QUADRUINO ROBOT

INDEX

QUADRUINO ROBOT

Quadruino is a quadruped walking robot controlled by the Arduino microprocessor and with servo actuators.

A quadruped robot is a type of robot that has four legs and resembles the movement of a four-legged animal. These robots are designed to walk, run or even jump, and are used in a variety of applications, such as exploration in difficult terrain, aiding in rescues or simply as educational and research projects.

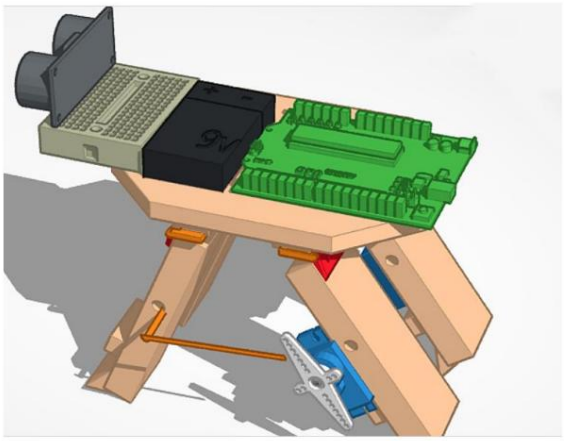
Quadruped robots can come in varying degrees of complexity, from basic models made from 3D-printed parts like this one to advanced robots that incorporate artificial intelligence technologies and sophisticated sensors.

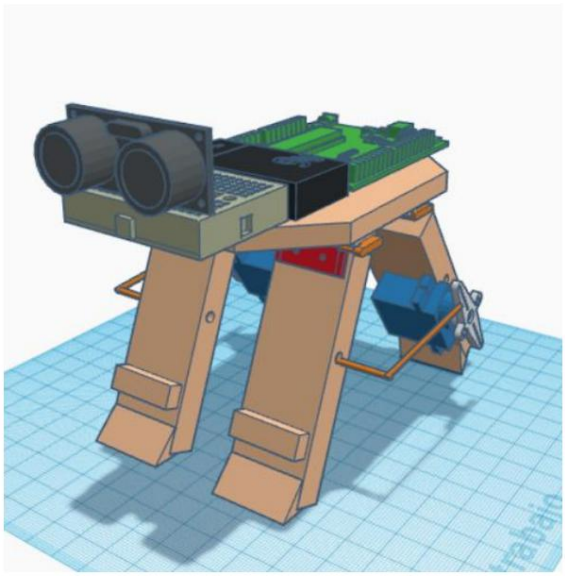
Arduino is an open source hardware platform consisting of an electronic board and an integrated development environment (IDE). It is designed to make it easy to create electronic and interactive projects in an accessible way for beginners and experts alike.
Arduino hardware consists of a board with a microcontroller and a series of input/output pins that allow the connection of sensors, actuators, and other electronic devices. The Arduino IDE provides a C/C++-based programming environment for uploading code to the board and controlling connected devices.

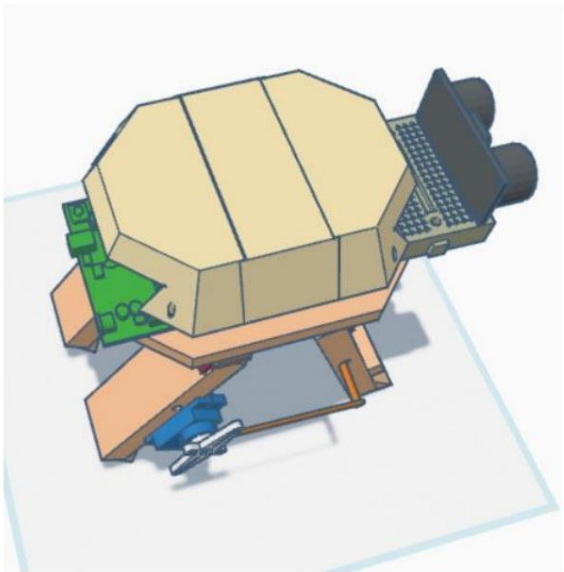
The combination of a quadruped robot and the Arduino platform could be an exciting project and spark interest in robotics and programming.
Using Arduino, you can control the movements and actions of a quadruped robot, and you can program it to perform various tasks.

