PERSONAL PROJECT

REMOTE-CONTROLLED

ARDUINO TANK

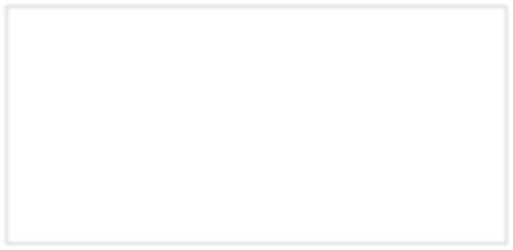
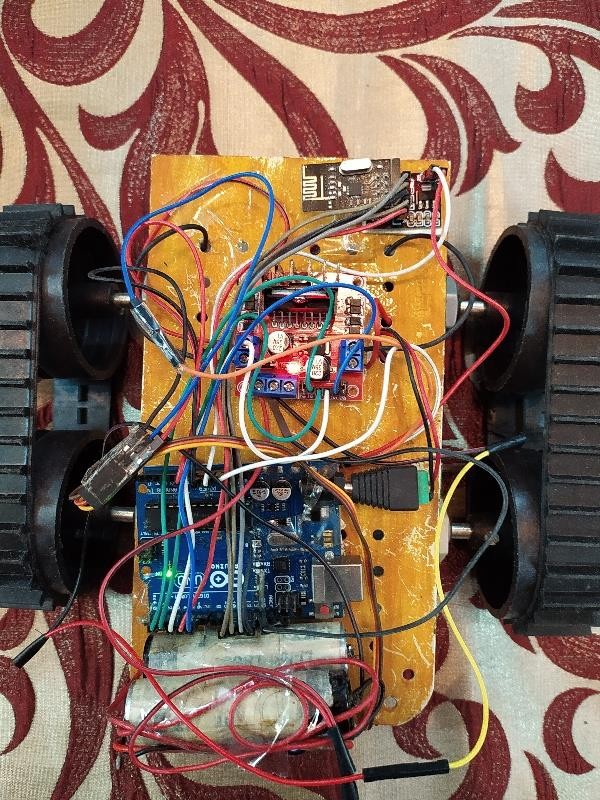
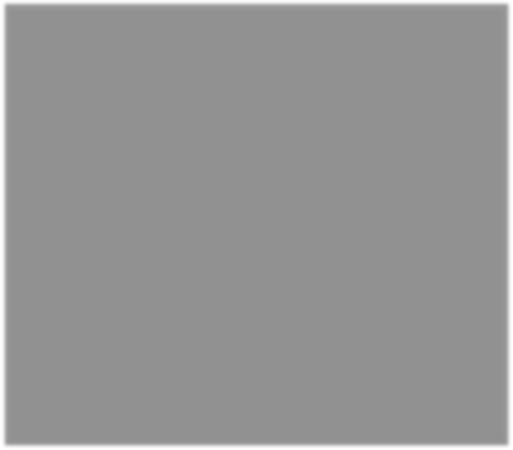
PROJECT BY

HITESH DAGA

VAIBHAV NEMANI

Model Made By :-

Arduino Software



**TITLE :**

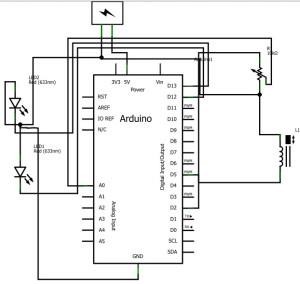
**REMOTE CONTROL TANK USING ARDUINO and nRF24L01**

### INTRODUCTION :

Arduino is an open source electronics platform based on easy to use hardware and software. Arduino boards are able to read inputs such as the light on a sensor, a finger on a button and so on, to convert it into an output, hence activating a motor, turning on an LED, or publishing something online. Arduino is basically a microcontroller board, with on board power supply and USB port. It is a prototype platform which consists of a circuit board and the Arduino IDE, which is used to write and upload the code into the physical board. A remote controlled tank using Arduino is a project where the motor functions of the tank is implemented and controlled via the code in the Arduino IDE, which is reflected onto the physical board.

