Car Balancing System

3CT3 – Control Theory I

Elliot Wong (1302982)

Dylan DePodesta (1327812)

Abstract

The purpose of this project is to create a model car that can automatically find the middle between 2 objects (one in the front and one in the back). The car will then move a certain distance such that the distance in front of it and the distance behind it will be the same. The car also uses a visual aid in determining the mid point between 2 objects. The LED dot matrix uses the left side of the dot matrix to say the car is too far left and the right side of the dot matrix to say the car is too far right, when the 2 light columns in the middle light up, the car is now in the middle.

Table of Contents

Introduction………………………………………………………………………………………………………….…………………....Page 1

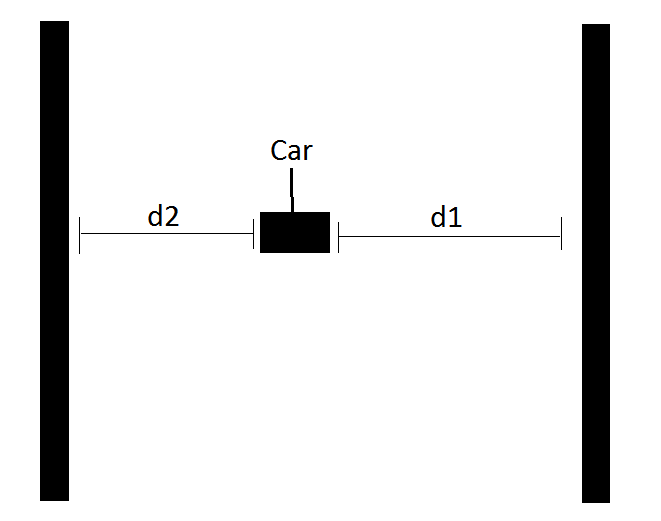
Design, Development and Implementation……………………………………………………………………………….. Page 2

Discussion and Conclusion……………………………………………………………………………………………………….… Page 11

References……………………………………………………………………………………………………….………………………… Page 12

Introduction

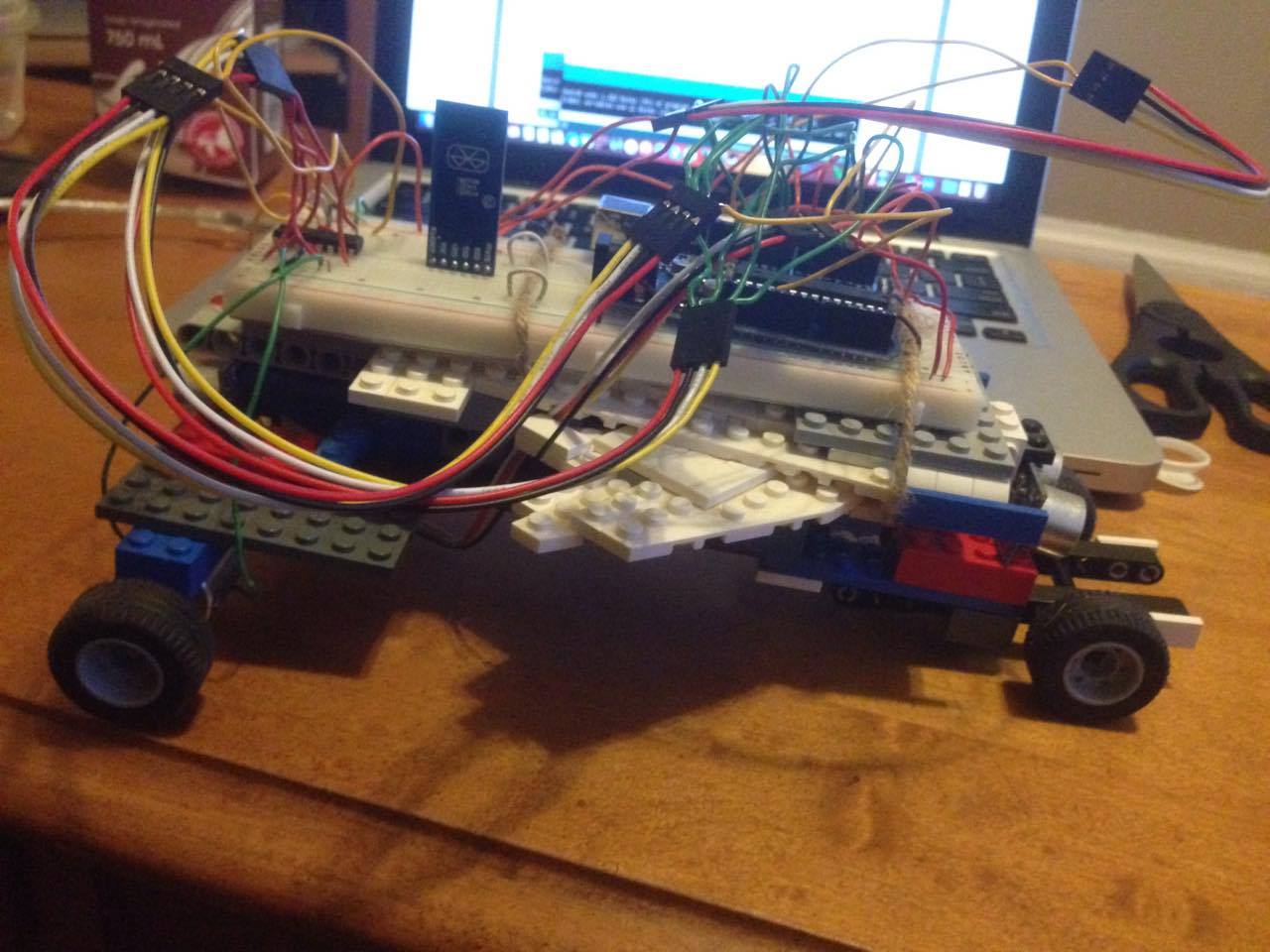
This project aims to using ultra sonic sensors and DC motors to control a model car such that the car will find the middle point between two walls. There is also an 8x8 LED Matrix that indicates the positon of the car relative to the two walls.



