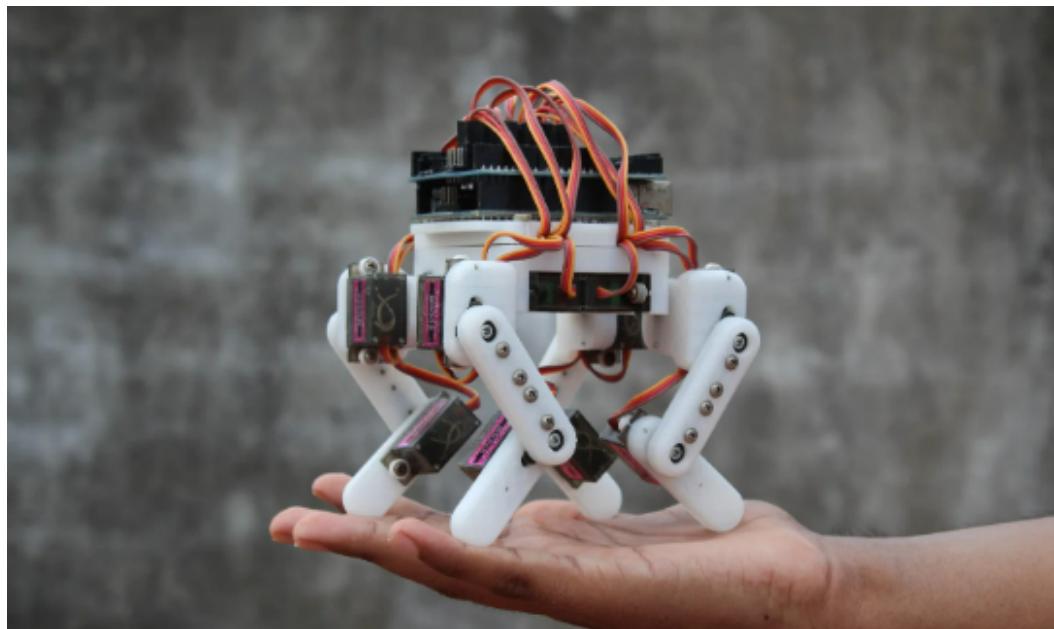


9 de Diciembre de 2024

## Taller de Proyecto II

# Informe Final Proyecto A3. Micro Quadruped.



### Grupo de Desarrollo

- ★ Gobbi, Leandro - 02955/1
- ★ Gonzalez, Martina Lucía - 01807/4

# 1. Descripción General del Proyecto

---

## Propuesta Original

El proyecto "Micro Quadruple" se centra en optimizar el control y el desplazamiento de un robot cuadrúpedo de pequeño tamaño. La propuesta inicial planteaba rearmar un prototipo existente para continuar la investigación del desplazamiento del mismo y agregar nuevas funcionalidades al proyecto. Objetivos principales:

- **Control remoto a través de un Access Point Wi-Fi:** Permitir la interacción con el robot sin depender de cables, facilitando el acceso y manejo desde cualquier dispositivo conectado a la red.
  - **Implementación de un sensor SW520D para detectar inclinaciones:** Este sensor permitiría al robot identificar desequilibrios, lo cual es esencial para mejorar su estabilidad y su capacidad de adaptarse a terrenos irregulares
- 

## Modificaciones

Durante el desarrollo, la cátedra sugirió varias modificaciones y ajustes técnicos para alinear el proyecto con los objetivos académicos:

- **Clarificación de términos técnicos:**  
Se solicitó una revisión del uso de conceptos como "tiempo real" para evitar confusiones en la evaluación y documentación. Este término se refinó para referirse específicamente a la detección inmediata y la respuesta automatizada del robot a cambios en su entorno.
  - **Cambio de plataforma:**  
Originalmente, se utilizaba un Arduino UNO como base del proyecto. Sin embargo, se recomendó migrar al NodeMCU (ESP8266), ya que este microcontrolador integra capacidades Wi-Fi, optimizando el manejo de la conectividad y reduciendo la complejidad del diseño electrónico.
  - **Cambio en el lenguaje utilizado:**  
El proyecto realizado por el grupo de alumnos anterior utilizó ReactJS para el Frontend del proyecto mientras que nosotros utilizamos HTML y CSS.
-

## Decisiones Tomadas Durante el Desarrollo

A medida que avanzó el desarrollo, se implementaron mejoras basadas en las necesidades detectadas. Entre las decisiones más destacadas se encuentran:

- **Implementación de una interfaz web:**

Se desarrolló una plataforma interactiva que permite visualizar el estado de los servomotores y el sensor SW520D. Esta interfaz facilita al usuario el control de las funciones del robot, incluyendo desplazamientos, posicionamiento e inclinación.

