

**Universidad Nacional de Colombia**

**Sede Bogotá**

Facultad de Ingeniería

Departamento de Ingeniería Mecánica y Mecatrónica

# **Documentación de la pata del robot Cheetah**

**Daniel Esteban Ramírez Chiquillo**

6 de marzo de 2025

# Índice

<b>1. Montaje</b>	<b>2</b>
1.1. Estructura de aluminio . . . . .	2
1.2. Panel eléctrico . . . . .	2
<b>2. Eslabones</b>	<b>3</b>
<b>3. Sistema de coordenadas y posición home</b>	<b>4</b>
<b>4. Conectando el robot a un computador</b>	<b>5</b>
4.1. Conexión física . . . . .	5
4.2. Software . . . . .	6
<b>5. Conectando el robot a una red Wifi</b>	<b>7</b>
5.1. Cómo cambiar la red Wifi de la ESP32 . . . . .	7
<b>6. Uso básico del robot a través de la interfaz de usuario</b>	<b>10</b>
6.1. Encendiendo el robot . . . . .	10
6.2. Moviendo el robot . . . . .	12
6.3. Apagando el robot . . . . .	13
6.4. Mecanismos de seguridad . . . . .	13
<b>7. Interactuando a través de la API</b>	<b>14</b>
7.1. Llamado sin cuerpo . . . . .	15
7.2. Llamado con cuerpo . . . . .	15
7.3. Lotes de comandos . . . . .	15
7.4. Seguridad . . . . .	16

# 1. Montaje

El Mini Cheetah es un robot cuadrúpedo desarrollado por el MIT. El robot descrito en esta documentación representa una de las cuatro patas del Mini Cheetah. Para operar esta pata sin necesidad de contar con el cuerpo y las otras tres patas del robot, se usa el montaje que se muestra en la figura 1.

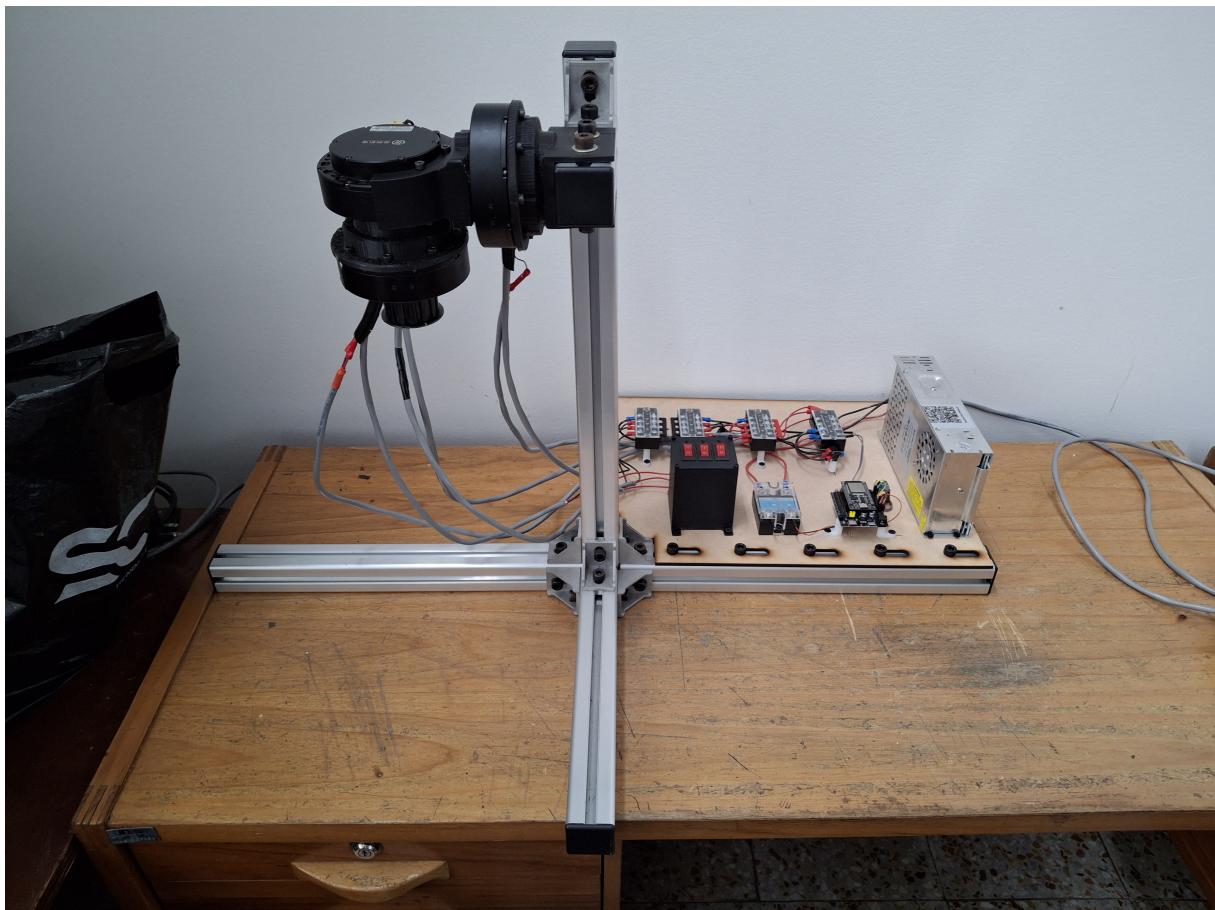


Figura 1: Montaje de la pata, sin eslabones.

Las partes principales del montaje son: la estructura de aluminio y el panel eléctrico.

## 1.1. Estructura de aluminio

Todo el robot (menos el botón de emergencia) se encuentra anclado a la estructura de aluminio. Esta está construida con perfiles 4040, con tapas plásticas de 4 mm en sus extremos. Las uniones entre perfiles miden 40 mm en toda dirección. El perfil al que está atornillada la pata mide 120 mm de largo. El resto de la estructura mide: 848 x 648 x 554 mm.

## 1.2. Panel eléctrico

Todos los componentes eléctricos se organizan en el panel eléctrico, tal como se observa en la figura 2. Vistos de izquierda a derecha y de arriba a abajo, los componentes son:

1. 4 bloques de terminal.
2. Fuente de alimentación.
3. Switches, uno para cada motor.
4. Relé.
5. Microcontrolador ESP32.
6. Botón de emergencia.
7. Cable de alimentación de la fuente.