### THEORY/WORKING PRINCIPLE :

The Arduino UNO board has 14 digital input/output pins in which six can be used as PWM outputs, 16MHz quartz crystal, a ceramic resonator, ICSP header, a USB connection, 6 analog inputs, power jack and reset button. The Arduino board is connected to a power supply like a battery.



The working of the remote controlled tank using Arduino is based on the principle of Infrared transmission, which means the IR LED at the transmitter remote will send HEX codes and the receiver in the motor tank will get those values and the tank will move accordingly. The Arduino board used in the transmitter will read analog values from the joystick and will send the required HEX codes. The TSOP receiver at the tank can read those HEX value and hence, the tank will move likewise.

## METHOD :

TANK CHASSIS :



Initially, the components are placed in the tank chassis of the tank and the L298N Motor drive is assembled in the Tank chassis. Then, the DC motors and power is connected to the motor driver L298N, following which the battery used for Arduino and the Arduino board is inserted into the tank. The nRF24L01 transceiver module is also inserted into the tank.

After which, the following connections are made:

* nRF24L01 Adapter VCC is connected to the Arduino at +5V
* nRF24L01 GND is connected to the Arduino GND
* nRF24L01 CE is connected to the Arduino Digital 5
* nRF24L01 CSN is connected to the Arduino Digital 10
* nRF24L01 SCK is connected to the Arduino Digital 13
* nRF24L01 MOSI is connected to the Arduino Digital 11
* nRF24L01 MIS is connected to the Arduino Digital 12

In the L298N Motor Control Module, the following connections are made:

* L298N/IN1 is connected to the Arduino Digital 2
* L298N/IN2 is connected to the Arduino Digital 3
* L298N/IN3 is connected to the Arduino Digital 4
* L298N/IN4 is connected to the Arduino Digital 6

## PROGRAMMING OF THE TANK :

In the code for the tank, reference calls to the following libraries are made:

* (“RF24.h” – Library to control radio modem nRF24L
* <SPI.h> - Communication interface with the modem

**/\***

**---- Receiver Code ----**

**\*/**

**#include <Servo.h> //the library which helps us to control the servo motor #include <SPI.h> //the communication interface with the modem**

**#include "RF24.h" //the library which helps us to control the radio modem (nRF24L)**

**//define our L298N control pins**

**//Motor A**

**const int RightMotorForward = 2; // IN1 const int RightMotorBackward = 3; // IN2**

**//Motor B**

**const int LeftMotorForward = 4; // IN3 const int LeftMotorBackward = 6; // IN4**

**//define the servo name Servo myServo;**

**RF24 radio(5,10); /\*This object represents a modem connected to the Arduino.**

**Arguments 5 and 10 are a digital pin numbers to which signals CE and CSN are connected.\*/**

**const uint64\_t pipe = 0xE8E8F0F0E1LL; //the address of the modem,that will receive data from the Arduino.**

**int data[1];**

**void setup(){ pinMode(RightMotorForward, OUTPUT); pinMode(LeftMotorForward, OUTPUT); pinMode(LeftMotorBackward, OUTPUT); pinMode(RightMotorBackward, OUTPUT);**

**//define the servo input pins myServo.attach(14); //A0**

**radio.begin(); //it activates the modem.**

**radio.openReadingPipe(1, pipe); //determines the address of our modem which receive data.**

**radio.startListening(); //enable receiving data via modem**

**}**

**void loop(){ if(radio.available()){ radio.read(data, 1);**

**if(data[0] < 11 && data[0] > 6){**

**// This is backward**

**// Set a Motor A backward digitalWrite(RightMotorForward, LOW); digitalWrite(RightMotorBackward, HIGH);**

**// Set a Motor B backward digitalWrite(LeftMotorForward, LOW); digitalWrite(LeftMotorBackward, HIGH);**

**}**

**if(data[0] > -1 && data[0] < 4){**

**// This is forward**

**// Set a Motor A forward digitalWrite(RightMotorForward, HIGH); digitalWrite(RightMotorBackward, LOW);**

**// Set a Motor B forward digitalWrite(LeftMotorForward, HIGH); digitalWrite(LeftMotorBackward, LOW);**

**}**

**if (data[0] == 5){**

**// Stop Motors digitalWrite(RightMotorForward, LOW); digitalWrite(RightMotorBackward, LOW); digitalWrite(LeftMotorForward, LOW); digitalWrite(LeftMotorBackward, LOW);**

