

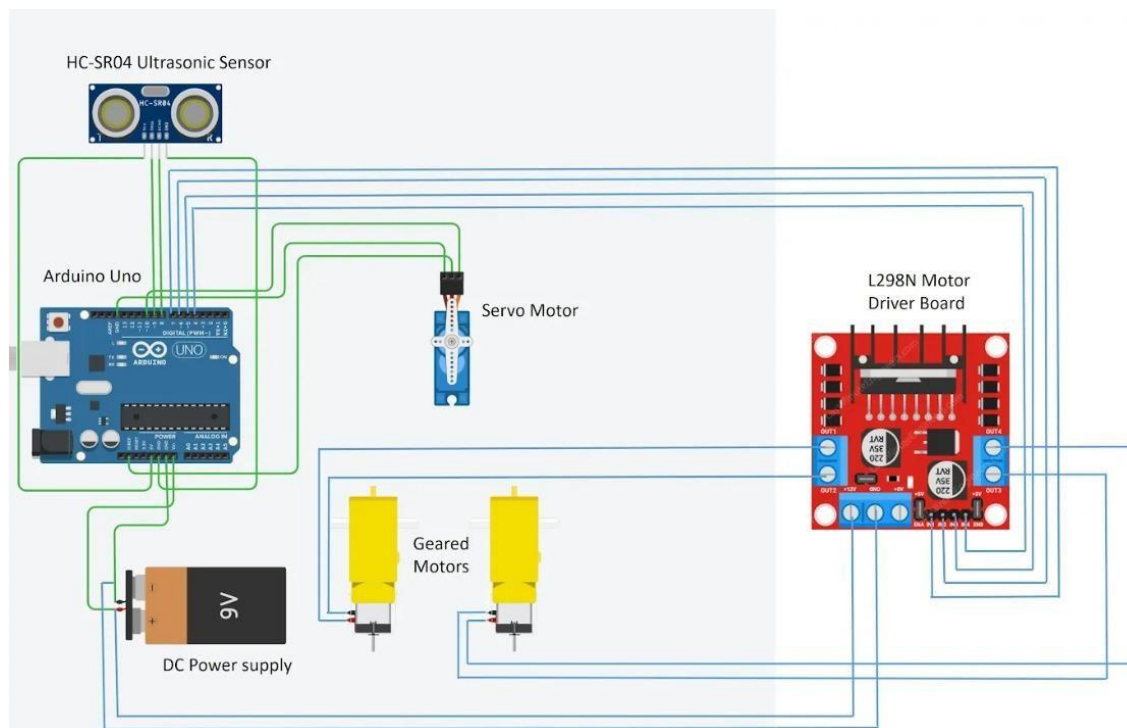
Problem Statement

To make an obstacle-avoidance robot consisting of Arduino UNO and Ultrasonic Sonic Sensor, driven by DC Motors, that publishes ROS messages about the distance between the obstacle and the Ultrasonic Sensor using `rosserial_python`.

Components

- Arduino UNO
- HC-SR04 Ultrasonic Sensor
- SG90 Micro Servo Motor
- L289N Motor Driver
- 12V DC Motors
- Li-ion Cells
- Breadboard

Circuit Diagram



Code

```
#include <Servo.h>
#include <ros.h>
#include <std_msgs/Int8.h>

Servo Myservo;
#define trigPin 9           // Trig Pin of Ultrasonic Sensor
#define echoPin 8          // Echo Pin of Ultrasonic Sensor
#define MLa 4              // 1st Pin of Left Motor
#define MLb 5              // 2nd Pin of Left Motor
#define MRa 6              // 1st Pin of Right Motor
#define MRb 7              // 2nd Pin of Right Motor

long duration, distance;

ros::NodeHandle nh;         // Instantiate the node handle to
create publisher and subscriber
std_msgs::Int8 str_msg;     // Message object to hold ultrasonic
sensor data
ros::Publisher chatter("chatter", &str_msg); // Instantiate the
publisher with the name chatter

void setup() {
    Serial.begin(9600);
    nh.initNode();           // Initialize the ROS node handle
    nh.advertise(chatter);   // Declare the topic "chatter"

    pinMode(MLa, OUTPUT);    // Set Motor Pins as output
    pinMode(MLb, OUTPUT);
    pinMode(MRa, OUTPUT);
    pinMode(MRb, OUTPUT);
    pinMode(trigPin, OUTPUT); // Set Trig Pin as output to transmit
the waves
    pinMode(echoPin, INPUT);  // Set Echo Pin as input to receive
the reflected waves

    Myservo.attach(10);
}

void loop() {
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
```

```

digitalWrite(trigPin, HIGH);          // Transmit the waves for 10µs
delayMicroseconds(10);
duration = pulseIn(echoPin, HIGH); // Receive the reflected waves
distance = duration / 58.2;          // Obtain the distance
Serial.println(distance);

// Publish ultrasonic sensor data
str_msg.data = distance;
chatter.publish(&str_msg);
nh.spinOnce();

delay(10);

if (distance > 15) { // Condition for the absence of an obstacle
    Myservo.write(90);
    digitalWrite(MRb, HIGH); // Move forward
    digitalWrite(MRa, LOW);
    digitalWrite(MLb, HIGH);
    digitalWrite(MLa, LOW);
} else if ((distance < 10) && (distance > 0)) { // Condition for
the presence of an obstacle
    digitalWrite(MRb, LOW); // Stop
    digitalWrite(MRa, LOW);
    digitalWrite(MLb, LOW);
    digitalWrite(MLa, LOW);
    delay(100);

    Myservo.write(0);
    delay(500);
    Myservo.write(180);
    delay(500);
    Myservo.write(90);
    delay(500);

    digitalWrite(MRb, LOW); // Move backward
    digitalWrite(MRa, HIGH);
    digitalWrite(MLb, LOW);
    digitalWrite(MLa, HIGH);
    delay(500);
    digitalWrite(MRb, LOW); // Stop
    digitalWrite(MRa, LOW);
    digitalWrite(MLb, LOW);
    digitalWrite(MLa, LOW);
    delay(100);
    digitalWrite(MRb, HIGH); // Move left