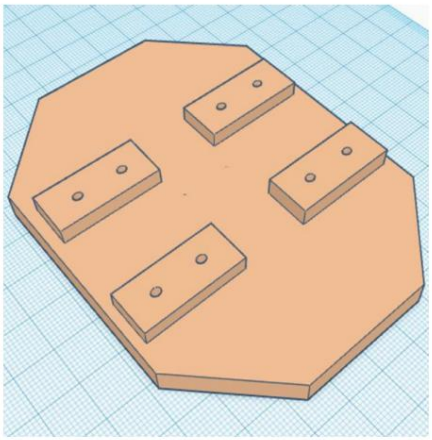
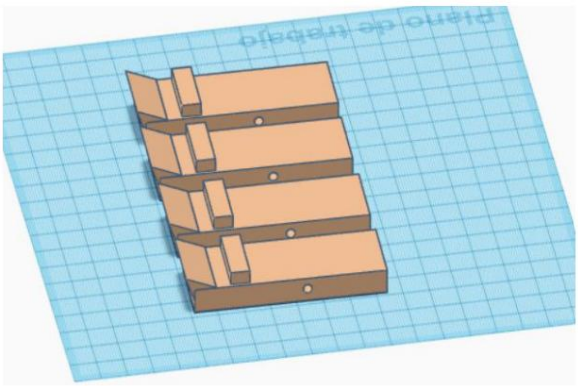
QUADRUINO MK2 (OVERALL DESIGN)

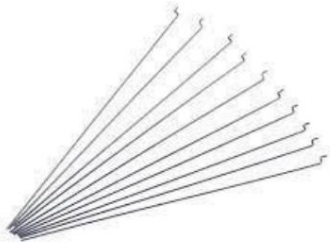
https://www.tinkercad.com/things/fZa6mmhZS58







COMPONENTS	
DENOMINATION	IMAGE
Quadruino Base (3D Printing) https://www.tinkercad.com/things/aKz1JOmdb5N	
Quadruino Legs (3D Printing) https://www.tinkercad.com/things/aKz1JOmdb5N	
Quadruino Shell https://www.tinkercad.com/things/3yrjjhPH EYt-quadruino-shell	

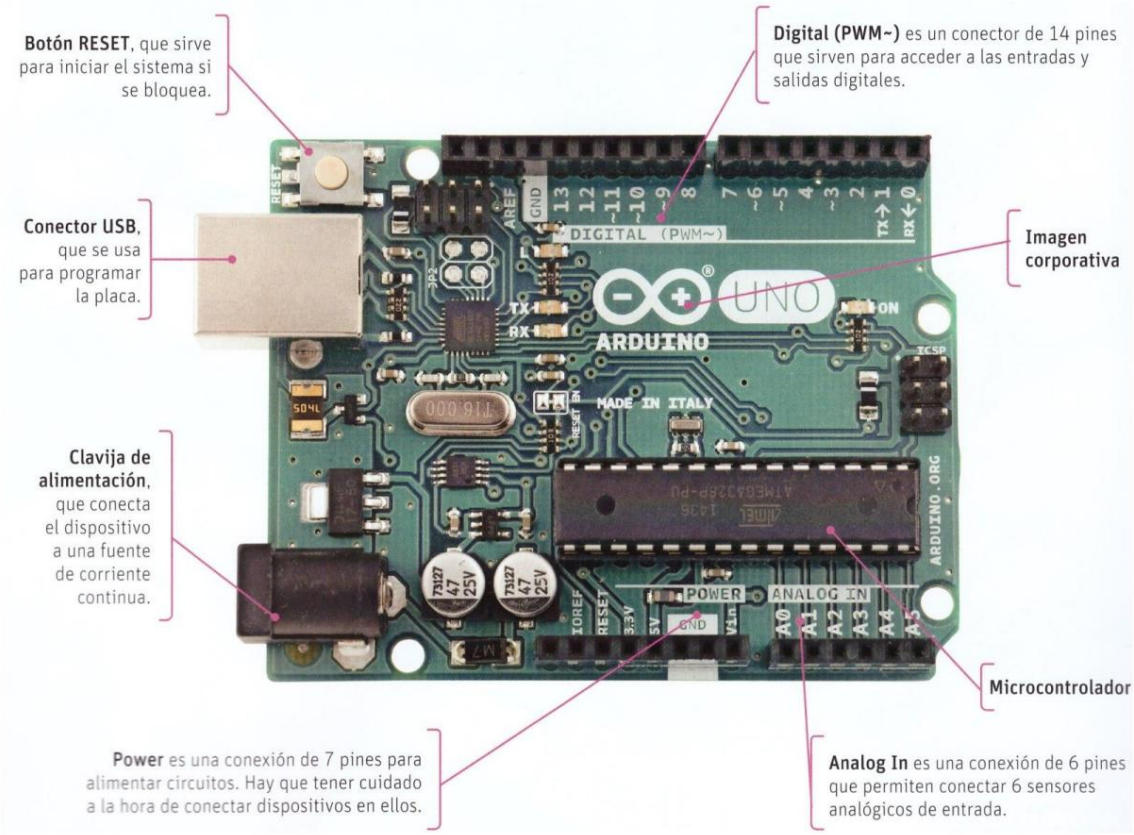
Mini Retro hinges €0.75 x 4	
10 Pieces Z Type Direction Gear Stick Diameter 1.2mm Length 20cm Stainless Steel Pull Rod RC Airplane Part (10 Pieces) €2.15 / lot (Bend the rod at a 90º angle and 9 x 5 cm segments)	
1pc SYB-170 Mini Solderless Breadboard Prototype 170 Breakout Points 35*47*8.5mm For DIY Kit €0.37x2	
Rc Mini Micro 9g 1.6KG Servo SG90 for RC 250 450 Helicopter Plane Car Boat for Arduino DIY with Bracket 360 degrees €0.91x2 free return Total:€3.82	