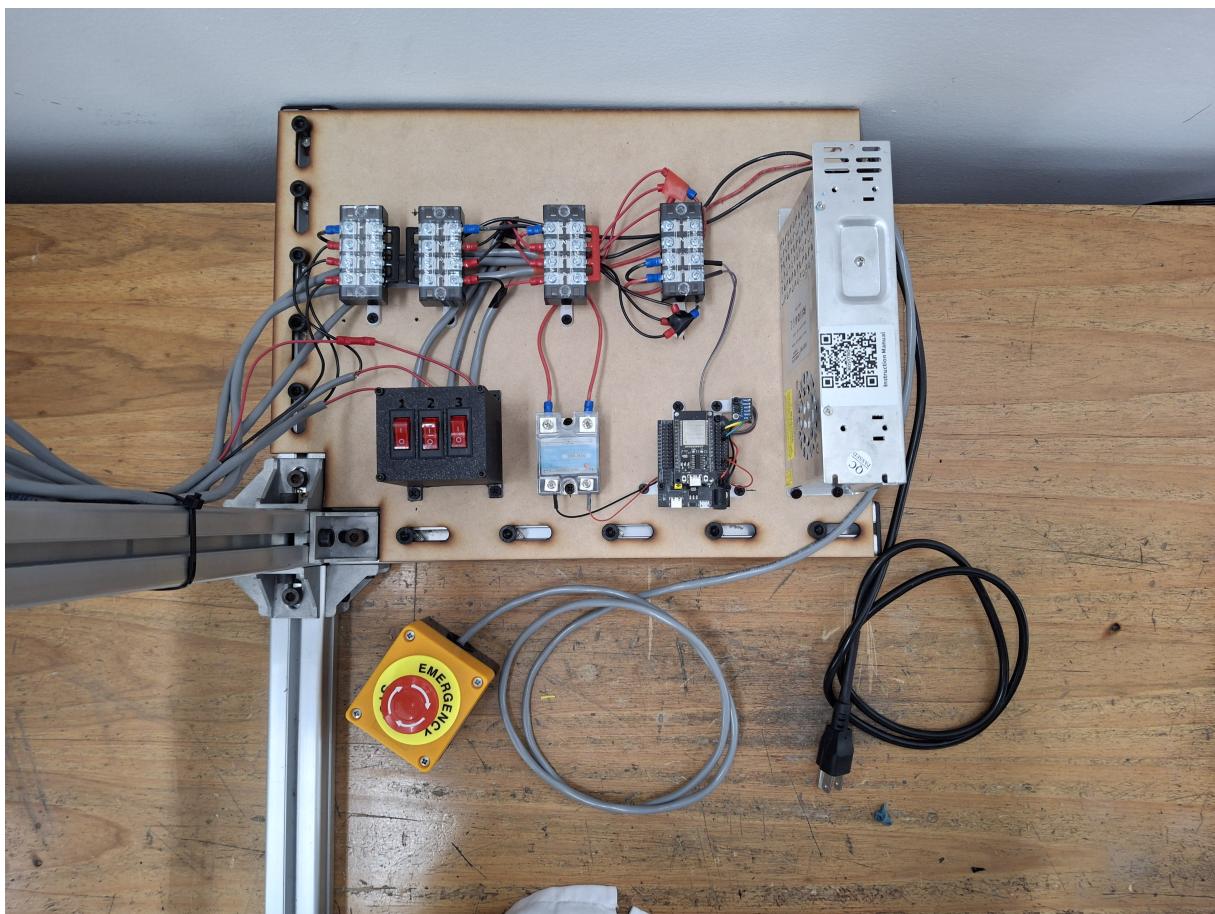


Figura 2: Panel eléctrico.

## 2. Eslabones

En la figura 4 se aprecia la configuración de los eslabones del robot. En el repositorio se encuentran los archivos CAD (Inventor y STL) de todos los componentes.

### 3. Sistema de coordenadas y posición home

Se recomienda trabajar con el sistema de coordenadas que se muestra en la figura 3.

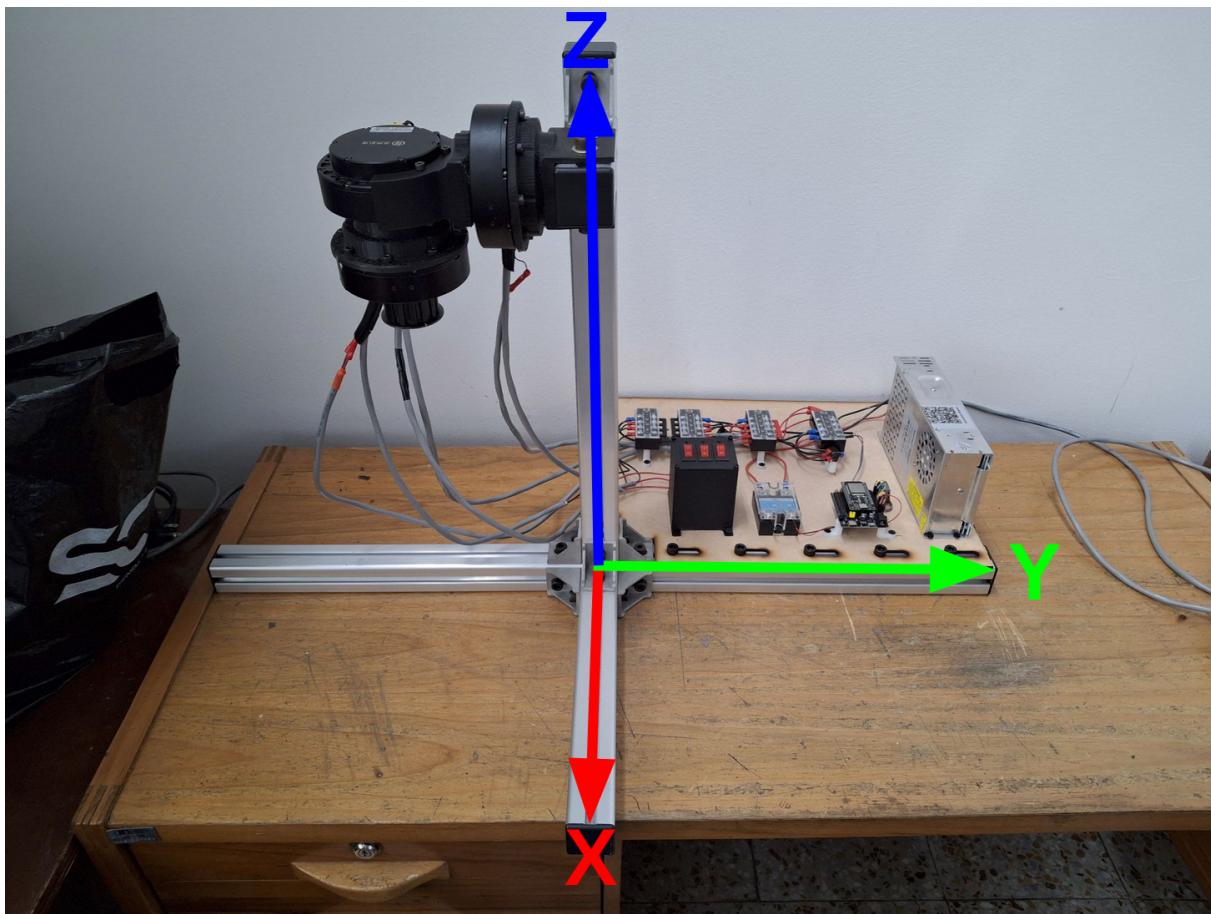


Figura 3: Sistema de coordenadas del robot

La posición home del robot se aprecia en la figura 4. Se usa esta posición como home porque resulta la más cómoda para almacenar el robot, y los eslabones se mantienen en ella con los motores apagados.<sup>1</sup>

---

<sup>1</sup>Los puertos de conexión de todos los motores apuntan hacia X negativo, lo cual no se aprecia explícitamente en la imagen.

