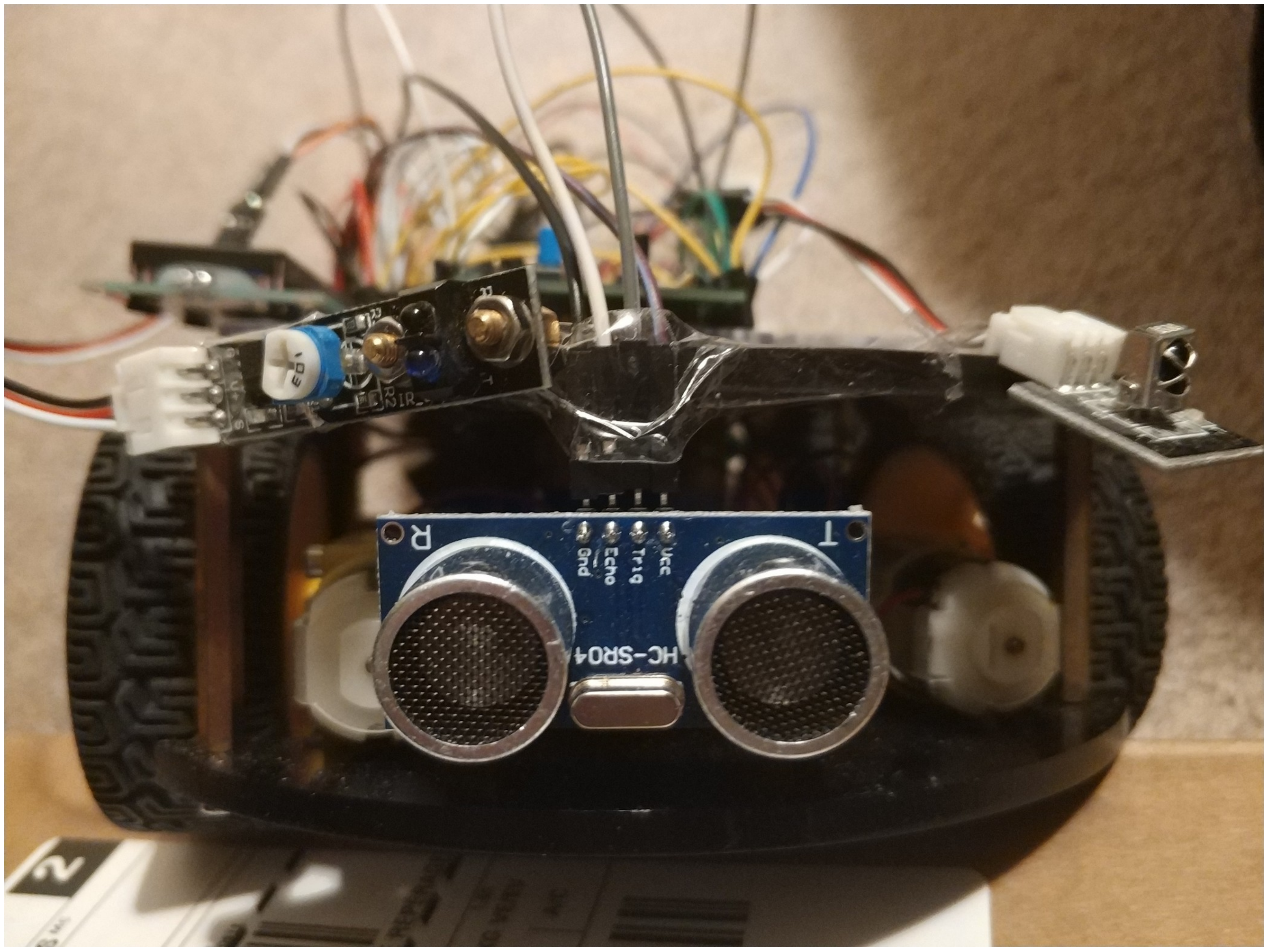
Enel 387 Project final report

Robot Car



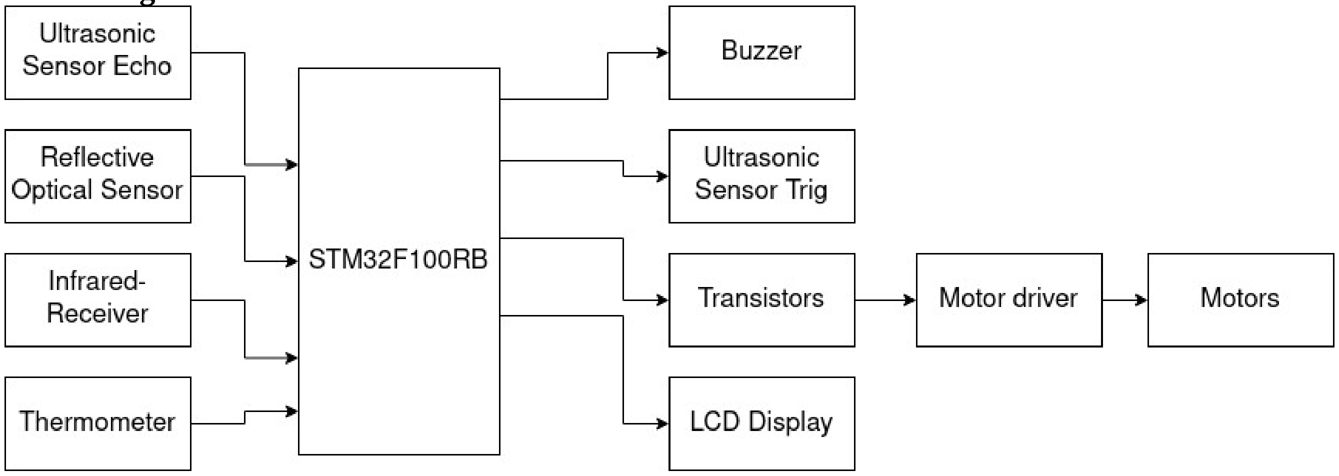
Chengyu Lou
200364972
April 8, 2020

Introduction

This project implements a robot car that is controlled by remote controller with collision prevention feature. The initial design include a path finding algorithm to let the car navigate through a maze by itself. However, due to coronavirus pandemic, there will be no maze, so I decided to modify this project to allow user to control it by a remote controller. This robot car has four motors controlled by a motor driver, a IR receiver to receive command, an ultrasonic sensor to measure distance ahead, and other features such as thermometer, LCD display and buzzer. The general idea is to allow user to control the direction to move and when it detects obstacle ahead, it will stop and wait for next instruction from user.

Hardware description

Block Diagram:



System Inputs:

- Ultrasonic Sensor Echo

The Echo pin on HC-SR04 Ultrasonic Sensor will go high when it detects reflection of ultrasonic burst, and then it will remain high for awhile, this time is corresponds to the distance ahead. Once the system receives a value smaller than a threshold value, the car will stop to prevent collision.

- Reflective optical sensor

TCRT5000 Reflective optical sensor has two components, an infrared emitter shoots light at certain angle and a photo-transistor receiving reflection of light. Black colour absorbs light and reflects none, thus this sensor will read black as zero and white as one. The initial idea was to use three of them to distinguish patterns provided by our instructor, but two of them are not working.

- Infrared Receiver

TL1838 Infrared Receiver is one of the most important module on my robot car. It receives signals coming from my remote controller, and convert light signal to digital electric signal. Thus, STM32F100RB can read numerical value by following NEC protocol. This essential module allow user to communicate with the robot. We are using it to control directions of the robot and display other useful information such as temperature, colour and distance to a LCD display.

- Thermometer

DS18B20 digital thermometer. It can measure -55°C to +125°C.

System outputs:

- Buzzer

An active buzzer, this component is used to alert user when there are obstacles ahead.

- Ultrasonic Sensor Trig

Active Trig pin on ultrasonic sensor will send ultrasonic burst.

- Transistors

Four 2N3904 transistors to control 7.4V power input. Those transistors along with PWM signals are used to active/deactivate motor driver, thus we can control the speed, direction, and state of the car.