10/20/30/40 CM Dupont Line 40 Pin Male to Male + Male to Female & Female to Female Jumper Wire for Arduino DIY KIT Total: €2.95	
Original Leonardo R3 Microcontroller Development Board Atmega32u4 with Cable USB, Arduino Compatible, Starter Kit DIY Leonardo R3 Set €5.39x1	
10pcs 9V Battery Snap Connector Clip Lead Wires Holder Total: €2.65	
Panasonic Alkaline Power 6 LR 61 5410853039303 - 9 V battery €278	
Repair Screws, M1.2 M1.4 M2	

	
1pc HC-SR04 SR04 4Pin Ultrasonic Module Distance Measuring Transducer Sensor €1.09	
Hot Glue Gun Glue Gun Caliente Mini - Silicone Gun Crafts 20W Glue Gun 7mm*130mm 30 Sticks Glue Gun €11	

ARDUINO

Arduino is an open source electronics creation platform, which is based on free hardware and software that is flexible and easy to use for creators and developers. This platform allows the creation of different types of microcomputers on a single board to which the community of creators can give different types of use.



ARDUINO QUICK GUIDE

The card can be powered by an external 6 to 20 volt supply. if supplied with less than 7V, however, the 5V pin can supply less than five volts and the board may be unstable. If more than 12V is used, the voltage regulator can be overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

VIN. Power can be supplied via this pin, or, if power is supplied via from the power socket, access it through this pin.
This pin as output regulates the voltage to 5V. The board can be powered either from the DC power connector (7 – 12 V), the USB connector (5V), or by the VIN pin (7-12V). He voltage supply through the 5V or 3.3V pins does not go through the regulator, and can damage the board. We do not advise her.
c 3V3. A 3.3 volt supply generated by the on-board regulator. current consumption maximum is 50 mA.
GND. Ground pins.
IOREF. This pin on the Arduino board provides the voltage reference with which it operates the microcontroller.

Digital inputs and outputs: They are located at the top of the board, they go from 0 to 13, this last pin has an internal resistor included. The digital signal can be on or off (LOW or HIGH). Pins zero and one can be used to load the program on the board. For example, they are used to blink an LED or as an input, a button.

Analog outputs: They are pins 11, 10, 9, 6, 5 and 3, if you look closely they have a curved line next to them, They are called PWM (Pulse Width Modulation) outputs, which are actually digital outputs that they imitate analog outputs, modifying the separation between the different signal pulses. The PWM signal can give different values up to 255, they are used, for example, to vary the

intensity of an LED or operate a servo. It must be said that these pins work as digital inputs or outputs or as analog outputs.

