



Mantenimiento y elaboración de guías de laboratorio para pierna de robot cheetah

Daniel Esteban Ramirez Chiquillo

Universidad Nacional de Colombia
Facultad de Ingeniería, Departamento de Ingeniería mecánica y Mecatrónica
Bogotá, Colombia
2025

Mantenimiento y elaboración de guías de laboratorio para pierna de robot cheetah

Daniel Esteban Ramirez Chiquillo

Trabajo de grado presentado como requisito parcial para optar al título de:
Ingeniero Mecatrónico

Director:
Dr.-Ing. Ricardo Emiro Ramirez Heredia

Línea de Investigación:
Robótica
Grupo de Investigación:
Proyecto 58848

Universidad Nacional de Colombia
Facultad de Ingeniería, Departamento de Ingeniería Mecánica y Mecatrónica
Bogotá, Colombia
2025

Resumen

El presente trabajo de grado se enfoca en el diagnóstico, mantenimiento y mejora de una pierna robótica inspirada en la familia Cheetah, cuyo desarrollo se orienta a su uso en prácticas de laboratorio dentro del programa de Ingeniería Mecatrónica de la Universidad Nacional de Colombia. El estudio abarcó un análisis integral de los componentes mecánicos, eléctricos y de software del sistema. Durante la fase de diagnóstico se identificaron problemas tales como fracturas en componentes estructurales, limitaciones en el control debido al uso de Arduino, cableado desorganizado y la ausencia de un software operativo adecuado. Para solucionar estas incidencias se rediseñaron las piezas mecánicas, se optimizaron las conexiones eléctricas, se sustituyó el Arduino por una ESP32 y se reescribió el firmware utilizando el framework ESP-IDF. Además, se implementaron perfiles de movimiento y mecanismos de seguridad que permiten suavizar las transiciones y evitar golpes violentos. Paralelamente, se desarrollaron guías de laboratorio y se documentó detalladamente el uso, la arquitectura y las pruebas del sistema, facilitando su integración en el entorno educativo.

Abstract

This project focuses on the diagnosis, maintenance, and enhancement of a robotic leg inspired by the Cheetah platform, designed for use in laboratory sessions within the Mechatronics Engineering program at the Universidad Nacional de Colombia. The study involved a comprehensive analysis of the mechanical, electrical, and software components of the system. During the diagnostic phase, issues such as structural fractures, limitations in control due to the use of Arduino, disorganized wiring, and the absence of appropriate operational software were identified. To address these issues, mechanical components were redesigned, electrical connections were optimized, and the Arduino was replaced by an ESP32—leveraging its integrated CAN control capability—and the firmware was completely rewritten using the ESP-IDF framework. Additionally, movement profiles and safety measures (including the incorporation of a relay for forced motor shutdown) were implemented to smooth transitions and prevent abrupt movements. Furthermore, laboratory guides and comprehensive documentation was developed to teach the use, architecture, and testing of the system, facilitating its integration into the educational setting.

Contenido

1 Introducción	2
2 Diagnóstico	4
2.1 Explicación del montaje	4
2.2 Diagnóstico inicial	4
2.2.1 Diagnóstico de Hardware	4
2.2.2 Diagnóstico de Software	9
2.3 Otros problemas encontrados	9
2.3.1 Límite de memoria en Arduino	9
2.3.2 Instalación eléctrica deficiente	9
2.3.3 Seguridad	9
3 Corrección y mejora	11
3.1 Hardware	11
3.1.1 Mecánico	11
3.1.2 Eléctrico	15
3.2 Software	17
3.2.1 HTTP server	17
3.2.2 Control de posición	18
3.2.3 Seguridad y Apagado Forzoso	18
4 Plan de Pruebas y Validaciones	20
4.1 Pruebas básicas de API y seguridad	20
4.1.1 Uso básico	20
4.1.2 Desconectar la ESP32 durante ejecución	20
4.1.3 Desconectar un motor durante ejecución	20
4.2 Uso de la interfaz gráfica y scripts	21
4.2.1 Comandos a cada motor	21
4.2.2 Script de cinematica directa	21
4.2.3 Script de cinematica inversa	21
4.2.4 Script de pasos (caminata)	21
5 Documentación y guías de laboratorio	23
5.1 Documentación	23

Contenido	1
-----------	---

5.2 Guías de laboratorio	23
------------------------------------	----

6 Conclusión	25
---------------------	-----------

1 Introducción

La robótica se ha convertido en un área de gran interés dentro de la Ingeniería Mecatrónica, debido a su capacidad para automatizar procesos, optimizar sistemas y abrir nuevas líneas de investigación. Entre las plataformas robóticas que han marcado hitos en el desarrollo de robots cuadrúpedos se destaca la familia de robots Cheetah, impulsada por el Instituto Tecnológico de Massachusetts (MIT). Estas versiones sucesivas han sido pioneras al demostrar la viabilidad de movimientos rápidos y precisos, así como la posibilidad de navegar en terrenos complejos y realizar labores de manipulación en entornos exigentes. Cheetah 1 y 2 introdujeron mejoras en la eficiencia energética y la capacidad de salto, aprovechando motores de alto torque y diseños optimizados para minimizar las pérdidas de energía. Con Cheetah 3 se agregaron aplicaciones prácticas de rescate y respuesta a emergencias, expandiendo la robustez y la versatilidad del robot. Por su parte, el Mini Cheetah exploró la agilidad y la capacidad de ejecutar movimientos dinámicos complejos, respaldado por algoritmos de control y optimización de trayectorias.

Siguiendo estos desarrollos, el Departamento de Ingeniería Mecánica y Mecatrónica de la Universidad Nacional de Colombia cuenta con una pierna robótica inspirada en los principios del Cheetah. Esta pierna tiene un gran potencial para ser utilizada en prácticas de laboratorio dentro del currículo de Ingeniería Mecatrónica, pero requiere algunas mejoras antes de que pueda incorporarse plenamente al entorno educativo. El proyecto inicial de esta pierna surgió en la asignatura Proyecto Aplicado de Ingeniería (PAI), bajo la supervisión del profesor Ricardo Ramírez; en él, se lograron avances notables que permitieron un funcionamiento básico, pero quedaron diversos detalles por refinar.

