POLITÉCNICA

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES
UNIVERSIDAD POLITÉCNICA DE MADRID

José Gutiérrez Abascal, 2. 28006 Madrid
Tel.: 91 336 3060
info.industriales@upm.es

www.industriales.upm.es

Raúl Molina Gómez

05 TRABAJO FIN DE GRADO

INDUSTRIALES

TRABAJO FIN DE GRADO

SENSORIZACIÓN Y CONTROL DE UN ROBOT BLANDO MANIPULADOR

TRABAJO FIN DE GRADO PARA LA OBTENCIÓN DEL TÍTULO DE GRADUADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

SEPTIEMBRE **2023**

Raúl Molina Gómez

DIRECTORES DEL TRABAJO FIN DE GRADO:
Antonio Barrientos Cruz
Jorge F. García-Samartín

INDUSTRIALES
ETSII | UPM

POLITÉCNICA

UNIVERSIDAD
POLITÉCNICA
DE MADRID

Universidad Politécnica de Madrid

Escuela Técnica Superior de Ingenieros Industriales

Grado en Ingeniería en Tecnologías Industriales

Trabajo Fin de Grado

Sensorización y Control de un Robot Blando Manipulador

Autor: **Raúl Molina Gómez**

Tutor Académico: **Antonio Barrientos Cruz**

Cotutor: **Jorge F. García-Samartín**

septiembre, 2023

Agradecimientos

En primer lugar, quiero expresar mi agradecimiento a todas las personas cuyo apoyo ha sido muy valioso para mí tanto durante la realización de este proyecto como durante todos mis estudios universitarios.
Agradezco a mis padres por haberme brindado siempre la mejor educación posible, por haberme ayudado cuando más lo necesitaba y por los excelentes valores personales basados en el esfuerzo, el respeto y la humildad que me han inculcado.
Agradezco a mi hermana por su ayuda y apoyo durante toda la carrera universitaria.
Agradezco también a todos mis amigos y compañeros de carrera, pues sin ellos no sería quien soy actualmente.

Asimismo, quiero agradecer a todas aquellas personas que me han ayudado a llevar a cabo este proyecto.
Agradezco a Antonio Barrientos, por confiar en mí y darme la oportunidad de realizar este Trabajo Fin de Grado en el laboratorio Robcib. También quiero agradecer su ayuda y la motivación que me ha dado en los momentos de mayor desesperación.
Agradezco de manera especial a Jorge García-Samartín por su paciencia, amabilidad, tiempo y guía durante este trabajo. Sin ti no habría sido posible completar este proyecto.
Agradezco también a todos los que me habéis acompañado durante toda la realización del proyecto en el laboratorio Robcib. Chrystian, David e Ysmaldo, me habéis acogido como un doctorando más y me habéis ayudado a conseguir este logro creando un clima de trabajo extraordinario.

Raúl Molina Gómez

Resumen Ejecutivo

En la actualidad, el desarrollo de nuevos robots que permitan realizar y automatizar cada vez más tareas se ha convertido en una prioridad en la industria robótica. Estas investigaciones se enfocan tanto en nuevos sistemas de control como en el diseño y fabricación de robots con nuevos materiales. Es así como surge hace aproximadamente 20 años un nuevos campo de la robótica: la robótica blanda o *Soft Robotics*.

Este nuevo tipo de robots surge con el objetivo de crear máquinas cuyas propiedades y comportamiento se asemejen a los organismos biológicos y tejidos blandos. Este nuevo enfoque se aleja completamente del concepto de robótica que se venía empleando en las épocas pasadas, pues no se usan para su construcción materiales rígidos.

Estos factores en su diseño y estructura les otorgan una gran ventaja: son capaces de interactuar de manera segura y eficiente con entornos cambiantes y complejos. Esto permite su uso en múltiples aplicaciones hasta ahora extremadamente complejas de realizar, como el trabajo con humanos o la adaptación a terrenos de difícil acceso para un robot móvil tradicional. Un ejemplo de esto último serían los escombros de un edificio o tuberías estrechas y serpenteantes.

Son todas estas potenciales aplicaciones para la sociedad las que motivan la investigación acerca de esta nueva tecnología, como es el caso de este proyecto. Así, el objetivo que presenta este Trabajo Fin de Grado es el de realizar sobre el diseño del Robot *PAUL*, fruto del trabajo de Adrián Rieker [1], un sistema de control en posición en cadena cerrada para uno de los segmentos que lo componen. Para ello se precisa además modificar ciertos aspectos del robot, como su diseño o proceso de fabricación, así como seleccionar y caracterizar el método de sensorización a emplear.

Cabe señalar que el Robot *PAUL* se compone de varios módulos blandos en serie, cada uno de los cuales es actuado mediante 3 vejigas neumáticas que, al deformarse por efecto del aire a presión que se introduce en ellas, consiguen modificar la forma del módulo. Para lograrlo, y debido a la falta de sensores de presión, la cantidad de aire comprimido introducido en cada módulo se regula mediante el tiempo de apertura de los circuitos de

admisión y escape del aire. A este tiempo de apertura se le ha denominado como *tiempo de actuación*.

Para completar el bucle de control en cadena cerrado se han empleado unos sensores resistivos elastómeros comerciales (figura R1), fabricados por la empresa Adafruit [2]. Estos sensores permiten medir cambios en sus resistencias producidos por variaciones en su longitud. De esta forma, se mide de manera indirecta la deformación y forma final de cada uno de los segmentos del robot a los que están fijos.

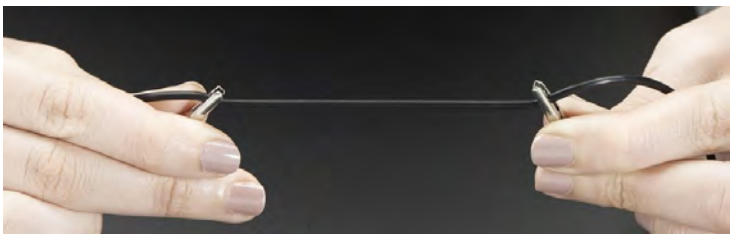


Figura R1 Sensores empleados para el sistema de control en cadena cerrada del robot. *Fuente [2]*

Estas medidas se realizan gracias a un hardware específico de medición y monitoreo de parámetros eléctricos denominado *INA3221*. Este dispositivo permite obtener medidas de tensión, corriente y/o potencia con elevada precisión gracias a un filtro paso bajo y un convertidor analógico-digital. Además, cuenta con tres canales de medida, con lo que se pueden realizar mediciones de los tres sensores que incorpora cada segmento. Así, la técnica empleada consiste en alimentar mediante una fuente de tensión un divisor de tensiones compuesto por una resistencia fija y un sensor, lo que permite medir las variaciones de tensión que sufre el sensor cuando se produce la actuación y correspondiente deformación del segmento.

Asimismo, para la caracterización de los sensores se ha diseñado y construido un banco de pruebas (figura R2) que estira de manera constante y uniforme el sensor. De esta forma se puede extraer la relación entre la tensión medida y su variación de longitud. A su vez, para su fijación e implementación en el robot, se ha usado un sellador adhesivo de silicona denominado comercialmente *DOWSIL 732*.

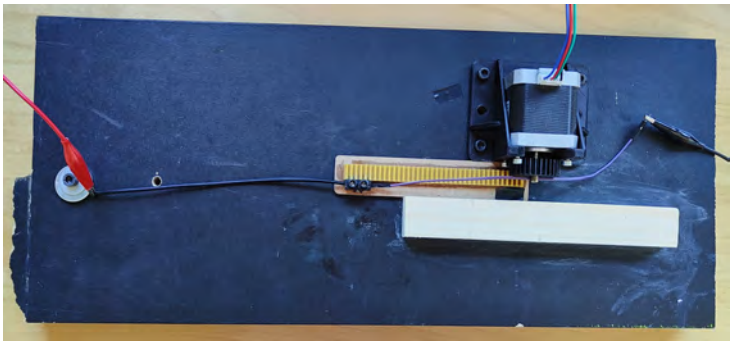


Figura R2 Banco de pruebas diseñado para caracterizar el comportamiento de los sensores. *Fuente: elaboración propia*

Por su parte, el sistema de control emplea el diagrama de bloques de la figura R3 para que, mediante la introducción de un punto objetivo contenido en el espacio de trabajo del robot, se consiga mover dicho robot al punto deseado obteniendo un error mínimo. Este sistema surge como resultado de numerosas pruebas e iteraciones de búsqueda y perfeccionamiento del método de control adecuado. La justificación a este largo proceso es la complejidad presente en el modelado de los *Soft Robots*, pues su estructura continua y maleable les otorga una característica nunca vista en la robótica tradicional que es la inexistencia de cadena cinemática, o lo que es lo mismo, se podría considerar que estos robots presentan infinitos grados de libertad.

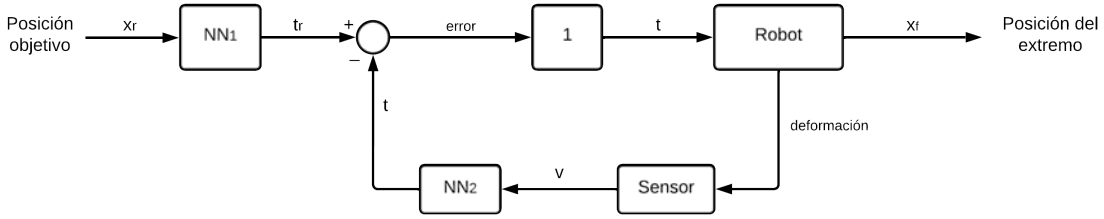


Figura R3 Diagrama de bloques empleado para el control del segmento. *Fuente: elaboración propia*

La resolución a este problema finalmente pasa por el entrenamiento de dos redes neuronales prealimentadas o *feedfoward*. En el caso de la primera de ellas (la denominada como NN_1), transforma los valores de posición introducidos en tiempos de actuación objetivo, mientras que la segunda (NN_2) convierte las mediciones de tensión en tiempos de actuación del robot. Así, se suprime la necesidad de un regulador multivariable.

Para su obtención, es necesario realizar el entrenamiento de las redes usando múltiples datos contenidos en *datasets* del comportamiento del robot. Así, se ha diseñado además un método de obtención de estos datos (tensiones, posiciones y tiempos) de forma automática, basado en el método de captura de posiciones por visión con computador ideado por Adrián Rieker [1]. Uno de los resultados de esta captura de datos se ve en la figura R4, que muestra el espacio de trabajo del robot.

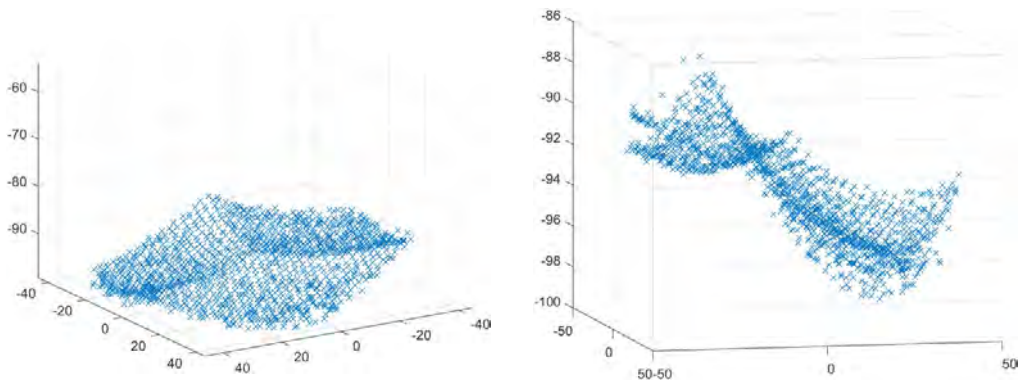


Figura R4 Espacio de trabajo del robot desarrollado. *Fuente: elaboración propia*

Durante esta fase del proyecto, se observaron además múltiples problemas relacionados con el diseño y la fabricación del robot, como las numerosas fugas de aire que se producían. Es por ello que se acometieron como un objetivo adicional del trabajo modificaciones en el diseño, la programación y el proceso de fabricación del robot.

Finalmente, y tras las numerosas iteraciones de obtención de datos y diseño y entrenamiento de las redes, se procedió a realizar varias pruebas y experimentos para verificar el cumplimiento de los objetivos. Los resultados obtenidos fueron muy satisfactorios, logrando errores de en torno a los 5mm, obteniendo así un robot que incluso supera en precisión a otros robots similares desarrollados en la literatura.

Con todo ello, el trabajo final realizado se puede ver en la figura R5.

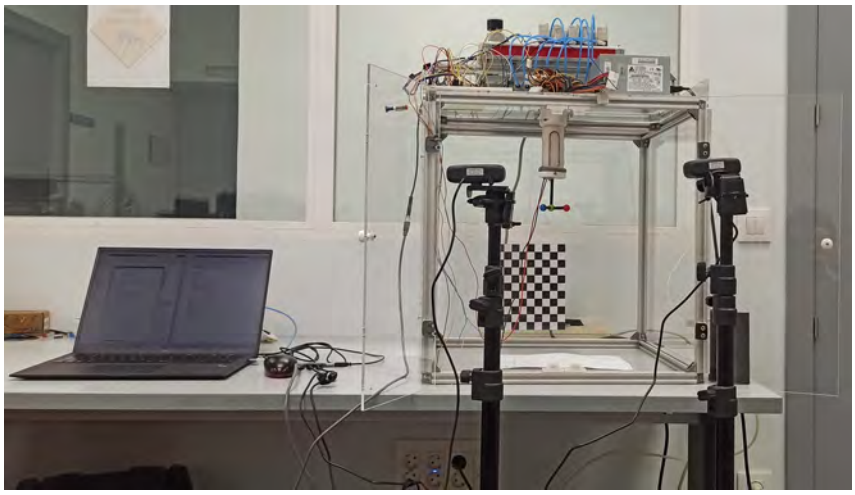


Figura R5 Imagen del entorno del robot durante su desarrollo. *Fuente: elaboración propia*

Códigos UNESCO

1203 Ciencia de los ordenadores
- 1203.04 Inteligencia Artificial
- 1203.25 Diseño de sistemas sensores

3304 Tecnología de los ordenadores
- 3304.12 Dispositivos de control
- 3304.19 Robótica

3311 Tecnología de la Instrumentación
- 3311.01 Tecnología de la automatización
- 3311.02 Ingeniería de control
- 3311.07 Instrumentos electrónicos

Palabras clave

Robótica blanda, Soft Robot, manipulador, sensor resistivo elastómero, red neuronal, feedforward neuronal network, control en posición, MATLAB, Arduino, silicona.

Índice

Índice de Figuras

Índice de Tablas

Capítulo 1

Introducción

Desde la aparición de los primeros homínidos, el ser humano siempre ha mostrado un gran interés por idear métodos que permitieran realizar las tareas más complejas de una forma sencilla. Es gracias a esta necesidad de avance lo que permitió la invención de la rueda, la fabricación de las primeras herramientas de trabajo o la construcción de las primeras edificaciones modernas.

No será sin embargo hasta la Revolución Industrial cuando el ser humano verá en la automatización de los procesos una necesidad para lograr mayores avances en la sociedad. Es así como surgen los primeros robots, en forma de computadoras analíticas, que permitían la realización de cálculos matemáticos a una velocidad muy superior a la del ser humano. Fue gracias al desarrollo de las computadoras durante el siglo XX lo que permitió finalmente el desarrollo de los primeros robots industriales, dotados de mecanismos que permitían la realización de las tareas manuales de un humano a una velocidad asombrosa.

Desde entonces, los robots están presenten en todas las facetas de la vida, desde la fabricación de un automóvil o el transporte de las personas hasta la limpieza del hogar o, incluso, el cocinado de alimentos. Es precisamente esta capacidad de adaptación a las distintas tareas las que han motivado desde su aparición el desarrollo de nuevos robots.

Es gracias a la investigación de nuevos tipos de robots lo que permitirá en el año 2000 la aparición de los primeros **robots blandos (*Soft Robots*)** en el mundo. Desde entonces, la robótica blanda se ha convertido en una de las tendencias que más interés genera entre la comunidad científica, como muestra la figura 1.1.

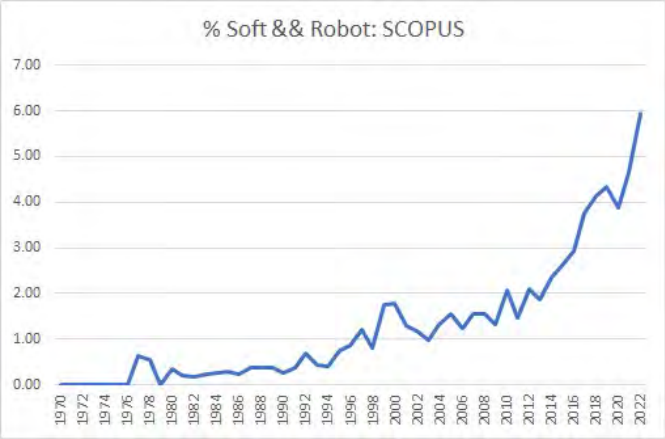


Figura 1.1 Aumento de las búsquedas de carácter académico relacionadas con la robótica blanda. *Fuente: Scopus*

Este nuevo campo de la robótica busca crear robots y dispositivos con propiedades y características similares a las de los organismos biológicos y tejidos blandos. Esto se aleja completamente del concepto de robótica que se venía empleando en las épocas pasadas, pues no se emplean para su construcción materiales rígidos.

Esta estructura flexible otorgan a estos robots unas cualidades excepcionales que justifican su construcción y desarrollo. Una de ellas es su estructura blanda, pues a diferencia de los robots rígidos convencionales, los robots blandos están diseñados para ser flexibles, por lo que son capaces de interactuar de manera segura y eficiente con entornos cambiantes y complejos. Esto les permite trabajar en conjunción con humanos reduciendo al mínimo los accidentes laborales. Además, su estructura maleable les permite introducirse en huecos y adaptarse a terrenos en los que un robot móvil tendría dificultades en la accesibilidad, como en los escombros tras el derrumbamiento de un edificio o en la exploración de tuberías.

Son todos estos aspectos los que motivan su desarrollo y justifican su creciente investigación tanto en universidades como en empresas del sector de la robótica.

1.1. Motivación

La robótica blanda (Soft Robotics), es un área reciente de la robótica que ha cobrado gran interés por sus ventajas en determinadas aplicaciones, como en aquellas donde hay un contacto directo con seres vivos. Su desarrollo presenta retos complejos, tanto en lo relativo a su fabricación (materiales, modos constructivos,...), como a su actuación (actuadores blandos) y control, pues su falta de rigidez origina que el movimiento del actuador y el robot estén en cierta medida desacoplados.

En el Centro de Automática y Robótica (CAR) se ha abordado el desarrollo y control de diferentes tipologías de robots blandos. Una de ellas es la de robots manipuladores con accionamiento neumático. Como resultado de ello se cuenta con algún prototipo funcional

de robot y de sus sistema de accionamiento. Su control, sin embargo, es aún incipiente, realizándoselo en cadena abierta en base a un modelo no explicito, obtenido experimentalmente y formulado mediante una red neuronal. Para avanzar en la investigación en robótica blanda en general y en particular en el control del robot manipulador, es preciso avanzar en su control, utilizando la realimentación de su localización y evaluando diferentes opciones de técnicas de control en cadena cerrada.

Son estos retos los que motivan el desarrollo de este proyecto, pues sus aplicaciones futuras pueden ser ampliamente variadas. Además, el trabajo aquí realizado puede ayudar al desarrollo de la industria de la robótica blanda y de la sociedad en un futuro.

1.2. Objetivos

Este trabajo tiene como finalidad el diseño, desarrollo y verificación de un sistema de control para el robot neumático desarrollado por Adrián Siegbert Rieker [1]. Este sistema debe ser capaz de posicionar el extremo del robot con la mayor precisión posible.

Para lograrlo, es necesario además realizar la caracterización y selección de los sensores que permitan cerrar el bucle de control en cadena cerrada y su implementación en el robot. Para ello, se espera realizar cambios en su diseño y técnica de fabricación.

Adicionalmente, se desea estudiar su aplicabilidad a futuros proyectos de desarrollo del control del robot de forma modular o segmentada.

De esta forma, los objetivos del Trabajo Fin de Grado se pueden resumir en los siguiente puntos:

- Diseño y desarrollo de un sistema de control en posición en cadena cerrada.
- Verificación del funcionamiento de dicho sistema de control.
- Selección y caracterización de los sensores que permiten el bucle de control.
- Implementación de los sensores en el robot.
- Rediseño del método de fabricación de los segmentos para solventar problemas surgidos y adaptarlos a las necesidades actuales.
- Estudio de la aplicabilidad a futuros proyectos de desarrollo del control del robot de forma modular o segmentada.

1.3. Proyectos relacionados - El robot PAUL

Para la realización de este proyecto se ha empleado el robot llamado *PAUL* diseñado por Adrián Rieker durante su Trabajo Fin de Grado [1]. Es por ello que en muchas

ocasiones durante el trascurso de esta memoria se darán por supuestas determinadas características o factores del robot.

Entre los avances ya realizados para este robot se encuentran:

- El diseño y método de fabricación de los segmentos uq ecomponen el robot.
- Los diseños de las piezas que se emplean como moldes en la fabricación del robot.
- Los diseños de las piezas que unen los distintos segmentos del robot, así como las piezas que permiten sujetar el robot al utillaje externo.
- El utillaje que actúa como soporte del robot. Este se componía inicialmente de una estructura formada por perfiles de aluminio. Posteriormente, Antonio Barrientos junto a los doctorandos Jorge García-Samartín e Ysmaldo Landáez diseñaron y montaron sobre dicho utillaje unas placas de metacrilato que actuaban como cerramiento y protección del robot.
- El banco de actuación, conformado por electroválvulas que permiten el hinchado o deshinchado de los segmentos del robot, mosfets que se encargan de mandar las señales de apertura y cierre de las electroválvulas y un circuito neumático, incluyendo compresor y tubos neumáticos, que permiten el llenado de las cavidades del robot.
- Todo el software y hardware relacionado con la visión por computador posteriormente empleada para la captura de las posiciones del extremo del robot.
- El software de actuación del robot, incluyendo una interfaz de control manual y mediante tabla.

El robot final desarrollado por Adrián Rieker previo a este proyecto se puede ver en la figura 1.2.

Figura 1.2 Robot final desarrollado en el trabajo previo de Adrián Rieker. *Fuente [1]*

Sobre estos avances se realizará la sensorización y el desarrollo de un sistema de control automatizado para el robot PAUL.

1.4. Estructura del trabajo

La memoria aquí expuesta engloba todos los pasos en el desarrollo de este proyecto.

Esta comienza con el **capítulo 2. Estado del Arte**. En él, se contextualizará el trabajo realizado en este proyecto y se comentarán todos los avances y actuales tecnologías que presenta la robótica blanda (*Soft Robotics*) en materia de sensorización, modelado y control.

Después, con el **capítulo 3: Estudio y caracterización de los sensores**, se explicarán todos los requerimientos en materia de sensorización del robot, la elección de los sensores, materiales y hardware de medida necesarios y, finalmente, se expondrán los resultados obtenidos del comportamiento ante distintos estímulos.

Tras ello, se introducirá el **capítulo 4: Herramientas utilizadas**. En él, se explicarán con mayor detalle las herramientas de *Machine Learning* empleadas durante el proyecto, así como sus fundamentos teóricos.

Seguidamente, el **capítulo 5: Modificaciones del diseño, fabricación y control de los segmentos**, se expondrán los cambios de fabricación y diseño sobre el robot ya existente con el objetivo de solventar múltiples problemas acontecidos durante el desarrollo del trabajo.

Finalmente, en el **capítulo 6: Diseño y desarrollo del sistema de control para un segmento** se expondrá el desarrollo del sistema de control y su funcionamiento, comprobando su efectividad y precisión en el posicionamiento en el **capítulo 7: Pruebas y resultados**.

Asimismo, también se han incluido el **capítulo 8: Conclusiones y líneas futuras**, así como los **apéndices A: Estudio del impacto ambiental y social** y **B: Planificación temporal y presupuesto**.

Por último, se incluye en el **apéndice C: Código de interés** algunas de las implementaciones más relevantes en materia software del poryecto.

Capítulo 2

Estado del arte

En este capítulo se presentarán algunos de los avances más representativos acerca de sensorización y control en el campo de la robótica blanda. También se evaluarán las aportaciones a esta industria de cada una de las técnicas empleadas en los distintos proyectos aquí expuestos.

2.1. Sensorización en robótica blanda

La sensorización en la robótica blanda dista mucho de la empleada en la robótica tradicional. La principal causa de este motivo es su estructura maleable y flexible, que le impide hacer uso de los sensores más comunes como, por ejemplo, *encoders*, acelerómetros, galgas extensiométricas o unidades de medición inercial (IMU). Es por ello que, en los últimos años y siguiendo el acelerado crecimiento de los *Soft Robots* en la industria robótica, se han desarrollado nuevos métodos de sensorización, utilizando muy diversos tipos de materiales.

Debido a esto, se han definido una serie de requisitos que deben cumplir los sensores utilizados en robótica blanda [3]. Estos requisitos son:

1. Permitir el movimiento del *Soft Robot*, sin interferir en su comportamiento normal ante los estímulos de actuación.

2. Ser lo suficientemente resilientes y flexibles para evitar roturas por fallo tras un elevado número de ciclos de movimient

3. Carecer de características que actúen como concentradores de tensiones o que puedan causar daños en la estructura del robot o en el espacio de trabajo.

Además, los valores en sus mediciones deben ser fácilmente computables y suficientemente precisos para la correcta realización de la tarea a completar por el robot.

En este sentido, sensores con un bajo módulo elástico que permitan seguir y detectar los movimientos de flexión de los *Soft Robots* suelen ser la elección principal. A este grupo también pertenecen aquellos sensores que emplean materiales en fase líquida, pues poseen un alto grado de maleabilidad.

Sin embargo, han surgido otras técnicas de fabricación de sensores que emplean algunos de los materiales más avanzados y vanguardistas con el objetivo de optimizar y ampliar el abanico de posibilidades en este ámbito. Estos nuevos métodos de sensorización se pueden clasificar atendiendo al método de detección de estímulos en 6 principales grupos [4]:

1. Sensores resistivos
2. Sensores capacitivos
3. Sensores ópticos
4. Sensores magnéticos
5. Sensores inductivos
6. Sensores basados en corrientes de Foucault

2.1.1. Sensores resistivos

Los sensores pertenecientes a este grupo son habitualmente la primera elección cuando se trata de conocer el comportamiento y la posición de un *Soft Robot*.

Estos sensores poseen unas características físicas que les permiten modificar su resistividad en función de los estímulos recibidos del exterior, como cambios en la presión ejercida sobre ellos, cambios en su elongación o incluso cambios de temperatura. De esta forma, mediante la correcta intrepretación y medición de estas variaciones en la resistencia del material se puede conocer la posición e, incluso, la forma que presenta el *Soft Robot*.

En este grupo destacan sobre el resto dos tipos de sensores, atendiendo a los materiales empleados en su fabricación: los que usan como base un hidrogel y los que utilizan materiales elastómeros como matriz.

Sensores con base hidrogelada

En la actualidad, este tipo de sensores son los más destacados. Los hidrogeles son materiales obtenidos por la combinación de un polímero y agua con estructuras tridimensionales similares a las esponjas, cuya característica principal es la posibilidad de absorber y retener grandes cantidades de agua.

Los sensores de este grupo se componen de un hidrogel, que se emplea como matriz, al que se le añaden otros materiales, principalmente micropartículas de carburos o algún otro conductor que permita su incorporación como pequeñas partículas. Existen diversos tipos que se encuentran en desarrollo a día de hoy, como se aprecia en la figura 2.1. De todos ellos, se abordarán los polímeros rellenos de partículas conductoras en esta sección y los basados en MXene en la sección 2.1.2.

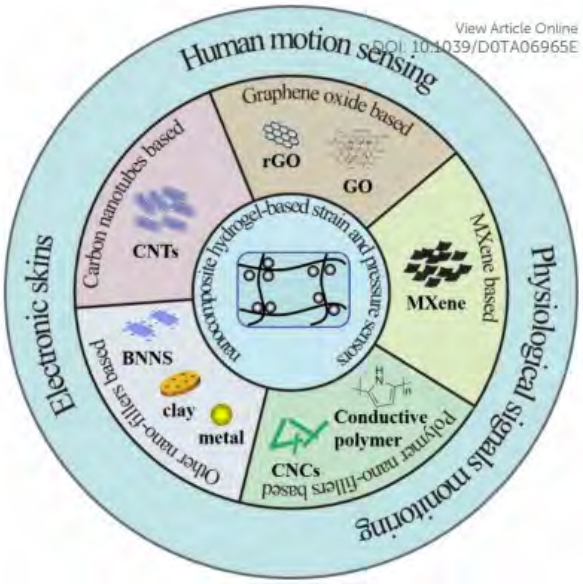


Figura 2.1 Clasificación de los sensores con base de hidrogel según su aplicación y composición. *Fuente [5]*

Los hidrogeles fabricados como polímeros conductores presentan normalmente muy buena sensibilidad, relativa buena adherencia e incluso propiedades anticongelantes o autoreparables debido a la combinación de materiales presentes en su creación. Algunos ejemplos son los trabajos realizados por [6], [7] o [8].

En el primero, se fabrica un hidrogel orgánico con una combinación de polivinilpirrolidona (PVP), ácido acrílico (AA) y acrilamida (AAm) en una disolución de agua y dimetilsulfóxido (H2O/DMSO) como solventes. Gracias a los enlaces de hidrógeno entre las cadenas de polimerización tras el curado del gel se logran muy buenas propiedades mecánicas, con una deformación de hasta un 1554 % respecto al reposo antes de la rotura, además de propiedades anticongelantes (aguantando temperaturas de hasta -40ºC) y de auto-reparabilidad. Con todo ello se logran altos grados de sensibilidad hasta un 500 % de deformación con una muy buena estabilidad térmica. Algunas de estas propiedades se pueden ver en la figura 2.2.

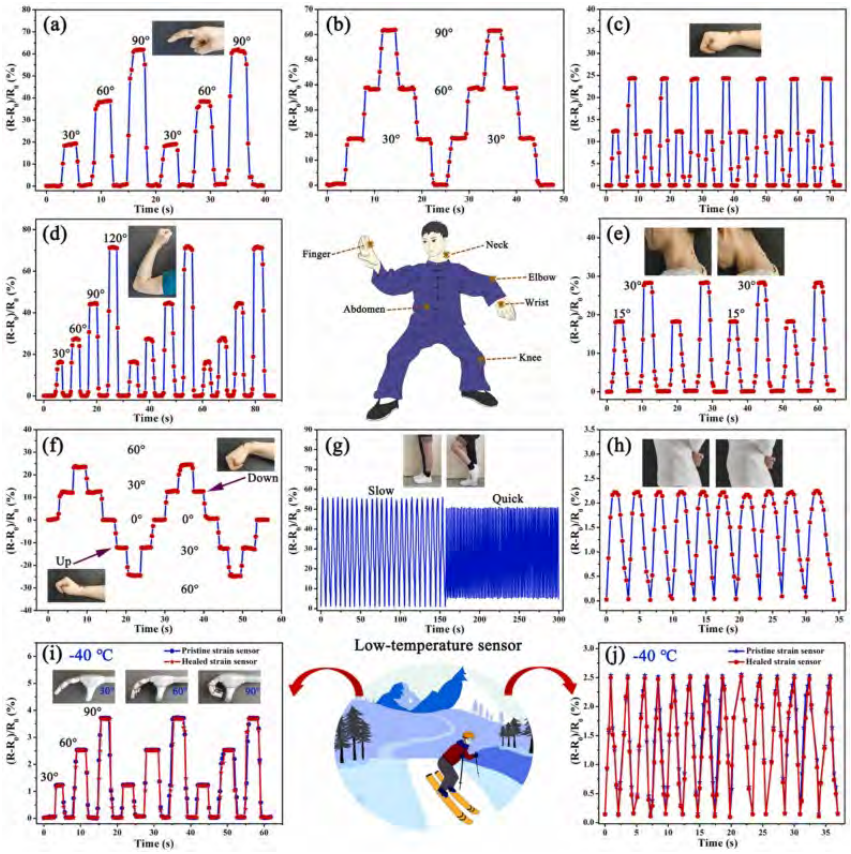


Figura 2.2 Comportamiento y características de un sensor fabricado con un hidrogel orgánico. *Fuente [6]*

En el segundo se emplea un copolímero adhesivo, conocido principalmente por su nombre abreviado *P(THAM/AM)*, como base al que se entrecruza con nanofribras de celulosa obteniendo una matriz de gran adherencia y durabilidad. Posteriormente, se introducen iones de hierro Fe^{3+}, aportando al material una buena conductividad, produciendo así un sensor para medir deformacones con elevada sensibilidad y reducido tiempo de respuesta. En este proyecto además se comprobó su eficacia para detectar movimientos humanos, incluyendo algunos más amplios y otros más discretos, como se puede observar en la figura 2.3.

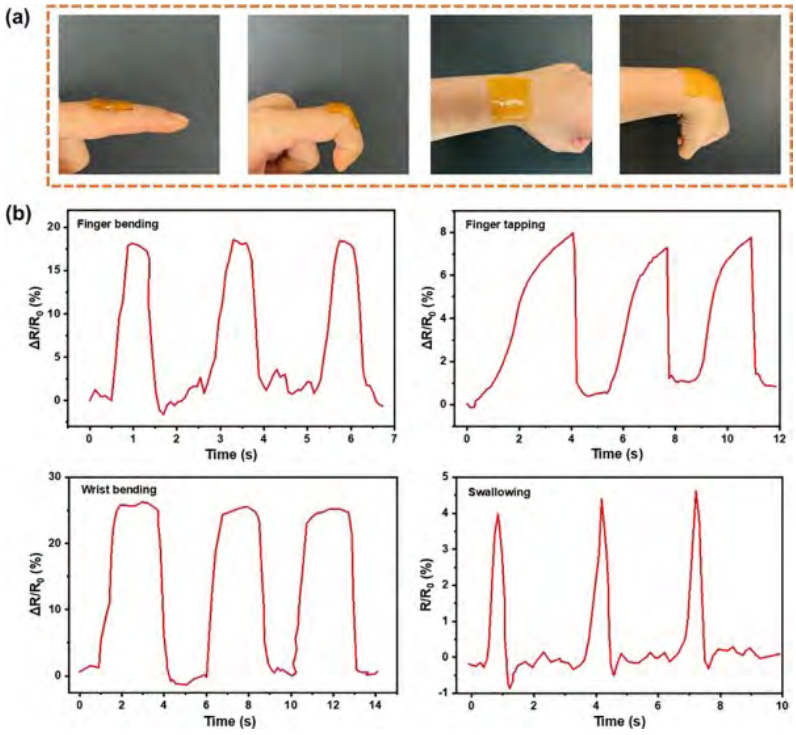


Figura 2.3 Comportamiento ante el movimiento humano de un sensor hidrogelado. *Fuente [7]*

Por último, en el tercero (figura 2.4) se crea un sensor capaz de medir tanto la temperatura como la deformación a la que está sometido. Mediante el empleo de un hidrogel con iones contenidos en su interior se consigue un sensor capaz de detectar deformaciones en un rango de hasta un 1973 % de la deformación relativa del material, además de permitir mediciones de la temperatura ambiental con resoluciones de hasta 0.2°C entre los 36.5°C y los 40°C.

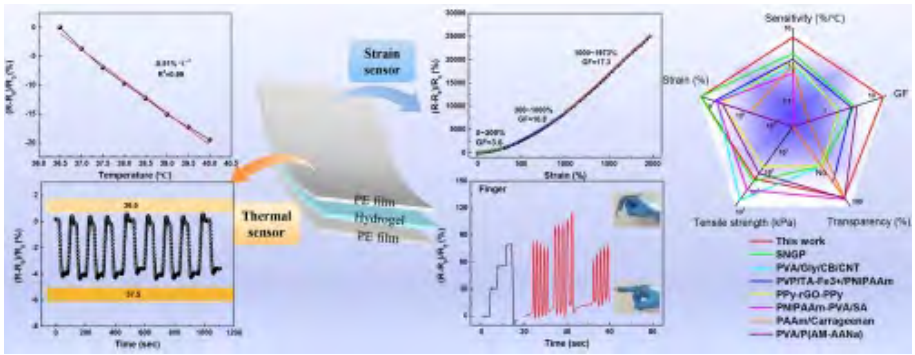


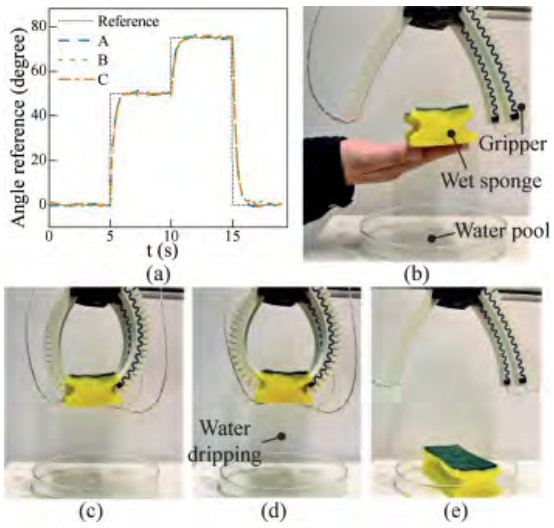
Figura 2.4 Propiedades de un sensor con base de hidrogel e iones de hierro. *Fuente [8]*

Como se ha visto, los sensores fabricados con este tipo de materiales son especialmente empleados en el modelado del movimiento humano, lo que permite extrapolar con facilidad sus aplicaciones a la robótica blanda, pues ambos tipos de comportamientos son muy similares en el método de sensorización.

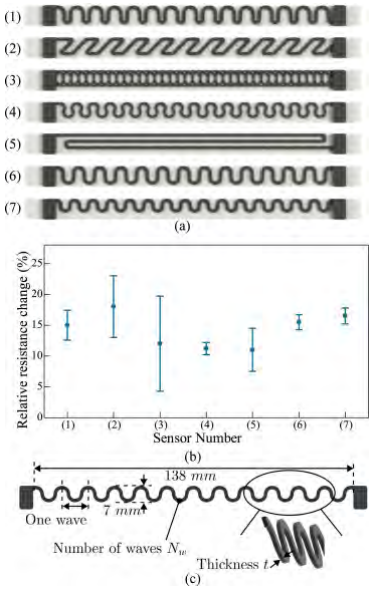
Sensores con base elastomérica

Además de los sensores ya mencionados anteriormente, existen otro tipo de sensores que se vienen usando en los últimos años. Este es el caso de sensores fabricados con materiales elastómeros. En estos casos, la principal técnica de fabricación es la impresión 3D, dada la versatilidad de materias primas y formas que permite.

EL principio de funcionamiento de este tipo de sensores elásticos es muy similar a los anteriores, pues basan sus medidas en el cambio de resistividad del material al aplicar una deformación en él. Un ejemplo del empleo de estos sensores es el trabajo de [9]. En él, los investigadores Qinglei Ji et al. de la Universidad de KTH consiguen con éxito la impresión de unos sensores elásticos con los que controlarán posteriormente una garra blanda de forma precisa (figura 2.5a) gracias a las variaciones de resistencia que presenta el sensor fabricado cuando se le aplica una deformación. También se procede a un estudio de la forma óptima del sensor dependiendo del rango de detección que se desee, como se puede apreciar en la figura 2.5b.



(a) Control de una garra neumática mediante el uso de sensores elastómeros (en negro)



(b) Rangos de medida según la forma del sensor

Figura 2.5 Estudio del comportamiento y uso de un sensor elastómero fabricado con impresión 3D. *Fuente [9]*

A pesar de su mayor facilidad de fabricación y sus buenas propiedades tanto mecánicas como de sensorización, los sensores basados en elastómeros están comenzando a ser desplazados en aplicaciones relacionadas con la bioelectrónica, pues los hidrogeles, como ya se ha visto, presentan un mayor potencial en aplicaciones que incluyan estructuras blandas y flexibles debido a sus estructuras biomiméticas y, especialmente, a sus excelentes propiedades mecánicas, eléctricas y sensoriales [10]. Aún así, en el campo de la robótica blanda continúan siendo los más empleados gracias a su mayor desarrollo y fiabilidad.

2.1.2. Sensores capacitivos

En este segundo grupo se encuentran sensores empleados principalmente en robots quirúrgicos y médicos o para el modelado del comportamiento humano. También destaca su uso en aplicaciones que requieran medir presiones o fuerzas.

En la actualidad, destacan especialmente los sensores de presión basados en el uso de MXene. El uso de estos materiales bidimensionales de reciente descubrimiento está creciendo exponencialmente debido a sus excelentes propiedades eléctricas, mecánicas y químicas. Su nombre se deriva de su estructura química. La M representa el elemento de transición, generalmente titanio (Ti), vanadio (V), niobio (Nb) o tantalio (Ta), mientras que la X representa el elemento ligador, generalmente carbono (C) o nitrógeno (N) [11]. La composición de estos materiales es apreciable en la figura 2.6. Destacar que estos materiales proceden de otros denominados MAX. Los MAX, como se aprecia en la imagen, contienen varias capas del MXene al que dan lugar entre las cuales se encuentra un elemento adicional que es eliminado durante la extracción del MXene (denominado elemento grabado o A) [12].

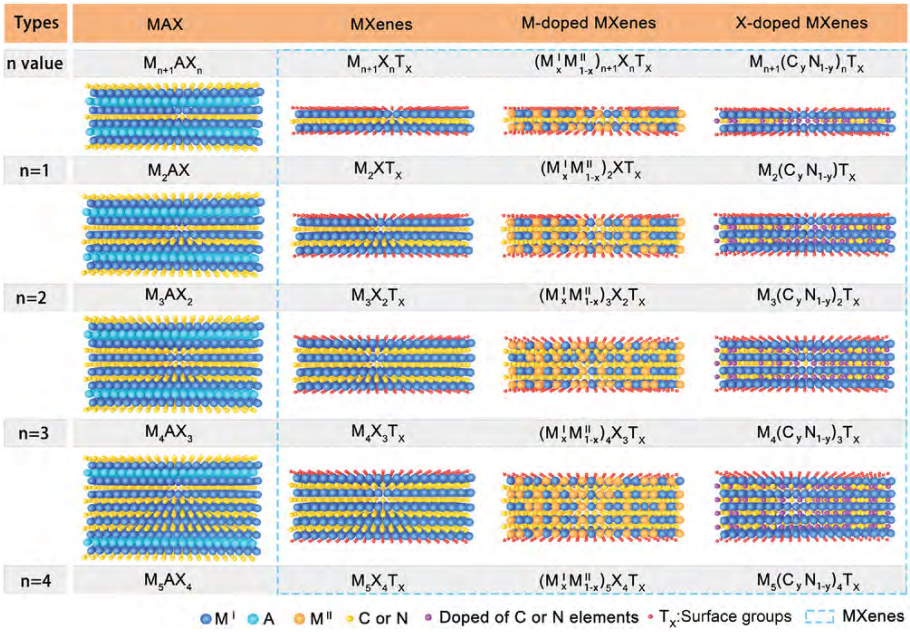


Figura 2.6 Estructura esquemática de los MXene, incluyendo los dopados con elementos de transición M y los dopados con elementos ligadores X, y los MAX de los que provienen. *Fuente [12]*

Sus aplicaciones son muy amplias: desde la industria energética hasta la industria electrónica. A pesar de ello, esta sección se centrará en sus aplicaciones como sensor de presión.

