VOLUMEN 48
**XXXI Verano De la Ciencia**
ISSN 2395-9797
www.jovenesenlaciencia.ugto.mx

# Diseño de una interfaz para el control a distancia de un robot cuadrúpedo de ocho grados de libertad

Héctor Javier Estrada García[1], Lisseth Celeste Razo Vargas[2], Brandon Michelle Estrada Solis[1].

[1] División de Ingenierías, Departamento de Ingeniería Eléctrica, Universidad de Guanajuato
[2] Escuela del nivel Medio Superior de Salamanca, Colegio del Nivel Medio Superior, Universidad de Guanajuato
hestrada@ugto.mx
celeste.razo@ugto.mx
bm.estradasolis@ugto.mx

## Resumen

En este trabajo se presenta el desarrollo de un robot cuadrúpedo de laboratorio con ocho grados de libertad, diseñado como plataforma didáctica para la enseñanza de programación de dispositivos, control de sistemas y robótica móvil. El sistema está basado en una tarjeta Arduino Nano y emplea ocho servomotores tipo SG90/MG90, organizados en cuatro patas con dos articulaciones por extremidad. La estructura mecánica fue diseñada e implementada mediante manufactura aditiva utilizando piezas impresas en 3D, lo que facilita su reproducción, mantenimiento y expansión mediante la integración de sensores o módulos adicionales. Además, se añade un módulo de audio para reproducir sonidos. El control de motores se realiza con un modulo de expansión para Arduino nano, la activación de las rutinas se realiza a través de una app en Android y el envío de datos es a través de un módulo bluetooth.

**Palabras clave**: robot cuadrúpedo, Arduino Nano, servomotores, cinemática inversa, robótica educativa, impresión 3D.

## Objetivos

- Diseñar un robot móvil capaz de desplazarse sobre cuatro extremidades, que sea reprogramable para su **uso en prácticas de laboratorio**.
- Implementar control de servomotores para simular el movimiento de un cuadrúpedo.
- Diseño e impresión de las piezas del robot en 3D.
- Capacidad de expansión para añadir piezas de movimiento o sensores.
- Utilizar licencias de uso abierto o gratuito.
- Diseño de una interfaz gráfica de la aplicación móvil utilizando MIT App Inventor, considerando criterios de claridad y facilidad de uso.
- Integración con el microcontrolador mediante comunicación Bluetooth.
- Integración del módulo **DFplayer Mini Reproductor MP3,** para reproducir sonidos.
- Reproducibilidad.

## Materiales

- Controlador: Arduino Nano (Microcontrolador ATmega328P).
- Actuadores: 8 Servomotores SG90 (9g).
- Expansión: para Arduino Nano (facilita la conexión de VCC y GND para múltiples servos).
- Baterías de litio 18650 con base.
- Filamento PLD para impresión 3D.
- DFplayer Mini Reproductor MP3 y bocina de 3 W.
- Modulo Bluetooth HC-06.
- Interruptor sencillo para encender el prototipo.

**Piezas de impresión 3D.**

La impresión de las piezas se realizó por separado para reducir el tiempo de impresión y facilitar el diseño del modelo completo. Se muestran las imágenes de las piezas en 3D. Se utilizo un modelo de robot con licencia GNU, el cual está disponible en Marín Villalobos (s. f.). A este diseño se le añadieron base para las patas debido a su inestabilidad y deslizamiento al momento de caminar.