En vista de lo anterior, el presente trabajo de grado tiene como objetivo diagnosticar el estado actual de la pierna, realizar los ensambles y la programación necesarios para su correcta puesta a punto, y diseñar guías de laboratorio detalladas para su uso en la carrera de Ingeniería Mecatrónica. Para lograrlo, se llevarán a cabo las siguientes acciones específicas:

- Analizar el estado de la pierna robótica, evaluando la parte mecánica, electrónica y de software.
- Identificar y corregir errores o fallas detectadas durante el proceso de análisis.
- Realizar pruebas para asegurar que el dispositivo funcione de manera confiable y eficiente.
- Redactar documentación que explique:

- Cómo usar el dispositivo.
- Diseño y arquitectura de software, para facilitar su desarrollo futuro.
- Desarrollar guías de laboratorio que faciliten la incorporación del robot en la enseñanza.

2 Diagnóstico

En este capítulo se detalla el estado del robot al inicio del trabajo de grado.

2.1. Explicación del montaje

Para facilitar la comprensión de este capítulo, se presenta un recorrido breve por el montaje del robot. La figura 2-1 muestra el estado general del montaje al comienzo del proyecto. Se identificaron los siguientes componentes:

- **Estructura de aluminio:** Perfiles 4040 que sostienen los motores y eslabones.
- **Fuente de alimentación:** Fuente de 24 V y 25 A (AC–DC) conectada a la red eléctrica para alimentar los motores.
- **Botón de emergencia:** Dispositivo conectado a la fuente que permite encenderla y apagarla rápidamente.
- **Interruptores:** Regulan el flujo de energía desde la fuente hacia cada motor.
- **Motores:** Elementos centrales encargados del movimiento de los eslabones.
- **Eslabones:** Piezas impresas en plástico que emulan la pata de un cuadrúpedo. Se dividen en 4: cadera, muslo, pierna baja y pie.
- **Polea y correa:** Conectan el motor 3 a la pierna baja.

2.2. Diagnóstico inicial

2.2.1. Diagnóstico de Hardware

Las figuras 2-1 a 2-12 ilustran el estado del robot durante el diagnóstico inicial.

Mediante el uso de un multímetro en modo continuidad se verificaron las conexiones entre la fuente de alimentación, los interruptores y los motores, constatándose que eran correctas.

No habían elementos de control en el montaje.



Figura 2-1: Foto del montaje inicial

Se revisó el estado general del montaje. Los perfiles de aluminio estaban en buen estado, aunque la organización de los componentes eléctricos y el cableado requería mejoras para evitar el desorden.

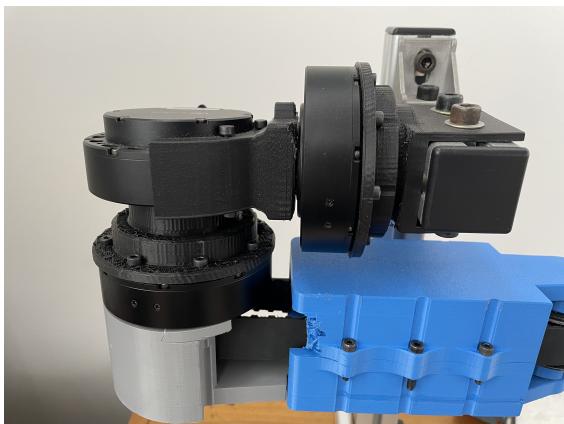


Figura 2-2: Foto de los motores

Los acoplos de los motores se encontraban en condiciones aceptables, aunque la calidad de impresión de algunos no era óptima.

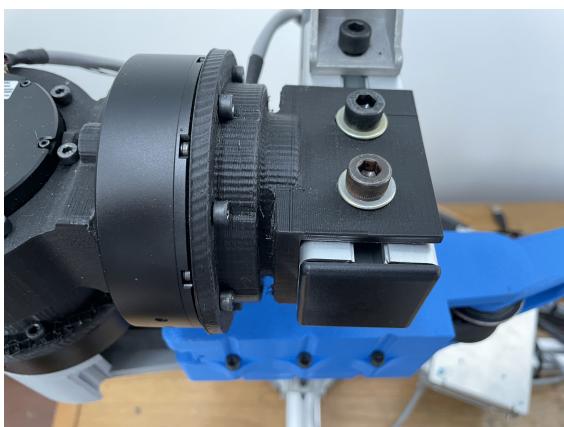


Figura 2-3: Foto del Motor 1

El *Motor 1* rotaba con facilidad. Su movimiento se veía limitado por la disposición del cableado.



Figura 2-4: Foto del Motor 2

El *Motor 2* funcionaba sin inconvenientes, limitándose únicamente por la estructura de aluminio.

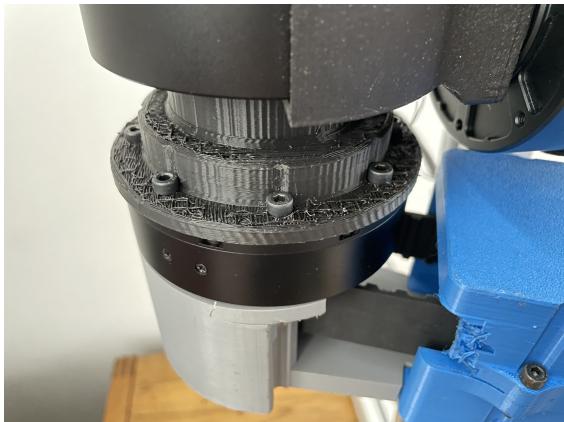


Figura 2-5: Foto del Motor 3

El *Motor 3* requirió mayor esfuerzo para moverse, atribuible a la elección de la correa y la polea, sin evidenciar fallas en el motor.