- **Control dinámico de inclinación:**

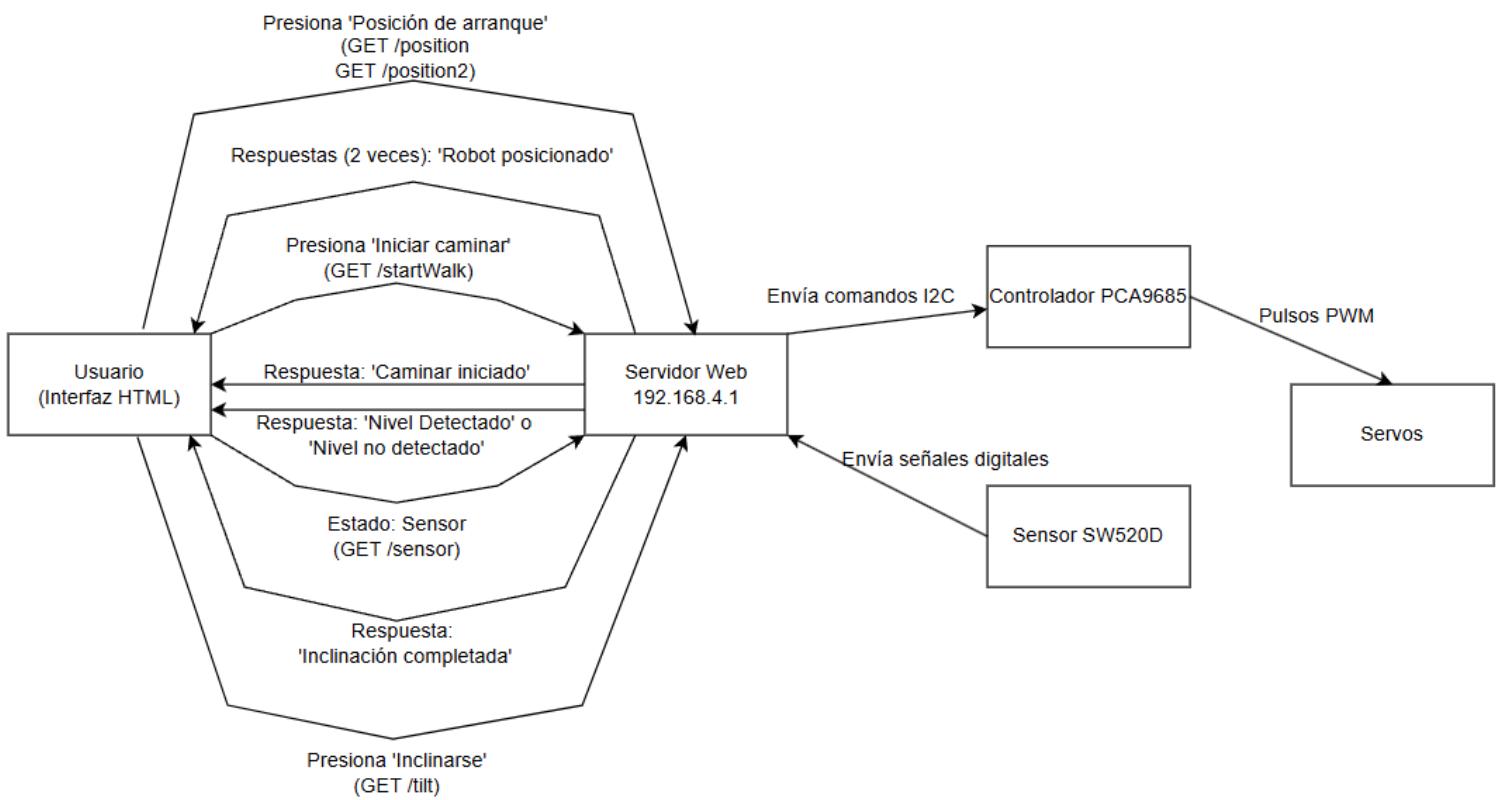
Se definió un comportamiento específico para manejar el equilibrio del robot. Cuando se presiona el botón correspondiente para iniciar la inclinación el robot comienza a des-nivelarse con las "patas" delanteras. Una vez que el sensor confirma el desequilibrio, el robot retorna automáticamente a su "posición inicial". Este mecanismo asegura que el robot sea capaz de adaptarse y estabilizarse sin intervención manual ante el estado de inclinación iniciada.

## 2. Presentación Esquemática del Proyecto

---

### Gráfico de procesos y funciones del proyecto

El gráfico que se muestra a continuación describe las interacciones entre los componentes principales del sistema desde el punto de vista del usuario hasta el control del hardware del robot.



### Descripción del flujo de trabajo:

**Usuario (Interfaz HTML):** El usuario interactúa con la interfaz web alojada en el MCU (NodeMCU ESP8266), presionando botones para realizar acciones como "Posición de arranque", "Iniciar caminar" o "Inclinarse". Cada acción del usuario genera una solicitud HTTP GET al servidor web en el MCU.

**Servidor Web (NodeMCU ESP8266):** Recibe solicitudes del usuario y responde con mensajes de estado, como "Robot posicionado" o "Caminar iniciado". Envía comandos I2C al controlador PCA9685 para mover los servos y ejecutar los movimientos correspondientes. Además, lee el estado del sensor SW520D mediante señales digitales para determinar si el robot está inclinado.

**Controlador PCA9685 y Servos:** El PCA9685 recibe comandos I2C del MCU y genera pulsos PWM para controlar los servos. Los servos realizan los movimientos del robot según las instrucciones recibidas.

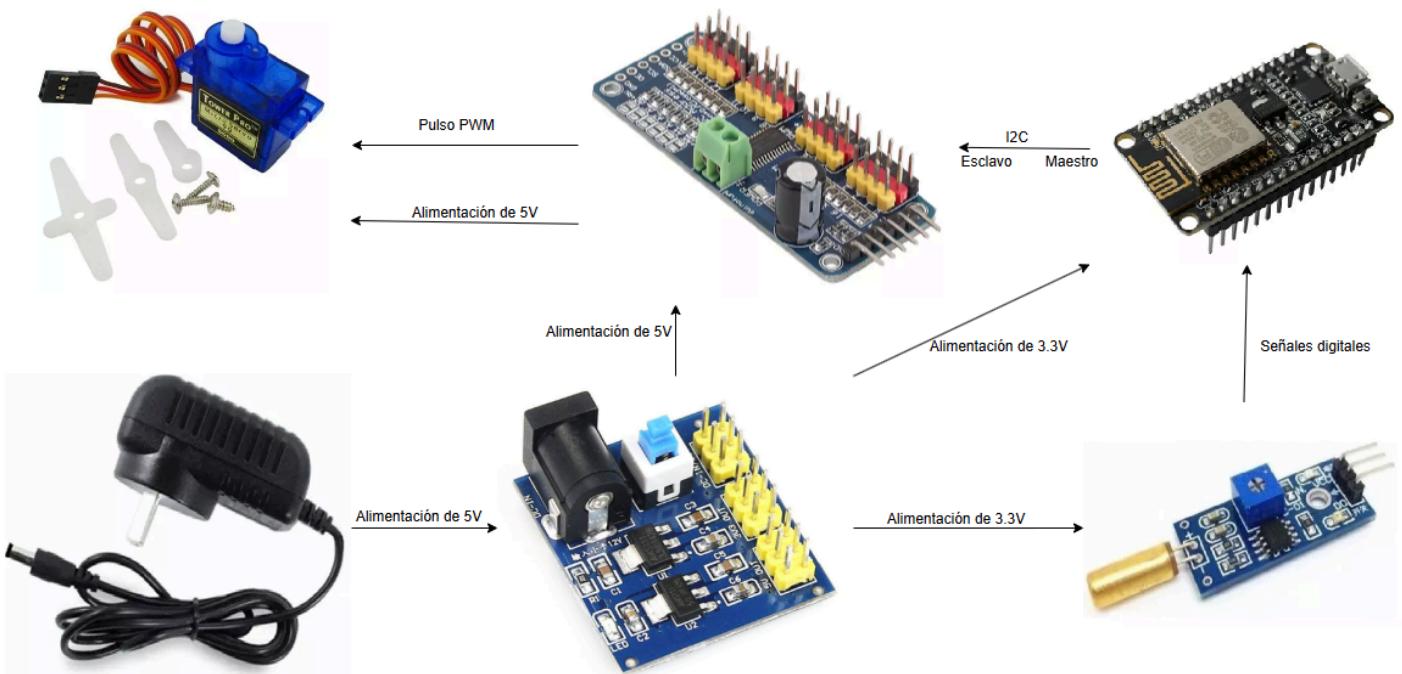
**Sensor SW520D:** Proporciona señales digitales al MCU para detectar la inclinación del robot. Esto permite que el sistema tome decisiones, como volver a una posición inicial cuando se detecta que el robot está desnivelado.

