PHYS 124 Project Report

Yuntong Z.

June 26, 2024

Group Member: Yuntong Zhou.

1 Motivation and Overview

Smart transportation has been an ongoing field of technology development that has attracted attention from all industries. Most of these projects focus on large-distance auto-driving, such as Tesla models. However, pin-point, short-distance smart transport has not been as extensively investigated, despite its utility in daily life like automated delivery carts.

The purpose of the project is to design a smart transport prototype that could safely carry loads to a set destination nearby. This project represents the prototype as a smart car, but it can be adapted to smart carts or other transport devices.

2 Functional Definition

The transport prototype has a 4-wheel DC-motor-powered system. The prototype is connected to a GPS module and equipped with a Wifi Board which allows it to receive real-time user inputs. It contains a distance sensor system that detects nearby obstacles and adjusts the path accordingly. Users are able to text control the motion of the prototype via the app telegram, from which an Arduino library called Universal Telegram Bot Library can convert the text messages into commands for Arduino.

3 Sensors

The project uses two Ultrasonic Sensors HC-SR04 to determine when the smart car is approaching barriers or objects. An example of connection of pins to the Arduino board is illustrated by the following figure. The trigger pin reads the input while the echo pin reads the output, which is the time duration from the sending of the signal to the reception of its echo. `pulseIn(pin, value)` is used to read the sensor output and convert it to distance. The sensor needs to be triggered on `HIGH` to give a pulse. Further reference can be found from this document.

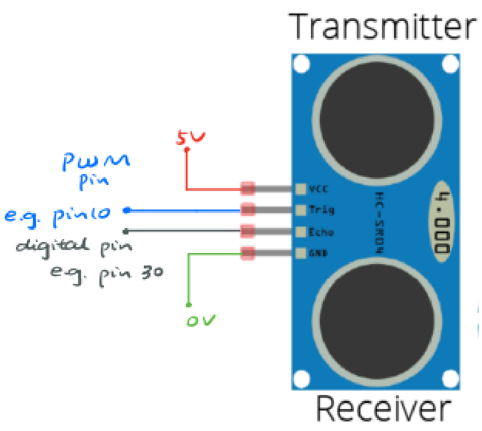

Figure 1: Pinout for Ultrasonic Sensors HC-SR04

To give the smart car the direction it currently points to, one QMC5883L Triple Axis Compass Magnetometer module is used. The changes in current in the sensor reflect the direction of the earth's magnetic field, which allow the sensor to detect the direction of the smart car. The sensor outputs an (x,y,z) coordinate, and its behavior is illustrated in the following figure. The magnetometer is particularly sensitive to electrical interferences and needs careful calibrations. Luckily, a calibration program is available from one of the community libraries.

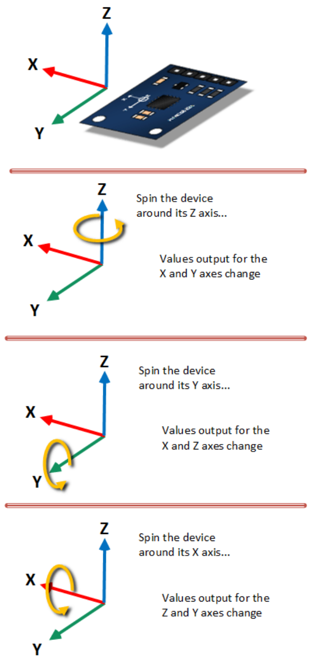

Figure 2: The behavior of the compass sensor

The wiring of the sensors is illustrated in the following figure. The SCL pin is an I2C clock pin and SDA is an I2C data pin. The DRDY (DataReady) pin is pulled up by default but goes to `LOW` (triggers an interrupt) when the output of the module is ready. Both SDA and SDA provide analog output.