El principio de funcionamiento es similar al de un condensador, añadiendo la posibilidad de deformación del mismo. Esto se logra gracias a la adición de varias capas de un MXene una encima de otra (denominadas *nanosheets*), empleando como substrato para

adherir las distintas capas un dieléctrico como una silicona o un hidrogel. De esta forma, el material se comporta como varios condensadores en serie. Así, al aplicar una fuerza compresiva sobre el material se modifica la capacidad entre las distintas capas, lo que se puede emplear como medida indirecta de la presión a la que está sometido el material. Un ejemplo se puede encontrar en [13], donde se emplea una matriz salina elastomérica en conjunción con láminas del MXene $Ti_3C_32T_x$, logrando así una elevada precisión en las medidas de presión realizadas, incluso para los movimientos más discretos como el de las cuerdas vocales al hablar o el latido del corazón. Todos estos resultados se puede ver en la figura 2.7.

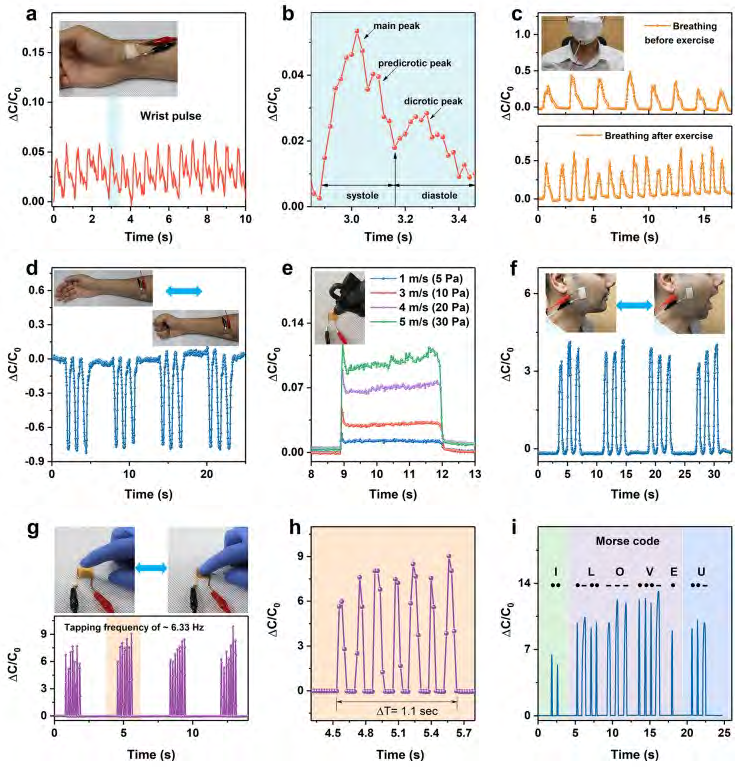


Figura 2.7 Resultados de un sensor capacitivo realizado con un MXene y una matriz elastomérica. *Fuente [13]*

También se pueden encontrar sensores piezorresistivos que emplean esta misma tecnología, empleando como substrato un material conductor y midiendo la diferencia de resistencia que presentan las capas de MXene. Un ejemplo es el trabajo realizado por Zhidong Zhang et al. [14] cuyas conclusiones se aprecian en la figura 2.8.

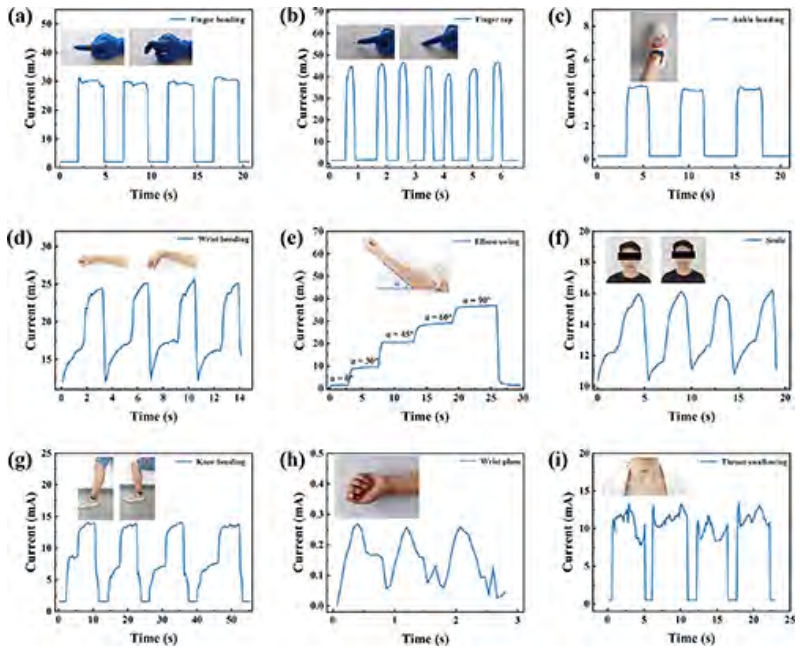


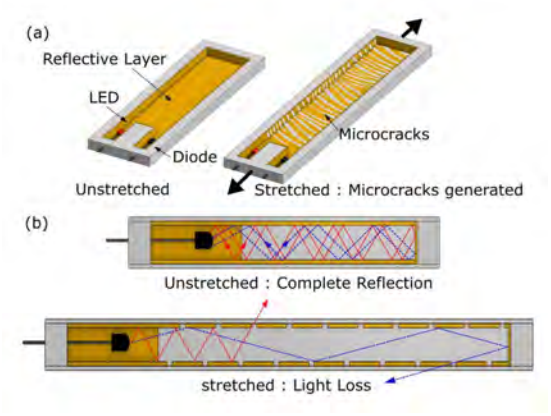
Figura 2.8 Resultados de un sensor flexible piezorresistivo realizado con un MXene. *Fuente [14]*

Asimismo se ha logrado la creación exitosa de otros sensores capacitivos, como es el caso del trabajo en [15], donde se obtiene un sensor capacitivo empleando fibras elásticas rellenas de metal líquido (la aleación eutéctica de galio e indio, conocida como *EGaIn*). Al deformar dicho sensor se obtiene un cambio en la capacidad del material gracias al movimiento del metal líquido en el interior de las fibras. Así, se logra una buena sensibilidad en la medida.

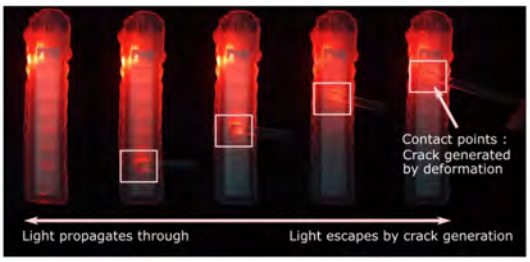
2.1.3. Sensores ópticos

Estos sensores se basan en la modificación o el movimiento de alguna propiedad visible del sensor, como la intensidad lumínica o el color, para la posterior captura de imágenes con una cámara que serán procesadas mediante un software de visión por computador para así controlar la posición y el comportamiento del *Soft Robot*.

Un ejemplo de este tipo de sensores lo encontramos en el trabajo realizado por Jaewoong Jung, Myungsun Park, DongWook Kim y Yong-Lae Park [16]. En él se emplea un método por el cual dependiendo de la deformación que sufra el *Soft Robot* se iluminará más o menos. Se basa en el uso de un emisor led y una capa reflectora. Cuando el *Soft Robot* está en reposo, la luz se relfeja hasta el final, mientras que conforme se deforma surgen en la capa reflectora pequeñas roturas que permiten que la luz escape, lo que reduce el rango que alcanza la luz dentro del *Soft Robot*. Este método se puede apreciar en la figura 2.9, junto a su resultado visual.



(a) Esquema de funcionamiento de un sensor óptico



(b) Resultado visual de la deformación producida en el *Soft Robot*

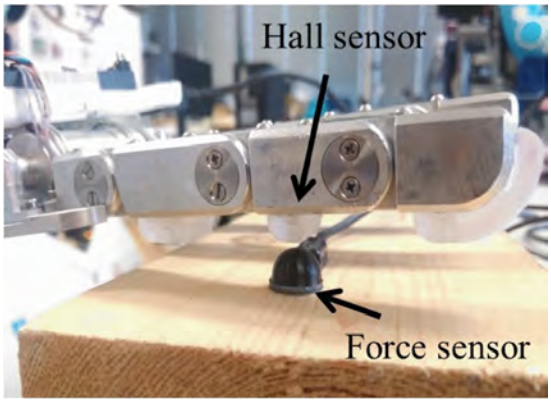
Figura 2.9 Ejemplo de funcionamiento de un sensor óptico empleado en robótica blanda. *Fuente [16]*

2.1.4. Sensores magnéticos

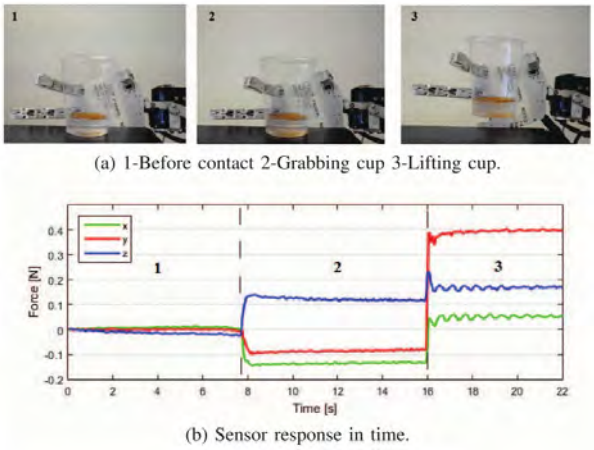
Los sensores pertenecientes a este grupo más empleados en robótica blanda se basan en el Efecto Hall. Este fenómeno físico que se da en presencia de materiales conductores y campos magnéticos (generados habitualmente por un electroimán o un imán permanente) se emplea en el ámbito de la robótica blanda para determinar la fuerza que ejerce el extremo o la presión ejercida por o sobre él.

Para el caso de la robótica blanda, estos sensores deben presentar una estructura flexible para no impedir el correcto comportamiento del *Soft Robot*. Es por ello que normalmente se emplea un imán embebido en algún tipo de material elastómero, como puede ser una silicona.

Un ejemplo del uso de este tipo de tecnología es el trabajo publicado por el IEEE proveniente de la Conferencia Internacional en Robótica y Automática del 2017 [17]. En él se presenta un sensor magnético de Efecto Hall con medición en los 3 ejes. Mediante el uso de un pequeño imán permanente embebido en una matriz de silicona con un sensor de Efecto Hall ubicado justo en el extremo del *Soft Robot*, se puede detectar la fuerza y presión ejercidas por el extremo del robot. La disposición del sensor se puede observar en la figura 2.10a, donde se aprecia el sensor de fuerza para validar los sensores de Efecto Hall, los cuales se ubican en las tres pequeñas protuberancias de silicona de los dedos del robot. Se logra así un sensor de fuerza con mediciones en los 3 ejes espaciales, detectando así tanto esfuerzos normales como cortantes con una resolución de 0.007N. Se probó en el robot *Vizzy* para el trabajo con humanos obteniendo resultados exitosos, como se observa en la figura 2.10b.



(a) Ubicación y uso de un sensor de Efecto Hall en un *Soft Robot*



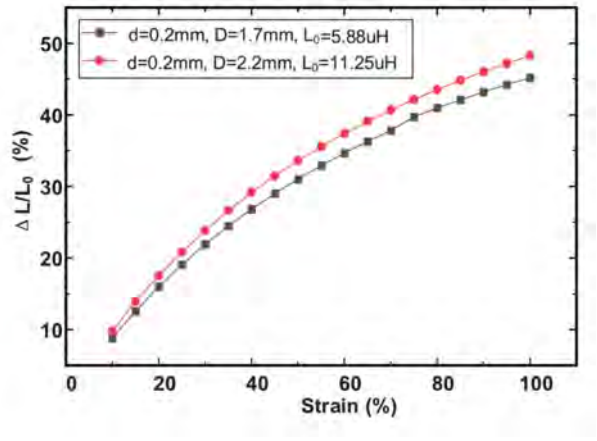
(b) Resultados del uso de un sensor de Efecto Hall en un *Soft Robot*

Figura 2.10 Ejemplo de funcionamiento de un sensor magnético empleado en robótica blanda. *Fuente [17]*

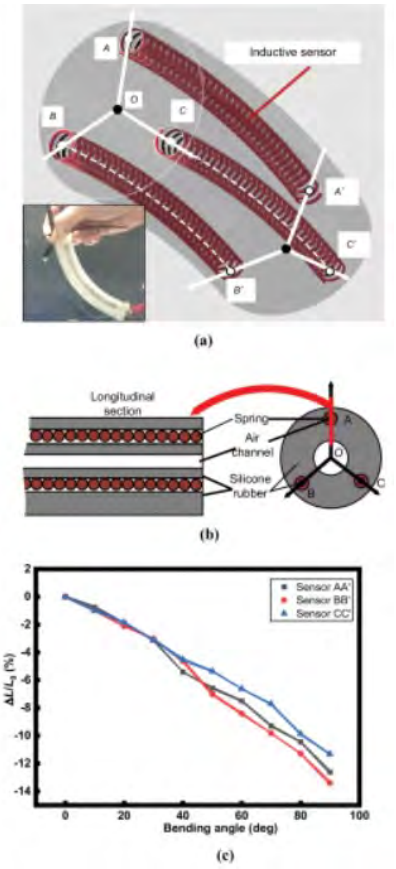
2.1.5. Sensores inductivos

Estos sensores quizá sean los menos empleados en el campo de la robótica blanda. La principal causa de este hecho es su principio de funcionamiento, el cual se basa en el cambio de la inductancia del sensor o la inductancia mutua con otro objeto debido a la variación de la permeabilidad relativa de medio. Al emplear bobinas para la fabricación de estos sensores, los dispositivos fabricados suelen tener un tamaño excesivo o una rigidez elevada para su implementación en un *Soft Robot*. A pesar de ello, este tipo de sensores suele contar con ciertas ventajas, como su mayor rango dinámico o la posibilidad de obtener salidas lineales a la medida a realizar, que hacen que la investigación acerca de ellos no se detenga de forma inmediata.

Así, existen algunos trabajos de cierta relevancia que utilizan este tipo de sensores. Un ejemplo es el trabajo en [18]. En él se basan en el principio de la variación de la inductancia propia de una bobina al sufrir una deformación (similar a un muelle). Estos da lugar a una función logarítimica, como se puede apreciar en la figura 2.11a. Esta variación de inductancia se puede aprovechar para medir deformación en el material, como en la figura esquemática 2.11b. El inconveniente surgido es la necesidad de utilizar un inductor suficientemente flexible para que no impida la deformación normal del *Soft Robot* que tenga implementado esta tecnología.



(a) Variación de la inductancia propia de una bobina en función de la deformación que sufre



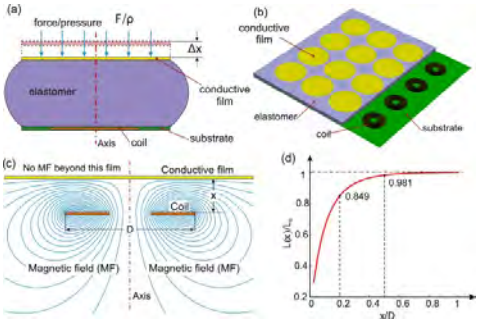
(b) Aplicaciones de los sensores inductivos en la robótica blanda

Figura 2.11 Ejemplo de uso y caracterización de un sensor inductivo aplicado a la robótica blanda. *Fuente [18]*

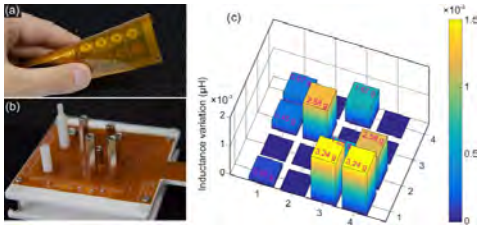
2.1.6. Sensores basados en corrientes de Foucault

Los sensores pertenecientes a este grupo no han tenido mucho recorrido en el ámbito de la robótica blanda. A pesar de ello, en los últimos años algunos grupo de investigación han ido desarrollando ciertos dispositivos que permiten mediciones de excepcional precisión mediante el uso de esta técnica. Algunos de los motivos por los que se ha comenzado a investigar acerca de estos sensores son su baja histéresis, su buena repetitibilidad, su elevada robustez ante contaminantes atmosféricos y su elevada sensibilidad. Habitualmente se emplean para mediciones de presión o fuerza [4].

En el trabajo en [19] se nos presenta una matriz de sensores (denominado Soft Inductive Tactile Sensor o *SITS* (figura 2.12a) que, basándose en el principio de la inducción electromagnética en materiales conductores, es capaz de cuantificar la fuerza ejercida por el *Soft Robot* midiendo las corrientes parásitas o de Foucault que surgen en cada uno de los sensores que la compone. De esta forma se logran resoluciones de 0.82mN en un rango que va desde los 0 a los 15N de fuerza. En el experimento mostrado en la figura 2.12b se observa la excelente detección del peso de los materiales ubicados sobre cada sensor.



(a) Modelo esquemático y principio de funcionamiento del sensor *SITS*



(b) Comprobación de la eficacia del sensor *SITS*

Figura 2.12 Creación, uso y validación de un sensor basado en corrientes de Foucault. *Fuente [19]*

2.2. Métodos de modelado y control en robótica blanda

El modelado de un *Soft Robot* es una de las tareas más complejas a las que se enfrenta este campo. El principal motivo es la estructura flexible que presentan este tipo de robots, lo que en muchas ocasiones lleva al empleo de herramientas muy distintas a las utilizadas en el modelado de los robots rígidos tradicionales. Este hecho a su vez dificulta en gran medida el control de estos robots.

La principal característica que poseen los *Soft Robots* es su cuerpo blando y maleable, al no contar con elementos rígidos no existe una cadena cinemática rígida a modelar. De esta forma, un *Soft Robot* puede verse como una estructura continua de infinitos grados de libertad, lo que impide el uso de la gran mayoría de los métodos de modelado y control utilizados en la robótica tradicional.

Otra de las diferencias puede verse en la figura 2.13. En ella se aprecia que en la robótica blanda lo que se actúa no es lo que se controla. En otras palabras, el control de los actuadores no se hace de manera directa como en un motor, sino que es necesario controlar otra variable (como el caudal de aire en un *Soft Robot* neumático), que es la que "ordena" al actuador a realizar una determinada acción según el control previo.

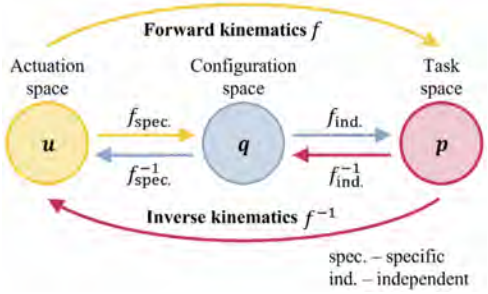


Figura 2.13 Los dos pasos necesarios para el modelado de robots blandos: modelado específico (entre el espacio de los actuadores y el de las configuraciones) y modelado independiente (entre el espacio de configuraciones y el de la tarea). *Fuente [20]*

Con el fin de abordar con éxito el modelado y control de los *Soft Robots*, han surgido diversas técnicas que permiten realizar estas tareas. Estos métodos, resumidos en la figura 2.14, se pueden clasificar en tres principales grupos [21]:

- En el primer grupo se incluyen todos aquellos métodos que tienen como base el modelado del comportamiento del *Soft Robot* mediante métodos numéricos, como pueden ser las técnicas que emplean elementos finitos. Esta metodología suele ser la más precisa para el modelado se los *Soft Robots*, si bien se suele aplicar cuando el problema presenta una complejidad de cálculo excesiva que impidan usar otros métodos. Un ejemplo sería el modelado del caudal introducido en las cámaras de un *Soft Robot* neumático.

- El segundo de ellos recoge aquellos procedimientos relacionados con el modelado del robot mediante el uso de ecuaciones cinemáticas aproximadas, similares a las empleadas en la robótica tradicional. En estos casos , se hace una aproximación de los parámetros que caracterizan la deformación, como puede ser la longitud de la generatriz de un robot cilíndrico o de cables. Estos procedimientos suelen emplearse cuando la complejidad del problema no es muy elevada, o las hipótesis necesarias para su aplicación se cumplen. Un ejemplo es el modelado suponiendo curvatura constante o *PCC*.

- El tercer grupo incluye aquella técnicas que se basan en el empleo de métodos experimentales. En este caso, y debido a la elevada complejidad que supone el modelado del comportamiento del robot, se recurre a la adquisición de datos empíricos, obtenidos mediante la realización de experimentos, que relacionan diversas características del robot entre sí, como, por ejemplo, la posición del extremo con las señales de control de los actuadores. Posteriormente, estos datos se podrán tratar de múltiples formas para obtener un modelo final del comportamiento del robot. Algunas de estas técnicas de tratamiento de los datos están relacionadas con el entrenamiento de redes neuronales.

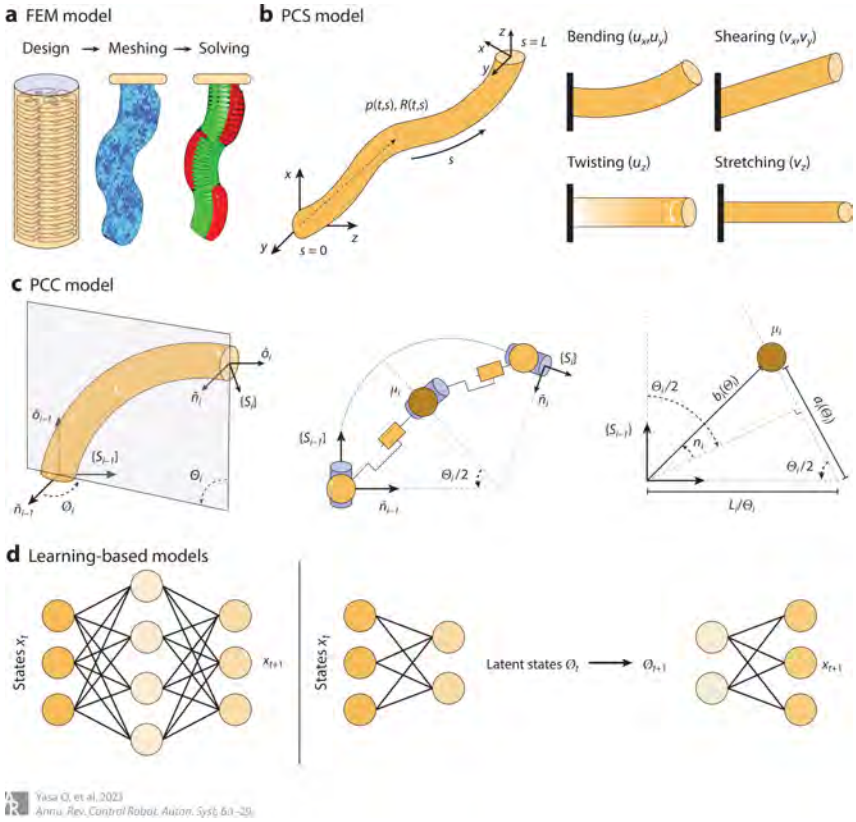


Figura 2.14 Principales métodos para el modelado de *Soft Robots*. *Fuente [21]*

Respecto al control de estos robots, existen métodos muy similares a los empleados en robótica tradicional. Estas técnicas se resumen en la figura 2.15 y principalmente consisten en la formulación de reguladores para controlar tanto la dinámica como la cinemática del *Soft Robot* conociendo los respectivos modelos del comportamiento del robot o en la extracción y el empleo de datos experimentales del movimiento del robot para posteriormente entrenar una red neuronal que permita el control del robot sin tener conocimiento explícito de su modelo.

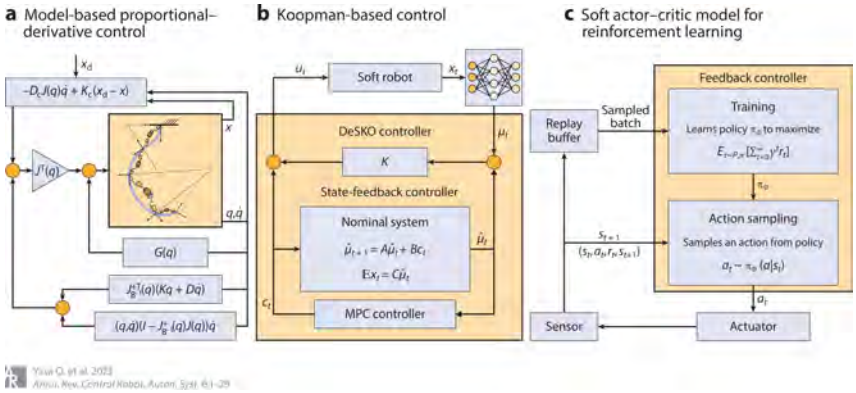


Figura 2.15 Principales métodos para el control de *Soft Robots*. *Fuente [21]*

A continuación, se explicarán en mayor detalle los métodos aquí nombrados para el modelado y el control de *Soft Robots*.

2.2.1. Métodos numéricos. Modelado mediante elementos finitos

Este tipo de técnicas consiste en la simulación por ordenador del *Soft Robot* a controlar para predecir el comportamiento que tendrá en la realidad. Se han desarrollado diversas herramientas para logran este objetivo, aunque destacan principalmente las que emplean elementos finitos o FEM.

El objetivo del empleo de estas herramientas, entre las que se encuentra SOFA o *Simulation Open Framework Architecture* de sus siglas en inglés, es modelar el comportamiento del robot ante los estímulos de actuación, obteniendo así un modelo similar al cinemático directo e inverso. De esta forma, mediante la correcta caracterización del robot es posible conocer la deformación que debe sufrir y su correspondiente forma de actuación para lograr la posición deseada.

Un ejemplo de esta metodología se encuentra en el trabajo realizado en [22], donde medinte el empleo de un modelo no lineal en SOFA y la técnica FEM aquí descrita se logra posicionar un robot neumático de tres segmentos con errores de alrededor de 2.5 cm en cadena cerrada. En la figura 2.16 se puede ver el robot empleado, su modelado en elementos finitos en reposo y su modelado con control en tiempo real.

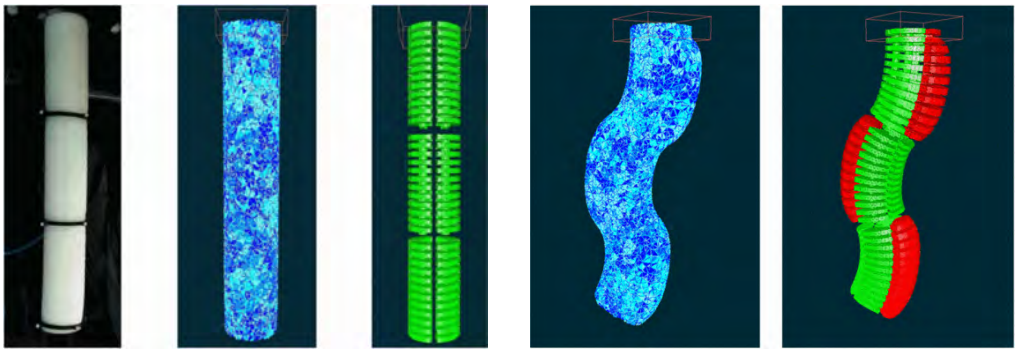


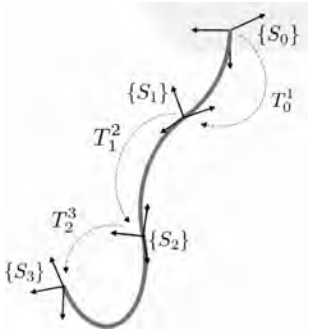
Figura 2.16 Modelado y control mediante elementos finitos. *Fuente [22]*

2.2.2. Métodos analíticos

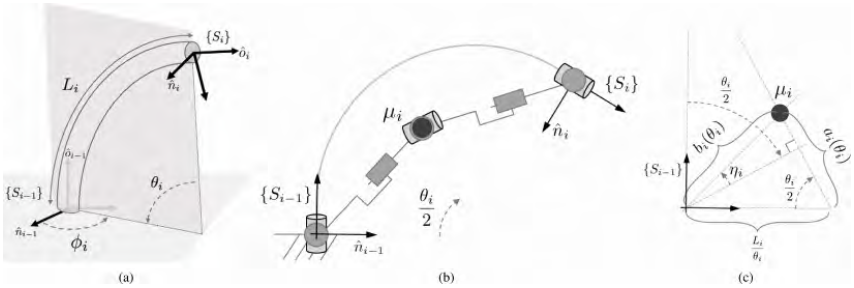
Modelado basado en curvatura constante - PCC

El principal de estos métodos analíticos consiste en considerar al *Soft Robot* como un robot continuo compuesto por varios segmentos que al actuarse presentan una curvatura constante en su deformación (figura 2.17a). Esta hipótesis permite simplificar en gran medida la forma de obtención de una matriz de transformación homogénea desde el extremo hasta la base de cada uno de los segmentos que componen el robot, pues es posible transformar dicho *Soft Robot* en un equivalente modular rígido con los suficientes grados de libertad. De esta forma, se pueden emplear las mismas herramientas que se empleaban en la robótica tradicional para el modelado de los robots, obteniendo una relación entre los parámetros característicos de la curvatura con las longitudes de los segmentos, obteniendo

tanto posición como orientación del extremo del segmento y, por lo tanto, del robot. Esta transformación de blando a rígido se puede observar en la figura 2.17b, proveniente del trabajo realizado en [23].



(a) Esquema cinemático de un robot con curvatura constante



(b) Modelo rígido equivalente

Figura 2.17 Modelado basado en curvatura constante. *Fuente [23]*

Modelado basado en la teoría de vigas de Cosserat, *Cosserat rod model* o PCS

Además de los modelos basados en curvatura constante existen también otros intentos de modelar el comportamiento de los *Soft Robots*. Un ejemplo es el uso de la teoría de vigas de Cosserat, conocida también como deformación constante por segmentos o PCS por sus siglas en inglés *piecewise constant strain*. Este modelo aproxima la estructura blanda del *Soft Robot* a varios segmentos cilíndricos largos, similares a una viga. De esta forma, al conocer las simetrías entre segmentos siguiendo esta hipótesis, se simplifica la mecánica continua de cada uno de ellos considerando exclusivamente los esfuerzos de flexión, torsión, cizalladura y alargamiento a los que están sometidos cuando se actúa el robot (figura 2.18). Esta hipótesis es especialmente útil es robots blandos con morfologías modulares, al igual que en el caso del PCC. Un ejemplo del uso de esta metodología se encuentra en el trabajo realizado en [24], donde se desarrollan las ecuaciones necesarias para modelar un *Soft Robot* actuado mediante tendones flexibles, como se observa en la figura 2.18.

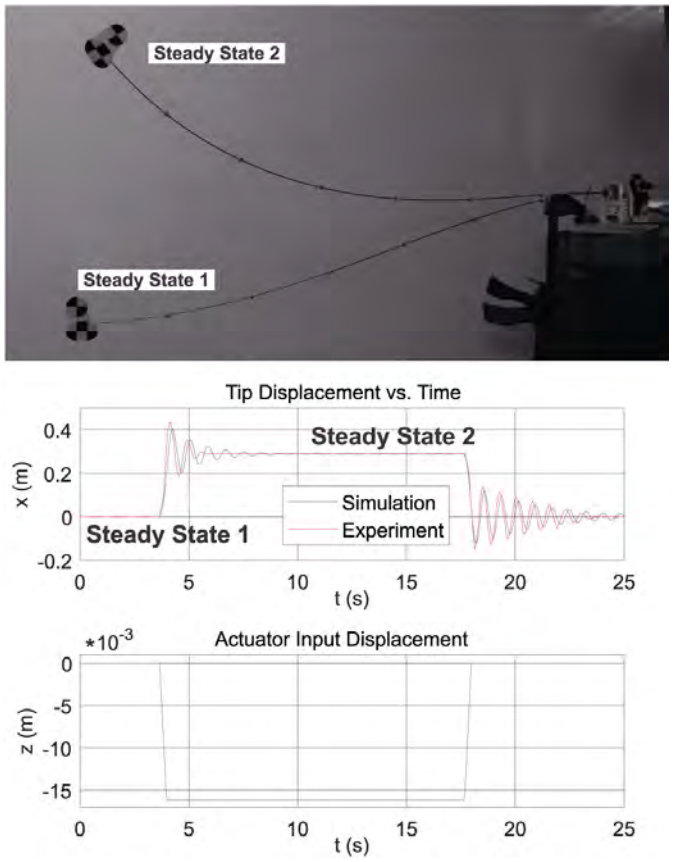


Figura 2.18 Control de un *Soft Robot* actuado con tendones mediante el uso de la teoría de vigas de Cosserat. *Fuente [24]*

Otros modelos analíticos utilizados en robótica blanda

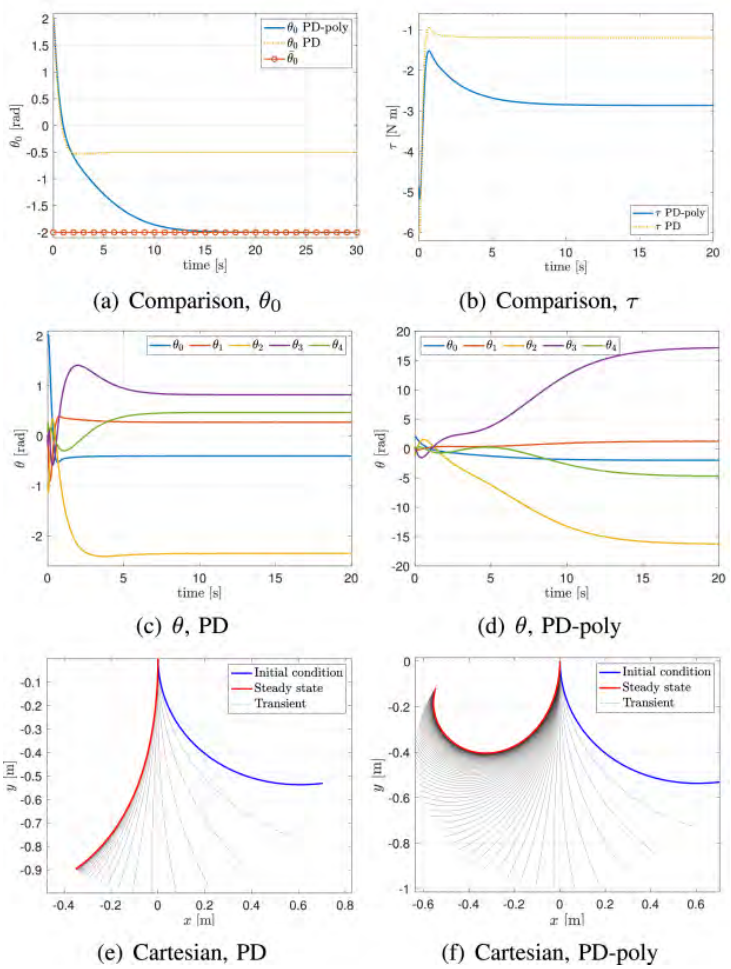
Además de los modelos mencionados anteriormente, existen otros que emplean distintas técnicas más complejas para lograr resultados de mayor precisión en robots que no se adaptan bien a los modelos más utilizados, como el PCC o el PCS. Algunos de estos diferentes acercamientos al problema del modelado de *Soft Robots* se encuentran en los trabajos realizados en [25] y [26].

En el primero, se formula un modelo a partir del uso de la curvatura polinómica (ecuación 2.2), que es una aproximación finita de la ecuación 2.1, que define la curvatura del robot en un determinado instante de tiempo para una posición dada. En este sentido, para un grado de aproximación $m = 0$, el modelo propuesto es el mismo que en el caso del PCC, pues sería curvatura constante.

$$q(s,t) = \sum_{i=0}^{\infty} \Theta_i(t) s^i \tag{2.1}$$

$$q(s,t) \simeq \sum_{i=0}^{m} \Theta_i(t) s^i \tag{2.2}$$

De esta forma, se desarrollan las ecuaciones necesarias para el modelado dinámico y cinemático empleando un regulador PD basado en el uso de esta curvatura polinómica, denominado *PD-poly*, que permite al robot alcanzar las posiciones correctamente gracias a la mejor aproximación de su forma así como a la consideración del peso y el resto de factores que afectan a su dinámica, como las fuerzas centrífugas y de coriolis o la propia inercia del robot. Los resultados se observan en la figura 2.19, donde se compara el torque y las posiciones en coordenadas modulares y en el espacio cartesiano.



(a) Comparison, θ_0 (b) Comparison, τ

(c) θ, PD (d) θ, PD-poly

(e) Cartesian, PD (f) Cartesian, PD-poly

Figura 2.19 Resultados de una simulación realizada para un robot blando manipulador utilizando un modelo basado en la curvatura polinómica de 4º grado. *Fuente [25]*

En el segundo, se emplean Curvas Hodógrafas Pitagóricas; un tipo especial de curvas paramétricas que tienen la propiedad de que tanto la magnitud como la dirección de su vector velocidad son funciones cuadráticas de su parámetro. Mediante el uso de estas curvas paramétricas, se logra unos modelos cinemáticos directo e inverso que aplicados en el robot *RobotinoXT* se logran errores de unos 2 mm de media.

2.2.3. Métodos experimentales. Métodos basados en el entrenamiento de redes neuronales

Por último, se encuentran las técnicas experimentales. Esta metodología de control de *Soft Robots* ha tenido una expansión significativa en los últimos años debido al auge de las tecnologías que hacen uso de la Inteligencia Artificial. Su principio de uso es la extracción y manipulación de datos obtenidos mediante la realización de experimentos que emplean directamente al robot físico sin previa simulación para posteriormente entrenar una red neuronal o procesarlos mediante otras técnicas de *Deep Learning*, como el aprendizaje por refuerzo (*Reinforcement Learning*).

La ventaja que presentan estos métodos con respecto a los anteriores es que para abordar con éxito el control no es necesaria la caracterización del comportamiento del robot y, por lo tanto, no se requiere un modelo del mismo. Esto supone una gran ventaja, pues la obtención de modelos en robótica blanda como se ha visto son tareas de muy elevada complejidad que en muchos casos solo se logran mediante hipótesis que aproximan los comportamientos de los robots, perdiéndose así precisión en los resultados deseados. Es por ello que en la actualidad son estas las técnicas más empleadas para el control de los *Soft Robots*.

Un ejemplo de lo aquí expuesto se encuentra en el trabajo realizado por Gabor Soter et al. [27], donde mediante el entrenamiento de una red neuronal y un autoencoder convolucional se logra que un tentáculo blando conozca su forma y posición a partir de la medida de unos sensores contenidos en su interior. Los resultados obtenidos se pueden ver en la figura 2.20.

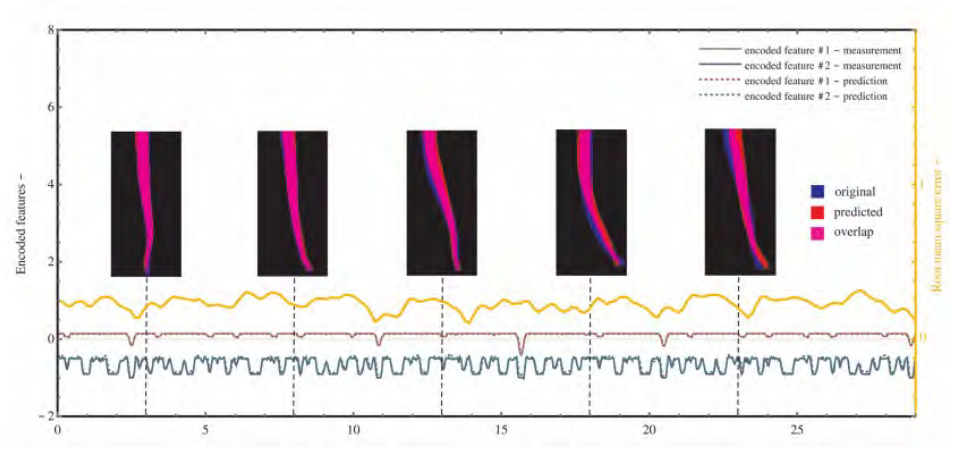


Figura 2.20 Resultados del empleo de una red neuronal para conocer la posición de un robot blando a partir de las medidas de sus sensores de deformación. *Fuente [27]*

Finalmente, cabe destacar el empleo de otras técnicas que incluyen *Machine Learning*, como es el caso de los robots entrenados mediante aprendizaje por refuerzo. En estos casos, se parte de un modelo aproximado, como puede ser el PCC, y a partir de él se ajusta el comportamiento real del robot mediante la toma de datos con las que se entrena una red neuronal de corrección. Un ejemplo se da en [28], donde mediante el uso de esta técnica y partiendo de las ecuaciones del PCC se logra el control en posición de un robot blando manipulador, logrando errores en el seguimiento de las trayectorias de en torno a 1 cm sin carga y a 2 cm con carga. El robot a controlar se aprecia en la figura 2.21.

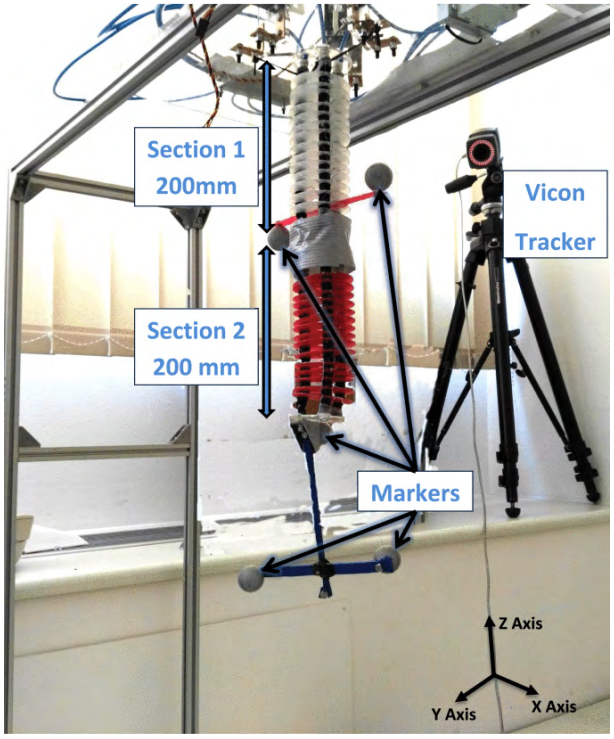


Figura 2.21 Robot blando y su setup para el control con el uso de PCC y aprendizaje por refuerzo. *Fuente [28]*

Capítulo 3

Estudio y caracterización de los sensores

En robótica, la correcta selección de los sensores es de vital importancia para poder abordar con éxito el control del robot para los que están destinados. En el caso de la robótica blanda esta problemática se acentúa debido a las restricciones presentadas en la sección 2.1, con lo que se debe hacer especial hincapié en la búsqueda de sensores que cumplan con las características necesarias para que el *Soft Robot* se mueva de forma adecuada mientras que permita mediciones con una elevada resolución para lograr resultados precisos en el movimiento posterior al control del robot.

