**INTRODUCTION**

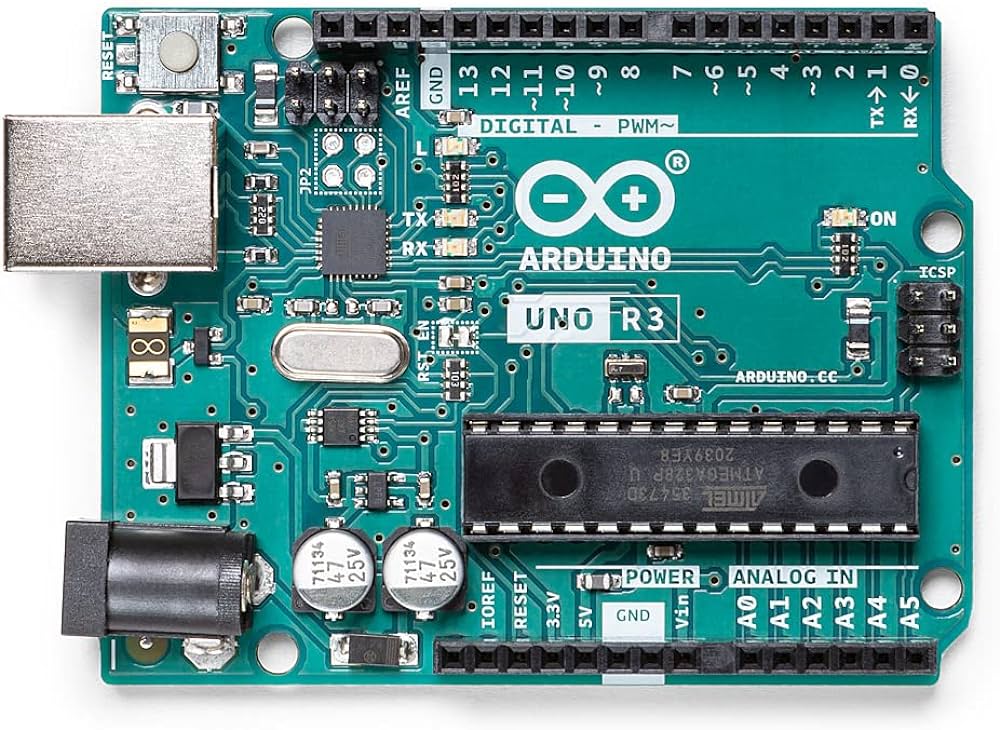
Radar systems, masters of long-range detection, are now augmented with the precision of ultrasonic sensing technology. While radar tracks objects from afar, ultrasonic sensors step in to halt movement upon close encounters, pinpointing obstacles and directing attention. a technology that detects objects with precision, halts at their presence, and points towards their direction. Seamlessly resuming from the same position, it navigates through obstacles with unparalleled accuracy, revolutionizing perception and ensuring safe traversal in complex environments.

**Components Required**

|  |  |  |
| --- | --- | --- |
| **S.no** | **Component Name** | **Quantity** |
| 1 | Arduino Uno | 1 |
| 2 | SG-90 Servo Motor | 1 |
| 3 | HC-SR04 Ultrasonic Sensor | 1 |
| 4 | Male to Female Jumper Wire | 4 |
| 5 | Male to Male Jumper Wire | 3 |
| 6 | 5mm Blue LED | 1 |
| 7 | Screw & Nut | 4 |

**Component Details**

1. **Arduino Uno**

Bottom of Form

Arduino Uno, a staple in the world of electronics prototyping, serves as a versatile microcontroller board for countless projects. With its user-friendly interface and extensive community support, it empowers enthusiasts and professionals alike to bring their ideas to life. Featuring a robust set of inputs and outputs, Arduino Uno offers flexibility for tasks ranging from simple LED blinking to complex robotics. Its open-source nature encourages innovation, making it an essential tool for beginners and seasoned makers alike.

1. **SG-90 Servo Motor**



Servo motors are compact, precise actuators commonly used in robotics and automation. They provide controlled angular movement with high accuracy, making them ideal for applications such as robotic arms, RC vehicles, and precision positioning systems. With their ability to maintain a set position, servo motors offer reliability and versatility in various mechanical and electronic projects.