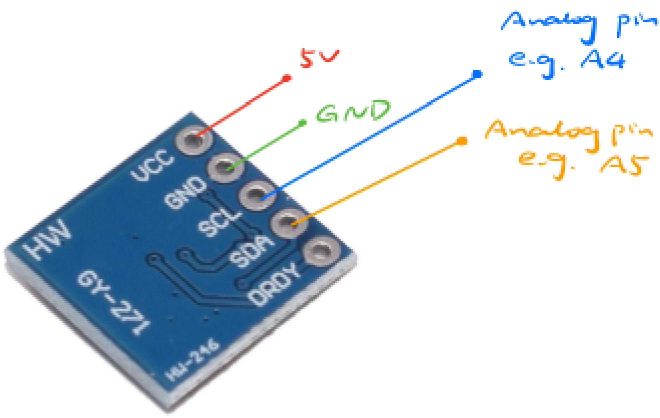

Figure 3: Pinout for compass sensor

To track the current location of the smart car and express it in terms of GPS, a GPS NEO-6M module is used. The module takes 5 V and 0 V at pin VCC and GND. The TX pin is used for serial transmission and is connected to a digital pin, such as pin 22. The RX pin is used for serial reception and is connected to a digital PWM such as pin 10. A layout is shown below.

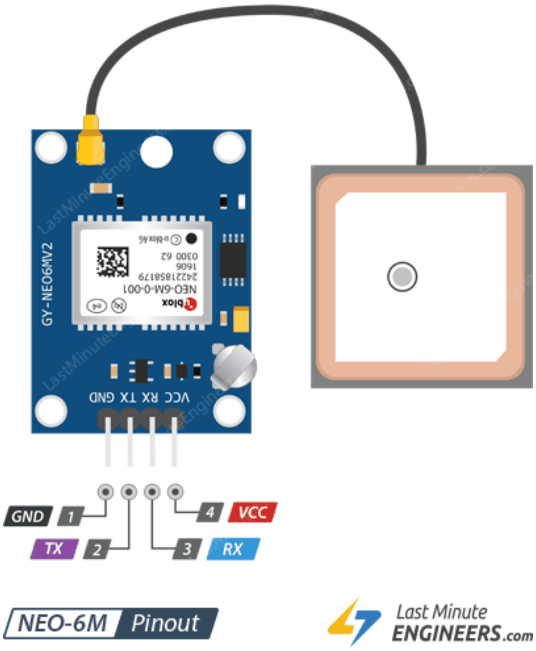

Figure 4: NEO-6M GPS module pinout

4 Mechanical Considerations

The delivery car uses a purchased chassis kit with the platforms and four separate DC motors.

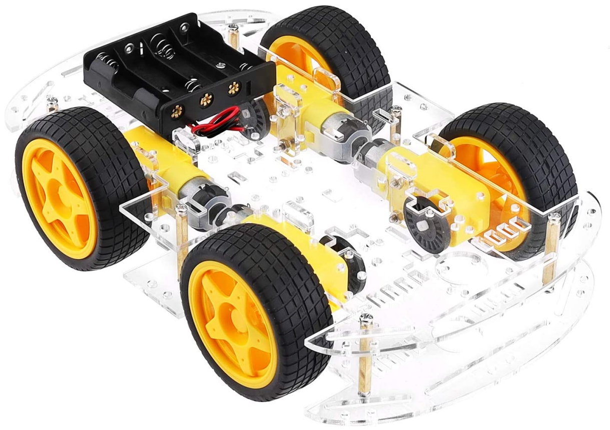

Figure 5: Purchased chassis platform

The mega board, the ESP8266 Wifi board, the GPS module, the two ultrasonic sensors, the compass sensor and the two driver modules are directly glued to the top platform, along with the 9V battery supplying the power of Arduino board. The 5.5V power supply for the motors are placed on the bottom platform.