- Motor Driver

L298N Motor Driver has a duel full bridge driver, which allows motors to rotate forward and backward. It has six input pins, four output pins (two positive two negative at any time), and a pair of power pins. Out of six input pins, four control pins controls the direction of motors, two enable/PWM pins controls the speed of the motor. However, with four timer1 PWM channels, we can control the speed and directions at same time, so those two enable pins are connected to 5V.

- Motors

Standard 4.5V DC motors with wheels.

- LCD Display

HD44780U LCD, It has two lines, 16 digits for each line. It is used mainly as debug screen. It also displays readings from sensors.

Software Description

System Functions:

- Drive forward and backward, Turn Left and Right

Set 70% duty cycle on pin PA8 and PA11 and 0% duty cycle on PA9 and PA10, the the car will move forward.
Set 70% duty cycle on pin PA9 and PA10 and 0% duty cycle on PA8 and PA11, the the car will move backward.

Set 70% duty cycle on pin PA9 and PA11 and 0% duty cycle on PA8 and PA10, the the car will turn left.
Set 70% duty cycle on pin PA8 and PA10 and 0% duty cycle on PA9 and PA11, the the car will turn right.

L298N Motor Driver truth table:

	PA8	PA9	PA10	PA11	Note
Move forward	high	low	low	high	Both pair move forward
Move backward	low	high	high	low	Both pair move backward
Turn left	low	high	low	high	Left pair move backward, right pair move forward
Turn right	high	low	high	low	Right pair move backward, Left pair move forward

- Receive Infrared signal

Start reading NEC protocol once pin PC10 go low. The data will be a 32 bit value. Compare that value with value tested before to carry out commands.

- Send and Receive ultrasonic signal

Turn Trig pin high and wait, count for time witch Echo pin return high. If this time is smaller than a threshold value, stop the car by setting all PWM channels with 0% duty cycle.

- Read Temperature

Read temperature at PA1 with ADC1, and display Hex value on LCD display.

- Read colour

Read PC13, if it returns 1, the colour is white, otherwise it is black.

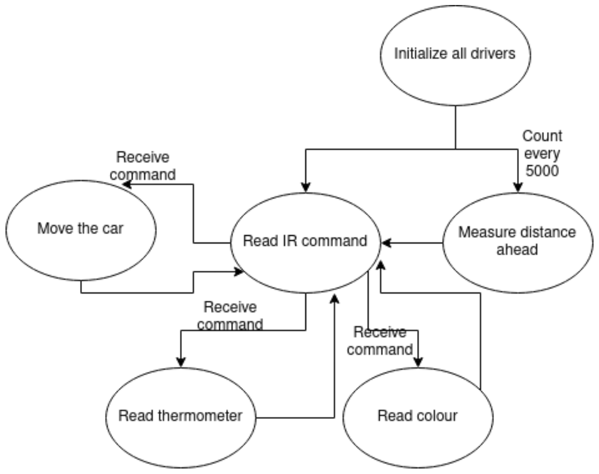
- Turn on/off Buzzer

Write one to PC9 to turn on buzzer, write zero to turn it off.