Durante este proyecto, y debido a lo comentado con anterioridad, la parte que aquí se aborda adquirió especial relevancia, pues todo lo aquí realizado afectaba al resto de partes del trabajo. Es por este motivo que se procedió a analizar las características que debían presentar los sensores y su comportamiento final para el correcto planteamiento de los métodos de control del robot aquí mostrado. Asimismo, se ideó un sistema de medida de los sensores que lograra la máxima resolución y precisión posible.

En este capítulo se presentan las distintas fases que componen la parte de sensorización de este proyecto, desde la selección y compra de los dispositivos hasta la caracterización y prueba de los mismos.

3.1. Selección de los sensores

Para este proyecto, además de las restricciones mencionadas anteriormente, se han impuesto otras debido a la naturaleza elástica del material y a las restricciones de alcance y temporalidad del trabajo. Esto último es el principal motivo que explica la imposibilidad de fabricación de un sensor propio, como suele ser la principal técnica de sensorización en robótica blanda (sección 2.1). También cabe destacar que los sensores ópticos como

cámaras o web-cams quedaban descartados debido a que el objetivo de este proyecto es controlar un robot de forma autónoma sin depender del exterior. Es por ello que la única posibilidad era la de encontrar un sensor comercial de bajo coste con la mayor calidad de medición posible que cumpliera con todos los requisitos para abordar de manera correcta el problema del control de este *Soft Robot*.

Debido a la escasez de sensores elásticos para robótica blanda en el mercado, la elección se pudo realizar de forma sencilla y rápida. La decisión final fue un sensor resistivo elastómero de la empresa Adafruit [2] (Figura 3.1). Las razones en la que se basa esta decisión son:

- **Facilidad de medición**: Al ser un sensor resistivo, como los vistos en la sección 2.1.1, cuyo funcionamiento se basa en la variación de su resistividad ente una deformación, en este caso, de tipo longitudinal, permite buenas mediciones con una instrumentación sencilla.
- **Material de fabricación**: Este sensor está compuesto de negro carbón en una matriz de goma. Esto le aporta una excelente durabilidad y flexibilidad además de facilitar su manejo, pues no es un material tóxico ni que se vea afectado por campos electromagnéticos que puedan modificar los valores medidos debido a interferencias con el exterior.
- **Diseño del *Soft Robot***: Debido a la forma y diseño de los segmentos que componen a *PAUL*, estos sensores eran los más óptimos en cuanto a facilidad de incorporación. Esto se debe a que en el anterior trabajo [1] donde se abordó el problema del diseño, se implementaron unos canales pensados para sensores de este tipo, con lo que la incorporación de estos sensores al diseño era inmediata.
- **Coste del sensor**: La última gran ventaja que presentaban estos sensores era su relativamente bajo coste (9.95 €/m), con lo que se reducía en gran medida el presupuesto al adquirir varios metros de este sensor. A esto se suma la fiabilidad de los productos de la empresa Adafruit y su rápido envío.

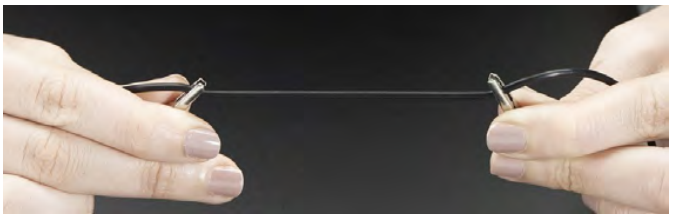


Figura 3.1 Sensor resistivo elastómero empleado en el control en cadena cerrada de *PAUL*. *Fuente [2]*

Aún así, se observó que estos sensores presentaban a su vez tres grandes inconvenientes. El primero estaba relacionado con el rango de medición, pues este era excesivamente pequeño para las deformaciones a las que estaría sometido el sensor y se saturaba la medida con rapidez. El segundo de ellos era la elevada histéresis en la resistividad del material, pues una vez deformado y posteriormente liberado de este esfuerzo normal tardaba de 1 a 2 minutos en regresar a su punto de equilibrio. La tercera desventaja se relacionaba con la baja resolución que ofrecía el sensor comercial.

Los dos primeros problemas asociados a este sensor se solucionaron debido a un cambio de comportamiento en los sensores y el tercero de ellos se solventó gracias al uso de hardware de medición. Todos estos problemas y su respectiva solución se explicarán con mayor detalle en secciones posteriores.

3.2. Método de medición

Tras la selección y la adquisición de los sensores, el siguiente objetivo era establecer el método de medición de las variaciones de resistencia que presentaban los sensores ante la deformación longitudinal. Dado que la resistencia se relaciona con tensión e intensidad mediante la Ley de Ohm ($V = R \cdot I$) se ofrecían tres posibilidades: o medir resistencia, o medir tensión o medir intensidad.

La primera de ellas implica medir tanto una tensión como una intensidad, pues no hay dispositivo de medida capaz de obtener de manera "directa" la resistencia que presenta un objeto. Es por ello que nos centraremos en las ventajas e inconvenientes de la medición de tensiones o intensidades.

En el caso de la medida de una corriente, es necesario un dispositivo que o bien detecte intensidades muy pequeñas o bien que mida corrientes más elevadas. En el primero de los casos, el dispositivo de medida tendría un coste muy elevado, pues requiere de resoluciones incluso de microamperios, aunque el consumo energético sería muy bajo. El segundo supone un menor precio al no necesitar de tecnología tan avanzada a cambio de un gran consumo de energía. Además, dependiendo de la resistencia a medir será necesario un voltaje menor o mayor.

En el caso que aquí se estudia, la segunda opción sería inviable, pues el sensor inicialmente se encuentra en un rango de varios kilohmnios, con lo que la tensión a aplicar debería ser muy elevada (de varios cientos de voltios), con lo que la potencia y el calor disipados serían excesivamente altos y podría dañar la instrumentación electrónica, el sensor o incluso el propio robot. Esto implica que si se quiere implementar un método de medición basado en medidas de corriente, se ha de adquirir un hardware con la suficiente sensibilidad que, para este proyecto, se encontraba en las decenas de microamperios (suponiendo fuentes de tensión comerciales de en torno a los 30V).

En cambio, en el caso de la medida de tensiones, la sensibilidad de la medida no supone un problema en sí mismo, pues mientras el dispositivo de medida reciba una señal clara con suficiente energía se puede realizar una medida de forma sencilla. A esto se une la facilidad de lectura de los datos, pues se encuentra en decenas de voltios y no es necesario escalar la medida con el uso de amplificadores. Es debido a estas ventajas con respecto a la medición de corriente que se decidió emplear este método.

Para poder aplicar esta técnica de medición era necesario emplear un divisor de tensión o un puente de resistencias, pues de otra manera siempre se mediría el mismo valor de tensión, al ser completamente independiente de la resistencia si está siendo aplicado por una fuente de tensión constante. Así, mediante un divisor de tensión adecuado se logra que al variar una de las resistencias (en este caso el sensor) y mientras la otra se mantenga siempre fija se mida de manera indirecta los cambios de resistencia en el sensor y, con ello, la deformación que está sufriendo.

3.2.1. Hardware de medición

Una vez seleccionada la técnica de medición de las deformaciones del *Soft Robot* el siguiente paso a completar es elegir el hardware que capture los datos medidos para posteriormente procesarlos de manera adecuada. En el anterior trabajo realizado sobre este mismo robot ya se empleaba un microcontrolador Arduino Due (figura 3.2) que permitía generar las señales de control necesarias para abrir o cerrar las válvulas en función de la acción que se desease realizar. Es por ello que el hardware de captura de datos que se empleó fue este mismo Arduino Due.

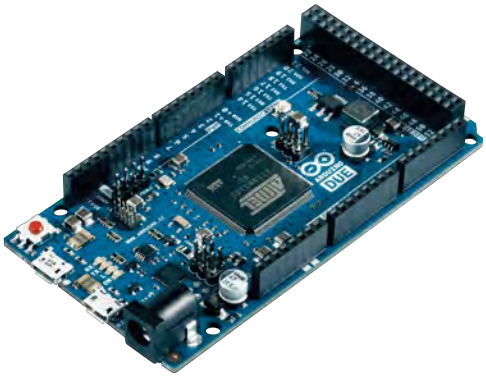


Figura 3.2 Microcontrolador Arduino Due empleado. *Fuente: elaboración propia*

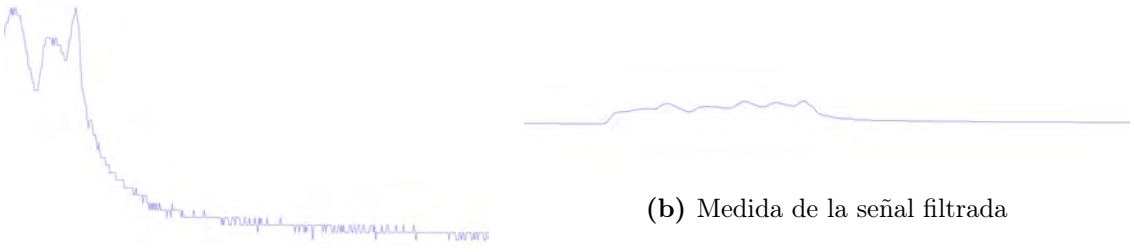
Con esta decisión se abrían otras dos posibilidades: una captura directa de los datos mediante uno de los puertos con convertidor analógico-digital (ADC) y con ello medir tensiones o utilizar un hardware específico que mediante puerto serie se comunique con el microcontrolador enviando los datos ya convertidos a señales digitales para que sean procesados con posterioridad por el Arduino Due empleado.

La medida directa es más sencilla ya que emplea un hardware específico implementado en el microcontrolador, con lo que la comunicación y captura de los datos es inmediata. Sin embargo, presenta algunos inconvenientes:

- El primero de ellos es la resolución que ofrece el Arduino Due en sus pines analógicos, pues el valor nominal de esta es de 10 bits ampliables a un máximo de 12 bits mediante comandos software. Esto supone una limitación a la hora de medir variaciones de deformación longitudinal en los sensores.

- El segundo inconveniente que presenta esta técnica es el reducido rango de medida que ofrece, pues el máximo voltaje a aplicar en los pines analógicos es de 3.3V. Este hecho adquiere gran relevancia cuando la tensión a aplicar nunca es la máxima, como sucede en este caso al emplear un divisor de tensión. Esto limita enormemente las tensiones a emplear, lo que afecta a su vez a la resolución de la medida, pues la diferencia entre dos valores de voltaje medidos para una misma variación de longitud en el sensor es mucho menor ($\frac{3{,}3V}{2^{10}} = 0{,}0032V$ frente, por ejemplo, al uso de una fuente de 12V: $\frac{12V}{2^{10}} = 0{,}0117V$).

- El tercer gran problema, que es a su vez el de mayor importancia, es el ruido que se puede producir en la medida. Al no poseer un filtro previo a la medida y estar el sistema expuesto a las señales electromagnéticas del exterior, la precisión de la medida se puede ver comprometida, como se puede ver en las medidas de la figura 3.3 con y sin filtrado de la señal.



(a) Medida de la señal sin filtrar

(b) Medida de la señal filtrada

Figura 3.3 Ruido de la señal medida y su solución por filtrado. *Fuente: elaboración propia*

Son estas tres razones las que decantaron la balanza en favor del uso de un hardware de medida externo al microcontrolador. Este debía solventar estos problemas, con lo que se fijaron tres requisitos:

1. **Tener un valor máximo de medida superior a 12V** y, preferiblemente, a 24V, pues estos son los valores de tensión que puede ofrecer una fuente para ordenador. El motivo de emplear una fuente de tensión para ordenadores es que ya se contaba con una en el laboratorio donde se realizó este proyecto.

2. **Tener una resolución mayor o igual a la ofrecida por el microcontrolador**, pues sino no supondría una ventaja con respecto al primero.

3. **Tener un comportamiento de filtro paso alto** con el fin de eliminar las frecuencias provenientes del ruido exterior. Además, no debe amplificar la señal medida, pues los valores de tensión de alimentación ya se encontrarían en el rango de medida del dispositivo. Esto implica una ganancia a la salida de 0dB.

En conjunción a estos requisitos hay que añadir las características de comunicación serie del Arduino Due. Dado que ya se venían empleando 2 puertos UART, se abría la posibilidad de una comunicación de este tipo (pues el máximo de puertos UART existente en la placa Arduino Due es de cuatro), una con protocolo I2C o una de tipo SPI.

Atendiendo a todos estos requisitos, los INAs o Amplificadores de Instrumentación suponen una gran opción, pues son circuitos especializados diseñados para medir pequeñas diferencias de voltaje con alta precisión. Asimismo, se abrían tres posibles opciones de compra que cumplieran las características mencionadas anteriormente: INA219 (con dos versiones A y B, se comparará la versión B por ser ligeramente mejor a la A), INA226 e INA3221. Todos estos dispositivos se pueden ver en las figuras 3.4a, 3.4b y 3.4d, respectivamente. En la tabla comparativa 3.1 se aprecian las diferencias entre los tres dispositivos. Todos los datos se han obtenido de sus respectivas *datasheets* ofrecidas por Texas Instruments [29], [30] y [31], respectivamente.

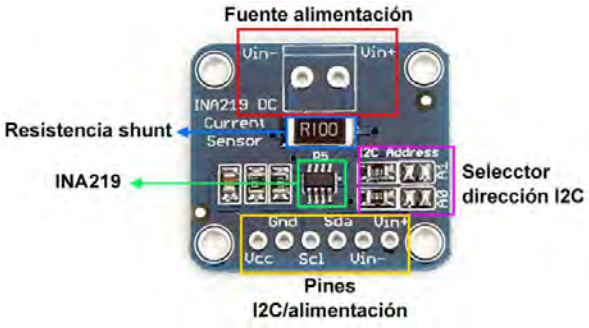
Dispositivo	Rango de medida (V)	Tensión de alimentación (V)	*Input Offset (max)*	Error en la ganacia	Corriente de reposo (max)
INA219	0 - 26	3 - 5.5	100 μV	0.3 %	1 mA
INA226	0 - 36	2.7 - 5.5	10 μV	0.1 %	0.42 mA
INA3221	0 - 26	2.7 - 5.5	80 μV	0.25 %	0.35 mA

Dispositivo	CMRR (min)	Error de medida de tensión (max)	Resolución	Número de canales	Número de direcciones I2C
INA219	100 dB	0.8 %	12 bits	1	16
INA226	126 dB	0.2 %	16 bits	1	16
INA3221	110 dB	0.5 %	13 bits	3	4

Tabla 3.1 Tabla comparativa de las distintas características del INA219, el INA226 y el INA3221. *Fuente*: Elaboración propia

Mencionar que los tres dispositivos se comunican con protocolo I2C y presentan ganancia unitaria. A su vez, los tres cuentan con un muy buen rechazo al ruido, medido indirectamente mediante el CMRR (a más alto el valor, mejor rechazo del ruido). Importante señalar que todos estos dispositivos poseen un diagrama de Bode muy similar, con una frecuencia de corte superior de en torno a los 300Hz. Dado que la medida es una constante (0Hz) son más que aptos para rechazar el ruido exterior. Además, como se puede observar en la tabla 3.1, todas ellas poseen más de 9 direcciones I2C configurables (en el caso del INA3221 al contar con 3 canales por dispositivo solo son necesarios un mínimo de 3 direcciones diferentes, pues $3 \cdot 3 = 9$), lo que permite que se puedan emplear todos estos dispositivos para los tres segmentos del robot y sus nueve grados de libertad

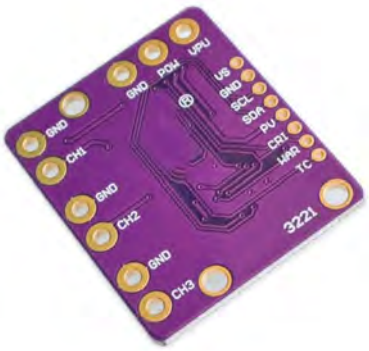
totales. Destacar además que estas direcciones se programan mediante la unión por soldadura blanda de dos conectores a modo de *jumpers*, como se observan en las figura 3.4. Posteriormente han de programarse además por software.



(a) INA219

(b) Parte frontal del INA226

(c) Parte trasera del INA226



(d) Parte frontal del INA3221

(e) Parte trasera del INA3221

Figura 3.4 Selección del hardware de medición. *Fuente [29], [30] y [31]*

Cabe destacar especialmente el hecho de que solo es necesaria una INA3221 por segmento, pues es capaz de medir tres canales gracias a un multiplexor incorporado en su interior, lo que reduce significativamente el presupuesto y la complejidad de diseño y manejo de la instrumentación final. Además, gracias a esto el hecho de contar solo con 4 direcciones programables de I2C no supone un problema a priori, pues se usarán como máximo 3 segmentos. Así, a pesar de su menor resolución, precisión y rango en comparación con el INA226, se decidió finalmente adquirir el INA3221, pues los valores en las propiedades de la medida ofrecidos cumplían sobradamente todos los requisitos mencionados con anterioridad para la aplicación en los que se iban a emplear estos dispositivos.

3.3. Caracterización del comportamiento de los sensores

Tras la adquisición y compra de los dispositivos mencionados con anterioridad, era necesario comprobar el correcto funcionamiento de dicho hardware electrónico junto al sensor para posteriormente poder caracterizar su comportamiento con éxito. Es durante este proceso donde ocurre un fenómeno inesperado y fortuito que llevará a uno de los mayores logros durante este proyecto. Es gracias a este suceso que se solucionaron todos los problemas derivados de la falta de rango y la elevada histéresis de los sensores, lo que permitió un funcionamiento mucho mejor de lo esperado inicialmente. A continuación se introduce y explica el fenómeno del cambio en el comportamiento de los sensores.

3.3.1. Cambio de comportamiento de los sensores

Este evento surge como consecuencia de la comprobación del correcto funcionamiento del INA3221 junto al sensor elastómero. Como parte de este proceso de verificación se procedió a montar los sensores (sin ningún tipo de adhesivo ni fijador) en uno de los segmentos del robot con el objetivo de comprobar el rango en el que se encontraban las medidas de dicho sensor y si era suficientemente amplio para el correcto funcionamiento del *Soft Robot* en el campo de trabajo objetivo. De esta forma, tras la debida comprobación de la que se obtuvieron unos resultados negativos, pues el sensor se saturaba mucho antes de lo esperado al igual que su histéresis suponía un grave problema; se dejaron dichos sensores montados en contacto con la silicona curada de la que está fabricado cada segmento (TinSil 8015). Cabe destacar además que para este experimento se empleó un divisor de tensión con una resistencia fija de 1kΩ, pues el rango de variación de la resistencia del sensor estaba entre los 2 y los 6kΩ, aproximadamente.

Tras esto y al cabo de una semana se procedió a examinar de nuevo el comportamiento de los sensores para idear una forma de solucionar los problemas observados en la semana anterior. Al proceder con esta nueva prueba se apreciaron pequeñas incrustaciones de silicona en los puntos de mayor contacto con el sensor. Inicialmente, no se dio excesiva importancia a este hecho, hasta que al volver a realizar la medición correspondiente el valor mostrado no se correspondía con lo obtenidos anteriormente. Es más, no solo eran diferentes, sino que además la medición era nula (señala que se incialmente se medía la caída de tensión en la resistencia fija). Dado que la tensión medida era de 0V, se concluyó que el sensor se había roto y se había generado un circuito abierto.

Como la conclusión no era lógica, pues nada podía haber producido la rotura del sensor, se procedió a examinar si el material presentaba alguna grieta no detectada anteriormente y, para facilitar esto, se estiró el sensor para poder comprobar si se había producido una sección en algún lugar. Fue aquí donde se pudo apreciar la verdad de lo que había ocurrido,

pues al estirar sin haberlo desconectado del dispositivo de medida se observó como los valores medidos ya no eran nulos, sino que eran positivos e iban aumentando conforme se estiraba el sensor.

Con posteriores pruebas y experimentos se pudo extraer conclusiones más certeras del nuevo comportamiento del sensor. Las principales diferencias son cuatro: rango de la medida, rango de la deformación hasta la saturación, histéresis del material y comportamiento ante la deformación.

Con respecto al **rango de la medida**, el nuevo sensor presenta un comportamiento muy diferente. Mientras el sensor comercial presenta un rango de medida de entre 2 a 6kΩ el sensor modificado se mueve entre los 100kΩ hasta las decenas de MΩ. Este rango varía entre los distintos sensores, dependiendo de su longitud y tiempo de contacto con la silicona del segmento, aunque siempre se encuentra entre los mismos órdenes de magnitud. Cabe destacar además que el incremento de resistencia tiende a un valor constante, por lo que para tiempos de contactos superiores a los 5 días aproximadamente no se observan variaciones significativas en estos valores de resistencia.

En lo referente al **rango de la deformación**, el material ahora presenta una muy superior capacidad de estiramiento previa a la saturación de la medida de resistencia del sensor. En algunos casos llega a ser de hasta 4 veces superior, como se aprecia en la figura 3.6. Esto soluciona el problema que existía relacionado con este hecho, pues gracias a esto el sensor puede detectar sin saturarse las deformaciones que pueda sufrir cada segmento en el campo de trabajo objetivo.

En cuanto a la **histéresis del material**, esta se vió eliminada por completo. Tras la modificación de sus propiedades, el sensor no presenta histéresis alguna, pues regresa a su estado de equilibrio casi de forma inmediata tras la liberación del esfuerzo normal aplicado. De esta forma, se soluciona otro de los inconvenientes que mostraba este sensor, pues no es necesario esperar para que se estabilice la medida y se puede actuar el robot de forma inmediata y continuada. En la figura 3.6 se aprecia este cambio.

Por último, el comportamiento ante la deformación también es significativamente distinto. En el sensor comercial se observaba un aumento exponencial de resistencia al sufrir un estiramiento (lo cual es lógico, ya que aumenta la distancia entre las partículas conductoras de negro carbón y además se disminuye la sección, con lo que la resistencia aumentaría según la ecuación $R = \rho \cdot \frac{L}{S}$), mientras que en el sensor modificado la resistencia parte de las decenas de megaohmnios y cae de forma mayoritariamente lineal hasta alcanzar la saturación. Este hecho no guarda lógica aparente, pero puede facilitar el tratamiento de las medidas dependiendo del punto de trabajo del sensor (en la zona lineal se hace más sencillo operar). Sí es han dado casos de este tipo de fenómenos anteriormente, como el trabajos realizados en [32] y en [33], como se ve en la figura 3.5, aunque no se trataban de sensores blando como en este caso, sino que se emplearon fibras de carbono.

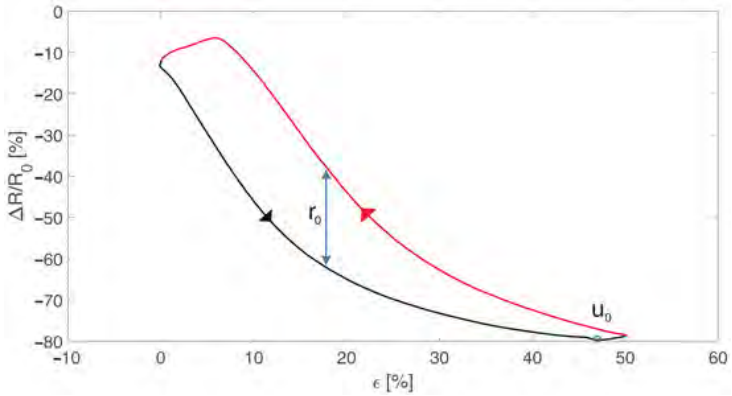
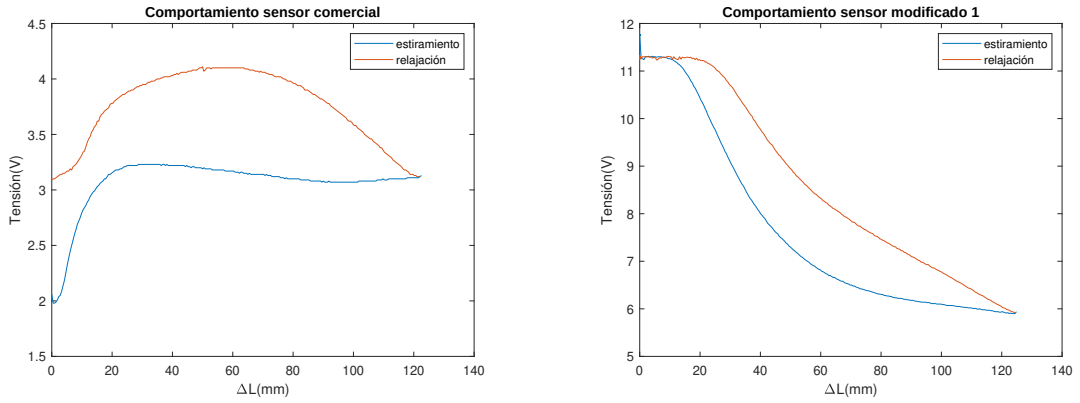


Figura 3.5 Comportamiento de un sensor similar al empleado en este proyecto. *Fuente [32]*

Todas estas diferencias son observables en la figura 3.6, donde se compara a la izquierda el comportamiento del sensor comercial ante la elongación y a la derecha el comportamiento del sensor modificado.



(a) Comportamiento del sensor comercial al variar su longitud

(b) Comportamiento de un sensor modificado al variar su longitud

Figura 3.6 Comparación del comportamiento de los sensores comercial y modificado. *Fuente: elaboración propia*

Como se puede ver, la diferencia entre ambos comportamientos es muy significativa, especialmente en el retorno al estado de reposo.

Cabe señalar además que a día de hoy no ha sido posible encontrar una explicación para este fenómeno, pues no es lógico su funcionamiento. Una hipótesis inicial podría ser que se haya producido una reacción química entre los materiales que conforman el sensor y la silicona curada con la que se moldean los segmentos de PAUL, ya que cualquier otra explicación no parece físicamente probable. Para verificar esta hipótesis sería necesario observar mediante microscopía electrónica la microestructura del nuevo material y compararla con la del sensor comercial.

3.3.2. Diseño y construcción del banco de pruebas

Como se ha visto en el punto anterior, los sensores ya no se comportaban de la forma original, sino que lo hacían de manera muy diferente. Es por ello que no se procedió a realizar el conexionado final hasta que no se hubo caracterizado correctamente.

Para caracterizar el sensor de forma correcta y precisa era necesario diseñar y fabricar un banco de pruebas que estirara el sensor de manera constante y midiera los valores de tensión para cada variación de longitud. De esta forma, el banco de actuación a emplear debía cumplir con los siguientes requisitos:

- Contar con longitud suficiente como para alcanzar deformaciones longitudes que se acerquen a la saturación del sensor. El desplazamiento necesario para alcanzar dicha saturación se determinó de forma experimental con un estirado manual previo a la construcción del banco de pruebas y debía estar en torno a los 15 o 20 cm.
- Contar con un medio que estire y devuelva a la posición inicial al sensor de forma controlada, continua y uniforme.
- Contar con un punto de anclaje del sensor.
- Contar con dos puntos para cerrar el circuito eléctrico a través del sensor.

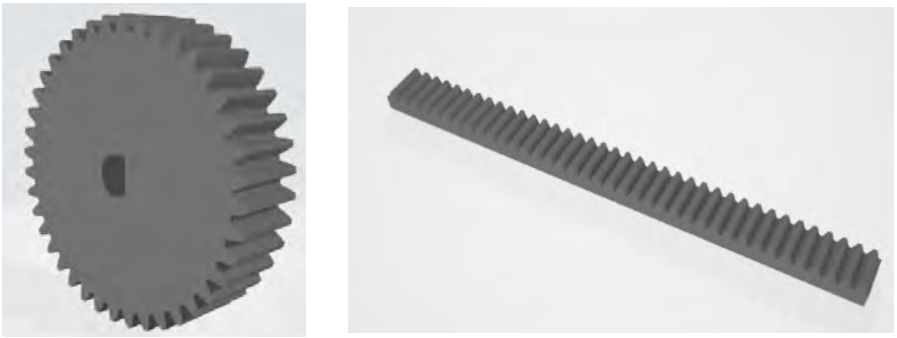
Para cumplir el segundo objetivo, que es el más crítico, se pensó inicialmente en emplear un actuador lineal para esta tarea. Estos actuadores cuentan con un motor que mueve un vástago con una carrera variable. Así se logra de manera muy sencilla y eficaz un movimiento longitudinal.

Sin embargo, estos actuadores presentan ciertos inconvenientes, como su coste y, especialmente, la longitud que presentan cuando el émbolo se ha desplazado hasta su máximo. Estas razones explican el posterior rechazo de esta idea, pues la longitud que presentaría un actuador lineal de 20 cm de vástago sería de aproximadamente 60 cm, lo cual es inviable si a eso se le suma la longitud del propio sensor.

Es por ello que se procedió a idear una nueva forma de lograr este objetivo. A continuación se describen los pasos tomados para realizar este diseño:

1. **Sistema de actuación**
   Para este fin se dispuso de un motor paso a paso o *stepper*, ya adquirido previamente, en cuyo extremo se había montado un engranaje. De esta forma, gracias al giro del motor paso a paso se puede desplazar una cremallera empujada por el engranaje del eje del motor, logrando así un par piñón-cremallera. Dado que la altura del *stepper* es de 40 mm (a lo que hay que sumar la base de apoyo de 4 mm de espesor), se diseñó un engranaje de 30 mm de diámetro primitivo que junto a una cremallera de altura de línea primitiva respecto a la base de 5 mm y un pequeño bloque de

madera de 4 mm de altura que facilita tanto el guiado como el deslizamiento de la cremallera, cubren perfectamente la mitad de la altura del motor según: $\frac{H}{2} + h_{base} = \frac{\Phi}{2} + h_{crem} + h_{apoyo\_cremallera}$. En la figura 3.7 se aprecian tanto el piñón como la cremallera empleados.



(a) Piñón **(b)** Cremallera

Figura 3.7 Par piñón-cremallera empleado en el banco de pruebas para caracterizar el comportamiento de los sensores. *Fuente: elaboración propia*

2. **Base del banco de actuación**
   Con el diseño de la parte actuadora realizado, el siguiente paso a realizar era encontrar un lugar donde posicionar dicho motor con piñón-cremallera. Para ello, se dispuso de un tablón de madera de 460x200x16mm, de esta forma se tenía la suficiente longitud para alcanzar la saturación del sensor a la vez que se aportaba robustez al conjunto.

3. **Sistema de anclaje de todos los elementos**
   Para la correcta fijación de todos los elementos a la base se diseñó un sistema de sujeción del motor stepper (figura 3.8). Para su posterior fijación en la base, se taladraron 4 agujeros de 6.5 mm de diámetro, lo que garantizaba un correcto posicionamiento, junto a pernos de M6. Se usaron además tornillos de M2.5 para anclar el soporte al motor.

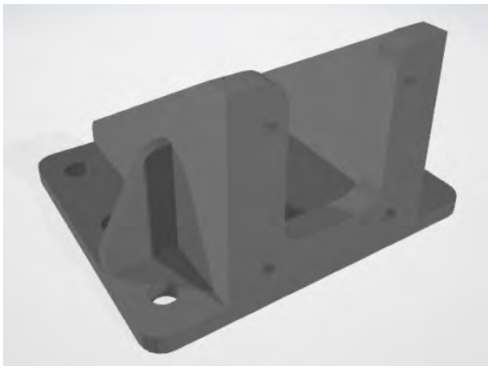


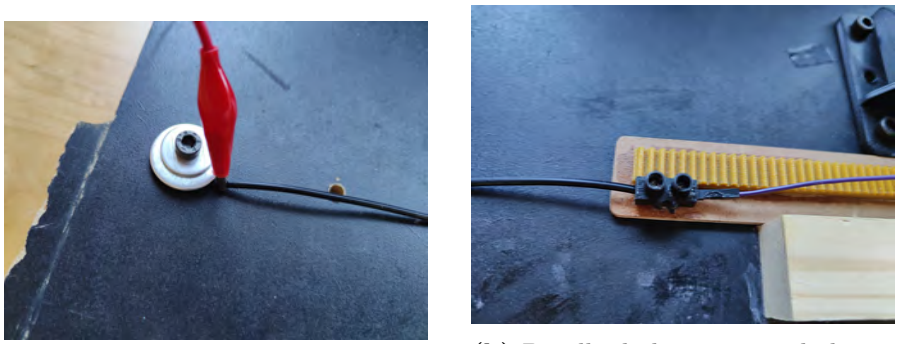
Figura 3.8 Base de apoyo para anclar y fijar el motor paso a paso en el banco de pruebas. *Fuente: elaboración propia*

Como punto de anclaje del extremo fijo del sensor se taladró de nuevo un agujero de 6.5 mm a una distancia de 295 mm, que resulta de la suma de la longitud del sensor a la medida necesaria para su uso en el robot y la longitud de la cremallera diseñada. Así, mediante el uso de un perno de M6, se fija la posición del extremo del sensor gracias al uso de dos arandelas que, con la presión ejercida por la tuerca al ser roscada sobre el tornillo, mantienen al sensor en correcta posición. Este sistema de fijación se puede ver en la figura 3.9.

Figura 3.9 Sistema de anclaje del extremo fijo del sensor a probar. *Fuente: elaboración propia*

4. **Mejoras en el diseño y conexionado del sensor**

   Para finalizar la parte constructiva, era necesario realizar el conexionado del sensor con el hardware de medida. Para ello se emplearon dos cables con pinzas de cocodrilo en sus extremos y un empalme de cables. Este último se pegó a la cremallera en un lateral. De esta forma permitía el anclaje del extremo móvil del sensor a la cremallera. Con todo esto, el conexionado consistía en colocar una de las pinzas de cocodrilo directamente sobre el extremo fijo del sensor mientras que el otro extremo se anclaba a la cremallera con el uso de un empalme de cables mencionado anteriormente. Así, en la otra entrada del conector se colocaba un cable al que iba sujeto por su extremo posterior la pinza de cocodrilo restante. Lo descrito aquí se puede ver en la figura 3.10.



(a) Detalle de la conexión de la conexión del extremo fijo del sensor

(b) Detalle de la conexión de la conexión del extremo móvil del sensor

Figura 3.10 Detalles de las conexiones de ambos extremos del sensor a caracterizar. *Fuente: elaboración propia*

A su vez, se incorporó una guía (figura 3.11) que impedía el movimiento lateral de la cremallera, evitando posibles desconexiones con el engranaje del piñón.

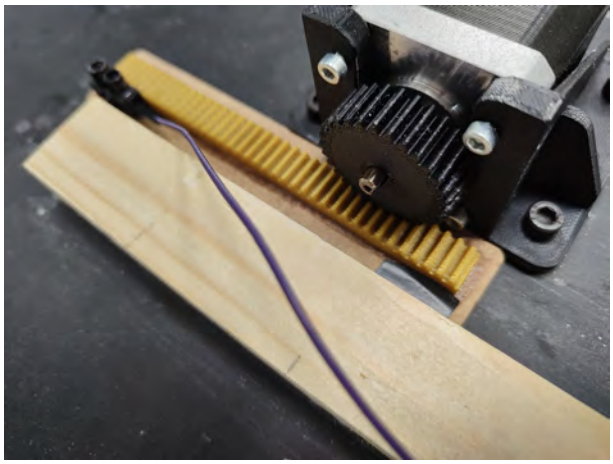


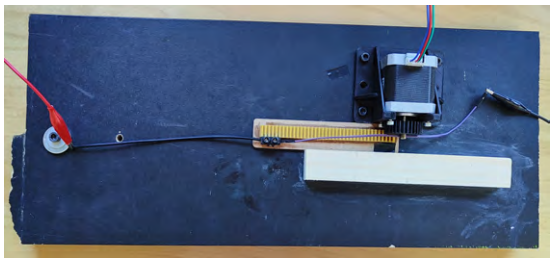
Figura 3.11 Guía para evitar el movimiento lateral de la cremallera. *Fuente: elaboración propia*

5. **Escritura del código de control**

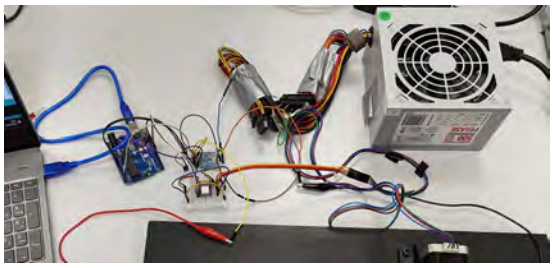
   Finalmente, tras el montaje del banco de pruebas se procedió a la escritura del software de control en la IDE de Arduino.

   El funcionamiento del banco de actuación consiste en mover paso a paso el **stepper** a la vez que se leen las medidas realizadas por el INA3221. Tras ello, se enviaban los datos extraídos a través de la comunicación serie (UART) con una App de MATLAB, realizada a través de la herramienta *App Designer* presente en este software. De esta forma se podrían procesar y almacenar de forma más sencilla y segura. Cada ciclo de prueba consistía en estirar y devolver a la posición inicial el sensor únicamente con la acción del motor paso a paso (sin liberar el sensor en ningún momento).

El banco de pruebas ya montado se puede observar en la figura 3.12, junto a sus conexiones para la prueba.



(a) Banco de pruebas montado y listo para su funcionamiento. *Fuente: elaboración propia*

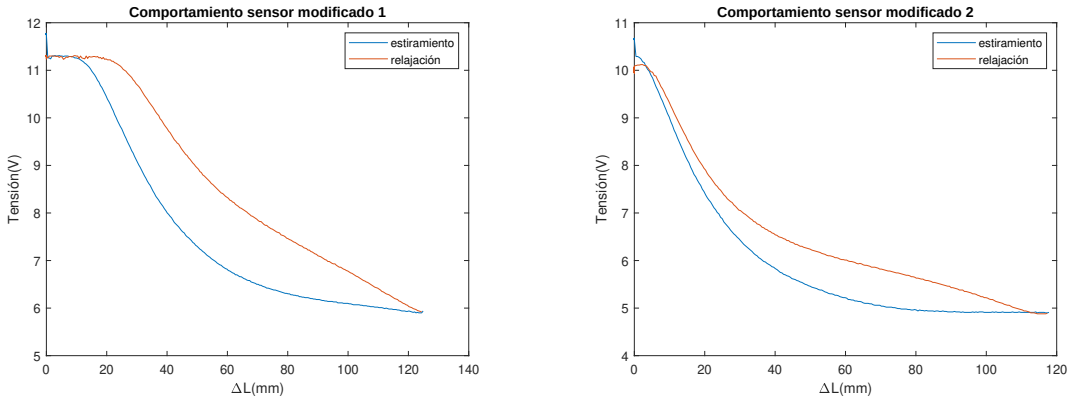


(b) Conexiones y elementos necesarios para el uso del banco de pruebas. *Fuente: elaboración propia*

Figura 3.12 Banco de pruebas para la caracterización de los sensores. *Fuente: elaboración propia*

3.3.3. Conclusiones extraídas del comportamiento de los sensores

Una vez diseñado, construido y programado el banco de pruebas, se procedió a realizar la caracterización de los sensores. Los resultados obtenidos de los experimentos realizados a temperatura ambiente (25ºC) con los distintos sensores se pueden observar en la figura 3.13. Se emplearon dos sensores diferentes, todos ellos empleando el mismo procedimiento. De esta forma se puede comparar si presentan un comportamiento similar o no. Las medidas de tensión se hicieron con una tensión de bus de 12V, pues es más seguro emplear 12V que 24V y no necesitamos aquí resolución.



(a) Comportamiento del sensor 1 **(b)** Comportamiento del sensor 2

Figura 3.13 Comportamiento ante la elongación de dos sensores modificados distintos. *Fuente: elaboración propia*

De aquí se extraen las siguiente conclusiones:

1. Los sensores modificados presentan un rango muy amplio de medición hasta la saturación.

2. Los sensores modificados presentan cuatro zonas claramente diferenciadas:

   a) **Zona umbral**: Al comienzo, en un rango variable de 0 a 10 mm no se aprecian cambios en la medida de los sensores.

   b) **Zona lineal**: Tras la zona umbral, el sensor presenta una zona lineal variable de entre 20 a 30 mm.

   c) **Zona exponencial**: Posterior a la zona lineal y previo a la saturación, el sensor presenta un perfil curvo similar al presentado por las funciones exponenciales.

   d) **Zona de saturación**: Finalmente, llegado cierto valor variable, el sensor satura su medida y no se aprecian variaciones en valores medidos.

3. Los sensores modificados presentan histéresis al devolverlos lentamente a su posición inicial. A pesar de ello, se cree que las vibraciones del motor al devolver al sensor a

su estado de reposo pueden haber inducido este efecto en los sensores. Además, la relajación de los sensores se podrá solucionar fácilmente procediendo a una relajación previa o aumentando ligeramente el tiempo entre posiciones. Es por ello que no se considera un problema significativo.

4. Los sensores modificados presenta un comportamiento similar. Sin embargo, los valores medidos y su histéresis varían de un sensor a otro. Esto implica que no será posible emplear el mismo modelo para cada uno de ellos.

Adicionalmente, se repitieron las mismas pruebas para los sensores a diferentes temperaturas. Los resultados de estos experimentos se observan en la figura 3.14. A la vista de los resultados, se puede afirmar que la temperatura no afecta de manera significativa a los resultados obtenidos para la temperatura ambiente de 25ºC, especialmente teniendo en cuenta que la temperatura a la que se trabajaría posteriormente se mantendría constante y estable. Se observa además que estas variaciones son muy pequeñas cuando se estira el sensor, variando más en la vuelta a la posición inicial, especialmente si la temperatura es baja.