## Implementaciones realizadas por el grupo de desarrollo:

- Desarrollo del servidor web en el NodeMCU para manejar solicitudes HTTP.
  - Creación de la interfaz HTML para el control del robot.
  - Implementación de la lógica de control de movimiento e inclinación en el código del MCU.
  - Configuración del protocolo I2C para la comunicación con el PCA9685.
  - Integración del sensor SW520D para detectar inclinaciones.
- 

## Esquemático de conexiones del hardware utilizado

El esquema de hardware detalla las conexiones físicas y las relaciones entre los diferentes componentes del sistema.



## Descripción de los componentes y conexiones:

**Fuente de alimentación:** Alimenta al controlador PCA9685 y los servos con 5V.

Proporciona 3.3V para el NodeMCU ESP8266 y el sensor SW520D a través de un regulador de voltaje.

**NodeMCU ESP8266:** Se comunica con el PCA9685 a través del protocolo I2C para enviar comandos de control a los servos.

Recibe señales digitales del sensor SW520D para detectar inclinaciones.

**Controlador PCA9685:** Controla hasta 16 servos (en nuestro caso particular 12) utilizando señales PWM generadas a partir de los comandos recibidos del NodeMCU.

**Servos:** Realizan los movimientos físicos del robot cuadrúpedo, como caminar, inclinarse o volver a la posición inicial.

**Sensor SW520D:** Detecta inclinaciones mediante señales digitales, enviadas directamente al NodeMCU.

---

## Funcionalidades y requerimientos cumplidos

- **Cumplidos completamente:**

- Control del robot cuadrúpedo mediante una interfaz web.
- Implementación de una red WiFi mediante el NodeMCU para servir la interfaz al usuario.
- Control de servos a través del controlador PCA9685 y del NodeMCU utilizando el protocolo I2C.
- Integración del sensor SW520D para la detección de inclinaciones.

- **Cumplidos parcialmente:**

- Optimización de los movimientos del robot, que puede requerir ajustes adicionales en la lógica de control y los ángulos de los servos para mejorar la estabilidad.
- Representación dinámica en la interfaz de los estados de los servos: Resaltado dinámico de los servos activos en la interfaz web mediante texto actualizado periódicamente.

- **Agregados durante el desarrollo:**

- Implementación de una lógica específica para la inclinación del robot, que lo obliga a seguir inclinándose hasta que el sensor SW520D detecte que está nivelado.

# 3.Documentación del Software del Proyecto

---

## Organización del Código Fuente

El proyecto está organizado de forma modular, utilizando PlatformIO y Visual Studio Code, lo que facilita la gestión del código fuente y las dependencias. A continuación, se detallan las principales funciones implementadas, cómo cumplen con los requisitos del sistema y los métodos utilizados para validar su funcionalidad.

### Estructura de directorios:

Carpeta/Archivo	Descripción
platformio.ini	Archivo principal de configuración de PlatformIO. Contiene los parámetros de plataforma, librerías utilizadas y configuración de compilación.
src/main.cpp	Archivo principal del proyecto que implementa todas las funcionalidades del robot cuadrúpedo, incluyendo el control de los servos, la comunicación con el servidor web y el manejo del sensor SW520D.
lib/	Directorio destinado a librerías privadas específicas. No se utilizó en este caso
include/	Carpeta opcional para declarar encabezados, si se separan funciones adicionales. No se utilizó en este caso.
.pio/	Carpeta generada automáticamente por PlatformIO para la compilación del código.

---

## Instalación y Configuración del Entorno de Desarrollo

Para la instalación y configuración del entorno se comenzó por importar el proyecto del grupo anterior ([GitHub](#) - [tpII/2022-A.3-Micro-Quadruple: 2022](#)

[A.3-Micro-Quadruple](#)), a partir de este se decidió continuar el uso de la misma plataforma, Visual Studio en conjunto con la herramienta PlatformIO.

Para lograr utilizar PlatformIO, luego de instalar la extensión, se configuró de la siguiente forma el archivo 'platformio.ini' de la siguiente manera:

```
[env:nodemcu2]
platform = espressif8266
board = nodemcu2
framework = arduino
monitor_speed = 115200
lib_deps =
    esphome/ESPAsyncWebServer-esphome@^3.2.2
    esphome/ESPAsyncTCP-esphome@^2.0.0
    adafruit/Adafruit PWM Servo Driver Library@^2.4.0
```

Donde se especificó la plataforma, en nuestro caso el ESP8266, la placa 'nodemcu2', el framework 'arduino' y el BaudRate de 115200.

Además, en este archivo se configuran las librería utilizadas para el proyecto:

- **ESPAsyncWebServer:**

- Esta librería permite la creación de un servidor web asincrónico en ESP8266. Permite manejar solicitudes simultáneamente sin bloquear la ejecución del programa principal. Es el encargado de la gestión de solicitudes HTTP (GET en nuestro caso).

- **ESPAsyncTCP:**

- Es una librería base requerida por ESPAsynwbServer. Proporciona las herramientas necesarias para manejar conexiones TCP asincrónicas en el ESP8266. Actúa como el núcleo y permite la comunicación entre entre el servidor web y el/los clientes.

- **Adafruit PWM Servo Driver:**

- Librería encargada de facilitar la comunicación de servos por medio del PCA9685 mediante el protocolo I2C. Permite controlar hasta 16 salidas PWM (en nuestro caso solo se necesitan 12). Se encarga de generar las señales PWM precisas para controlar los servomotores, configurando la frecuencia y el ancho del pulso PWM.