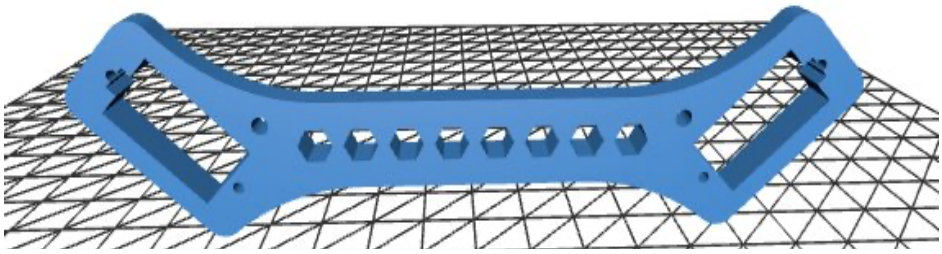


*Fig. 1 Chasis.*

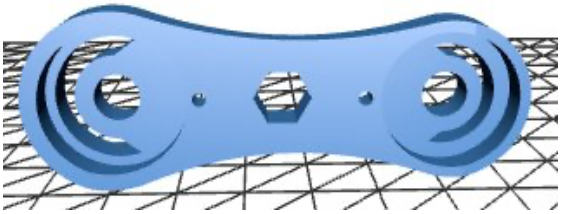


*Fig. 2 Fémur*

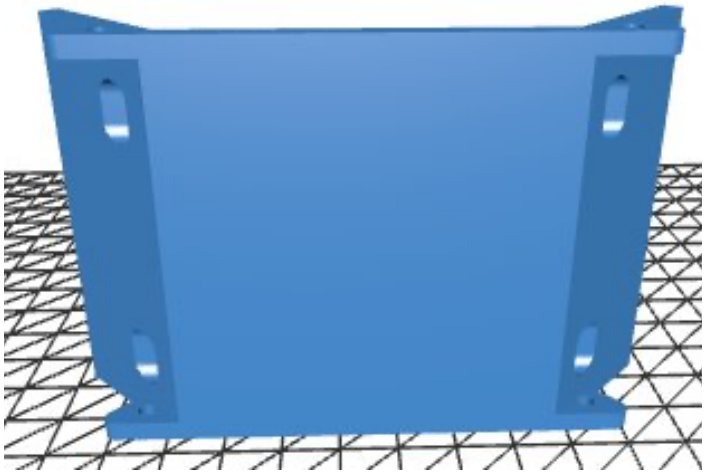


*Fig. 3 Base soporte*

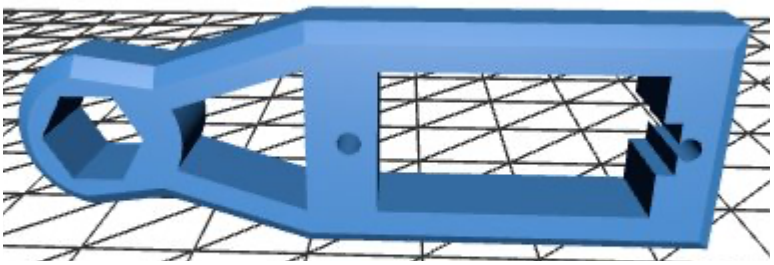


*Fig. 4 Tibia*

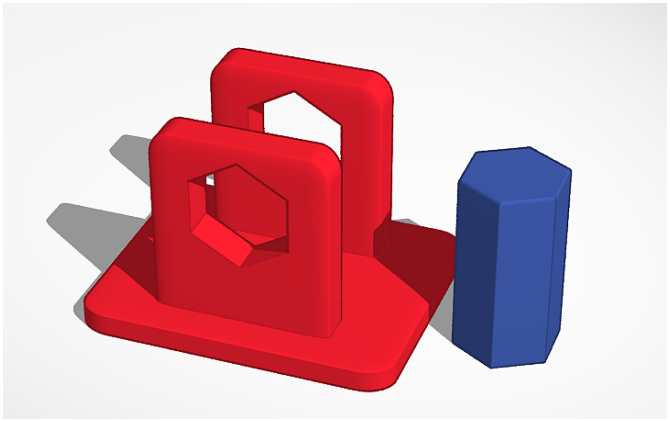


*Fig. 5 Base para las patas.*

## Funcionamiento

La tarjeta de expansión nos ofrece pines dedicados de voltaje y tierra para cada uno de los 8 motores, evitando tener que soldar o usar una protoboard. Con esta tarjeta podemos conectar una batería externa que alimente directamente a los motores, mientras el Arduino solo se encarga de enviar las señales de control. Si conectáramos los 8 motores directamente al Arduino sin la placa, probablemente se reiniciaría al intentar caminar o se podría ocasionar un daño al controlador.

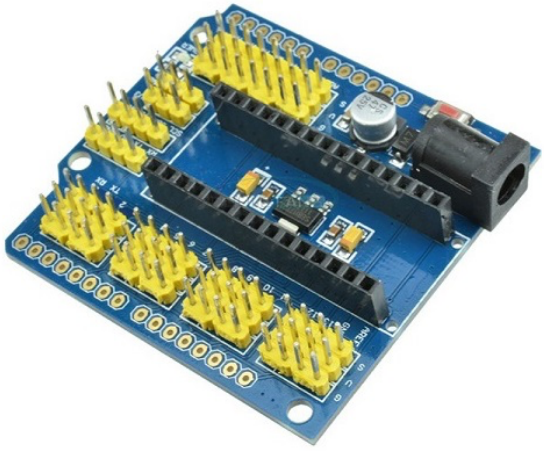


*Fig 6. Shield de expansión Arduino Nano.*

Para la alimentación se utilizan un par de baterías de litio 18650, las cuales tienen un voltaje de funcionamiento en paralelo de 7.5 V aproximadamente.