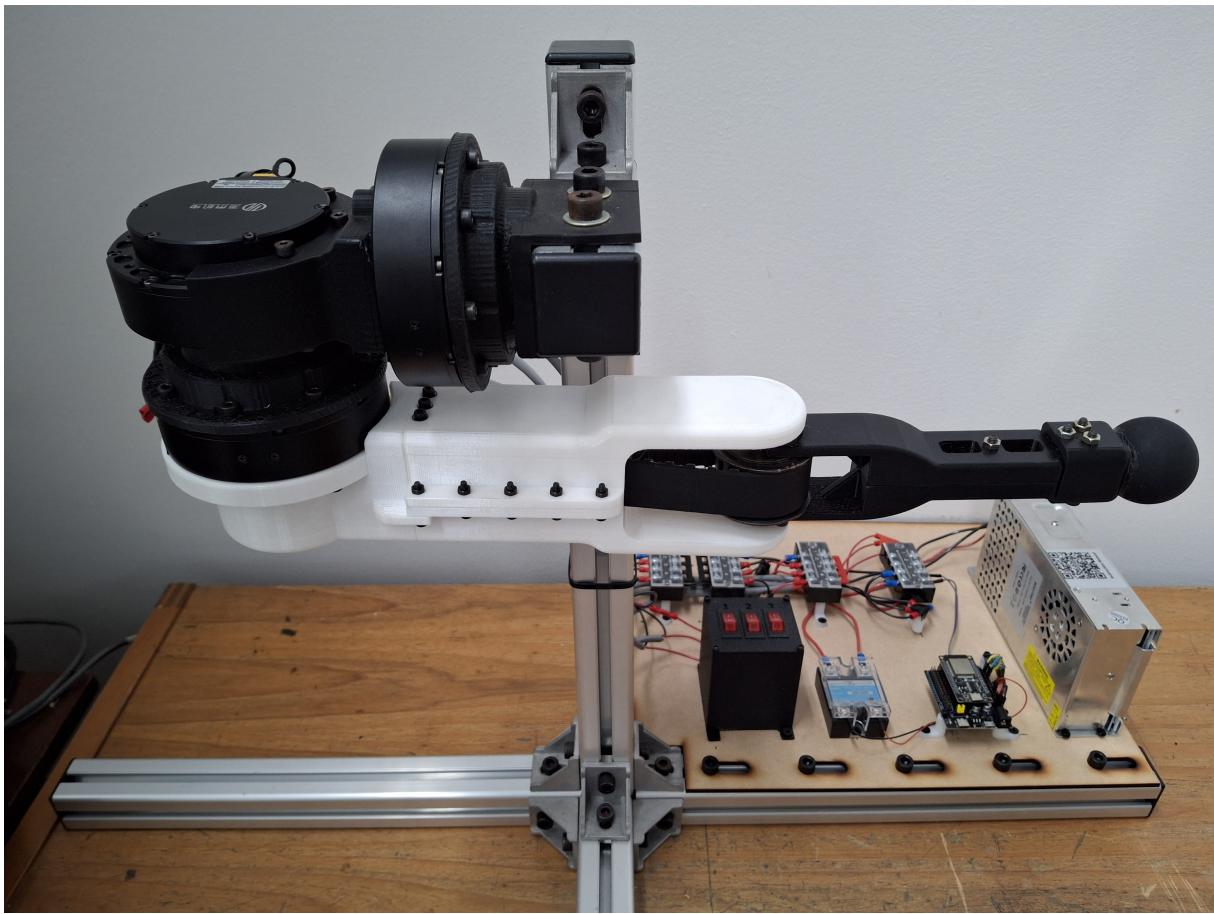


Figura 4: Posición home.

## 4. Conectando el robot a un computador

### 4.1. Conexión física

Se usa una tarjeta ESP32 para controlar todo el sistema. Aunque los comandos del robot se acceden vía HTTP en la red local, es necesario conectar la tarjeta a un computador para leer los *logs* del sistema. Es importante que el cable sea USB-A a USB-C, el computador no reconoce la tarjeta si se usa un cable USB-C a USB-C.

Otro aspecto a tener en cuenta es el puerto correcto a usar en la ESP32. En la figura 5 se observan dos puertos USB-C. El inferior corresponde a la alimentación de la tarjeta de expansión, utilizada para simplificar el montaje. El superior es el puerto de la ESP32, y es el que se debe usar. Cabe aclarar que solo se necesita conectar la ESP32, no la tarjeta de expansión, ya que al encenderse el microcontrolador provee suficiente energía a esta última.

