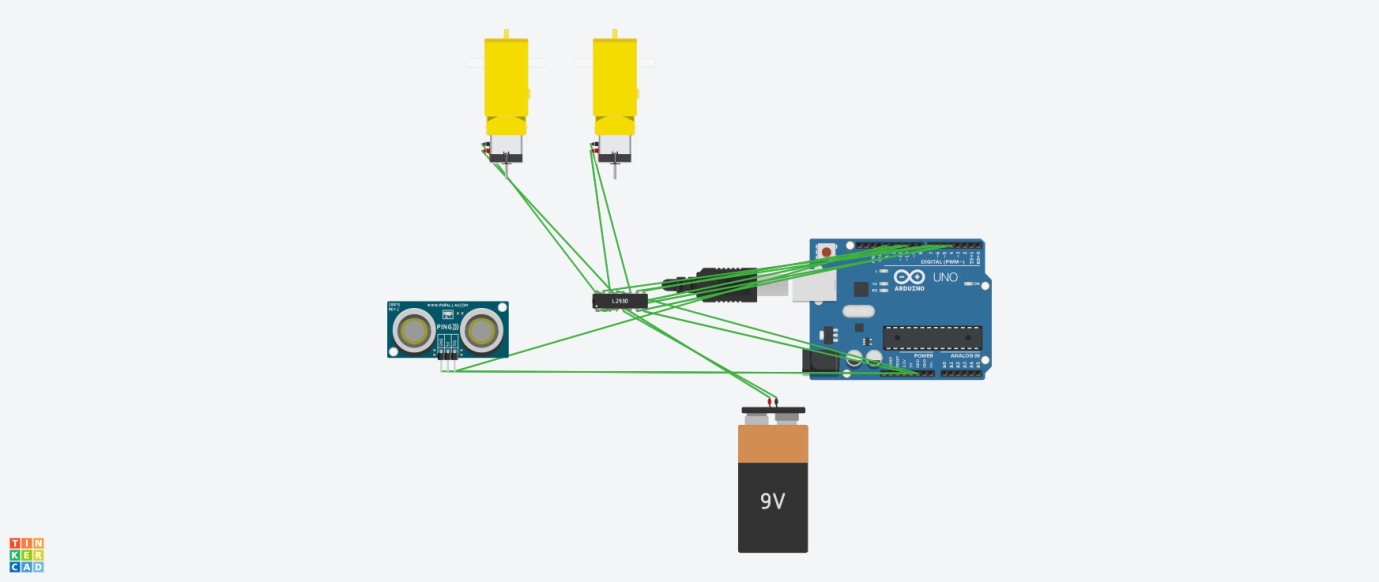
**PHONE CONTROL ROBOT**

**Cell Phone Controlled Robotic Vehicle:**

When we talk about wireless robotic vehicles, We generally think about RF technology circuits. But this project is very different. It uses a cell phone to control the motion of the robotic vehicle. Here we have used DTMF technology to control the robotic vehicle by using the mobile phone. We used a cell-phone and a gsm module, one is connected to robot and another is user phone. Communication is established between these a mobile and Simcom module and if any key is pressed, the robots moves or work as its programmed.

This project is designed to develop a robotic vehicle that is controlled by the cell phone. This is based on Simcom modules and Arduino UNO. Simcom modules uses a simcard to communicate with different mobile device to gets command from.

**How will it work?**

****

**(Circuit design without SIM808 module)**

I have written my program in Arduino ide which I have already uploaded in the folder with the details so it is easier to understand. My code is based on dtmf tones**.**

## Sending DTMF tones using a GSM modem

The following code sample demonstrates how to send a numeric text message to a pager using a GSM modem. The sample checks if there is a PIN code on the SIM card in the GSM modem and sets the code.

If the code is correct, the modem will dial the number in voice mode, waits till the connection has been established and sends the DTMF code. Finally, the connection is terminated.

The following AT Commands are used:

**ATDxxxxxxxxxxxxxxx;**

The ATD command is used to dial a number. The semi-colon at the end of this number indicates that this connection has to be setup as a voice call. When leaving this semi-colon, the connection is set up as a data call.

**AT+VTD=xx**

This command is optional, and is used to set the DTMF tone duration in 100mS steps.

**AT+VTS=x**

To send the actual DTMF code, you have to call this command for every digit sent. To send, for instance "\*1234#", you have to send:

**AT+VTD=\*;+VTD=1;+VTD=2;+VTD=3;+VTD=4;+VTD=#**

**ATH0**

Terminates the phone connection (hangup).

So, whenever I press a button on my phone that signal will go to gsm module and based on how I programmed the robot for each button it will move. The Distance Sensors in it are used to avoid any collisions. I have set a particular value if the distance between the robot and and the obstacle is less than the max distance then it will come backwards.