---

**Configuración WiFi:** La placa NodeMCU está configurada como Access Point (AP), lo que permite que otros dispositivos se conecten directamente al microcontrolador mediante WiFi, sin necesidad de un router externo. Para la configuración del mismo se tuvo en cuenta:

- **Configurar las credenciales del AP** mediante el comando 'WiFi.softAP(ssid, password)'
    - SSID: "NodeMCU\_AP"
    - Contraseña: "12345678"
  - **Obtención de la dirección IP del Access Point:** La línea WiFi.softAPIP() obtiene la dirección IP asignada al Access Point, la cual es 192.168.4.1. Esta dirección IP se utiliza como entrada en el navegador web del usuario para acceder a la interfaz web y controlar el robot.
  - **Servidor Web Asíncrono:** Utilizando la librería ESPAsyncWebServer, el microcontrolador ejecuta un servidor web en el puerto 80. Este servidor gestiona las solicitudes HTTP (GET) enviadas desde el cliente (usuario) para realizar acciones como mover el robot, cambiar su posición inicial o consultar el estado del sensor SW520D, etc.
- 

### Funciones Implementadas en el Código

El archivo principal del proyecto 'main.cpp' implementa las siguientes funciones para controlar el comportamiento del robot:

Función	Objetivo
initPCA9685()	Inicializa el controlador PCA9685 y establece la frecuencia de los pulsos PWM de 60Hz para los servos. Se llama al inicio en setup() para inicializar el hardware del controlador de servos.
moveServo(uint8_t, uint16_t)	Mueve un servo específico a una posición determinada, indicando el ángulo o posición mediante señales PWM.

position() y position2()	Lleva el robot a la posición inicial. position() para acomodar las patas delanteras y position2() para acomodar las patas traseras.
walkSequence()	Ejecuta la lógica de caminata del robot. Divide el movimiento en pasos (sequenceStep) para mover cada pata en el orden correcto y realiza hasta 3 secuencias del mismo.
tiltRobot()	Inclina el robot hasta que el sensor SW520D detecte que está nivelado.
setup()	Configura el Access Point, el servidor web y el PCA9685.
loop()	Ejecuta el estado actual del robot, como caminar o inclinarse.
server.on()	Define las rutas HTTP para cada acción que puede realizar el robot.

---

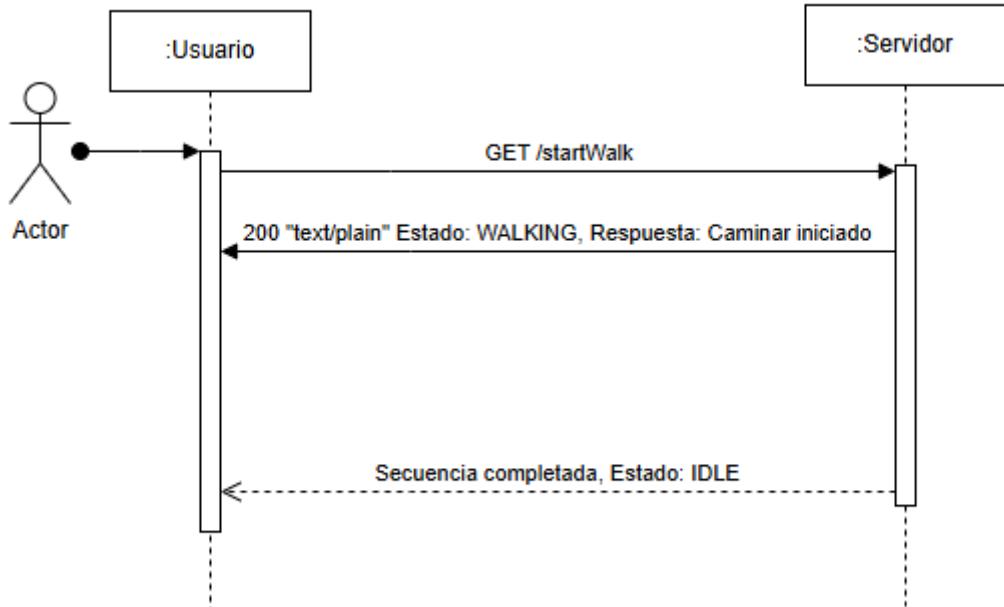
## Relación con los Requisitos

- **Control de Servos:** Las funciones moveServo() y walkSequence() permiten controlar el movimiento preciso de los servos según lo requerido para la caminata y las posiciones iniciales.
  - **Comunicación WiFi:** Utilizando las bibliotecas ESPAsyncWebServer y ESPAsyncTCP, se implementó un servidor web que permite la interacción remota con el robot.
  - **Manejo del Sensor SW520D:** La función tiltRobot() utiliza el sensor para ajustar la inclinación del robot, garantizando la estabilidad en terrenos irregulares.
- 

## Caso de Uso: Iniciar Caminar

Este caso de uso describe el flujo de acciones involucradas cuando un usuario interactúa con la interfaz web del sistema para iniciar el movimiento del robot cuadrúpedo. El objetivo principal es permitir que el usuario envíe un comando desde

la interfaz HTML, el cual es procesado por el servidor web alojado en el NodeMCU, para activar la secuencia de caminata del robot. Durante este proceso, se verifican estados del sistema, se controlan los servos mediante comandos I2C y se proporciona retroalimentación al usuario en tiempo real.



Se describe el proceso cuando el usuario presiona el botón "Iniciar caminar" en la interfaz HTML:

1. El usuario envía una solicitud HTTP GET /startWalk desde la interfaz HTML al servidor web en el NodeMCU.
2. El servidor verifica el estado del robot (IDLE o WALKING).
3. Si el estado es IDLE, el servidor:
  - a. Cambia el estado a WALKING.
  - b. Llama a la función walkSequence() para iniciar el movimiento.
4. La función walkSequence() envía comandos I2C al PCA9685 para controlar los servos en la secuencia programada.
5. El servidor responde al cliente con un mensaje de confirmación: "Caminar iniciado".
6. Al completar la secuencia, el estado del robot vuelve a IDLE.

## Pruebas

Para garantizar la funcionalidad del robot cuadrúpedo, el proceso de validación y pruebas se realizó en varias etapas, incrementando progresivamente la complejidad:

- **Pruebas Individuales de Servos:**

- Inicialmente, cada servo fue probado de manera individual. Se implementó una interfaz básica con un botón dedicado para controlar cada servo.
- Esto permitió verificar la correcta conexión eléctrica, respuesta a los comandos PWM enviados desde el PCA9685 y el rango de movimiento de cada servo.