```

```

    digitalWrite(MRa, LOW);
    digitalWrite(MLa, LOW);
    digitalWrite(MLb, LOW);
    delay(500);
}
}

```

Explanation

- ❖ The code begins by including necessary libraries such as Servo for controlling servo motors and ros.h for integrating with ROS. Additionally, it includes std_msgs/Int8.h for defining a standard message type for ROS communication.
- ❖ Constants are declared for various pins used in the circuit, including pins for the ultrasonic sensor, motor control, and the servo motor. The ROS node handle nh is initialized to facilitate communication with ROS. A publisher named "chatter" is created to publish ultrasonic sensor data.
- ❖ In the setup function, serial communication is initiated for debugging purposes, and the ROS node is initialized. Pin modes are set for input and output, and the servo motor is attached to a specific pin.

The main loop continuously performs the following steps:

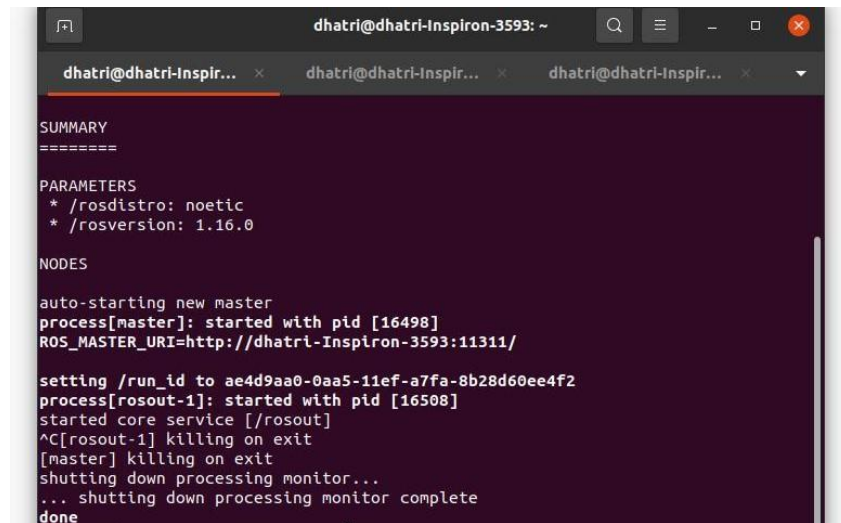
- ❖ Sends a pulse to the ultrasonic sensor to trigger it.
- ❖ Measures the duration of the pulse returned by the sensor to calculate the distance of the obstacle.
- ❖ Publishes the distance data to the ROS topic.
- ❖ Checks the distance measurement to determine the robot's behavior:
 - If no obstacle is detected within a certain range, the robot moves forward.
 - If an obstacle is detected, the robot stops, performs scanning with the servo motor, executes avoidance maneuvers, and then resumes forward motion.

When an obstacle is detected, the robot performs a series of actions to avoid the obstacle: It stops momentarily.

- The servo motor scans left and right to detect the direction of the obstacle.
- The robot backs up, turns, and then moves forward again to navigate around the obstacle.

