

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](http://www.industriales.upm.es)



POLITÉCNICA

INDUSTRIALES

05 TRABAJO FIN DE GRADO

Adrian Siegbert Rieker González



TRABAJO FIN DE GRADO

## Diseño fabricación y control de un robot blando neumático



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

SEPTIEMBRE 2014

**Adrián Siegbert Rieker  
González**

DIRECTOR DEL TRABAJO FIN DE GRADO:  
**Antonio Barrientos Cruz**



## **AGRADECIMIENTOS**

Agradezco a ...

Gracias a ...

A ... por ...



## RESUMEN EJECUTIVO

Este trabajo se engloba dentro del campo del conocimiento de la robótica flexible. Esta es una nueva rama de la robótica que apareció hace aproximadamente diez años y actualmente está en auge, publicándose cada vez más al respecto.

Los robots de este tipo, a los que comúnmente se denomina como *SoftBots*, tienen una característica fundamental que los diferencia de los robots clásicos que llevan desarrollándose desde los años sesenta. Esta es que se pasa de una cadena cinemática sólida a una flexible, con todo lo que ello conlleva. Debido al poco tiempo que ha tenido esta rama de la robótica para desarrollarse, todavía hay muchos temas que investigar. Desde nuevos materiales, procesos de fabricación hasta nuevos y sofisticados métodos de control.

Esta nueva clase de robots se consiguen actualmente mediante el empleo de distintos tipos de siliconas que se unen a otros materiales que permiten la actuación, como pueden ser los muelles de materiales con memoria de forma o cámaras neumáticas hinchables.

El empleo de materiales flexibles confieren a los *SoftBots* distintas capacidades que los convierten en interesantes alternativas frente a los robots tradicionales basados en cadenas cinemáticas rígidas. Una de estas es una alta posibilidad de interacción con el usuario, gracias a un extremadamente bajo riesgo en caso de impacto. Esto los diferencia de los robots industriales tradicionales, que en la actualidad todavía necesitan de espacios y perímetros de seguridad para poder operar, mermando el aumento en productiva que podría alcanzarse en condiciones de mayor cooperación entre máquinas y operarios. Lo mismo puede aplicar en campos como la medicina o la producción alimentaria, donde es necesario un estricto control ambiental. En estos escenarios, muchos robots blandos pueden ser una mejor opción frente a un robot tradicional al no disponer de elementos que puedan ser fuente de contaminantes como por ejemplo motores o distintos elementos mecánicos. Además, la estructura flexible que caracteriza a los *SoftBots*, les permite adaptarse de una forma totalmente nueva a escenarios desconocidos, al poder alterar su propia geometría acorde a las circunstancias.

Motivado por ello, este trabajo fin de grado comprende el diseño, fabricación y control de un nuevo robot blando que utiliza la actuación neumática. A continuación, en la figura 0.1, puede observarse el resultado de este proyecto:

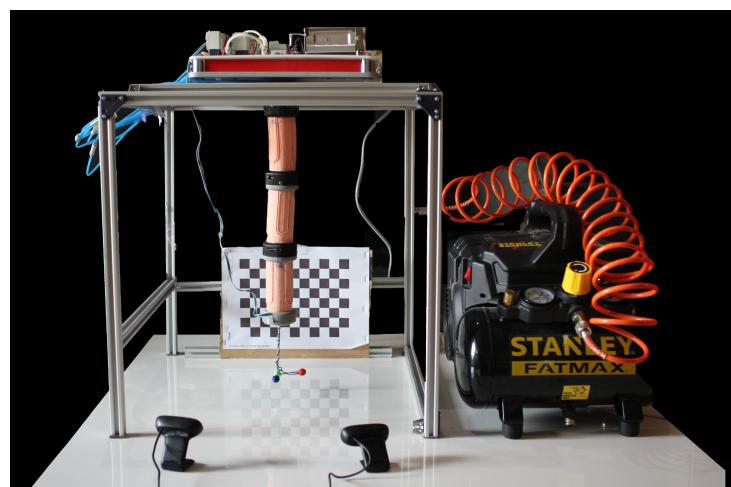


Figura 0.1: Robot final desarrollado, junto al sistema de actuación y de adquisición de datos.  
Fuente: Elaboración propia

El objetivo es desarrollar un *SoftBot* con morfología de tubo continuo, que disponga de nueve grados de libertad controlables de forma independiente. Este debe ser capaz de moverse en las tres direcciones espaciales y realizar tareas de manipulación.

Para la elaboración física del robot se ha empleado silicona de estaño y un método de fabricación basado en el moldeo a la cera perdida. Este método de fabricación, así como el diseño final del robot, son fruto de numerosas iteraciones que han servido para perfeccionar el diseño inicial y encontrar el proceso más óptimo para fabricarlo. Los moldes se han fabricado mediante impresión 3D en PLA, y posteriormente se han rellenado con la silicona para obtener el modelo deseado. Para conformar los huecos interiores, a los que se ha denominado *vejigas* en el contexto de este trabajo, se han realizado mediante modelo positivo de cera de parafina que se extrae en fase líquida una vez solidificada la silicona. Se han ensayado distintos tipos de siliconas y materiales para el conformado del robot. Para el diseño CAD se ha utilizado la herramienta *Autodesk Fusion 360* y se ha realizado mediante un fichero de diseño paramétrico que se puede observar en la siguiente figura:

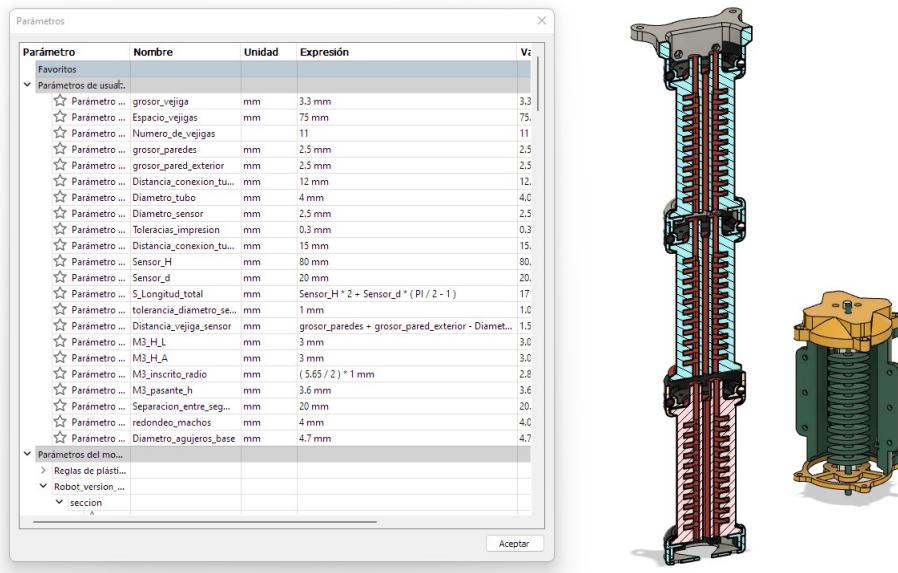


Figura 0.2: Diseño final del robot junto al molde y la lista de parámetros utilizada. Fuente: *Elaboración propia*