- **Pruebas de Patas:**

- Una vez validados los servos individuales, las pruebas se ampliaron para abarcar las patas completas.
- Se creó una interfaz en la que un botón controlaba todos los servos correspondientes a una pata específica.
- Estas pruebas ayudaron a identificar problemas de sincronización o ajustes de posiciones iniciales para cada pata.

- **Prueba Completa de la Caminata:**

- En la etapa final, se integraron todas las funcionalidades en un único comando de caminata accesible desde la interfaz web.
- Al presionar un botón dedicado, se ejecutaba la secuencia de movimiento completa del robot, validando la coordinación de las patas y la estabilidad durante el desplazamiento.

## 4. Documentación Relacionada

---

### Enlaces del proyecto:

- Sistema de almacenamiento y versionado: [Repositorio GitHub](#)
- Bitácora: [Bitácora Proyecto A3](#)

### Videos finales del funcionamiento del robot

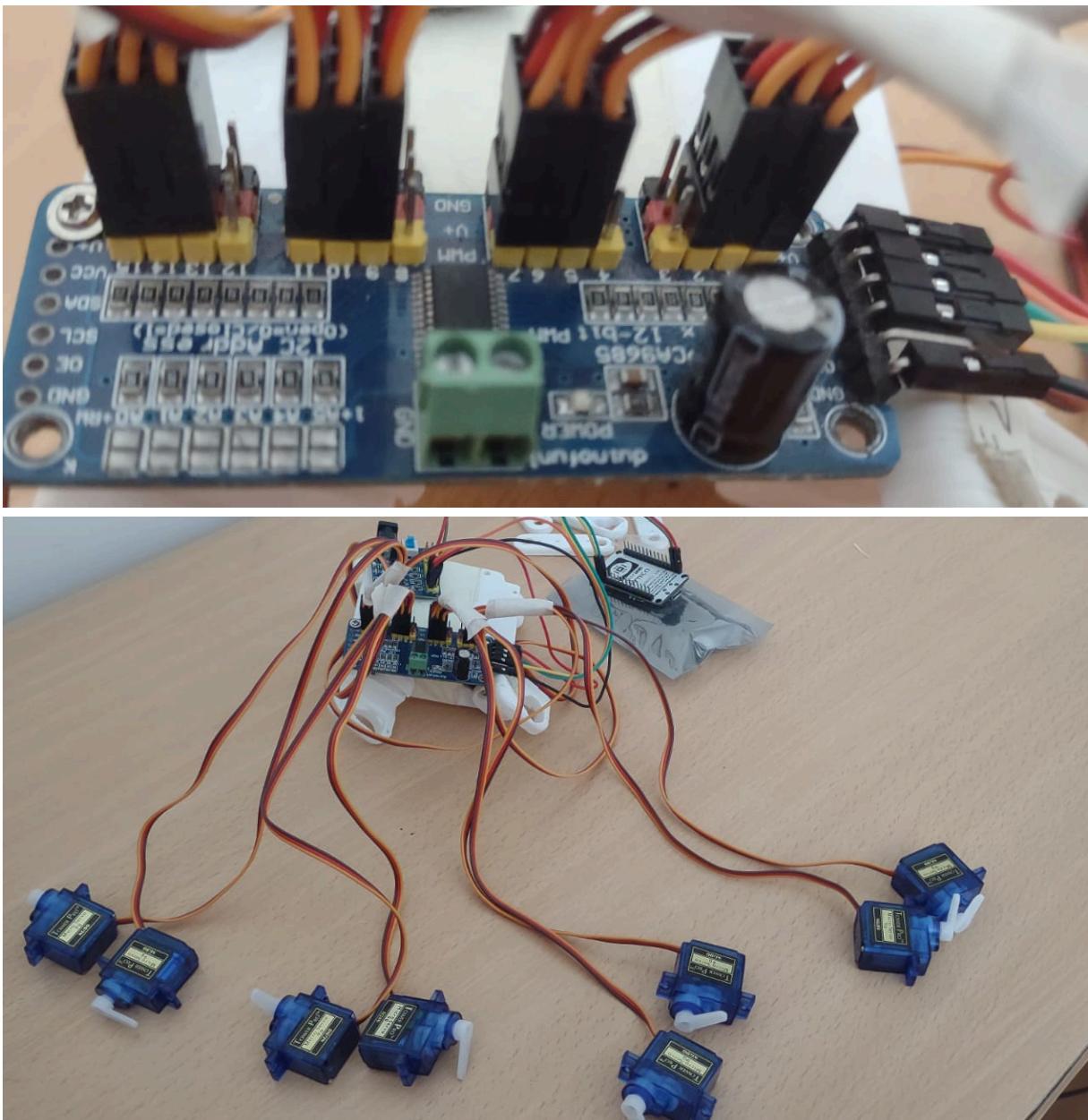
 Video de inclinacion del robot.mp4

 Video de caminata del robot.mp4

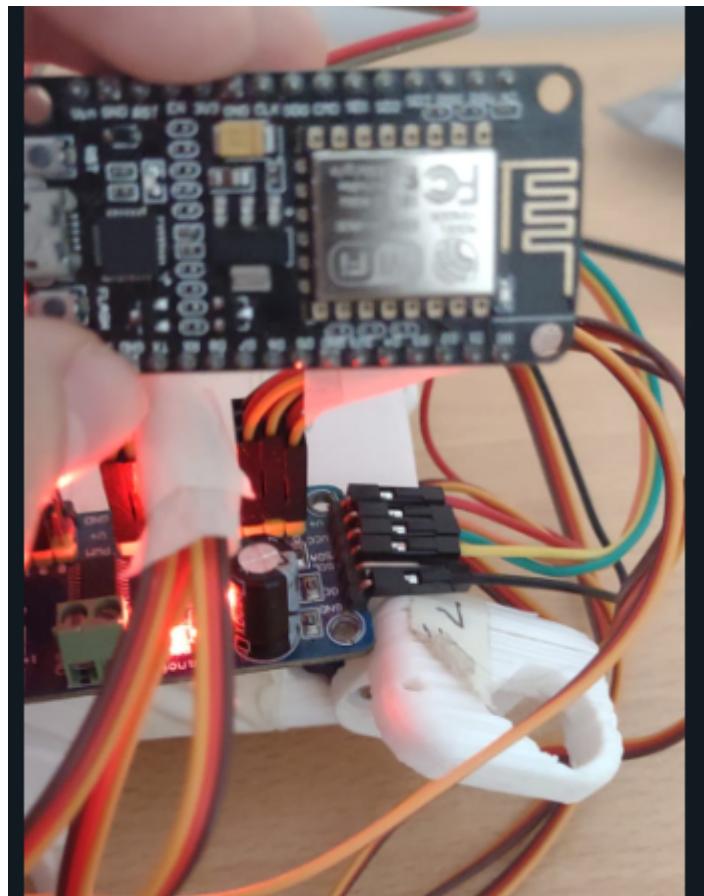
 video del sensor sws520d en funcionamiento.mp4

### Fotos de las partes físicas

Inicialmente se comenzó por el montaje de a partes del robot como se puede observar en la siguiente imagen, esto se hizo así para realizar el testeo inicial de cada servo.

