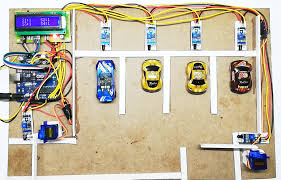
Automated Car parking system

**(A Basic Idea)**



-(The LED display is not used)

**Basic Idea: -**

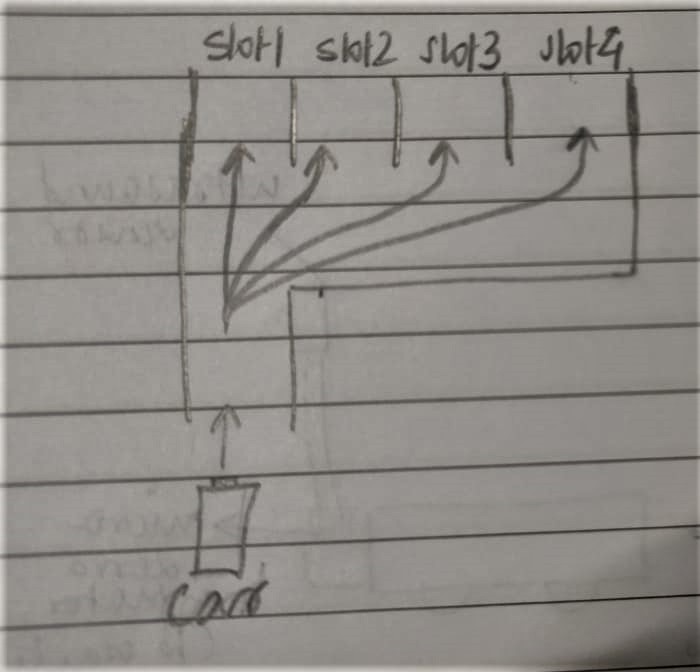
This Car-Parking system will detect an empty slot in the parking area. In addition to that the incoming car will be connected to the parking system, thus the parking system will help the car get parked in that empty slot. This will help save time and confusion in crowded places. This is just a basic system and won’t help the car detect any obstacles on its way.

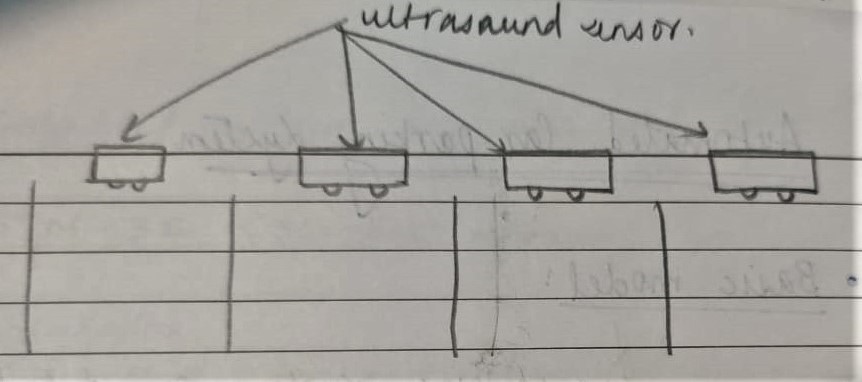
**Sensors/equipments used: -**

1. Arduino UNO
2. 2 Motor drivers
3. 4 motors
4. 1 servo motor
5. 5 ultrasound sensors
6. Jumper wires
7. A car chassis

**Basic Model and setup: -**

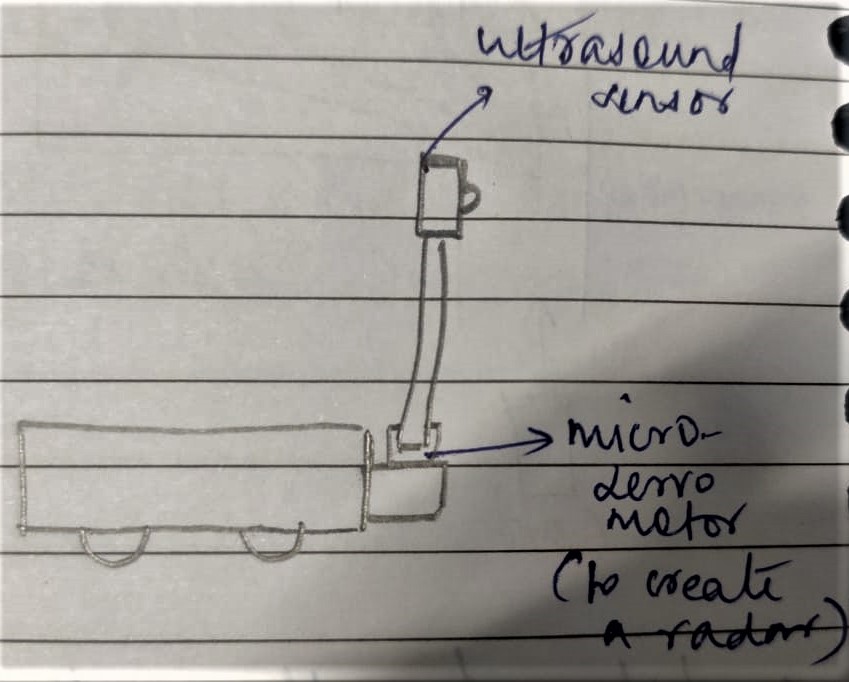
1. Parking system:

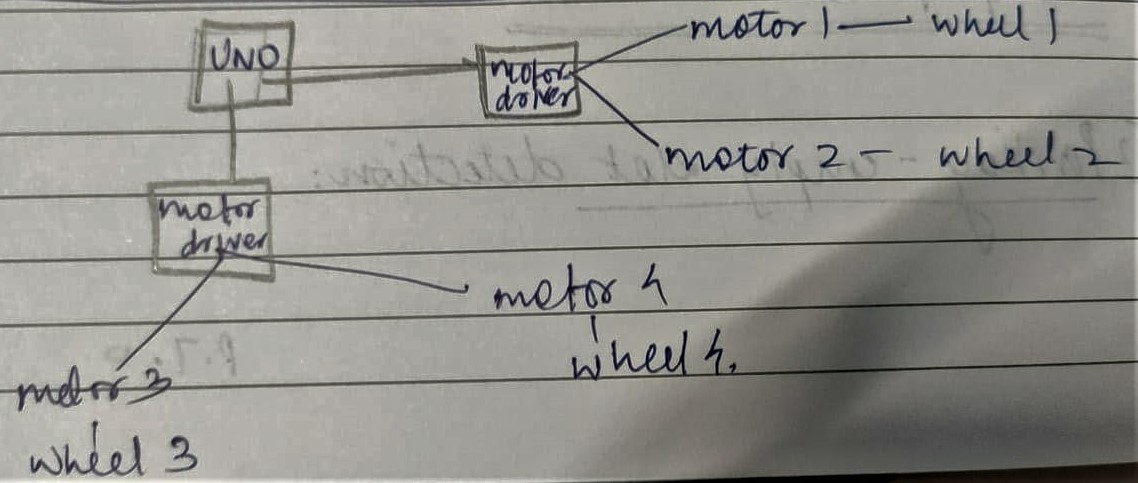




* In the second image, the ultrasound sensor will help detect if a slot is empty or not. We can also use an IR sensor in place of the ultrasound sensor here.

1. Car:





In the above connections, the motor drivers are used for i2c connection, so as to control the speed (PWM) and direction of movement of the car.

**Logic: -**

* Firstly, the parking system will send the info about which slot is empty. Only if a slot is empty, the gates of the parking lot will open.
* Now, the car will enter (assuming that a slot is empty).

1. Case 1: The slot one is empty. The car will go straight ahead, and once the distance between the wall and car is (say x cm), the car will be instructed to stop.
2. Case 2: The slot one is NOT empty. The car will the go a certain distance, turn right and the ultrasound sensor will be instructed to turn 90 degrees. Now, once the ultrasound sensor finds that an empty slot, the car turns left, the sensor turns left too and the car gets itself parked.

**Coding: -**

1. Parking system:

**(For each ultrasound sensor)**

const int trigPin =9;

const int echoPin =10;//defines variables---- trig==ouput;echo==input

long duration;

int distance;

void setup() {

pinMode(trigPin,OUTPUT);

pinMode(echoPin,INPUT);

Serial.begin(9600);

}

void loop() {

digitalWrite(trigPin,LOW);

delayMicroseconds(10); //sets trigPin on HIGH for state for 10 microseconds

digitalWrite(trigPin,HIGH);

delayMicroseconds(10);

digitalWrite(trigPin,LOW);

duration=pulseIn(echoPin,HIGH); //to get the time duration for which echoPin was on

distance=duration\*0.034\*0.5; //calculates distance

if(distance>5){

println("This slot is Empty");

}

delay(2000);

}

1. Car System:
2. Case 1:

#include<Wire.h> //including the wire library which allows you to communicate with I2C/TWI devices

int in2 = 5;

int in3 = 6;

int in4 = 7;

int enA = 9; // initialising pin 9 as PWM for motor A

int enB = 10; // initialising pin 10 as PWM for motor B

const int trigPin =1;

const int echoPin =2;//defines variables---- trig==ouput;echo==input

long duration;

int distance;

void setup() {

Wire.begin(SLAVE\_ADDRESS); //begin I2C transmission with the slave device(motor driver in this case)