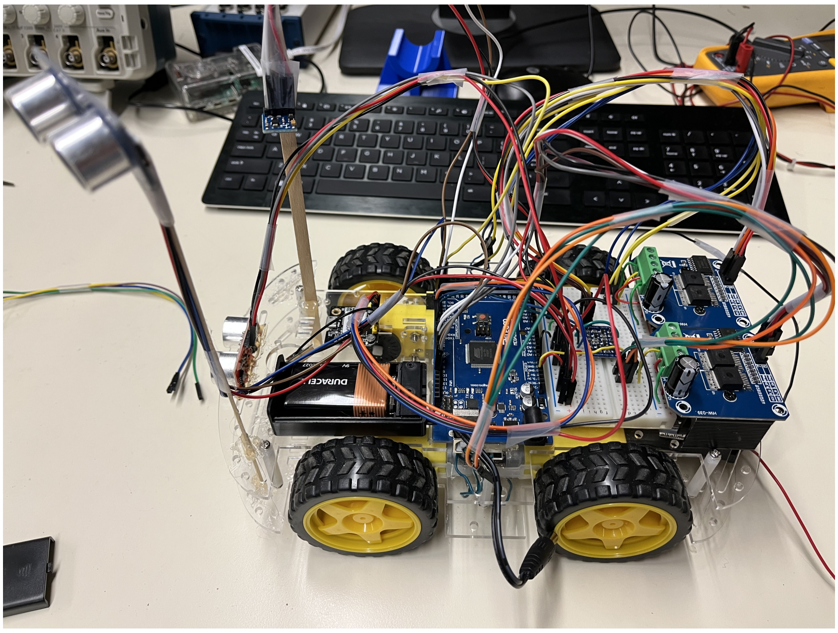

Figure 6: Completed project with all parts assembled and attached

5 Electrical Considerations

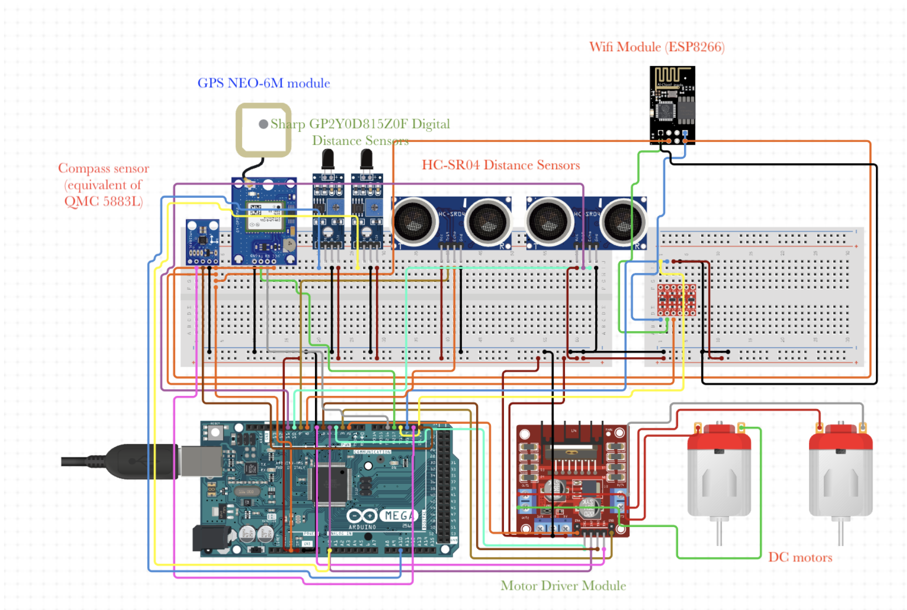

Figure 7: Basic wiring schematic for reference

A breadboard is needed for all the connections. The Mega board feeds data into the motor driver module based on the input from the compass module, distance sensor, wifi board, and GPS module. The actual pinout is illustrated below.

Driver 1
RPWM -- white -- 5
LPWM -- gray -- 6
R_EN -- orange -- 53
L_EN -- yellow -- 53

Driver 2
RPWM -- white -- 7
LPWM -- gray -- 8
R_EN -- purple -- 52
L_EN -- brown -- 52

Compass Module
SCL -- brown -- 21
SDA -- white -- 20

GPS Module (Serial 3)
RX on GPS -- orange --14 (TX3)
TX on GPS -- blue -- 15 (RX3)

Figure 8: Pin connections between Arduino Mega Board and sensors

The communication between Arduino Mega and the Wifi board is serial. Since the ESP8266 board operates on 3.3V, a level shifter is needed between the two boards.