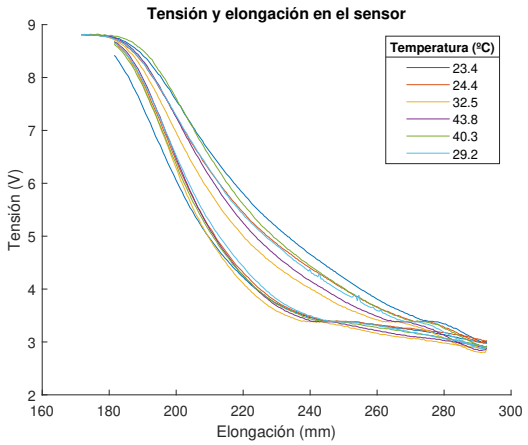


Figura 3.14 Comportamiento de uno de los sensores modificados ante las distintas temperaturas exteriores. *Fuente: elaboración propia*

Tras la caracterización de los sensores a emplear, se procedió a la modificación de la forma de medida. Dado que la resistencia *shunt* con la que cuenta el INA3221 es de 0.1Ω, mucho menor que la resistencia mínima que presenta el sensor, se procede a medir ahora la tensión de la carga, incluyendo el shunt, ya que al ser mucho menor su resistencia también lo es su caída de tensión, por lo que no modifica de forma apreciable la tensión en el sensor. Esta medida se da respecto a una tensión de bus que será de 24V (ya que el máximo es 26V), pues de esta forma se aumenta la diferencia entre dos medidas consecutivas realizadas por el INA3221. Esto se justifica gracias a que la resolución del hardware de medición es siempre la misma (13 bits), lo que implica que los saltos entre cada variación de 1 bit en la conversión sea mayor cuanto mayor sea la tensión de referencia.

Destacar a su vez que la tensión de la carga será el voltaje que surja de la resolución de la ecuación 3.1, que estará siempre contenida entre 0 y la tensión en el bus (24V). Así, con la selección de una resistencia fija correcta se podrá medir tensiones en un rango variable en función de la deformación sufrida por el sensor.

$$V_{LOAD} = V_{BUS} \cdot \frac{R_{SENS}}{R_{SENS} + R_{FIJA}} \tag{3.1}$$

En el caso que aquí se describe, dado que al comienzo la deformación es exponencial con una pendiente muy pronunciada (figura 3.13a, por ejemplo), interesa que la tensión en reposo sea la del bus, mientras que para una elongación máxima donde se produzca la saturación se reduzca hasta alcanzar un valor próximo a 0, cubriendo así el máximo rango posible. Este valor de resistencia fija se puede obtener de forma analítica, aproximando la función resultante de la figura 3.13a mediante un polinomio de x grado y sustituyendo en ella la ecuación del divisor de tensión 3.1 para cada uno de los sensores empleados, pues como se puede ver en la figura 3.13 muestran diferentes comportamientos, con lo que al derivar la función se podría extraer el valor de resistencia que maximiza el rango de medida; o se puede obtener de forma experimental. Dado que cada sensor posee rangos diferentes y debido al gran gasto computacional que supondría realizar los cálculos analíticos, se ha optado por la solución experimental ajustando mediante prueba y error el valor deseado de resistencia.

3.4. Instrumentación final e implementación de los sensores en el robot

El último paso previo al comienzo de la parte de control es fijar los sensores a los segmentos que conforman el robot y montar y cablear toda la instrumentación necesaria para abordar las siguientes fases.

3.4.1. Justificación de la elección del método de fijación de los sensores

Con respecto a la fijación de los sensores, surgen varias condiciones que debe cumplir este sistema:

- Debe ser flexible y elástico, pues de lo contrario impediría el correcto movimiento del robot y los sensores, alterando las medidas realizadas y modificando el comportamiento normal de los segmentos ante los estímulos de actuación.
- Debe adherir correctamente la superficie del sensor con la de la silicona curada o, en su defecto, hacer de unión entre ambas superficies.

- Debe ser resistente a los esfuerzos normales y cortantes y a la fatiga surgidas por el movimiento del segmento al hincharse durante su actuación.
- Debe ser químicamente estable y neutro. Bajo ningún concepto debe reaccionar ni con la silicona curada ni con el material de los sensores, pues podría comprometer le funcionamiento correcto del sistema.
- Debe ser un material aislante, pues de lo contrario se producirían falsas mediciones.

Todas estas características eran propias de los selladores de silicona. Es por ello que se procedió a buscar un adhesivo que tuviera el comportamiento descrito y fuera similar a los selladores de silicona comerciales. Así se encontró el sellador multiuso **DOWSIL 732 Multi-Purpose Sealant** [34] (figura 3.15), adquirido en Feroca.

Figura 3.15 Sellador de silicona empleado en la fijación de los sensores al robot. *Fuente [34]*

El DOWSIL 732 Multi-Purpose Sealant es un sellador de silicona de un solo componente con una elevada adherencia en múltiples superficies. Alguna de sus excelentes propiedades físicas se pueden apreciar en la tabla 3.2.

CTM[a]	**ASTM**[b]	**Propiedad**	**Unidad**	**Valor**
1377		Viscosidad (alto corte - 10/s)	Pa·s	821
0098		Tiempo de formación de piel	min	7
0095		Tiempo hasta libre pegajosidad al tacto	min	20
Propiedades mecánicas, curado 7 días al aire a 25°C (77°F) y 50 % humedad relativa				
0097B	D1475	Densidad	g/cm^3	1.04
0099	D2240	Dureza	Shore A	25
0137A	D412	Resistencia a tracción	Mpa	2.3
0137A	D412	Elongación hasta rotura	%	540
Propiedades eléctricas, curado 7 días al aire a 25°C (77°F) y 50 % humedad relativa				
0114	D149	Rigidez dieléctrica	kV/mm	21.6
0112	D150	Resistividad de volumen	Ω·cm	$1{,}5 \cdot 10^{15}$

Tabla 3.2 Principales propiedades del sellador DOWSIL 732 Multi-Purpose Sealant. *Fuente: Adaptación de [34]*

[a] CTM: Corporate Test Method
[b] ASTM: American Society for Testing and Materials

Además de estas muy buenas propiedades mecánicas y eléctricas, también presenta otros aspectos que lo hacen óptimo para este proyecto:

- Cura a temperatura ambiente cuando se expone a la humedad del aire.
- Consistencia pastosa de fácil aplicación.
- Cura a una goma resistente y flexible.
- Estable y flexible de -60°C a + 180°C.
- Sistema de curado acetoxi (no tóxico).

Todos estos son los motivos por lo que se decidió adquirir este adhesivo. Posteriormente en el capítulo 5 se explicará el otro uso que se le dio a este sellador debido a estas excelentes propiedades.

A pesar de estas propiedades, se quiso comprobar que realmente no habría ningún inconveniente a la hora de aplicarlo sobre el sensor. Para ello se construyó y diseñó un pequeño molde de prueba con una cavidad para el sensor similar a la empleada, pues de esta forma la prueba sería mucho más precisa. El molde se puede ver en la figura 3.16a y la pieza de silicona ya moldeada con el sensor pegado sobre ella en la figura 3.16b.

(a) Molde para fabricar la pieza de prueba del sensor junto a la silicona

(b) Sensor fijado en la pieza de prueba

Figura 3.16 Molde y pieza para la prueba de la fijación del sensor a la silicona con el sellador. *Fuente: elaboración propia*

Tras la prueba, se concluyó que no afectaba al comportamiento del sensor ni variaba apreciablemente la resistencia del conjunto. por ello, se procedió a su implementación, como se observa en la figura 3.17.

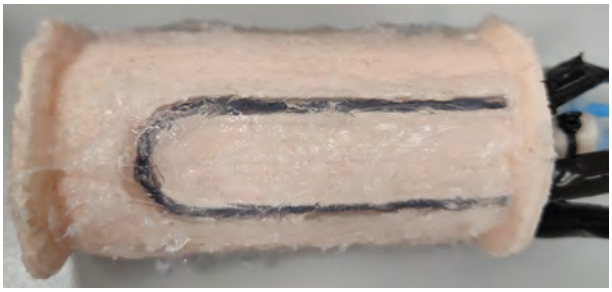


Figura 3.17 Detalle de la fijación de los sensores con el sellador Dowsil 732. *Fuente: elaboración propia*

3.4.2. Instrumentación y montaje final

Para finalizar este capítulo es necesario explicar el montaje de todos los elementos de medición sobre el utillaje previamente instalado en el robot.

Para posicionar los dispositivos de medida y sus conexiones se emplearon tres placas *protoboard* de tamaño mediano, colocadas sobre la esquina inferior izquierda de la cubierta de metacrilato en la parte superior, sobre las que irían los tres INA3221[1]. En esta posición, el hardware de medida se encontraba lo suficientemente cerca tanto de la placa de Arduino Due como del propio robot, además de permitir la manipulación de los componentes y el cableado.

Para la alimentación de 24V de tensión de bus se utilizó una fuente de ordenador ya adquirida por el laboratorio de Robótica y Cibernética del CAR. Dado que la fuente generaba sobre su masa interna tensiones de 3.3, 5 y ±12V, se cortocircuitaron la tierra del Arduino Due con las masas de los dispositivos de medida y con la toma de -12V de la fuente de alimentación. Así, se lograban tensiones de hasta 24V, al usar como "positivo"la toma de 12V.

Es importante señalar que la fuente de alimentación cumple con los requisitos establecidos por la INA3221, pues la corriente máxima que es capaz de generar cuando se utiliza el conector de -12V es de 0.3A, lo cual no supone ningún problema, pues la intensidad de medida máxima será de $\frac{24V}{50 \cdot 10^3 \Omega + 33 \cdot 10^3 \Omega} = 0{,}29 mA$[2], mucho menor que la corriente máxima de la fuente y superior a la corriente mínima para que el INA3221 mida (que experimentalmente se comprobó que se encontraba en el orden de los microamperios).

Para el conexionado de todos estos elementos entre sí se emplearon cables unifilares rígidos de cobre estañado, crimpados en sus extremos, aportando resistencia a las conexiones críticas, y cables dupont que facilitaban la manipulación y el cableado de los dispositivos.

En la figura 3.18, se puede apreciar la posición de cada uno de los componentes comentados con anterioridad.

[1]Debido a la complejidad y a los problemas acaecidos durante el desarrollo de la parte de control solo se dispuso de uno, pues solo se controló uno de los segmentos

[2]Se han empleado resistencias mínimas de trabajo para este cálculo

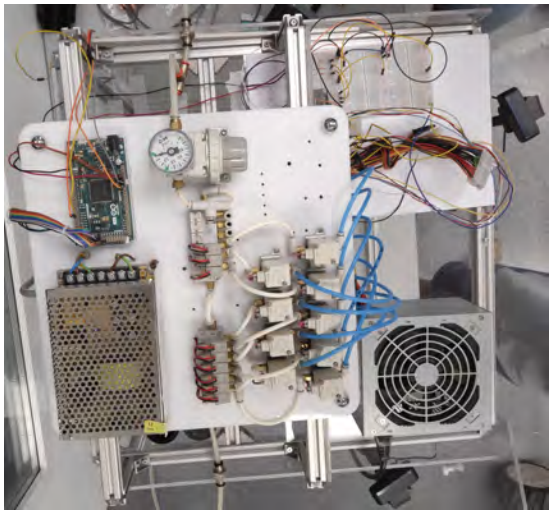
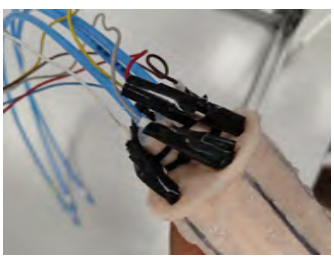


Figura 3.18 Posición de todos los elementos electrónicos para el posterior control del robot. *Fuente: elaboración propia*

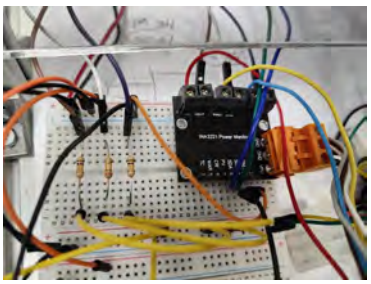
Por último, las conexiones con cada uno de los tres sensores que componían cada segmento se emplearon 4 borneras, que se soldaron al INA3221, y 6 pinzas de cocodrilo (una por extremo de los sensores). Tres de las borneras se emplearon para conectar tanto la parte positiva como la negativa de cada uno de los tres canales de medición del INA3221. La restante se empleó para alimentar la tensión de bus en el dispositivo y dejar otros dos terminales de masa habilitados.

La conexión de cada sensor con el respectivo canal del dispositivo de medida consiste en sujetar los dos extremos de cada sensor con pinzas de cocodrilo, las cuales posteriormente se envolverían en cinta adhesiva para aislarlas del resto y evitar así cortocircuitos. Estas pinzas se conectaban con el INA3221 mediante unos cables multifilares provenientes de un cable HDMI, pues eran flexibles y maleables, lo que no impedía el movimiento normal del robot. Tras ello, se conectaba uno de los extremos del sensor al borne negativo de cada canal de medición, mientras que el otro se conectaba directamente a masa. Posteriormente, el borne positivo del canal de medición se conectaba a uno de los extremos de una resistencia, cuyo otro extremo estaba alimentado con la tensión del bus de 24V. Es así como se completaba el divisor de tensión explicado en las secciones anteriores. Estos detalles en la conexión se pueden apreciar en las imágenes de la figura 3.19.



(a) Conexión de los sensores

(b) Posición de las pinzas de cocodrilo



(c) Cableado y conexión del INA3221 de un segmento

(d) Aislamiento de las pinzas de cocodrilo

(e) Conexión cable - pinza de cocodrilo

Figura 3.19 Detalle de las conexiones entre dispositivo de medida y sensor. *Fuente: elaboración propia*

Esquema eléctrico de conexiones

Por último, el la figura 3.20, se pueden ver las conexiones realizadas entre los dispositivos de medida INA3221 y los sensores.

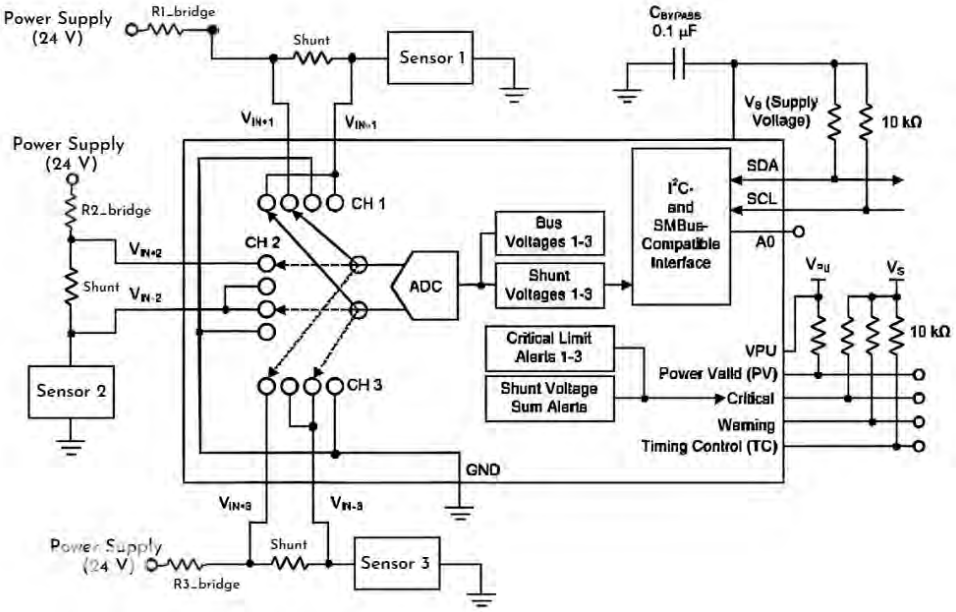


Figura 3.20 Esquema de la conexión entre los dispositivos. *Fuente: elaboración propia basado en [31]*

Cabe mencionar algunas de las conexiones más relevantes. La comunicación entre INA3221 y Arduino Due se inicia a través del puerto I2C previamente configurado de dicho microcontrolador (pin 20 para la línea de datos y pin 21 para el reloj). Dado que esta comunicación emplea el protocolo I2C, se cortocircuitan las líneas de datos y de reloj de todos los dispositivos, pues se emplean distintas direcciones para cada uno de ellos (es necesario además soldar los *jumpers* que habilitan las distintas direcciones del dispositivo).

Capítulo 4

Herramientas utilizadas

Durante el transcurso del proyecto, se emplearon múltiples herramientas para completarlo con éxito, desde herramientas hardware, constructivas o software. Las que más importancia cobraron fueron quizás estas últimas, pues sin ellas no se podría haber el control del *Soft Robot*. Entre ellas destacan las redes neuronales, una herramienta que ha cobrado gran relevancia durante los últimos años debido a su gran potencia de cálculo y resolución de problemas en todo tipo de industrias. Debido a su complejidad técnica, se realiza en este capítulo una explicación en profundidad de sus conceptos más significativos y los detalles tenidos en cuenta para realizar correctamente este trabajo.

También es de gran importancia señalar que se han empleado otras herramientas para el desarrollo de este trabajo, como el software de visión por computador desarrollado por Adrián Siegbert Rieker en su Trabajo de Fin de Grado [1], predecesor a este trabajo, como se ha explicado en el punto 1.3.

A continuación se presentarán las herramientas que han adquirido mayor relevancia durante este proyecto.

4.1. Deep Learning Toolbox de MATLAB

Para este proyecto, en concreto para realizar la parte de control, se ha empleado la Deep Learning Toolbox de MATLAB. Esta es una herramienta poderosa y versátil diseñada para facilitar la creación, entrenamiento y despliegue de redes neuronales profundas y superficiales en una variedad de aplicaciones de aprendizaje automático y procesamiento de datos. Esta *toolbox* proporciona una interfaz sencilla, eficiente y de fácil aprendizaje que permite a todo tipo de usuario trabajar con redes neuronales profundas o superficiales de manera efectiva.

La Deep Learning Toolbox de MATLAB ofrece una gama completa de herramientas y funciones que cubren todo el flujo de trabajo, desde la definición y diseño de modelos

hasta la evaluación y despliegue. Algunos de los aspectos más importantes que incluye esta *toolbox* son:

- **Creación de Modelos**: La *toolbox* proporciona una variedad de arquitecturas de red predefinidas, como redes convolucionales (CNN), redes recurrentes (RNN) y redes neuronales generativas (GAN). También ofrece la posibilidad de diseñar modelos personalizados utilizando la función *layer* para definir capas individuales y la función *layerGraph* para construir arquitecturas completas.

- **Carga y Preprocesamiento de Datos**: Es posible además cargar datos en formatos comunes como imágenes, texto y señales. Incluye también funciones para realizar transformaciones y preprocesamientos, como la normalización de imágenes y la *tokenización* de texto.

- **Entrenamiento y Optimización**: Con la *toolbox* se puede emplear la función *trainNetwork* para entrenar modelos en conjuntos de dato, con la posibilidad de elección entre diferentes algoritmos de optimización. Asimismo, permite la transferencia de aprendizaje para aprovechar modelos preentrenados en tareas similares.

- **Evaluación y Validación**: Permite evaluar el rendimiento de los modelos creados utilizando funciones como *classify*, *predict* y *evaluate*. Estas funciones muestran métricas clave, como precisión y matriz de confusión, para comprender cómo está funcionando el modelo en los datos de prueba.

- **Despliegue y Generación de Código**: Con esta herramienta es posible además desplegar los modelos diseñados a diferentes entornos, como aplicaciones web o sistemas embebidos, generando código C++, CUDA y/o MATLAB para ejecutarlos en un hardware específico.

- **Interfaz Gráfica y AutoML**: La *Deep Learning Toolbox* incluye una interfaz gráfica llamada *Deep Network Designer*, que permite diseñar modelos visualmente sin necesidad de escribir código. También ofrece herramientas de *AutoML* para ayudar a seleccionar y ajustar automáticamente hiperparámetros.

En conclusión, esta herramienta proporcionada por MATLAB permite abordar tareas de diseño, entrenamiento y validación de redes neuronales, ya sean superficiales, como en el caso que aquí se expone, o profundas, como la visión por computador o el procesamiento del lenguaje natural. Con su amplia gama de funciones y su integración con otros programas y productos de MATLAB, la *toolbox* simplifica, agiliza y optimiza el proceso de implementación de redes neuronales en múltiples aplicaciones para la ingeniería, y más concretamente en la industria de la robótica y de la robótica blanda, como ya se vio en el punto 2.2.3.

A continuación se explicarán con mayor detalle los dos principales tipos de redes neuronales empleadas: las redes neuronales prealimentadas o *feedforward neural network* y las redes neuronales de tipo LSTM (Long-Short Term Memory, de sus siglas en inglés).

4.2. Redes Neuronales Prealimentadas (Feedforward Neural Network)

Las *Feedforward Neural Networks* son uno de los tipos de redes neuronales más empleados, especialmente en el campo de la robótica, debido a su excelente versatilidad y sencillez de uso. De hecho, es posible resolver casi cualquier problema numérico con la correcta configuración de una red de este tipo.

Estas redes se caracterizan por tener un flujo de información unidireccional, donde los datos de entrada viajan a través de capas intermedias de neuronas hacia la capa de salida sin formar bucles o conexiones retroactivas. A continuación, se proporciona una explicación del funcionamiento de las redes neuronales *feedforward* y posteriormente se hablará en mayor profundidad de su implementación en MATLAB.

4.2.1. Funcionamiento de una red neuronal prealimentada

Las redes neuronales de tipo *feedforward* son una forma poderosa de modelar relaciones complejas en datos. Un ejemplo que ayuda a entender su funcionamiento es imaginar estas redes como una analogía al sistema nervioso humano, donde cada neurona artificial procesa información y transmite señales a través de conexiones ponderadas a neuronas vecinas. Esta estructura jerárquica de transmisión de impulsos a diferentes neuronas de distintas capas se asemeja a cómo el cerebro humano procesa información en cascada.

En una red *feedforward*, las capas se organizan de manera secuencial:

- **Capa de entrada**: Recibe los datos de entrada, llamados predictores.
- **Capas ocultas**: Transforman y procesan la información. Contienen nodos (neuronas) no observables por el usuario. El valor que exporta cada neurona es una función de los datos de entrada. Asimismo, la forma exacta de la función depende, por un lado, del tipo de red y, por otro lado, de especificaciones controlables por el usuario, como los algoritmos de retropropagación y las funciones de activación.
- **Capa de salida**: Genera los resultados de acuerdo a las interacciones entre neuronas de las capas anteriores ...

Esta disposición unidireccional permite que la información fluya sin retroalimentación, simplificando el análisis y la optimización, así como el tiempo empleado para el entrenamiento de la red. La estructura aquí descrita se puede observar en la figura 4.1.

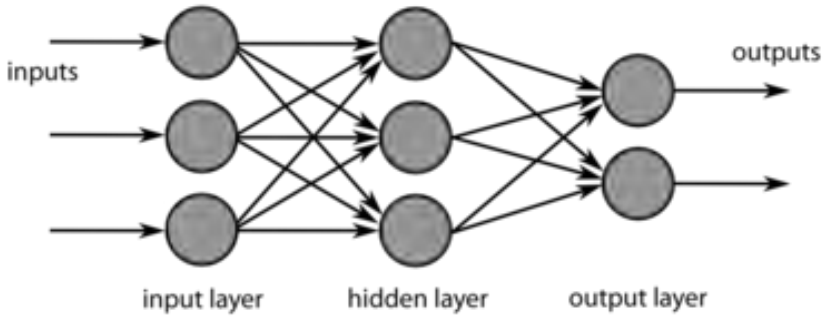


Figura 4.1 Modelo computacional de red neuronal prealimentada de una sola capa oculta. *Fuente [35]*

Cada conexión entre neuronas tiene un peso asociado que modula la influencia de la neurona de origen en la neurona de destino. Al pasar la señal a través de las capas, cada neurona aplica una función de activación a la suma ponderada de las entradas, produciendo una salida. La selección de funciones de activación influye en la no linealidad del modelo, permitiendo la captura y resolución de relaciones tanto sencilla (lineales) como complejas (no lineales).

A continuación se explicará con mayor detalle la función de activación, pues es un componente esencial en una neurona artificial de una red neuronal. Esta define cómo una neurona combina sus entradas ponderadas y produce una salida.

La función de activación toma la suma ponderada de las entradas y el sesgo (*bias*) de la neurona, según la ecuación 4.1, donde w_i es la ponderación (entre 0 y 1) aplicada al dato de entrada i y b es el *bias*, y produce una salida que se pasa a la siguiente capa de la red. Sin funciones de activación, una red neuronal sería equivalente a una serie de transformaciones lineales, lo que limitaría su capacidad para modelar problemas no lineales.

$$\varsigma = w_1 \cdot x_1 + w_2 \cdot x_2 + ... + w_n \cdot x_n + b \tag{4.1}$$

Se trata de funciones matemáticas simples, pues de lo contrario aumentaría su "peso computacional"; es decir, serían muy lentas de procesar por un ordenador, lo que es de gran importancia teniendo en cuenta que las funciones de activación se ejecutan para cada neurona de la red, por cada muestra y por cada ciclo. De esta forma permiten reducir la complejidad de cómputo y evaluar el rango resultante en valores más manejables.

A continuación se explica con mayor detalle el funcionamiento de las funciones de activación más comunes:

- **Función logística** Esta función de activación es un tipo de función sigmoidea muy empleada en el entrenamiento de redes neuronales de todo tipo, aunque más especialmente en problemas de clasificación. Induce no linealidad, lo que es efectivo en la resolución de problemas no lineales, y restringe la salida en un rango entre 0 y

1. Sin embargo, presenta un inconveniente, que es su suavidad. Esto implica que en ocasiones se llegue a un determinado porcentaje de precisión y, a pesar de aumentar las iteraciones, no se mejore el resultado. En la figura 4.2 se puede apreciar su forma y ecuación.

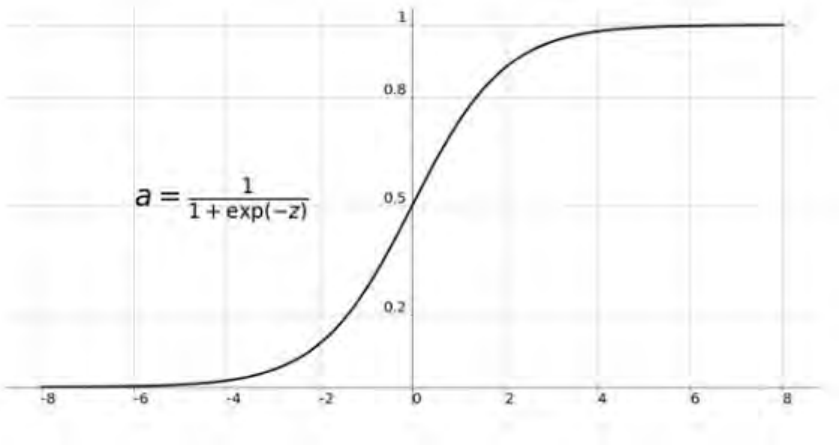


Figura 4.2 Función de activación logística. *Fuente [36]*

- **Función tangencial hiperbólica** Esta otro tipo de función sigmoidea cuyo comportamiento es muy similar al de la función anterior aunque restringiendo las salidas en un rango entre -1 y 1. Esto es útil si se cuenta con valores negativos de entrada, pues se mapearán fuertemente negativos a diferencia de los positivos. Tiene utilidad para problemas no lineales y puede ser más efectiva y rápida que la logística debido a su menor suavidad. En la figura 4.3 se puede observar su forma y ecuación.

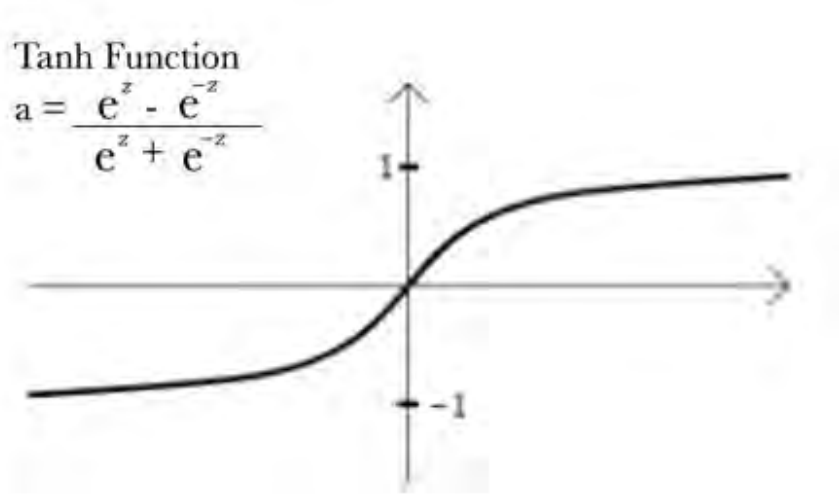
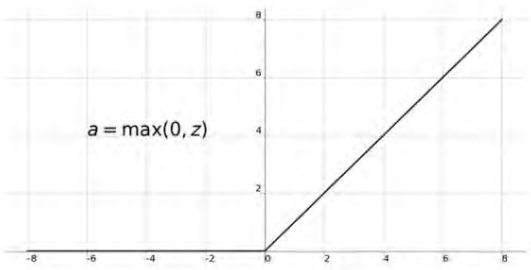


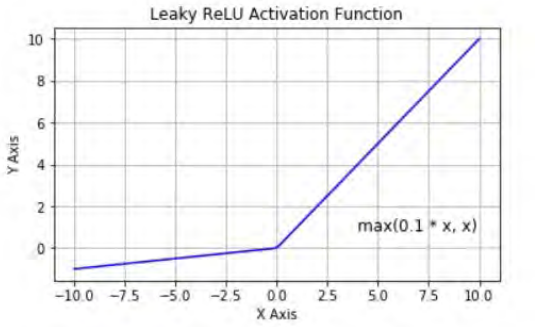
Figura 4.3 Función de activación tangencial hiperbólica. *Fuente [36]*

- **Función ReLU** En este caso, la función ReLU se trata de un tipo de función lineal, pues se limita a suprimir todos los valores negativos y solo tener en cuenta los positivos, que tomarán el mismo valor a la salida que a la entrada. Estos es

especialmente útil para acelerar procesos en los que las entradas y salidas deben ser positivas, pues todos aquellos valores negativos serán inmediatamente descartados al aplicar esta función. Es por ello que se emplea principalmente en el entrenamiento de redes neuronales convolucionales (CNN) de procesamiento de imágenes. En la figura 4.4a se puede ver su forma y ecuación. Existe además un tipo especial de red ReLU, llamada *Leaky ReLU*, que mantiene los valores negativos de los datos pero con menor peso que los positivos. Esta función se puede ver en la figura 4.4b.



(a) Función de activación ReLU



(b) Función de activación Leaky ReLU

Figura 4.4 Funciones de tipo ReLU. *Fuente [36]*

Además de estas funciones, existen muchas otras empleadas en la resolución de distintos problemas, como la función softmax. Aún así, suelen ser más específicas para determinados problemas concretos.

Tras obtener finalmente los datos de la salida de la red, en este tipo de arquitectura de redes neuronales, se comparan con los resultados reales a obtener o *targets*. Esto es lo que se conoce como propagación hacia atrás, retropropagación o *backpropagation*. Con estas diferencias se calcula la función de costo (figura 4.5), también llamada error cuadrático medio, según la ecuación 4.2. Es así que el objetivo del entrenamiento de la red es minimizar esta función mediante la variación de los pesos y los sesgos de cada neurona.

$$C(w,b) = \frac{1}{2n}\sum_{x} \|y(x) - a\|^2 \tag{4.2}$$

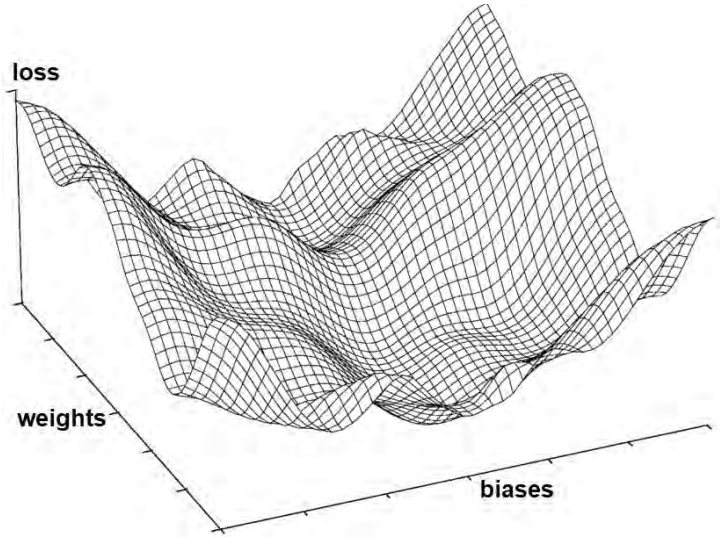


Figura 4.5 Ejemplo de función de costo respecto a los pesos y sesgos de las neuronas. *Fuente [37]*

El método más empleado es el de descenso de gradiente. Consiste en seguir la dirección con sentido opuesto a la otorgada por el gradiente de la función de costo, pues el vector gradiente indica donde se encuentran los mayores incrementos de la función para un determinado punto de su dominio (por ello el cambio de sentido). De esta forma, en cada iteración se modifican los parámetros de las neuronas para dar un pequeño paso en esta dirección. Para ello existen numerosos métodos estadísticos, denominados algoritmos de retropropagación. Estos algoritmos calculan gradientes y realizan ajustes en los pesos utilizando métodos como el descenso del gradiente. A medida que se realiza el entrenamiento durante varias épocas, la red ajusta gradualmente los pesos para mejorar la precisión de las predicciones. Los algoritmos de retropropagación más comunes son Levenberg-Marquardt, gradientes conjugados, gradientes descendentes con momento, regularización bayesiana, BFGS quasi-Newton y retropropagación resiliente.

4.2.2. Redes Neuronales Feedforward en MATLAB

En MATLAB, la creación y entrenamiento de redes *feedforward* se simplifica gracias a la *Deep Learning Toolbox* descrita anteriormente. Siguiendo un enfoque estructurado, aquí están los pasos ampliados para crear y entrenar una red neuronal *feedforward* en MATLAB:

1. **Definir la Arquitectura**
   Para este objetivo se emplea la función *feedforwardnet* que se encarga de crear una red. En ella se puede especificar el número de neuronas en cada capa oculta, el número de capas ocultas y la función de activación de cada una de ellas.

2. **Cargar Datos**
   Previo al entrenamiento de la red, es necesario preparar los datos de entrenamiento y validación. Habitualmente se emplean matrices para almacenar de forma ordenada los datos de entrada y sus respectivas etiquetas de salida.

3. **Dividir los Datos**
   El siguiente paso es dividir los datos en conjuntos de entrenamiento, validación y test utilizando las funciones específicas para ello, como *dividerand*. También es posible modificar los pesos de cada una, pues por defecto se utilizan el 70 % de los datos para entrenamiento y el 15 % para validación y test. Esto permite evaluar el rendimiento de la red en diferentes etapas.

   Destacar que los conjuntos de entrenamiento, validación y test se obtienen automáticamente a partir de los datos de entradas y salidas del sistema proporcionados y los pesos de cada uno de ellos. De esta forma, mediante, por ejemplo, el uso de la función *dividerand*, se consigue una aleatoriedad en los datos contenidos en dichos conjuntos para mejorar el entrenamiento de la misma. Existen también otras funciones de división de datos, como *divideint* o *divideblock*, que en lugar de aleatorizar el reparto de los datos los dividen según un criterio distinto, aunque siempre con el mismo objetivo y según los pesos asignados a cada bloque. En caso de sumar el 100 % o sobrepasarlo, se ajustan los parámetros con la siguiente ecuación:

$$nuevoRatio = \frac{antiguoRatio}{trainRatio + valRatio + testRatio} \tag{4.3}$$

4. **Entrenamiento Personalizado**
   Tras ello se utiliza la función *train*, que se encarga de entrenar la red a partir de la misma y de los datos de entrada y salida objetivo. Es posible además elegir los algoritmos de retropropagación.

5. **Validación y Prueba**
   Una vez entrenada la red neuronal, se debe usar la función *sim* para obtener las predicciones ante las entradas de entrenamiento. También es posible obtener estas predicciones empleando el comando *net(inputData)*.

   Para evaluar el rendimiento de la red y de sus predicciones, se dispone de la función *perform*, que devuelve el error cuadrático medio de la diferencia los valores predichos por la red y los valores objetivo reales.

6. **Ajuste de Parámetros**
   Si es necesario, se pueden ajustar además los hiperparámetros como la tasa de aprendizaje y el número de épocas para mejorar el rendimiento y evitar el sobreentrenamiento de la red.

Señalar que los hiperparámetros son aquellas características de las redes neuronales controlables y definibles por el usuario. Entre ellas se encuentran el número de capas y de neuronas ppor capa, las épocas o número de iteraciones hasta lograr hallar un mínimo del error cuadrático medio, las funciones de activación, los algoritmos de retropropagación o el número de abandonos de neuronas (consistente en ignorar aleatoriamente las neuronas evitando así codependencias entre ellas, la pérdida de independencia de cada una de estas unidades de procesamiento y, finalmente, limitando el sobreajuste de la red, lo que aumenta a su vez el tiempo hasta converger a un mínimo pero reduce a su vez el tiempo de computación de cada iteración).

7. **Visualización**
   Asimismo, se incluyen funciones como *plotperf* y *plotconfusion* para visualizar el rendimiento de la red en términos de rendimiento y matriz de confusión. También es posible observar los gráficos de regresión y el histograma de errores, entre otros, a través de la interfaz de entrenamiento.

8. **Predicciones**
   Una vez entrenada, la red puede utilizarse para realizar predicciones en nuevos datos. Para ello, se debe hacer uso de la función *sim*, introduciendo unas entradas arbitrarias (en principio relacionadas con el problema para el cual ha sido entrenada la red). También es posible obtener estas predicciones empleando el comando *net(inputData)*.

Como se ha explicado anteriormente, MATLAB ofrece una serie de productos como la *Deep Learning Toolbox* que facilita enormemente la creación y entrenamiento efectivo de redes neuronales *feedforward* para diversas aplicaciones. Esta es una de las principales razones del uso de este software, pues permitía entrenar una red suficientemente precisa para la tarea a desarrollar y compatible con toda la arquitectura software desarrollada en MATLAB.

4.3. Redes Neuronales de tipo LTSM

Las redes neuronales de memoria a largo plazo (LSTM) son una variante avanzada de las redes neuronales recurrentes (RNN) diseñada para superar los desafíos de capturar patrones en secuencias temporales largas. Su estructura única permite la retención y el acceso a información relevante a lo largo de intervalos de tiempo extensos.

A continuación, se explicará con mayor profundidad en los detalles técnicos de cómo funcionan estas redes y cómo se implementan en el software de MATLAB.

4.3.1. Arquitectura Básica de una Celda LSTM

La unidad básica en una red LSTM es la celda LSTM. Cada celda almacena información a lo largo de múltiples pasos de tiempo y consta de tres componentes principales:

- **Puerta de Olvido (Forget Gate)**: Esta puerta decide qué información almacenada previamente en la celda debe eliminarse. Utiliza una función sigmoide para generar valores entre 0 y 1, donde 0 implica olvidar completamente y 1 significa retener por completo.
- **Puerta de Entrada (Input Gate)**: Esta puerta determina qué nueva información debe agregarse a la celda. Usa otra función sigmoide para controlar qué partes de los datos de entrada se actualizarán en la celda.
- **Actualización de la Celda**: Aquí, se calcula una "actualización de celda propuesta" utilizando la función tangente hiperbólica. Luego, la puerta de olvido y la puerta de entrada se combinan para actualizar la celda actual. La puerta de olvido controla cuánto de la celda anterior se conserva, mientras que la puerta de entrada controla cuánto de la actualización propuesta se agrega.
- **Puerta de Salida (Output Gate)**: Esta puerta determina la salida de la celda. Emplea una función sigmoide para filtrar el contenido actualizado de la celda. La salida se calcula utilizando la función tangente hiperbólica aplicada al contenido de la celda actualizada multiplicado por la salida de la puerta de salida.

Esta estructura de puertas se ve en la figura 4.6.

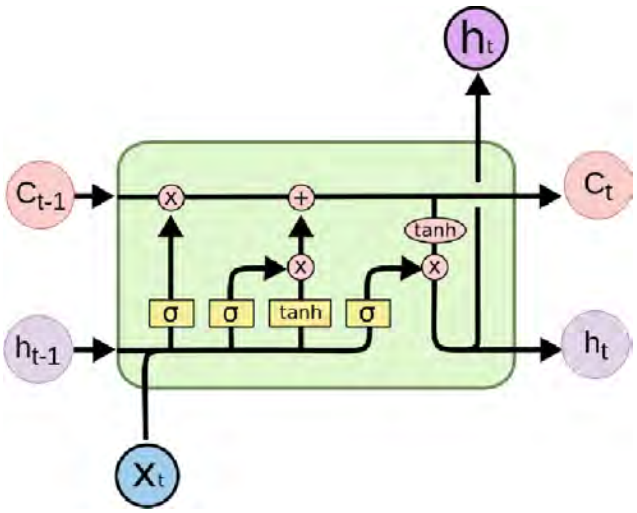


Figura 4.6 Estructura de una celda de una red neuronal recurrente LSTM, donde los valores de c son los correspondientes a la memoria, h representa las salidas y x las entradas. Las celdas en amarillo son redes neuronales simples con sus funciones de activación y los puntos rosa son operaciones específicas para descartar, actualizar y calcular el nuevo estado oculto de la memoria. *Fuente [38]*

Las RNN tradicionales pueden enfrentar el problema del desvanecimiento de gradientes, donde los gradientes de error se vuelven muy pequeños a medida que se retropropagan a través de muchas capas. Esto dificulta el aprendizaje de dependencias a largo plazo. Las LSTM resuelven este problema utilizando la estructura de puertas controladoras. Al ajustar el flujo de información, pueden aprender y retener patrones a lo largo de secuencias largas sin que los gradientes se desvanezcan.

4.3.2. Implementación Práctica en MATLAB

- **Definición de la Capa LSTM**: En MATLAB, es posible definir una capa LSTM utilizando la función *lstmLayer*. Tambien se puede configurar parámetros como la cantidad de unidades LSTM, la función de activación y el tipo de salida deseada.
- **Creación de la Red LSTM**: Para ello se debe usar la función *layerGraph* para construir una gráfica de capas. También permite agregar capas LSTM junto con capas de entrada y salida según la tarea.
- **Preparación de Datos y Entrenamiento**: MATLAB permite además preprocesar datos secuenciales y dividirlos en conjuntos de entrenamiento, validación y prueba. Utilizando la función *trainNetwork* se puede entrenar la red diseñada. Importante señalar que se debe configurar la función de pérdida y el optimizador para mejorar el rendimiento del entrenamiento.
- **Evaluación y Predicción**: Después del entrenamiento, se puede evaluar el rendimiento de la red en el conjunto de prueba. Para ello se emplea la función *predict*, que realiza predicciones en nuevas secuencias.

Cabe destacar que las redes neuronales LSTM representan un avance significativo en el campo del aprendizaje profundo al abordar el desafío de capturar patrones en secuencias temporales largas. Su estructura de celdas con puertas controladoras permite el aprendizaje efectivo de dependencias a largo plazo, lo que las convierte en una herramienta poderosa para aplicaciones que involucran datos secuenciales, como el procesamiento del lenguaje natural, la traducción automática y la predicción de series temporales. Con una implementación adecuada en plataformas como MATLAB, las LSTM pueden ofrecer soluciones sofisticadas a problemas complejos en una variedad de campos.