### Especificaciones baterías 18650:

- Dimensiones: 18mm (diámetro) x 65mm (largo).
- Química: Ión-Litio (Li-ion).
- Voltaje Nominal: 3.7V.
- Voltaje de Carga Máxima: 4.2V.
- Voltaje de Corte de Descarga: ~2.75V.
- Capacidad (mAh): entre 2200mAh y 3500mAh.
- Ciclos de Vida: >= 1000 ciclos (típico).
- Peso: ~45-50g.

### Patron de Caminata

Un patrón de marcha común y muy estable para robots de cuatro patas es la marcha estática o caminata lenta (similar al paso de un animal lento), que asegura que al menos tres patas estén siempre en contacto con el suelo para mantener la estabilidad estática (el Centro de Gravedad proyectado siempre cae dentro del polígono de apoyo).

A continuación, se muestra una secuencia de seis pasos para un movimiento hacia adelante, siguiendo la estrategia de levantar patas en diagonal opuesta, por ejemplo, Pata Delantera Derecha (PDD) y Pata Trasera Izquierda (PTI), luego Pata Delantera Izquierda (PDI) y Pata Trasera Derecha (PTD).

1. Posición Inicial (Apoyo 1-3-4)

   En esta posición, el robot está completamente apoyado en el suelo, listo para comenzar.
   - Apoyo en el suelo: PDD, PDI, PTD, PTI (Las 4 patas).
   - Acción: Ninguna. El robot está estático.

2. Primer Movimiento (Levantar 1)

   Se levanta una de las patas delanteras. Elegiremos la PDI.
   - Pata levantada (Fase de oscilación): PDI. La pata se levanta del suelo y avanza.
   - Apoyo en el suelo (Fase de apoyo): PDI, PTD, PTI. Estas tres patas soportan el peso y proporcionan el empuje inicial.
   - Resultado: El cuerpo del robot se desplaza ligeramente hacia adelante y hacia la izquierda.

3. Segundo Movimiento (Apoyo y Avance 1)

   La pata que estaba en el aire vuelve a apoyarse en el suelo, proporcionando un nuevo punto de apoyo más adelante.
   - Pata apoyada: PDI. Se apoya en una posición adelantada.
   - Apoyo en el suelo: PDD, PDI, PTD, PTI (Las 4 patas, momentáneamente).
   - Pata a levantar (próxima): Se prepara para levantar una pata diagonalmente opuesta, PTD.
   - Resultado: El robot tiene un apoyo estable en las 4 patas, con la PDI ligeramente más avanzada que el resto.

4. Tercer Movimiento (Levantar 2)

   Se levanta la segunda pata, siguiendo el patrón diagonal.
   - Pata levantada (Fase de oscilación): PTD. La pata se levanta del suelo y avanza.
   - Apoyo en el suelo (Fase de apoyo): PDD, PDI, PTI. Estas tres patas mantienen la estabilidad y empujan.
   - Resultado: El cuerpo del robot se desplaza hacia adelante y hacia la derecha.

5. Cuarto Movimiento (Apoyo y Avance 2)

La pata que estaba en el aire vuelve a apoyarse.

- Pata apoyada: PTD. Se apoya en una posición adelantada.
- Apoyo en el suelo: PDD, PDI, PTD, PTI (Las 4 patas, momentáneamente).
- Pata a levantar (próxima): Se prepara para levantar la PDD (el siguiente par diagonalmente opuesto es PDD y PTI).
- Resultado: El robot ha completado la mitad del ciclo de marcha.

6. Continuación del Ciclo

Los pasos 2 a 5 se repiten con el par de patas diagonal restante (PDD y PTI) para completar un ciclo de avance completo.

1. Levantar PDD y avanzar (apoyo en PDI, PTD, PTI).
2. Apoyar PDD y preparar la PTI.
3. Levantar PTI y avanzar (apoyo en PDD, PDI, PTD).
4. Apoyar PTI.