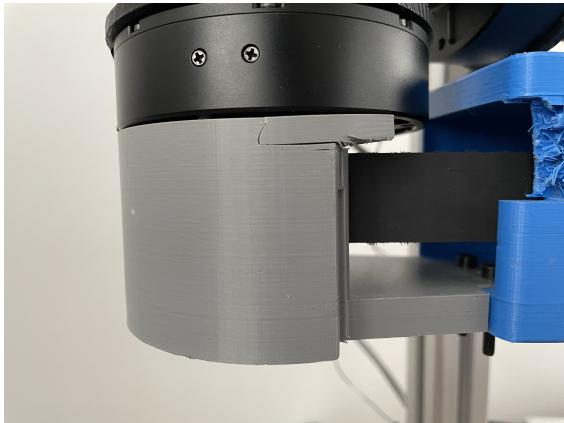
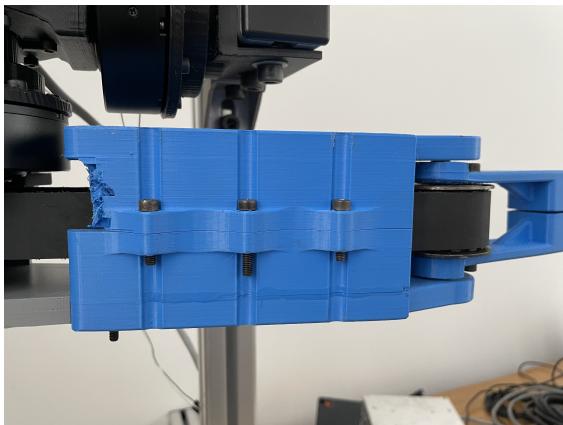


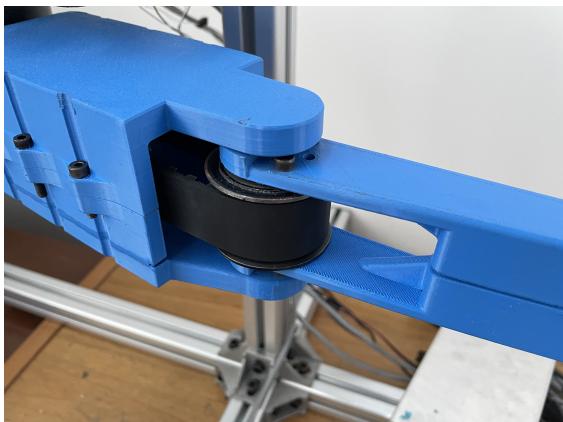
Figura 2-6: Foto de la Cadera

Se detectó una fractura considerable en la cadera.



El muslo estaba fracturado.

Figura 2-7: Foto del Muslo



La rodilla funcionó correctamente, sin presentar problemas en su giro.

Figura 2-8: Foto de la rodilla



La pierna inferior se encontraba en buen estado, salvo el uso de tornillos de longitud inadecuada.

Figura 2-9: Foto de la pierna inferior

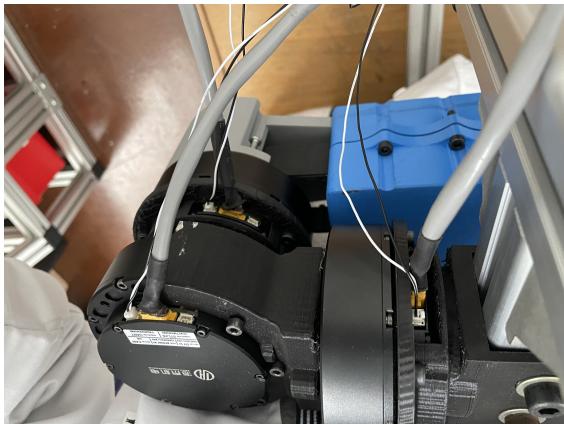


Figura 2-10: Foto de las conexiones de los motores

Se constató que la longitud de los cables en el sistema de conexión era inadecuada; algunos eran excesivamente largos, generando desorden, y otros, demasiado cortos, provocaban desconexiones en ciertos movimientos.



Figura 2-11: Foto de la fuente y el botón de emergencia

La fuente de alimentación y el botón de emergencia estaban sueltos en el montaje, contribuyendo al desorden. Al accionar el botón, se percibió una pieza suelta en su interior, sin afectar su funcionamiento.

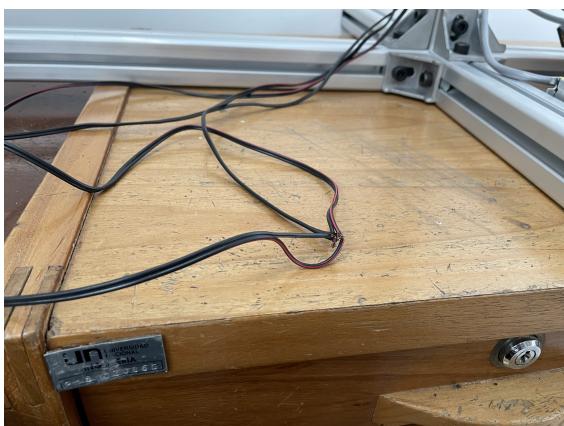


Figura 2-12: Foto de los cables CAN

Se observó que los cables del bus CAN estaban conectados entre sí, sin integrarse al resto del sistema.

2.2.2. Diagnóstico de Software

Según el informe del proyecto PAI, el código se encontraba en el repositorio de GitHub https://github.com/jolayam/proyecto_PAI. Sin embargo, tras revisar dicho repositorio no se encontró el software correspondiente. Aunque se recuperaron algunos archivos de la memoria SD de la Raspberry Pi, la mayoría de los necesarios para el control del robot estaban ausentes.

Ante la pérdida del software de control, se optó por reescribir por completo el código, ya que no era posible evaluar la funcionalidad del sistema debido al estado de las conexiones y a la falta de un software operativo.

2.3. Otros problemas encontrados

Durante el avance del proyecto se identificaron otros aspectos que requerían atención.

2.3.1. Límite de memoria en Arduino

Miembros del equipo PAI, relataron haber enfrentado diversos problemas al controlar la pata con un Arduino debido a su limitada capacidad de memoria. Por ello, se recomendó sustituir el Arduino por una ESP32.

2.3.2. Instalación eléctrica deficiente

Se encontraron conexiones deficientes que podían resultar peligrosas. Por ejemplo, la conexión mostrada en la figura 2-13 llegó a causar un cortocircuito. Afortunadamente, la fuente de alimentación contaba con mecanismos de seguridad que evitaron daños mayores.

2.3.3. Seguridad

Se detectó que los motores, al moverse con gran velocidad y fuerza, representaban un riesgo tanto para la integridad de las piezas del montaje como para las personas. Esto subraya la importancia de adoptar medidas de seguridad en el sistema.