Wire.onReceive(receiveEvent); //register the event

Serial.begin(9600); //start the serial output

pinMode(in1, OUTPUT);

pinMode(in2, OUTPUT);

pinMode(enA, OUTPUT);

pinMode(in3, OUTPUT); //Declaring the pin modes, obviously they're outputs

pinMode(in4, OUTPUT);

pinMode(enB, OUTPUT);

pinMode(trigPin,OUTPUT);

pinMode(echoPin,INPUT);

Serial.begin(9600);

}

void loop() {

digitalWrite(trigPin,LOW);

delayMicroseconds(10); //sets trigPin on HIGH for state for 10 microseconds

digitalWrite(trigPin,HIGH);

delayMicroseconds(10);

digitalWrite(trigPin,LOW);

duration=pulseIn(echoPin,HIGH); //to get the time duration for which echoPin was on

distance=duration\*0.034\*0.5; //calculates distance

if(distance <5){

digitalWrite(in1, LOW);

digitalWrite(in2, LOW);

digitalWrite(in3, LOW);

digitalWrite(in4, LOW);

analogWrite(enB,255); //this varies the speed of the motors (255 implies max) maybe 255 here is PWM

analogWrite(enA,255); // both the motors are stopped-BRAKE

}

delay(2000);

}

1. Case 2:

#include <Servo.h> // Declare the Servo pin

int servoPin = 3; // Create a servo object

Servo Servo1; #include<Wire.h> //including the wire library which allows you to communicate with I2C/TWI devices

int in2 = 5;

int in3 = 6;

int in4 = 7;

int enA = 9; // initialising pin 9 as PWM for motor A

int enB = 10; // initialising pin 10 as PWM for motor B

const int trigPin =1;

const int echoPin =2;//defines variables---- trig==ouput;echo==input

long duration;

int distance;

void setup() {

Wire.begin(SLAVE\_ADDRESS); //begin I2C transmission with the slave device(motor driver in this case)

Wire.onReceive(receiveEvent); //register the event

Serial.begin(9600); //start the serial output

pinMode(in1, OUTPUT);

pinMode(in2, OUTPUT);

pinMode(enA, OUTPUT);

pinMode(in3, OUTPUT); //Declaring the pin modes, obviously they're outputs

pinMode(in4, OUTPUT);

pinMode(enB, OUTPUT);

pinMode(trigPin,OUTPUT);

pinMode(echoPin,INPUT);

Serial.begin(9600);

Servo1.attach(servoPin);

}

void loop() {

digitalWrite(trigPin,LOW);

delayMicroseconds(10); //sets trigPin on HIGH for state for 10 microseconds

digitalWrite(trigPin,HIGH);

delayMicroseconds(10);

digitalWrite(trigPin,LOW);

duration=pulseIn(echoPin,HIGH); //to get the time duration for which echoPin was on

distance=duration\*0.034\*0.5; //calculates distance

if(distance <3){ //distance from the car parked in slot 1

analogWrite(EA, 180); //speed 0 - 255

analogWrite(EB, 180); //speed 0 - 255

digitalWrite(EN2, HIGH);

digitalWrite(EN3, LOW);

digitalWrite(EN4, LOW);

digitalWrite(EN5, HIGH); // turns the car right

Servo1.write(90);

delay(1000); //will rotate the servo motor by 90 degrees and thus the ultrasound sensor will turn 90 degrees.

if(distance>5){ //now the 90 degrees turned ultrasound sensor looks for an empty slot

digitalWrite(in1, LOW);

digitalWrite(in2, LOW);

digitalWrite(in3, LOW);

digitalWrite(in4, LOW);

analogWrite(enB,255);

analogWrite(enA,255); // both the motors are stopped-BRAKE

analogWrite(EA, 180);

analogWrite(EB, 180);

digitalWrite(EN2, LOW);

digitalWrite(EN3, HIGH);

digitalWrite(EN4, HIGH);

digitalWrite(EN5, LOW); // car is turned left, ready to park

Servo1.write(0);

delay(1000); //will rotate the servo motor by 90 degrees and thus the ultrasound sensor will turn 90 degrees COMING BACK TO ITS ORIGINAL POSITION

if(distance <5){

digitalWrite(in1, LOW);

digitalWrite(in2, LOW);

digitalWrite(in3, LOW);

digitalWrite(in4, LOW);

analogWrite(enB,255);

analogWrite(enA,255); // both the motors are stopped-BRAKE

}

}

}

delay(2000);

}