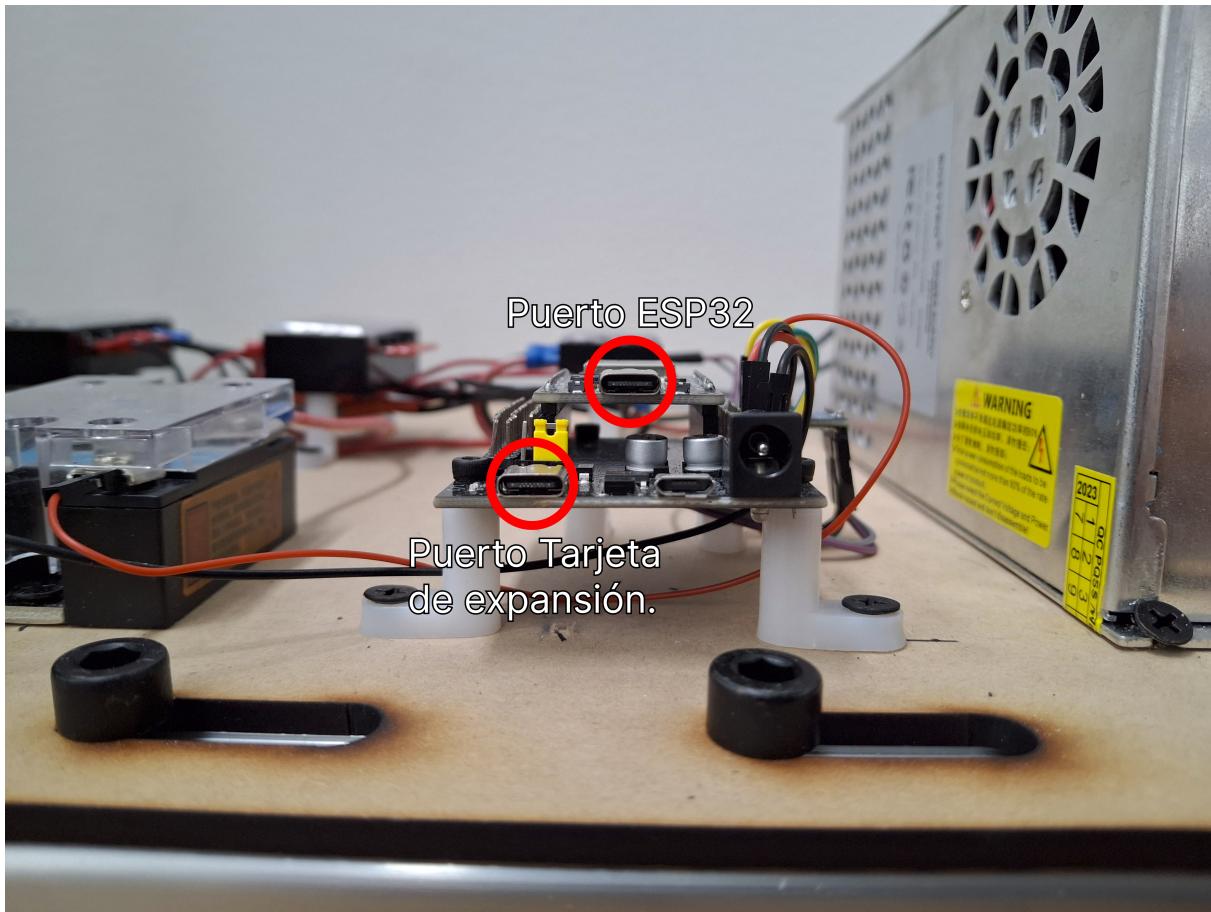


Figura 5: ESP32.

#### 4.2. Software

Para leer los *logs* desde el computador se recomienda instalar el framework ESP-IDF y utilizar el monitor desde VSCode, como se observa en la figura 6.

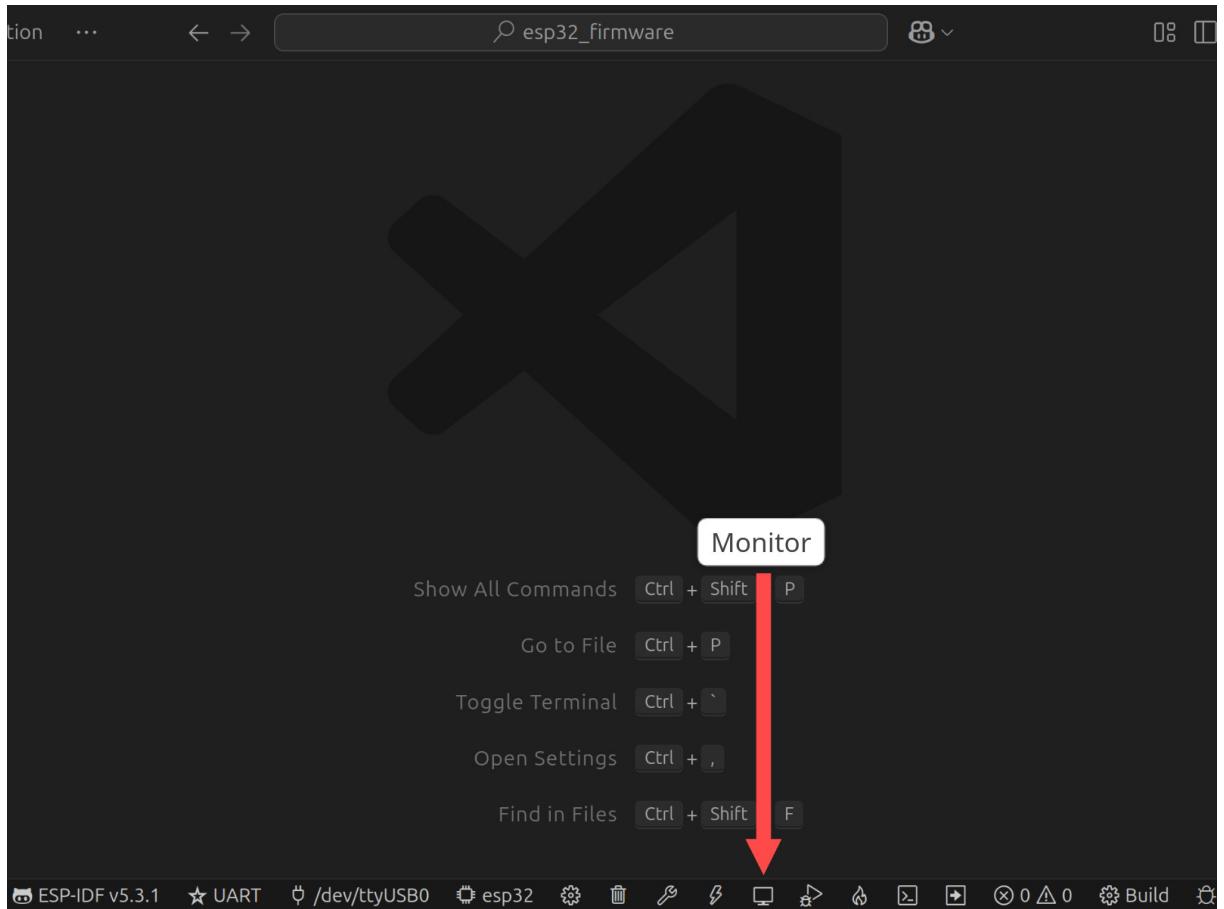


Figura 6: Monitor para ESP32 desde VSCode.

Si ya se tiene instalado el ESP-IDF y los botones no aparecen en la barra inferior de VSCode, presiona **Ctrl+Shift+P** y escribe el comando **ESP-IDF: Monitor Device**.

## 5. Conectando el robot a una red Wifi

Al momento de escribir esta guía, la ESP32 está configurada para conectarse automáticamente a la red LabFabEx, en el Laboratorio Fábrica Experimental / Sala CAM, cada vez que se enciende. Si se quiere cambiar la red Wifi a la que se conecta, es necesario volver a programar la tarjeta.

Antes de describir los pasos para hacer esto, es importante aclarar que la ESP32 solo se conecta a redes de 2.4 GHz.

### 5.1. Cómo cambiar la red Wifi de la ESP32

- Clonar el repositorio del robot.
- Ir a la carpeta `esp32_firmware` y abrirla en VSCode.
- En la barra inferior, hacer click en **SDK Configuration Editor** (`menuconfig`), como se señala en la figura 7.