Este ciclo asegura que el robot avanza de forma estable manteniendo siempre tres puntos de apoyo, rotando el grupo de patas de apoyo.

## Diagrama de Conexión

El diagrama de conexiones muestra como deben de ser conectados los modulos y motores a la tarjeta de expansión nano shield, para la conexión de los motores tambien es necesario que cada motor tenga su propio cable de alimentación Vcc Gnd, el cual no se muestra en la figura para mejorar la visibilidad de conexiones.

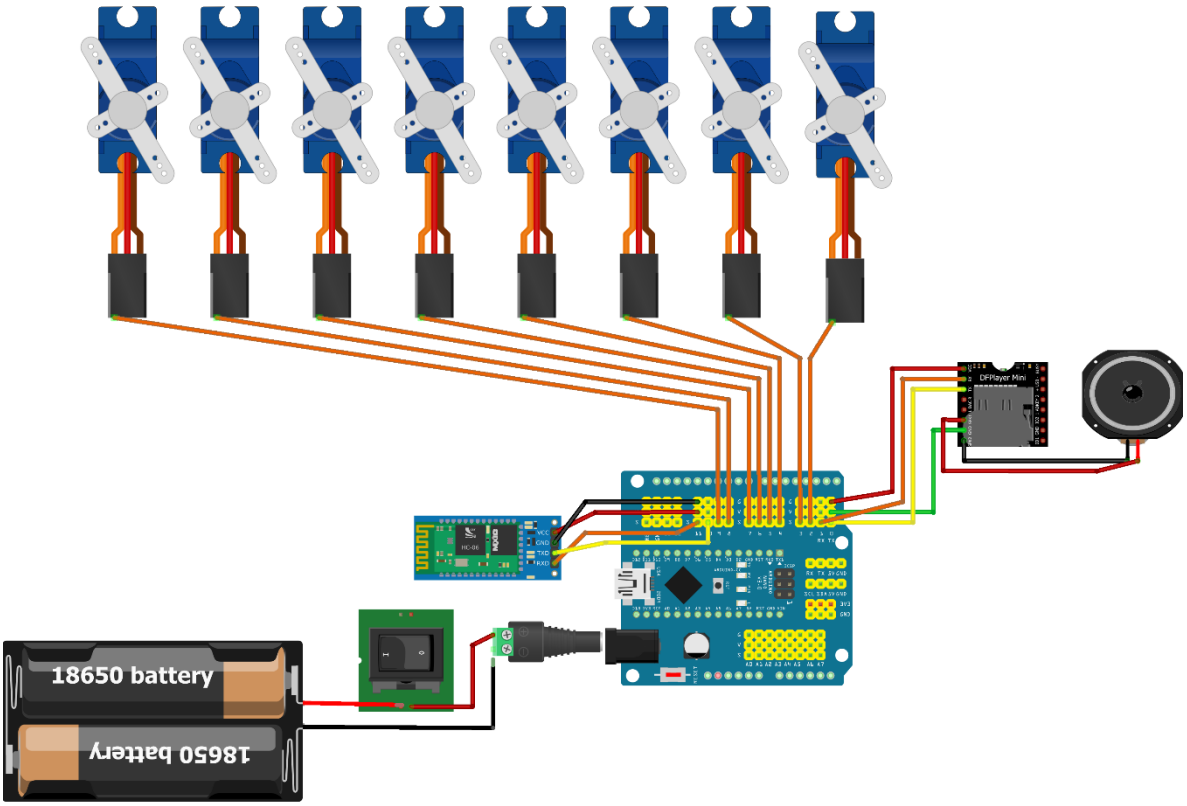



*Fig 7. Diagrama de conexión de motores y baterías.*

**Código Arduino para Control de 8 Servos.**

```
#include <Servo.h>
Servo servos[8];
int zPos[8] = {92, 83, 80, 91, 92, 84, 88, 80}; //FFL, TFL, FFR, TFR, FRR, TRR, FRL, TRL
//femur a la izquierda incrementa ángulo,
void setup() {
  for (int i = 0; i < 8; i++) { //Asignamos servomotores a pines del Arduino
    servos[i].attach(i + 2);    //Iniciamos con el pin 2 hasta el 10
    delay(20); }
  for (int i = 0; i < 8; i++) {  //Asignamos posiciones iniciales a los servomotores
    servos[i].write(zPos[i]);
    delay(30); }
  delay(2000);}

void loop() {
  // Movimiento básico: adelantar patas cruzadas, luego las otras
  pasoAdelante();delay(10);
  pasoAdelanteD();delay(10);}
void pasoAdelante(){
  for(int i=0;i<7;i++)
  { for (int j = 0; j < 4; j++)
    {zPos[j] -= 5; // Los primeros 4 disminuyen}
    for (int j = 4; j < 8; j++)
    {zPos[j] += 5; // Los últimos 4 aumentan}
    moverPierna(servos[0], servos[1], (zPos[0]), (zPos[1]));delay(8);
    moverPierna(servos[4], servos[5], (zPos[4]), (zPos[5]));delay(18);
    moverPierna(servos[2], servos[3], (zPos[2]), (zPos[3]));delay(8);
    moverPierna(servos[6], servos[7], (zPos[6]), (zPos[7]));  }
  delay(220);
  for(int i=0;i<7;i++)
  { for (int j = 0; j < 4; j++)
    {zPos[j] += 5; // Los primeros 4 aumentan}
    for (int j = 4; j < 8; j++)
    {zPos[j] -= 5; // Los últimos 4 disminuyen}
     moverPierna(servos[6], servos[7], (zPos[6]), (zPos[7])); delay(8);
```

```
    moverPierna(servos[2], servos[3], (zPos[2]), (zPos[3])); delay(18);
    moverPierna(servos[4], servos[5], (zPos[4]), (zPos[5])); delay(8);
    moverPierna(servos[0], servos[1], (zPos[0]), (zPos[1]));
  }
  delay(220);
}
void pasoAdelanteD() {
  for(int i=0;i<7;i++)
  {for (int j = 0; j < 4; j++)
    {zPos[j] += 5; // Los primeros 4 aumentan}
    for (int j = 4; j < 8; j++)
    {zPos[j] -= 5; // Los últimos 4 disminuyen}
    moverPierna(servos[2], servos[3], (zPos[2]), (zPos[3])); delay(8);