Capítulo 5

Modificaciones del diseño, fabricación y control de los segmentos

Uno de los aspectos más problemáticos del proyecto fue la compatibilización del diseño realizado en el trabajo de Adrián Siegbert Rieker [1] con el desarrollo del control y la implementación de los sensores en el robot.

Durante el desarrollo de este trabajo surgieron numerosos problemas relacionados no solo con el proceso de fabricación y los materiales implicados sino también con ciertas decisiones de diseño que, aunque excelentemente ejecutadas, no eran las más óptimas para el correcto desarrollo de un control autónomo en el robot.

Es por ello que se procedió además a realizar algunas variaciones que indujeron a su vez a diversos problemas, finalmente resueltos de forma exitosa. A continuación se especifican y comentan en mayor detalle los cambios realizados, sus problemas asociados y la resolución final.

Asimismo, también se modificaron algunos aspectos menores de la arquitectura software ya desarrollada.

5.1. Cambios en el proceso de fabricación

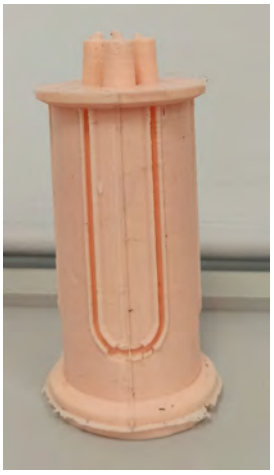
En esta sección se explicarán los principales cambios con respecto a la fabricación de los segmentos que componen a PAUL. Es importante destacar que todos estos cambios tuvieron el objetivo de complementar y optimizar el diseño final del robot para poder facilitar el desarrollo del control automatizado.

5.1.1. Cambio de posición de las vejigas

Este fue uno de los primeros cambios realizados respecto al método de fabricación de los sensores, y quizá el que más problemas conllevó a posteriori.

Inicialmente, las vejigas internas de los segmentos, aquellas que permiten la deformación del robot por la entrada de aire comprimido, estaban ubicadas internamente en el espacio entre el hueco disponible para los dos sensores, como se puede ver en la figura 5.1a. Esto tenía una idea clara, evitar fugas producidas por el estiramiento de la silicona con la que se fabricaban los segmentos al introducir en la cavidad el aire del sistema de actuación del robot. Esta es la principal razón por la que se dispuso de esta manera.

(a) Ubicación inicial de las cámaras internas del robot



(b) Ubicación final de las cámaras internas del robot

Figura 5.1 Comparación de la posición de las cámaras internas de PAUL antes y después de su cambio. Su ubicación se puede deducir a partir de los salientes superiores para introducir los tubos neumáticos que conducen el aire a su interior *Fuente: elaboración propia*

El principal problema surge posteriormente, con la implementación de un método de control. Este implica que debe ser lo más sencillo posible, pues la complejidad del problema del control es elevada para este tipo de robots. Es por ello que si se disponían de aquella manera los sensores, la deformación que sufría el robot al introducir aire en la cámaras internas provocaría que los sensores que se encontraban a los lados de dicha vejiga medirían cambios en la forma del robot. Esto es un problema en sí mismo, pues las medidas de los sensores estarían acopladas entre sí (pues por vejiga medirían dos sensores de los tres, por lo que para detectar un cambio se debería adquirir los datos de dos de los tres sensores, como mínimo) y la dificultad del problema incrementaría excesivamente. Además, este problema se une al hecho de que al ser sensores resistivos elastómeros, cambios iguales en su longitud varían de la misma forma la resistencia del sensor, con lo que se darían medidas iguales para la misma deformación en las cámaras situadas a la izquierda y a la

derecha de cada uno de los sensores.

Todo ello llevó a la decisión de ubicar las vejigas internas justo detrás de la cavidad diseñada para el posicionamiento de los sensores en el segmento. Esta nueva ubicación se puede apreciar en la figura 5.1b. De esta forma, se logra que cada hinchado de una de las cámaras se vea reflejado mayoritariamente en uno de los sensores, que será el que variará en mayor medida su resistencia. Así, aunque había todavía pequeños acoplamientos entre las medidas de los sensores, se facilita el control del mismo al tener una noción más clara y a simple vista del desplazamiento que ha sufrido y su forma.

Sin embargo, apareció como consecuencia un gran inconveniente: debido a la delgada pared que separaba la cavidad interna de la vejiga y el hueco disponible donde ubicar los sensores, unido a las burbujas contenidas en el material (pues no se disponía de una bomba de vacío), se producían múltiples fugas y pinchazos que entorpecían la labor de extracción de datos y actuación del robot.

Con el objetivo de disminuir o, incluso, eliminar este defecto, se idearon varias soluciones. Algunas de ellas se describen a continuación.

5.1.2. Método de sellado de fugas

Otro de los principales cambios en la fabricación fue el método de sellado de los agujeros que quedaban tras la eliminación de los machos de cera.

Anteriormente, se empleaba la propia silicona de la que estaban fabricados los segmentos (TinSil 8015) junto a una pieza impresa en 3D. De esta forma, tras el curado de la mezcla, quedaban correctamente sellados dichos huecos.

En cambio, en el caso aquí expuesto y tras la adquisición del sellador de silicona Dowsil 732, se empleó, además de para fijar los sensores, para el sellado de estos orificios. De esta forma se reducía el tiempo de fabricación en 24h, pues este es el tiempo de curado de la silicona TinSil 8015, a la vez que se conseguía reducir el peso del conjunto y rellenar de forma más eficaz dichos huecos. Esta modificación se puede ver en la figura 5.2.

Figura 5.2 Detalle de la parte inferior de un segmento. *Fuente: elaboración propia*

Asimismo, este sellador se empleó en otras múltiples tareas, como la reparación de pinchazos que surgían durante el proceso, debido a la gran cantidad de pequeñas burbujas en el interior de la silicona curada y al cambio comentado en el punto anterior, o en el sellado y fijación de los tubos neumáticos dentro de las cámaras de los segmentos. Esta última aplicación se puede observar en la figura 5.3.

Figura 5.3 Detalle de la fijación de los tubos neumáticos a un segmento del robot. *Fuente: elaboración propia*

5.1.3. Eliminación del espacio interno del segmento

La tercera modificación sustancial en la fabricación de los segmentos de silicona fue la eliminación del macho central impreso en 3D durante el moldeado de los segmentos. Esta fue una consecuencia y una de las pocas soluciones factibles (que no implicara rehacer o cambiar partes importantes del diseño de los moldes de fabricación de los segmentos del robot) al problema presentado en punto 5.1.1.

El problema que lleva a la eliminación de esta parte interna del segmento es la pequeña pared que separaba las cámaras internas del hueco donde introducir los sensores, como ya se explicó anteriormente. Para resolverlo, una primera solución fue utilizar al máximo posible el juego existente en los orificios donde se apoyaban los machos de cera que creaban estas cavidades internas. De esta forma, se podían separar del relieve que generaba el surco donde posteriormente irían los sensores y generar así una pared interna más gruesa. Sin embargo, el alejar el macho de cera de la pared exterior implica acercarlo cada vez más al macho central que generaba este espacio interno donde iban los tubos neumáticos de los segmentos inferiores, lo que implica una disminución del grosor de dicha pared de separación. Es este inconveniente el que obligará posteriormente a la retirada de este macho interior.

Para las primeras pruebas de moldeo poniendo a prueba la idea mencionada en el párrafo anterior no se dio especial importancia al problema que presentaba, pues se pensaba que no supondría un inconveniente excesivamente difícil de resolver en comparación

con el que ya presentaban los segmentos al modificar la posición de las vejigas internas. Sin embargo, resultó en un problema aún mayor, pues los pinchazos y fugas exteriores no habían cesado y, además, ahora se pinchaba con mucha mayor frecuencia en la zona interna.

Así, antes de tomar la decisión de eliminar por completo este hueco interno, se probaron otros métodos, como reforzar y reparar las paredes internas con el sellador Dowsil 732, con la esperanza de que, al presentar mayor dureza que la silicona de fabricación, impidiera el cierta medida el afloramiento de nuevas fugas. Aún así, la naturaleza viscosa del sellador hacía muy difícil su aplicación en la zona central del hueco. Se intentaron múltiples métodos, como aplicarlo en la parte superior y esperar a que descendiera como si de una cascada se tratase por el efecto de la gravedad; extender la sustancia con un objeto alargado, como un palo; o incluso aplicarlo utilizando una jeringuilla con el extremo alargado. Sin embargo, las su viscosidad, su adherencia y la rápida formación de una capa exterior que presentaba el sellador hicieron inviable cualquier método de aplicación en esa zona central.

Debido a todo ello, se tomó la decisión en última instancia de eliminar el macho que generaba este espacio interno, pues de esta forma se podrían alejar lo máximo posible los machos de cera de la pared exterior sin afectar al interior del segmento. En la figura 5.4 se aprecia un segmento ya curado con el espacio interno completamente relleno debido a la falta de la pieza interior comentada.

Figura 5.4 Detalle del moldeo de un segmento sin cavidad interna. *Fuente: elaboración propia*

5.1.4. Cambio de forma de los machos de cera

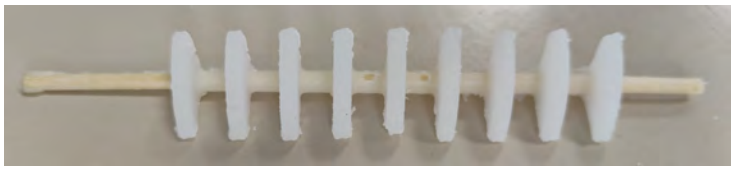
La decisión de implementar esta última modificación tuvo lugar después de un incidente observado al alejar los machos de cera de las paredes exteriores tras la eliminación del espacio interno central. Este fenómeno estaba directamente relacionado con la geometría

de los machos.

La forma geométrica de los machos de cera era especialmente compleja. Estos a su vez se componían de nueve pequeñas piezas de cera con la forma de un semicírculo cuyos vértices habían sido redondeados para evitar los concentradores de tensión y mejorar su funcionalidad y estética. Sin embargo, esta geometría presentaba un inconveniente, inicialmente imperceptible debido a la prescindencia de separar los machos de cera del relieve que genera el hueco para los sensores: las paredes que separan las cámaras internas entre sí adelgazan cuanto más cerca del centro se encuentran los machos de cera, pues el diámetro de la circunferencia interna que pasa por los puntos más interiores de cada cámara se reduce.

Es debido a este problema que se decidió recortar de forma manual los machos de cera una vez fabricados por sus extremos, generando una forma similar a un cuadrante de círculo. Se puede ver esta nueva forma en la figura 5.5 junto a la forma de los machos anteriores.

(a) Antigua forma de los machos de cera. *Fuente [1]*



(b) Nueva forma de los machos de cera. *Fuente: elaboración propia*

Figura 5.5 Comparación entre la geometría nueva y antigua de los machos de cera que crean las cavidades interiores de los segmentos

Las consecuencias de esta reducción fueron en su mayoría beneficiosas, destacando el engrosamiento de las paredes internas de separación entre vejigas y una reducción en el número de fugas en las paredes exteriores, pues ahora era posible alejar lo máximo posible los machos de la pared exterior. Cabe destacar la considerada única desventaja de esta modificación: la reducción de la deformación del segmento para una misma presión de aire. Esto se pudo contrarrestar posteriormente con un aumento de los tiempos de hinchado de las vejigas.

5.2. Cambios en el diseño

Además de los cambios explicados en la sección anterior relacionados con el proceso de fabricación, también se llevaron a cabo modificaciones en el diseño de determinadas piezas de unión. En su mayoría, estos cambios estuvieron motivados por los problemas presentados durante el proceso de fabricación y la resolución que aquellos implicaron, en muchas ocasiones relacionada directamente con las piezas de unión entre segmentos.

A continuación se presentan los principales cambios de diseño de las piezas 3D que conforman el robot.

5.2.1. Nuevas piezas

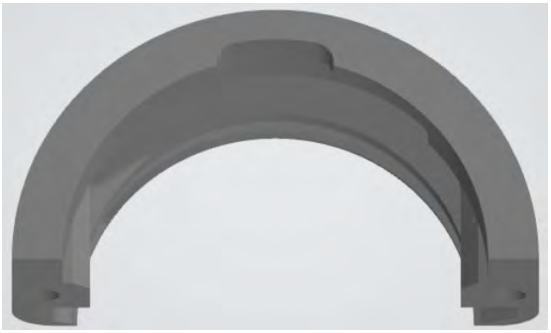
En este punto se presentarán aquellas piezas que sufrieron algún cambio en su diseño debido a los nuevos requisitos que debían presentar tras las modificaciones realizadas en el proceso de fabricación o debido a la incorporación de los sensores en el robot.

Pieza de conexión a la base o Sol

El primero de los cambios acometidos en lo respectivo al diseño de las piezas que conforman el robot fue la pieza de unión del primer segmento con la base (parte superior del utillaje). Se hicieron dos cambios, el primero consistió en un ensanchamiento de la pieza y el segundo en un alargamiento junto a la realización de una ranura en su pared exterior. A continuación se explicará con mayor detalle dichos cambios.

1. **Primera versión**
   Esta primera versión empleada posteriormente en el control consistía principalmente en ensanchar el hueco interno existente, pues al colocar los sensores junto a las pinzas de cocodrilo que realizaban la conexión con el dispositivo de medida se observó que estos no cabían correctamente en dicho espacio. Es por ello que se realizó un agujero de mayor tamaño, como se puede observar en el antes y el después de la figura 5.6.

(a) Antigua forma del *sol. Fuente [1]*



(b) Primera modificación del *sol. Fuente: elaboración propia*

Figura 5.6 Comparación entre los diseños antiguo y nuevo de la pieza de conexión con el utillaje (sol).

2. **Segunda versión**
   La segunda versión fue un diseño realizado para el control de los tres segmentos que conformaban el robot, aunque finalmente no se pudo emplear. Este diseño resolvía el problema generado al eliminar el espacio interno entre segmentos, pues este estaba destinado a albergar los tubos neumáticos y cables de conexión de los segmentos

inferiores. De esta forma, mediante la realización de una ranura en su pared exterior se lograba poder conectar dichos elementos con los segmentos restantes por el exterior. Es por ello que la pieza fue además alargada, lo que ayudaba a ampliar el espacio interno disponible para las pinzas de cocodrilo del segmento superior. Esta nueva versión se puede apreciar en la figura 5.7.

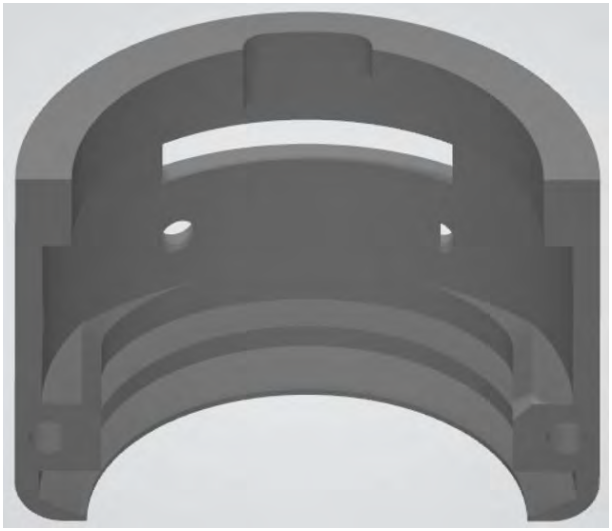


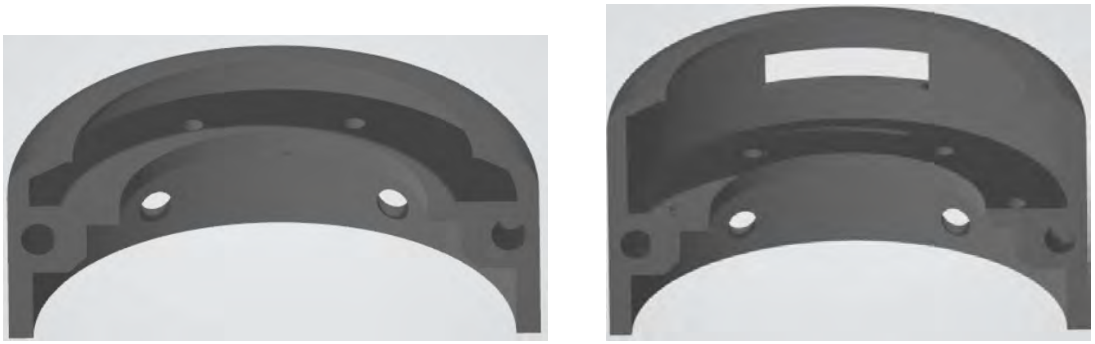
Figura 5.7 Segunda modificación del *sol. Fuente: elaboración propia*

Señalar que las piezas mostradas son diseños en 3D con perspectivas ortográficas, para una mejor visión de los detalles. Además, se debe tener en cuenta que las piezas aquí mostradas son la mitad de un conjunto de dos piezas iguales que irán unidas posteriormente mediante tornillería.

Pieza de conexión entre segmentos

Al igual que ocurría con la pieza anterior, las uniones de los segmentos, en concreto las denominadas *hembras*, debían ser modificadas para poder introducir a través de ellas los tubos neumáticos, a la vez que se ampliaba para lograr un mayor espacio interno donde introducir las pinzas de cocodrilo que conectarían los sensores de los segmentos inferiores al dispositivo de medida.

La modificación realizada con respecto a la pieza ya existente fue una ampliación de la cavidad interna que dejaba una vez montada mediante el alargamiento de su figura. También se realizó una ranura de 16 mm de longitud y 5 mm de altura, pues los tubos miden 4 mm de diámetro exterior y se albergan 3, por lo que se deja espacio adicional más que suficiente para introducir los cables de conexión con los sensores. De esta forma se solucionaban todos los problemas surgidos por la decisión de eliminar el macho interno de PLA. La comparación de ambos diseños se puede ver en la figura 5.8.



(a) Pieza de unión *hembra* anterior. *Fuente [1]*

(b) Pieza de unión *hembra* nueva. *Fuente: elaboración propia*

Figura 5.8 Comparación entre los diseños antiguo y nuevo de la pieza de conexión entre segmentos de tipo *hembra*.

Como se puede deducir de lo anterior, el diseño de la pieza de tipo *macho* se mantuvo, pues no era necesario aplicar ninguna modificación adicional.

Triedro de captura de posiciones

Adicionalmente se modificó la baliza luminosa que se empleaba en la captura de la posición del extremo del robot.

Anteriormente, el *triedro* [1] contaba con una base para el anclaje al extremo del robot sobre la que se disponía una estructura compleja que permitía la captura de las posiciones y orientaciones del extremo del robot. Esta baliza luminosa se componía de tres pequeñas varillas en cuyos extremos se hallaban tres esferas huecas fabricadas en PLA mediante impresión 3D con tres diodos LED contenidos en su interior. De esta forma, y gracias a una varilla central que separaba el conjunto de la base, se conseguía capturar todos los puntos de paso del robot. Los diodos LED permitían una mejor captura de los colores por parte de las cámaras tras el procesamiento de la imagen mediante el software de visión por computador de MATLAB.

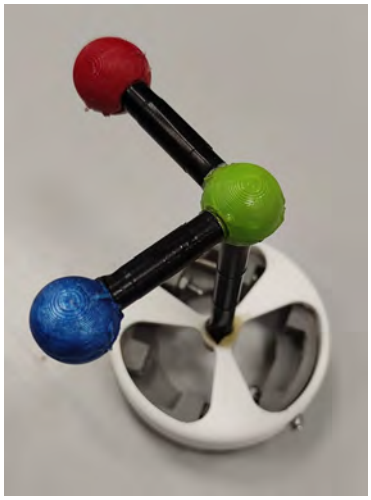
Para el nuevo *triedro* se sustituyeron todos los elementos salvo la base, de la que se aprovechó el diseño ya realizado. La nueva baliza estaría compuesta por tres esferas macizas fabricadas en PLA mediante impresión 3D, completamente rediseñadas para poder introducir y fijar mediante pegamento termofusible tres varillas de aluminio huecas de 6 mm de diámetro exterior. De esta forma se obtiene una mayor rigidez en la estructura del conjunto.

Tras ello, se procede al pintado de las esferas según los colores que presentaba la baliza luminosa anterior mediante el uso de pintura en spray. De esta forma se obtiene un color mucho más uniforme y se elimina la necesidad de controlar, cablear y encender los diodos LED del anterior diseño. A pesar de ello, la luminosidad afectará en mayor medida en este caso, al no contar con la luz desprendida por los diodos LED.

Finalmente, se procederá al recubrimiento del aluminio con cinta adhesiva negra, pues

de esta forma se evitan reflejos indeseados a la vez que se aporta contraste a los colores de las esferas. La comparación entre ambos diseños se puede ver en la figura 5.9.

(a) Antiguo diseño del triedro. *Fuente [1]*



(b) Nuevo diseño del triedro. *Fuente: elaboración propia*

Figura 5.9 Comparación entre los diseños antiguo y nuevo de la baliza de captura de posición y orientación del extremo del robot

5.3. Cambios en la forma de control del robot

Adicionalmente a los cambios realizados en el proceso de fabricación y en el diseño de las piezas, se aplicaron algunos cambios a la arquitectura e interfaz de control del robot. La motivación para realizar estos cambios fue principalmente la prescindencia de la interfaz de control manual, pues uno de los objetivos de este proyecto era el de diseñar un método de control del robot autónomo, sin necesidad de operarlo de manera manual. Algunos de estos cambios también se realizaron con el objetivo de solventar algunas pequeñas problemáticas surgidas durante el desarrollo del control.

En este punto se explicarán estos últimos cambios, los menores, puesto que el cambio más radical se comentará con mucho mayor detalle en el capítulo 6. Se puede adelantar que este cambio de mayor peso fue la eliminación y sustitución de la interfaz de control.

1. **Reducción del campo de trabajo**

   Para el control manual anterior, se limitó el campo de trabajo mediante una actuación máxima (tiempo de apertura de válvula) de 1000ms. La razón de esto era que para mayores caudales de aire podría producirse el pinchazo del segmento.

   En el trabajo que aquí se describe se redujo este tiempo de apertura máxima a 900ms, pues, después de toda la problemática surgida por la modificación de la ubicación de las cámaras internas del robot, reduciría la probabilidad de perforar por la

presión interna la silicona y duraría un mayor tiempo. Posteriormente se demostró que así era, pues alcanzó, como se verá posteriormente, más de 3000 posiciones diferentes sin romperse (mayor al número alcanzado en el anterior proyecto). Además, con este cambio también se lograba una reducción en los efectos de la fatiga en el material.

2. **Modificación de los umbrales (*thresholds*) de captura de colores por visón con computador**
   Al haber eliminado los diodos LED de la baliza (*triedro*) y cambiado ligeramente los colores, era necesario modificar los valores de los *thresholds* que permitían capturar los colores rojo, azul y verde a través de las *webcams*. Es por ello que se procedió a su modificación manual por prueba y error hasta que se comprobó que eran captados y procesados correctamente.

3. **Corrección del software de apertura de válvula**
   Este cambio fue muy ligero. Se procedió a suprimir un error que se encontraba en el código original, pues se aumentó el valor teórico de tiempo de desinflado un 40 % adicional de lo que le correspondía. Esto se intuye que fue un error, pues el valor real sí debía ser modificado por la asimetría existente entre el tiempo de inflado con respecto al de desinflado, pues uno es forzado mientras que el otro en "natural". Esto afectaba al modo de desinflado, pues en la realidad se desinflaba menos tiempo del que debía.

Capítulo 6

Diseño y desarrollo del sistema de control para un segmento

6.1. Introducción

En este capítulo se comentará y explicará con detalle todo lo relacionado con el desarrollo del sistema de control del *Soft Robot* diseñado por Adrián Siegbert Rieker en su trabajo [1], continuado durante este proyecto. Para ello, se ha empleado parte del software ya realizado por Adrián, especialmente aquel relacionado con la visión con computador y con el accionamiento de la válvulas y del banco neumático, y se han añadido otras funcionalidades de control. Todo ello ha sido recogido en una clase, la clase *Robot*, como atributos y métodos para facilitar su empleo por el usuario en futuras aplicaciones. Para llevarlo a cabo se ha utilizado la versión R2022b de MATLAB, igual que en el trabajo anterior, pues de esta forma se suprimían los problemas de incompatibilidad.

Cabe destacar que por motivos de alcance del proyecto y debido a la complejidad que presentó, solo se realizó y probó el sistema de control de uno de los tres segmentos que componían el robot. A pesar de ello, el sistema aquí diseñado es perfectamente extrapolable al robot completo, como se explicará más adelante.

Para comenzar con este capítulo, se describirá el cambio referente a la interfaz de control con respecto al trabajo anterior.

6.2. La clase Robot

Para el control del *Soft Robot*, era necesario un software donde añadir determinadas funciones de forma rápida y eficiente, así como almacenar los datos capturados y las pruebas realizadas de una manera organizada y fácilmente accesible.

La interfaz de usuario hasta ahora realizada no terminaba de cumplir con estos obje-

tivos, pues aunque muy cuidada y excelentemente ejecutada, contenía múltiples pestañas innecesarias para cumplir con los objetivos de este proyecto, como el control manual o mediante tabla desarrollados por Adrián. Además, la aplicación se debía reiniciar por cada modificación de parámetros o en las funciones internas, lo que sumado a la lentitud en el inicio de la aplicación debido a la gran cantidad de elementos a iniciar hacía que se perdiera gran cantidad de tiempo.

Es por ello que se tomó la decisión de crear una clase que concentrara todas aquellas funciones empleadas en la comunicación mediante UART con el microcontrolador o en la captura de imágenes mediante las cámaras como métodos de la clase y todos aquellos parámetros internos de cada una de ellas como atributos. De esta forma, se podrían añadir con extrema sencillez nuevas funcionalidades sin la necesidad de un reinicio pesado y lento de la aplicación de MATLAB, pues solo era necesario volver a instanciar la clase generando un objeto que contuviera todos los cambios realizados en sus métodos y atributos.

Adicionalmente, se mantenía el principio de encapsulamiento, pues se diseñaban los métodos para ser usados por el usuario sin necesidad de que este supiera de su implementación y sin poder acceder directamente a los atributos de la clase.

La única desventaja que presentaba con respecto al anterior planteamiento era la intuitividad que presenta una aplicación en comparación con una clase para el usuario inexperto, aunque es contrarrestable gracias a la facilidad en el aprendizaje del uso de los métodos de la clase.

Asimismo, para el almacenamiento y disponibilidad de los *datasets* o de los métodos de entrenamiento, se dispuso de carpetas en la que se almacenaban todos aquellos archivos que contenían las variables generadas durante el procedimiento de extracción de datos para cada *dataset* y los *scripts* con los códigos para el entrenamiento.

Las distintas partes en las que se dividen los métodos de la clase Robot son:

Atributos

Los atributos de nueva implementación en la clase son:

- *serialDevice*: Este atributo guarda las características (como el nombre o la tasa de baudios (*baudrate*) del puerto serie que conecta el microcontrolador al ordenador.
- *realMode*: Este atributo indica si se está trabajando con el robot real o no. Se utiliza para simular las características de la clase.
- *deflatingTime*: En este atributo se guarda el tiempo empleado para el desinflado total de las vejigas.
- *deflatingRatio*: Esta variable almacena el valor experimental aproximado que relaciona de forma lineal (a pesar de no serlo) la relación entre el tiempo de inflado y de desinflado. Más adelante se explicará esta asimetría con mayor detalle.

- *maxAction*: Este atributo contiene el valor máximo de actuación por cada iteración del bucle de control. Al igual que en el caso anterior, el sistema de control del robot se explicará posteriormente.

- *max_millis*: En esta variable se guarda el valor del máximo tiempo de inflado que puede experimentar cada una de la vejigas del segmento. Esta es una medida indirecta de la cantidad de aire máxima que puede contener cada vejiga en su interior.

- *nValves*: Aquí se almacena el número de válvulas con el que cuenta el robot.

- *nSensors*: Similar al atributo anterior, el número aquí guardado indica el número de sensores con el que se cuenta.

- *geom*: Este atributo de tipo *struct* se emplea para almacenar los datos relacionados con la geometría del segmento, como su longitud, diámetro, distancia desde la esfera verde de la baliza 1 hasta la parte inferior del robot o la distancia desde este mismo punto hasta la esfera verde del segundo *triedro*.

- *base*: Este valor recoge el punto en el que se encuentra el sistema de referencia fijo del robot. En este caso, se corresponde con la ubicación de la parte superior del segmento superior, coincidente a su vez con el *sol* y el centro de la placa de metacrilato superior.

- *bottom*: En este atributo se guarda el valor del punto donde se encuenta la parte inferior del robot. En este caso, coincide con la parte inferior del segmento a controlar.

- *origin*: Similar a los atributos anteiriores, aqui se recogen las posiciones de las tres esferas que conforman la baliza denominada *triedro*.

- *millisSentToValves*: Este valor almacena el último valor de tiempos enviado a las cámaras del robot.

- *voltages*: Este atributo almacena todos los valores de tensiones cada vez que se realiza una medida.

- *positions*: Similar al caso anterior, aquí se guardan todas las posiciones capturadas desde la creación del objeto.

- *serialData*: Este atributo recoge las línea de texto con los datos a enviar a través de la comunicación UART MATLAB-Arduino.

- *max_min*: En esta variable se guardan los valores de máxima y mínima medida de cada uno de los sensores.

- *tol*: Mediante este atributo se especifica el valor de error máximo (tolerancia) para considerar que se ha alcanzado la posición objetivo durante el control del robot.

- *net_pt*: Este atributo recoge la red neuronal entrenada cuyas entradas son posiciones y cuyas predicciones son tiempos de actuación.

- *net_vt*: Este atributo recoge la red neuronal entrenada cuyas entradas son voltages y cuyas predicciones son tiempos de actuación. Este atributo y el anterior serán explicados más adelante.

- *NewMeasure*: La finalidad de este atributo es indicar si se ha recibido una nueva medida o por el contrario no se ha enviado la señal para la recepción de los datos de una nueva medición. De esta forma, se genera un evento que permitirá activar un *callback* asociado a la medición de tensiones

- *OP_mode*: Por último, este atributo recoge el modo e operación en el que se encuentra. Su finalidad será especificar si se está empleando para la captura de posiciones el la baliza número 1 o la baliza número 2. Esta segunda baliza se detallará en el capítulo siguiente.

Aquí solo se han considerado los atributos de nueva creación. Además de estos, existen muchos otros relacionados con los parámetros extrínsecos e intrínsecos de las cámaras, así como otros parámetros necesarios para la comunicación o el control de las válvulas de acuación, todos ellos ya descritos en el trabajo de Adrián Rieker [1].

Métodos de la clase

Adicionalmente a los atributos, se han implementado gran cantidad de nuevas funcionalidades al robot. Estos nuevos métodos se dividen según la función que realizan según los siguientes grupos:

1. **Construcción, destrucción e inicialización del objeto de la clase**
   Los atributos pertenecientes a este grupo son dos:

   - Método *Robot*: Este es el constructor de la clase. Además de crear el nuevo objeto, inicializa los valores geométricos contenidos en el atributo *geom*, genera las direcciones de los ejes del sistema fijo y activa el *callback* asociado a la medida de los sensores.
   - Método *delete*: Este método hace las veces de destructor del objeto instanciado.

2. **Conexión MATLAB-Arduino**
   Este grupo cuenta con 3 métodos:

- Método *Connect*: Este método inicializa el puerto de comunicación serie entre el ordenador y el microcontrolador. A su vez, configura el *callback* de recepción de datos enviados por el microcontrolador.
- Método *Disconnect*: Al contrario que en el caso anterior, esta funcionalidad destruye la conexión entre MATLAB y el Arduino Due.
- Método *Rearme*: Este método proviene del trabajo de Adrián [1] y se emplea para reiniciar la conexión con el microcontrolador en caso de que se alcance un estado de emergencia.

3. **Calibración y captura de la posición con las cámaras**
   El grupo aquí expuesto cuenta con 4 métodos. Los tres primeros provienen del trabajo de Adrián Rieker [1], por lo que solo se expondrá su funcionalidad. Para maor detalle, consultar la fuente de origen.

   - Método *MakeAxis*: Este método inicializa las gráficas donde se incluirán posteriormente las imágenes tomadas.
   - Método *CalibrateCameras*: La finalidad de este método es el de calibrar las cámaras mediante la definición de sus parámetros extrínsecos. Asimismo, se debe tener previamente calculados los parámetros intrínsecos. Estos se hallan mediante el uso de una cuadrícula o damero y la herramienta de visión por computador de MATLAB.
   - Método *CapturePosition*: Este método se emplea para hallar la posición del extremo del robot mediante la captura y procesamiento de las imágenes tomadas por ambas cámaras gracias a la baliza construida.
   - Método *Set_ OP_ mode*: Este método permite modificar el modo de operación, empleado para seleccionar uno u otro *triedo* de posicionamiento del extremo.

4. **Envío y lectura de las señales de actuación y las mediciones de los sensores**
   En este grupo se pueden encontrar 10 métodos. Al igual que para el grupo anterior, muchos de estos métodos proceden o están basados en los implementados en el trabajo de Adrián Rieker [1], especialmente aquellos que guardan relación con la actuación de las válvulas.

   - Método *Deflate*: Este método será el empleado para efectuar un desinflado total de todas las vejigas. Los tiempos de actuación enviados son el resultado de la multiplicación de los atributos *deflatingRatio* y *deflatingTime*.
   - Método *WriteOneValveMillis*: La funcionalidad aportada por este método es la de enviar un valor de tiempo de actuación (tanto positivo como negativo) exclusivamente a una de las válvulas de actuación del robot.

- Método *WriteSegmentMillis*: Como su nombre en inglés indica, este método hace uso del anterior para hinchar o deshinchar un segmento entero, pasado como parámetro a la función.
- Mñetodo *GetMillisSent*: Este método se emplea para conocer el estado del volumen de aire dentro de cada vejiga, medido como tiempo de actuación enviado acumulado.
- Método *GetMillis*: Su funcionalidad es idéntica al método anterior. La diferencia es que en este caso, el tiempo enviado sí está multiplicado por el valor del atributo *deflatingRatio*.
- Método *getVoltages*: Finalmente, este método devuelve el valor de la última medida de los sensores realizada.
- *Callback* asociado a la recepción de información del microcontrolador *ReadSerialData*: Este método es el encargado de recibir la información enviada por el bus de datos desde el Arduino Due únicamente cuando hay un dato en el canal de entrada. Asimismo, en caso de que la línea de texto enviada contenga las mediciones de los sensores, se encarga de notificarlo al *callback* asociado a la medición, cuyo funcionamiento se verá a continuación.
- Método *Measure* y callback asociado (*CallbackMeasurement*): Este método es el que permite obtener los valores de medida de los sensores a través del INA3221. El primero únicamente envía la señal de medición al microcontrolador mediante una línea de texto. De esta forma, cuando el microcontrolador "conteste" con el envío de los datos medidos, será recibido por el método *ReadSerialData*, el cual enviará la notificación de que se ha encontrado una nueva medida (atributo *NewMeasure*). Dicha notificación activa el *callback* correspondiente a la medida, que realiza las separaciones y trasformaciones pertinentes de la línea de texto recibida y guarda los valores medidos en el atributo *voltages*.
- *ResetVoltagesPositions*: Este método se emplea para eliminar todos los datos guardados en los atributos *voltages* y *positions*.

Posteriormente, se añadieron otros tres grupos de métodos, uno para el modelado por curvatura constante, otro para el intento de calibración de los sensores y un tercero para el control del segmento. Todos ellos se introducirán y explicarán posteriormente.

A su vez, también se han añadido las funciones correspondientes en el código implementado en la IDE de Arduino. A continuación de detalla dicho funcionamiento:

6.2.1. Código Arduino

La medición de los sensores requería la creación de un código en Arduino actualizado. Es por ello que se empleó el código ya realizado por Adrián Rieker [1] para el control

del banco de actuación del robot y se añadieron los cambios respectivos para la correcta lectura de los datos enviados a través del bus I2C del microcontrolador.

Estos cambios, como todo código realizado en la IDE de Arduino, se dividen en dos secciones: el *setup* o inicialización y el *loop* o bucle de repetición.

Cambios en el *setup*

Para abordar de la manera más efectiva la obtención de los datos enviados por el dispositivo de medida INA3221 se ha empleado la librería de Arduino *INA3221*. En ella se recogen funciones que permiten inicializar los valores internos del hardware, así como métodos para obtener los datos de tensión, corriente y potencia medidas por el dispositivo gracias a la resistencia *shunt*, enviadas al microcontrolador a través del bus I2C.

Asimismo, para el empleo, inicialización y configuración de las comunicaciones del bus I2C, es necesario emplear la librería *Wire* de Arduino, que contiene todo lo necesario para usar este protocolo de forma sencilla y accesible por el usuario.

De esta forma, y gracias al empleo de estas librerías, se debe instanciar un objeto de la clase INA3221 mediante la selección de su dirección I2C. Es importante señalar que esta debe coincidir con la configurada mediante la soldadura de los *jumpers* del hardware físico.

Posteriormente, se debe inicializar la comunicación I2C mediante el uso del método *begin* interno en la librería.

Cambios en el *loop*

Respecto a los cambios realizados para poder recibir y comunicar los datos de medición al ordenador, se empleó el método *getVoltage(channel)*, donde se introduce como parámetro el canal a medir y se obtiene como salida la medida de tensión de la carga con una precisión de dos decimales.

Este método se implementó siguiendo la selección de operaciones a realizar diseñada por Adrián Rieker [1] en el trabajo anterior al aquí expuesto. De esta forma, mediante el envío de la respectiva línea de texto, que constaba únicamente de un caracter “M”, y el *switch* interno de selección del modo de operación del programa, se procedía a la medida de los tres canales del INA3221. Tras ello, el microcontrolador enviaba a través del puerto serie una nueva línea de texto con los tres valores medidos junto a un carácter “M” al comienzo.

6.3. Planteamiento inicial del problema - Control analítico

Una vez introducidos los cambios en la arquitectura software de PAUL, se continuará con una descripción de todas las fases de planteamiento del problema hasta alcanzar la solución final implementada.

La primera idea para encontrar una resolución al problema de control del robot fue la de dividir el esquema de control en el mayor número posible de bloques. La esperanza que se tenía con ello era reducir la dificultad y complejidad del problema de modelado y control del *Soft Robot*, pues aunque hubiera un gran número de bloques y subsistemas, cada uno de ellos sería más fácil de modelar por ser más sencillos y de menor tamaño que bloques que modelen sistemas enteros.

Respecto al método de control, inicialmente se quería estudiar la viabilidad de emplear un modelo analítico basado en curvatura constante. Como ya se vio en la sección 2.2.2 del estado del arte este método es ampliamente empleado en la actualidad para el control de *Soft Robots* con un comportamiento continuo y que están compuestos de múltiples segmentos, como es el caso del robot aquí expuesto.

La aplicabilidad de este método de control se basaba en la idea de segmentación del problema descrita en el apartado anterior, pues era necesario para su aplicación encontrar un método de conversión de la medida de los sensores (tensión) a tres longitudes, posteriormente introducidas en las ecuaciones del MCI (modelo cinemático inverso) proporcionadas por el PCC. Recordar que este modelo relaciona un punto en el espacio con tres longitudes arbitrarias que describan el comportamiento total de la forma del robot.

Tras la conversión de los datos medidos a las longitudes empleadas por las ecuaciones del PCC para hallar el punto del espacio en el que se encuentra el extremo del robot, surgían dos posibilidades:

- Emplear el MCI para hallar el punto real en el que se encuentra el extremo del robot a partir de dichas longitudes, con lo que la realimentación en cadena cerrada daría como error la diferencia entre el punto real y el punto objetivo introducido. Esta técnica implicaría una elevada complejidad en el diseño del regulador, cuya función sería la de transformar dichos errores en los estímulos de actuación correctos dados por los tiempos de apertura de las válvulas, así como el desarrollo de las ecuaciones del MCI para este caso, pues las ecuaciones del PCC para aplicaciones 3D está aún en desarrollo en muchos casos y actualmente se encuentran solo bien definidos los MCD (modelo cinemático directo). Este esquema de control se puede apreciar en la figura 6.1.

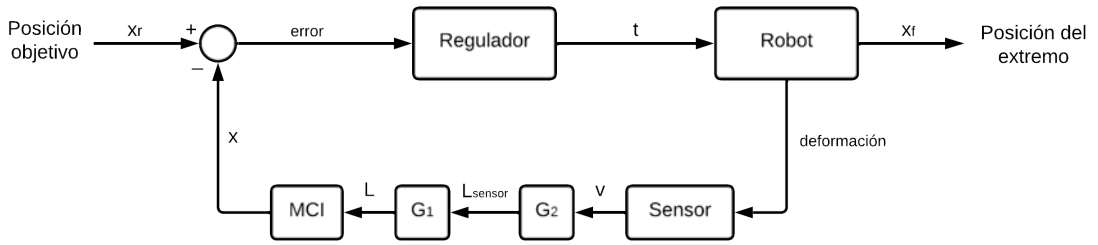


Figura 6.1 Esquema de control empleando MCI. *Fuente: elaboración propia*

- Cerrar el bucle de control directamente con las longitudes calculadas y emplear el MCD para transformar el punto objetivo en longitudes objetivo, siendo el error igual a la diferencia entre las longitudes reales y objetivos. Esta técnica suprime la necesidad de cálculo de MCI pero sigue añadiendo una gran complejidad al diseño del regulador. Este esquema de control se puede apreciar en la figura 6.2.

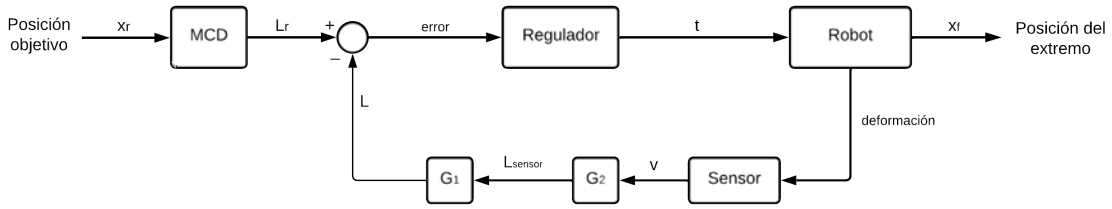


Figura 6.2 Esquema de control empleando MCD. *Fuente: elaboración propia*

Para llevar a cabo la implementación de las ecuaciones del modelado por curvatura continua (PCC), se crearon dos nuevos métodos para la clase *Robot*: los métodos*MCD* y *MCI*. Como se puede intuir por su nombre, el primero contenía las ecuaciones del modelo cinemático directo, al que se pasaban las tres longitudes características para el modelado mediante PCC y, tras las operaciones pertinentes, devolvía la matriz de transformación homogénea al que correspondía dicho conjunto de longitudes junto a los parámetros empleados para ello (curvatura, longitud de arco y rotación). El segundo, por su parte, contenía las ecuaciones del modelo cinemático inverso, cuyo funcionamiento consistía en introducir, entre otros parámetros de diseño, el punto dado por sus tres coordenadas respecto al sistema de la base del robot (almacenado en el atributo *base*) y, tras el cálculo de sus ecuaciones, devolvía como salidas un vector con las tres longitudes características. Asimismo, este segundo método también proporcionaba los parámetros empleados en el cálculo, al igual que para el método anterior.