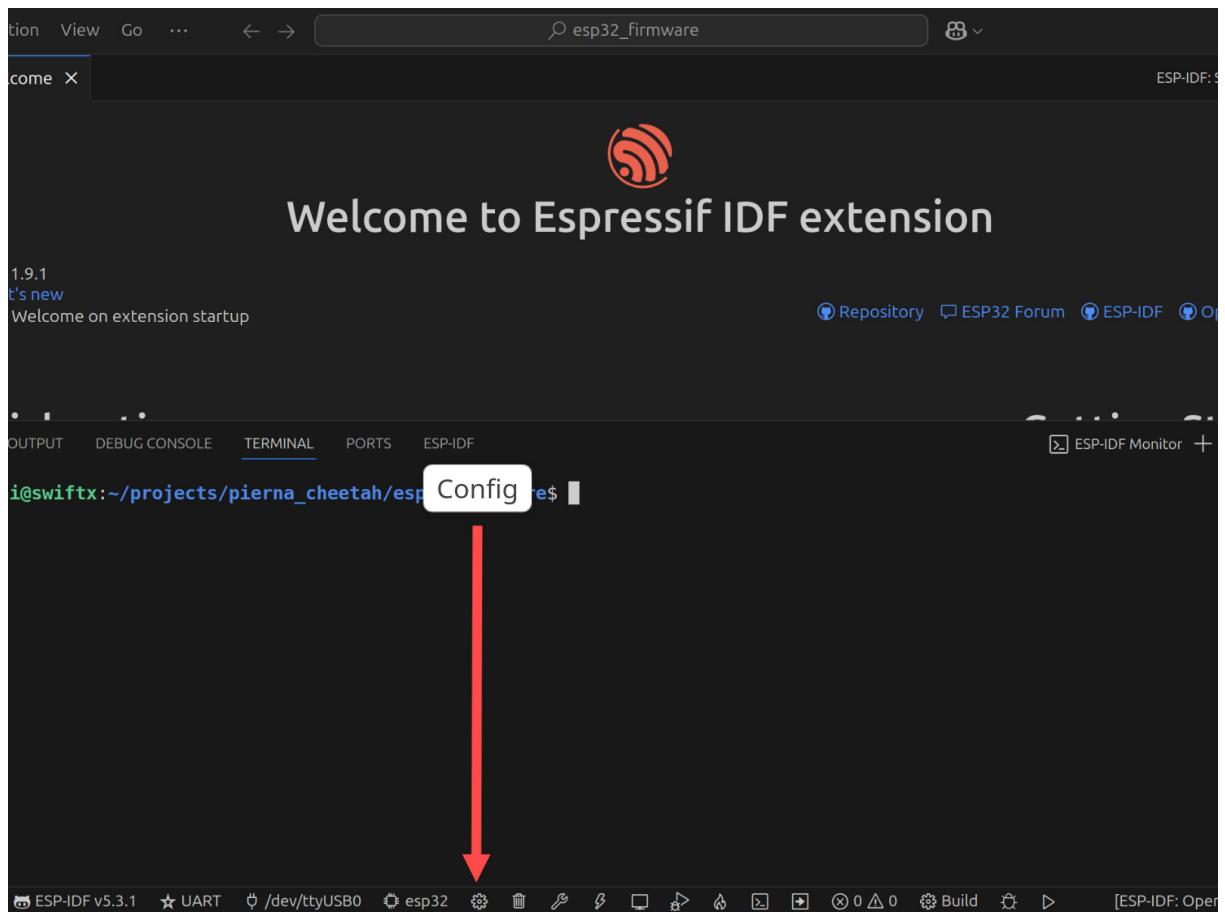


Figura 7: SDK Configuration Editor.

- Ir a **Config** y editar la información de la red Wifi. Ver la figura 8.

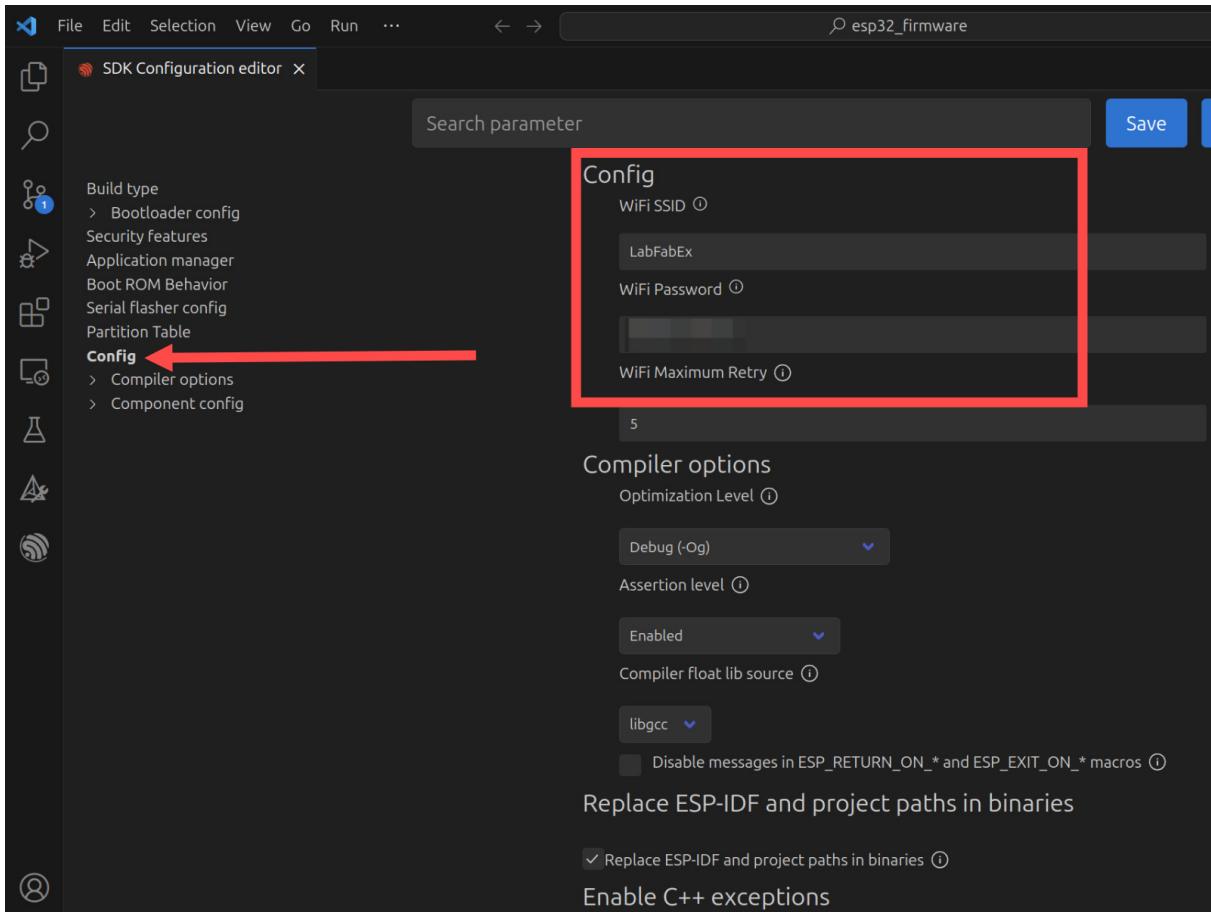


Figura 8: Configuración de Wifi.