    moverPierna(servos[6], servos[7], (zPos[6]), (zPos[7])); delay(18);
    moverPierna(servos[0], servos[1], (zPos[0]), (zPos[1])); delay(8);
    moverPierna(servos[4], servos[5], (zPos[4]), (zPos[5]));}
  delay(220);
  for(int i=0;i<7;i++)
  {for (int j = 0; j < 4; j++)
    {zPos[j] -= 5; // Los primeros 4 disminuyen}
    for (int j = 4; j < 8; j++)
    {zPos[j] += 5; // Los últimos 4 aumentan}
    moverPierna(servos[2], servos[3], (zPos[2]), (zPos[3])); delay(8);
    moverPierna(servos[6], servos[7], (zPos[6]), (zPos[7])); delay(18);
    moverPierna(servos[0], servos[1], (zPos[0]), (zPos[1])); delay(8);
    moverPierna(servos[4], servos[5], (zPos[4]), (zPos[5]));}
  delay(220);}
void moverPierna(Servo &femur, Servo &tibia, int angFemur, int angTibia) {
  femur.write(angFemur);
  tibia.write(angTibia);}
```

## Interfaz en teléfono inteligente Android(APP)

La aplicación de control del robot fue desarrollada utilizando la plataforma MIT App Inventor, un entorno de programación visual basado en bloques que facilita la creación de aplicaciones para dispositivos móviles Android (Massachusetts Institute of Technology, s. f.). La idea principal fue facilitar la interacción entre el usuario y el sistema robótico. Su distribución es sencilla y permite acceder de manera rápida a las funciones principales relacionadas con la locomoción, la ejecución de acciones predefinidas y la administración de la comunicación mediante Bluetooth.

En la parte superior de la pantalla se encuentra el indicador del estado de la conexión Bluetooth, el cual informa si existe comunicación entre el teléfono inteligente y el controlador del robot. Debajo de este elemento se ubican los controles de desplazamiento: el botón fwd (forward), representado por una flecha hacia arriba, permite ordenar el avance del robot, mientras que el botón back, identificado con una flecha hacia abajo, se utiliza para realizar el movimiento de retroceso.




*Fig 8. Captura de pantalla de la aplicación en Android.*

Se utilizan tres botones para ejecutar acciones previamente programadas: Saluda, Voz y Sentado. La opción Saluda hace que el robot realice una secuencia de movimientos con una de sus extremidades delanteras simulando un saludo a la vez que emite un sonido en la bocina. El botón Voz activa una respuesta asociada al módulo de audio o a un comportamiento interactivo previamente definido. Por su parte, la opción Sentado coloca al robot en una postura estable que simula la posición de sentado.