**Items needed to create the robot:-**

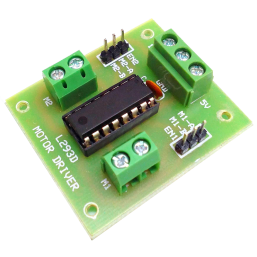
**Battery-12v(2)**

****

**BO motors-**

****

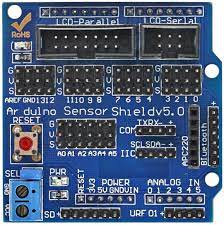
**Motor driver- L293d**

****

**Arduino UNO-**

****

**Arduino Sensor Shield-**

****

**Wheels for BO motors-**

****

**SIM808 module (This in the one I used in making my phone controlled robot)-**

****

**Picture of phone controlled robot made by me-A picture containing text, floor

Description automatically generated**

**Code**

**#include <SoftwareSerial.h>**

**// defining communicationion pins for Software Serial**

**# define GSM\_RX 10 // Connect TX of GSM module**

**# define GSM\_TX 11 // Connect RX of GSM module**

**// Defining interfacing pins for Motor Control (ie pins to L293D)**

**// Code is designed such that the two motor terminals are named A & B. and is assumed that the motor will rotate clock wise when A terminal is HIGH and B is LOW**