Figura 2-13: Conexión peligrosa

3 Corrección y mejora

En este capítulo se describen las modificaciones realizadas al robot, tanto en hardware como en software.

En el repositorio del proyecto: https://github.com/danielrachi1/pierna_cheetah/ se recopilan todos los archivos del proyecto. Esto incluye archivos CAD, código del firmware, documentación y scripts de ejemplos de uso.

3.1. Hardware

3.1.1. Mecánico

Cadera, muslo y su acople

El diseño previo presentaba dimensiones excesivas y varios concentradores de esfuerzo. Como se explicó en el capítulo anterior, la cadera original llegó a fracturarse debido a uno de estos concentradores.

La forma de acoplar la cadera y el muslo consistía en un saliente en la cadera que se ajustaba en una trinchera del muslo. Esto provocaba que el peso de la pierna recayera únicamente sobre dicho saliente, volviéndolo frágil.¹

En la figura 3-1 se muestran imágenes del diseño anterior.

Para resolver estos problemas, se redujeron las dimensiones mediante una correcta medición de motor, polea y correa. Además, se eliminaron concentradores de esfuerzo y se reforzaron las zonas débiles. Por último, se transformó el acople en un marco completo con más agujeros para tornillos, incrementando la rigidez y la fijación en todas las direcciones.

La figura 3-2 ilustra el nuevo diseño.

Pierna baja y pie

Aunque la pierna baja no presentaba fallas significativas, se encontró una oportunidad de mejora: no contaba con un punto de fijación para herramientas o un pie. Esto resultaría problemático si el robot llegara a apoyar todo su peso en el plástico al caminar.

Se rediseñó la pierna baja para permitir atornillarle una herramienta, definiendo como interfaz un cubo de 20 mm x 20 mm x 20 mm con tres orificios para tornillos M4 dispuestos alrededor de un eje central, cuyos centros se ubican a 6 mm de dicho eje.

¹Esta parte se rompió durante uno de mis experimentos.

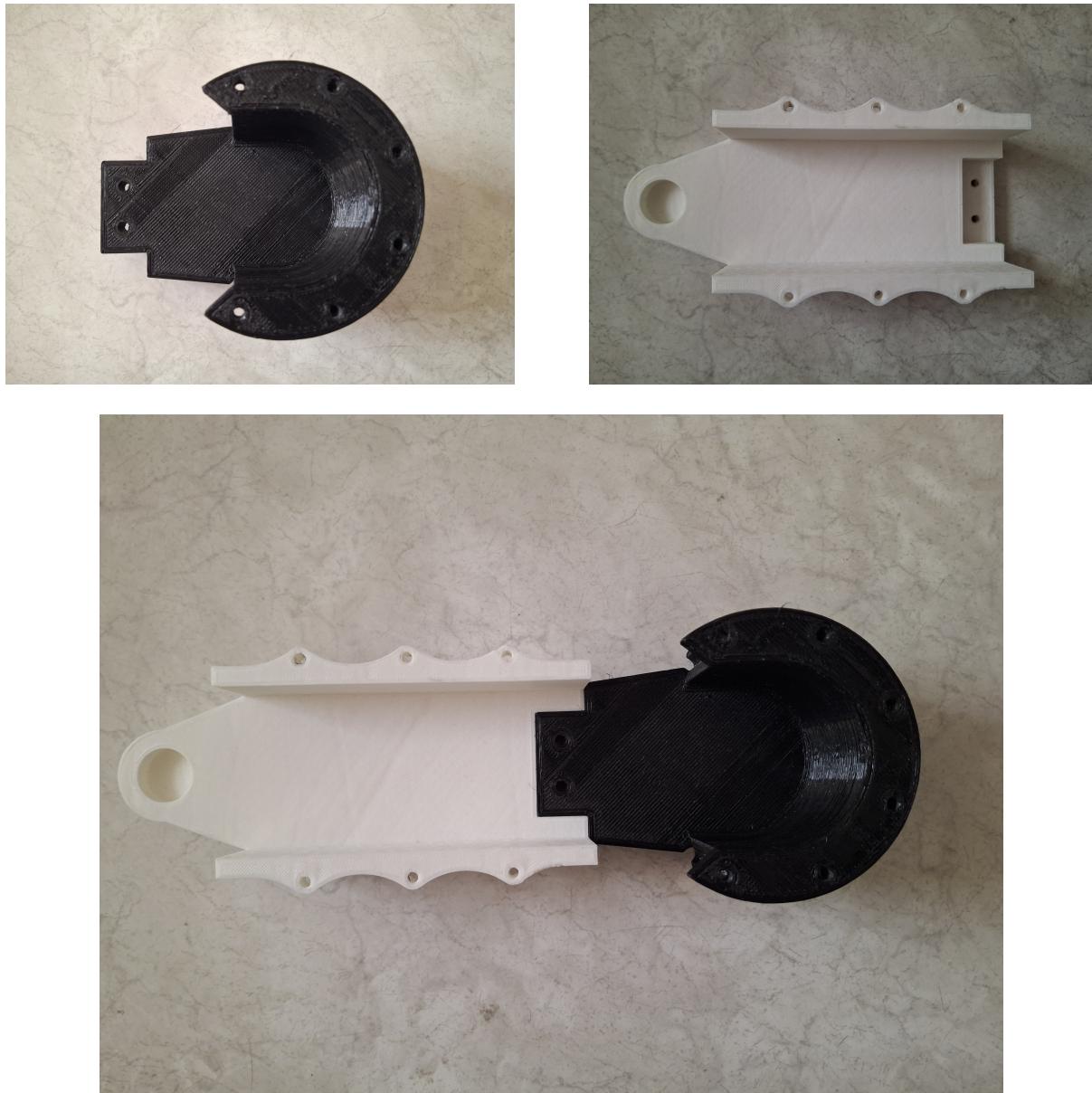


Figura 3-1: Diseño anterior: cadera, muslo y acople.