En la parte inferior de la pantalla se incorporan el escudo y el logotipo de la Universidad de Guanajuato como elementos de identidad institucional. Asimismo, se incluye un botón denominado Salir, cuya función es cerrar la aplicación de forma segura.

Durante el desarrollo de la interfaz se buscó privilegiar la simplicidad y la facilidad de uso, de modo que cualquier usuario pueda controlar el robot sin requerir conocimientos especializados en programación o robótica. La distribución de los controles permite ejecutar tanto los movimientos básicos de desplazamiento como las acciones predefinidas de manera intuitiva, lo que facilita el empleo del sistema en prácticas de laboratorio, actividades docentes, demostraciones tecnológicas y eventos de divulgación científica.

## Resultados

A continuación, se muestra el diseño armado del prototipo, donde se puede ver la accesibilidad de programación, portabilidad y su capacidad de expansión ya que el diseño de piezas es totalmente en 3D.

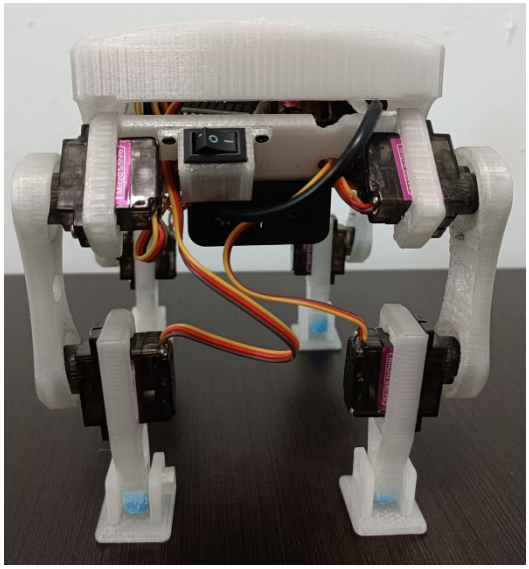


*Fig. 9 Vista trasera.*

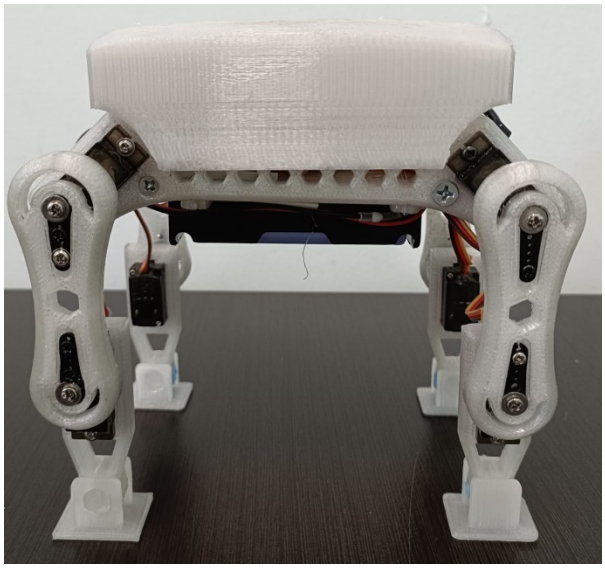


*Fig. 10 Vista lateral*

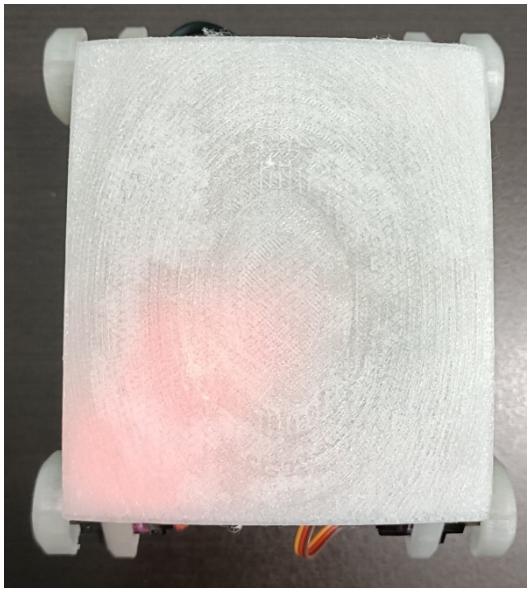


*Fig.11 Vista superior.*

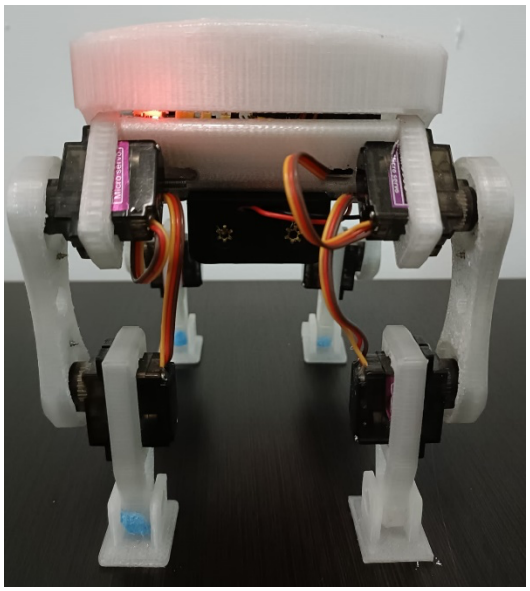


*Fig. 12 Vista frontal.*

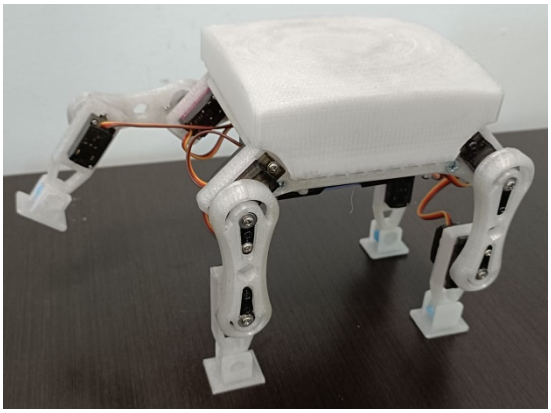


*Fig.13 Acción de saludo.*

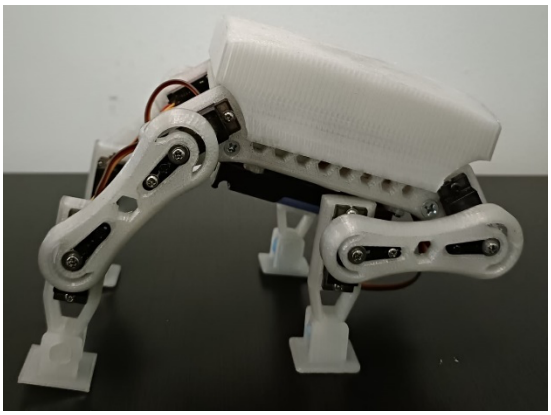


*Fig. 14 Acción de sentado.*