Estas vejigas están muy relacionadas con el método de actuación que se ha escogido. Este ha sido la actuación neumática. De esta manera, la fuerza que deformará el robot en las direcciones deseadas proviene del inflado y desinflado de estas vejigas presentes en su interior. Esta presión de deshinflado/ inflado, proviene de nueve líneas neumáticas (una por grado de libertad), que se pueden controlar mediante un banco de actuación de forma independiente. Este banco de actuación, inicialmente ideado por Victor Barroso [13], permite controlar el caudal de aire comprimido proveniente de un compresor, en nueve líneas neumáticas, mediante un conjunto de nueve válvulas 2/2 y nueve válvulas 3/2. Uno de los objetivos de este proyecto ha sido la mejora de dicho banco de actuación acorde a las necesidades del robot. De esta manera se han ampliado las iniciales seis líneas neumáticas a nueve, se han sustituido los relés que hacían de interfaz de potencia por mosfets y se ha programado una interfaz software a bajo nivel que realizará la comunicación con un ordenador. Las señales de control que se han utilizado para controlar los grados de libertad del robot son los tiempos de hinchado o desinchado de las válvulas. Debido a que la presión y caudal proporcionados por el compresor son significativamente mayores al consumido por el robot, es óptimo el control mediante tiempos. En la figura 0.3, puede observarse

el banco de actuación, una vez introducidas las modificaciones necesarias.

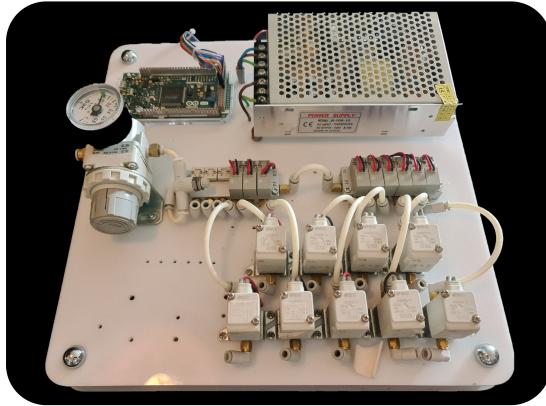


Figura 0.3: Banco de actuación mejorado. *Fuente: Elaboración propia*

Respecto a la parte de control, en este trabajo se ha optado por un método basado en el análisis de conjuntos de datos que se toman de forma empírica. Estos datos consisten en tablas que relacionan la posición y orientación que alcanza el extremo del robot, con el conjunto de presiones que hay en las vejigas en dicho momento. Para recoger estos datos, se ha ideado un sistema automatizado que hace uso de un sistema de adquisición de datos, basado en visión por computador, que también se ha desarrollado en este proyecto. Este sistema, utilizando dos cámaras rgb en configuración de estereovisión, permiten captar la orientación y posición de una baliza luminosa respecto a un sistema fijo que se establece previamente.

Todos los desarrollos software que se han realizado a lo largo de este trabajo se han unificado en un único software de control con interfaz gráfica (figura 0.4), para hacer más amigable el robot a futuros proyectos. Este software permite controlar el banco de actuación a bajo nivel, realizar la captura de datos mediante visión por computador, realizar el proceso de toma de datos de forma automática y controlar en posición y orientación el extremo del robot.

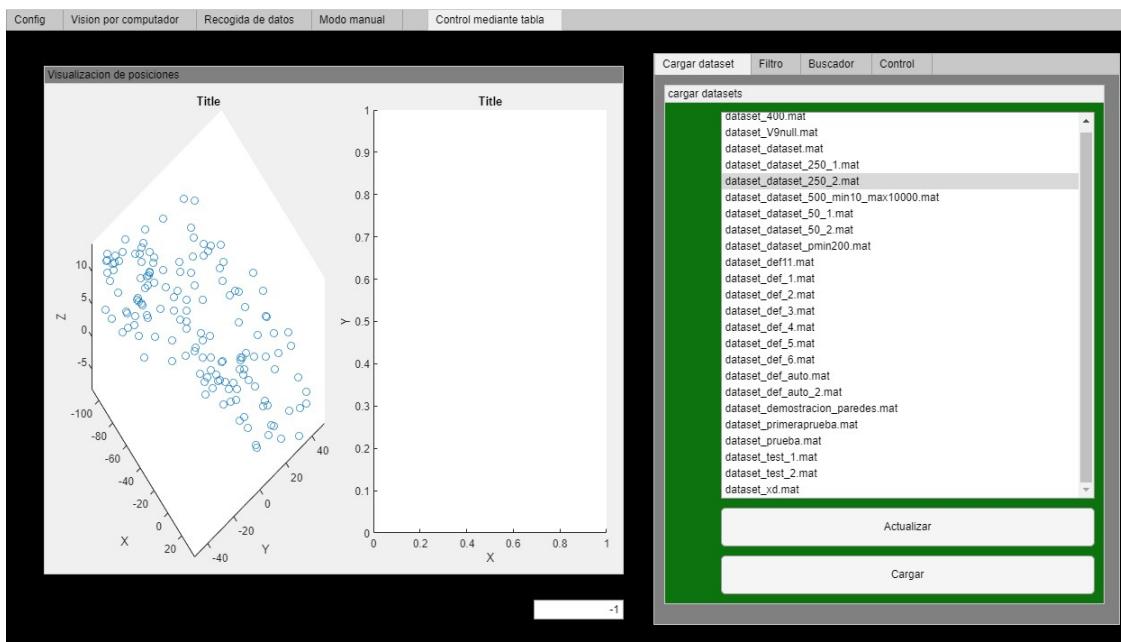


Figura 0.4: Software desarrollado. *Fuente: Elaboración propia*

Una vez toda terminadas todas las fases de las que dispone este proyecto, se han realizado

## **RESUMEN EJECUTIVO**

---

los ensayos necesarios para verificar que el prototipo realizado cumple con las especificaciones propuestas.

# ÍNDICE

<b>AGRADECIMIENTOS</b>	<b>I</b>
<b>RESUMEN EJECUTIVO</b>	<b>III</b>
<b>ÍNDICE DE TABLAS</b>	<b>xiii</b>
<b>ÍNDICE DE FIGURAS</b>	<b>xvii</b>
<b>1. Introducción</b>	<b>1</b>
1.1. Motivación . . . . .	1
1.2. Estado actual . . . . .	2
1.3. Objetivo del trabajo fin de grado . . . . .	2
1.4. Estructura de la memoria . . . . .	3
<b>2. Estado del arte</b>	<b>4</b>
2.1. Contexto . . . . .	4
2.2. Tipos de actuadores . . . . .	6
2.2.1. Mecanismos de actuación intrínseca . . . . .	6
2.2.1.1. Neumáticos . . . . .	7
2.2.1.2. Polímeros electroactivos . . . . .	8
2.2.1.3. Magnéticos . . . . .	10
2.2.2. Mecanismos de actuación extrínseca . . . . .	10
2.2.2.1. Materiales con memoria de forma (SMA) . . . . .	10
2.2.2.2. Twisted - Coiled Actuators (TCAs) . . . . .	11
2.2.2.3. Actuado por cables . . . . .	13
2.3. Control de softbots . . . . .	14
2.3.1. Modelado mediante elementos finitos . . . . .	14
2.3.2. Control basado en curvatura constante . . . . .	15
2.3.3. Entrenamiento mediante red neuronal . . . . .	16