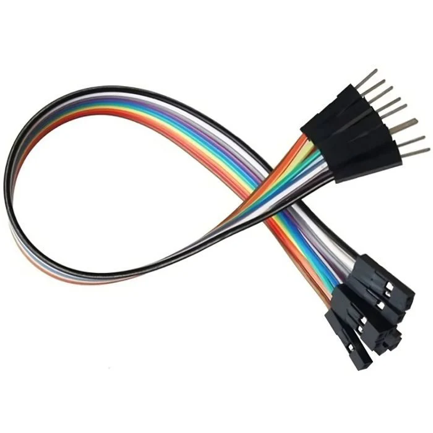
1. **HC-SR04 Ultrasonic Sensor**



The HC-SR04 ultrasonic sensor is a popular choice for distance measurement in electronics projects. Utilizing ultrasonic waves, it accurately detects objects within its range and provides distance information to microcontrollers like Arduino. With its simplicity and reliability, the HC-SR04 sensor finds applications in robotics, security systems, and smart devices, making it an essential component for hobbyists and professionals alike.Top of Form

1. **Male to Female Jumper wire**

Bottom of Form



Male to female jumper wires simplify connections between male pins on components like microcontrollers and female headers on sensors, aiding in hassle-free circuit assembly for electronic projects of all scales.

1. **Male to Male Jumper Wire**



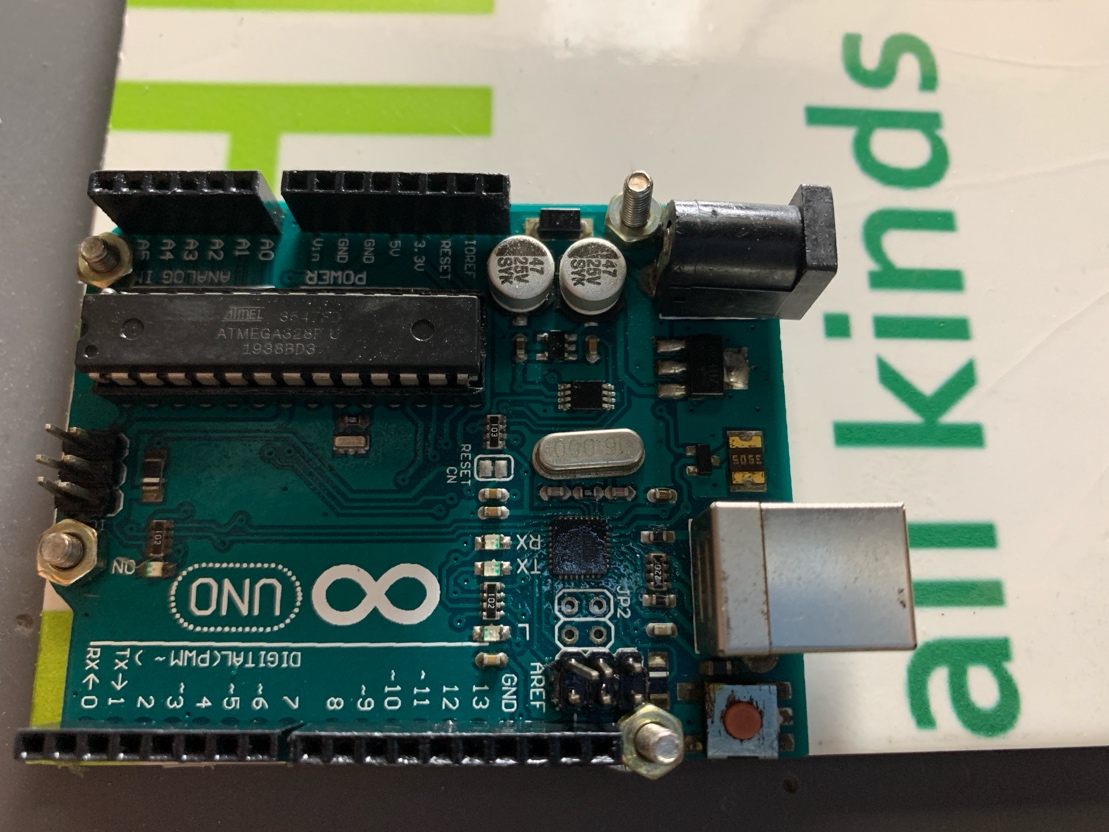
Male to male jumper wires enable direct connections between male pins on components, facilitating quick and reliable circuit assembly without the need for additional adapters. These versatile wires are essential for electronics projects, offering seamless integration and flexibility in prototyping and experimentation.