Analog inputs: They are pins A0, A1, A2, A3, A4 and A5 (analog in). They are used for between a signal from an analog sensor, such as a potentiometer or a temperature sensor, which give a variable value. They can also be used as digital pins.

Power Pins:

- GND: They are the ground pins of the board, the negative.
- 5v: This pin supplies 5v
- 3.3v: This pin supplies 3.3v
- Vin: Input voltage, through this pin the board can also be powered.
- RESET: Through this pin you can reset the board
- IOREF: It is used for the board to recognize the type of power required by the shields

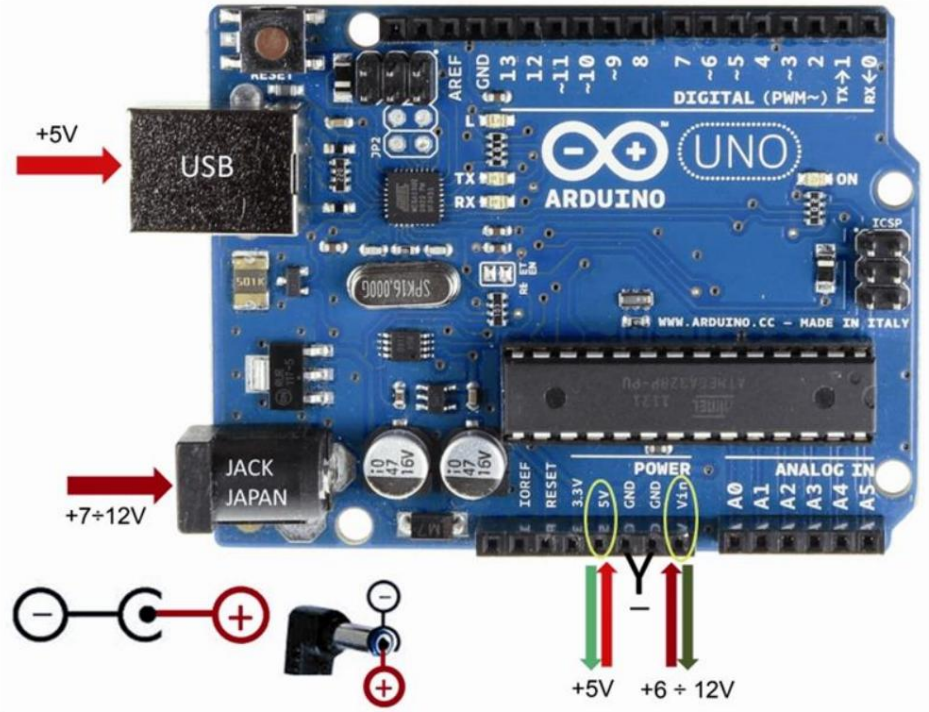
We can also find the AREF pin, above all to the left of the digital pins, this pin is used to supply a voltage other than 5v through the digital pins.

There is also the USB connector, to load the program and power the board; and the connector food, to feed her.