2.4. Aplicaciones . . . . .	16
<b>3. Diseño y fabricación</b>	<b>18</b>
3.1. Método de fabricación y materiales empleados . . . . .	18
3.1.1. Justificación de la elección de la silicona . . . . .	20
3.2. Diseño y fabricación de los módulos . . . . .	21
3.2.1. Caracterización de los sensores . . . . .	22
3.2.2. Módulos: primera versión . . . . .	22
3.2.2.1. Diseño . . . . .	22
3.2.2.2. Conformado . . . . .	25
3.2.2.3. Ensayos y conclusiones . . . . .	26
3.2.3. Módulos: segunda versión . . . . .	27
3.2.3.1. Cambios en el diseño . . . . .	27
3.2.3.2. Conformado . . . . .	28
3.2.3.3. Ensayos y conclusiones . . . . .	28
3.2.4. Módulos: tercera version . . . . .	29
3.2.4.1. Cambios de diseño en los módulos . . . . .	29
3.2.4.2. Diseño de nuevos elementos . . . . .	30
3.2.4.3. Conformado y montaje . . . . .	31
3.2.4.4. Ensayos y conclusiones . . . . .	32
3.2.5. Version final . . . . .	33
3.2.5.1. Cambios de diseno . . . . .	33
3.2.5.2. Conformado y montaje . . . . .	34
3.2.5.3. Ensayos y conclusiones . . . . .	38
<b>4. Banco de actuacion</b>	<b>39</b>
4.1. Punto de partida y objetivos . . . . .	39
4.2. Cambios introducidos . . . . .	40
4.2.1. Dimensionamiento de elementos neumáticos . . . . .	41
4.2.2. Dimensionamiento de elementos electrónicos . . . . .	42

4.2.3. Pinout final . . . . .	44
4.3. Arquitectura software . . . . .	46
4.3.1. Explicación conceptual . . . . .	46
4.3.1.1. La clase Valvula . . . . .	46
4.3.1.2. Comunicación UART . . . . .	48
4.3.1.3. Otras tareas del programa . . . . .	49
4.3.2. Resumen de ficheros y funciones . . . . .	49
<b>5. Sistema de adquisición de datos</b>	<b>51</b>
5.1. Hardware . . . . .	51
5.1.1. Utillaje para el robot . . . . .	51
5.1.2. Tabla de calibración . . . . .	51
5.1.3. Cámaras . . . . .	52
5.1.4. Baliza luminosa . . . . .	52
5.2. Visión por computador . . . . .	53
5.2.1. Calibración de cámaras . . . . .	54
5.2.2. Reconocimiento del objeto en la imagen . . . . .	56
5.2.2.1. Color Threshold . . . . .	56
5.2.2.2. MSER regions . . . . .	58
5.2.2.3. Integración: función findpoint.m . . . . .	59
5.2.3. Posicionamiento en el espacio: triangulación estéreo . . . . .	60
5.3. Implementación en Matlab . . . . .	62
<b>6. Diseño experimental y recogida de datos</b>	<b>66</b>
6.1. Introducción . . . . .	66
6.2. Datos a recoger y condiciones experimentales . . . . .	66
6.2.1. Datos a recojer . . . . .	66
6.2.2. Condiciones experimentales . . . . .	68
6.3. Procedimiento . . . . .	70
6.4. Filtros . . . . .	72

<b>7. Interfaz de usuario y control</b>	<b>73</b>
7.1. Control . . . . .	73
7.2. Interfaz de usuario: SoftControl . . . . .	74
7.2.1. Estructura general . . . . .	74
7.2.2. Configuración . . . . .	75
7.2.3. Visión por computador . . . . .	78
7.2.4. Modo manual . . . . .	80
7.2.4.1. Control porcentual . . . . .	81
7.2.4.2. Control curvatura - ángulo . . . . .	82
7.2.4.3. Control directo . . . . .	84
7.2.5. Recogida de datos . . . . .	85
7.2.5.1. Procedimiento de uso . . . . .	88
7.2.5.2. Funcionamiento interno . . . . .	89
7.2.6. Control mediante tabla . . . . .	89
7.2.6.1. Carga de un dataset . . . . .	90
7.2.6.2. Filtrado del dataset . . . . .	90
7.2.6.3. Selección de los puntos de interés . . . . .	91
7.2.6.4. Control del robot entre los puntos de interés buscados . . . . .	93
<b>8. Experimentos y resultados</b>	<b>95</b>
8.0.0.1. Experimentos . . . . .	95
8.0.0.2. Resultados . . . . .	98
<b>9. Sostenibilidad e impacto social</b>	<b>99</b>
<b>10. Planificación y presupuesto</b>	<b>101</b>
10.1. Planificación . . . . .	101
10.1.1. Estructura de descomposición de proyecto . . . . .	102
10.1.2. Diagrama de Gantt . . . . .	103
10.2. Presupuesto . . . . .	104

<b>11. Conclusiones y trabajos futuros</b>	<b>106</b>
11.1. Conclusiones . . . . .	106
11.2. Líneas futuras . . . . .	107
<b>ANEXOS</b>	<b>112</b>
C. Primer anexo: Caracterización de sensores elásticos . . . . .	112
D. Segundo anexo . . . . .	114



## ÍNDICE DE TABLAS

3.1. Siliconas usadas en el proyecto . . . . .	21
3.2. Parámetros de impresión utilizados . . . . .	21
4.1. Banco neumático: especificaciones iniciales . . . . .	40
4.2. Especificaciones iniciales del banco de actuación . . . . .	40
4.3. Cálculo de corrientes en banco de actuación . . . . .	43
4.4. Pinout . . . . .	45
5.1. Cámaras utilizadas . . . . .	52
8.1. Tiempo que tardan en recogerse distintos datasets. <i>Fuente: elaboración propia</i> . .	95