**Circuit Diagram**

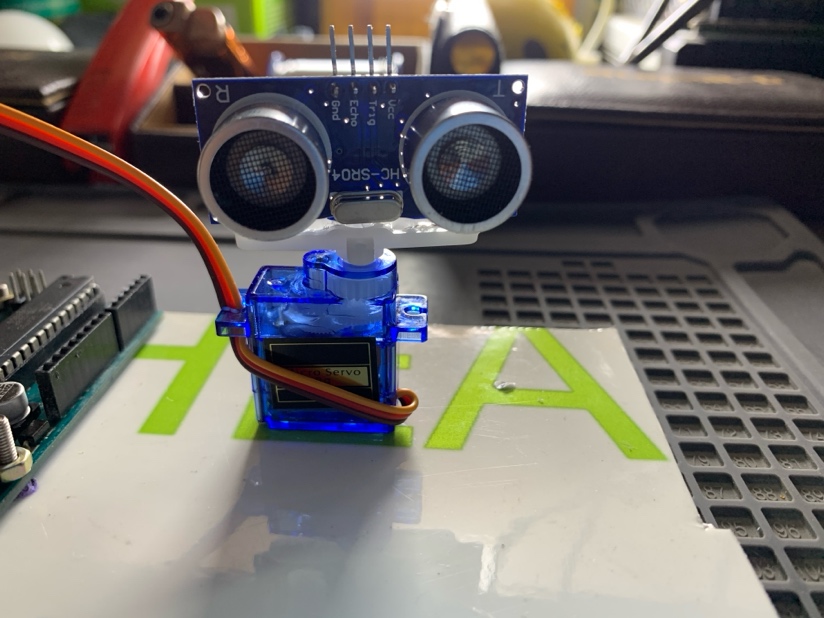
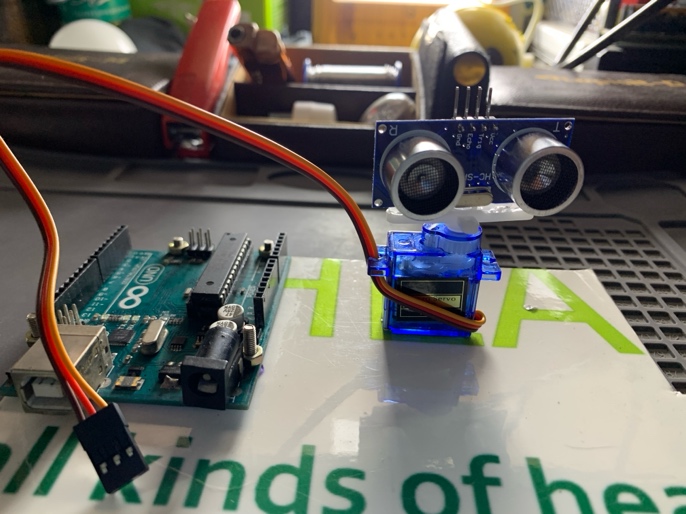