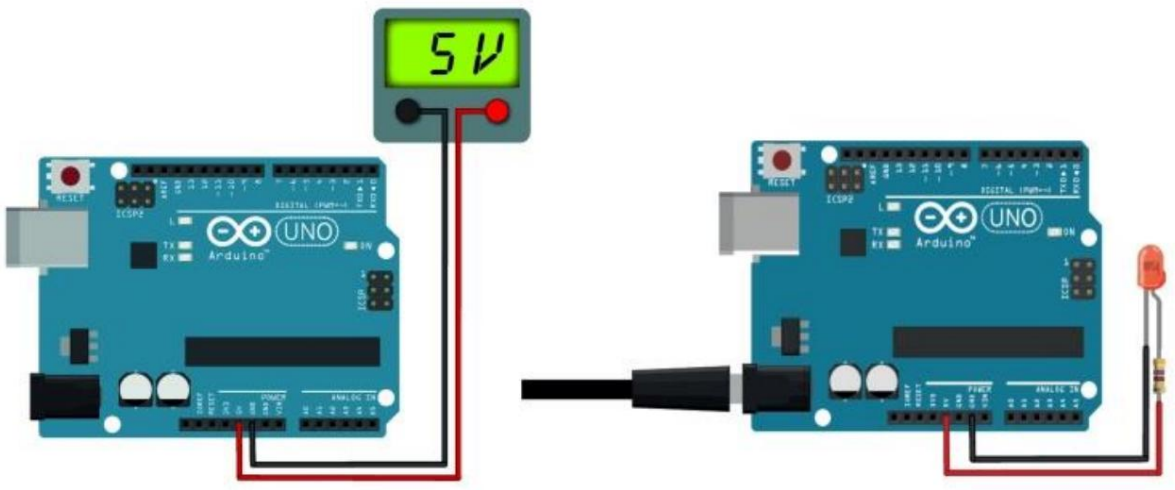
ARDUINO POWER

POWER SUPPLY OF THE ARDUINO

BOARD The following image summarizes the methods that we can use to power the Arduino UNO R3.

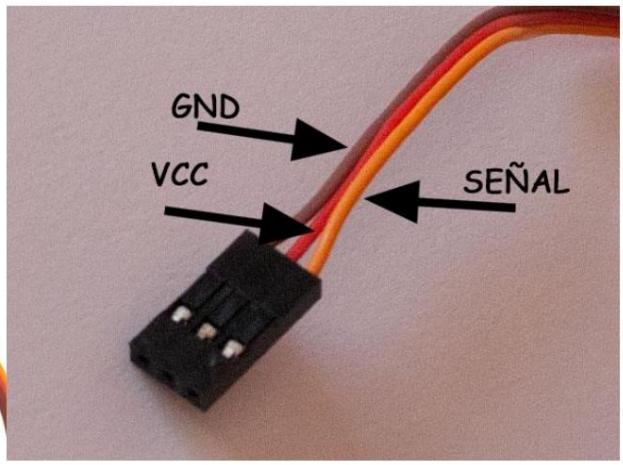


Jack for Arduino



9v DC battery clip connector to Jack connector for Arduino

SERVO CONNECTION DIAGRAM

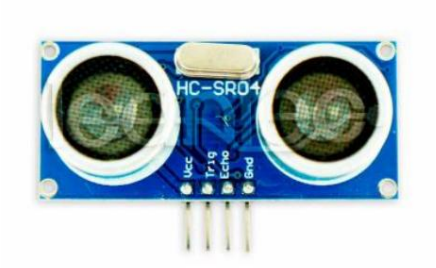


SERVO DIMENSIONS AND CONNECTIONS

To connect two servos to the Arduino board, you can follow the steps below:

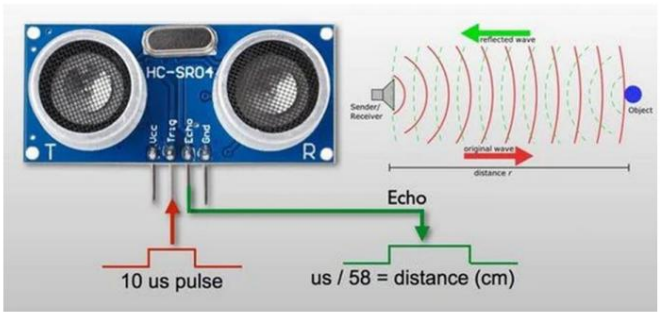
1. Identify the digital output pins that you will use on the Arduino board to control the servos. For example, you can use pins 9 and 10.
2. Connect the red (power) wire of one of the servos to pin 9 of the Arduino. Connect the red wire from the other servo to pin 10 on the Arduino.
3. Connect the black (ground) wire from both servos to the GND (ground) pin of the Arduino. Make sure all components share a common ground.
4. Connect the signal wires from the servos to the digital output pins corresponding on the Arduino. For the first servo, connect its signal wire (usually yellow or white in color) to pin 9. For the second servo, connect its signal wire to pin 10.