A pesar de ello, esta resolución presentaba el problema de diseño del regulador, como ya se ha comentado con anterioridad. Este inconveniente se intentó solucionar añadiendo un bloque adicional consistente en una tabla que tuviera en su interior relaciones entre puntos

del espacio y tiempos de hinchado. Esta tabla se crearía de forma dinámica cada vez que se iniciara el robot mediante un método de calibración. Este esquema de control se puede apreciar en la figura 6.3. El inconveniente que hizo descartar este método posteriormente fue que no se añadía nada nuevo al control en cadena abierta ya diseñado, pues como en este solo se podrían ir a determinados puntos del espacio capturados, presentando un gran error aquellos puntos que no se encontraran definidos en dicha tabla. Además, hacía necesarias las ecuaciones del modelo cinemático inverso provenientes de la aproximación por curvatura constante (PCC).

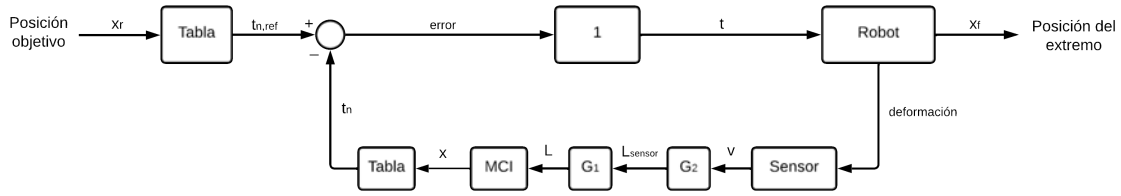


Figura 6.3 Esquema de control empleando tabla dinámica y MCI. *Fuente: elaboración propia*

De cualquier manera, todas estas ideas pasaban por realizar la conversión de tensión a longitud de generatriz del cilindro al que se puede aproximar PAUL (para cada uno de los tres sensores) mediante dos pasos intermedios: de tensión a longitud del sensor y de longitud del sensor a longitud de la generatriz.

El primero de los bloques convertiría tensiones a longitudes del sensor gracias a las curvas de caracterización de cada uno de los sensores, aproximadas mediante una función polinómica con la función *polyfit* disponible en MATLAB. Así, seleccionando el grado adecuado del polinomio, se podría transformar de forma sencilla una tensión en una longitud. A pesar de ello, existía una gran problemática, pues sería necesario caracterizar cada uno de los sensores a utilizar. El motivo de esto es que los sensores presentaban distintas curvas tensión-elongación, en algunos casos muy distintas entre sí (ver sección 3.3.3).

El segundo de los bloques debía transformar las longitudes de los sensores en las longitudes de las respectivas generatrices del robot. Esto supuso un gran problema, pues el diseño de las cavidades en las que iban introducidos los sensores no abarcaban toda la longitud del segmento, además de contar con una curva en la parte baja que permitía poder conectar ambos extremos de los sensores por la parte superior. Dado que la distancia entre el vértice de esta curva y la parte inferior del segmento variara según el grado de deformación que presentara el segmento, se hacía muy compleja la obtención de la relación entre ambas longitudes.

Como primera aproximación a este último problema se intentó obtener dicha relación mediante la medida de la longitud de la generatriz en distintas imágenes tomadas por las

cámaras, contando además con los datos de medida. Sin embargo, al requerir tanto tiempo este proceso y además contar con muchos otros problemas para abordar de esta forma el control del robot, finalmente se abandonó esta primera idea, pues se hizo evidente a la vista de todos los problemas descritos la inviabilidad del empleo de un método analítico basado en curvatura constante.

6.4. Segundo intento de resolución del problema - Control experimental

Tras el intento fallido de abordar el problema mediante el uso de ecuaciones analíticas, se llegó a la conclusión que sería necesario emplear un modelo de caja negra en el cual, mediante el uso de redes neuronales, se consiguiera que los puntos introducidos como entrada al sistema fueran convertidos a tensiones objetivo. De esta forma, se podría cerrar el bucle de control resultando como error la diferencia entre las tensiones medidas y las tensiones objetivo. De esta forma, mediante el uso de un regulador de tipo PID, se podría calcular el tiempo de apertura necesario de las válvulas para alcanzar la posición objetivo introducida inicialmente. En la figura 6.4 se puede apreciar el esquema de control descrito en este párrafo.

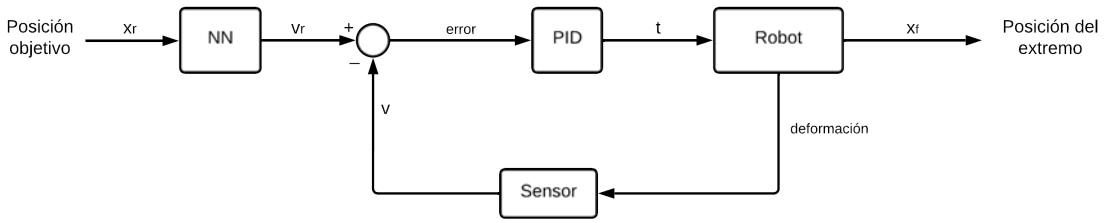


Figura 6.4 Esquema de control empleando una red neuronal y un regulador PID. *Fuente: elaboración propia*

Como resumen previo a la explicación, este nuevo enfoque requeriría extraer suficientes datos de las relaciones entre posición y tensión. Todos estos datos serían posteriormente almacenados en los denominados *datasets*, procesados y preparados para entrenar dicha red. Tras esto, se comprobaría manualmente (observando la diferencia existente entre las predicciones de la red y los valores reales (*targets*)) y mediante los procedimientos que MATLAB otorga, como el error cuadrático medio (MSE) o *performance*, como se denomina aquí; la efectividad de la red neuronal de predecir los valores dentro del rango de trabajo. Finalmente, se pondría a prueba su eficacia de manera experimental.

A continuación se explicará en mayor profundidad el procedimiento de generación de los *datasets* empleado, así como el diseño y el proceso de entrenamiento de la red neuronal que constituiría el bloque restante del esquema de control. Seguidamente, se profundizará

en la problemática surgida durante el diseño del regulador PID y el posterior rechazo de esta segunda idea de metodología de control.

6.4.1. Obtención de la red neuronal a emplear

El primer punto a resolver era la selección del tipo de red neuronal. Los tres principales tipos actualmente son las redes neuronales prealimentadas o *feedforward neural networks* (*FFN*), las redes neuronales convolucionales o *CNN* y las redes neuronales recurrentes o *RNN*. Las aplicaciones de cada una de ellas son muy diversas, por lo que era necesario fijar las características de las relaciones entre los datos de entrada y salida para decidir con mayor criterio el tipo a usar. Estas características son:

- La relación entre las posiciones y los voltajes medidos es **no lineal**; criterio basado en la forma de la curva que describe el comportamiento de los sensores (capítulo 3).
- La relación entre las posiciones y los voltajes medidos es **independiente del tiempo**, pues solo dependerá del grado de deformación que presente el segmento.
- La relación entre las posiciones y los voltajes medidos **no requiere del procesamiento de imágenes**, más allá de la captura de posición ya desarrollada. Esta característica puede parecer una obviedad, pero es necesario destacarla para la correcta selección del tipo de red a emplear.

Según estas características de los datos a predecir, el tipo de red neuronal que mejor se adapta a todas ellas es la prealimentada (*feedfoward*). Esto es debido a que estas redes presenta una mayor versatilidad que los otros dos tipos, pudiendo resolver casi cualquier problema con una configuración adecuada de la red, como ya se vio en el capítulo 4. A su vez, las aplicaciones de los otros dos tipos son muy diferentes a los presentados en este problema:

- Las redes neuronales convolucionales (CNN) se emplean principalmente en el entrenamiento con imágenes mediante visión con computador, pues procesan datos en forma de cuadrícula o matriz, como los píxeles de una foto.
- Las redes neuronales recurrentes (RNN) se emplean sobre todo en aplicaciones en las que el orden o el tiempo en la toma de los datos tiene relevancia. Aún así, dada la posibilidad que ofrecen algunas redes derivadas de este tipo, se empleará más adelante para la resolución de un problema similar una red de tipo LSTM, también explicada en el capítulo 4.

Todo ello llevó a la elección de una *feedforward network* como la red a emplear.

Datos a capturar

El siguiente paso a tener en cuenta es qué datos se van a capturar y cómo se van a almacenar.

Para resolver a estas preguntas, es necesario fijarse en las entradas y las salidas de la red. Es por ello que los datos más importantes serán tanto las posiciones como las tensiones. Las posiciones se capturarán de manera inmediata (el valor obtenido por el software de visión será directamente el almacenado), mientras que para las tensiones surgían dos posibilidades:

1. **Captura y almacenamiento inmediato de los datos**
   De esta forma es más sencillo su procesamiento, pero implica una calibración por cada segmento nuevo.

2. **Captura y almacenamiento relativo**
   El objetivo de esto es poder emplear la red entrenada para otros segmentos, pues la tensión no sería absoluta y dependería del porcentaje sobre la tensión máxima y mínima que midiera el INA3221.

   El procedimiento consistiría en calibrar los sensores hallando las tensiones máxima y mínima, que se creía que serían la de reposo y la de máximo hinchado de esa cámara, respectivamente. Con ello, se podría posteriormente calcular por una regla de tres el porcentaje sobre la tensión máxima que presentaba dicho sensor.

   Para ello, se dispusieron 2 nuevos métodos: *CalibrateSensor* y *Calibrate*. El primero capturaba la medida de los sensores en estado de reposo, considerada la máxima medida de tensión posible, y posteriormente hinchaba la vejiga correspondiente al sensor enviado como parámetro para recoger el mínimo de tensión medible. Posteriormente, se almacenaban estos valores en la columna respectiva de la matriz del atributo *max_ min*. Así, para realizarlo con un segmento entero, se dispuso del segundo método, que empleaba el primero con tres iteraciones distintas para cada uno de los sensores del segmento.

El problema del segundo procedimiento, que es a priori el más beneficioso, fue la dificultad presentada en cuanto a la obtención de las tensiones máxima y mínima de cada sensor, pues había un acoplamientos entre el hinchado de las cámaras sobre las que no se encontraba dicho segmento. Añadido a esto, la deformación de cada segmento se observó que era única y dependiente del proceso de fabricación. Todo ello llevó a decidir emplear el primer método, a pesar de que suponga un nuevo entrenamiento por cada nuevo segmento.

Adicionalmente, se capturaron los datos de tiempos de hinchado y de tiempo residual (medida indirecta del volumen de aire en el interior del segmento) para cada nuevo conjunto de datos obtenido. Esto se hizo con la intención de poder filtrar y comparar los

resultados en el supuesto de que ocurriera algún error durante el proceso de extracción de datos.

Destacar a su vez que todos los datos obtenidos se almacenarían en matrices de N filas y 3 columnas, siendo N el número de muestras a extraer. Las 3 columnas se explican dado que siempre se obtienen 3 datos por cada nuevo conjunto de datos obtenido. Esto es así ya que las posiciones son 3 (x, y y z) y para el resto de datos, se almacenan los valores de las tres vejigas que componen el segmento.

Procedimiento de obtención de los *datasets*

A continuación se explica el procedimiento de obtención de los datos que luego se usarán para el entrenamiento de la red neuronal. Este proceso ha sido automatizado a través de un *script* de MATLAB.

Este procedimiento de extracción de datos consistirá en una serie de operaciones a realizar en cada iteración del proceso. Como parte de la operación inicial, previa a la obtención de muestras, es necesario instancia un objeto de la clase Robot (que para este caso se ha llamado "R"). Posteriormente, se iniciará la comunicación UART con el microcontrolador empleando para ello el método *Connect*, tras lo que se calibrarán las cámaras *webcam* de Logitech, previamente posicionadas en ubicaciones que permitan la captura tanto frontal como de perfil de la baliza del extremo (preferiblemente en una ángulo próximo a los 90^{o}), mediante el método *CalibrateCameras* y el damero de calibración. Por último, se debe capturar la posición inicial mediante el método *CapturePosition* con el fin de posicionar el extremo del robot (tanto el real como la esfera verde de la baliza) con respecto a la base (parte superior del segmento, coincidente en este caso con la parte superior del utillaje). Todos estos pasos provienen y han sido descritos en el Trabajo Fin de Grado de Adrián Rieker [1].

Antes de proceder a nombrar los pasos a seguir, es importante destacar que el robot presenta una redundancia en las posiciones; es decir, se puede llegar a los mismos puntos si se actúan dos cámaras simultáneamente o las tres a la vez. Esto se ve claramente en el caso de la posición inicial, donde se puede llegar a ella mediante el desinflado completo de todas las cámaras o hinchando con sus respectivos tiempos todas las cámaras a la vez. Lo único que se logra en un ligero alargamiento del segmento y el aumento de la posibilidad de provocar una fuga en la pared externa. Es por ello que siempre se mantendrá una vejiga desinflada completamente, pues solo es necesario para alcanzar todos los puntos dos vejigas. Esta decisión se tomará de forma aleatoria en cada iteración del proceso.

Tras completar correctamente la inicialización del puerto serie y la calibración de las cámaras, los pasos a seguir, que se repetirán en bucle tantas veces como datos se quieran extraer, son los siguientes:

1. Generación de forma aleatoria los valores de tiempos a aplicar mediante el uso de la

función *randi* y su correcta configuración. Estos tiempos pueden ser tanto positivos (inflado) como negativos (desinflado).

2. Comprobación de si la acumulación de los tiempos de hinchado (incluyendo el último generado) excede el máximo previamente definido (en este caso de 900ms). En caso afirmativo, no se actuarán aquellas cámaras (tiempos de hinchado nulos) que cumplan esta condición.

3. Selección de la vejiga que se queda sin hinchar (desinflado en caso de contener aire) en el caso de que tras los dos pasos anteriores no hubiera ninguna vejiga sin aire tras la actuación. Esta selección se realiza de forma aleatoria, generando un número aleatorio de 1 a 3. Al tener registro del aire en su interior (mediante la suma de los tiempos de actuación) se puede saber cuánto es necesario deshinchar dicha cámara.

4. Comprobación de "nuevo punto alcanzado". Esto se refiere a que no se repiten las combinaciones de tiempos almacenados (aire contenido en el interior de las vejigas) en cada una de las cámaras. En caso de ser así, se regresaría al paso número 1 y se repetirían los pasos 1-4 hasta encontrar una combinación nueva.

5. Actuación del segmento con los tiempos determinados en los pasos 1-4.

6. Captura de una imagen con cada una de las cámaras, de donde, mediante el método *CapturePosition*, se obtendrá la posición real del extremo del segmento, así como su orientación (que no será empleada en este caso, pues solo se disponen de tres grados de libertad). También se realizará la medición de los sensores a través del método *Measure*, su *callback* asociado y el atributo *voltages*. Tras ello, se repetirá el proceso hasta alcanzar el número de muestras requerido.

Este procedimiento se puede ver con mayor detalle en la figura 6.5.

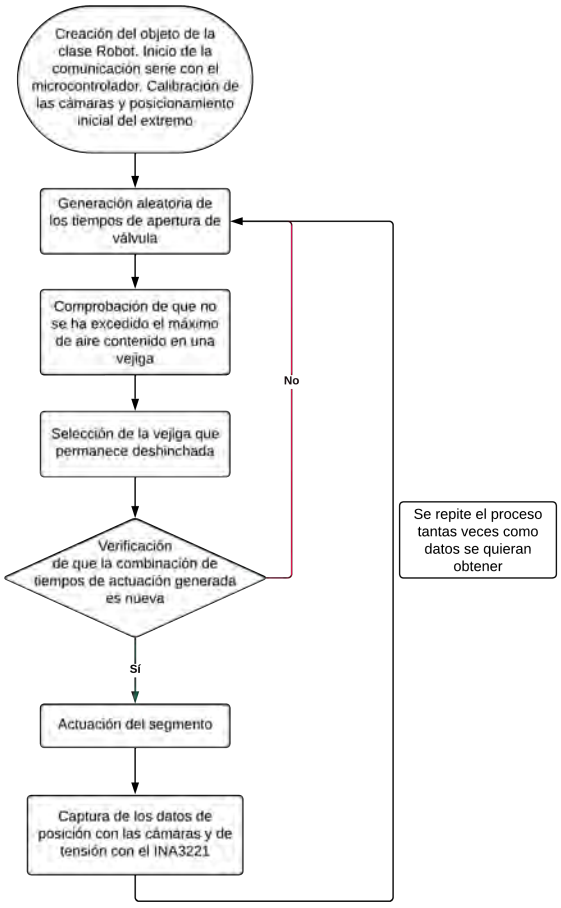


Figura 6.5 Diagrama del proceso de obtención de datos. *Fuente: elaboración propia*

Cabe mencionar que no se consideró necesario un desinflado intermedio entre las distintas iteraciones, pues el sensor, como se vio en el capítulo 3, no presentaba histéresis tras la liberación del sensor y solo requería de unos segundos para la estabilización de la medida. Esto posteriormente se verá que fue un grave error que supuso una gran pérdida de tiempo, pues, como ya se comentó en la sección 5.3 y se explicará más adelante, los segmentos presentan una asimetría entre los tiempos de hinchado y vaciado que hacía necesario un desinflado intermedio.

Por último, es necesario determinar el número de muestras a extraer que permita obtener una red neuronal bien entrenada con un valor bajo de MSE. Para ello, se estima el número de puntos diferentes que puede tener el campo de trabajo. Teniendo en cuenta que la actuación necesaria para que el robot experimente un cambio apreciable por el ojo humano en su forma es de alrededor de 70ms, se generarán los tiempos de actuación en saltos de 50 en 50 milisegundos. De esta forma se reduce el número de puntos que configuran el espacio de trabajo. Si se hace esto, teniendo en cuenta que la actuación máxima es de 900ms y que siempre hay una vejiga que se queda desinflada, el número

máximo de combinaciones de tiempos almacenados por vejiga (aire contenido en cada una) es de $3 \cdot (\frac{900}{50} + 1)^2 = 1083$. Esto es un valor alcanzable para el robot, pues, tras las modificaciones realizadas (capítulo 5), la probabilidad de que se produzca una fuga durante el proceso de obtención de datos es baja.

Diseño y entrenamiento de la red neuronal

En este apartado se explicarán las decisiones tomadas en lo referente al diseño y proceso de entrenamiento de la red neuronal empleada.

Una vez seleccionado el tipo de red a emplear (*feedforward*) y teniendo diseñado el proceso de obtención de los datos de entrenamiento de la red, faltaría únicamente configurar correctamente la red neuronal a través de sus hiperparámetros para conseguir una resolución óptima del problema para la que se usará. Los hiperparámetros que se modificarán serán el número de capas ocultas, el número de neuronas por capa, la función de activación de cada capa, el algoritmo de retropropagación de la red y los pesos en la división de los datos (entrenamiento, validación y prueba). Para una mayor información acerca de todas estas características de las redes neuronales prealimentadas, consultar la sección 4.2 del capítulo 4: Herramientas utilizadas.

1. **Número de capas y número de neruonas por capa**
   Los dos primeros hiperparámetros a definir son probablemente los más importantes, pues sin ellos no se podría definir la red neuronal.

   La elección del número de capas se ha basado en el *teorema de aproximación universal* para redes neuronales. Este teorema afirma que "una red neuronal prealimentada multicapa estándar con una sola capa oculta que contenga un número finito de neuronas ocultas son capaces de aproximar cualquier función continua con el uso de funciones de activación arbitrarias" [39]. En otras palabras, cualquier problema que incluya la conversión (*mapeado*) de un intervalo continuo de números reales a otro intervalo continuo de números reales puede ser aproximado por una red neuronal prealimentada con una sola capa oculta y un número suficiente de neuronas en dicha capa, independientemente de la función de activación empleada. La ecuación 6.1 resume este teorema, siendo $\varphi(.)$ una función de activación arbitraria. $A_n f(.)$ es una aproximación de $f(.)$, por lo que $\|f - A_n f\| < \varepsilon$, donde ε es una valor de error arbitrario.

$$(A_n f)(x_1, ..., x_m) = \sum_{i=1}^{n} w_i \, \varphi \left(\sum_{j=1}^{m} a_{ij} x_j + b_i \right) \tag{6.1}$$

   Este teorema facilita enormemente la elección del número de capas, pues la mejor opción será contar únicamente con una capa oculta, que junto a las capas de entrada

y salida hacen 3 capas en total. El contar únicamente con una capa oculta además reduce en gran medida el tiempo de entrenamiento de la red y evita el sobreajuste, pues no se producirán dependencias entre neuronas al solo contar con una capa oculta y se reducirán las combinaciones entre los pesos y sesgos de neuronas.

La selección del número de neuronas en las capas de entrada y salida serán tres en cada una, pues es un requisito de la red (una neurona por cada valor escalar en la entrada y en la salida). En cuanto al número de neuronas de la capa oculta, este se ajustará mediante prueba y error hasta dar con un valor de MSE suficientemente bajo para el problema a resolver, ya que la curva que muestra la relación entre MSE y número de neuronas empleado con una sola capa oculta tiende a un valor e incluso aumenta ligeramente (por el sobreajuste) a partir de cierto número de neuronas.

2. **Función de activación**
   Como se ha visto en el *teorema de aproximación universal*, la función de activación escogida es independiente de la posibilidad de conseguir una buena aproximación a la función de realción entre entradas y salidas de la red. Aún así, las funciones de activación sí que juegan un papel fundamental en la velocidad con la que se alcanza dicha aproximación, por lo que se intentará optimizar el tiempo de entrenamiento con esta elección.

   Así, para la capa de salida se usará la función ReLU, pues los valores de salida son siempre positivos, lo que aumenta la velocidad en caso de que aparezcan valores negativos como entrada a esta capa. Para la capa de entrada y la capa oculta se harán dos pruebas utilizando las funciones sigmoides tangencial hiperbólica y logística, pues ambas pueden resolver el problema de forma eficiente.

3. **Algoritmo de retropropagación**
   El algoritmo de retropropagación no suele jugar un papel muy importante en la optimización del entrenamiento de la red, más allá de reducir o aumentar ligeramente el tiempo de entrenamiento y utilizar más o menos memoria durante el proceso. Dado que la memoria en este caso no es un problema, pues el número de datos de entrenamiento es bajo y el problema es puramente numérico, nos centraremos en la eficiencia. Según el *centro de ayuda* de Mathworks [40], el algoritmo de retropropagación más rápido para este tipo de problemas (además de ser de los que más memoria usan) es el de Levenberg-Marquardt, por lo que será este el que se empleará.

4. **Pesos en la división de los datos**
   Al contar con pocos datos, se utilizará una división basada en el máximo número de datos para el entrenamiento. Así, los datos se dividirán en un 80 % para entrenamiento, un 10 % para validación y un 10 % para test. Además, se dividirán de forma

aleatoria utilizando la función *dividerand* de MATLAB.

Tras realizar todas estas elecciones, el siguiente paso es realizar distintas pruebas de configuración de red siguiendo estas características y entrenarlas usando los *datasets* adquiridos.

Para ello se emplearán la función *feedforwardnet(N,trainFcn)*, donde N es el número de neuronas de la capa oculta (si se quieren varias capas ocultas se debe especificar con un vector línea: $[N_1, N_2, ..., N_m]$, donde N_k es el número de neuronas de la capa k y m es el número de capas ocultas). La *trainFcn* sería el algoritmo de retropropagación a emplear. En este caso la sentencia sería *net = feedfrowardnet(x)*, pues el algoritmo de Levenberg-Marquardt es el que se encuentra por defecto. Posteriormente, se hará uso de la función *train(net,inputs,tragets)*. En este caso, al ser feedfroward, la sentencia a emplear sería: *net = train(net,pos,vol)*, donde *pos* es la matriz donde se encuntran las posiciones para cada valor de tensión en *vol*. Importante destacar que estos datos deben almacenarse en matrices de 3 filas y n columnas, siendo n es número de muestras tomadas.

Destacar además que la validación se realizará de forma manual, empleando para ello la predicción realizada por la red para valores tanto usados en el entrenamiento como arbitrarios (utilizando un pequeño número de valores del *dataset* no empleados en el entrenamiento de la red. Para la comprobación de la predicción se empleará la función *perform*, que calcula el MSE. También se observarán las diferencias existentes a simple vista.

6.4.2. Diseño del regulador PID

La idea inicial era diseñar un regulador de tipo PID, pues este habitualmente es capaz de solventar casi cualquier problema en el mundo de la robótica, además de ser ampliamente empleado en esta industria.

Para ello, se iniciaron las pruebas diseñando un regulador de tipo P (el más sencillo) para controlar únicamente una vejiga. De esta forma, mediante la introducción de un valor de tensión objetivo dentro del rango de medición se vería si se conseguía alcanzar el régimen permanente. Se limitó la acción por iteración del bucle de control a 150ms. Con ello se pretendía evitar un posible descontrol en la acción que diera lugar a la rotura del segmento. Así, utilizando la ecuación del regulador P (ecuación 6.2) con una constante de proporcionalidad $K_p = 500$, ajustada manualmente con ensayos, se consiguió una excelente respuesta, pues se alcanzaba con rapidez y suavidad el valor objetivo con un error cercano a 0. Se concluyó entonces que solo sería necesario un regulador P.

$$acción = K_p \cdot error \tag{6.2}$$

El principal inconveniente surgió al intentar extrapolar este regulador al segmento

entero, pues había pequeños acoplamientos entre los sensores al inflar las dos cámaras opuestas, lo que convirtió esta sencilla tarea en un obstáculo de gran complejidad.

Se intentó abordar mediante el uso de una matriz de constantes de proporcionalidad, según los principios de los reguladores PID multivariable. Sin embargo, al no contar con un modelo del robot, se hizo imposible determinar los 9 valores de la matriz (3 por sensor) manualmente mediante ensayos de prueba y error. Esta fue una de las razones que llevó a descartar nuevamente la idea de control mediante un regulador PID.

6.4.3. Conclusiones

Las conclusiones finales extraídas de los intentos de bordar el problema de control siguiendo el esquema de control de esta segunda idea pueden resumirse en dos problemas insalvables que llevaron a la adopción de una tercera idea, que se convertiría posteriormente en la solución al problema: la dificultad de obtención de los *datasets* y la inviabilidad del diseño de un regulador PID.

Respecto al primer problema, esta dificultad provino principalmente y como se descubrió más adelante de la asimetría en el inflado y desinflado de las 3 cámaras que componen cada segmento. Este hecho se explica siguiendo los principios de la Mecánica de Fluidos, donde el aire contenido en el interior de las vejigas irá disminuyendo su presión conforme se acerca a la presión atmosférica del exterior, lo que reduce el caudal de aire de salida. Sin embargo, la entrada de aire al interior de la vejiga, al encontrarse a una presión constante de 1 bar por encima de la presión atmosférica, tendrá un caudal prácticamente constante. Es por ello que, aunque se intente aproximar mediante un multiplicador, como ya se hizo en el trabajo anterior [1], siempre existirá un error, dado que la salida de aire es un proceso fuertemente no lineal.

Todo esto explica las incoherencias surgidas durante la extracción de datos y los elevados errores de predicción en la red (próximos al 30 %), pues los datos estaban falseados, ya que siempre quedaba aire residual en el interior de las cámaras aún cuando debían estar completamente desinfladas y nunca se correspondían a las medidas reales de volumen de aire en el interior. Esto además aumentó la cantidad de fugas y pinchazos en el segmento, lo que retrasó la investigación debido a la necesidad de fabricación de nuevos segmentos.

Es por ello que para el planteamiento final del problema se decidió desinflar completamente las cámaras entre cada iteración del proceso de obtención de los *datasets*, pues solo se podía controlar el hinchado del segmento.

En lo referente al segundo problema, se tomó la decisión de no emplear un regulador PID. Esta estuvo motivada por la imposibilidad de obtención de forma experimental los valores de las constantes de proporcionalidad que lograban generar de forma correcta las señales de actuación.

La decisión tomada en lo respectivo al regulador hacía necesaria la utilización de una

nueva red neuronal. Es por ello que, como se explicará en el siguiente punto, se procedió a diseñar un nuevo diagrama de control cuyo cierre del bucle de control diera como error directamente la diferencia entre el tiempo de actuación acumulado en cada vejiga, como medida indirecta del volumen de aire en su interior, y el tiempo objetivo a lograr en cada cámara para alcanzar el punto requerido.

Aún así, esta idea sentó las bases y el camino a seguir para alcanzar la resolución final del problema, pues, como se verá más adelante, se decidió conservar gran parte del procedimiento de obtención de datos y la estructura de la red neuronal aquí expuesta.

6.5. Resolución y planteamiento final del problema

En este apartado se explicarán todas aquellas decisiones en cuanto al diseño del sistema de control que llevaron a la correcta resolución del problema del control de uno de los segmentos que componen el robot.

6.5.1. Esquema de control final

Tras las conclusiones extraídas de los anteriores intentos de resolución del problema de control, se hizo evidente la necesidad de emplear 2 redes neuronales que, en primer lugar, evitasen la necesidad de uso de un regulador PID tradicional y, en segundo lugar, solventaran el problema de la conversión de las tensiones medidas a un parámetro más manejable. Se decidió así que el lazo del bucle cerrado se cerraría con los tiempos de actuación. De esta forma, el error que resultara de la diferencia entre el tiempo de actuación acumulado y el tiempo objetivo a lograr en cada vejiga sería directamente la señal de actuación, por lo que el regulador sería una constante de valor "1".

Para lograr este objetivo, se emplearían dos redes neuronales. La primera convertiría los datos de posición objetivo en tiempos de hinchados objetivo. La segunda transformaría los datos de tensión medidos a través del INA3221 en datos de tiempos de hinchado acumulados. Así, el diagrama de bloques resultante se puede apreciar en la figura 6.6.

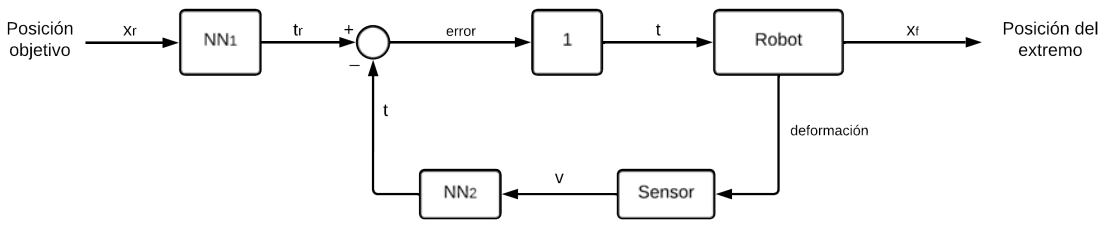


Figura 6.6 Diagrama de bloques empleado para el control del segmento. *Fuente: elaboración propia*

La ventaja que presenta este método de control además es la posibilidad de extrapo-

lación a un número arbitrario de segmentos, pues con el suficiente número de datos de entrenamiento de las redes y su correcta configuración se puede lograr abordar el problema de control con éxito.

A continuación, se explicará el procedimiento de obtención de ambas redes neuronales.

6.5.2. Obtención de la red neuronal a emplear

Para este último planteamiento se emplearon nuevamente redes neuronales de tipo *feedfroward*, pues como ya se vio en el apartado 6.4.1, es la que mejor se adaptaba a las necesidades del proyecto. Aún así, también se exploró una posible solución con una red de tipo LSTM, ya vista en el capítulo 4, que a su vez deriva de las redes neuronales recurrentes (RNN). A pesar de ello, y dado que finalmente se emplearon redes neuronales prealimentadas, este punto se centrará en estas últimas, dejando un pequeño apartado posteriormente de las decisiones que llevaron a no emplear redes LSTM.

Como ya se comentó con anterioridad, el primer paso para obtener una red neuronal bien entrenada es definir los datos a capturar y su procedimiento de obtención.

Datos a capturar

Como en el caso anterior, el punto de partida de entrenamiento de las redes es la selección de las muestras a recoger. Para ello, se empleará un mismo *dataset* en el que se almacenarán los datos obtenidos para el entrenamiento de ambas redes, pues de esta forma se facilita su manejo.

Estos *datasets* recogerán los mismos datos y serán guardados de la misma forma (mediante el uso de matrices) que en el planteamiento descrito en la sección anterior (6.4). Esto es debido a que serán necesarios los datos de tiempos acumulados y de posiciones alcanzadas para una de las redes y los datos de tiempos acumulados y de tensiones medidas para la red restante. Además, se suprime ahora la necesidad de guardar los datos de tiempos de actuación mandados al segmento en cada iteración, pues estos son exactamente iguales a los tiempos acumulados en las vejigas. Esto se debe a que, como se explicará a continuación, se procederá a un desinflado completo de las 3 cámaras que componen el segmento al comienzo de cada iteración del proceso de obtención de los datos.

Procedimiento de obtención de los *datasets*

El procedimiento empleado es muy similar al descrito en la sección 6.4.1, añadiendo el paso ya comentado consistente en desinflar todas las vejigas del segmento al comienzo de cada nueva iteración. En la figura 6.7 se puede apreciar el procedimiento empleado finalmente. Así, con la adición de este nuevo paso, se consigue que los tiempos de hinchado extraídos sean reales, a diferencia de lo ocurrido en el anterior planteamiento.

Las únicas diferencias con respecto al anterior proceso son los valores de tiempos de actuación aleatorios, que ahora son exclusivamente positivos (solo se hinchan las vejigas en cada iteración ya que inicialmente se encuentran deshinchadas), y la comprobación de que no se excede el máximo, que ahora no es necesaria ya que los valores de tiempos generados están siempre contenidos en el rango de 0 a 900ms (máximo).

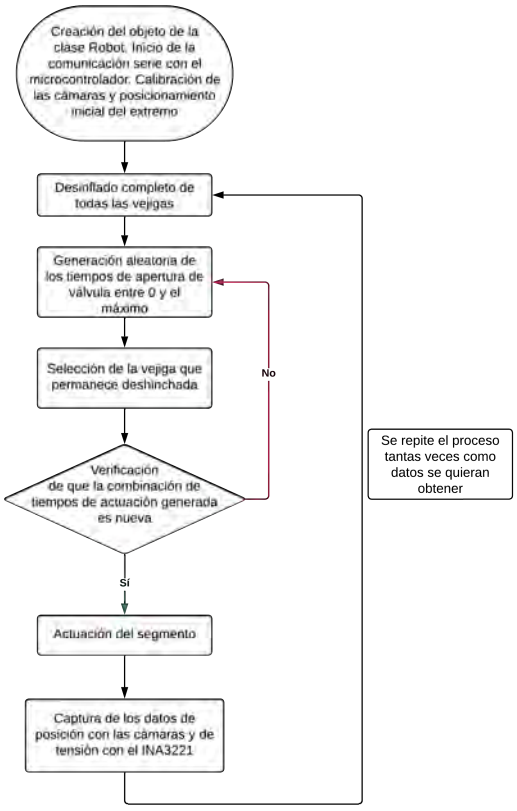


Figura 6.7 Diagrama final del proceso de obtención de datos. *Fuente: elaboración propia*

Destacar además que se extraerán 1050 datos del máximo de 1083 datos provenientes del cálculo realizado anteriormente que completan el campo de trabajo total para saltos de tiempos de actuación de 50ms (pues si los saltos fueran de 1ms en 1ms, el número total de muestras a adquirir sería superior a los 2 millones de datos). Se deja así un margen de 33 iteraciones para evitar "cuelgues" hacia el final del proceso de extracción de datos, pues al tener que repetir el proceso hasta encontrar una combinación nueva no existente entre los datos puede "bloquear" en cierta medida el proceso hasta encontrar por puro azar ese nuevo dato. Este hecho se comprobó en un *script* adicional, sin inicializar ni actuar el robot, y se observó que alrededor de la iteración 1060 se comenzaba a ralentizar este proceso. Por ello se decidió parar la toma de datos en la iteración 1050, pues el contar con 33 datos extras frente a 1050 no supondría una mejora apreciable en las predicciones de la red.

El código empleado para este proceso se puede ver en el código C.2 dentro del Apéndice C.

Diseño y entrenamiento de la red neuronal

Por último, para el diseño de ambas redes neuronales se emplearán los mismos requisitos y conceptos ya vistos en el punto 6.4.1. Es por ello que ambas redes contarán con una sola capa oculta y un número de neuronas que se determinará experimentalmente.

Adicionalmente, debido a los buenos resultados obtenidos en este segundo planteamiento, se utilizarán las funciones de activación logísticas para las capas de entrada y oculta y la función ReLU para la capa de salida para la red neuronal con posiciones como datos de entrada y tiempos como daos de salida. Para la red restante, debido a que tanto sus valores de entrada (tensiones) como de salida (tiempos) son positivos, se empleará la función de activación ReLU en todas sus capas para acelerar la convergencia de la red.

En cuanto al algoritmo de retropropagación, se usará el de Levenberg-Marquardt para ambas redes, pues como ya se vio, es el óptimo para casos similares al aquí expuesto.

Finalmente, la división de los datos se realizará de la misma manera que en el caso anterior para ambas redes: 80 % para entrenamiento, 10 % para validación y 10 % para prueba. También se usará la función *dividerand*.

En cuanto al proceso de entenamiento, destacar que se empleará el mismo procedimiento que en el caso descrito en el punto 6.4.1. La principal y única diferencia serán los datos de entrada y los *targets* empleados en el entrenamiento de las redes.

Para la primera red, los datos de entrada serán las posiciones alcanzadas en cada iteración del proceso, mientras que los datos de salida objetivo serán los tiempos de actuación que permitieron alcanzar dichas posiciones. En cambio, para la segunda red, los datos de entrada serán las tensiones medidas por los sensores tras la actuación (y un pequeño tiempo de estabilización de la medida) con los tiempos generados aleatoriamente en cada iteración del procedimiento de obtención de datos, cuyos valores, al igual que para la primera red neuronal, serán los *targets* a alcanzar como salida. Todos estos datos se procesarán y comprobarán primero manualmente para evitar las incoherencias o los errores surgidos durante la captura de las muestras. Tras ello, se introducirán los datos para el entrenamiento de la red en matrices de 1040 columnas (pues este es el número de datos que se usará para su entrenamiento) y 3 filas, una para cada conjunto. De esta forma, los 10 datos restantes se emplearán en la verificación del rendimiento de la red para valores del campo de trabajo no empleados en su entrenamiento.

6.5.3. Conclusiones y resultados

Como conclusión, comentar que tras una serie de pruebas con distintos valores en el número de neuronas de cada red, se determinó que el número que mejores resultados otorgaba para la primera red era de 275 neuronas en la capa oculta, mientras que para la segunda, este valor fue de 250 neuronas.

Respecto a los resultados que ofrecían las predicciones de ambas redes se consideraron

excelentes. En el caso de la primera de las redes, la que contaba con posiciones como datos de entrada, presentó un error cuadrático medio (MSE) de 11.7 para aquellos valores de entrada que se habían empleado en su entrenamiento y un valor de MSE de 35.8 para valores de entrada distintos a los de entrenamiento.

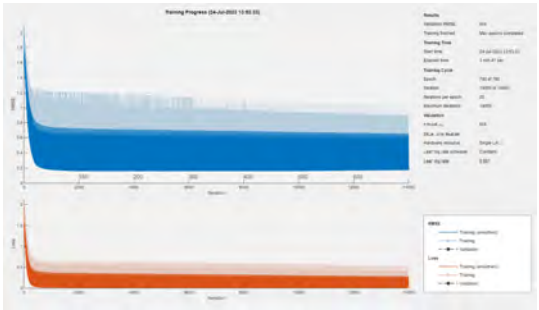
Igualmente, en la segunda red (la que contaba con tensiones como datos de entrada) se obtuvieron valores de MSE de 15.2 para datos utilizados en el entrenamiento, mientras que mostró un MSE de 40.4 para aquellos datos pertenecientes al espacio de trabajo pero no empleados en su entrenamiento.

Estos valores, aunque parecen elevados, no lo son, pues en el contexto en el que se dan implican error relativos muy pequeños. Esto se debe a que estos valores se miden en milisegundos y, además, se debe tener en cuenta que para valores inferiores a los 40ms, el robot sufre una variación en su posición casi inapreciable por el ojo humano.

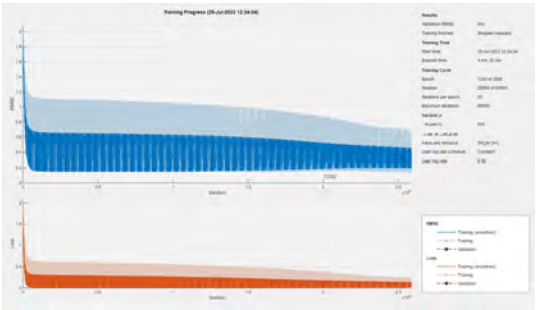
6.5.4. Descarte del uso de redes LSTM

Con el objetivo de mejorar la precisión del sistema en su conjunto se probó adicionalmente la sustitución de las redes neuronales de tipo *feedforward* por otras de tipología LSTM. Este cambio viene motivado por el hecho de que estas redes, como se ha comentado en la sección 4.3, guardan memoria de las entradas pasadas lo que, en un robot con histéresis, podría ayudar a combatirla, como muestran algunos ejemplos de la literatura [41], [42].

Los resultados del entrenamiento de esta red se pueden ver en la figura 6.8. Como se puede ver, los valores de MSE son mejores que los obtenidos para el entrenamiento de las redes *feedforward*. Sin embargo, las necesidades de normalizar los datos y de tener en cuenta el punto anterior hace más compleja su implementación. Esta dificultad no se ve compensada por los mejores resultados de error cuadrático medio, pues no mejoran significativamente los obtenidos para las redes ya entrenadas, especialmente teniendo en cuenta las particularidades en la actuación del robot, donde aperturas de válvula de menos de 20 ms no varían la posición del robot. Es por ello que se descartó su uso.



(a) Resultados del entrenamiento de una red LSTM con posiciones como entrada y tiempos como salida



(b) Resultados del entrenamiento de una red LSTM con tensiones como entrada y tiempos como salida

Figura 6.8 Resultados del entrenamiento de dos redes LSTM. *Fuente: elaboración propia*

6.6. Control del robot

Finalmente, en esta sección se explicará el software desarrollado para llevar a cabo las pruebas del control del robot y su modo de funcionamiento.

6.6.1. Métodos para el movimiento del segmento

Para abordar de la forma más intuitiva y sencilla posible el control del robot, se han implementado cuatro métodos adicionales a la clase *Robot* comentada en la sección 6.2.