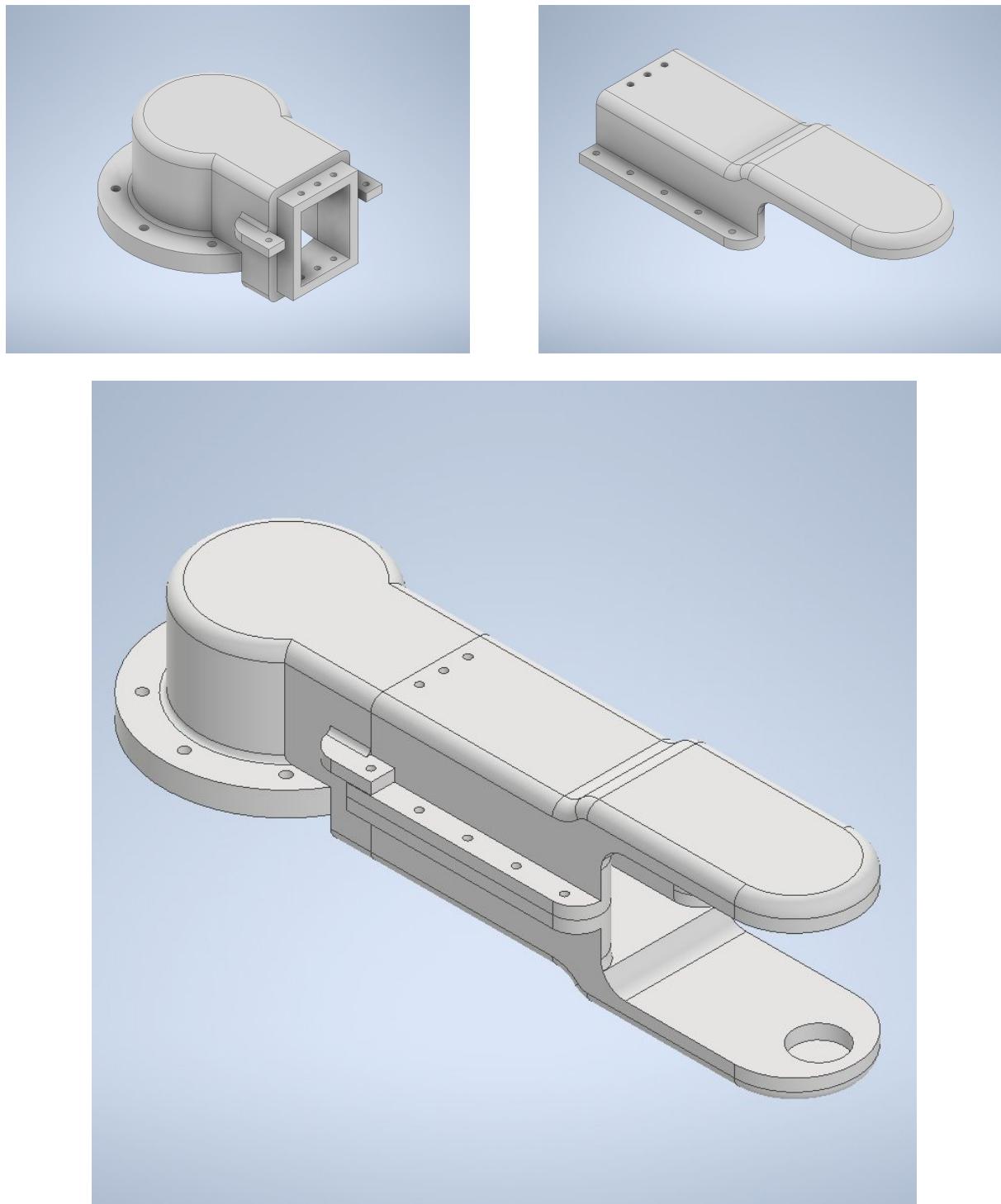


Figura 3-2: Nuevo diseño: cadera, muslo y acople.

Para validar la interfaz, se diseñó un pie sencillo que, además de la mencionada interfaz, posee un hueco esférico donde se incrusta parte de una pelota de squash, pegada a la herramienta.²

El nuevo diseño se muestra en la figura **3-3**.

