Sugnaykumar Patel

Brandon Watson

CS 207

4 April 2025

Introduction and Inspiration

Currently, this world is witnessing a paradigm shift in transportation industries. Autonomous driving cars are being treated like basic features in this competitive world. Not only does it provide convenience but it also increases efficiency and enhanced safety. With advancements in AI, sensor technology, and vehicle-to-everything communication, self-driving cars are steadily moving from concept to reality. Witnessing such a transforming world made me curious to create my own self-driving autonomous car with several features like autosteer, automatic speed adjustment, smart turning etc.This report delves into the development of an innovative autonomous driving car, inspired by the cutting-edge designs and technologies pioneered by leading automakers such as Tesla and emerging Chinese automotive brands. Tesla has redefined the automotive industry with its seamless integration of self-driving capabilities with the software and hardware integration.

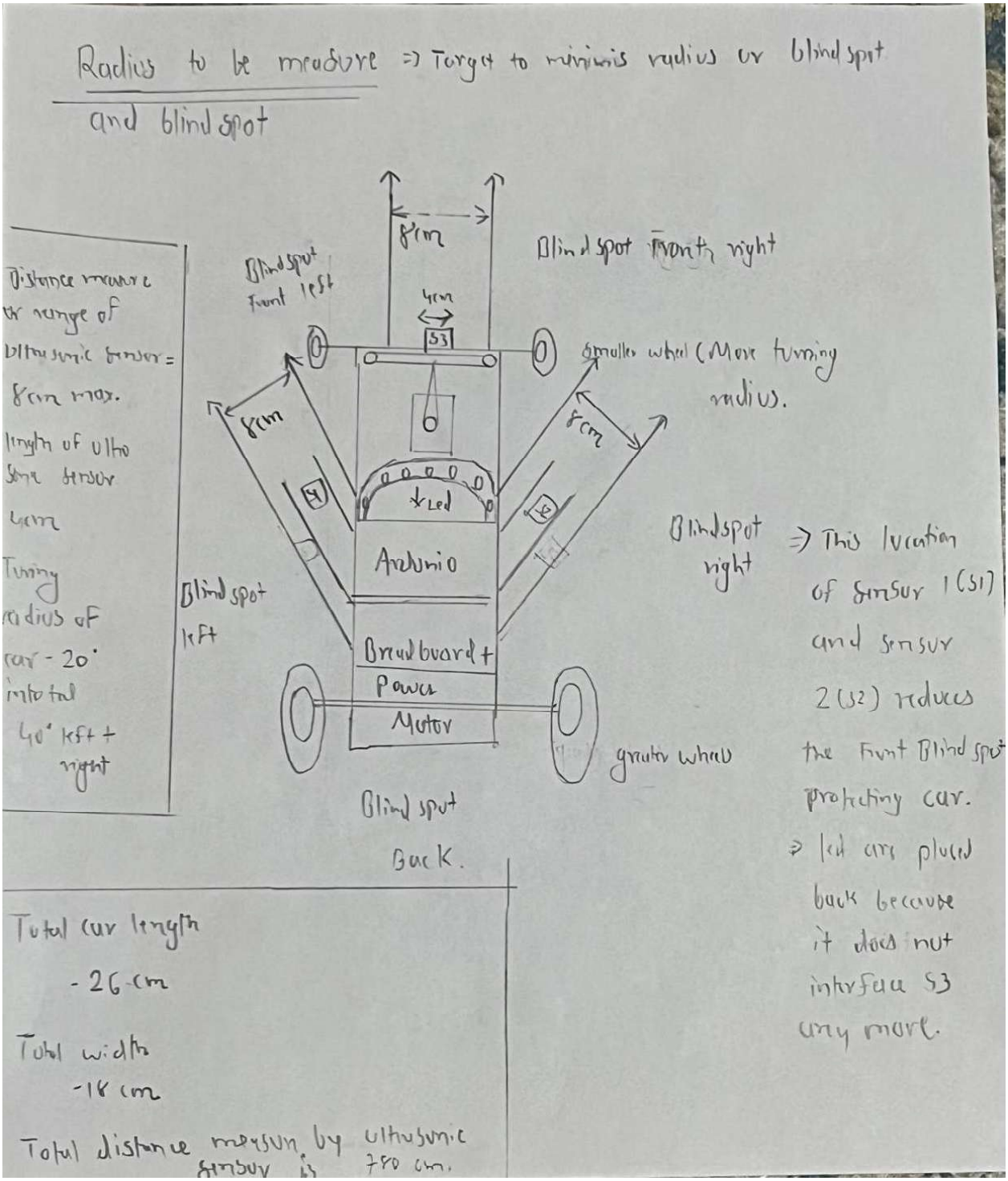
When I discovered that Tesla cars use ultrasonic sensors and digital cameras for detecting objects in their vehicles, I decided to self implement the concept for learning and fun as my final project. Similarly, Chinese brands such as Xpeng and NIO have gathered global attention for rapid advancement in autonomous driving by prioritizing user experience and embedded some mind boggling feather like in-car Facial Recognition, Omnidirectional wheels and Autonomous Valet Parking.