**# define LEFT\_A 5**

**# define LEFT\_B 7**

**# define RIGHT\_A 2**

**# define RIGHT\_B 4**

**# define enA 3 //pwm pins to control speed of the motor**

**#define enB 6**

**int a;**

**// Defining the key press to drive the car.**

**# define INC '3'**

**# define DEC '1'**

**# define FORWARD '2'**

**# define RIGHT '6'**

**# define LEFT '4'**

**# define BACKWARD '8'**

**# define STOP '5'**

**# define SLOW1 '\*' //slowest speed**

**# define SLOW2 '0' //slower speed**

**# define SLOW3 '#' //fastest speed**

**int sensorpin = 0; //collecting data from distance sensor**

**int val = 0;int Front\_D = 0;**

**SoftwareSerial gsm\_board(GSM\_RX,GSM\_TX);**

**boolean call\_flag=0;**

**int i=0,x=-1;**

**char n;**

**void gsm\_initilaize();// used to inilitize the gsm and chk all its parameters**

**void move(); // used to control motor outputs.**

**void forward(); //functions to set pins to control car movement**

**void backward();**

**void left();**

**void right();**

**void stop();**

**void setup() {**

**// put your setup code here, to run once:**

**gsm\_board.begin(9600);**

**Serial.begin(9600); //setting the baud rate in the serial monitor**

**pinMode(enA, OUTPUT);**

**pinMode(enB, OUTPUT);**

**pinMode(LEFT\_A,OUTPUT);**

**pinMode(LEFT\_B,OUTPUT);**

**pinMode(RIGHT\_A,OUTPUT);**

**pinMode(RIGHT\_B,OUTPUT);**

**stop();**

**gsm\_initilaize();**

**}**

**//////////////////////////////setup ends/////////////////////////////**

**/////////////////////loop begins///////////////////////////**

**void loop() {**

**move();**

**String gsm\_data; // to hold incomming communication from GSM module**

**while(gsm\_board.available())**

**{**

**char c=gsm\_board.read();**

**gsm\_data+=c;**

**delay(10);**

**} //read serial data and store it to gsm\_data STRING instance;**

**if(!call\_flag) // if call is not in connected, checking for ring**

**{**

**x=gsm\_data.indexOf("RING");**

**if(x>-1)**

**{**

**delay(5000); // change this delay value to change the delay in attending the call by your car.**

**gsm\_board.println("ATA");**

**call\_flag=1;**

**}**

**} // ring test over, call flag high if sim rings**

**if(call\_flag) // if call is connected**

**{**

**x=gsm\_data.indexOf("DTMF"); //checking dtmf and storing approprietly**

**if(x>-1)**

**{**

**n=gsm\_data[x+6];**

**Serial.println(n);**

**move(); // call motor control function to initiate the change demanded by controller**

**}**

**x=gsm\_data.indexOf("NO CARRIER"); // Checking whether call is still connected or not**

**if(x>-1)**

**{**

**gsm\_board.println("ATH");**

**call\_flag=0;**

**}**

**}**

**}**

**//////////////////////////////loop ends/////////////////////////////**

**/////////////////////gsm inilitize begins///////////////////////////**

**void gsm\_initilaize()**

**{**

**boolean gsm\_Ready=1;**

**Serial.println("initializing GSM module");**

**while(gsm\_Ready>0)**

**{**

**gsm\_board.println("AT");**

**Serial.println("AT");**

**while(gsm\_board.available())**

**{**

**if(gsm\_board.find("OK")>0)**

**gsm\_Ready=0;**

**}**

**delay(2000);**

**}**

**Serial.println("AT READY");**

**// GSM MODULE REPLIED 'OK' TO 'AT' INPUT, INDICAING THE MODULE IS OK AND FUNCTIONING**

**boolean ntw\_Ready=1;**

**Serial.println("finding network");**

**while(ntw\_Ready>0)**

**{**

**gsm\_board.println("AT+CPIN?");**

**Serial.println("AT+CPIN?");**

**while(gsm\_board.available())**

**{**

**if(gsm\_board.find("+CPIN: READY")>0)**

**ntw\_Ready=0;**

**}**

**delay(2000);**

**}**

**Serial.println("NTW READY");**

**// GSM MODULE REPLIED '+CPIN:READY' TO 'AT+CPIN?' INPUT, INDICAING THE NETWORK IS OK AND FUNCTIONING**

**boolean DTMF\_Ready=1;**

**Serial.println("turning DTMF ON");**

**while(DTMF\_Ready>0)**

**{**

**gsm\_board.println("AT+DDET=1");**

**Serial.println("AT+DDET=1");**

**while(gsm\_board.available())**

**{**

**if(gsm\_board.find("OK")>0)**

**DTMF\_Ready=0;**

**}**

**delay(2000);**

**}**

**Serial.println("DTMF READY");// GSM MODULE REPLIED '+OK' TO 'AT+DDET=1?' INPUT, INDICAING THE DTMF IS ON;**

**}**

**//////////////////////////////gsm inilitization ends/////////////////////////////**

**/////////////////////move begins///////////////////////////**

**void move()**

**{**

**switch(n)**

**{**

**case FORWARD:**

**forward();**

**break;**

**case BACKWARD:**

**backward();**

**break;**

**case RIGHT:**

**right();**

**break;**

**case LEFT:**

**left();**

**break;**

**case STOP:**

**stop();**

**break;**

**case SLOW1:**

**slow1();**

**break;**

**case SLOW2:**

**slow2();**

**break;**

**case SLOW3:**

**slow3();**

**break;**

**case INC:**

**inc();**

**break;**

**case DEC:**

**dec();**

**}**

**}**

**void inc()**

**{**

**if(a<250)**

**{a=a+10;**

**analogWrite(enB, a);analogWrite(enA, a);**

**}**

**}**

**void dec()**

**{**

**if(a>100)**

**{**

**a=a-10;**

**analogWrite(enB, a);analogWrite(enA, a);**

**}**

**}**

**/////////////////////move ends///////////////////////////**

**/////////////////movement control function begins///////////////////**

**void slow1()**

**{**

**analogWrite(enB, 100);analogWrite(enA, 100);//setting up speed**

**a=100;**

**}**

**void slow2()**

**{**

**analogWrite(enB, 170);analogWrite(enA, 170);//setting up speed**

**a=170;**

**}**

**void slow3()**

**{**

**analogWrite(enB, 250);analogWrite(enA, 250);//setting up speed**

**a=250;}**

**void forward()//making a robot car move forward**

**{**

**digitalWrite(LEFT\_A,LOW);**

**digitalWrite(LEFT\_B,HIGH);**

**digitalWrite(RIGHT\_A,HIGH);**

**digitalWrite(RIGHT\_B,LOW);**

**}**

**void backward()//making a robot car moce backward**

**{**

**digitalWrite(LEFT\_A,HIGH);**

**digitalWrite(LEFT\_B,LOW);**

**digitalWrite(RIGHT\_A,LOW);**

**digitalWrite(RIGHT\_B,HIGH);**

**}**

**void left()//making my robot car rotate left direction**

**{**

**digitalWrite(LEFT\_A,LOW);**

**digitalWrite(LEFT\_B,HIGH);**

**digitalWrite(RIGHT\_A,LOW);**

**digitalWrite(RIGHT\_B,HIGH);**

**}**

**void right()//making my robot car rotate in right direction**

**{**

**digitalWrite(LEFT\_A,HIGH);**

**digitalWrite(LEFT\_B,LOW);**

**digitalWrite(RIGHT\_A,HIGH);**

**digitalWrite(RIGHT\_B,LOW);**

**}**

**void stop()//to stop the robot from moving**

**{**

**digitalWrite(LEFT\_A,LOW);**

**digitalWrite(LEFT\_B,LOW);**

**digitalWrite(RIGHT\_A,LOW);**

**digitalWrite(RIGHT\_B,LOW);**

**}**