## ÍNDICE DE FIGURAS

0.1.	Robot final desarrollado, junto al sistema de actuación y de adquisición de datos. <i>Fuente: Elaboración propia</i>	III
0.2.	Diseño final del robot junto al molde y la lista de parámetros utilizada. <i>Fuente: Elaboración propia</i>	IV
0.3.	Banco de actuación mejorado. <i>Fuente: Elaboración propia</i>	V
0.4.	Software desarrollado. <i>Fuente: Elaboración propia</i>	V
2.1.	Cadena cinemática de un robot serie. <i>Fuente: UDLAP</i>	4
2.2.	Robots manipuladores tradicionales serie y paralelo	4
2.3.	Robot blando caminante. <i>Fuente: [3]</i>	5
2.4.	Clasificación de soft bots. <i>Fuente: [20]</i>	5
2.5.	Robot manipulador blando. <i>Fuente: [14]</i>	6
2.6.	Comparación entre robots blandos y rígidos. <i>Fuente: [23]</i>	6
2.7.	Actuación neumática. <i>Fuente: [2]</i>	7
2.8.	Ejemplos actuación neumática	8
2.9.	Músculos de McKibben. <i>Fuente: [21]</i>	8
2.10.	Clasificación de los polímeros electroactivos. <i>Fuente: [4]</i>	9
2.11.	Elastómeros dieléctricos. <i>Fuente: [9]</i>	9
2.12.	Actuación magnética. <i>Fuente: [19]</i>	10
2.13.	Materiales con memoria de forma	11
2.14.	Robot blando actuado mediante SMA. <i>Fuente: [11]</i>	11
2.15.	Fabricación de twisted-coiled actuators. <i>Fuente: [16]</i>	12
2.16.	Robot basado en TCA. <i>Fuente: [17]</i>	13
2.17.	Robot blando actuado por cables para aplicaciones quirúrgicas. <i>Fuente: [27]</i>	13
2.18.	Elementos finitos - control en tiempo real	15
2.19.	Elementos finitos - control en tiempo real	15
2.20.	Control basado en curvatura constante	15
2.21.	Herramienta basada en robótica blanda. <i>Fuente: [6]</i>	17
2.22.	Festo BionicSoftArm. <i>Fuente: [15]</i>	17

3.1. Detalle de las vejigas. <i>Fuente: elaboración propia</i>	19
3.2. Método de fabricación empleado. <i>Fuente: elaboración propia</i>	19
3.3. Impresora 3D: Artillery Genius. <i>Fuente: Artillery</i>	20
3.4. Escala Shore. <i>Fuente: recreus.com</i>	20
3.5. Bendlabs digital Flex sensor. <i>Fuente: Bendlabs</i>	22
3.6. Segmentos V1. Sección básica. <i>Fuente: elaboración propia</i>	23
3.7. Segmentos V1. Sección. <i>Fuente: elaboración propia</i>	23
3.8. Segmentos V1. Moldes. <i>Fuente: elaboración propia</i>	24
3.9. Molde. <i>Fuente: elaboración propia</i>	25
3.10. Segmento V1. <i>Fuente: elaboración propia</i>	26
3.11. Comparación entre el molde de la primera versión y la segunda. <i>Fuente: elaboración propia</i>	27
3.12. V2: Molde. <i>Fuente: elaboración propia</i>	28
3.13. Lista de parámetros principales junto al diseño del robot y su molde. <i>Fuente: elaboración propia</i>	29
3.14. Diseño V3. Cambios en los módulos. <i>Fuente: elaboración propia</i>	30
3.15. Uniones entre segmentos	30
3.16. Detalle uniones entre segmentos. <i>Fuente: elaboración propia</i>	31
3.17. Molde V3. <i>Fuente: elaboración propia</i>	31
3.18. Segmento V3. <i>Fuente: elaboración propia</i>	32
3.19. V3 final. <i>Fuente: elaboración propia</i>	32
3.20. Parámetros finales. <i>Fuente: elaboración propia</i>	34
3.21. Robot final	35
3.22. Versión final. Fab 1	35
3.23. Versión final. Fab 2	36
3.24. Versión final. Fab 3	36
3.25. Versión final. Fab 4	37
3.26. Versión final. Fab 5	37
3.27. Versión final. Fab 6	37
3.28. Versión final. Fab 7	38

## ÍNDICE DE FIGURAS

---

3.29. Robot final. <i>Fuente: elaboración propia</i> . . . . .	38
4.1. Banco de actuación de partida. <i>Fuente: a) [13], b) [3]</i> . . . . .	39
4.2. Válvulas neumáticas montadas en el banco de actuación. <i>Fuente: elaboración propia</i>	40
4.3. Válvulas neumáticas. Modelos. <i>Fuente: SMC</i> . . . . .	41
4.4. Compresor: STANLEY FATMAX DST 101/8/6. <i>Fuente: Stanley</i> . . . . .	41
4.5. Esquema neumático. <i>Fuente: elaboración propia</i> . . . . .	42
4.6. Microcontrolador utilizado: Arduino Due. <i>Fuente: RS</i> . . . . .	43
4.7. Circuito utilizado en la interfaz de potencia del banco de actuación. <i>Fuente: a) luisllamas.es b) amazon.es</i> . . . . .	44
4.8. Banco de actuación: layout final. <i>Fuente: elaboración propia</i> . . . . .	44
4.9. Conexiones Arduino Due. <i>Fuente: adaptación de datasheet</i> . . . . .	45
4.10. Interfaz UML de la clase valvula. . . . .	47
5.1. Tablero de calibración. <i>Fuente: elaboración propia</i> . . . . .	52
5.2. Cámaras utilizadas. <i>Fuente: a) Logitech, b) Krom</i> . . . . .	52
5.3. Baliza luminosa. <i>Fuente: elaboración propia</i> . . . . .	53
5.4. Proceso del reconocimiento de los puntos en la imagen: FindPoint. <i>Fuente: elaboración propia</i> . . . . .	56
5.5. Distintos espacios de color. <i>Fuente: elaboración propia</i> . . . . .	57
5.6. Matlab ColorThresholder. <i>Fuente: Matlab</i> . . . . .	57
5.7. Regiones MSER. <i>Fuente: Matlab</i> . . . . .	58
5.8. Triangulación estéreo. <i>Fuente: [29]</i> . . . . .	60
5.9. Posicionamiento inicial. <i>Fuente: elaboración propia</i> . . . . .	63
5.10. Transformación adicional 1. <i>Fuente: elaboración propia</i> . . . . .	64
5.11. Transformación adicional 2. <i>Fuente: elaboración propia</i> . . . . .	64
6.1. Ejemplo dataset. <i>Fuente: elaboración propia</i> . . . . .	68
6.2. Nomenclatura, grados de libertad. <i>Fuente: elaboración propia</i> . . . . .	69
6.3. Montaje final. <i>Fuente: elaboración propia</i> . . . . .	70
6.4. Proceso de toma de datos. <i>Fuente: elaboración propia</i> . . . . .	71
7.1. SoftControl: Configuración. <i>Fuente: elaboración propia</i> . . . . .	76
7.2. SoftControl: Visión por computador. <i>Fuente: elaboración propia</i> . . . . .	78