**Procedure**

1. Place Arduino Uno onto a board with the help of screws around it.



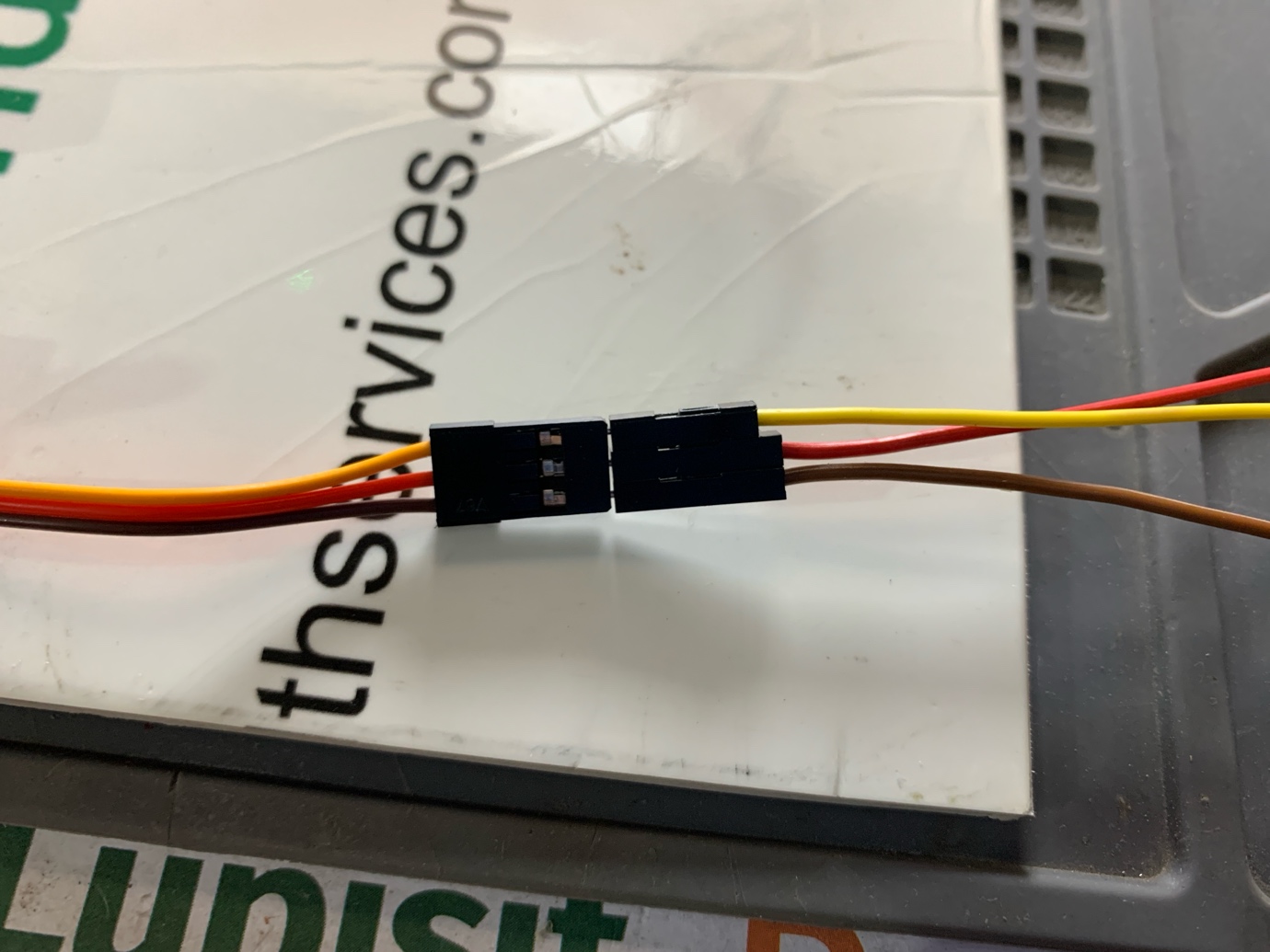
1. Glue the SG-90 Servo Motor and Ultrasonic sensor onto each other and then place them on the same board with Arduino Uno.

