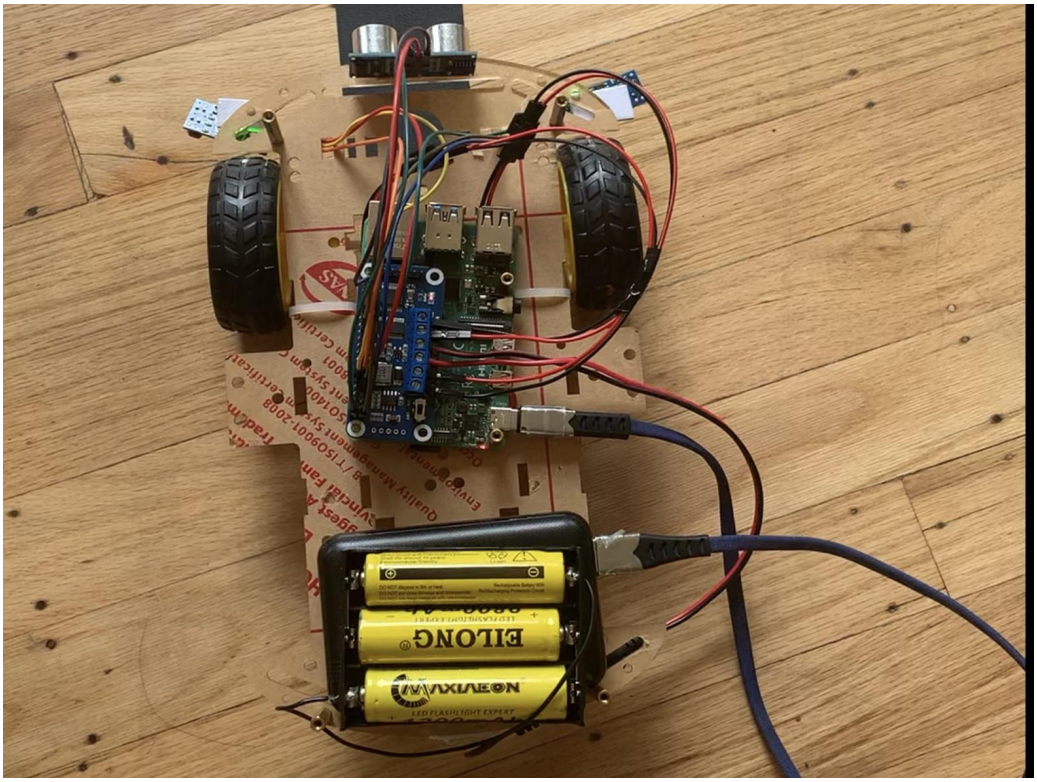


Fig: Top view of the car

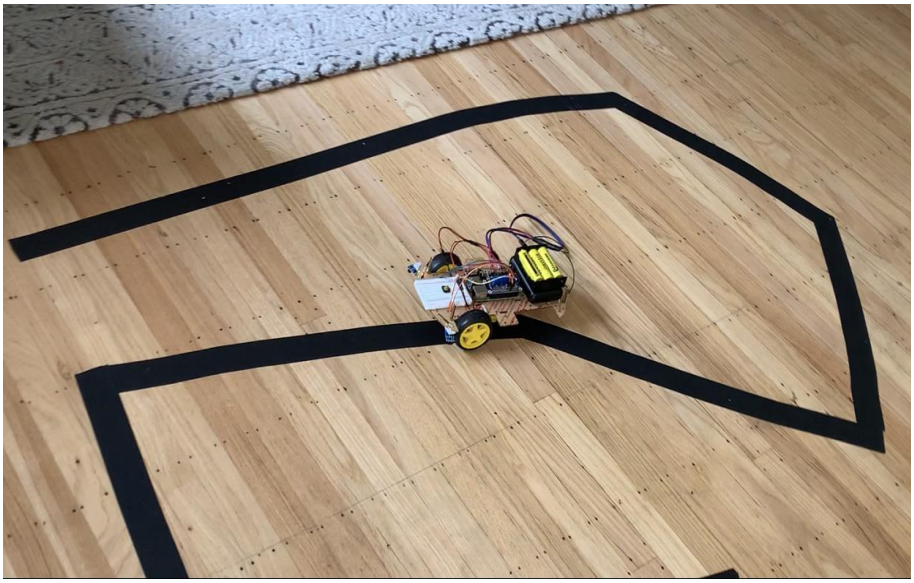


Fig: Car running autonomously on lane