## Conclusiones

Se obtuvo un prototipo de laboratorio para el control de un robot cuadrupedo con 8 articulaciones, el cual se controla a través de una tarjeta Arduino Nano, cada articulación esta actuada por un servomotor MG90, los cuales son de bajo consumo de potencia. Este dispositivo además de mostrar el uso de tarjetas de desarrollo también sirve para familiarizarse con los robots de múltiples grados de libertad, así como el uso de cinemática directa e inversa para el movimiento de las patas. El prototipo pesa aproximadamente 1 kg, y sus medidas finales con las patas extendidas es de 12 cm x 14cm x 14cm (ancho, largo y alto) lo cual lo hace fácilmente transportable, además de que el uso de baterías permite utilizarlo sin necesidad de estar conectado a la red eléctrica.

## Bibliografía/Referencias

Margolis, M. (2020). Arduino cookbook (3rd ed.). O'Reilly Media.

Marín Villalobos, C. (s. f.). Aston [Código fuente]. GitHub. Recuperado el 4 de agosto de 2026, de https://github.com/cmarinv2005/Aston

Massachusetts Institute of Technology. (s. f.). MIT App Inventor [Software de computadora]. https://appinventor.mit.edu/

Monk, S. (2023). Programming Arduino: Getting started with sketches (3rd ed.). McGraw-Hill Education.

Niku, S. B. (2020). Introduction to robotics: Analysis, control, applications (3rd ed.). Wiley.

Spong, M. W., Hutchinson, S., & Vidyasagar, M. (2020). *Robot modeling and control* (2nd ed.). Wiley.