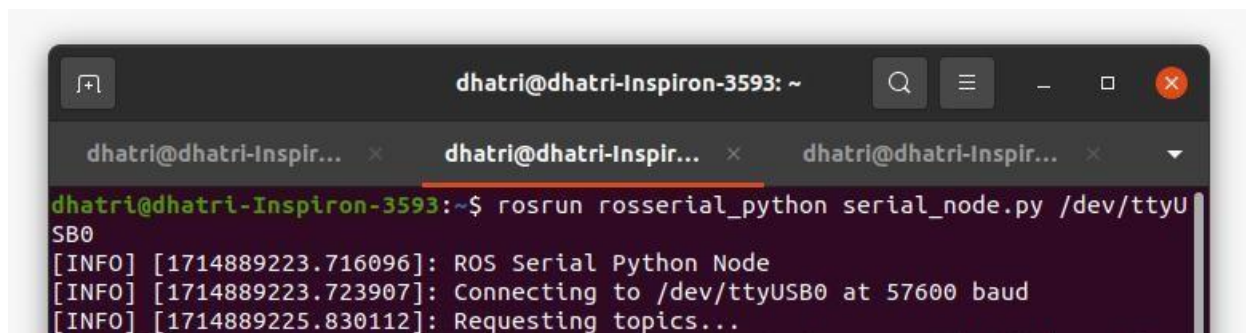
Overall, the robot's behavior is governed by these distance measurements, enabling it to autonomously navigate its environment while avoiding collisions.

Output



```
dhatri@dhatri-Inspiron-3593: ~  
dhatri@dhatri-Inspir... x dhatri@dhatri-Inspir... x dhatri@dhatri-Inspir... x  
SUMMARY  
=====  
PARAMETERS  
* /rostdistro: noetic  
* /rosversion: 1.16.0  
NODES  
auto-starting new master  
process[roscout-1]: started with pid [16498]  
ROS_MASTER_URI=http://dhatri-Inspiron-3593:11311/  
setting /run_id to ae4d9aa0-0aa5-11ef-a7fa-8b28d60ee4f2  
process[roscout-1]: started with pid [16508]  
started core service [/roscout]  
^C[roscout-1] killing on exit  
[master] killing on exit  
shutting down processing monitor...  
... shutting down processing monitor complete  
done
```

Fig 1: roscore



```
dhatri@dhatri-Inspiron-3593: ~  
dhatri@dhatri-Inspir... x dhatri@dhatri-Inspir... x dhatri@dhatri-Inspir... x  
dhatri@dhatri-Inspiron-3593:~$ roslaunch roscout roscout.launch  
SB0  
[INFO] [1714889223.716096]: ROS Serial Python Node  
[INFO] [1714889223.723907]: Connecting to /dev/ttyUSB0 at 57600 baud  
[INFO] [1714889225.830112]: Requesting topics...
```

Fig 2: roslaunch roscout roscout.launch

```
dhatri@dhatri-Inspiron-3593: ~  
dhatri@dhatri-Inspir... x dhatri@dhatri-Inspir... x dhatri@dhatri-Inspir... x  
data: -22  
---  
data: -22  
---  
data: -22  
---  
data: 14  
---  
data: 16  
---  
data: 15  
---  
data: -22  
---  
data: 12  
---  
data: 15  
---  
data: 13  
---  
data: 15  
---  
data: -22
```

Fig 3: rostopic echo chatter

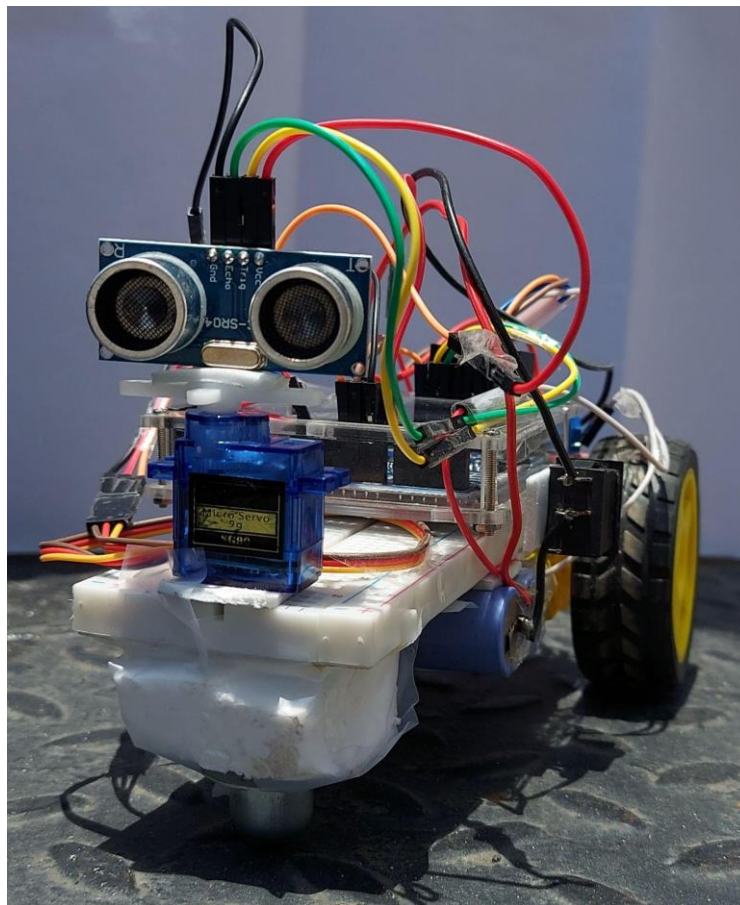


Fig 4: Final Robot Model