Design and Development

My first thought before designing my autonomous car was, it's gonna be a piece of cake for me. Turns out that's the wrong thought. I want my car to be like most of the cars on the road that is realistic rather than robotic. So rather than using 1 ultrasonic sensor and servo motor to measure the distance of the object around the car. I decided to use 3 ultrasonic sensors on my prototype that cover the most crucial distances which are most likely to be measured for preventing collision. Therefore, my prototype had 3 ultrasonic sensors which are angled at 45 degree which reduces the blindspot around the car and they can detect the objects smoothly without interruption of any other car parts. Having too many electronic devices to be fixed in a certain compact size was nothing but a hurdle. To fix that problem, I had created my car in 2 different parts. Front part includes the steering mechanism with servo motor handling it and the back part of my car includes the Transmission and my microcontroller. These both are connected through chassis which help me to create the compact design for my car. Creating both separately also helps me to test all motors and sensors manually. Smaller front wheel in my car also had a specific reason. The first reason is that a smaller wheel helps in increasing the turning radius. Second reason for using a tiny wheel is that it does not interrupt the ultrasonic sensor while turning.

Deploying and developing this autonomous car includes a total of 4 steps. The first step was to implement the circuit and functioning of the autonomous car in a virtual environment with tinkerCad. Creating the prototype of the car that holds up the components and sustains wiring. I have to break down and sketch a prototype multiple times due to multiple problems but the final sketch is provided in the picture below. Training the autonomous car model for real life

situations based on software and inputs accumulated by sensors. Testing and debugging the decision and improving accuracy of the device by manipulating ambience.