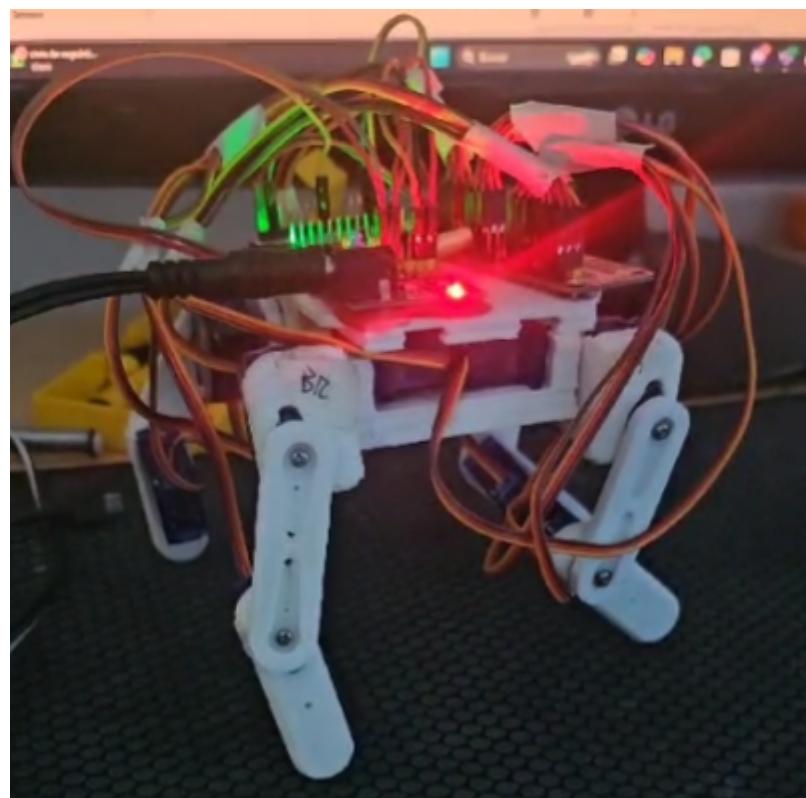
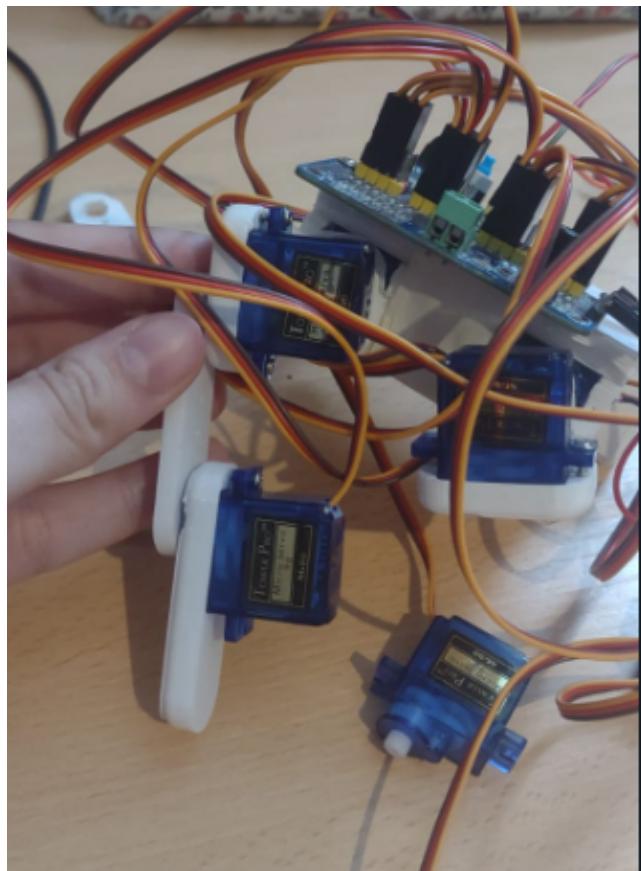


Con este test se encontraron dos servos que no funcionaban y uno que estaba defectuoso. Luego de informar a la cátedra y obtener el cambio de servos se continúo con los testeos del NodeMCU y del sensor:

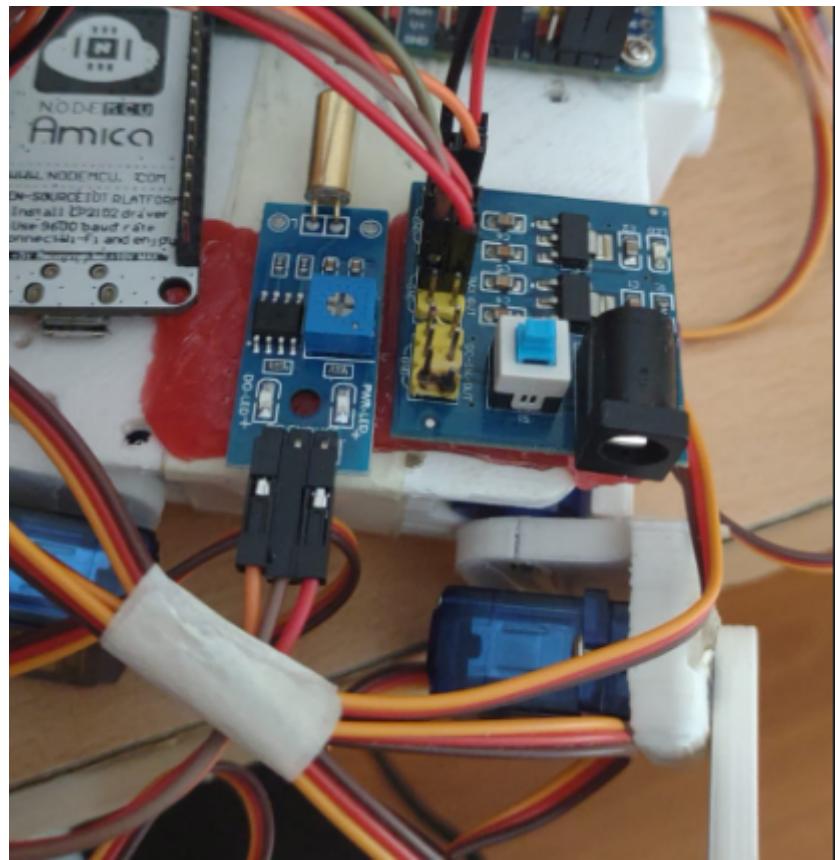


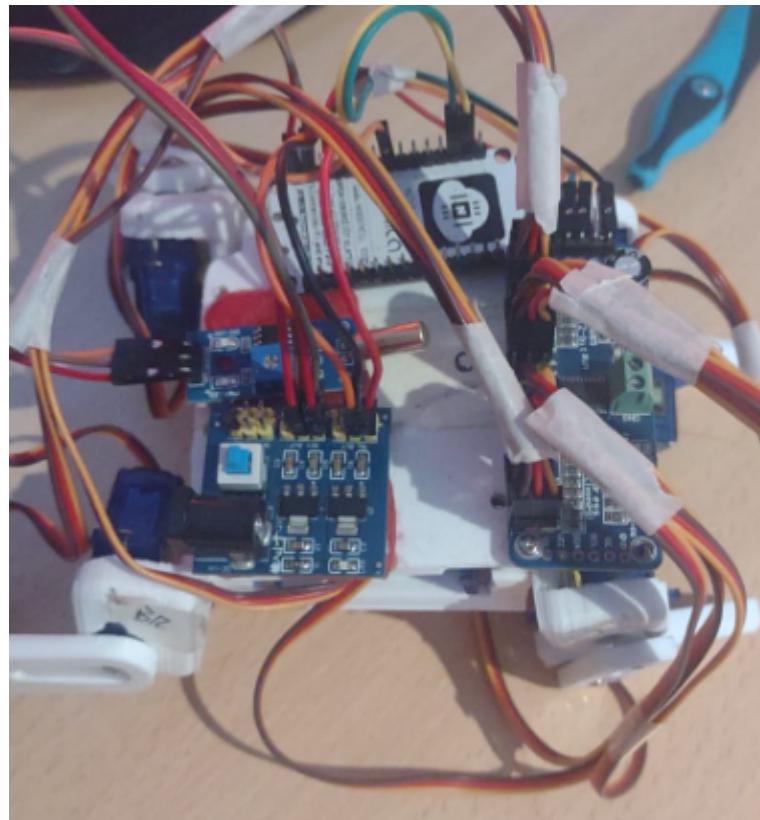
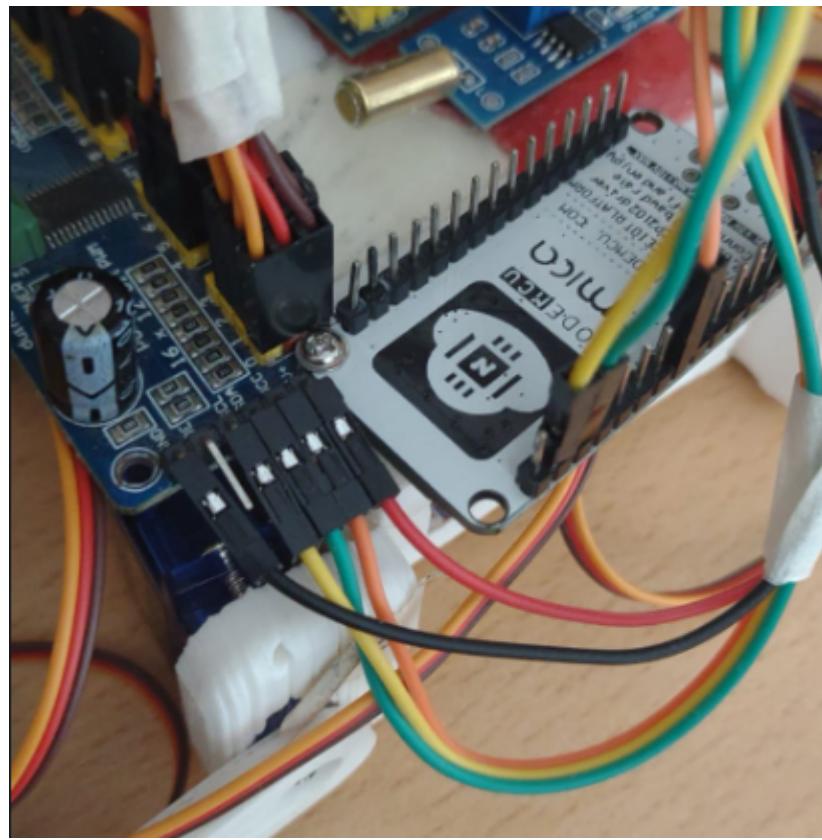
🎥 [video sensor sin ensamblar.mp4](#)

Una vez completados los test se comenzó con el ensamblado de las patas y finalmente se terminó de armar el robot completo, como puede observarse a continuación:

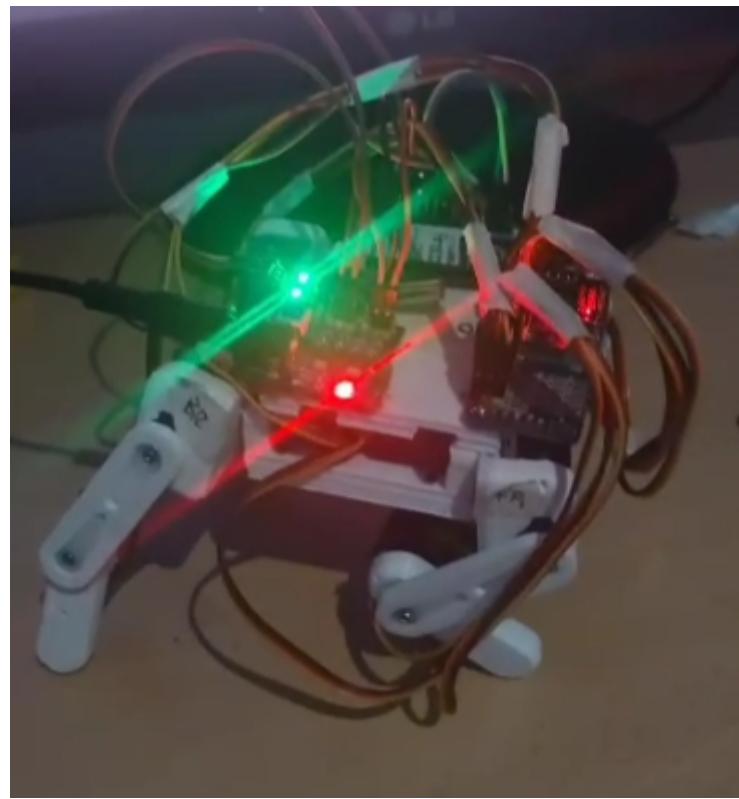


A continuación, se verán fotos que muestran más a detalle como se encuentran los componentes adheridos al “lomo” del robot:





Y por último, se muestra una foto del robot cuando se ejecuta el proceso de inclinación, justo antes de que retome su posición original:



A continuación se comparte una foto de la interfaz web que implementa el proyecto

# Control de Robot Cuadrúpedo

Iniciar caminar

Posición de arranque

Inclinarse

Sensor de Nivel (activo:  
30°)

**Nivel no detectado**

Servo  
1      Servo  
5      Servo  
11      Servo  
13

Servo  
0      Servo  
6      Servo  
9      Servo  
15

Servo  
2      Servo  
4      Servo  
10      Servo  
14

La imagen presentada corresponde a la interfaz web desarrollada para el control del robot del proyecto. Esta interfaz permite al usuario interactuar de manera sencilla con las funcionalidades principales del robot. Está diseñada para ser accesible desde cualquier dispositivo conectado al Access Point (AP) del NodeMCU.

En la sección superior se encuentra el botón “iniciar Caminar”, este botón envía una solicitud HTTP GET al servidor para ejecutar la función walkSequence() en el NodeMCU. Activa la secuencia de caminata del robot, coordinando las patas para lograr un movimiento fluido.

Luego se puede ver el botón de "Posición de arranque", este envía comandos para ejecutar las funciones position() y position2(), llevando las patas del robot a su posición inicial predefinida.

y por último, se encuentra el botón "Inclinarse", el cual permite al robot inclinarse hasta detectar un nivel estable mediante el sensor SW520D. Esta funcionalidad es útil para validar la inclinación y estabilidad del robot.

En la sección del centro se puede observar el estado del Sensor de Nivel. Muestra el estado actual del sensor de inclinación. Si el sensor detecta que el robot está nivelado, se despliega el mensaje "Nivel no detectado" en un texto resaltado. El ángulo activo ( $30^\circ$ ) está indicado como referencia, indicando el umbral a partir del cual el sensor responde.

En la sección inferior se encuentra la cuadrícula de servos. Los cuadros representan cada uno de los servos del robot, numerados de acuerdo con su asignación en el controlador PCA9685.

Cada cuadro debería poder actualizar su estado dinámicamente según el servo activo, proporcionando información visual sobre qué servo está en movimiento, pero no se logró completar la visualización de esto.

# Apéndice A: Materiales y Presupuesto

---

En esta etapa final del proyecto, se consolidaron los elementos necesarios para construir el robot cuadrúpedo "Micro Quadruple". Inicialmente, se había previsto utilizar tanto un microcontrolador Arduino UNO como un módulo Wi-Fi ESP8266. Sin embargo, tras evaluar las capacidades del ESP8266, se determinó que este era capaz de cumplir tanto las funciones de procesamiento como de conectividad Wi-Fi. Esto permitió simplificar el diseño y optimizar los recursos del proyecto.

A continuación, se detalla la lista actualizada de materiales y costos para la construcción del sistema:

Elemento	Cantidad	Costo Unitario (ARS)	Costo Total (ARS)	Descripción
Servomotores SG90	12 + 3 extra	2.519	37.785	Motores de 9g utilizados para el movimiento de las patas del robot. Se proporcionaron 3 servos adicionales por la cátedra para reemplazar posibles daños durante las pruebas.
Sensor de inclinación SW520D	1	2.936	2.936	Sensor utilizado para detectar inclinación y nivelar el robot automáticamente.
Módulo Wi-Fi ESP8266	1	8.239	8.239	Microcontrolador con conectividad Wi-Fi integrado, encargado del procesamiento y servidor web.
Controlador de servos PCA9685	1	12.999	12.999	Módulo PWM utilizado para manejar hasta 16 servomotores de manera simultánea mediante el protocolo I2C.
Fuente de alimentación 5V	1	8.673	8.673	Fuente utilizada para alimentar los servomotores y otros componentes electrónicos del sistema

Estructura del robot	1 (varias piezas)	18.000	18.000	Estructura diseñada e impresa en 3D para montar los componentes y soportar las patas del robot.
----------------------	-------------------	--------	--------	---

#### **Pruebas y ajustes:**

Durante las pruebas, se detectaron diferencias en el tamaño de los servomotores respecto a los utilizados en versiones previas del proyecto. Esto ocasionó que algunas piezas de la estructura, particularmente las patas, fueran incompatibles. Se realizaron nuevas mediciones y se reimprimieron las piezas mediante impresión 3D para ajustarlas a las especificaciones actuales.

#### **Servomotores adicionales:**

La cátedra proporcionó tres servomotores SG90 adicionales para garantizar el correcto funcionamiento del robot, especialmente durante las pruebas, donde algunos servos presentaron fallos debido a un desgaste prematuro.

#### **Actualización del hardware:**

Todos los elementos necesarios fueron proporcionados por la cátedra en la semana del 19/9, incluyendo los servomotores adicionales y el PCA9685.

Links de referencias:

[Mini Servo Tower Pro Sg90 9g Robotica Arduino Servomotor | MercadoLibre](#)

[Módulo De Sensor De Ángulo Inclinacion Sw-520d Arduino | MercadoLibre](#)

[Nodemcu Wifi Lua V2 Esp8266 Gpio Esp12f 4mb Uart | MercadoLibre](#)

[Controlador Servos 16 Canales I2c Pca9685 Pwm Arduino Nubbeo | MercadoLibre](#)

[Fuente De Alimentación Switching 5v 1a F5s1 | MercadoLibre](#)

[Calculadora de precios de Impresión 3D - 3DCREATIO](#)