7.3.	SoftControl: Orientación tablero. <i>Fuente: elaboración propia</i> . . . . .	79
7.4.	SoftControl: Modo manual. <i>Fuente: elaboración propia</i> . . . . .	80
7.5.	SoftControl: Modo manual - Control porcentual. <i>Fuente: elaboración propia</i> . .	81
7.6.	SoftControl: Modo manual - Control Radio ángulo. <i>Fuente: elaboración propia</i> .	82
7.7.	SoftControl: Control curvatura - ángulo. <i>Fuente: elaboración propia</i> . . . . .	83
7.8.	SoftControl: Control curvatura - ángulo: función gamma. <i>Fuente: elaboración propia</i> .	83
7.9.	SoftControl: Modo manual - Control directo. <i>Fuente: elaboración propia</i> . . . . .	85
7.10.	SoftControl: Recogida de datos. <i>Fuente: elaboración propia</i> . . . . .	86
7.11.	SoftControl: Ejemplo salida toma de datos. <i>Fuente: elaboración propia</i> . . . . .	87
7.12.	SoftControl: Ejemplo visión de las cámaras. <i>Fuente: elaboración propia</i> . . . . .	88
7.13.	SoftControl: Ejemplo rotaciones. <i>Fuente: elaboración propia</i> . . . . .	88
7.14.	SoftControl: Control. <i>Fuente: elaboración propia</i> . . . . .	89
7.15.	Control: Carga de un dataset. <i>Fuente: elaboración propia</i> . . . . .	90
7.16.	Control: Filtrado del dataset. <i>Fuente: elaboración propia</i> . . . . .	91
7.17.	Control: Búsqueda de los puntos de interés. <i>Fuente: elaboración propia</i> . . . . .	92
7.18.	Control: Movimiento a las posiciones indicadas. <i>Fuente: elaboración propia</i> . . .	93
8.1.	Experimentos 1. <i>Fuente: elaboración propia</i> . . . . .	96
8.2.	Experimentos 2. <i>Fuente: elaboración propia</i> . . . . .	97
8.3.	Experimentos 3. <i>Fuente: elaboración propia</i> . . . . .	98
10.1.	Estructura de descomposición de proyecto. <i>Fuente: elaboración propia</i> . . . . .	102
10.2.	Diagrama de Gantt. <i>Fuente: elaboración propia</i> . . . . .	103
C.1.	Anexo 1: Modelo realizado para la caracterización de sensores elásticos. <i>Fuente: elaboración propia</i> . . . . .	112
C.2.	Anexo 1: Resultados medición diferencial. <i>Fuente: elaboración propia</i> . . . . .	113

## 1. Introducción

Desde la aparición de los primeros robots industriales en el siglo pasado, el campo de la robótica está cada vez más presente en la vida del ser humano. Este cambio en la forma de cambiar la forma en la que tenía el ser humano de hacer las cosas supuso una enorme revolución en el campo de la automatización. De repente disponíamos de máquinas las cuales podíamos programar para que realizaran tareas de forma automatizada. Pero a diferencia de una máquina convencional, este tipo de dispositivos presentaban una gran ventaja: su flexibilidad. Estos dispositivos no estaban limitados a la ejecución de una sola tarea concebida durante su diseño, sino que gracias a sus características se pueden adaptar fácilmente a nuevos escenarios.

Gracias a esta característica, a medida que ha pasado el tiempo cada vez se ha hecho más sencillo encontrarse robots en las fábricas. En estas abundan principalmente los denominados *robots manipuladores*, realizando numerosas tareas. Desde soldadura hasta tareas de ensamblaje, pasando por tareas de empaquetado. Por ello, cada vez se han ido desarrollando más topologías de robots, pasando por brazos robóticos hasta robots paralelos como los robots SCARA. Gracias a esto ha disminuido sustancialmente la cantidad de tareas repetitivas y peligrosas que tienen que ser realizadas por el ser humano en la industria.

Por otra parte, en las últimas décadas, también han aparecido cada vez más robots fuera de las grandes fábricas. Estos realizan desde tareas de inspección, el aspirado del suelo del salón o directamente ayudando al ser humano en alguna tarea doméstica. Esto ha provocado que aparezcan nuevas ramas de investigación en el campo de la robótica.

Uno de estos es la denominada **robótica blanda**. Esta corriente en la robótica busca terminar con una de las mayores características que identifican a los robots tradicionales: su construcción mediante elementos rígidos. En la robótica blanda se busca eliminar todos los elementos rígidos y sustituirlos por elementos flexibles.

La construcción de robots mediante elementos flexibles viene motivada por numerosas razones. Una de ellas está ligada con esta búsqueda de robots cada vez más adaptables a entornos y tareas desconocidas. Esta clase de robots, dadas sus características constructivas, pueden adaptarse, por ejemplo, a terrenos en los que un robot móvil tradicional muchos problemas. O introducirse por, mediante la deformación de su propio cuerpo, en huecos en los que un robot tradicional no podría.

Aparte de la búsqueda de robots cada vez más polivalentes, hay más razones que llevan al desarrollo de la robótica blanda. Gracias a sus características son mucho más aptos para operar en condiciones humanas, gracias a que, en caso de ocurrir una colisión, estos no infligirían daño al usuario.

### 1.1. Motivación

Debido al escaso tiempo que lleva existiendo la robótica flexible, no hay demasiada bibliografía sobre algunos tipos de robots. Uno de estos son los denominados robots manipuladores blandos, cuyo procedimiento de fabricación y control suponen un gran reto debido a sus numerosos grados de libertad.

Esto motiva el desarrollo de un proyecto que en el que se diseñe y fabrique uno de estos robots para poder hacer experimentos con él. Se busca que sea un dispositivo que este a disposición en futuros proyectos para fomentar la investigación en este tipo de tecnología en el centro de

Automática y Robótica.

### 1.2. Estado actual

Actualmente, se parte de un banco de actuación neumática, que se desarrolló en el trabajo final de grado de Víctor Barroso [13], posteriormente modificado en el trabajo final de grado de Ángela Serrano [3]. También se toma inspiración de otros trabajos realizados en el centro de Automática y robótica en el campo de la robótica flexible, pero se parte de un diseño totalmente de cero.

### 1.3. Objetivo del trabajo fin de grado

Este trabajo tiene como finalidad el diseño, construcción y control de un robot blando manipulador de nueve grados de libertad. Este debe estar fabricado en su gran mayoría mediante algún material flexible y todos los grados de libertad deben de poder controlarse de forma independiente. El robot estará actuado neumáticamente, por lo que también es necesaria la modificación del banco neumático desarrollado por Víctor Barroso.

También se espera que el posterior control del robot sea mediante alguna estrategia basada en datos empíricos, por lo que es necesario un diseño experimental y se desarrollará un sistema de visión por computador que agilice este proceso. Todo esto lleva a desarrollar también una herramienta software que centralice todo esto y facilite desarrollos posteriores en el robot.

Para la realización de esto son necesarios otras investigaciones adicionales como la: investigación de materiales, el desarrollo de un proceso de fabricación o el diseño de un procedimiento experimental para recoger los datos.

De esta forma se pueden resumir los objetivos de este trabajo final de grado en los siguientes puntos:

- Concepción de un robot manipulador blando actuado neumáticamente. Elección de los materiales y diseño de un proceso constructivo.
- Iteración de distintos diseños hasta llegar a un robot que cumpla con los objetivos previstos.
- Concepción y diseño de un sistema de adquisición de datos, basado en visión por computador. Este debe identificar la posición del extremo del robot en el espacio, y obtener los parámetros que definen su posición y orientación con respecto a un sistema de referencia fijo.
- Rediseño del banco de actuación neumático disponible, de acuerdo con las necesidades de este y futuros proyectos.
- Desarrollo de un programa *software* que sintetice todos los desarrollos realizados en una sola herramienta de fácil uso por otros usuarios

Además, se realizarán pruebas con sensores elásticos que permitan un futuro control en cadena cerrada.