1. Place the Led on Arduino Uno using D13 (Positive side of LED) and GND pin next to it (Negative side of LED).



1. Using Male to Male Jumper wire connect the servo wires to 3 pieces of Male to Male Jumper Wire.



1. Using the connected Male Jumper Wire , Connect the SG-90 Servo Motor to Arduino Uno.

Red Wire -> VCC

Black Wire -> GND

Orange Wire -> D6 (Arduino Uno)



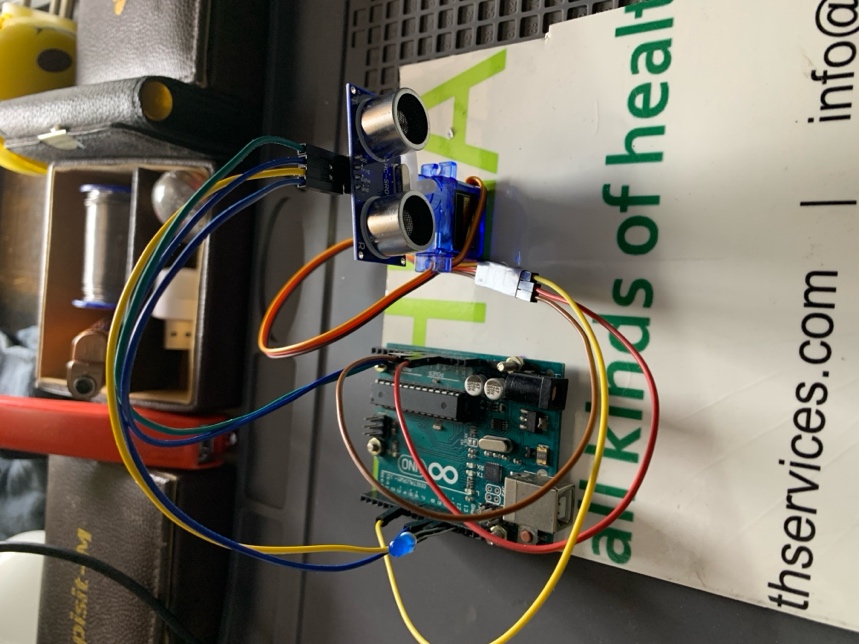
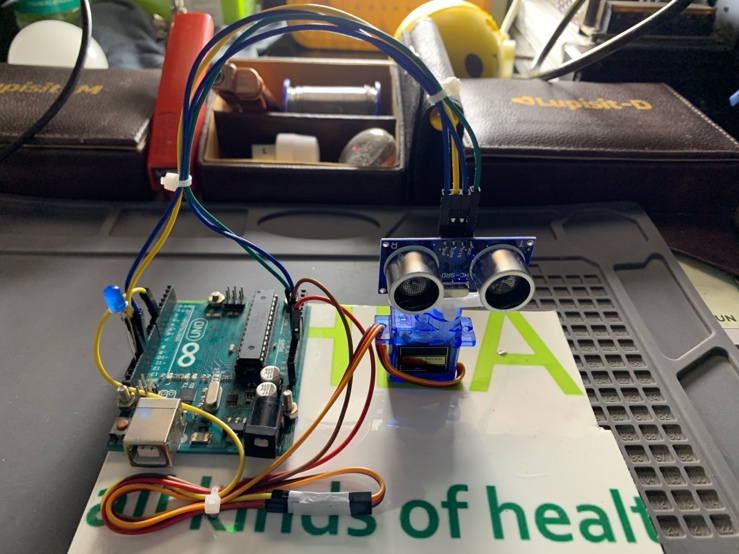
1. Using Male to Female Jumper Wire, Connect HC-SR04 Ultrasonic Sensor to Arduino Uno.