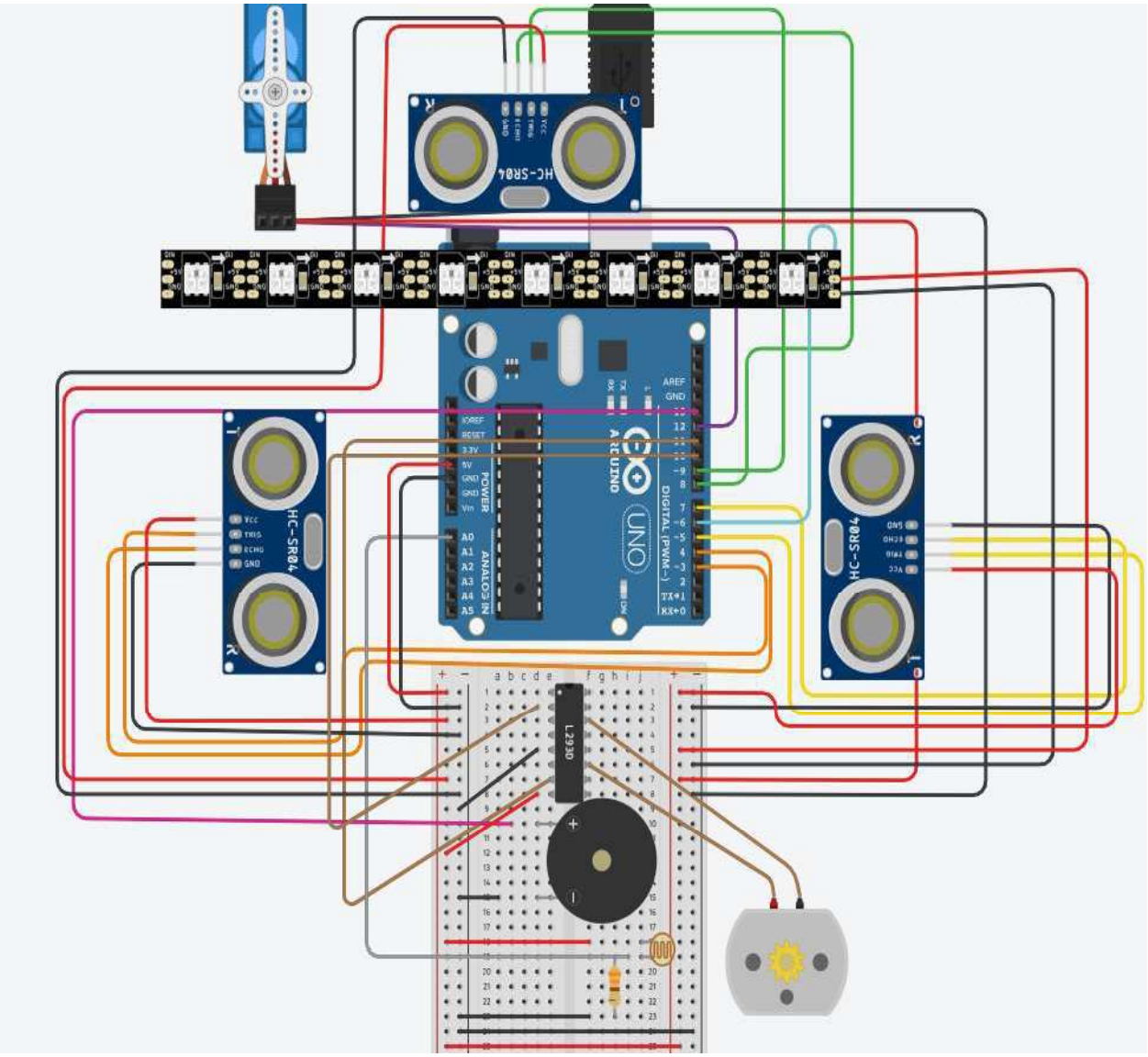
,

Technical Details

Making autonomous cars requires the knowledge of mechanical stuff, wiring and concepts of physics. Combining those concepts and making flawless cars need robust sensors and heavy built prototypes. All of the components are listed below :

- 3 Ultrasonic Module
- Power Module for breadboard
- TA6586 motor driver
- Arduino UNO R4 Wifi
- Speaker
- RGB Strip
- Photoresistor + Resistor (330 ohm)
- Jumper Wires
- DC motors and Servo Motor

Three ultrasonic sensors measure distance of the object and based on the input of the sensor the servo motor moves in certain angles. Thus this makes the car interactive with the objects around it. The rgb is also interactive along with the moving direction of the car and state of the car. Speaker is used as the safety feature and it also indicates the state of the car. The DC motor is used, it acts like the engine of the car. TA6586 motor drivers control the speed and direction of the motor. Photoresistor measures the brightness of the surrounding and sets up the brightness of the rgb strip relatively. All components are connected with the jumper wire and arduino and the power is provided by the external power module which is 5V. The complete wiring is provided below.



Beside all these components, the logic of the car had been written C++ language inside arduino IDE. All the components on this device are interactive and coordinate each other. The brain of the autonomous car hides inside the arduino code. Based on the inputs and code results the car functions and operates with overall 80% of accuracy. Library use for software implementation are inside the code down below.