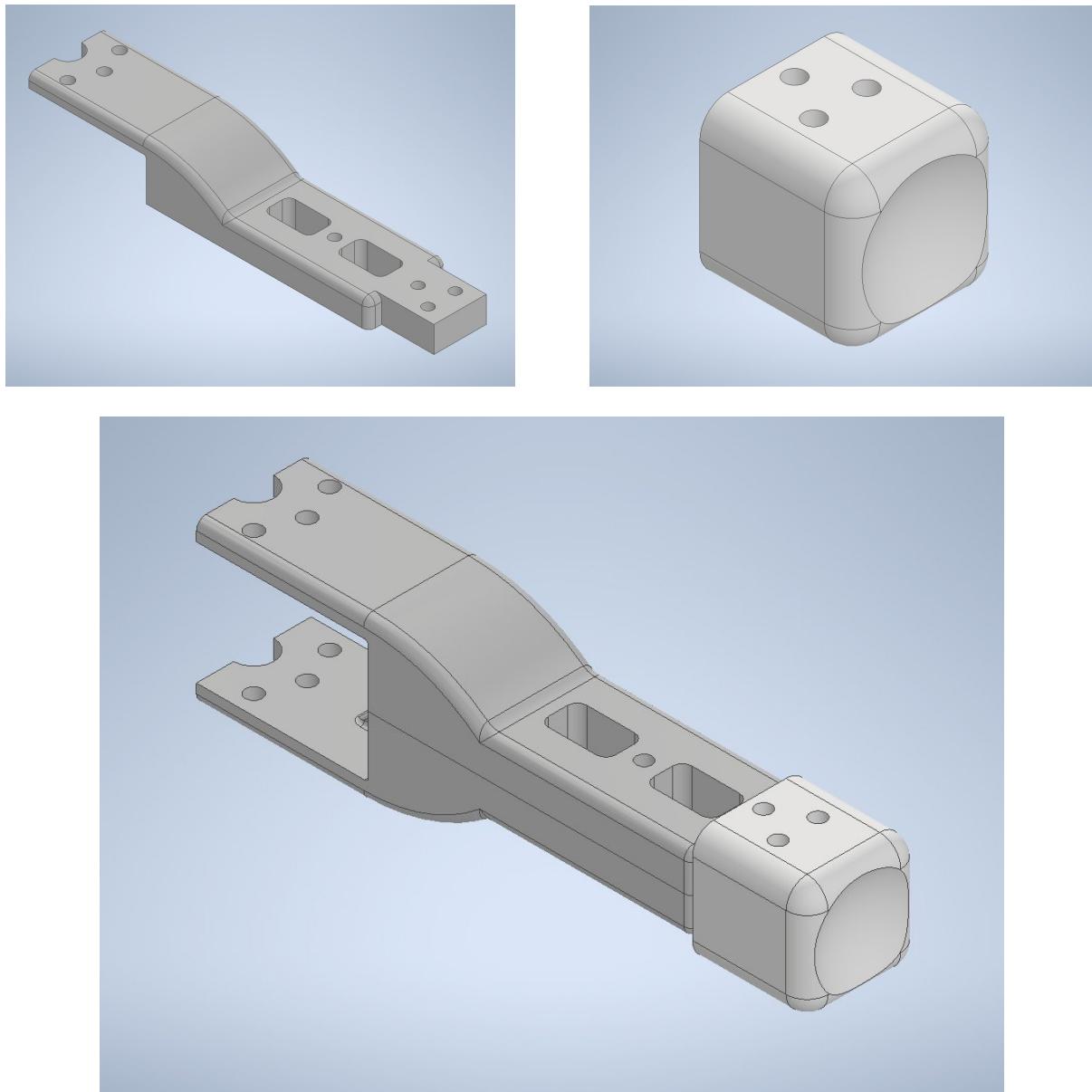


Figura 3-3: Nuevo diseño: pierna baja y pie.

²En los Cheetah del MIT se suelen usar pelotas de squash como “zapatos”. Estas ayudan a reducir el impacto del robot contra el suelo.

3.1.2. Eléctrico

ESP32

Siguiendo la recomendación de los miembros del equipo de PAI, se reemplazó el Arduino por una ESP32. Esta tarjeta, además de cumplir la función del Arduino, cuenta con Wi-Fi incorporado, por lo que se decidió dejar de utilizar la Raspberry Pi.

Nuevos módulos CAN

La lógica de 5 V del Arduino fue sustituida por la de 3,3V propia de la ESP32, que además ya incluye un controlador CAN (denominado “TWAI” por el fabricante). Por ello, se descartaron los módulos basados en MCP2515.CAN y se emplearon transceptores Sn65hvd230, compatibles con 3,3 V y sin controlador integrado.

Relés

Como se menciona en 2.3.3, era necesario un mecanismo que impidiera movimientos bruscos del robot. Para ello se agregaron relés que permiten cortar la alimentación de los motores, lo cual se detalla en 3.2.3.

Cables y conectores

Para mejorar la calidad de las conexiones, se procuró que:

- Los cables tuvieran longitud razonable.
- Todas las conexiones emplearan terminales y estuvieran bien aisladas.
- Los cables se agruparan de forma organizada.

Por ejemplo, en la figura: **3-4**, se muestra la versión arreglada de la conexión vista en **2-13**.

Organización

Además de un cableado seguro, era esencial fijar todos los componentes a una base. Se eligió una tabla de MDF de 4 mm, cortada con láser para ajustarla a la estructura. Una vez fija, se atornillaron los componentes sobre ella. La mayoría de módulos ya traían orificios de fijación. Para los *switches* se diseñó una carcasa impresa, y para ciertos circuitos se emplearon soportes plásticos en forma de “L”.

La figura **3-5** muestra la base con todos los elementos atornillados.

En la imagen **3-6** se puede observar el montaje final con todas las mejoras realizadas.



Figura 3-4: Conexión arreglada

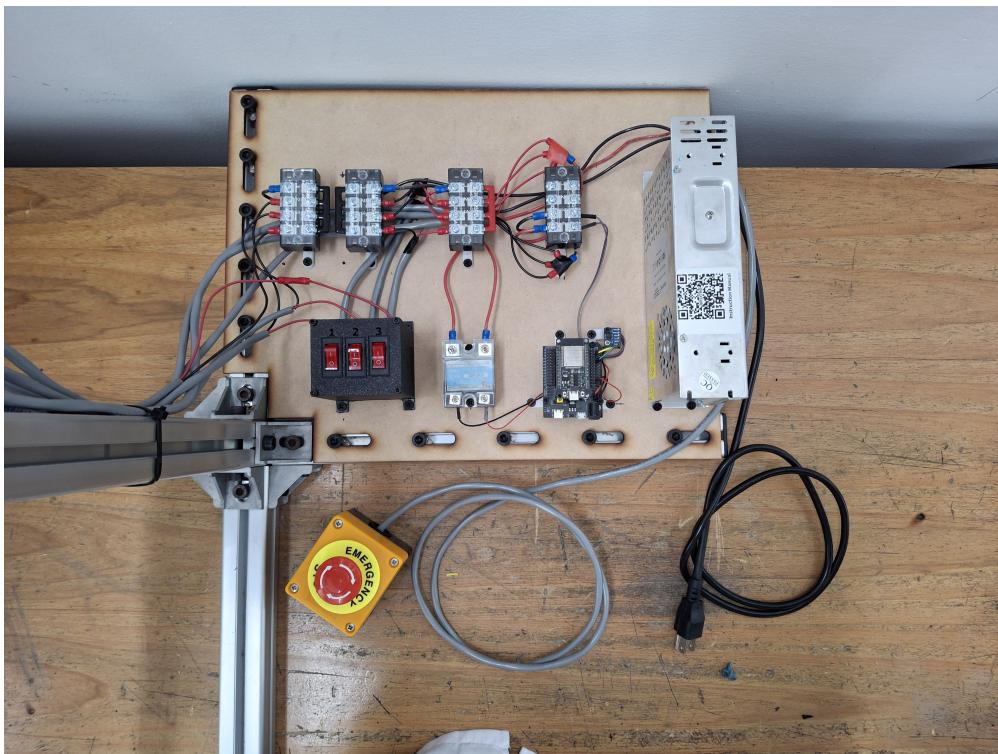


Figura 3-5: Base en MDF con los componentes atornillados.