**}**

**// This is Backward**

**// Set a Motor A Backward if(data[0] < 21 && data[0] > 16){**

**digitalWrite(RightMotorForward, HIGH);**

**digitalWrite(RightMotorBackward, LOW);**

**// Set a Motor B Backward digitalWrite(LeftMotorForward, LOW); digitalWrite(LeftMotorBackward, HIGH);**

**}**

**// Turn Right**

**if(data[0] > 10 && data[0] < 14){ digitalWrite(RightMotorForward, LOW); digitalWrite(RightMotorBackward, HIGH); digitalWrite(LeftMotorForward, HIGH); digitalWrite(LeftMotorBackward, LOW);**

**}**

**// Turn Left if(data[0] == 15){**

**digitalWrite(RightMotorForward, LOW); digitalWrite(RightMotorBackward, LOW); digitalWrite(LeftMotorForward, LOW); digitalWrite(LeftMotorBackward, LOW);**

**}**

**// for the servo motor**

**if(data[0] < 31 && data[0] > 21){ int potValue = data[0];**

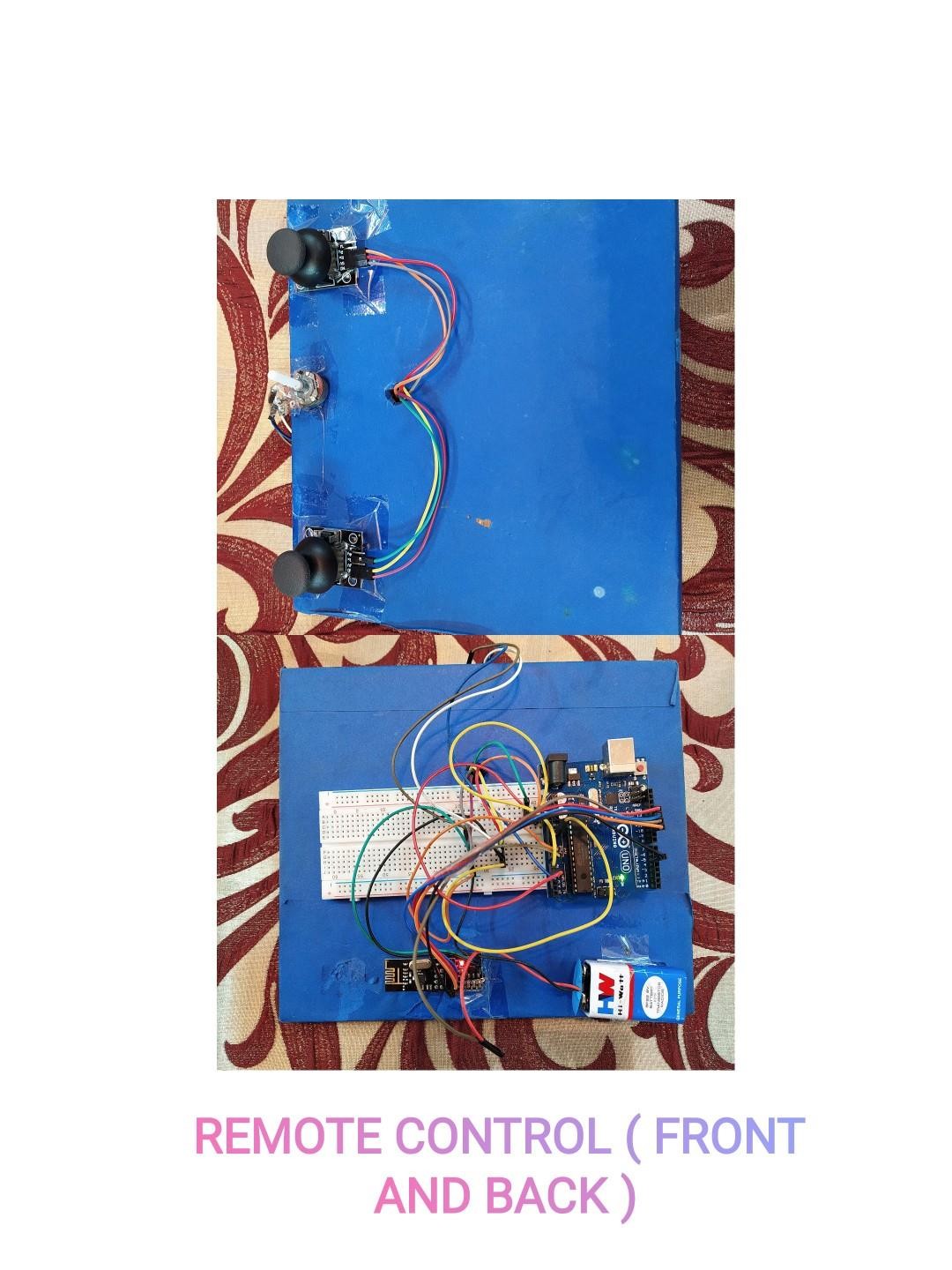
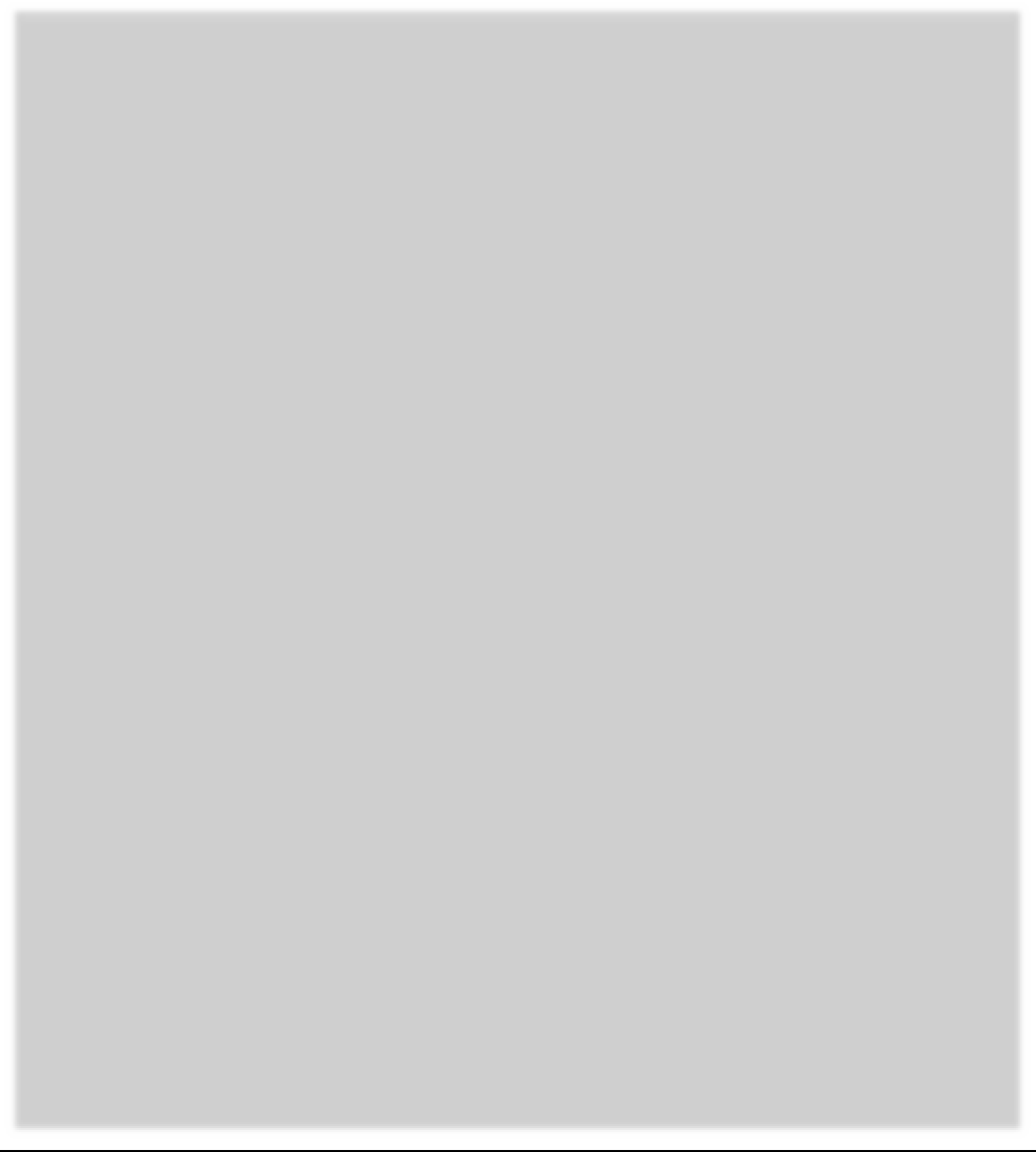
**int potPos = map(potValue, 21, 30, 10, 170); myServo.write(potPos);**