- Presionar **Build, flash and monitor**. Ver la figura 9.

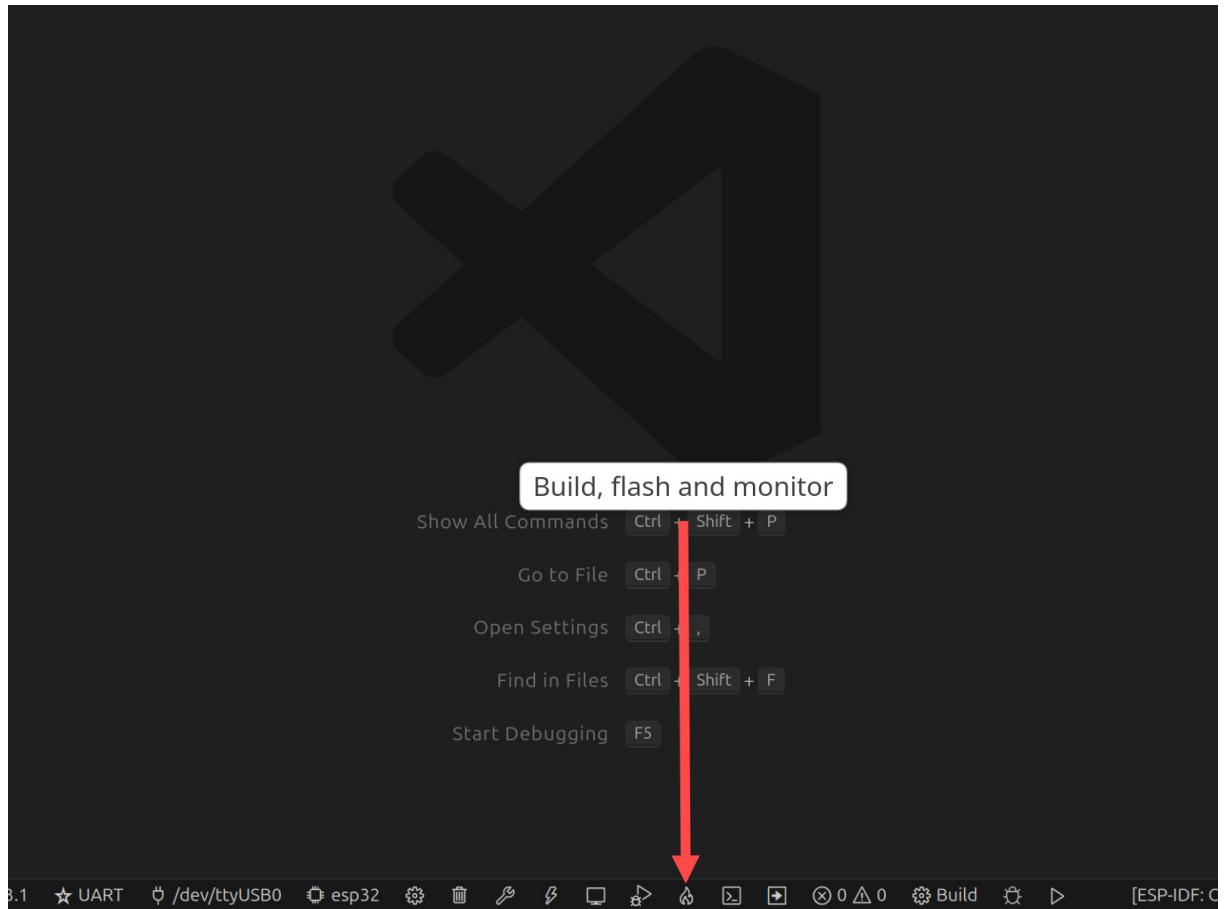


Figura 9: Botón: Build, flash and monitor.

Tras esto, el microcontrolador se reprograma, abre el monitor y se conecta a la nueva red Wifi.

## 6. Uso básico del robot a través de la interfaz de usuario

Una vez realizados los pasos anteriores, se pueden enviar comandos desde cualquier dispositivo en la misma red wifi que la tarjeta. Para un primer acercamiento se utiliza la interfaz de usuario, disponible en <http://cheetah.local/>.

### 6.1. Encendiendo el robot

Al acceder a la interfaz por primera vez se muestra la pantalla «Robot is off», tal como se observa en la figura 10.