VCC -> 3.3V

GND -> GND

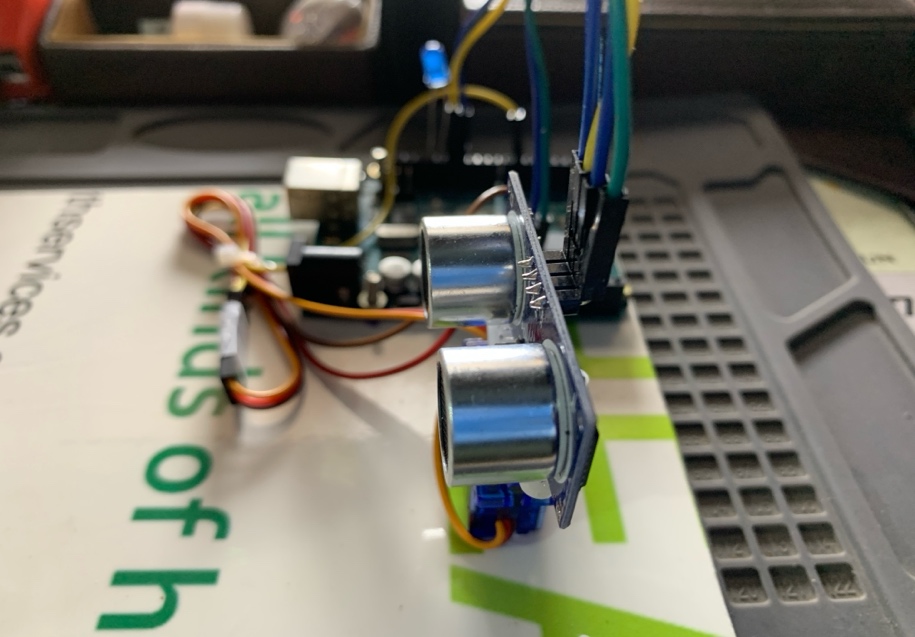
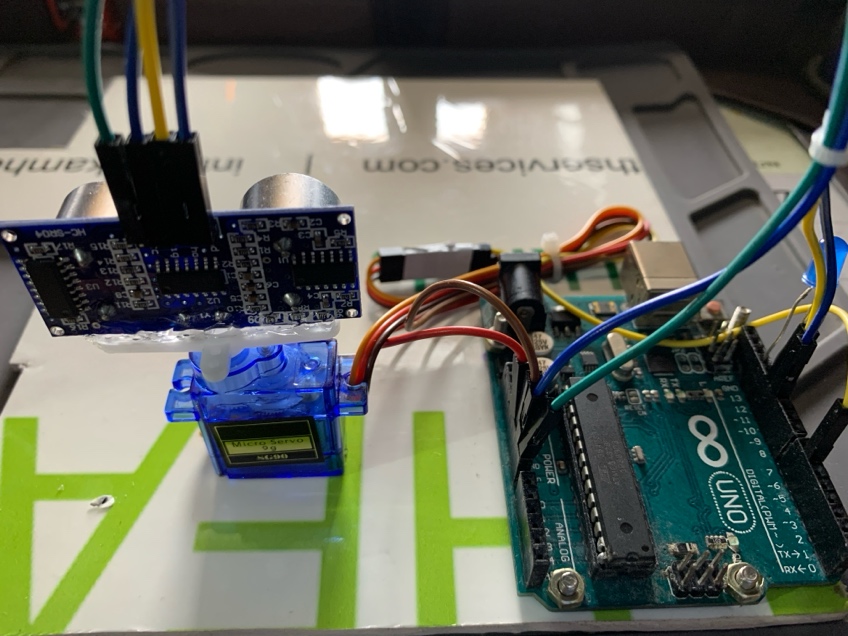
Trig -> D11