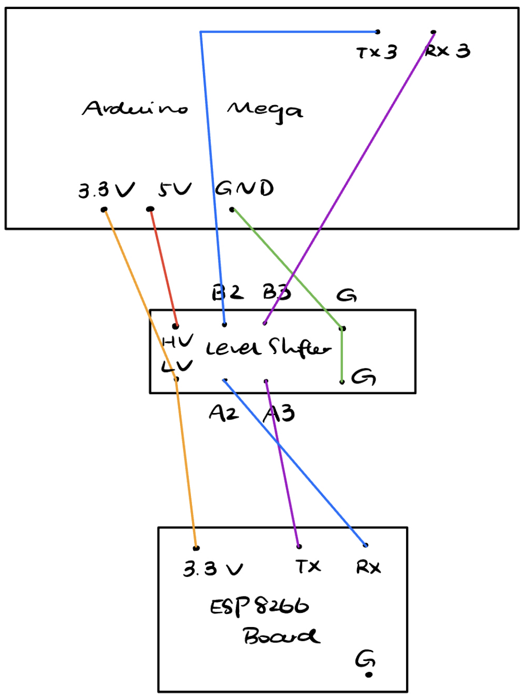

Figure 9: Serial communication between Arduino Mega and ESP8266 Wifi board with level shifter

6 Interfaces

The project needs one Arduino Mega board as the main microcontroller.

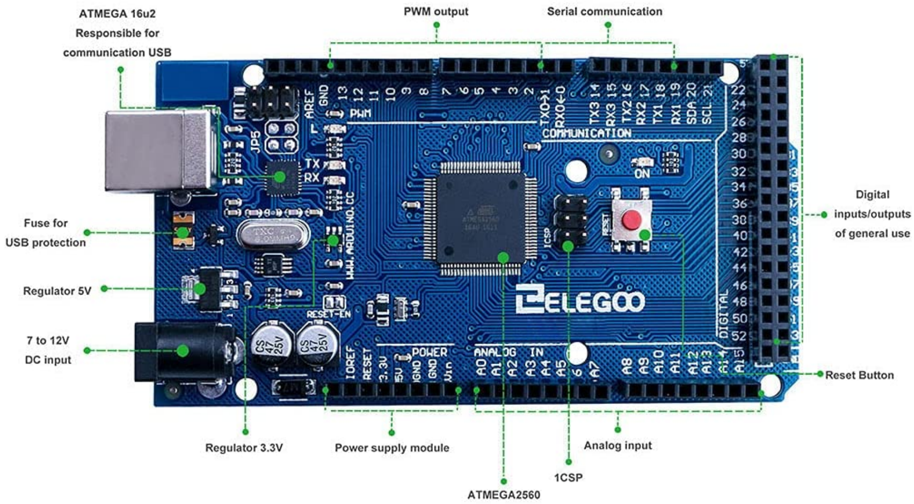

Figure 10: Arduino Mega layout

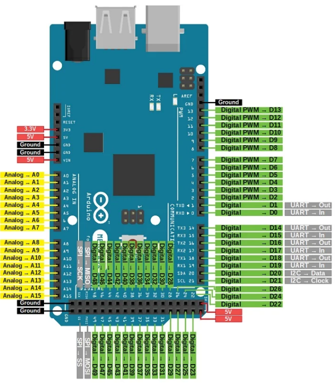

Figure 11: Mega pinout

The project also needs an ESP8266 Wifi board (Wemos D1 Mini model) to connect Arduino to Wifi and to receive user inputs. The signals between the two are translated by a level shifter, as shown in Figure 9. The wifi board lets users input directions like left, right, north, south, etc. A pinout diagram for the Wemos D1 Mini model alone is shown.

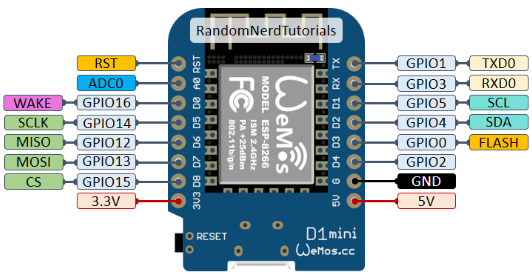

Figure 12: ESP8266 Wifi board pinout

Two BTS7960 Motor Driver modules are needed to control the motion of DC motors more precisely.