A continuación se procede a una descripción detallada de su funcionamiento:

- **Método *NN_ training***
  Esta función tiene dos funciones: automatizar el entrenamiento de las redes, obteniendo sus valores de rendimiento (MSE), y guardarlas una vez entrenadas como atributos del objeto instanciado de la clase. De esta forma, únicamente se necesitan tener ya preparados los *datasets* obtenidos durante el procedimiento de obtención de las muestras para crear ambas redes neuronales empleadas en el sistema de control.

  Se deben introducir en este método para su correcto funcionamiento los datos de posiciones, voltajes y tiempos obtenidos durante el proceso, al igual que un vector con el número de neuronas por capa oculta para cada una de las redes neuronales. De esta forma, tras ambos procesos de entrenamiento, que pueden tardar un tiempo variable de 5 a 10 minutos cada uno, devolverá un vector con los errores cuadráticos medios obtenidos de la comprobación de las predicciones de la red con muestras no empleadas para el entrenamiento de la red. Internamente, solo se selecciona para entrenamiento de la red un 95 % de los datos totales, apartando el 5 % restante para realizar esta comprobación.

- **Método *NN_ creation***
  Este método supone una alternativa al anterior para redes ya entrenadas. Su única finalidad es almacenar las redes entrenadas, pasadas como parámetros de la función, como atributos del objeto derivado de la clase *Robot*.

- **Método *Move***
  El objetivo de este método es el de recoger el bucle de control diseñado y describirlo mediante el código que permita su implementación. Su parámetro es únicamente el punto objetivo al que se quiere mover el robot y sus salidas son el vector "posición final" alcanzada según una captura de imagen al finalizar el proceso de control y el error entre los puntos de llegada y objetivo, medido como la distancia entre ellos.

  El funcionamiento básico del método es:

1. Detectar la correcta introducción de un punto. Esto implica tanto en formato (vector de tres componentes) como en pertenencia al espacio de trabajo.
2. Cálculo mediante la **primera red neuronal** de los tiempos de actuación objetivo. De esta forma se consigue iniciar el bucle de control.
3. Cálculo del error mediante el uso de la **segunda red neuronal**, en la que se introducen las medidas de tensión actuales y se predicen sus tiempos de hinchado equivalentes. Para ello, se emplea el método *Measure* ya descrito en la sección 6.2.
4. Cálculo y envío del estímulo de actuación (tiempos de apertura de válvula). De esta forma, se controla el hinchado en cada iteración, limitándolo a 200ms. Esto se hace como medida de seguridad para evitar roturas y pinchazos por el inflado descontrolado en caso de existir algún error durante el proceso. Destacar a su vez que esta actuación puede ser tanto positiva (inflado) como negativa (desinflado). En este caso, al asimetría entre ambos procesos no supone inconveniente, pues el error se calcula en cada iteración y en caso de no haber desinflado lo suficiente, se volverá a enviar una acción similar de menor valor. Su único inconveniente es aumentar en uno o dos iteraciones más la llegada al punto objetivo.
5. Comprobación de cumplimiento de la toleracia máxima en el error. De esta forma, solo se saldrá del bucle de control (y se considerará que se ha alcanzado el punto objetivo) si existe un error en cualquiera de las tres vejigas menor a la tolerancia impuesta. En este caso, se ha fijado este valor en los 40ms, pues tiempos de actuación menores no generan un cambio apreciable en la posición del extremo del robot. Adicionalmente, se considerará alcanzado el punto objetivo si se alcanza un máximo de iteraciones de 20. Esto se hace por la histéresis presentada en los sensores, que tras acondicionar sus medidas con algunos movimientos, presentan una pequeña desviación o *drift* en su medida cuando la vejiga está completamente desinflada. Es por ello que, en ocasiones, el bucle de control mandaba la orden de deshinchar dicha vejiga aunque ya lo estuviera. Esto se evita con la introducción d eun número de iteraciones máximo.
6. Captura de la posición final y cálculo del error entre la posición de llegada y la introducida como objetivo. Para ello, se calcula el módulo (norma 2) del vector diferencia entre ambos puntos.

- **Método *Move_ debug***

  Este método se creó con la intención de observar y depurar el movimiento del robot en cada iteración del bucle de control realizada. Su funcionamiento es exactamente igual al descrito para el método *Move*, con la únca diferencia que se captura la

posición en cada iteración del proceso, guardándola en un vector de posiciones intermedias, y, posteriormente, se muestra dicho vector de posiciones como salida de la función.

El código empleado para implementar estos métodos se puede ver en el código de definición de la clase Robot en el apéndice C (código C.1).

6.6.2. Procedimiento de uso

Una vez comentados todos los métodos que permiten el control y el movimiento del robot, se detallará seguidamente el procedimiento de empleo:

1. **Carga del *dataset***
   Este primer paso se da exclusivamente en caso de necesidad de generar y entrenar alguna de las dos redes neuronales empleadas por el sistema de control. Como primer paso, se debe cargar en la ventana del espacio de trabajo e *dataset* con el que se van a entrenar las dos redes neuronales. Tras el procedimiento de entrenamiento, se procederá a verificar su rendimiento y, posteriormente, a su guardado. Este proceso se puede automatizar gracias al método *NN_ training*.

2. **Carga de la red neuronal**
   En caso de tener previamente entrenadas las redes o no, el siguiente paso es cargarlas en los atributos del objeto de la clase Robot. Parae ello se debe emplear la función *NN_ creation* en caso de tener las redes ya entrenadas y *NN_ training* en caso contrario.

3. **Introducir el punto al que se pretende mover el robot**
   Una vez añadidas las redes neuronales, se procederá mediante el uso del método *Move* al movimiento del segmento en una serie de iteraciones hasta el punto seleccionado. Cabe destacar que este punto debe pertenecer al espacio de trabajo.

4. **Comprobación de punto de llegada**
   Finalmente, como paso opcional, se puede comprobar mediante el uso de las cámaras el error existente entre el punto objetivo y el punto alcanzado, medido como la distancia entre ambos puntos. Esto será útil especialmente para el capítulo siguiente: Pruebas y Resultados (capítulo 7).

Capítulo 7

Pruebas y Resultados

Finalmente, en esta última parte del trabajo, se describirán y expondrán los resultados obtenidos como cierre al desarrollo de este proyecto.

Para idear los experimentos necesarios para verificar la eficacia del trabajo realizado, es necesario fijarse en los objetivos del trabajo. Estos indican que se debe conseguir el control en posición del robot. Además, también se pretendía con este trabajo estudiar la posible modularidad que presenta este robot, pues es especialmente interesante al tratarse de una robot formado por varios segmentos como el controlado.

Esta es la razón por la que se ha ideado una serie de cuatro experimentos cuya finalidad es determinar no solo el error en el posicionamiento del robot, sino también la posibilidad que presenta en el futuro de implementar un control modular.

Destacar además que todos los datos de posición se dan respecto al sistema fijo del robot, observable en la figura 7.1. Además, las medidas están todas en **milímetros**.

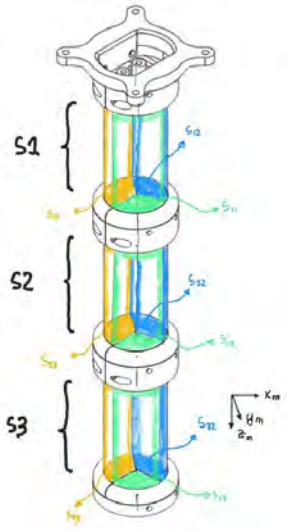


Figura 7.1 Sistema de coordenadas del robot y su descripción. *Fuente [1]*

A continuación se describen todas las pruebas realizadas.

7.1. Experimentos de posicionamiento

7.1.1. Primer experimento

Este primer experimento consiste en enviar al robot a posiciones aleatorias de su campo de trabajo previamente capturadas durante el proceso de obtención de los *datasets*. Estos serán puntos todos diferentes entre sí para verificar la capacidad de movimiento que ofrecen las combinaciones del hinchado de las distintas vejigas.

Destacar además que en este primer experimento se pretende evaluar exclusivamente el posicionamiento respecto de la posición de origen, por lo que para cada nuevo punto evaluado se procederá al desinflado completo de las vejigas. Algunos de los movimientos del robot se pueden ver en la figura 7.2.

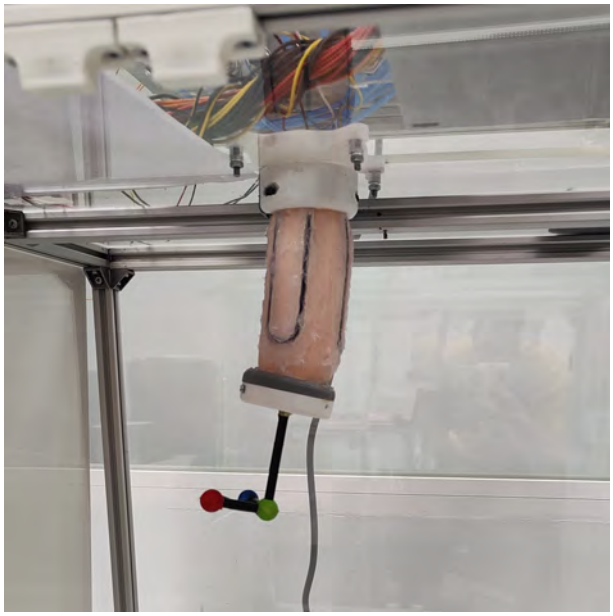


Figura 7.2 Robot PAUL durante el primer experimento de verificación de su capacidad de posicionamiento. *Fuente: elaboración propia*

El procedimiento consistía en aplicar el método *Move* y, al finalizar el posicionamiento según el funcionamiento descrito en el apartado 6.6, se capturaría el punto real en el que se encontraba el robot, calculando su error con el objetivo.

Se evaluaron 20 puntos. Los errores obtenidos junto a las posiciones evaluadas se pueden observar en la tabla 7.1.

punto	posición objetivo			posición alcanzada			error en milímetros
	x	y	z	x	y	z	(distancia entre los puntos)
1	22.214	98.35	-28.647	21.966	94.311	-28.239	4.0672
2	-9.462	92.37	27.803	-10.016	93.708	27.789	1.4486
3	28.38	97.02	18.53	26.1	94.301	14.61	5.2875
4	-23.184	89.366	-35.041	-20.190	90.213	-31.939	4.3927
5	-19.287	90.71	-15.217	-13.185	91.351	-14.734	6.1541
6	-11.49	91.1	-24.179	-11.3	90.812	-22.132	2.0758
7	35.663	98.92	-3.74	34.761	94.919	-3.561	4.1058
8	31.138	97.914	0.740	29.673	94.2126	5.393	6.1239
9	-4.186	90.809	-6.615	-4.977	90.979	-11.058	4.5166
10	-19.536	88.509	18.62	-19.465	90.609	17.725	2.2839
11	26.177	98.918	-30.373	26.177	94.724	-29.468	4.2907
12	-12.031	91.802	32.889	-11.153	93.786	28.739	4.6833
13	8.945	94.82	-12.83	7.931	92.535	-9.057	4.5249
14	18.934	97.15	26	17.205	95.535	22.496	4.2338
15	37.7	98	4.12	38.121	93.95	5.485	4.2895
16	-10.29	91.262	20.645	-8.261	92.404	17.035	4.2959
17	16.913	97.018	-31.2	16.054	93.597	-30.5	3.5962
18	-34.635	88.14	-7.377	-33.477	90.889	-7.294	2.9849
19	11.309	95.674	-31.46	10.452	92.891	-31.248	2.9197
20	-25.804	89.554	-8.239	-24.439	91.657	-7.894	2.5312
				Error medio			**3.9403**

Tabla 7.1 Resultados del primer experimento. *Fuente: elaboración propia*

7.1.2. Segundo experimento

Para este segundo experimento, se empleó un procedimiento muy similar al de la prueba anterior. En este caso, se enviaría también al robot a diferentes puntos de su espacio de trabajo. Sin embargo, en estas pruebas no se desinflarían las vejigas entre cada nuevo punto evaluado. Con ello, se quería demostrar y evaluar el comportamiento que presenta el segmento para moverse de un punto a otro sin necesidad de pasar por su estado de reposo, así como el efecto que tiene la histéresis de los sensores sobre dicho comportamiento.

Se evaluaron otros 20 puntos. Los errores obtenidos junto a las posiciones evaluadas se pueden observar en la tabla 7.2.

punto	posición objetivo			posición alcanzada			error en milímetros
	x	y	z	x	y	z	(distancia entre los puntos)
1	22.087	96.612	17.83	22.611	95.437	16.402	1.9218
2	-17.846	90.118	-28.355	-15.645	90.613	-29.956	2.7668
3	41.356	99.742	-4.877	49.054	95.814	-1.938	9.1275
4	0.048	91.749	5.21	1.262	91.533	1.172	4.2197
5	-40.714	86.415	-2.3	-41.524	90.037	-4.058	4.1076
6	14.206	96.273	-15.483	10.835	94	-14.491	4.1851
7	-27.496	87.232	12.22	-25.073	90.55	7.439	6.3036
8	20.944	97.82	-17.052	19.475	95.468	-16.697	2.7951
9	11.344	94.705	11.6	9.381	93.709	9.837	2.8175
10	-40.534	84.423	13.039	-42.464	88.108	12.750	4.1703
11	-17.874	89.479	-37.25	-17.840	89.031	-38.621	1.4437
12	38.926	100.719	-21.773	46.344	96.858	-17.652	9.3233
13	-1.165	94.342	39.452	0.631	97.266	37.371	4.0135
14	37.620	100.431	-21.705	39.571	96.881	-23.104	4.2857
15	42.721	99.21	-1.925	41.688	96.144	-0.673	3.4688
16	-0.154	91.505	-41.285	2.272	89.710	-42.070	3.1193
17	4.87	94.165	15.986	1.858	94.092	10.116	6.5980
18	-12.574	91.617	35.102	-6.353	95.748	32.360	7.9547
19	14.377	96.716	28.708	13.091	96.556	27.075	2.0848
20	9	94.534	-40.984	10.119	90.613	-46.806	7.1081
				Error medio			**4.5907**

Tabla 7.2 Resultados del segundo experimento. *Fuente: elaboración propia*

Con los resultados obtenidos en estos dos primeros experimentos se pone de manifiesto la excelente capacidad de posicionamiento que tiene, sobre todo teniendo en cuenta que se trata del control de un *Soft Robot* y que se han alcanzado los puntos tras una media de 5 iteraciones. Además, esta capacidad se da incluso cuando se "salta" de un punto a otro, sin necesidad de pasar por la posición de reposo. Esto indica que la histéresis presente en los sensores no afecta de forma significativa al posicionamiento, obteniendo alrededor de 0.6mm de error adicional de media con respecto al primer experimento. Por ello, se puede concluir además que el método de transformación de los sensores ha mejorado notablemente sus características, además de mejorar de manera sobresaliente su idoneidad para la sensorización de este tipo de robots así como su repetibilidad.

Se debe tener en cuenta además que los resultados obtenidos destacan cuando se comparan con la literatura, pues en trabajos como [22] o [23], realizados en el MIT, los resultados de error obtenidos para un robot muy similar al aquí expuesto es de alrededor de 20mm cuando se usan tres segmentos.

Para realizar una comparación más justa, ya que en el caso aquí expuesto solo se ha tenido en cuenta el control de uno de los segmentos, se va a considerar el mayor error posible que se puede dar suponiendo que cada módulo presenta, por separado y de forma independiente, un error con respecto a su posición objetivo igual al error medio máximo obtenido en los experimentos anteriores (4.59mm). La independencia entre los segmentos se justifica ya que la actuación de cada uno está completamente separada del resto de segmentos, por lo que para el cálculo del error que presenta con respecto a su posición objetivo (con respecto a su base) solo se debe tener en cuenta el error derivado de su propia actuación y no el procedente del error en posición del segmento superior, pues en lo único que afecta es en la nueva "base" del sistema de referencia móvil del segmento considerado.

Para ello, se considera la ecuación 7.1, que relaciona el módulo de la suma de dos vectores con los correspondientes módulos de dichos vectores. En esta ecuación, el valor de Θ es el ángulo que forman los vectores a y b entre sí, como se puede ver en la figura 7.3. Así, para el caso con dos segmentos, se consideran los vectores a y b como aquellos que unen los puntos objetivo y alcanzado correspondientes a cada segmento, por lo que su módulo será igual a la distancia entre ellos, equivalente a su vez al error considerado (4.59mm). De esta forma, el error acumulado máximo se da cuando los vectores a y b de la ecuación tienen la misma dirección y sentido ($\cos\Theta = 1$). Como ambos vectores tienen el mismo módulo (4.59mm), el error acumulado máximo tomaría un valor de $2a$, lo que es 9.18mm.

$$|\vec{a} + \vec{b}| = \sqrt{|\vec{a}|^2 + |\vec{b}|^2 + 2|\vec{a}||\vec{b}|\cos\Theta} \tag{7.1}$$

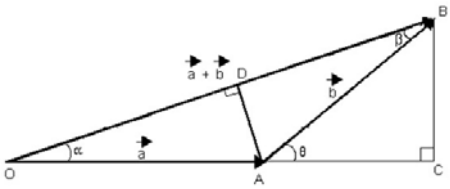


Figura 7.3 Módulo de la suma de dos vectores. *Fuente: elaboración propia*

Si se considera además un tercer segmento, la resolución de dicha ecuación daría lugar a que $error_{acum-max} = \sqrt{4{,}59^2 + 9{,}18^2 + 2 \cdot 4{,}59 \cdot 9{,}18 \cdot 1} = 13{,}77mm$. Tras estas operaciones se puede ver que la precisión obtenida es excelente, pues aún considerando el máximo error se obtiene un valor menor que el de otros proyectos similares realizados en universidades de renombre mundial.

Por último, cabe destacar que el rendimiento y la precisión en el posicionamiento del segmento, así como del robot, dependen en gran medida del número de muestras obtenidas para el entrenamiento de las redes neuronales, por lo que si se quieren obtener mejores resultados será necesario extraer más muestras del espacio de trabajo.

7.1.3. Tercer experimento

Finalmente, con este tercer experimento se quiso explorar la habilidad del segmento para dibujar figuras sencillas en el espacio.

Para ello, se seleccionaron 2 figuras: un cuadrilátero y un triángulo, pues de esta forma sería más sencillo ubicar los puntos de sus vértices. Para proceder al envío de los puntos de los vértices de las figuras, se creó un conjunto de tres puntos, en el caso del triángulo, y de cuatro puntos, en el caso del cuadrilátero, completamente nuevos e "inventados". Estos puntos se encontraban siempre dentro del interior del espacio de trabajo, observando aquellos puntos pertenecientes a la superficie que generaban los puntos extraídos durante la generación de los *datasets* de entrenamiento de las redes. Este espacio de trabajo se puede observar en la figura 7.4. Con ello se evaluaba además la respuesta que presentaba el robot ante puntos aleatorios de su espacio de trabajo.

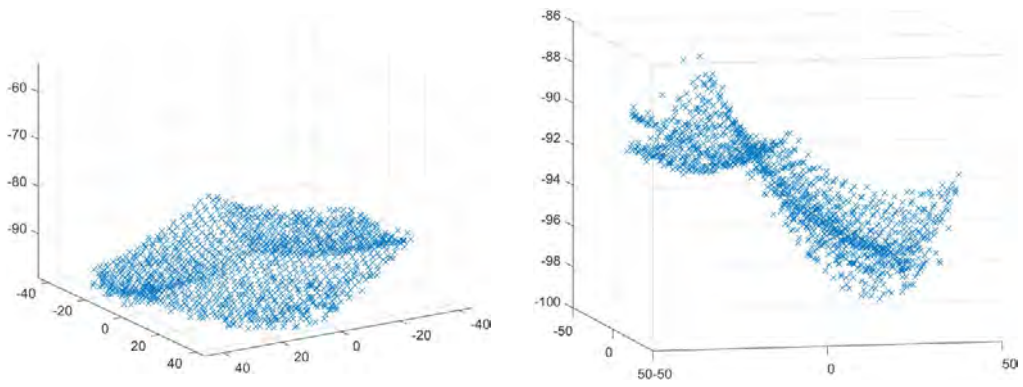
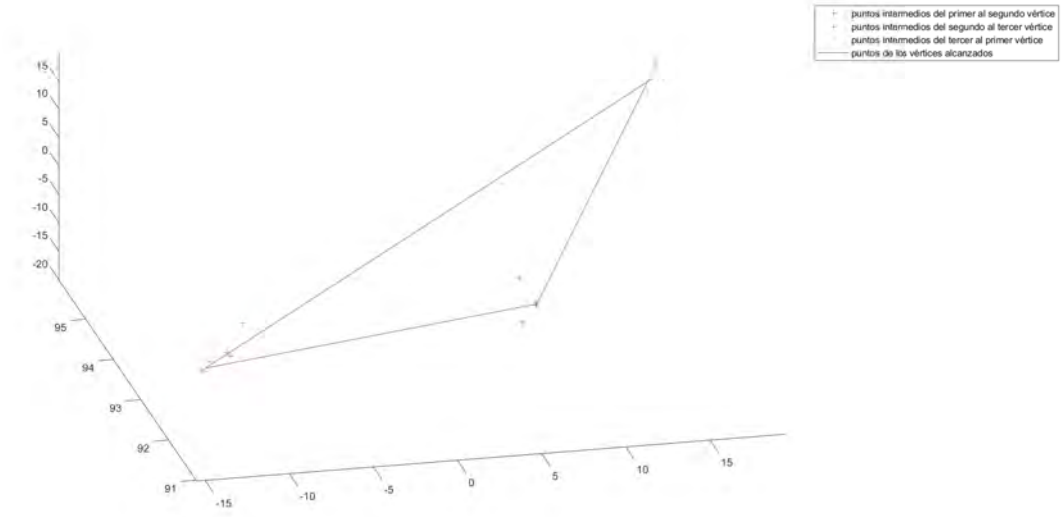


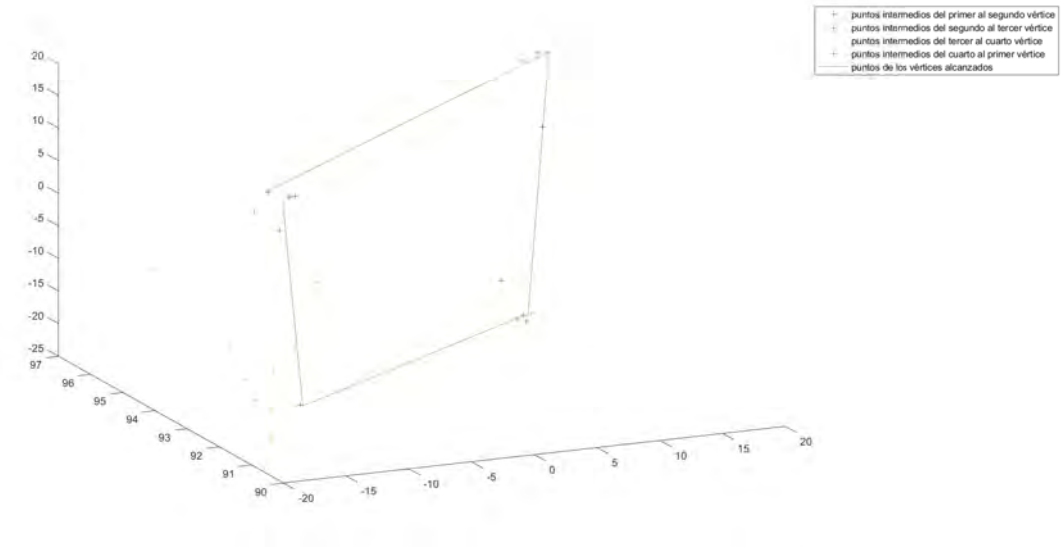
Figura 7.4 Espacio de trabajo de un segmento del robot PAUL. *Fuente: elaboración propia*

Como se puede ver, este espacio de trabajo está definido por tres esferas cortadas por el plano de unión entre ellos. Además, aquí se pues apreciar con especial claridad la asimetría en la deformación que sufre el segmento dependiendo de la vejiga hinchada. Aquí se pone de manifiesto este problema surgido durante el proceso

Los resultados obtenidos se pueden observar en la figura 7.5a para el triángulo y en la figura 7.5b para el cuadrilátero. Adicionalmente, la tabla 7.3, muestra las posiciones de los vértices a alcanzar y las realmente alcanzadas, así como la distancia entre ambos puntos.



(a) Posiciones tomadas durante el dibujo del triángulo



(b) Posiciones tomadas durante el dibujo del cuadrilátero

Figura 7.5 Movimientos realizados por el robot durante el tercer experimento. *Fuente: elaboración propia*

Triángulo							
Vértices	Punto real			Punto alcanzado			Error de posición (distancia entre puntos)
	x	y	z	x	y	z	
1	16	96	16	18.745	95.094	16.981	3.0531
2	16	96	-16	9.793	94.079	-16.928	6.5634
3	-16	90	0	-14.42	91.322	-2.955	3.6026
1	16	96	16	18.17	94.945	14.75	2.7191
				Error medio			**3.9846**

Cuadrilátero							
Vértices	Punto real			Punto alcanzado			Error de posición (distancia entre puntos)
	x	y	z	x	y	z	
1	-16	90	19	-17.491	91.51	15.005	4.5237
2	16	96	19	16.198	95.874	17.056	1.9576
3	16	96	-19	10.364	94.241	-17.429	6.1088
4	-16	91	-19	-14.917	91.386	-18.14	1.9583
1	-16	90	19	-16.359	91.447	13.385	5.809
				Error medio			**4.0715**

Tabla 7.3 Resultados de la prueba de realización de figuras. *Fuente: elaboración propia*

Las conclusiones extraídas a la vista de estos resultados son dos: el control del segmento es satisfactorio para puntos del espacio de trabajo no empleados en el entrenamiento, pues se mantienen los valores medios de errores observados con anterioridad, y es capaz de realizar figuras en el espacio. Esta última conclusión es fácilmente observable, pues los puntos intermedios tomados en el camino recorrido entre vértices están muy cercanos al propio vértice del cuadrilátero, lo que indica además una excelente movilidad para puntos lejanos unos de otros.

7.2. Experimentos de estudio de la modularidad

Para el estudio de la modularidad del robot, se procederá a realizar dos experimentos.

En el primero de ellos, se medirá el error de posicionamiento del robot cuando al segmento controlar se le cuelga otro segmento adicional. En la figura 7.6 se puede ver el montaje explicado.

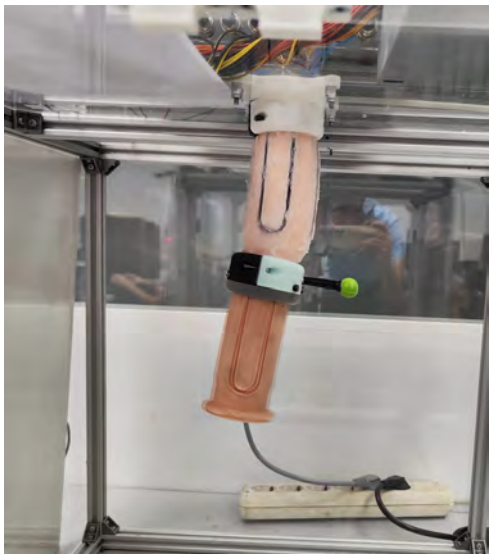


Figura 7.6 Robot PAUL durante el primer experimento del estudio de modularidad. *Fuente: elaboración propia*

Para el segundo de los experimentos, se realizará el mismo procedimiento pero colgando el segundo de los segmentos restantes que conforman el robot entero. En la figura 7.7 se puede apreciar este montaje.

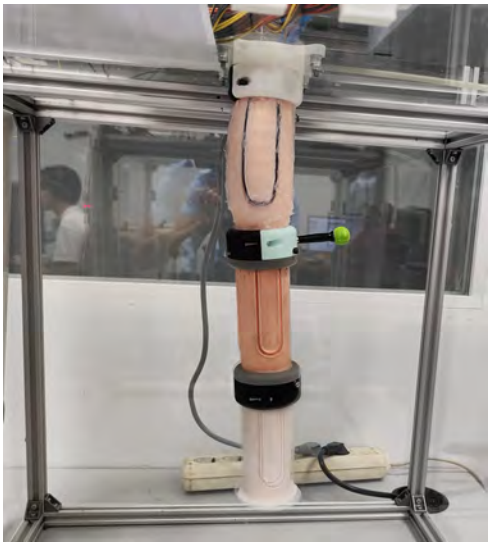
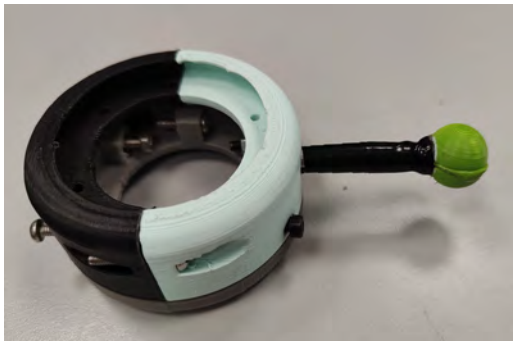


Figura 7.7 Robot PAUL durante el segundo experimento del estudio de modularidad. *Fuente: elaboración propia*

Para la captura de las posiciones, dada la imposibilidad del empleo de la baliza existente, se ha diseñado otra baliza con únicamente una esfera verde en ella, pues solo se necesita la posición y no las orientaciones. Esta nueva baliza se puede ver en la figura 7.8a. Así, mediante las correcciones oportunas realizadas mediante el atributo *geom* y seleccionando el modo de operación correcto (ya explicados en el capítulo anterior, sección 6.2), se puede capturar la posición del extremo inferior del segmento superior.



(a) Nueva baliza diseñada para realizar las pruebas de modularidad del robot

(b) Pieza de unión para fijar la nueva baliza

Figura 7.8 Piezas necesarias para la fabricación de la nueva baliza. *Fuente: elaboración propia*

Para llevar a cabo su fabricación, se emplearon las mismas técnicas que para la primera baliza (un tubo hueco de aluminio forrado con cinta aislante negra en cuyo extremo se encuentra una esfera pintada fabricada en PLA con impresión 3D). Para su correcto posicionamiento y fijación, se diseñó una nueva pieza *hembra* de unión entre segmentos, consistente en realizar un agujero a las piezas de este tipo ya existentes. Este diseño se puede apreciar en la figura 7.8b.

Los resultados del primer experimento se pueden observar en la tabla 7.4, mientras que los del segundo se encuentran en la tabla 7.5.

punto	**posición alcanzada sin peso**			**posición alcanzada con peso**			**error en milímetros (distancia entre los puntos)**
	x	y	z	x	y	z	
1	17.460	95.065	-4.691	12.142	95.024	-0.74	6.6254
2	22.936	100.815	-9.603	15.411	95.579	-2.004	11.9071
3	15.519	102.313	-10.921	6.532	98.855	-2.216	12.9804
4	-17.854	110.011	-9.296	-10.474	107.104	-4.915	9.0618
5	10.385	81.077	11.516	1.502	88.279	7.745	12.0413
6	0.882	85.955	9.619	-4.549	89.618	9.491	6.5527
7	0.293	87.987	5.585	-4.108	90.325	6.883	5.1503
8	17.556	85.956	2.173	13.055	85.302	4.383	5.0569
9	-1.007	87.417	6.686	-5.622	91.166	6.947	5.9520
10	6.977	109.607	-18.564	4	106.967	-11.874	7.7838
11	12.377	81.936	6.319	9.588	84.874	5.504	4.1321
12	-11.472	98.573	-0.611	-7.093	94.143	3.181	7.2929
13	-11.227	111.817	-14.788	-5.921	105.304	-4.495	13.2863
14	11.541	80.308	11.155	8.594	88.288	5.306	10.3235
15	-14.735	109.517	-10.193	-5.123	100.733	-2.765	14.9909
16	-16.415	112.833	-13.248	-3.426	103.698	-6.277	17.3426
17	-13.397	102.499	-3.303	-6.85	101	2.356	8.7833
18	-15.829	99.342	-0.243	-14.921	93.890	7.366	9.4057
19	13.757	108.230	-16.284	9.426	107.236	-11.452	6.5638
20	10.029	103.917	-12.311	6.133	98.165	-4.848	10.1959
				Error medio			**9.2714**

Tabla 7.4 Resultados del experimento con un segmento adicional. *Fuente: elaboración propia*

punto	**posición alcanzada sin peso**			**posición alcanzada con peso**			**error en milímetros (distancia entre los puntos)**
	x	y	z	x	y	z	
1	-9.694	88.473	9.414	-4.578	92.572	5.986	7.3972
2	-16.125	99.868	-1.095	-8.280	93.798	4.114	11.2035
3	-1.82	107	-14.754	0.744	100.452	-4.868	12.1277
4	14.939	105.751	-12.746	5.342	106.507	-5.714	11.9216
5	0.739	87.399	5.37	2.527	90.615	3.286	4.2286
6	-0.575	85.677	7.898	-4.925	92.727	5.4	8.6526
7	-4.634	83.575	13.788	-3.728	92.808	6.538	11.7746
8	-0.152	80.681	15.466	-5.339	93.449	6.157	16.6305
9	-13.277	89	8.249	-7.787	94.025	6.034	7.7606
10	9.115	77.865	12.863	2.399	90.5	5.45	16.1145
11	-12.644	88.909	7.891	-6.674	93.668	6.429	7.7737
12	-15.538	98.134	0.607	-5.239	93.412	2.83	11.5457
13	15.948	100.65	-7.439	4.115	91.161	0.621	17.1765
14	-2.051	88.116	6.039	-1.383	91.560	4.680	3.7617
15	12.906	78.245	9.931	10.119	86.684	3.983	10.6940
16	4.474	78.835	14.34	-1.2	91.969	7.724	15.7634
17	7.300	80.577	10.89	5.256	89.044	5.507	10.2394
18	-13.634	102.977	-4.27	-3.511	98.191	-0.3	11.8803
19	7.509	81.361	9.561	4.865	90.329	3.856	10.9526
20	-19.262	113.54	-12.773	-0.292	95.023	-1.184	28.9313
				Error medio			**11.8265**

Tabla 7.5 Resultados del experimento con dos segmentos adicionales. *Fuente: elaboración propia*

Los resultados que se muestran son los puntos alcanzados para una misma posición objetivo cuando se realiza el movimiento sin ningún peso (solo el del segmento a controlar) y con el peso de uno o dos segmentos adicionales, respectivamente.

Se ha realizado de esta manera, y no midiendo directamente el error con respecto al punto objetivo, porque la finalidad de estos experimentos es estudiar hasta que punto se asemeja el comportamiento de uno de los segmentos por separado al comportamiento del mismo segmento cuando se le aplica el peso de otros segmentos adicionales. De esta forma, si la distancia entre los puntos alcanzados con y sin peso es muy grande, se consideraría que el comportamiento está muy alejado de la idealidad y que, por lo tanto, el robot no está preparado para realizar un control modular sobre él; esto es, no se puede emplear el mismo sistema de control de un solo segmento para controlar el robot entero únicamente por cambios de base entre los distintos módulos que lo componen.

En definitiva, y a la vista de estos resultados, se puede concluir que el robot no presenta un comportamiento modular; es decir, no se puede apreciar una solución para su posicionamiento que implique separar el problema del control en un número igual al número de segmentos del que se componen el robot.

El principal motivo observado por el que se hace extremadamente complejo realizar esta labor es la gran oscilación presentada durante el movimiento debido al peso del conjunto, a la histéresis de los sensores, que se acentúa al estirarse el segmento por el peso que soporta, y a la no linealidad del comportamiento del robot. Todo ello modifica las medidas obtenidas de los sensores, lo que hace que sea muy complejo alcanzar el punto objetivo. Esto se verifica con la observación del comportamiento del robot durante el experimento, el cual, en aproximadamente la mitad de los puntos evaluados, salió del bucle de control por agotamiento del número de iteraciones máximo.

Es importante señalar que este comportamiento no se debe exclusivamente a las redes neuronales empleadas, sino también a la rigidez de la estructura del robot. Es por ello que exigiría reentrenar la red cada vez que se le añada un peso o un segmento adicional, lo cual hace impracticable el uso de un control modular si no se varían los materiales para su fabricación. Es por ello que se insta en futuros trabajos a abordar un control basado en el mismo sistema de control aquí propuesto modificando la cantidad de entradas y salidas de las distintas redes neuronales en función del número de sensores empleados y el número de grados de libertad de movimiento totales a controlar.

Capítulo 8

Conclusiones y líneas futuras

La realización de este trabajo ha supuesto el desarrollo de un sistema de control en posición para un *Soft Robot* neumático de tres grados de libertad, ampliables a nueve. Para ello, se han abordado distintas fases, como la caracterización e implementación de unos sensores resistivos elastómeros, la realización de cambios en el proceso de fabricación o diseño del robot, así como la resolución de los problemas de diseño del sistema de control y el proceso de entrenamiento de redes neuronales.

Todo lo aquí realizado ha conllevado la mejora de numerosas competencias relacionadas con el ámbito de la ingeniería industrial, como la programación en MATLAB y en C++, el diseño CAD, la impresión 3D o el uso de la creatividad para la resolución de problemas. A su vez, se han adquirido nuevos conocimientos en el campo de la robótica, como los fundamentos de la visión por computador, la fabricación con silicona, la caracterización de sensores o el fundamento y entrenamiento de redes neuronales.

A continuación, se recoge a modo de síntesis las conclusiones finales del proyecto, los distintos desarrollos realizados y los aprendizajes obtenidos durante este trabajo. Asimismo, se hace una propuesta de posibles mejoras y líneas futuras para la continuación de este proyecto.

8.1. Conclusiones

Este proyecto ha servido para poner de manifiesto muchas de las características y propiedades inherentes a los robots blandos. También se ha podido comprobar que la metodología de trabajo con estos robots difiere en gran medida de la empleada en la robótica tradicional. Estas diferencias incluyen todas las fases de creación y de vida útil del robot; esto es, desde su fase de diseño y fabricación hasta su puesta en marcha. Es por ello que su cuerpo flexible y su relativamente escaso desarrollo convierten el desarrollo de un robot de este tipo en un verdadero reto tecnológico.

Durante la realización de este trabajo, se ha podido comprobar además que todavía hay una gran necesidad de investigación en el ámbito de la sensorización de este tipo de robots, pues todavía no se han encontrado las técnicas de medición de las características propias del robot que permitan mecanismos de control o modelado similares a los conseguidos en la robótica tradicional. Además, se ha podido ver que los materiales empleados en la fabricación de sensores presentan una gran cantidad de defectos, relacionados principalmente con la necesidad de flexibilidad, que obstaculizan tanto su caracterización como su utilización.

Adicionalmente, a estos aspectos se le suma la dificultad y complejidad de su diseño y fabricación, pues, al no contar con partes rígidas, la probabilidad de rotura y fallo del robot es mayor, por lo que se debe tener un especial cuidado y meticulosidad durante las fases de creación.

De hecho, este último aspecto de los robots blandos es uno de los que más tiempo ha consumido durante este proyecto. Los innumerables problemas de fugas surgidos durante la fabricación de los segmentos fue uno de los problemas más complejos a solventar. Aún así, finalmente, y tras un largo proceso creativo y de búsqueda de soluciones, se consiguió encontrar una combinación de técnicas nuevas de fabricación y nuevas ideas de diseño que solventara este problema de forma definitiva sin aumentar el coste total del proyecto.

Sin embargo, esta no fue la única dificultad presentada ni la que más tiempo consumió. La fase que más complejidad presentó fue la del desarrollo del sistema de control, pues son todas estas características mencionadas de los robots blandos las que hacen que las técnicas de control y modelado sean irremediable y notablemente más complejas que las que se requieren en los robots tradicionales.

Este proceso de búsqueda de la mejor técnica de control que permitiera errores en el posicionamiento del robot mínimos requirió a su vez una gran cantidad de tiempo y de pruebas. Es esta una de las razones por las que finalmente no se puedo alcanzar el desarrollo total del control del robot de 3 segmentos. Sin embargo, como se vio en la sección 8.2, sí se ha ideado un método para llevar a cabo dicho objetivo. Además, también se ha estudiado la posibilidad de modularidad que presenta este robot.

A pesar de todas estas dificultades, se consiguió llevar a cabo un y con éxito el control en posición de uno de los segmentos del robot, consiguiendo a su vez una precisión similar e incluso mayor que la presentada en muchos de los principales artículos de la literatura relacionada con la robótica blanda.

Con todo ello, se concluye este trabajo afirmando que, a pesar de que se hubiera deseado desarrollar algunos aspectos en mayor profundidad, se ha diseñado un robot blando funcional y sensorizado que, junto a la programación software que lo acompaña, satisface todos los requisitos establecidos al comienzo de este proyecto.

8.2. Líneas futuras

A pesar de haber abordado y completado todos los objetivos descritos en la sección 1.2, se considera que todavía hay margen de mejora para la realización de nuevos trabajos y para su posible aplicación a la industria. Algunas de estas líneas futuras son:

- **En relación con la sensorización del robot**
  Estas mejoras están relacionadas tanto con los sensores empleados como con el propio método de sensorización. Estas son:

  - **Caracterización a nivel microestructural del nuevo comportamiento de los sensores:**
    Uno de los mayores descubrimientos realizados durante este proyecto ha sido el cambio radical en el comportamiento de los sensores comerciales. Sin embargo, este hecho ha generado múltiples preguntas que generan incertidumbre de su estabilidad y comportamiento en el futuro. Es por ello que se propone realizar una investigación en profundidad de los cambios en la microestructura del material que ha llevado a este cambio de comportamiento, así como su estabilidad en el tiempo y la conservación de dichas propiedades.
  - **Empleo y búsqueda de técnicas de sensorización más recientes:**
    Con el fin de obtener una mayor precisión en las medidas o para facilitar el control del robot en cadena cerrada, se propone emplear y caracterizar sensores blandos de más reciente uso, como los vistos en el capítulo 2: Estado del arte. También se puede estudiar y diseñar el proceso de fabricación de un sensor blando elástico que permita mejorar y ampliar el sistema de control del robot, desarrollando por ejemplo un control en velocidad, aceleración o fuerza.

- **En relación con el diseño del robot**
  Estas mejoras estarían motivadas por los problemas de fugas acaecidos durante la realización del proyecto y las aplicaciones del robot.

  - **Mejoras en el diseño y el proceso de fabricación:**
    Uno de los mayores problemas acontecidos durante el desarrollo del proyecto fueron las innumerables fugas y pinchazos que presentaba el robot al actuarlo. El origen de estas fugas era tanto la presión en el interior de las vejigas como el propio proceso de fabricación y diseño del robot. Es por ello que se propone realizar un diseño mejorado contando con todos los requisitos de sensorización y control necesarios, así como el empleo de materiales que permitan una mayor rigidez en las paredes de los segmentos sin perder flexibilidad y capacidad de deformación. También es aconsejable la adquisición de una bomba de vacío para extraer todo el aire atrapado en el interior de la silicona. De esta forma

se reduciría el número de burbujas del material ya curado y, con ello, el riesgo de fugas.