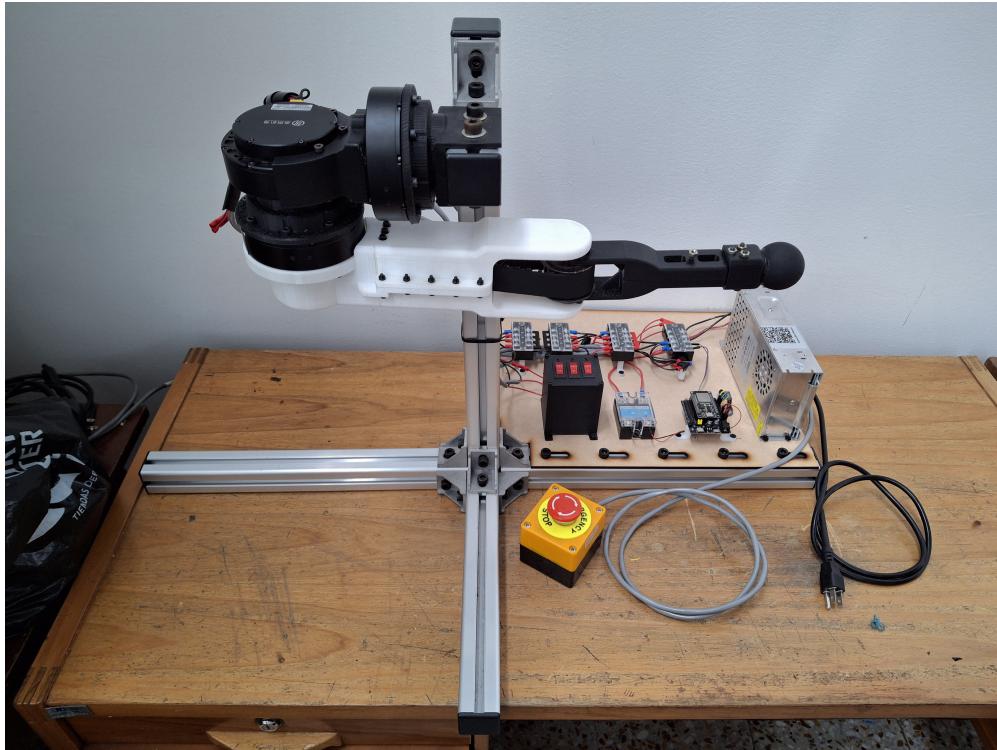


Figura 3-6: Montaje final del robot

3.2. Software

A raíz de las mejoras de hardware descritas, el proyecto migró a un solo proyecto de software: firmware de la ESP32, eliminando la necesidad de la Raspberry Pi y simplificando la arquitectura. El software se ha reescrito utilizando el *framework* oficial de Espressif (ESP-IDF), que ofrece una API robusta para Wi-Fi, GPIO, CAN (denominado “TWAI”) y otros periféricos.

Esta sección realiza un recorrido rápido del lo que permite hacer el nuevo software. Si se busca comprender el firmware a detalle, se recomienda leer el documento «Documentación de la Arquitectura del Firmware del Robot Cheetah». Disponible en el repositorio del proyecto.

3.2.1. HTTP server

Un servidor HTTP corre directamente en la ESP32. Este es el lugar en donde se interaccúa con el robot.

- **Rutas de Encendido/Apagado:**

- POST /api/robot/on: Activa el relé, envía el comando de cero de sensores, y pone los motores en modo activo.
- POST /api/robot/off: Mueve los motores a 0°, los desactiva y apaga el relé.

■ Rutas de Posición:

- POST `/api/command/move`. Recibe ID del motor, posición deseada y velocidad máxima del recorrido.
- POST `/api/command/batch`. Recibe un array de comandos iguales a los descritos en `/api/command/move`. El robot los ejecuta *en paralelo entre motores, secuencialmente para cada motor*.
- El firmware valida que la posición y la velocidad estén dentro de los límites permitidos para cada motor.

■ Página Web Estática:

- Incluye HTML, CSS y JavaScript para una interfaz donde el usuario ve 3 vistas: “Robot Off”, “Robot On” y “Recovery needed”, con los controles correspondientes para encender, apagar, mover motores, y realizar recuperación como se explica en 3.2.3.

Esta arquitectura elimina comandos directos de “enter mode” o “exit mode” en la interfaz, pues se agrupan en `/api/robot/on` y `/api/robot/off`, simplificando la experiencia del usuario y reduciendo riesgos de mal uso.

3.2.2. Control de posición

Aunque los motores internamente aceptan comandos de posición, se implementaron perfiles de movimiento para evitar desplazamientos violentos.

3.2.3. Seguridad y Apagado Forzoso

Los movimientos bruscos y la posibilidad de que la ESP32 o los motores quedaran mal apagados motivaron la introducción de múltiples protecciones:

Encendido/Apagado Único

Un componente central (llamado *robot_controller*) administra el estado global del robot.

Este componente se encarga de activar y desactivar banderas que indican situaciones cómo:

- Si los motores y el relé se encuentran activos o no.
- Si se realizó un apagado correcto la última vez que se utilizó el robot.

Forced Shutdown por Timeouts o Errores

Si un motor acumula **tres timeouts consecutivos** (no respondió antes del timeout a un mensaje CAN), el robot eliminará todos los comandos / trayectorias que tenga en espera, y se apagará.

Así se evita que el robot permanezca alimentado en un estado inestable de comunicación.

Perfiles Suaves e Impedimento de Comandos Peligrosos

Por defecto, cualquier comando “zero sensor” o “enter/exit mode”³ directo está bloqueado en la interfaz. Solo se ejecutan internamente cuando el usuario selecciona on/off, evitando secuencias inseguras. Esto, junto a los diferentes avisos de seguridad, hace que todos los movimientos del robot se vean dados por los perfiles de movimiento mencionados anteriormente.

Uso de Relés para la Energía de los Motores

Al apagar el robot, la ESP32 no solo envía el “exit motor mode” al final, sino que también desactiva el relé que alimenta físicamente los motores. Con ello se reducen los riesgos de movimientos involuntarios si la placa se reinicia o queda en un estado inestable.

³Estos comandos son nativos de los motores.

4 Plan de Pruebas y Validaciones

En este capítulo se describen las pruebas propuestas para garantizar el correcto funcionamiento del robot.