Functionality and Features

1. Setting auto-brightness of headlight and car light. One of the important features of the car is it measures the light present around the car and sets up the brightness of the car leds. It helps save energy and make it more efficient.

```
int dayOrnight = analogRead(ANALOG_PIN);
int brightness = map(dayOrnight,0,404,10,200);
int headlight = 200 - brightness;
strip.setBrightness(headlight);
```

2. Second most important feature of the car is based on the field of the safety of the user. If the car is blocked from front gate, left and right gate then the car does not move and the led turn red with the constant beeping sound of speaker with is the sign of warning of the car user.

```
if(distance3 <=20 && distance1 <=12 && distance2 <=12)
```

```
else if(distance3 <=15 )
```

```
for (int i = 0; i < NUM_LEDS ; i++)
  {
   strip.setPixelColor(i, strip.Color(255,0,0));
  }
tone(SPEAKER_PIN , 1500, 1000);
delay(2000);
```

3. If the Car is moving in the straight path and there is another car in front of our user's car then it would decrease the speed of the car to avoid collision.Then it would ultimately stop the car before collision.

```
else if(distance3 <= 40)
```

4. It also makes smart decisions when it is blocked on the front gate. It measures the distance of the left sensor and right sensor and navigates where there is more distance.If both of the distances are the same it randomly selects the left or right path which is based on a random algorithm. This is what I called smart turning decisions.

```
srand(time(0));
int randomNumber = rand() % 2;
```

5. It also had an feature of auto steering left and right based on the distance of objects which avoid the collision and make it autonomous.
6. Last but not least , the speed while turning , taking decisions and moving close to the object at the front, moving straight varies according to the object. Sometimes when the car is at a dead stop it requires the minimum torque to operate that is why I had created a starter function which only works when the speed is 0.

```
void starter1()
{
  Serial.print("Starter 1");
  int accerlation = 55;
  analogWrite(FOR_PIN,80);
  analogWrite(BACK_PIN,0);
  delay(200);
  analogWrite(FOR_PIN,53);
  analogWrite(BACK_PIN,0);
  delay(50);
  for(int i=0;i<10;i++)
   {
    accerlation = accerlation - i;
    analogWrite(FOR_PIN,accerlation);
    analogWrite(BACK_PIN,0);
    delay(5);
   }
}
```

```
void starter2()
{
  Serial.print("Starter 2");
  int accerlation = 60;
  analogWrite(FOR_PIN,110);
  analogWrite(BACK_PIN,0);
  delay(250);
  analogWrite(FOR_PIN,80);
  analogWrite(BACK_PIN,0);
  delay(400);
  for(int i=0;i<10;i++)
   {
    accerlation = accerlation - i;
    analogWrite(FOR_PIN,accerlation);
    analogWrite(BACK_PIN,0);
    delay(50);
   }
}
```

7. To increase the visibility of the user while driving the car. The led turn on and off relatively with the direction of the steering of the car. It improves visibility and makes roads safer.

The logic of the code does not stop here it is in very depth. One can learn the code and figure it out by comments. The important features are displayed above.

Challenges and solutions

Initially, the first and foremost challenge I faced was creating minimal design for my car by adjusting all components of the car inside the created prototype and wiring all of them. It requires a lot of patience and hardwork. While I was creating my prototype, I decided to create my own front steering mechanism but somehow I failed because I was not getting enough turning radius. To reconcile this problem, I tore apart a toy car and attached the steering mechanism to my car. However, I need more turning radius, So I decided to make the front wheel smaller, resulting in more turning radius. As soon as I thought the problem was solved I got into major trouble. I was using a DC motor of 3.3V to 12V which requires a minimum of 4500 rpm for rotations. This speed was so much for my lightweight device. The max speed was

60 cm/sec and the least I could get was 30 cm/sec. It was driving my car crazy as I had no manual speed control or geared motor. To solve this problem, I had to use software coding techniques for increasing and reducing speed. It was way challenging because it required multiple trials and different landscapes required different speeds. After numerous trials, I discovered the optimal speed for a car.

Future Improvements

- Currently, I feature which are lacking in this model and autonomous cars out there are adapting the speed based on the road conditions. I could use a friction measurement sensor for measuring the friction in between the sensor and tyre which could make it possible.
- More turning radius mechanism could make this car more effective and stable while sharp turning.
- Increasing the number of ultrasonic sensors can make the car and its decision more precise. With that I can also detect the dip and ditch of the road to make me free from free fall. Moreover, I could also add a lane assisting feature with the help of an infrared sensor (With combination of transistor and receiver).
- After all this improving code and algorithm to make the car reverse, parallel parking can be implemented after all above hardware updates. This code is gonna be more complex but fun to play with.