As can be seen in the diagram above, d1 and d2 should be equal after the car has “balanced” itself.

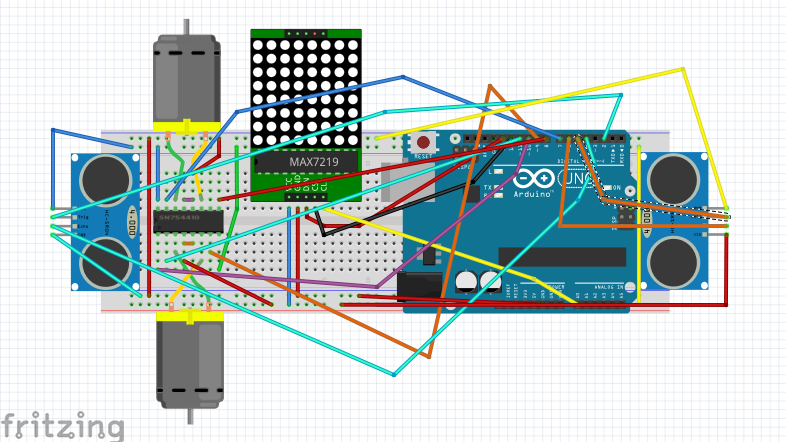
Design, Development and Implementation

Our initial design included the use of a Lego car. (As seen in the image below)