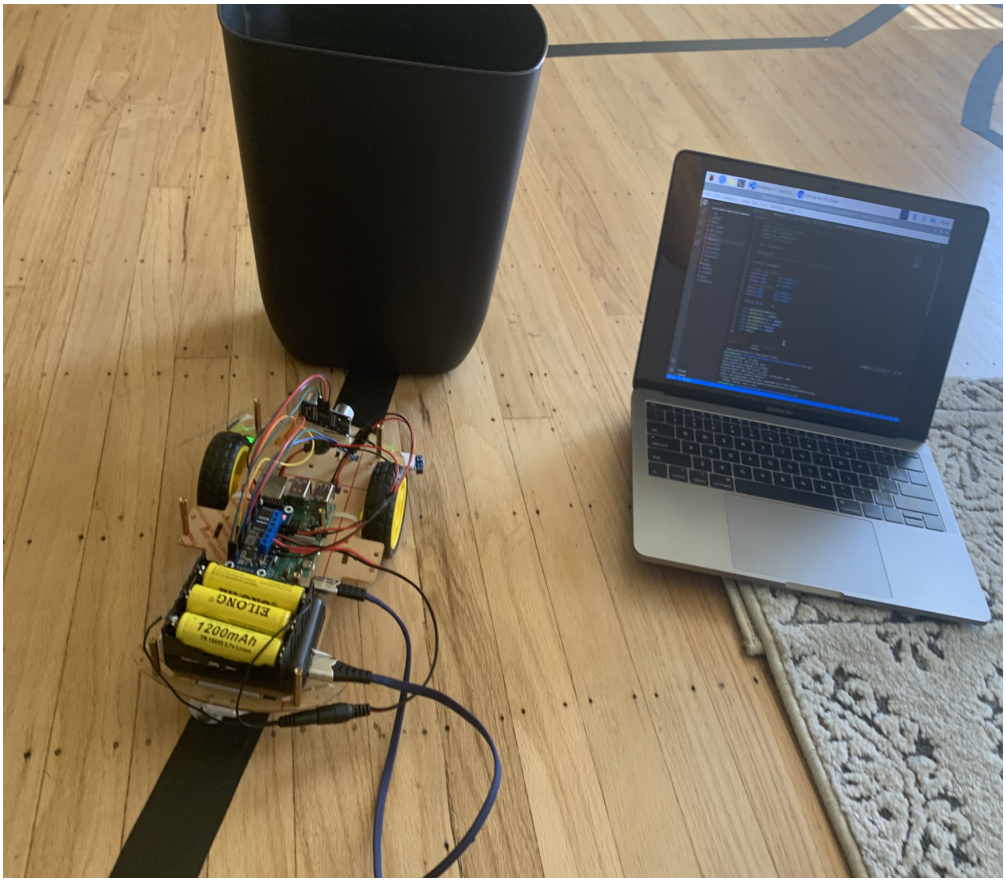
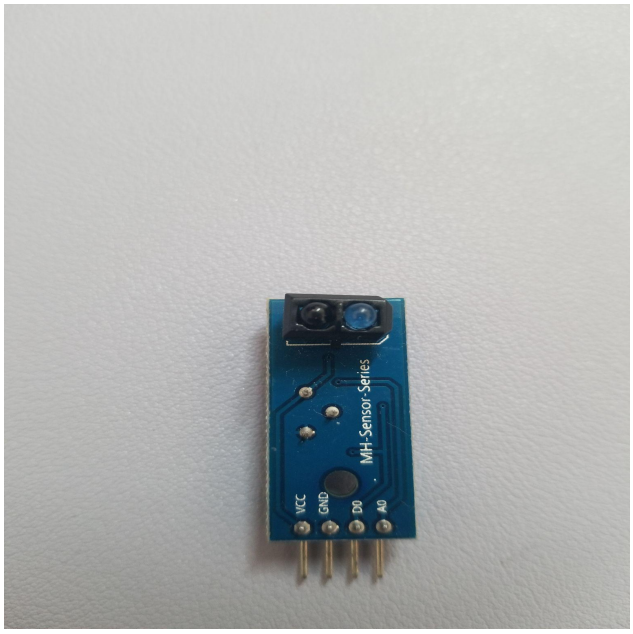
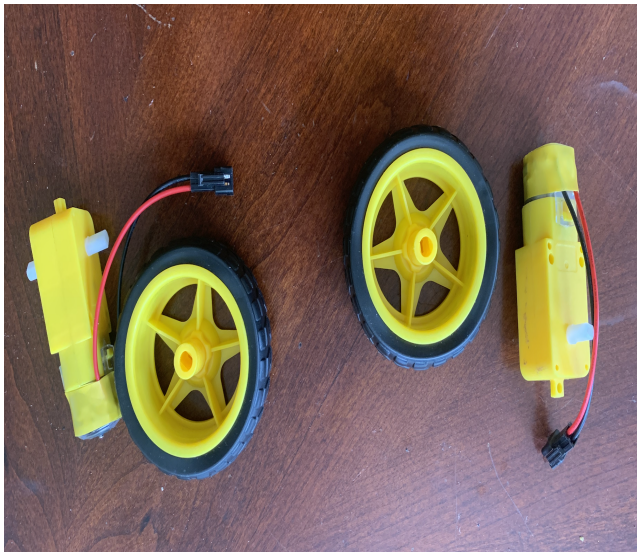