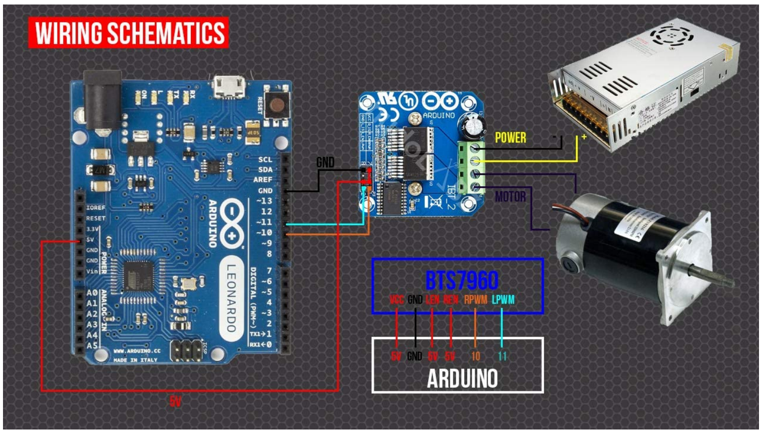

Figure 13: BTS7960 Motor Driver Module wiring schematics

A pinout diagram for BTS7960 Motor Driver is provided below. The two PWM pins (PWM-R and PWM-L) on BTS7960 are connected to PWM pins on Mega such as pins 5 and 6. The M+ and M- pins are connected to the positive and negative pins on DC motors. The B+ and B- pins will be connected to the positive and negative ends of the battery. The two output pins (EN-R and EN-L) on BTS7960 are connected to digital pins such as 53 on Mega.

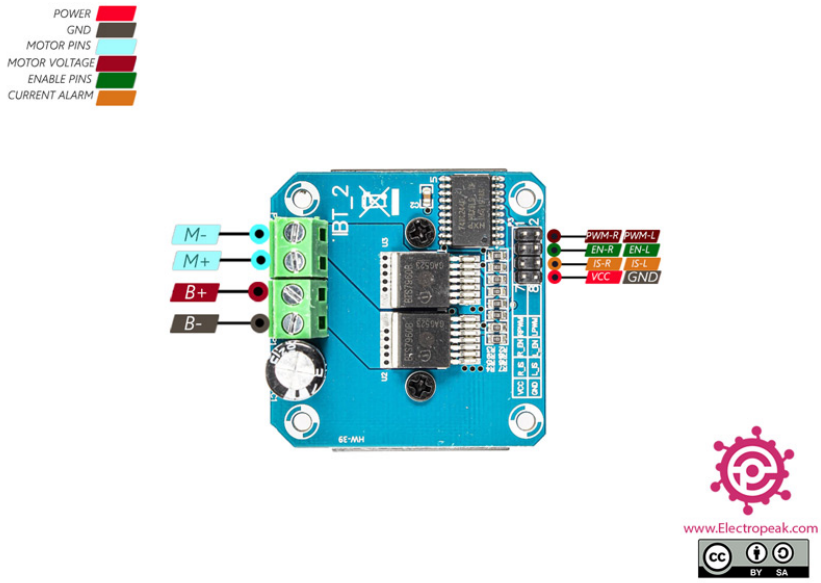

Figure 14: BTS7960 Pinout

7 Software

The final application consists of a user input program and a navigation program based on GPS. The project depends on a lot of community-developed Arduino libraries. QMC5883L compass driver library is used to calibrate the compass and obtain the heading direction of the smart car, which will be used in calculating the next movement of the smart car based on the GPS of the destination. TinyGPSPlus library and Adafruit GPS library will be used to get the current GPS of the smart car (e.g. `currentLat = gps.location.lat();` and `currentLong = gps.location.lng();`). A NewPing library is used to convert time to distance for the input from ultrasonic sensors.