## 1.4. Estructura de la memoria

Esta memoria engloba todos los desarrollos realizados en el proyecto.

Se comienza con el capítulo 2. **Estado del Arte**, en el cual se hará un breve repaso sobre los desarrollos actuales en el campo de la robótica blanda. Se dará un contexto sobre el área del conocimiento que engloba a este proyecto.

Después, en el capítulo 3. **Diseño y fabricación**, se detallarán en mayor profundidad los requerimientos de diseño del robot a desarrollar. También se tratará la elección de materiales y el diseño de un proceso de fabricación adecuado. Se irán relatando las distintas iteraciones que se han desarrollado hasta llegar a la versión final.

En el siguiente capítulo 4. **Banco de actuación**, se abordan todos los desarrollos y modificaciones relacionados con el banco neumático de actuación. Esto incluye también el código de bajo nivel que se encarga de la comunicación con el software.

A continuación, en el capítulo 5. **Sistema de adquisición de datos**, se explicará el sistema de visión por computador que se ha desarrollado. Esto incluye el algoritmo, el hardware y las funciones necesarios para implementarlo posteriormente en *MATLAB*. A este capítulo le sigue el capítulo 4. **Diseño experimental y recogida de datos**, en el cual se detalla el procedimiento propuesto para recoger *datasets* de una forma automatizada.

Finalmente, en la sección 7. **Interfaz de usuario y control**, se explica paso por paso el funcionamiento del programa de control realizado. También se explica la metodología utilizada para el control del robot fruto de este trabajo.

También se incluye un capítulo de 8. **Experimentos y resultados**, uno de 10. **Planificación y presupuesto**, 9. **Sostenibilidad** y 11. **Conclusiones y trabajos futuros**.

Se han añadido en los anexos los desarrollos realizados con sensores elásticos y algunos planos de elementos diseñados en el contexto de este trabajo.

## 2. Estado del arte

A continuación, se pretende presentar un resumen del contexto que rodea al campo de conocimiento de la robótica flexible, en que consiste, y cuál es el marco que engloba el robot fruto de este trabajo. También se explican las ventajas que un robot blando presenta frente a un robot rígido tradicional.

### 2.1. Contexto

Tradicionalmente, los robots manipuladores han consistido en un conjunto de eslabones rígidos que unidos entre sí mediante articulaciones formando la llamada *cadena cinemática* (figura 2.1). En estas articulaciones se encuentran los actuadores, que son los que hacen posible los movimientos del robot. Estos elementos pueden estar dispuestos de una manera más o menos compleja, determinando el rango de movimientos que el robot puede realizar.

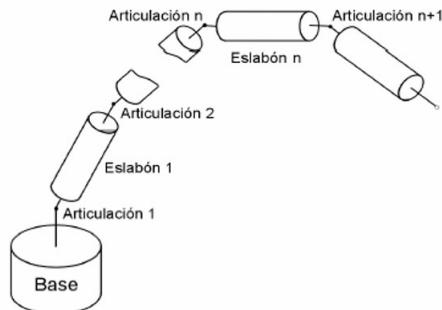


Figura 2.1: Cadena cinemática de un robot serie. *Fuente: UDLAP*

Comúnmente estos elementos rígidos se concatenan de dos maneras distintas, dando lugar a los robots paralelos y a los robots serie. Nos centraremos en los segundos, que se dan cuando los eslabones de la cadena cinemática se agrupan uno tras otro, dando lugar a la llamada cadena cinemática abierta.



(a) Robot serie. *Fuente: Kuka*  
(b) Robot paralelo. *Fuente: Omron*

Figura 2.2: Robots manipuladores tradicionales serie y paralelo

La complejidad de los movimientos que el robot puede realizar viene determinada por el número de grados de libertad. Estos consisten en el número de movimientos espaciales independientes que el robot puede realizar, determinados por el número de actuadores que se pueden controlar

de manera independiente. De esta manera, para un robot de  $n$  grados de libertad son necesario mínimo  $n$  actuadores. Además, al ser los elementos de conexión entre articulaciones indeformables, para  $n$  grados de libertad serán necesarios  $n-1$  cuerpos sólidos. Otra característica propia de los robots manipuladores tradicionales es lo compleja que resulta su adaptación a escenarios desconocidos y su respuesta ante perturbaciones.

Recientemente, se han comenzado a investigar una nueva clase de robots denominados robots blandos o comúnmente *soft bots*. En estos, en vez de estructuras sólidas, se emplean estructuras flexibles. Se pasa de una cadena cinemática sólida, en la que los movimientos vienen determinados por la actuación de motores que mueven los cuerpos sólidos, a la deformación del propio cuerpo en sí. En la figura (2.3) puede observarse el *soft bot* caminante fruto del trabajo de Ángela Serrano



Figura 2.3: Robot blando caminante. *Fuente:* [3]

En la tesis doctoral de Silvia Terrile "Soft robotics: Applications design and control" ([20]) se explica que actualmente no hay aceptada una definición universal para *soft bot*, pero que se pueden definir como robots cuyos cuerpos estén formados por módulos de Young comprendidos entre  $10^4 - 10^9 \text{ Pa}$ . Esta característica los permite adaptarse mejor al entorno y no ser dañinos. En la imagen (2.4) se puede observar la clarificación que hace Silvia de los soft bots.

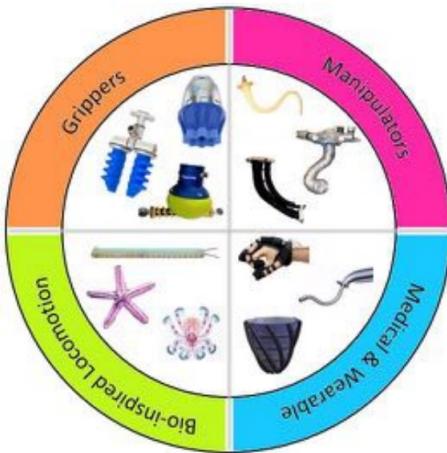


Figura 2.4: Clasificación de soft bots. *Fuente:* [20]

Volviendo de nuevo a la comparación con los robots rígidos tradicionales, aparte de la facilidad de adaptarse a entornos desconocidos y la mayor seguridad en el entorno humano, los *soft bots manipuladores* presentan otra gran ventaja. En los rígidos es necesario disponer como mínimo de  $n-1$  cuerpos rígidos por cada  $n$  grados de libertad, mientras que en un *soft bot* el número de cuerpos no está determinado por el número de grados de libertad. De esta manera se puede construir un robot manipulador en el cual cada segmento pueda deformarse en varias direcciones diferentes (figura 2.5) disminuyendo el número de cuerpos necesarios para alcanzar los grados de libertad deseados

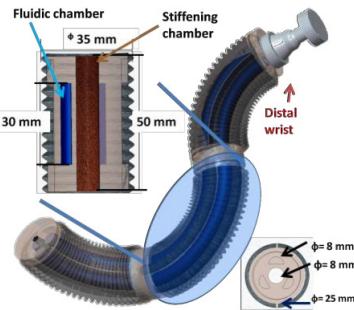


Figura 2.5: Robot manipulador blando. *Fuente: [14]*

En la figura (2.6) se puede observar una comparación entre los robots manipuladores blandos y rígidos

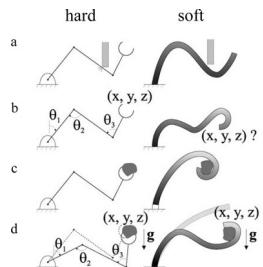


Figura 2.6: Comparación entre robots blandos y rígidos. *Fuente: [23]*

## 2.2. Tipos de actuadores

En los *soft bots* la forma de actuación es una parte muy compleja del diseño, puesto que tiene que ser compatible con la característica flexible del robot. Los actuadores tienen que cumplir la funcionalidad de deformar los cuerpos que forman el robot de la manera adecuada, sin comprometer la estructura de este. Las técnicas de actuación empleadas actualmente se pueden dividir en dos grupos principales [28].