**}**

**}**

**}**

## REMOTE CONTROL:



In the remote control segment, two pieces of Joystick modules are used which can be moved along the X and Y axes. In this segment, the following connections are made:

* nRF24L01/CE is connected to the Arduino Digital 5
* nRF24L01/CSN is connected to the Arduino Digital 10
* nRF24L01/SCK is connected to the Arduino Digital 13
* nRF24L01/MOSI is connected to the Arduino Digital 11
* nRF24L01/MISO is connected to the Arduino Digital 12
* VCC and GND is connected to Arduino +5 and GND respectively
* Positive pole of the battery plug is connected to VIN input of Arduino.
* GND pin of the joystick is connected to the GND input of Arduino.
* VCC pin of the joystick is connected to the +5V input of the Arduino.
* The X-axis signal pin of the joystick is connected to the analog 0 input of the Arduino.
* The Y-axis signal pin of the joystick is connected to the analog 1 input of the Arduino.

## PROGRAMMING OF THE REMOTE CONTROL :

In the code for the remote control, reference calls to the following libraries are made:

* “RF24.h” – Library to control radio modem nRF24L
* <SPI.h> - Communication interface with the modem

**/\***

**---- Transmitter Code ----**

**\*/**

**#include <SPI.h> //the communication interface with the modem #include "RF24.h" //the library which helps us to control the radio modem**

**//define the input pins int x\_axis = A0;**

**int y\_axis = A1; int potPin = A2;**

**//define variable values int xValue;**

**int yValue; int potValue;**

**int data[1];**

**RF24 radio(5,10); //5 and 10 are a digital pin numbers to which signals CE and CSN are connected.**

**const uint64\_t pipe = 0xE8E8F0F0E1LL; //the address of the modem, that will receive data from Arduino.**

**void setup(void){ Serial.begin(9600);**

**radio.begin(); //it activates the modem.**

**radio.openWritingPipe(pipe); //sets the address of the receiver to which the program will send data.**

**}**

**void loop(){**

**//Send X-axis data**

**xValue = analogRead(x\_axis);**

**xValue = map(xValue, 0, 1023, 0, 10); data[0] = xValue;**

**radio.write(data, 1);**

**Send Potentiometer data potValue = analogRead(potPin);**

**potValue = map(potValue, 0, 1023, 21, 30); data[0] = potValue;**

**radio.write(data, 1);**

**}**

**//Send Y-axis data**

**yValue = analogRead(y\_axis);**

**yValue = map(yValue, 0, 1023, 11, 20); data[0] = yValue;**

**radio.write(data, 1);**

**//Send Potentiometer data potValue = analogRead(potPin);**

**potValue = map(potValue, 0, 1023, 21, 30); data[0] = potValue;**

**radio.write(data, 1);**

**}**

## CONCLUSION & ADVANTAGE :

Hence, by following the appropriate code and connections using the Arduino board, we will be able to construct the remote controlled tank with Arduino.