CONNECTION OF THE ULTRASOUND MODULE, HC-SR04



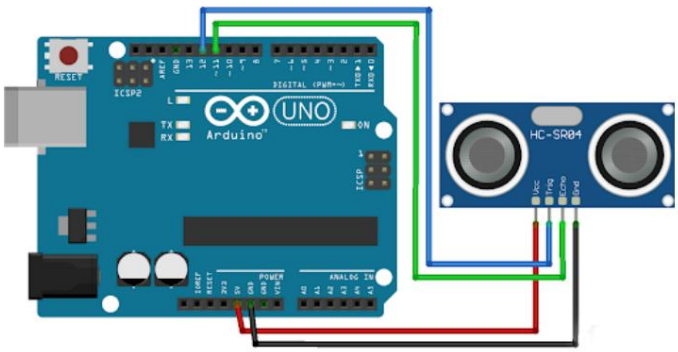
-Vcc: Power pin. (5V)
-Trigger: Trigger pin. This pin is an input, so in the control system, for example Arduino, it has to be connected to an output.
-Echo: This pin is a sensor output, so it must be connected to a control system input.

-Gnd: Negative power pin



The sensor has 4 pins. VCC and GND connect to the 5V and GND pins on the Arduino, and Trig and Echo connect to any digital pin on the Arduino. Using the Trig pin we send the ultrasonic wave from the transmitter, and with the Echo pin we listen to the reflected signal.

If we receive a reflected pulse, the Echo pin will go low before 38ms. Based on how long the Echo pin is in the HIGH state, we can determine the distance the sound wave has traveled, and therefore the distance from the sensor to the object.



BASIC CODES FOR QUADRUINO

Once you have made the physical connections, you can program the Arduino to control the servos using the Arduino "Servo" library. Here is a basic example of how to control two servos connected to pins 9 and 10

copy code
This program controls two servo motors connected to pins 9 and 10 of the Arduino Leonardo. In the main loop() loop, the servos are rotated sequentially in both directions. First servo1 moves from 0 degrees to 90 degrees and then back to 0 **SERVO TEST PROGRAM**

```
#include <Servo.h>

Servo servo1; // Object to control the first servo
servo servo2; // Object to control the second servo

int servo1Pin = 9; // Pin to which the first servo is connected
int servo2Pin = 10; // Pin to which the second servo is connected

void setup() {
  servo1.attach(servo1Pin); // Attach servo1 to the corresponding pin
  servo2.attach(servo2Pin); // Attach servo2 to the corresponding pin
}

void loop() {
  // Rotate servo1 clockwise
  servo1.write(90); // Write the angle of 90 degrees
  servo2.write(90); // Write the angle of 90 degrees
  delay(2000); // wait 1 second
  servo2.write(0); // Write the angle of 0 degrees
  servo1.write(0); // Write the angle of 0 degrees
  delay(2000); }        // wait 1 second
```

ULTRASOUND MODULE TEST PROGRAM Calculate distance

```
// Define the pins
const int trigPin = 12;
const int echoPin = 11;

void setup() {
  // Initialize the pins
  pinMode(trigPin, OUTPUT);
```

```
  pinMode(echoPin, INPUT);

  Serial.begin(9600); // Start serial communication }


void loop() {
  // Generate a 10 microsecond pulse on the Trig pin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Measure the duration of the pulse on the pin
  Echo long duration = pulseIn(echoPin, HIGH);

  // Calculate the distance in centimeters e=V 340 m/sx time float
  distance = duration          * 0.034 / 2;

  // Display the distance on the Serial Monitor
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println("cm");

  delay(1000); // Wait 1 second before doing the next measurement }
```