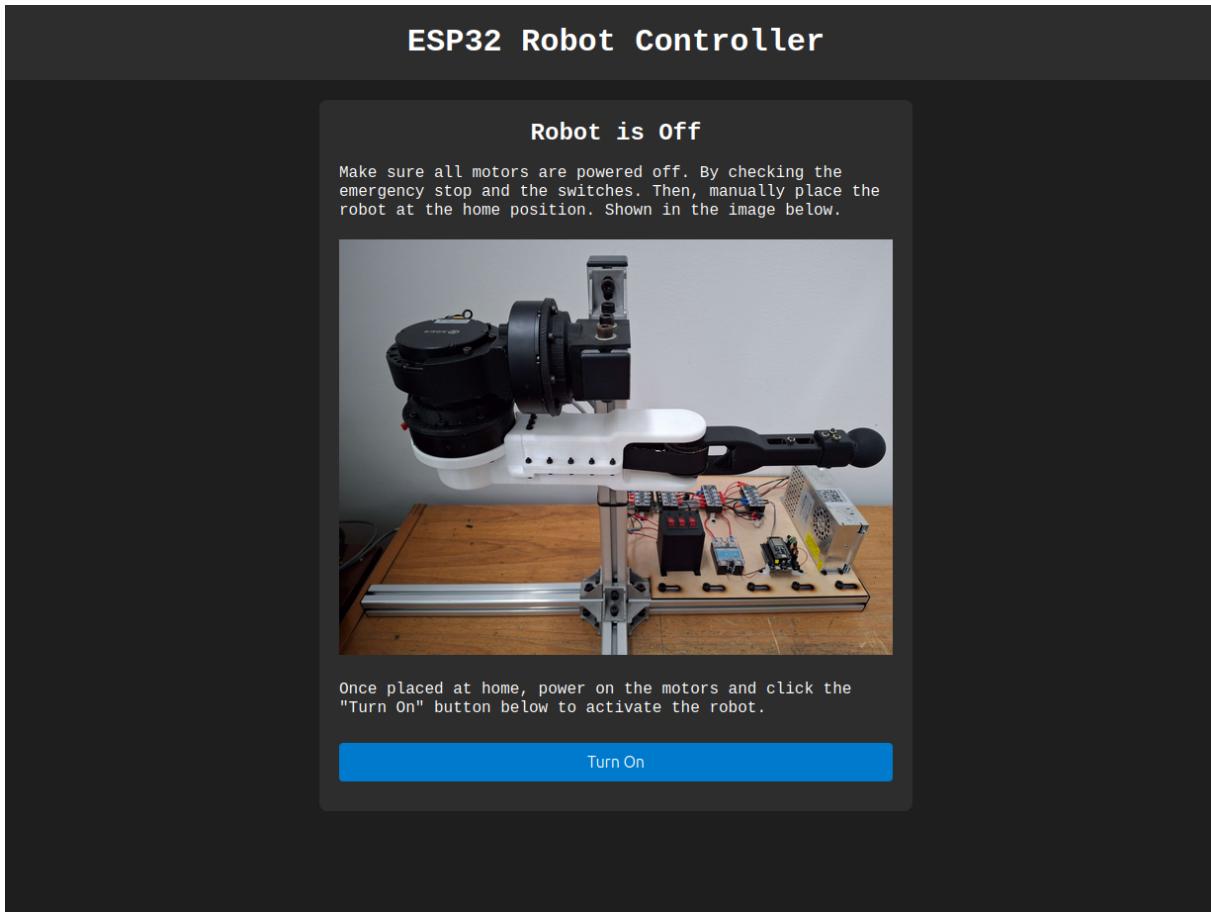


Figura 10: UI: Robot Off.

Según lo indicado en la página, se siguen los siguiente pasos para encender el robot:

1. Asegurarse de que todos los motores están apagados.
2. Mover el robot a home, como se indica en el capítulo 3.
3. Encender los motores.
4. Hacer click en el botón Turn On.

El robot enciende el relé, calibra los sensores y activa los motores. Todo esto se observa en los *logs*. Tras encenderse, se muestra la página «Robot is on», como se observa en la figura 11.

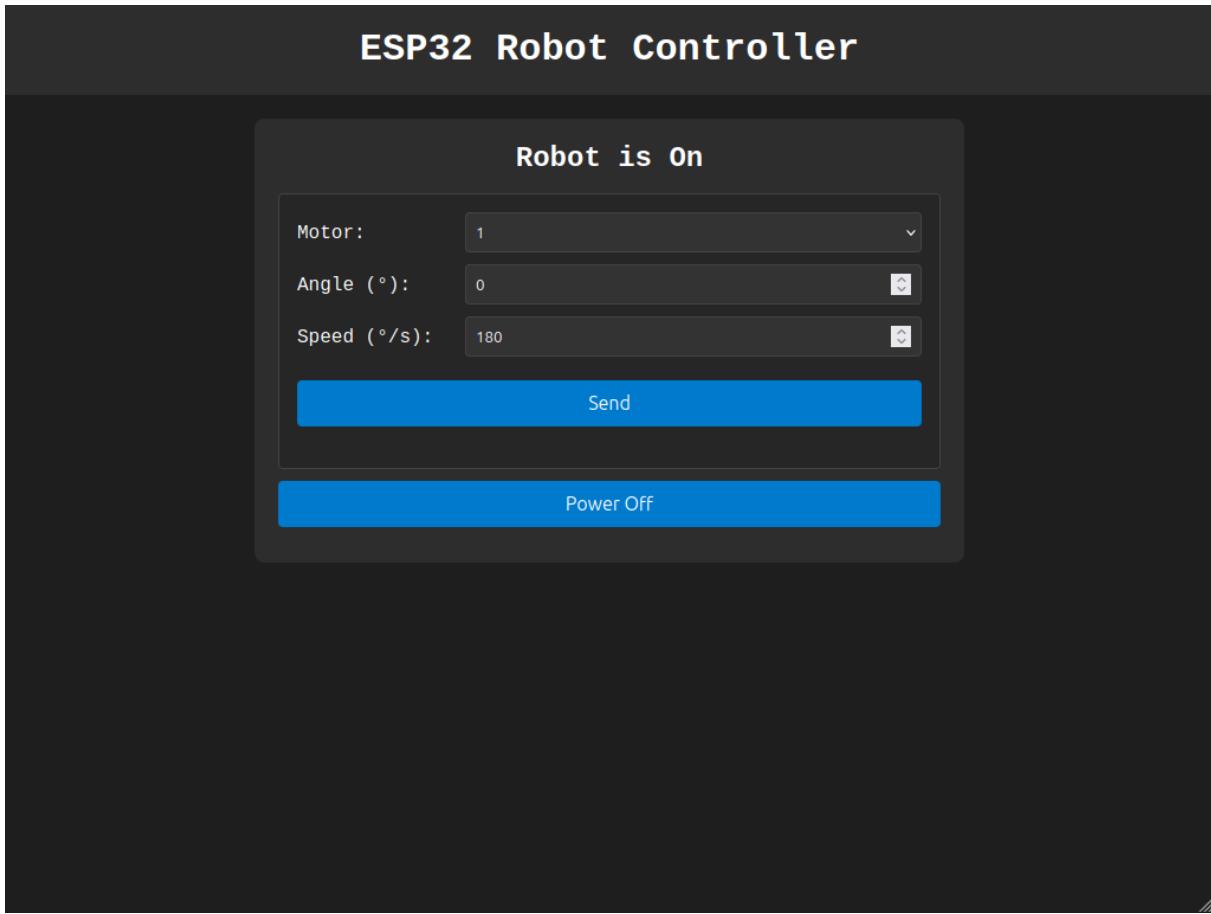


Figura 11: UI: Robot is On.

## 6.2. Moviendo el robot

En esta pantalla se envían comandos de posición al robot. Cada comando acepta tres parámetros:

- **Motor:** la ID del motor al que se envía el comando.
- **Angle:** el ángulo (en grados) al que se desea que llegue el motor.
- **Speed:** la velocidad máxima (en grados por segundo) a la que se permite que llegue el motor.

El servidor verifica que los valores se encuentran dentro de los límites permitidos. Cada motor tiene límites articulares y velocidades máximas programadas.

Motor	Límites articulares [grados]	Velocidad máxima [grados/s]
1	[0, 90]	720
2	[-90, 0]	1440
3	[-135, 135]	1440

Al usar este tipo de comandos, se recomienda esperar a que un comando termine su trayectoria antes de enviar el siguiente.

### 6.3. Apagando el robot

Cuando se terminan de enviar comandos, se presiona «Turn Off» en la misma pantalla. El robot se mueve a home, desactiva los motores y apaga el relé.