1. Mecanismos de actuación intrínseca
  - a) Actuación neumática
  - b) Polímeros electroactivos
  - c) Magnéticos
2. Mecanismos de actuación extrínseca
  - a) Materiales con memoria de forma: SMA
  - b) Twisted-Coiled actuators: TCA
  - c) Robots actuados por cables: RAC

### 2.2.1. Mecanismos de actuación intrínseca

A este grupo pertenecen todos los mecanismos de actuación en los que el elemento que produce la deformación del robot es el mismo que forma su masa estructural. De esta manera, no hacen

falta elementos auxiliares dentro del cuerpo del robot para lograr la actuación, al ser este mismo el que se deforma. A continuación, se repasan algunos mecanismos de actuación perteneciente a este grupo

### 2.2.1.1 Neumáticos

En los *soft bots* con actuación neumática, la deformación proviene del inflado y desinflado de unas vejigas presentes en el cuerpo elástico del robot. Esta deformación es provocada mediante una diferencia de presiones que normalmente se logra gracias a un compresor y unas electroválvulas que se encargan de gestionar la entrada y salida del aire de cada una de estas vejigas. La variación del volumen de estas vejigas en conjunto logra la deformación deseada del robot. En la figura (2.7) se puede observar un esquema del funcionamiento de este tipo de actuación.

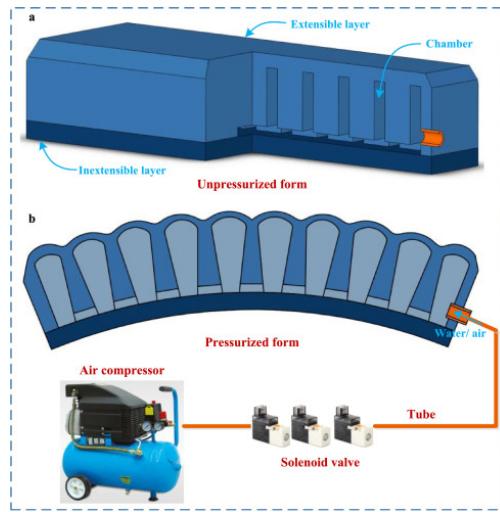
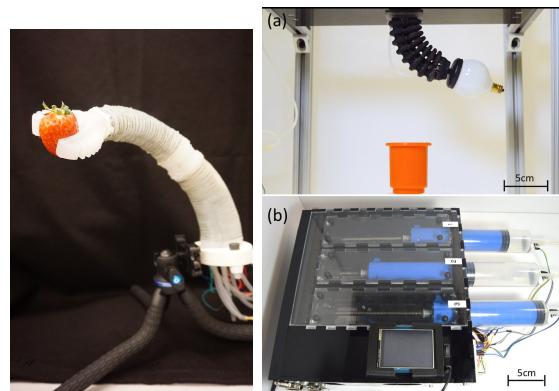


Figura 2.7: Actuación neumática. Fuente: [2]

Dentro de la actuación de tipo neumática, puede diferenciarse la convencional y la diferencial. En la primera, el conjunto neumático consiste principalmente en las electroválvulas y un compresor, siendo esta la fuente de aire a presión. Un ejemplo de aplicación de este tipo de actuación es el denominado *robot Sopra*, [22], que puede apreciarse en la figura (2.8.a).

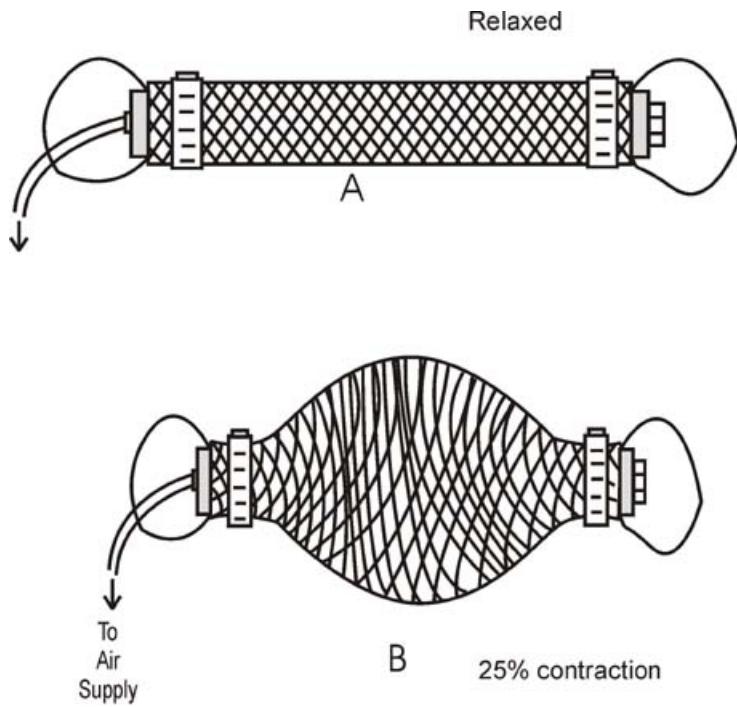
Por otra parte, se encuentra la actuación neumática diferencial. En este caso, la diferencia de presiones proviene del cambio de volumen de algún elemento conectado a las vejigas del robot. De esta manera, el volumen de aire que se puede introducir a las vejigas es *finito*. Un ejemplo de esto es el robot que puede apreciarse en la figura (2.8.b), fruto del trabajo de Tom Kalinsky [7]



(a) Robot sopra. (b) Actuación diferencial.  
Fuente: [22] Fuente: [7]

Figura 2.8: Ejemplos actuación neumática

Un último ejemplo de actuación neumática se encuentra en los denominados músculos de McKibben. Estos se basan en una cámara neumática elástica recubierta de una malla que restringe los movimientos para que estos se realicen en la dirección deseada.