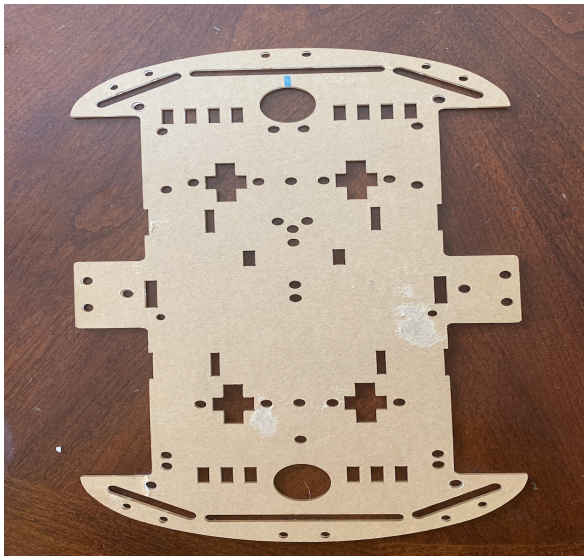
Fig: Car detecting obstacle and on its course to avoid it

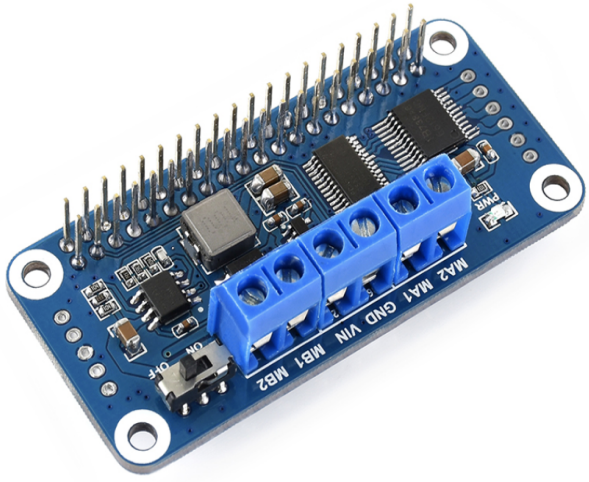
Parts/Sensors Used:

2x TCRT5000 Lane sensors
1x Ultrasonic Module HC-SR04
2x motors
2x wheels
2x chassis frame
1x power bank
3x 9800mAh Battery Rechargeable
1x Motor Driver Hat









Libraries/Software Used:

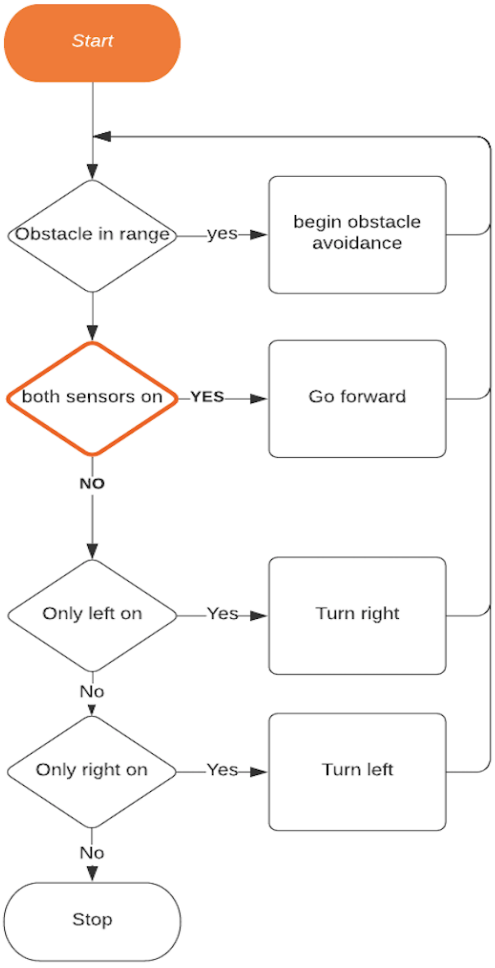
- Wiringpi.h :

  http://wiringpi.com/reference/

- Motor Driver Hat Sample Code:

  https://www.waveshare.com/wiki/File:Motor_Driver_HAT_Code.7z

Flowchart of Code:



Pins Assignment:

1. HC-SR04 Sensor

   VCC → 5V power //physical pin 1

   Trig → GPIO 5 //physical pin 29

   Echo → GPIO 6 //physical pin 31

   Gnd → Ground //physical pin 39

2. Right TCRT500 Sensor

   VCC → 3V3 power //physical pin 17

   GND → Ground //physical pin 14

   D0 → GPIO 27 //physical pin 13

   A0 → not used

3. Left TCRT500 Sensor

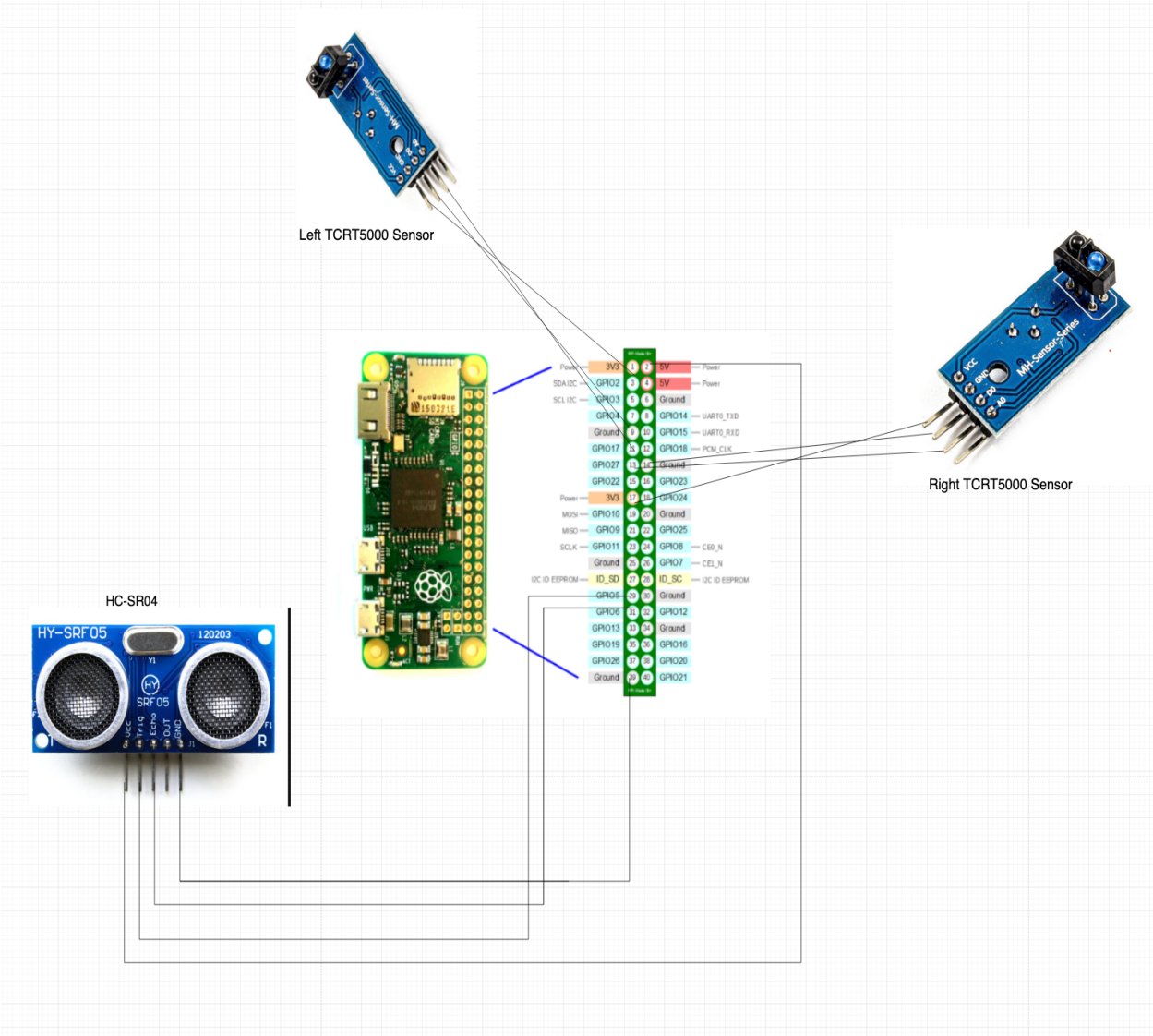
   VCC → 3V3 power //physical pin 1

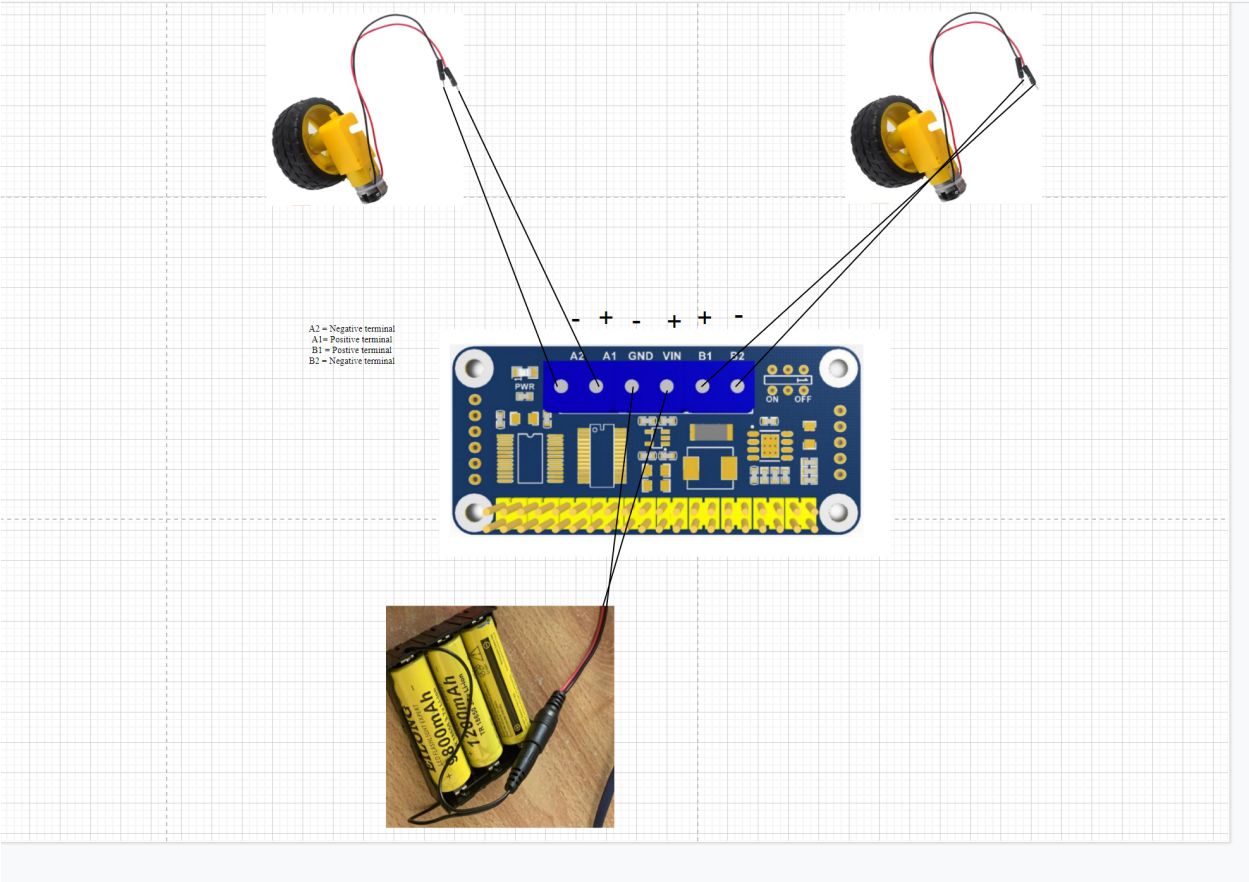
   GND → Ground //physical pin 9

   D0 → GPIO 17 //physical pin 11

   A0 → not used

Hardware Diagram:





What worked well:

At first, our team thought of using four motors but then we had issues with the waveshare board and we decided to use two motors only with Motor Driver Hat on raspberry pi. Both the motors worked well on both rough and plain surfaces.

After running the motors, our next priority was to make it follow the lanes. We first thought of using three IR-TCRT5000 sensors and it worked fine until the car went to the share ninety degree turn. So, we only used two IR-TCRT5000 sensors and it worked fine every time. It looked quite simple that we were using two lane sensors only, but it was clinical and it got our work done. After making the car follow the lane and steer accordingly, our next priority was to

detect obstacles and make sure not to hit the obstacle first. Using one HC-SR04 ultrasonic sensor, we were able to detect obstacles and make the car stop when the distance between car and obstacle is around 20 cm. It worked perfectly.

What were issue:

Our last priority was to make a stopped car go around the obstacle through a clear path and come back to the lane and keep going to complete the course. We implemented giving direction by switching one motor on and other off (left turn then go for some seconds and take right turn and get back to lane). As the car is stopped detecting obstacles, it takes a sharp right turn and goes forward for about 3 seconds, and then takes a quick sharp left turn for about 6 seconds to come back to lane. This did work, but not every time and our team believed this was one feature we could not implement completely.