***¡¡¡ Aviso de seguridad !!!*** Cada vez que el robot se apaga por software, es necesario apagar los motores físicamente. En otras palabras, cada vez que en consola aparece el log: «Robot is now OFF.», el usuario debe apagar el switch de cada motor o presionar el botón de emergencia. No hacer esto puede causar movimientos súbitos y peligrosamente rápidos la próxima vez que se enciende el robot, lo cual puede dañar alguna pieza o lesionar al usuario.

### 6.4. Mecanismos de seguridad

Si un motor deja de enviar respuestas por CAN a la ESP32, tras tres respuestas fallidas, la ESP32 inicia un procedimiento para apagar todo el robot. Al finalizar el procedimiento aparece el log «Robot is now OFF.». Entonces se deben:

1. Apagar todos los motores.
2. Colocar manualmente el robot en home.
3. Volver a encender los motores.
4. Reiniciar la ESP32.
5. Recargar la página web.

Si la ESP32 se apaga mientras el robot está encendido, la próxima vez que se enciende se detecta un apagado erróneo. Esto bloquea todos los endpoints de la API y, en consecuencia, todas las acciones de la página web. Si se intenta enviar algún comando, se muestra el mensaje de la figura 12.



Figura 12: UI: Recovery required.

Para volver a usar el robot, se deben seguir los pasos indicados en el mensaje:

1. Apagar físicamente todos los motores.
2. Mover el robot a home.
3. Volver a encender los motores.
4. Hacer click en el botón «Clear recovery».

*¡¡¡ Aviso de seguridad !!!* Aunque el robot no puede verificar que el usuario ha seguido los pasos indicados, es **muy** peligroso hacer click en el botón sin antes realizar el procedimiento.

## 7. Interactuando a través de la API

La página mostrada en <http://cheetah.local/> es simplemente una interfaz amigable para interactuar con la API del robot. Esta sección se enfoca en mostrar el uso básico de la API y en advertir sobre algunos aspectos al interactuar con ella. La referencia de la API se puede encontrar en su propio documento, disponible en el repositorio de Github.

Para empezar, existen varias maneras de interactuar con una API HTTP. En estos ejemplos se utiliza Curl, ya que viene instalado por defecto en la mayoría de sistemas

Linux. No obstante, existen librerías en prácticamente todos los lenguajes de programación para facilitar este tipo de llamadas.

Los endpoints de esta API se dividen en dos grupos: los que requieren cuerpo en el request y los que no.

## 7.1. Llamado sin cuerpo

Uno de los endpoints que no requiere cuerpo en el request es POST /api/robot/on, por lo que para llamarlo simplemente se invoca:

```
$ curl -v -X POST http://cheetah.local/api/robot/on
```

El robot procesa el mensaje y devuelve una respuesta en JSON. Si todo resulta correcto se observa:

```
{
  "status": "ok",
  "message": "Robot turned ON successfully"
}
```

Si ocurre algún error, se muestra un JSON similar pero con un mensaje de error.

## 7.2. Llamado con cuerpo

El endpoint POST /api/command/move es equivalente a enviar un comando de movimiento, como se describe en el capítulo 6.2. Por lo tanto, se agrega la misma información al cuerpo del request:

```
$ curl -v -X POST http://cheetah.local/api/command/move \
-H "Content-Type: application/json" \
-d '{"motor_id": 1, "position": 0, "speed": 180}'
```

Si todo resulta correcto, se muestra:

```
{
  "status": "ok",
  "message": "Move command accepted"
}
```

Si ocurre algún error, se recibe un JSON con un mensaje de error.

## 7.3. Lotes de comandos

El endpoint POST /api/command/batch permite enviar un lote de comandos para uno o más motores. Este endpoint recibe un cuerpo JSON con la siguiente estructura:

```
{
  "batch": [
    {"motor_id": 1, "position": 45, "speed": 100},
    {"motor_id": 2, "position": 90, "speed": 50},
    {"motor_id": 3, "position": 135, "speed": 25},
    {"motor_id": 1, "position": 0, "speed": 50},
```

```

        {"motor_id": 2, "position": 0, "speed": 75},
        {"motor_id": 3, "position": 0, "speed": 100}
    ]
}

```

El objeto JSON contiene un único campo, `batch`, cuyo valor es un arreglo de comandos. En este arreglo se pueden incluir tantos comandos de posición como sea necesario.

La ejecución de los comandos sigue el siguiente criterio: **los comandos para distintos motores se ejecutan en paralelo, mientras que los comandos para un mismo motor se procesan de forma secuencial**.

Por ejemplo, con el lote mostrado, el flujo de ejecución es el siguiente:

1. Se inician simultáneamente los tres primeros comandos (uno para cada motor).<sup>2</sup>
2. Cuando finaliza el comando del motor 1 (primer comando), los comandos para los motores 2 y 3 continúan ejecutándose.
3. A pesar de que los comandos 2 y 3 sigan en ejecución, se inicia el procesamiento del cuarto comando, que corresponde nuevamente al motor 1.
4. Tras concluir el comando del motor 2 (segundo comando), se inicia el quinto comando para dicho motor.
5. Finalmente, al terminar el comando del motor 3 (tercer comando), se procede con el sexto comando.

## 7.4. Seguridad

Una vez más, la UI mostrada en `http://cheetah.local/` es simplemente una envoltura amigable de la API; por lo tanto, **las medidas de seguridad explicadas en el capítulo 6 se deben seguir con la misma urgencia que si se trabajara desde la página**.

Se recuerdan los dos avisos mostrados en el capítulo 6:

***¡¡¡ Aviso de seguridad !!!*** Cada vez que el robot se apaga por software, es necesario apagar los motores físicamente. Es decir, cada vez que en consola aparece el log: «Robot is now OFF.», el usuario debe apagar el switch de cada motor o presionar el botón de emergencia. No hacerlo puede causar movimientos súbitos y peligrosamente rápidos la próxima vez que se enciende el robot, lo que puede dañar alguna pieza o lesionar al usuario.

***¡¡¡ Aviso de seguridad !!!*** Aunque el robot no verifica que el usuario haya seguido los pasos indicados en el mensaje [de «Recovery required»], resulta **muy** peligroso hacer click en el botón [en este caso, llamar al endpoint `POST /api/recovery/clear`] sin antes realizar el procedimiento.

Nota final de ***¡¡¡ Seguridad !!!***: si en cualquier momento se tiene duda del estado del robot, o en general de que algún procedimiento descrito en esta documentación se realiza adecuadamente, se debe presionar el botón de seguridad, desconectar la ESP32, mover el robot a home y reiniciar el proceso según se indica en el capítulo 6.1. Esto garantiza que ningún motor, ni la ESP32, mantienen un estado residual en memoria. Por supuesto, no se debe acercar al espacio de trabajo del robot si los motores están encendidos. Estos son mucho más fuertes de lo esperado y pueden ocasionar daños o lesiones considerables.

---

<sup>2</sup>Aunque se inician en paralelo, la ESP32 puede presentar una pequeña latencia al iterar los comandos.