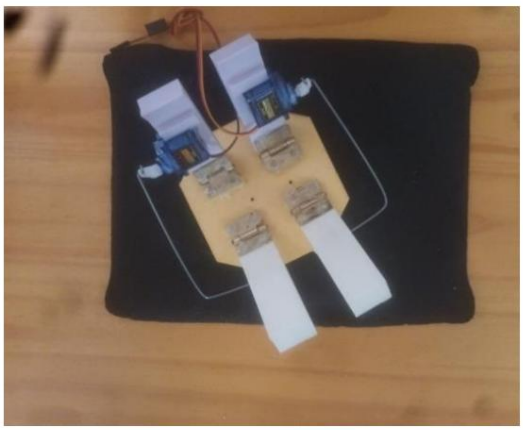
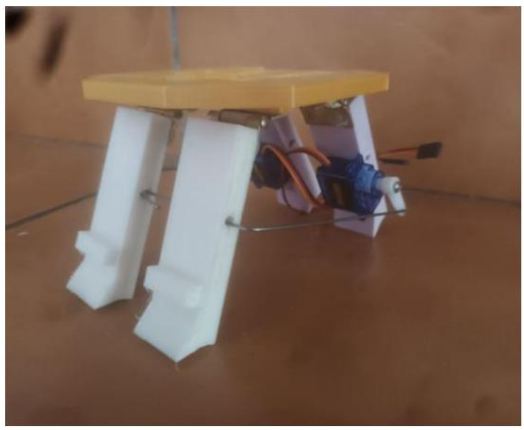
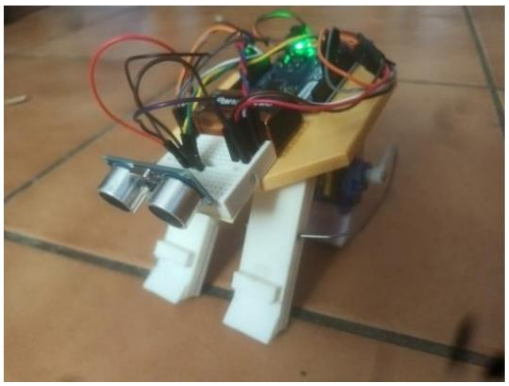
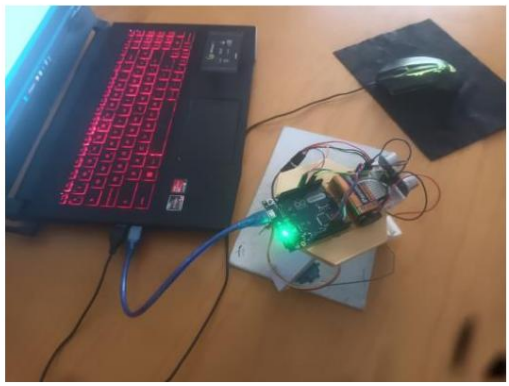
BASIC PROGRAM FOR QUADRUINO

This program uses the Arduino Servo library to control the two servo motors.
When the ultrasonic sensor detects an obstacle within 20 cm, servo 1 rotates between 0 and 180 degrees and servo 2 stops at its center position (90 degrees). If the object is more than 20 cm away, both servos rotate alternately between 0 and 180 degrees.
You can adjust the turning angles and delay times according to your needs.

```
#include <Servo.h>

// Define ultrasonic sensor pins const int
trigPin = 5; const int
echoPin = 6;

// Define servo motor pins const int
servoPin1 = 9; const int
servoPin2 = 10;

// Define detection distance const
int distanceThreshold = 20; // Detection distance in centimeters

// Create objects for the servo motors
Servo servo1;
servo servo2;

void setup() {
  // Initialize the ultrasonic sensor pins
```

```
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);

  // Initialize the servo motors
  servo1.attach(servoPin1);
  servo2.attach(servoPin2);

  // Set the servos to their home position
  servo1.write(90); // Initial position of servo1 (90 degrees)
  servo2.write(90); // Initial position of servo2 (90 degrees) }


void loop() {
  // Perform distance measurement long
  distance = getUltrasonicDistance();

  // Check if there is an obstacle within 20 cm if (distance
  < distanceThreshold) { // Rotate
    servo1 to the left (example: 45 degrees) servo1.write(45);
    delay(1000); //
    You can adjust the time of turning to the left

    // Stop servo1
    servo1.write(90);

    // Rotate servo2 to the right (example: 135 degrees)
    servo2.write(135);
    delay(1000); // You can adjust the turn time to the right

    // Stop servo2
    servo2.write(90); }
  else { //
    Both servos go forward (example: 0 degrees) servo1.write(0);
    servo2.write(0); } }



long getUltrasonicDistance() {
  // Generate a 10 microsecond pulse on the Trig pin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Measure the duration of the pulse on the pin
  Echo long duration = pulseIn(echoPin, HIGH);

  // Calculate the distance in centimeters
  long distance = duration * 0.034 / 2;

  return distance; }
```

IMAGES OF THE CONSTRUCTION

QUADRUINO VIDEOS

https://youtu.be/pNWBSWQYAWg

https://www.youtube.com/watch?v=TJQ5V3J5jA0