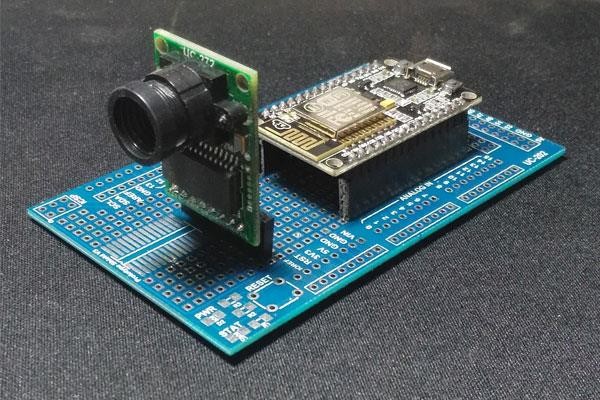
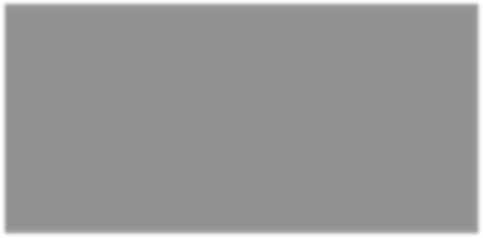
Nowadays, we see a lot of Radio Frequency remote control toys but there are certain disadvantages when we use them.

There is no secure connection in the radio frequency remote controlled tanks, which means that the tank can be controlled by any radio frequency remote. Signal attenuation occurs when the tank is controlled by two radio frequency remotes which implies that the tank gets confused and does not respond, hence unable to implement its motor functions effectively. The radio frequency receivers come in fixed frequencies and hence cannot be tuned to avoid the problem.

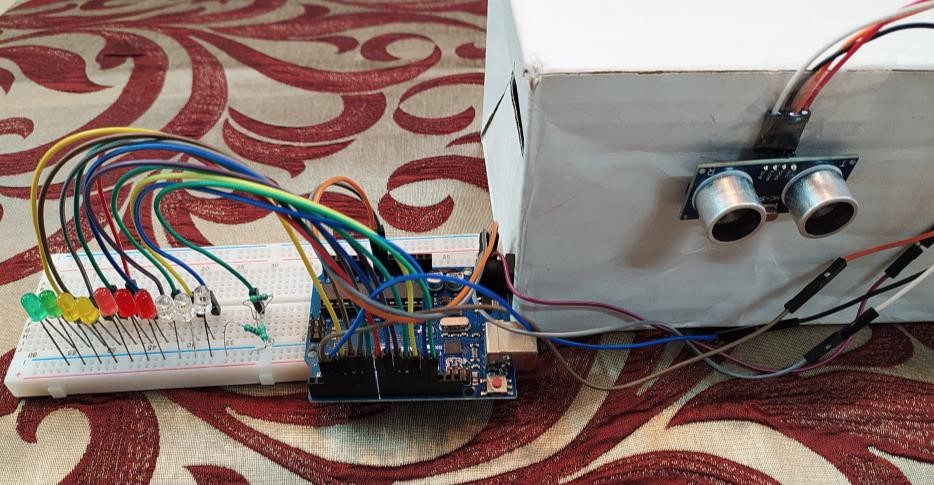
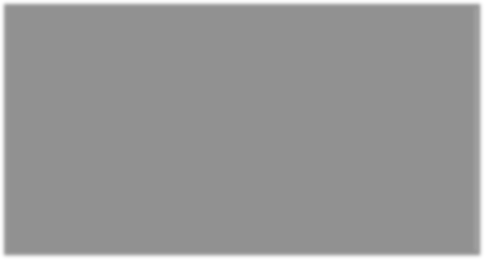
This is where Arduino boards are used, which implement much more effective and unique remote controlled tank system.

## FUTURE DEVELOPMENTS :

1. We can upgrade the remote controlled tank by connecting a wifi enabled camera to the model. The footage can be relayed to any electronic device which has a wifi facility. Such upgradations prove to be useful when the footage relayed by the camera can be used for security and intelligence purposes and also serve as a marketable product in the recreational domain.



1. We can also add ultrasonic sensors to the tank and take the readings and find the distance of the nearest object from the tank. We can program the tank such that if the tank comes to a particular distance from the object, the motors will turn off. This can be used in cars too to prevent accidents happening on road.



**DATE OF COMPLETION : 03/01/2021**