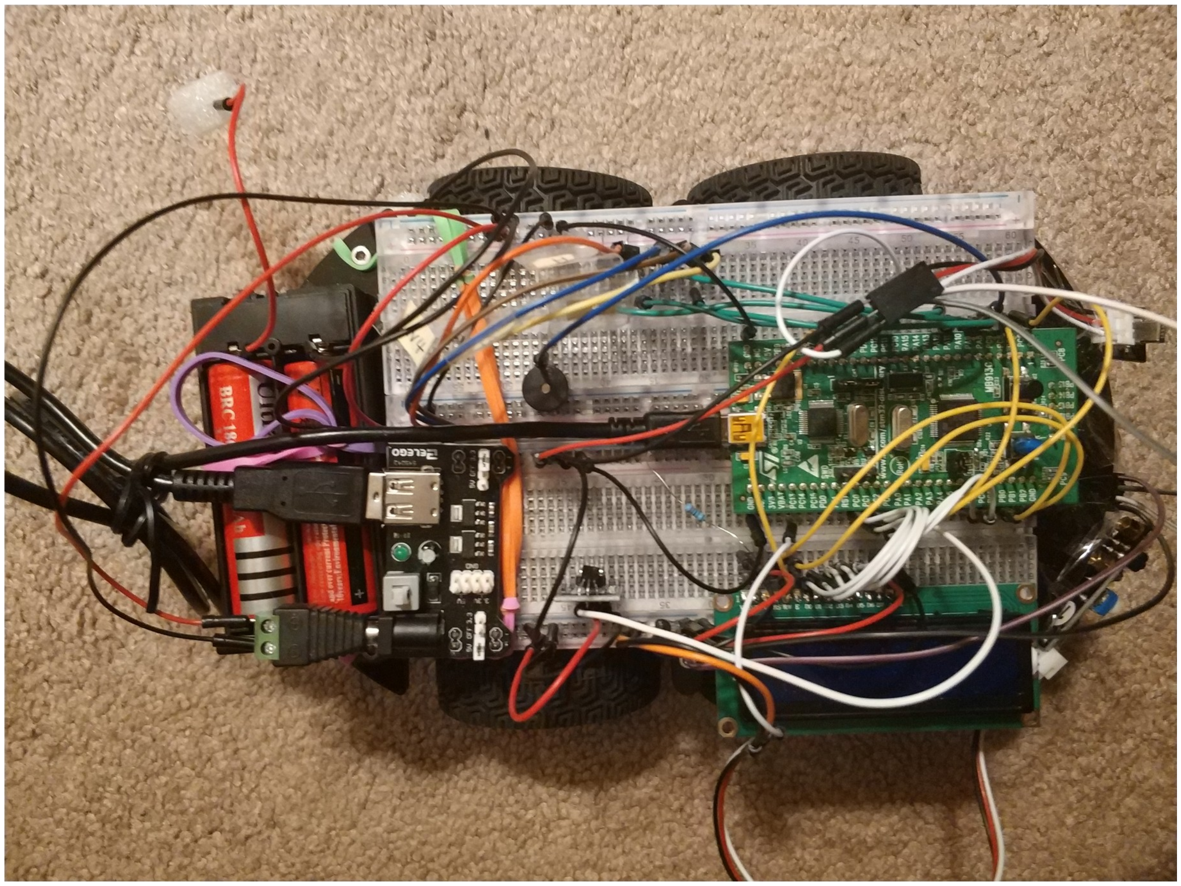
Program Listing:

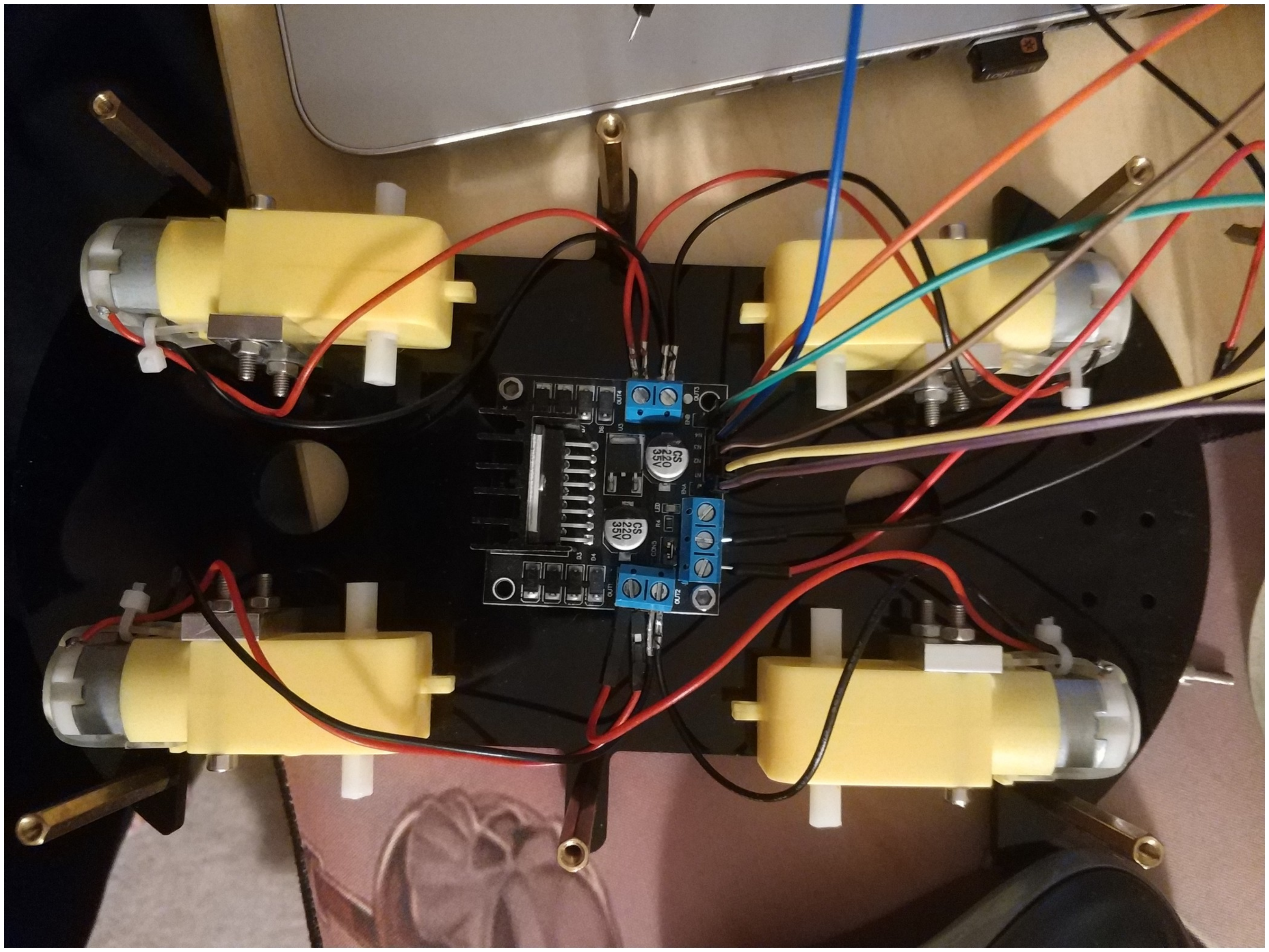
ADC1.c Clock.c GPIO.c HCSR04.c IR.c LCD.c PWM.c TCRT5000.c
ADC1.h Clock.h GPIO.h HCSR04.h IR.h LCD.h PWM.h TCRT5000.h main.c

State Diagram:



Physical Construction





Testing Documentation

The biggest fear for me during this project is when error happens, I cannot tell whether it is a software misconfiguration or hardware failure. So, I created a testing procedure to carry out. This testing procedure is very module based.

Testing procedures:

Software Testing:
I run code for each module independently first to verify my driver is working. Then, I run them all together to see if they interfering each other.

PWM signals:
I use oscilloscope to visually exam output signals come from four PWM channels (PA8, PA9, PA10. PA11). Auto setup button is really convenient. After university lock down, I no longer have access to oscilloscope, so I use the brightness of LED to verify my PWM signals.

Transistors:

I use DC generator with one channel on 3V and another on 7.4V to simulate base and collector voltage. Then, measure emitter voltage and record the number for later testing on motor driver.

L298n Motor driver:
I use DC generator channel one with voltage value I got from emitter of transistor to test out input pins, and channel two with 7.4V to power up the driver module. Then, I use multimeter to measure output voltage.

Infrared Receiver:
My initial testing plan was to use oscilloscope to find timing diagram, and then write code accordingly. However, labs are locked down due to pandemic, but I am lucky enough to find a good tutorial online and succeed on my first try.

LCD Display:
I connected LCD with same configuration as my instructor’s, and ran the same code from lab tutorial. It succeeded.

Testing logs:

2020 March 1:
Wired up LCD on breadboard. Confirmed LCD working.

2020 March 5:
Tested out buzzer working at 3.3V, motor working at 5V.

2020 March 9:
Tested out L298N motor driver logic and minimum voltage requirement. Turn out I need same level of voltage on input pins as power pins to turn on output pins.