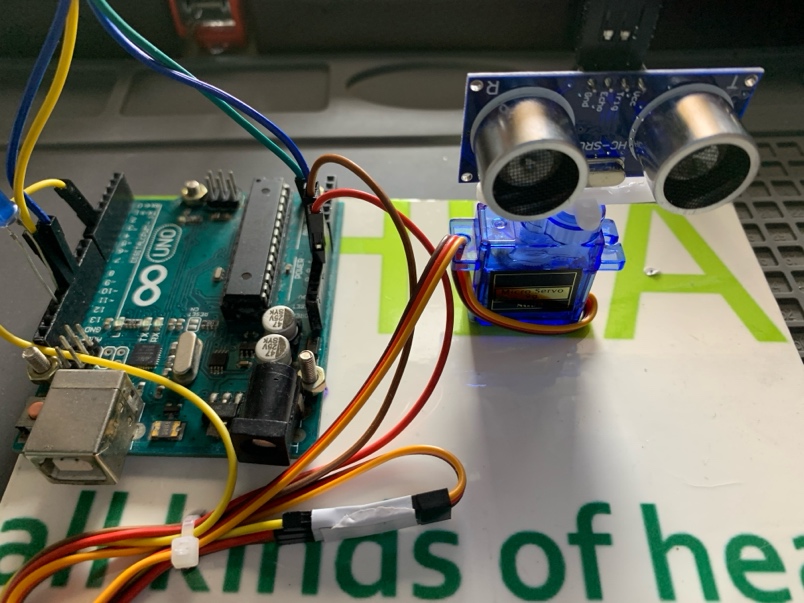
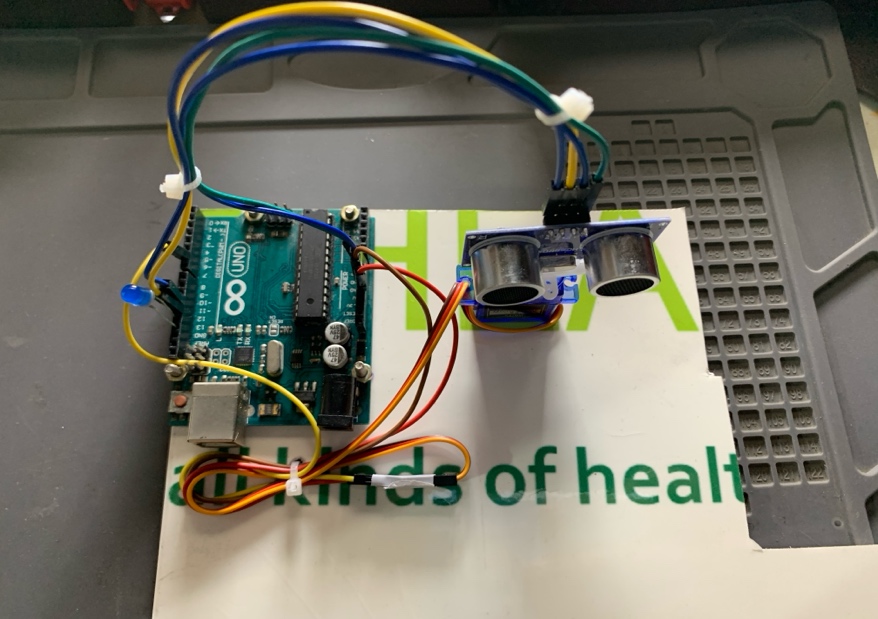
Echo -> D12