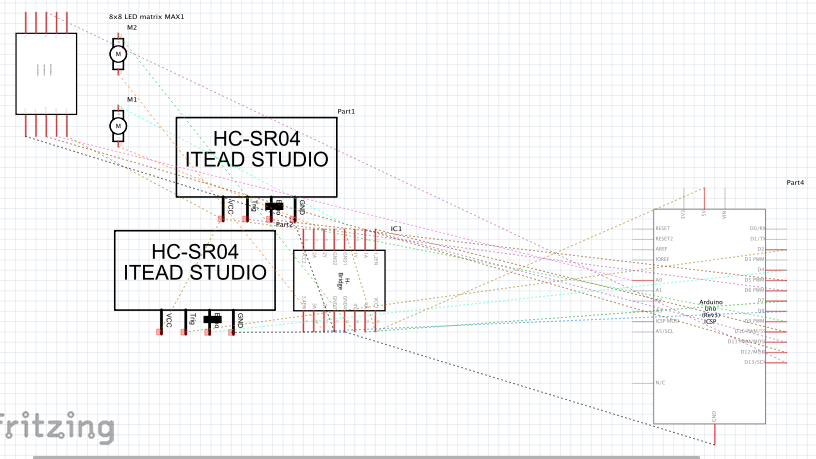


But our DC motors were not strong enough to move the vehicle. And so, the chassis of the car was removed and the body of our car was only the breadboard. The car itself consists of two ultrasonic sensors, two DC motors, an Arduino Uno, LED Matrix board and the appropriate driver chips to operate these sensors/actuators.

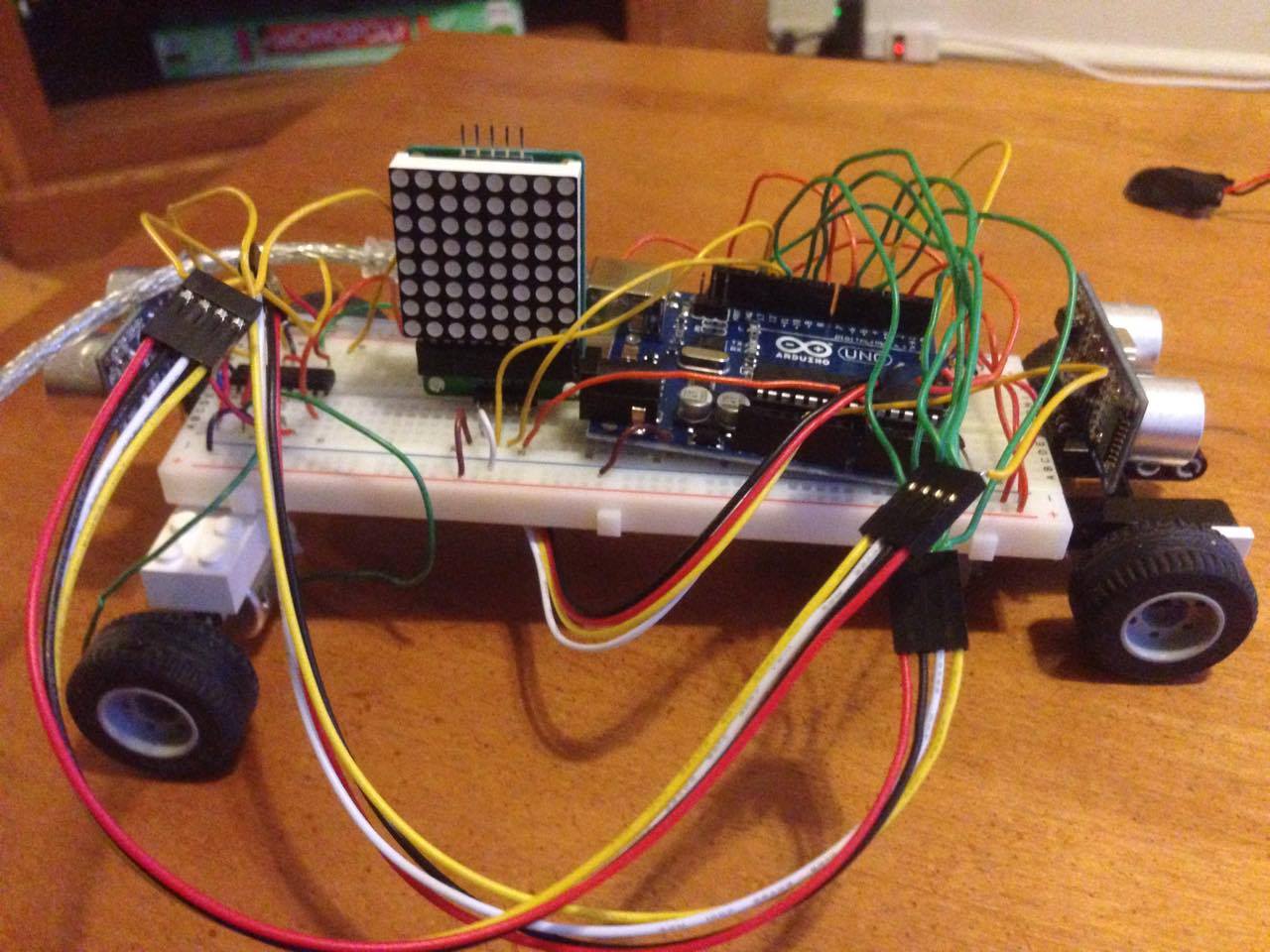
Fritzing Diagram:



Fritzing Schematic:



New Model Car:



Code:

#include <LedControl.h> // Library for Dot Matrix

#include <NewPing.h> // Library for Ultrasonic Sensor

#include <SoftwareSerial.h> // Library for DC motor driver

#define TRIGGER\_PIN1 8 // Front Ultrasonic Sensor

#define ECHO\_PIN1 7

#define MAX\_DISTANCE1 400

#define TRIGGER\_PIN2 2 // Back Ultrasonic Sensor

#define ECHO\_PIN2 4

#define MAX\_DISTANCE2 400

int Reverse1 = 10; // Wheel 1

int Speed1 = 3;

int Forward1 = 11;

int Forward2 = 5; // Wheel 2

int Reverse2 = 6;

int Speed2 = 9;

NewPing sonar1(TRIGGER\_PIN1, ECHO\_PIN1, MAX\_DISTANCE1); // Activation of forward ultrasonic sensor

NewPing sonar2(TRIGGER\_PIN2, ECHO\_PIN2, MAX\_DISTANCE2); // Activation of backward ultrasonic sensor

LedControl lc = LedControl(13,14,12,1); // Activation of dot matrix

unsigned long delayTime = 200;

byte row1[] = // Dot matrix left most row is on

{

B11111111,

B00000000,

B00000000,

B00000000,

B00000000,

B00000000,

B00000000,

B00000000

};

byte row2[] = // Dot matrix second left most row is on

{

B00000000,

B11111111,

B00000000,

B00000000,

B00000000,

B00000000,

B00000000,

B00000000

};

byte row3[] = // Dot matrix third left most row is on

{

B00000000,

B00000000,

B11111111,

B00000000,

B00000000,

B00000000,

B00000000,

B00000000

};

byte rowMiddle[] = // Dot matrix middle 2 rows are on

{

B00000000,

B00000000,

B00000000,

B11111111,

B11111111,

B00000000,

B00000000,

B00000000

};

byte row6[] = // Dot matrix third right most row is on

{

B00000000,

B00000000,

B00000000,

B00000000,

B00000000,

B11111111,

B00000000,

B00000000

};

byte row7[] = // Dot matrix second right most row is on

{

B00000000,

B00000000,

B00000000,

B00000000,

B00000000,

B00000000,

B11111111,

B00000000

};

byte row8[] = // Dot matrix right most row is on

{

B00000000,

B00000000,

B00000000,

B00000000,

B00000000,

B00000000,

B00000000,

B11111111

};

void setup() {

pinMode(Speed1, OUTPUT); // Setting up pinmodes for motor 1

pinMode(Forward1, OUTPUT);

pinMode(Reverse1, OUTPUT);

pinMode(Speed2, OUTPUT); // Setting up pinmodes for motor 2

pinMode(Forward2, OUTPUT);

pinMode(Reverse2, OUTPUT);

Serial.begin(115200); // Channel to use on serial monitor

lc.shutdown(0,false); // Wake up displays

lc.shutdown(1,false);

lc.setIntensity(0,5); // Set intensity levels

lc.setIntensity(1,5);

lc.clearDisplay(0); // Clear Displays

lc.clearDisplay(1);

}

void loop() {

delay(600);

Serial.print("Sensor1 (FRONT): "); // Prints ultrasonic sensor values in serial monitor

Serial.print(sonar1.ping\_cm());