User input from the APP Telegram will be read with the help of ESP8266-TelegramBot library and the core library EPS8266 Wifi (`#include <ESP8266WiFi.h>`, `#include <WiFiClientSecure.h>`). After setting up a carbot in Telegram, users are able to send instructions such as "Right" or "Left". The smart car reads user instructions from the app through the example codes below.

```
WiFiClientSecure client;
TelegramBot bot(ARDUINO_MAPS_BOT_TOKEN, client);
bot.sendMessage(CHAT_ID, "Starting Up", "");
bot.sendMessage(chat_id, "Set Destination: " + String(text), "");
```

Luckily, the driver program which computes direction and feeds into the driver module can be referenced from autonomous robot program. This program enables waypoint-to-waypoint driving with preset destinations based on GPS. The existing program is taken as the basis for this project and adjusted based on the particular driver modules and compass sensors that are used in the actual implementation.

8 Testing

The testing in this project was divided into several phases.

1. Testing the performances of separate programs/sensors.
   - QMC5883L triple-axis compass module and its library
   - the performance of NEO-6M GPS and its library
   - the connection between Wemos D1 Mini model and Arduino Mega
   - the performance of BTS7960 motor driver and its connection with the autonomous robot program
2. Testing the linked program
   - reproduce the autonomous robot program with hard-coded GPS input to combine the driver program, compass sensor, and GPS module
   - testing the user input from Telegram and its successful transmission to Arduino Mega through serial communication by Wifi Board

The performance has been tested both indoors and on relatively flat but coarse ground like the Revelle plaza. The tested destinations are Fairbanks Coffee and the bench opposite Revelle Fountain. The two locations are close but distinct enough in terms of GPS locations.

9 Safety

The project does not include any mechanical parts that require laser cutting or 3D printing. Therefore, the only safety concern is the soldering of pins for the GPS module or the compass sensor. Coated thermal gloves are recommended.

10 Parts, Reusability, and Shopping List

1. parts which definitely need purchase
   - QMC5883L Triple Axis Compass Magnetometer Sensor (includes 5 pieces) $ 12, reusable
   - GPS NEO-6M Module $ 13, reusable
   - D1 Mini WiFi Dev Board (ESP8266) (pack of 3) $ 10, reusable
   - BTS7960 Motor Driver Module $ 12, reusable
   - Smart robot car platform with DC motors $20
   - Lithium Batteries, 4AA 1.5 V
2. parts that might need restock
   - HC-SR04 Distance sensor $\times 2$, reusable
3. parts on hand
   - Sharp GP2Y0D815Z0F Digital Distance Sensors $\times 2$, reusable

The total cost of the project is estimated to be around $60.

11 De-scope/Expansion Options

The project has been significantly descoped from its original proposal, which included the combination of user input with autonomous navigation to allow interruptions during operation. The program to read in Google Map json file to obtain user input GPS location for the delivery car was also eliminated due to timing constraints.

The main function of the end product is to navigate on the plain ground from the current location to the destination GPS, with the ability to avoid obstacles and compute a relatively optimized route.

12 The deliverables: a report, a video, and a demonstration

The final project is a smart transport prototype in the form of a car that can self-drive to the destination through hard-coded GPS locations written in the program. It will be able to steer clear of all obstacles on its path. In addition, the following items are turned in.

1. Video: a demonstration of the car starting from a random point in Revelle Plaza and self-drive to the dining hall (or another nearby location)

2. Code: Autonomous driving program, User input program for Wifi board ESP 8266, a user-controlled driving program

3. Schematics: pin connections from different sensors/boards to Mega

13 Lessons Learned

The biggest challenges in this project are the insufficient power of the motors and the inability of the GPS module to receive satellite signals indoors. The car was frequently stuck on its course to its destination during the multiple tests on Revelle Plaza. The NEO-6M GPS module, being a relatively outdated model, could take about thirty minutes after its first powering to receive interpretable GPS signals. The accuracies of real-time updates as the car moves also do not meet the standard for this project, since the direction could not be computed for less than four decimals.

Other minor issues included improper soldering, ill-maintained community libraries, and a time lag between user input and signal reception.

14 Appendix

1. autonomousrobot: the program to drive the car based on waypoint GPS locations

2. userInstructions: the program to drive the car based on user input from telegram

3. sendercode: the program to upload to ESP8266 board, which connects the board to Wifi and receives user inputs for serial transmission to Mega