* **Operating Voltage:** 3.3v - 5v

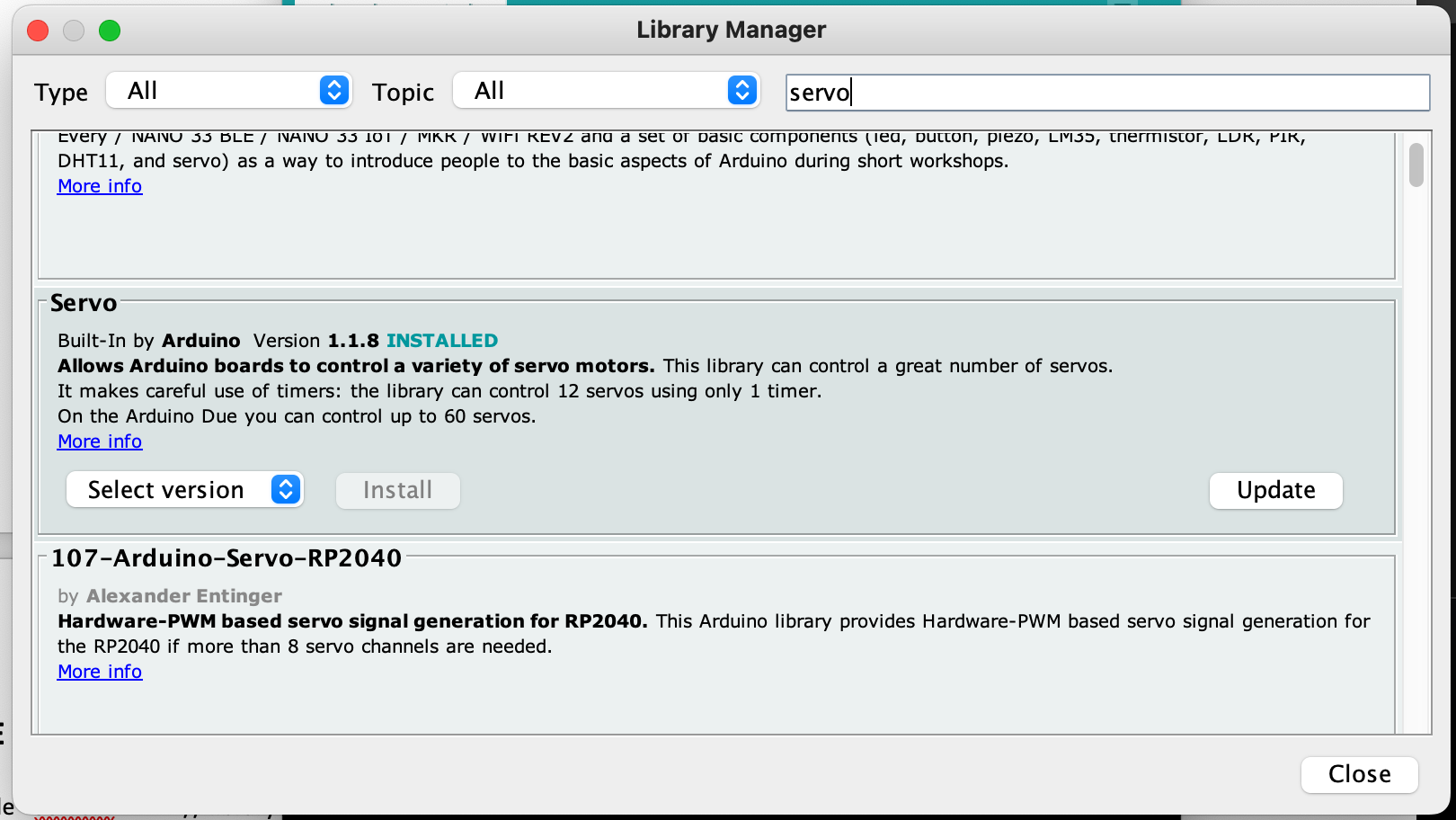
**More View**

**SOFTWARE:**

Installing Library

****

**CODE : -**

#include <Servo.h> // library to use servo motor

const int trigPin = 11; // trigger pin from ultrasonic Sensor

const int echoPin = 12; // Echo pin from ultrasonic Sensor

const int ledPin = 13; // Pin for controlling the light bulb

long duration; // duration for echo pin for ultrasonic sensor

int distance; // store the calculated distance from ultrasonic distance

bool flag=false; // condition

int i=0; // initalise for loop

int target\_distance=5; // distance at which u what to detect

Servo myServo;

void setup() {

pinMode(trigPin, OUTPUT);

pinMode(echoPin, INPUT);

pinMode(ledPin, OUTPUT); // Setting the LED pin as an output

Serial.begin(9600); // For Serial Monitor at the rate of 9600

myServo.attach(6); // signal pin from servo motor

digitalWrite(ledPin,LOW); // to keep the LED state off

}

void loop() {

**// for one half anti-clock cycle**

for(i=15;i<=200;i+=5)

{

flag=false;

distance=calculateDistance();

if(distance<target\_distance)

{

myServo.write(i);

digitalWrite(ledPin,HIGH);

delay(2000);

flag=true;

i=i-5; // to use the previous value of i

continue; // skip the next part of the code and go to next iteration

}

else

{

digitalWrite(ledPin,LOW);

flag=false;

}

if(!flag){ // if flag state is false then continue on its track

myServo.write(i);

delay(100);

}

}

**// one half clockwise rotation**

for(i=200;i>=15;i-=5)

{

flag=false;

distance=calculateDistance();

if(distance<target\_distance)

{

myServo.write(i);

digitalWrite(ledPin,HIGH);

delay(2000);

flag=true;

i=i+5; // to use the previous value of i

continue; // skip the next part of the code and go to next iteration

}

else

{

digitalWrite(ledPin,LOW);

flag=false;

}

if(!flag){ // if flag state is false then continue on its track

myServo.write(i);

delay(100);

}

}

}

int calculateDistance() {

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);

distance = duration \* 0.034 / 2;

return distance;

}