2020 March 10:
Assembled my motor driver and motors. Labelled all input wires on the motor driver. Confirmed PWM output from PA8.

2020 March 13:
Tried to get other PWM channels to work, but did not succeed. Checked microcontroller datasheet, turns out that pin was connected to another timer. So, I revised my pin layout, start using PA8-PA11 as PWM output pins.

2020 March 14:
Studied tutorials online regarding PWM output and IR remote control.

2020 March 15:
I went to the lab to finish off all PWM code, but PA8 still the only pin has output. I check numerous online tutorial and my lab code, but could not figure out what is going on. Finally, I followed my software testing procedure, initial one pin at a time, and it worked.

Mr.Wagner walks in and told me he is going to take all multimeters since electronic labs may lock down for the rest of semester, but he leave one for me today. So, I use that multimeter to test out my transistors. I have two transistor models, 2N2222 and 2N3904. They both work for my design, but in order to keep consistency of my project I decided to use four 2N3904 transistors.

2020 March 25:
I have to install windows on my laptop in order to use Keil IDE. However Keil remain unusable, after email exchange with lab instructor Mr.Duguid, I followed instruction he posted earlier, it turns out new version of Keil cannot run project initialized in old version of Keil.

2020 March 26:
Wired up IR receiver and modified IR receiver code from online tutorial. The online tutorial uses hal library witch I failed to import into Keil project. Therefore, I modified that code so it can run on my project. It worked.

2020 March 27:
I fixed a bug on LCD display function so I can display Hex number on it. I pushed many buttons on my IR remote controller, and values for each keys appear on my LCD. I recorded them for future use.

2020 March 28:
Finished coding car movement functions.

2020 April 1:
Tried to use interrupt to receive IR signals, but failed.

2020 April 2:
I got ultrasonic sensor working.

2020 April 3:
Wired up thermometer and buzzer. Fixed a bug on ADC1 driver that initialize eight pins all together. This might be the reason why my interrupt does not work, but I ran out of time.

2020 April 4:
Tested three Reflective optical sensors, two of them are not working. I decided to discard this feature that distinguish patterns provided by instructor since there is no physical patterns anyway. However, I keep one those sensor on my car to show the my ability to implement that feature.

I shoot the demo video at university.

2020 April 5:
I modified my code to better arrange timing to run IRReceive and UltrasonicReadDistance functions. Better solution would be to use an operating system to manage that.
I shoot another short video to show my car can read distance while moving and me controlling the car using IR remote controller.

Conclusion & Future Work

Function Checklist:

requirement	Function	Complete
Dynamic input/analog sensor	Thermometer	1
Dynamic input	IR receiver	1
Dynamic input	Ultrasonic Echo	1
Controlled output/PWM/7.4V	Motors	1
Controlled output	LCD	1
Dynamic input/digital sensor	Infrared reflection sensor	1
Controlled output/GPIO	Ultrasonic Trig	1
Controlled output/GPIO	Buzzer	1
Controlled output/PWM	Step Motor	0

Feature Checklist:

Feature	Complete
Receive IR command	1
Robot car move all directions	1
Read thermometer	1
Display to LCD	1
Read colour	1
Measure distance	1
Emergency stop if distance ahead too close	1
Sound buzzer	1
Use interrupt to receive IR commands	0
Rotate Step motor	0
Navigate out of maze	0

I successfully completed most of features I designed. The robot can go all directions following commands from IR remote controller. It will stop if distance to obstacle ahead is smaller than a preset threshold value. However, there are deflects can be fixed via software patches. The timing to receive commands and measure distance ahead is not running parallel, so it would not receive command while measuring distance, vice versa. This flaw can be solved by using an operating system or using interrupt to receive command. My future work will be focusing on those solutions.

Reference:

https://controllerstech.com/hc-sr04-ultrasonic-sensor-and-stm32/
https://controllerstech.com/ir-remote-with-stm32/

https://lastminuteengineers.com/l298n-dc-stepper-driver-arduino-tutorial/