- **Implementación de herramientas en el extremo:**
  Una vez realizados los trabajos de diseño, sensorización y control entre el trabajo de Adrián [1] y el aquí presentado, el siguiente objetivo es idear las posibles aplicaciones del robot. Para ello, será necesario incorporar distintas herramientas al robot para poder realizar las numerosas tareas diseñadas y realizar un control robusto ante los cambios de peso para permitir la manipulación de objetos.

- **En relación con el modelado del robot**
  Esta mejora viene motivada por la imposibilidad debido al alcance del proyecto de diseñar un modelo del Robot PAUL. Es por ello que se propone realizar y encontrar un modelo que, mediante técnicas de elementos finitos y simulación, permita conocer y predecir el comportamiento ante la deformación del robot y el posicionamiento final de su extremo.

- **En relación con el sistema de control**
  Estas mejoras están relacionadas con el diseño del sistema de control del robot.

  - **Diseño de un sistema de control global:**
    En este proyecto solo se ha realizado el sistema de control de uno de los segmentos de los que consta el robot. Es por ello que aquí se propone continuar con la idea expuesta, pues se considera que la aplicación de un sistema de control de N segmentos directamente obtenible del aquí desarrollado. Esto se justifica con la facilidad de extracción de los datos con el procedimiento diseñado y el entrenamiento de las redes neuronales necesarias, teniendo en cuenta un mayor número de entrada (igual al producto del número de segmentos por 3 sensores en cada uno) y un mayor número de salidas (idealmente los 5 grados de libertad posibles para el robot, pues el giro del extremo sobre su eje no es posible realizarlo). También existe la posibilidad de diseñar un sistema de control modular; es decir, que mediante el modelado de un segmento se pueda conseguir el control total de un robot con N segmentos únicamente mediante la traslación y rotación de sus sistemas de referencia.
  - **Implementación de nuevos tipos de control:**
    Durante este proyecto solo se ha abordado el control en posición. Sin embargo, dependiendo de las aplicaciones futuras del robot, será necesario implementar otros tipos de técnicas de control, como el control en velocidad, en aceleración o el control de la fuerza ejercida por el extremo. Por ello, se propone investigar acerca de la posibilidad de desarrollo de este tipo de metodología de control.

Bibliografía

[1] A. S. Rieker González, "Diseño, fabricación y control de un robot blando neumático", Trabajo Fin de Grado, Universidad Politécnica de Madrid, feb. de 2023. dirección: https://oa.upm.es/72987/.

[2] A. Industries. "Conductive Rubber Cord Stretch Sensor + Extras!" (), dirección: https://www.adafruit.com/product/519.

[3] P. Polygerinos, N. Correll, S. A. Morin, B. Mosadegh, C. D. Onal, K. Petersen, M. Cianchetti, M. T. Tolley y R. F. Shepherd, "Soft robotics: Review of fluid-driven intrinsically soft devices; manufacturing, sensing, control, and applications in human-robot interaction", *Advanced Engineering Materials*, vol. 19, n.º 12, pág. 1 700 016, 2017.

[4] D. Kar, B. George y K. Sridharan, "A review on flexible sensors for soft robotics", *Systems for Printed Flexible Sensors: Design and implementation*, págs. 1-1, 2022.

[5] X. Sun, F. Yao y J. Li, "Nanocomposite hydrogel-based strain and pressure sensors: a review", *J. Mater. Chem. A*, vol. 8, págs. 18 605-18 623, 36 2020. DOI: 10.1039/D0TA06965E. dirección: http://dx.doi.org/10.1039/D0TA06965E.

[6] E. Feng, G. Zheng, M. Zhang, X. Li, G. Feng y L. Cao, "Self-healing and freezing-tolerant strain sensor based on a multipurpose organohydrogel with information recording and erasing function", *Colloids and Surfaces A: Physicochemical and Engineering Aspects*, vol. 672, pág. 131 781, 2023, ISSN: 0927-7757. DOI: https://doi.org/10.1016/j.colsurfa.2023.131781. dirección: https://www.sciencedirect.com/science/article/pii/S0927775723008658.

[7] C. Shan, M. Che, A. Cholewinski, J. KI Kunihiro, E. K. Yim, R. Su y B. Zhao, "Adhesive hydrogels tailored with cellulose nanofibers and ferric ions for highly sensitive strain sensors", *Chemical Engineering Journal*, vol. 450, pág. 138 256, 2022, ISSN: 1385-8947. DOI: https://doi.org/10.1016/j.cej.2022.138256. dirección: https://www.sciencedirect.com/science/article/pii/S1385894722037391.

[8] J. Zhang, W. Xue, Y. Dai, C. Wu, B. Li, X. Guo, B. Liao y W. Zeng, "Ultrasensitive, flexible and dual strain-temperature sensor based on ionic-conductive composite hydrogel for wearable applications", *Composites Part A: Applied Science and Manufacturing*, vol. 171, pág. 107 572, 2023, ISSN: 1359-835X. DOI: https://doi.org/10.1016/j.compositesa.2023.107572. dirección: https://www.sciencedirect.com/science/article/pii/S1359835X23001483.

[9] Q. Ji, J. Jansson, M. Sjöberg, X. V. Wang, L. Wang y L. Feng, "Design and calibration of 3D printed soft deformation sensors for soft actuator control", *Mechatronics*, vol. 92, pág. 102 980, 2023, ISSN: 0957-4158. DOI: https://doi.org/10.1016/j.mechatronics.2023.102980. dirección: https://www.sciencedirect.com/science/article/pii/S0957415823000363.

[10] X. Sun, F. Yao y J. Li, "Nanocomposite hydrogel-based strain and pressure sensors: a review", *Journal of Materials Chemistry A*, vol. 8, n.º 36, págs. 18 605-18 623, 2020.

[11] U. Shahzad, H. M. Marwani, M. Saeed, A. M. Asiri y M. M. Rahman, "Two-dimensional MXenes as Emerging Materials: A Comprehensive Review", *ChemistrySelect*, vol. 8, n.º 25, e202300737, 2023.

[12] T. Du, X. Han, X. Yan, J. Shang, Y. Li y J. Song, "MXene-Based Flexible Sensors: Materials, Preparation, and Applications", *Advanced Materials Technologies*, vol. 8, n.º 12, 2023, Cited by: 0. DOI: 10.1002/admt.202202029. dirección: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85152246971&doi=10.1002%2fadmt.202202029&partnerID=40&md5=21808a1a5bd7af3a32f6f73712300a07.

[13] S. Sharma, A. Chhetry, S. Zhang, H. Yoon, C. Park, H. Kim, M. Sharifuzzaman, X. Hui y J. Y. Park, "Hydrogen-Bond-Triggered Hybrid Nanofibrous Membrane-Based Wearable Pressure Sensor with Ultrahigh Sensitivity over a Broad Pressure Range", *ACS Nano*, vol. 15, n.º 3, págs. 4380-4393, 2021, PMID: 33444498. DOI: 10.1021/acsnano.0c07847. eprint: https://doi.org/10.1021/acsnano.0c07847. dirección: https://doi.org/10.1021/acsnano.0c07847.

[14] Z. Zhang, Q. Zhang, H. Zhang, B. Li, J. Zang, X. Zhao, X. Zhao y C. Xue, "A novel MXene-based high-performance flexible pressure sensor for detection of human motion", *Smart Materials and Structures*, vol. 32, n.º 6, 2023. DOI: 10.1088/1361-665X/accee9. dirección: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85158821723&doi=10.1088%2f1361-665X%2faccee9&partnerID=40&md5=1c862df6564461e552a3a00be9a3847d.

[15] S. Lee, P. Bhuyan, K. J. Bae, J. Yu, H. Jeon y S. Park, "Interdigitating Elastic Fibers with a Liquid Metal Core toward Ultrastretchable and Soft Capacitive Sensors: From 1D Fibers to 2D Electronics", *ACS Applied Electronic Materials*, vol. 4, n.º 12,

págs. 6275-6283, 2022. DOI: 10.1021/acsaelm.2c01382. eprint: https://doi.org/10.1021/acsaelm.2c01382. dirección: https://doi.org/10.1021/acsaelm.2c01382.

[16] J. Jung, M. Park, D. Kim e Y.-L. Park, "Optically Sensorized Elastomer Air Chamber for Proprioceptive Sensing of Soft Pneumatic Actuators", *IEEE Robotics and Automation Letters*, vol. 5, n.º 2, págs. 2333-2340, 2020. DOI: 10.1109/LRA.2020.2970984.

[17] T. Paulino, P. Ribeiro, M. Neto, S. Cardoso, A. Schmitz, J. Santos-Victor, A. Bernardino y L. Jamone, "Low-cost 3-axis soft tactile sensors for the human-friendly robot Vizzy", en *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, págs. 966-971. DOI: 10.1109/ICRA.2017.7989118.

[18] Z. Xing, J. Lin, D. McCoul, D. Zhang y J. Zhao, "Inductive Strain Sensor With High Repeatability and Ultra-Low Hysteresis Based on Mechanical Spring", *IEEE Sensors Journal*, vol. 20, n.º 24, págs. 14 670-14 675, 2020. DOI: 10.1109/JSEN.2020.3010345.

[19] H. Wang, J. Kow, N. Raske, G. de Boer, M. Ghajari, R. Hewson, A. Alazmani y P. Culmer, "Robust and high-performance soft inductive tactile sensors based on the Eddy-current effect", *Sensors and Actuators A: Physical*, vol. 271, págs. 44-52, 2018, ISSN: 0924-4247. DOI: https://doi.org/10.1016/j.sna.2017.12.060. dirección: https://www.sciencedirect.com/science/article/pii/S0924424717317855.

[20] X. Wang, Y. Li y K. W. Kwok, "A Survey for Machine Learning-Based Control of Continuum Robots", *Frontiers in Robotics and AI*, vol. 8, n.º September, págs. 1-14, 2021, ISSN: 22969144. DOI: 10.3389/frobt.2021.730330.

[21] O. Yasa, Y. Toshimitsu, M. Y. Michelis, L. S. Jones, M. Filippi, T. Buchner y R. K. Katzschmann, "An Overview of Soft Robotics", *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 6, págs. 1-29, 2023, Cited by: 1; All Open Access, Green Open Access. DOI: 10.1146/annurev-control-062322-100607. dirección: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85159846351&doi=10.1146%2fannurev-control-062322-100607&partnerID=40&md5=3b4c475ff075236320a4b8882d638b22.

[22] R. K. Katzschmann, M. Thieffry, O. Goury, A. Kruszewski, T.-M. Guerra, C. Duriez y D. Rus, "Dynamically Closed-Loop Controlled Soft Robotic Arm using a Reduced Order Finite Element Model with State Observer", en *2019 2nd IEEE International Conference on Soft Robotics (RoboSoft)*, 2019, págs. 717-724. DOI: 10.1109/ROBOSOFT.2019.8722804.

[23] R. K. Katzschmann, C. D. Santina, Y. Toshimitsu, A. Bicchi y D. Rus, "Dynamic Motion Control of Multi-Segment Soft Robots Using Piecewise Constant Curvature Matched with an Augmented Rigid Body Model", en *2019 2nd IEEE International Conference on Soft Robotics (RoboSoft)*, 2019, págs. 454-461. DOI: 10.1109/ROBOSOFT.2019.8722799.

[24] J. Till, V. Aloi y C. Rucker, "Real-time dynamics of soft and continuum robots based on Cosserat rod models", *The International Journal of Robotics Research*, vol. 38, n.º 6, págs. 723-746, 2019. DOI: 10.1177/0278364919842269. eprint: https://doi.org/10.1177/0278364919842269. dirección: https://doi.org/10.1177/0278364919842269.

[25] C. D. Santina y D. Rus, "Control Oriented Modeling of Soft Robots: The Polynomial Curvature Case", *IEEE Robotics and Automation Letters*, vol. 5, n.º 2, págs. 290-298, abr. de 2020, ISSN: 2377-3766. DOI: 10.1109/LRA.2019.2955936.

[26] I. Singh, Y. Amara, A. Melingui, P. Mani Pathak y R. Merzouki, "Modeling of Continuum Manipulators Using Pythagorean Hodograph Curves", *Soft Robotics*, vol. 5, n.º 4, págs. 425-442, 2018, PMID: 29746203. DOI: 10.1089/soro.2017.0111. eprint: https://doi.org/10.1089/soro.2017.0111. dirección: https://doi.org/10.1089/soro.2017.0111.

[27] G. Soter, A. Conn, H. Hauser y J. Rossiter, "Bodily Aware Soft Robots: Integration of Proprioceptive and Exteroceptive Sensors", en *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, págs. 2448-2453. DOI: 10.1109/ICRA.2018.8463169.

[28] T. G. Thuruthel, E. Falotico, F. Renda y C. Laschi, "Model-Based Reinforcement Learning for Closed-Loop Dynamic Control of Soft Robotic Manipulators", *IEEE Transactions on Robotics*, vol. 35, n.º 1, págs. 124-134, 2019. DOI: 10.1109/TRO.2018.2878318.

[29] "INA219 Data Sheet, product information and support | TI.com". (), dirección: https://www.ti.com/product/INA219.

[30] "INA226 Data Sheet, Product information and Support | TI.com". (), dirección: https://www.ti.com/product/INA226.

[31] "INA3221 Data Sheet, Product information and Support | TI.com". (), dirección: https://www.ti.com/product/INA3221?dcmp=dsproject&hqs=pf.

[32] J. Di Tocco, D. L. Presti, A. Rainer, E. Schena y C. Massaroni, "Silicone-Textile Composite Resistive Strain Sensors for Human Motion-Related Parameters", *Sensors*, vol. 22, n.º 10, págs. 1-16, 2022, ISSN: 14248220. DOI: 10.3390/s22103954.

[33] S. Błazewicz, B. Patalita y P. Touzain, "Study of piezoresistance effect in carbon fibers", *Carbon*, vol. 35, n.º 10-11, págs. 1613-1618, 1997, ISSN: 00086223. DOI: 10.1016/S0008-6223(97)00120-6.

[34] "DOWSILTM 732 Multi-Purpose Sealant". (), dirección: https://www.dow.com/es-es/pdp.dowsil-732-multi-purpose-sealant.01890581z.html#properties.

[35] colaboradores de Wikipedia, "Red neuronal prealimentada", *Wikipedia, la enciclopedia libre*, 24 de oct. de 2022. dirección: https://es.wikipedia.org/wiki/Red_neuronal_prealimentada.

[36] F. Murzone, "Funciones de activación para redes neuronales - EscuelaDeInteligenciaArtificial - Medium", 30 de mar. de 2022. dirección: https://medium.com/escueladeinteligenciaartificial/funciones-de-activaci%C3%B3n-para-redes-neuronales-de00fefb7150.

[37] D. L. González, "Red Neuronal feedforward", *es.linkedin.com*, dirección: https://es.linkedin.com/pulse/red-neuronal-feedforward-diego-l%C3%B3pez-g-.

[38] "Fig 3. Celda de una red neuronal recurrente LSTM. donde los valores de..." (), dirección: https://www.researchgate.net/figure/Celda-de-una-red-neuronal-recurrente-LSTM-Donde-los-valores-de-C-son-los_fig3_343450066.

[39] B. C. Csáji y otros, "Approximation with artificial neural networks", *Faculty of Sciences, Etvs Lornd University, Hungary*, vol. 24, n.º 48, pág. 7, 2001.

[40] "Train and apply multilayer shallow neural networks - MATLAB y Simulink - MathWorks España". (), dirección: https://es.mathworks.com/help/deeplearning/ug/train-and-apply-multilayer-neural-networks.html.

[41] A. Centurelli, L. Arleo, A. Rizzo, S. Tolu, C. Laschi y E. Falotico, "Closed-Loop Dynamic Control of a Soft Manipulator Using Deep Reinforcement Learning", *IEEE Robotics and Automation Letters*, vol. 7, n.º 2, págs. 4741-4748, 2022, ISSN: 23773766. DOI: 10.1109/LRA.2022.3146903.

[42] W. Sun, N. Akashi, Y. Kuniyoshi y K. Nakajima, "Physics-Informed Recurrent Neural Networks for Soft Pneumatic Actuators", *IEEE Robotics and Automation Letters*, vol. 7, n.º 3, págs. 6862-6869, 2022, ISSN: 23773766. DOI: 10.1109/LRA.2022.3178496.

[43] K. Mojsiewicz-Pieńkowska, M. Jamrógiewicz, K. Szymkowska y D. Krenczkowska, "Direct human contact with siloxanes (silicones)–safety or risk part 1. Characteristics of siloxanes (silicones)", *Frontiers in pharmacology*, vol. 7, pág. 132, 2016.

[44] P. Líder y P. Líder, "Precio hora ingeniero técnico industrial", *Presupuesto Líder*, 14 de abr. de 2023. dirección: `https://www.presupuestolider.es/precio-hora-ingeniero-tecnico-industrial/`.

Apéndice A

Estudio de impacto ambiental, social, ético y legal

En la actualidad, el desarrollo de proyectos de ingeniería se ve especialmente influenciado por sus potenciales impactos en el medio ambiente, la sociedad o en la propia industria. Es por ello que para evaluar su viabilidad es necesario analizar las consecuencias que tendría la puesta en marcha del proyecto. A esto se le conoce como *Estudios de Impacto*.

Estos estudios son de especial relevancia, pues en la gran mayoría de los casos implican nuevos requisitos que se deben incluir a los inherentes al propio proyecto.

En el trabajo aquí expuesto, también se ha tenido en cuenta el impacto que puede llegar a tener en la sociedad y en el medio ambiente. A continuación, se expone un estudio de los impactos tanto ambiental como social, ético y legal.

A.1. Impacto ambiental

Con respecto al impacto ambiental, en proyectos similares a este se debe estudiar en primer lugar el *ciclo de vida* del producto, en este caso el robot, pues se trata de un producto cuyas consecuencias medioambientales no solo tienen relevancia en el presente, sino que pueden afectar a la vida en el futuro. A este tipo de productos se los conoce como *tangibles*.

Sin embargo, por motivos de alcance y tiempo del proyecto no ha sido posible realizar un estudio del ciclo de vida de los productos empleados en la fabricación del robot. Aún así, sí se ha realizado un análisis del impacto ambiental de los distintos elementos empleados durante el transcurso del proyecto.

Uno de los materiales más empleados en este trabajo fue el PLA de impresión 3D con el que estaban hechos tanto moldes como uniones entre segmentos, así como otros muchos

componentes claves del robot (como las balizas de captura de imágenes). La ventaja del empleo de este material es su elevada reciclabilidad, pues al ser un polímero termoplástico, se puede fundir y moldear de nuevo sin perder ninguna de sus propiedades ni cantidad de material. Asimismo, los moldes empleados pueden servir como piezas con otras funciones o, incluso, pueden ser reutilizados realizando alguna operación de mecanizado sobre ellos, como un taladrado.

También se deben tener en cuenta la gran facilidad de reciclaje que presentan los machos de cera que generan las cámaras internas del robot. Estos, como ya comentó Adrián Rieker [1], permiten volver a ser fundidos para usarse como nuevos machos.

En lo referente a los componentes electrónicos empleados, destaar que todos cumplen la reciente directiva europea *RoHS* (Directiva 2011/65/UE), aplicada en España mediante el Real Decreto 993/2022 del 16 de enero de 2023, con el que se pretende reducir el uso de sustancias peligrosas, como los metales pesados, entre los que se encuentra principalmente el plomo, usados en dispositivos electrónicos, especialmente en sus soldaduras blandas.

Con todos estos requisitos aplicados al proyecto se colabora de forma directa e indirecta con los **ODS 13: Acción por el clima y 15: Vida de ecosistemas terrestres**, pues ayuda a reducir la necesidad de fabricación de nuevos componentes gracias a su reutilización y reciclaje. Con ello, se evita la generación de desechos potencialmente contaminantes tanto a la atmósfera como a los ecosistemas terrestres.

Por otra parte, uno de los aspectos más relevantes en materia de impacto ambiental es el uso de la silicona de fabricación de los segmentos y del sellador DOWSIL 732 de fijación de los sensores.

Con respecto a la silicona empleada en la fabricación del robot, un análisis más profundo se puede hallar en el Trabajo Fin de Grado de Adrián Rieker [1]. Aún así, comentar que se trata de un material difícilmente reciclable y no biodegradable. Sin embargo, presenta otros beneficios más que notables, como su menor impacto en la vida humana en comparación con los plásticos. La principal razón que explica este hecho es precisamente su no biodegradabilidad, lo que los hace inertes, no tóxicas cuando están curadas y más estables, pues no generan microplásticos [43].

En cuanto al sellador DOWSIL 732, destacar que a pesar de no ser biodegradable ni reciclable, tampoco es perjudicial para el ser humano y, en pequeñas cantidades, para los ecosistemas marinos. Es un producto no tóxico, sin componentes peligrosos y no inflamable. Es por ello que se considera su uso un impacto limitado en el medio ambiente. Aún así, se ha limitado su uso al mínimo posible y necesario, así como se han seguido las medidas pertinentes para su eliminación.

Por último, los sensores empleados, al estar fabricados con goma, son fácilmente reciclables, lo que les otorga una gran ventaja al compararlos con otros materiales vistos en el capítulo 2. A pesar de ello, se ha intentado limitar su uso mediante la reutilización de los mismos sensores en las diferentes pruebas realizadas durante el proyecto. Lo mismo aplica

para el banco de pruebas construido para su caracterización, pues se puede aprovechar fácilmente para otros trabajos en el futuro.

Con todo ello se contribuye además al **ODS 12: Producción y consumo responsables**.

A.1.1. Impacto social

Con respecto al impacto social del proyecto aquí realizado, se pueden realizar varias menciones destacables.

La primera de ellas es que, al tratarse de un robot con una estructura blanda, es mucho más seguro su uso en entornos de trabajo con humanos. El motivo de esto es que cualquier pérdida de control en sus movimientos o cualquier choque accidental que se pueda producir se ve mitigado gracias a su estructura maleable, sobre todo en comparación con los robots tradicionales rígidos. Es por ello que los *Soft Robots* están a la vanguardia en seguridad de trabajo con seres humanos, solo comparables con los denominados *cobots*.

La segunda observación a realizar está relacionada con la economía de su diseño. Mientras que los robots tradicionales presentan unos costes elevados debido a la necesidad del empleo de motores de corriente continua o alterna, que en ocasiones conllevan además el uso de batería externas ampliamente contaminantes, los robots blandos emplean materiales mucho más asequibles, así como métodos de actuación más económicos, como la neumática. Esto favorece a su utilización en la industria en el futuro.

Por último, cabe destacar además que la industria de la robótica blanda se encuentra actualmente en proceso de desarrollo, por lo que cualquier contribución en este ámbito ayuda a acelerar el proceso de crecimiento del número de este tipo de robot en la industria. Es por ello que se puede considerar además que este proyecto contribuye con el cumplimiento del **ODS 9: Industria, innovación e infraestructura**.

A.1.2. Impacto ético

En lo referente a la ética del trabajo realizado, no se considera nigún aspecto que vaya en contra de los valores éticos presentes en la sociedad.

El proyecto aquí expuesto no vulnera ningún derecho fundamental de la humanidad, no atenta contra la voluntad de los seres humanos ni manifiesta la intención de violar la integridad física de las personas. Es por ello que no se considera ningún impacto ético negativo en la sociedad.

Si se tienen en cuenta los impacto positivos, se puede reiterar el beneficio social que este proyecto podría aportar a la sociedad en un futuro, como ya se ha visto en el apartado anterior.

A.1.3. Impacto legal

Finalmente, como ya se ha comentado en el punto A.1, este trabajo cumple con la normativa vigente de uso de dispositivos electrónicos promulgada en el Real Decreto 993/2022 del 16 de enero de 2023.

A su vez, todos los materiales empleados proceden de comercios y/o fabricantes legales, como Feroca, que cumplen con toda la legislación referente a la fabricación, manejo, transporte y venta de productos no biodegradables. Asimismo, se ha seguido la recomendación del fabricante de eliminación de los residuos generados tras la manipulación de los productos y materiales adquiridos.

A.1.4. Conclusión: ODS

Como resumen final, en la figura A.1 se pueden ver marcados con un círculo los ODS a los que contribuye este proyecto.

Figura A.1 ODS a los que contribuye este proyecto. *Fuente: elaboración propia*

Apéndice B

Planificación temporal y presupuesto

B.1. Estructura de Descomposición del Proyecto

En la figura B.1, se muestran las actividades comprendidas dentro del proyecto.

Figura B.1 Estructura de Descomposición del Proyecto. *Fuente: elaboración propia*

B.2. Planificación

En el diagrama de Gantt de la figura B.2, se traducen a tiempos las diferentes actividades definidas en la EDP. El tiempo de realización del proyecto fue de aproximadamente 345 horas, que se corresponde con el tiempo esperado en un Trabajo Fin de Grado.

Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
Comienzo del proyecto	1 día	lun 12/12/22	lun 12/12/22	
Definición del proyecto	4 días	mar 13/12/2	vie 16/12/22	1
Investigación y lectura de la literatura	30 días	mar 13/12/22	lun 23/01/23	1
Sensorización	**54 días**	**mar 24/01/2**	**vie 07/04/23**	
Selección de los sensores	2 días	mar 24/01/23	mié 25/01/23	3
Gestión de compras	7 días	jue 26/01/23	vie 03/02/23	5
Desarrollo de la técnica de medición	**12 días**	**jue 02/02/23**	**vie 17/02/23**	
Selección del hardwa	5 días	jue 02/02/23	mié 08/02/2	5
Gestión de compras	7 días	jue 09/02/23	vie 17/02/23	8
Diseño del método de medición	3 días	jue 09/02/23	lun 13/02/23	8
Construcción del banco de pruebas	**18 días**	**jue 09/02/23**	**lun 06/03/23**	
Diseño del banco de pruebas	3 días	jue 09/02/23	lun 13/02/23	8
Fabricación e impresión de sus	7 días	lun 20/02/23	mar 28/02/23	12;9
Programación	3 días	mar 14/02/2	jue 16/02/23	12
Pruebas de verificación de	4 días	mié 01/03/23	lun 06/03/23	13;14
Caracterización del comportamiento de los sensores	5 días	mar 07/03/23	lun 13/03/23	15
Implementación en el robot	**47 días**	**jue 02/02/23**	**vie 07/04/23**	
Fabricación de nuevos segmentos	7 días	jue 02/02/23	vie 10/02/23	3
Selección del método de fijación de los sensores y	7 días	mar 14/03/23	mié 22/03/23	5;16
Montaje final y prueba de	12 días	jue 23/03/23	vie 07/04/23	19
Cambios realizados al rot	**63 días**	**vie 21/04/23**	**mar 18/07/2**	
Identificación de problemas	55 días	vie 21/04/23	jue 06/07/23	28
Cambios en el proceso de fabricación	59 días	vie 21/04/23	mié 12/07/23	28
Cambios en el diseño	57 días	vie 21/04/23	lun 10/07/23	28
Verificación de funcionamiento	4 días	jue 13/07/23	mar 18/07/23	23;24
Desarrollo del sistema de control	**77 días**	**lun 10/04/23**	**mar 25/07/23**	
Lectura de código previo y formación en el uso del sistema de	4 días	lun 10/04/23	jue 13/04/23	20
Diseño inicial	5 días	vie 14/04/23	jue 20/04/23	27
Diseño: iteraciones y pruebas	25 días	vie 21/04/23	jue 25/05/23	28
Adquisición de datos	**66 días**	**vie 14/04/23**	**vie 14/07/23**	
Programación	14 días	vie 14/04/23	mié 03/05/2	27
Pruebas	4 días	jue 04/05/23	mar 09/05/2	31
Iteraciones de programación y	48 días	mié 10/05/23	vie 14/07/23	32
Entrenamiento de redes neuronales	**43 días**	**vie 26/05/23**	**mar 25/07/23**	
Diseño de las redes	2 días	vie 26/05/23	lun 29/05/23	29
Entrenamiento de las redes	3 días	mié 19/07/23	vie 21/07/23	35;33;25
Validación de errore	2 días	lun 24/07/23	mar 25/07/2	36
Puesta en servicio	1 día	mié 26/07/2	mié 26/07/2	37
Pruebas de funcionamiento y	2 días	jue 27/07/23	vie 28/07/23	38
Redacción	25 días	lun 31/07/23	vie 01/09/23	39
Fin del proyecto	0 días	lun 04/09/23	lun 04/09/23	40

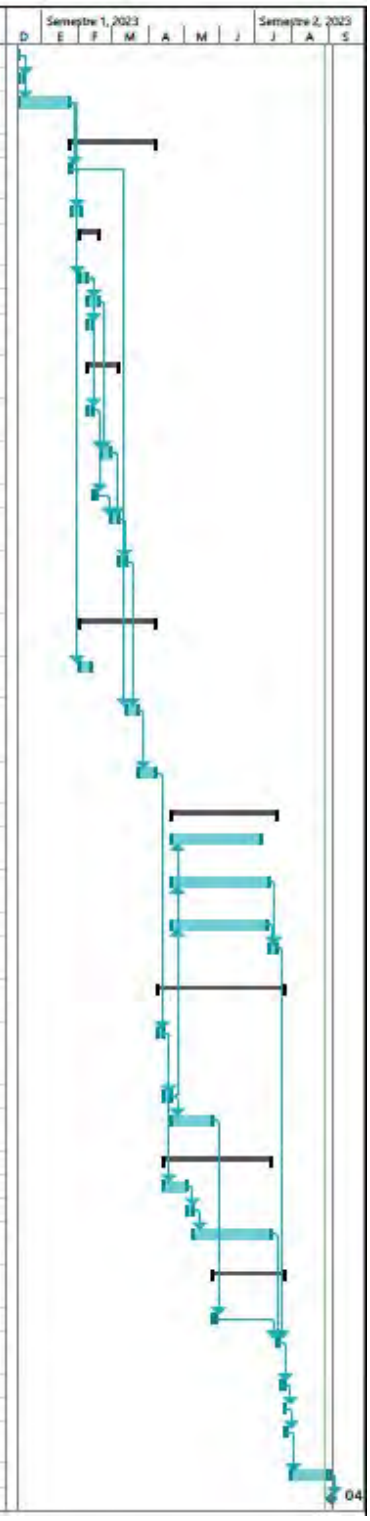


Figura B.2 Cronograma del proyecto. *Fuente: elaboración propia*

B.3. Presupuesto

Se presenta a continuación el presupuesto total del proyecto desglosado en partidas **material**, **de amortización** y **de personal**.

B.3.1. Coste material

En este apartado se incluyen todos aquellos gastos de material no amortizable de vida útil inferior a un año. También se incluyen aquellos elementos que precisarán mantenimiento o modificaciones en los futuros proyectos relacionados.

Cabe destacar que se han incluido partidas ya recogidas en el Trabajo Fin de Grado de Adrián Rieker [1], pero necesarias para este trabajo, como el banco de actuación. Es por ello que estas partidas no han sido desglosadas y se ha incluido su precio tal y como figura en dicho proyecto.

Concepto	Unidades	Precio unitario (€/u)	Importe total (€)	Tipo
Robot				
Silicona				
Silicona TinSil 80-15	3	34,97	104,91	Fungible
Sensores				
Conductive Rubber Cord Stretch Sensor de Adafruit (1m)	2	9,21	18,42	Inventariable
Sellador de silicona DOWSIL 732 (90ml)	3	21,78	65,34	Fungible
Componentes electrónicos				
Arduino Due	1	51,24	51,24	Inventariable
INA3221	1	6,21	6,21	Inventariable
Fuente de alimentación 12V/500W	1	16,99	16,99	Inventariable
Juego de borneras (50pcs)	1	9,87	9,87	Inventariable
Juego de pinzas de cocodrilo (10pcs)	1	5,75	5,75	Inventariable
Fabricación				
Cera de parafina	5	2	10	Fungible
Guantes de látex (caja de 100 unidades)	1	11,49	11,49	Fungible
Banco de actuación	1	363,96	363,96	Inventariable
Banco de pruebas				
Construcción y montaje				
Tablero aglomerado de 4 cantos azabache 39,7x80x1,6cm	1	10,99	10,99	Fungible
Componentes electrónicos				
Motor paso a paso Nema 17	1	9,99	9,99	Inventariable
Driver DRV8825	1	6,49	6,49	Inventariable
Material común				
Tornillería	1	21,99	21,99	Inventariable
Juego de cables dupont (120pcs)	1	6,99	6,99	Inventariable
Impresión 3D				
PLA 3DFILS 1.75mm 1kg	2	14	28	Fungible
		SUBTOTAL	**748,63**	

Tabla B.1 Coste de material

B.3.2. Coste de amortización

A continuación se presenta el presupuesto de la partida de amortización. Este caso únicamente se ha incluido el coste del ordenador portátil con el que se ha hecho el proyecto

y el coste de la licencia de MATLAB.

Concepto	Coste total (€)	Años de amortización lineal	Importe final (€)
Ordenador portátil Lenovo ideapad s540-15iwl gtx	1100	5	220
Licencia de estudiante MATLAB	69	4	17,25
		SUBTOTAL	**237,25**

Tabla B.2 Coste de amortización del proyecto

B.3.3. Coste de personal

Por último, en el cuadro B.3 se presenta el coste de personal que ha participado en la ejecución de este proyecto.

Importante señalar que se ha considerado que el coste horario de un ingeniero industrial técnico en España ronda los 30€ de media para los recién titulados [44]. Teniendo en cuenta que este proyecto habilita a dicha posición profesional, se ha tenido en cuenta un cose horario de 20€. Por otro lado, también se ha estimado que el coste horario de un ingeniero industrial superior ronda los 37€ de media, y que el de un catedrático de universidad experimentado se encuentra en los 50€. Estos han sido las consideraciones de coste-hora de los participantes en el proyecto.

Concepto	Unidades	Precio unitario (€/u)	Importe total (€)
Horas de trabajo autor	345	20	6.900
Horas de trabajo tutor principal	20	50	1.000
Horas de trabajo cotutor	60	37	2.220
		SUBTOTAL	**10.120**

Tabla B.3 Coste de personal colaborador del proyecto

B.3.4. Presupuesto final

Se muestra en el cuadro B.4 el presupuesto total del proyecto.

Partida	Concepto	Importe total (€)
1	**Coste de material**	748,63
2	**Coste amortizable**	237,25
3	**Coste de personal**	10.120
	TOTAL	**11.105,88**

Tabla B.4 Presupuesto total del proyecto

Apéndice C

Código de interés

```
%% Measuring the sensors
function Measure(this)
    % Robot.Measure() sends the order to the microcontroller to
    % measure the sensors' values

    % Sending measure order
    writeline(this.serialDevice, "M");
end

function CallbackMeasurement(this)
    % Callback associated to the mesure

    measurement = this.serialData;
    measurement = split(measurement);
    measurement = str2double(measurement);
    measurement = measurement(2:1+this.nSensors);
    this.voltages(:,end+1) = measurement;
end

%% Kinematic modelling
function [l, params] = MCI(this, xP, phi0, a)
    % Calculate the inverse kinematic model of a three-wire robot
    % using the PCC method.
    %
    % l = Robot.MCI() returns the lengths of the three wires of a
    % robot, knowing the position of its end (x) and the diameter
    % of the circumference they form (a). Orientation may or may
    % not be included in x.
    %
    % [l, params] = Robot.MCI() returns, in addition to the lengths
    % of the wires, a structure with the values of lr (average
```

```
32      % length), phi (orientation) and kappa (curvature)
33      %
34      % l = Robot.MCI(phi0) allows to rotate, counter-clockwise, an
35      % angle phi0 the robot reference system.
36
37      % Initial setup
38      switch nargin
39          case 1
40              xP = this.x(1:3);
41              phi0 = pi/2;
42              a = this.geom.phi0;
43
44          case 2
45              phi0 = this.geom.phi0;
46              a = this.geom.radius;
47
48          case 3
49              a = this.geom.radius;
50
51      end
52
53      % Dependent modelling
54      % General case
55      if (sum(xP(1:2) - [0 0]))
56          phi = atan2(xP(2),xP(1));
57          kappa = 2 * norm(xP(1:2)) / norm(xP)^2;
58          if xP(3) <= 0
59              theta = acos(1 - kappa * norm(xP(1:2)));
60          else
61              theta = 2*pi - acos(1 - kappa * norm(xP(1:2)));
62          end
63          theta2 = wrapToPi(theta);
64          lr = abs(theta2 / kappa);
65
66      % Singular configuration
67      else
68          phi = 0;     % Cualquier valor es posible
69          kappa = 0;
70          lr = xP(3);
71      end
72      params.lr = lr;
73      params.phi = phi;
74      params.kappa = kappa;
75
76      %% Independent modelling
77      phi_i = phi0 + [pi pi/3 -pi/3];
78      l = lr * (1 + kappa*a*sin(phi + phi_i));
```

```
79      this.l = l;
80  end
81
82  function [T, params] = MCD(this, l, a)
83      % Calcula el modelo cinem tico directo de un robot de tres cables
84      % utilizando el m todo PCC.
85      %
86      % T = MCD(l, a) devuelve la matriz de transformaci n homog nea que
87      % permite pasar de la base al extremo del robot, conocidas las
88      % longtiudes de sus cables (l) y el di metro de la circunferencia
89      % que forman (a).
90      %
91      % [T, params] = MCD(l, a) devuelve, adem s de la matriz de
92      % transformaci n homog nea , una estructura con los valores de
93      % lr (longitud media), phi(orientaci n) y kappa (curvatura).
94
95      % Comprobaciones iniciales
96      if length(l) ~= 3
97          error("Introduce un vector de tres longitudes")
98      end
99      if nargin == 2
100         a = this.geom.radius;
101     end
102
103     % Modelado dependiente
104     % Caso general
105     if ~all(l == l(1))
106         lr = mean(l);
107         phi = atan2(sqrt(3) * (-2*l(1) + l(2) + l(3)), 3 * (l(2) - l(3))
    );
108         kappa = 2 * sqrt(l(1)^2 + l(2)^2 + l(3)^2 - l(1)*l(2) - ...
109             l(3)*l(2) - l(1)*l(3)) / a / (l(1) + l(2) + l(3));
110     % Posici n singular
111     else
112         lr = l(1);
113         phi = 0;
114         kappa = 0;
115     end
116     params.lr = lr;
117     params.phi = phi;
118     params.kappa = kappa;
119
120     % Modelado independiente
121     if ~all(l == l(1))
122         Trot = [cos(phi) -sin(phi) 0 0;
123             sin(phi) cos(phi) 0 0; 0 0 1 0; 0 0 0 1];
124         Tarc = [cos(kappa*lr) 0 sin(kappa*lr) (1-cos(kappa*lr))/kappa;
```

```
125             0 1 0 0; -sin(kappa*lr) 0 cos(kappa*lr) sin(kappa*lr)/kappa;
126             0 0 0 1];
127         T = Trot*Tarc;
128     else
129         T = [1 0 0 0; 0 1 0 0; 0 0 1 lr; 0 0 0 1];
130     end
131
132 end
133
134 %% Neural Network
135 function [perform_pt, perform_vt] = NN_training(this, pos, volt, ...
136     tiempo, capas_pt, capas_vt)
137     % [perform_pt, perform_vt] = Robot.NN_training(pos, volt,
138     % tiempo, capas_pt, capas_vt) trains and creates the two
139     % required neuronal networks for the control system of the
140     % robot
141
142     n = fix(0.95*length(pos));
143
144     this.net_pt = feedforwardnet(capas_pt);
145     this.net_pt = train(this.net_pt,pos(1:n,:)',tiempo(1:n,:)');
146
147     out_pt = this.net_pt(pos(n+1:end,:)');
148     perform_pt = perf(this.net_pt,tiempo(n+1:end,:)',out_pt);
149
150     this.net_vt = feedforwardnet(capas_vt);
151     this.net_vt = train(this.net_vt,volt(1:n,:)',tiempo(1:n,:)');
152
153     out_vt = this.net_vt(volt(n+1:end,:)');
154     perform_vt = perf(this.net_vt,tiempo(n+1:end,:)',out_vt);
155 end
156
157 function NN_creation(this, network_pt, network_vt)
158     % Robot.NN_creation(network_pt, network_vt) creates and
159     % storages the already created neural networks
160
161     this.net_pt = network_pt;
162     this.net_vt = network_vt;
163 end
164
165 %% Control of a single segment
166 function [pos_final, error_pos] = Move(this, x)
167     % [pos_final, error_pos] = Robot.Move(x) moves the robot to an
168     % specified point (x) in the workspace of the robot
169
170     niter = 0;
171     action = zeros(1,3);
```

```
172     err = [900 900 900];
173     max_accion = 300;
174     toler = 40;
175 
176     if length(x) ~= 3
177         errordlg("Introduce un punto en el espacio " + ...
178             "(vector fila de 3 componentes)","Execution Error");
179         return
180     end
181 
182     t_obj = this.net_pt(x');
183 
184     while (abs(err(1)) > toler || abs(err(2)) > toler || abs(err(3)) ...
185             > toler) && niter < 10
186         niter = niter + 1;
187         this.Measure();
188         pause(0.1)
189         vol_current = this.getVoltages();
190 
191         t_current = this.net_vt(vol_current);
192         err = t_obj - t_current;
193         pause(0.1)
194 
195         for i = 1:3
196             if err(i) > 0
197                 action(i) = min(err(i), max_accion);
198             else
199                 action(i) = max(err(i), -max_accion);
200             end
201         end
202 
203         this.WriteSegmentMillis(action);
204         pause(0.5)
205     end
206     pos_final_raw = this.CapturePosition;
207     pos_final = pos_final_raw(2,:);
208     error_pos = norm(pos_final - x);
209 end
```

Listing C.1 Métodos más significativos para el control de la clase Robot

```
1 MAX_MILLIS = 900;
2 N_ITER_MAX = 1050;
3 SAVE = 100;
4 pos = zeros(N_ITER_MAX,3);
5 vol = zeros(N_ITER_MAX,3);
6 t = zeros(N_ITER_MAX,3);
7 
```

```
R.Measure();
pause(0.1)
pos_raw = R.CapturePosition();
vol_raw = R.getVoltages();
pos(1,:) = pos_raw(2,:);
vol(1,:) = vol_raw';

for i = 1:N_ITER_MAX

    while 1

        t(i,:) = -50 + 50*randi(fix((MAX_MILLIS/50 + 1)), [1 3]);

        indice = find(t(i,:) == min(t(i,:)));

        if (t(i,indice(1)) ~= 0) && (isempty(find(t(i,:) == 0, 1)))
            t(i,indice(1)) = 0;
        end

        if isempty(find(ismember(t(i,:),t(1:i-1,:),'rows') == 1,1))
            break;
        end
    end

    R.WriteSegmentMillis(t(i,:));
    pause(2 * MAX_MILLIS / 1000)
    R.Measure();
    pause(0.1)
    pos_raw = R.CapturePosition();
    vol_raw = R.getVoltages();
    pos(i,:) = pos_raw(2,:);
    vol(i,:) = vol_raw';

    if ~mod(i,SAVE)
        save(strcat('DatasetNN/prueba_',num2str(i/SAVE)),'pos','vol','t'
   );
    end

    R.Deflate();
    pause(0.5)
end
```

Listing C.2 Código de obtención de datos final