La descripción de las pruebas se mantiene concisa. Si se quieren más detalles acerca de cómo realizar cada uno de estos pasos, y de el comportamiento que se debe obtener, revisar la documentación del robot.

4.1. Pruebas básicas de API y seguridad

Para la primera serie de pruebas, vamos a probar la API y los mecanismos de seguridad del robot. Como vamos a realizar pruebas de seguridad, es necesario retirar los eslabones. Esta primera serie de pruebas consiste en 3 pruebas:

4.1.1. Uso básico

1. Encender el robot.
2. Enviar comandos de posición a todos los motores.
3. Apagar el robot.

4.1.2. Desconectar la ESP32 durante ejecución

1. Encender el robot.
2. Enviar un comando de posición a un motor.
3. Desconectar la ESP32.

4.1.3. Desconectar un motor durante ejecución

1. Encender el robot.
2. Enviar un comando de posición a un motor.
3. Apagar el motor mientras se está moviendo.

El desarrollo de esta serie de pruebas se puede observar en el siguiente video: <https://youtu.be/EpyHmZFmkk>.

4.2. Uso de la interfaz gráfica y scripts

Teniendo la seguridad de que los mecanismos del robot funcionan adecuadamente, podemos instalar los eslabones y realizar una segunda serie de pruebas.

En esta serie, vamos a utilizar probar la funcionalidad de la interfaz de usuario, y a correr algunos scripts que llaman varios endpoints de la API.

4.2.1. Comandos a cada motor

1. Encender el robot.
2. Enviar comandos de posición a todos los motores (desde la UI, uno por uno).
3. Apagar el robot.

4.2.2. Script de cinematica directa

1. Encender el robot.
2. Correr el script de cinemática directa, disponible en el repositorio del proyecto.
3. Apagar el robot.

4.2.3. Script de cinematica inversa

1. Encender el robot.
2. Correr el script de cinemática inversa, disponible en el repositorio del proyecto.
3. Apagar el robot.

4.2.4. Script de pasos (caminata)

1. Encender el robot.
2. Correr el script de pasos, disponible en el repositorio del proyecto.
3. Apagar el robot.

El desarrollo de esta serie de pruebas se puede observar en el siguiente video: <https://youtu.be/v7Muz0Q2M44>.

Todas las pruebas dieron un resultado satisfactorio.

El único problema encontrado fue en 4.2.4. En esta prueba, la trayectoria no fue tan continua como se esperaba. La razón por la que esto ocurre, es que el control implementado está optimizado para llegar a una posición y detenerse por completo en esa posición.

Esto hace que la transición entre los puntos de la trayectoria se note demasiado, ya que el robot:

- Llega a velocidad 0.
- Procesa la siguiente trayectoria a seguir.
- Inicia el recorrido de la trayectoria, siguiendo los perfiles de velocidad y aceleración.

Aunque teóricamente esto no debería significar un problema, la velocidad del procesador de la ESP32 hace notorio este proceso.

5 Documentación y guías de laboratorio

En este capítulo se hace un recorrido de los diferentes documentos relacionados al proyecto. Todos estos se pueden encontrar en la carpeta `docs/` del repositorio: https://github.com/danielrachi1/pierna_cheetah/tree/main/docs.

Es importante aclarar que en la carpeta se incluyen archivos PDF y ZIP para cada documento. Los archivos ZIP son proyectos Latex, que fueron descargados directamente de Overleaf. Y se pueden volver a importar si se necesita realizar alguna modificación en el futuro.

5.1. Documentación

La documentación del proyecto se dividió en 3 documentos:

1. «**Documentación de la pata del robot Cheetah**». Explica el uso básico del robot. Se encarga de guiar al usuario para poder trabajar con el robot, desde cero.
2. «**Referencia de la API de la pata del robot Cheetah**». Explica a detalle todos los endpoints a los que se pueden realizar peticiones a través de la API. Está dirigida a usuarios que ya leyeron y completaron los pasos de la documentación anterior, y que tienen experiencia interactuando con APIs HTTP.
3. «**Documentación de la arquitectura del firmware del robot Cheetah**». Explica el diseño del firmware de la ESP32 y cómo funciona internamente. Está dirigido a personas que quieran modificar el firmware.

5.2. Guías de laboratorio

Se escribieron 3 guías de laboratorio:

1. **Cinemática directa.** En la cual los estudiantes encontrarán los parámetros DH del robot, lo simularán en Matlab, aprenderán a usar el robot, y enviarán sus primeros comandos por API.
2. **Cinemática inversa.** En la cual los estudiantes obtendrán las ecuaciones de cinemática inversa del robot, y programarán un script que acepte puntos (x,y,z) y envíe comandos al robot.

3. **Rutinas de movimiento.** En donde los estudiantes planearán rutinas de movimiento del robot y usarán los scripts de los dos laboratorios anteriores para ejecutar estas rutinas en el robot.

6 Conclusión

Las pruebas realizadas tras la implementación de las mejoras evidenciaron que los problemas relacionados con movimientos bruscos y golpes violentos se han solucionado de manera satisfactoria. La introducción de perfiles de movimiento, junto con una secuencia de arranque segura y la inclusión de un relé que corta la alimentación de los motores en situaciones críticas, garantizó la estabilidad del sistema. Asimismo, el rediseño mecánico y la optimización del cableado han reforzado la estructura del dispositivo, mejorando su confiabilidad. Por otro lado, la reestructuración del software mediante el uso del ESP-IDF ha simplificado la arquitectura del sistema—eliminando la dependencia de una Raspberry Pi—y ha proporcionado una interfaz de control intuitiva a través de un servidor HTTP. En conjunto, estos avances posicionan a la pierna robótica como una herramienta segura y robusta para la realización de prácticas de laboratorio y futuros desarrollos en el área de la robótica y la mecatrónica.

Cabe destacar que, pese a estos avances, se identificaron áreas de mejora en el diseño del procesador de comandos y en el control de movimiento. En particular, la prueba de caminata reveló que las transiciones entre los puntos de la trayectoria no fueron tan fluidas como se esperaba, lo que sugiere la necesidad de optimizar la interpolación y ejecución de los perfiles de movimiento. Este aspecto representa un desafío adicional que, de ser abordado en futuras investigaciones, podría lograr una respuesta más continua y dinámica, incrementando aún más la robustez y eficiencia del sistema.