Serial.print("cm ");

Serial.print("Sensor2 (REAR): ");

Serial.print(sonar2.ping\_cm());

Serial.println("cm");

if (sonar1.ping\_cm() > sonar2.ping\_cm()){ // When sensor 1 is greater than sensor 2

analogWrite(Speed1, 250); // the motor will go forward

analogWrite(Speed2, 250);

digitalWrite(Reverse1, LOW);

digitalWrite(Reverse2, LOW);

digitalWrite(Forward1, HIGH);

digitalWrite(Forward2, HIGH);

}

else if (sonar2.ping\_cm() > sonar1.ping\_cm()){ // When sensor 2 is greater than sensor 1

analogWrite(Speed1, 250); // the motor will go backward

analogWrite(Speed2, 250);

digitalWrite(Forward1, LOW);

digitalWrite(Forward2, LOW);

digitalWrite(Reverse1, HIGH);

digitalWrite(Reverse2, HIGH);

}

while (sonar1.ping\_cm() == sonar2.ping\_cm()){ // When the sensors are equal, the motors will stop

analogWrite(Speed1, 0);

analogWrite(Speed2, 0);

digitalWrite(Reverse1, LOW);

digitalWrite(Reverse2, LOW);

digitalWrite(Forward1, LOW);

digitalWrite(Forward2, LOW);

delay(2000);

}

// These if statements turn on the varies rows of light on the dot matrix

if (sonar1.ping\_cm()/sonar2.ping\_cm() <= (1 / sonar2.ping\_cm())) {

for (int i = 0; i < 8; i++){

lc.setRow(0,i,row8[i]);

}

}

if (sonar1.ping\_cm()/sonar2.ping\_cm() > (1 / sonar2.ping\_cm()) && (sonar1.ping\_cm()/sonar2.ping\_cm() < (2 / sonar2.ping\_cm()))){

for (int i = 0; i < 8; i++){

lc.setRow(0,i,row7[i]);

}

}

if (sonar1.ping\_cm()/sonar2.ping\_cm() > (2 / sonar2.ping\_cm()) && (sonar1.ping\_cm()/sonar2.ping\_cm() < (3 / sonar2.ping\_cm()))){

for (int i = 0; i < 8; i++){

lc.setRow(0,i,row6[i]);

}

}

if (sonar1.ping\_cm()/sonar2.ping\_cm() == 1){

for (int i = 0; i < 8; i++){

lc.setRow(0,i,rowMiddle[i]);

}

}

if (sonar1.ping\_cm()/sonar2.ping\_cm() > (2 / sonar1.ping\_cm()) && (sonar1.ping\_cm()/sonar2.ping\_cm() < (3 / sonar1.ping\_cm()))){

for (int i = 0; i < 8; i++){

lc.setRow(0,i,row3[i]);

}

}

if (sonar1.ping\_cm()/sonar2.ping\_cm() > (1 / sonar1.ping\_cm()) && (sonar1.ping\_cm()/sonar2.ping\_cm() < (1 / sonar1.ping\_cm()))){

for (int i = 0; i < 8; i++){

lc.setRow(0,i,row2[i]);

}

}

if (sonar1.ping\_cm()/sonar2.ping\_cm() <= (1 / sonar1.ping\_cm())){

for (int i = 0; i < 8; i++){

lc.setRow(0,i,row1[i]);

}

}

}

Discussion and Conclusion

This project is an example of a practical application of the ultrasonic sensor because of the fact that we are using it to detect objects in its path and correct itself to the mid point between the objects. Some challenges we faced when doing this project is the fact that our ultrasonic sensors refuse to output the correct values, but other than that our code for the project was working. Another challenge we faced was that our first model was too heavy so we created a lighter more efficient model, one that the 2 motors we were using could handle. One last problem we faced was getting the Bluetooth sensor to work with the phone, so instead we changed that idea to a LED dot matrix instead.

References

<https://brainy-bits.com/tutorials/how-to-control-max7219-led-matrix/>

<http://www.instructables.com/id/Ultrasonic-Range-detector-using-Arduino-and-the-SR/>