When muscle is pressurized (B), it can contract up to about 75% of its relaxed length

Figura 2.9: Músculos de McKibben. Fuente: [21]

### 2.2.1.2 Polímeros electroactivos

En este grupo se encuentran todos los actuadores basados en distintos tipos de polímeros electroactivos (*Electroactive polymers o EAPs*). Los *EAPs* son polímeros que cambian sus característi-

cas ópticas u mecánicas en el momento en el que son estimulados por un fuerte campo eléctrico [4]. Estos se pueden dividir en dos principales grupos: los polímeros electroactivos dieléctricos y los polímeros electroactivos iónicos. Estos dos grupos a su vez pueden subclaseficarse tal y como se aprecia en la figura 2.10

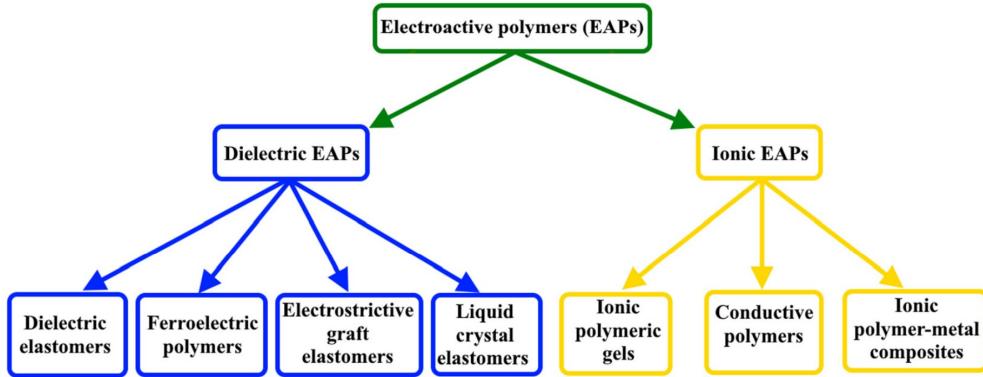


Figura 2.10: Clasificación de los polímeros electroactivos. *Fuente [4]*

Dentro de esta clasificación, unos de los más estudiados hasta el momento son los elastómeros dieléctricos. Estos se basan dos electrodos entre los cuales se encuentra una fina lámina dieléctrica conductora. De forma similar a como sería un condensador flexible, formando un material compuesto que da forma al cuerpo del actuador.

Cuando se somete el cuerpo a una fuerte diferencia de potencial, se genera una fuerza atractiva entre ambos electrodos, llevando a la deformación del conjunto [12] tal y como puede apreciarse en la figura 2.11.

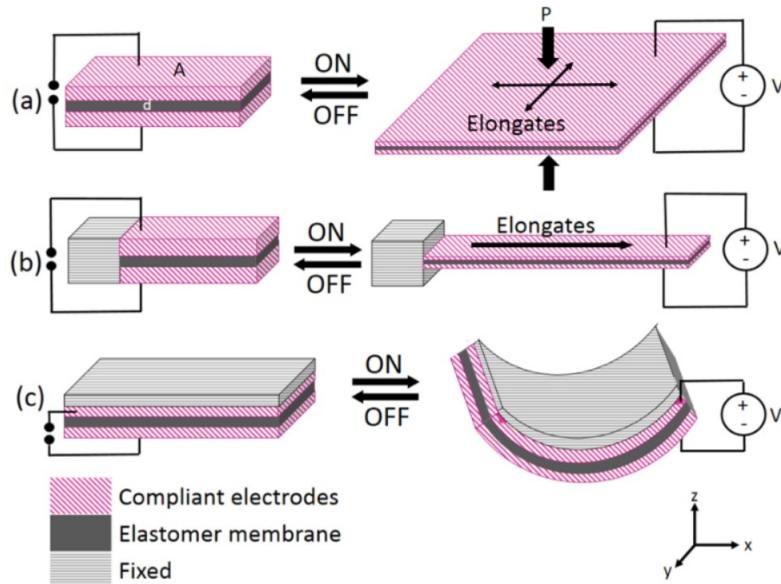


Figura 2.11: Elastómeros dieléctricos. *Fuente: [9]*

### 2.2.1.3 Magnéticos

Otra forma de actuación de *soft bots*, son los basados en *limos* magnéticos. En este caso, el *robot* en sí mismo está constituido por un ferro fluido o metal líquido que se deforma en presencia de campos magnéticos. De esta manera, variando el campo magnético, es posible controlar el robot.

Un ejemplo de actuación de este tipo es el trabajo realizado en *Reconfigurable Magnetic Slime Robot: Deformation, Adaptability, and Multifunction* [19], donde se demuestra la capacidad de adaptación y deformación de robots basados en fluidos magnéticos no newtonianos (figura 2.12).

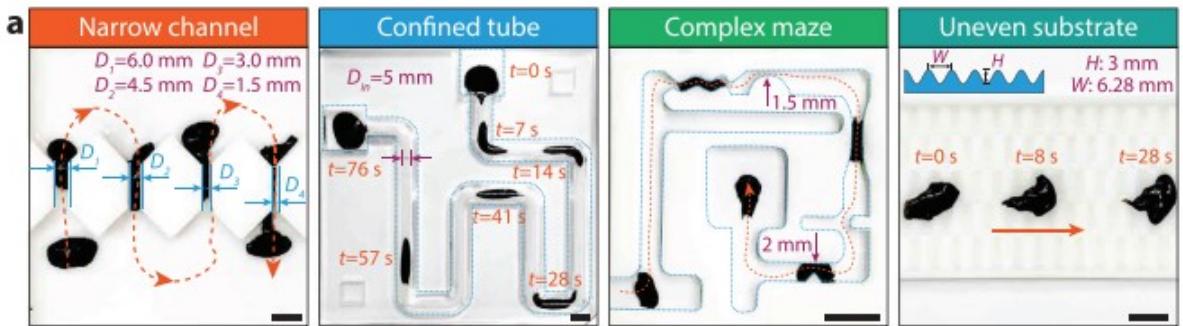


Figura 2.12: Actuación magnética. Fuente: [19]

Una posible aplicación de los soft bots basados en este principio de actuación es en el ámbito médico. Gracias a su método constructivo y principio de funcionamiento, existe la posibilidad de construir *nano bots* para operaciones quirúrgicas mínimamente invasivas.

### 2.2.2. Mecanismos de actuación extrínseca

Este segundo grupo de mecanismos de actuación de *softbots* engloba los denominados *mecanismos de actuación extrínsecos*, tal y como se refleja en la clasificación que se realiza en el trabajo [28]. Los mecanismos incluidos en este grupo comparten la característica de que el elemento que produce la deformación del cuerpo flexible al robot no es solidario a este. En otras palabras, el cuerpo del robot y el elemento que produce su deformación, son dos elementos distinguibles que no se encuentran pegados de forma sólida.

A continuación, se hace un breve resumen de los principales mecanismos de actuación que forman este grupo.

#### 2.2.2.1 Materiales con memoria de forma (SMA)

Los materiales con memoria de forma o *Shape Memory Alloys (SMA)* son un grupo de aleaciones metálicas que comparten la capacidad de volver a una forma definida previamente cuando son sometidos al tratamiento térmico adecuado.

Este cambio de forma se basa en el cambio entre la fase martensítica, a bajas temperaturas, y la austenítica, a altas temperaturas. En la primera fase el material es maleable y en la segunda se vuelve más rígido. En la transformación entre estas fases se produce un cambio de densidad en la aleación que lleva al cambio de forma, tal y como se aprecia en la figura a continuación.

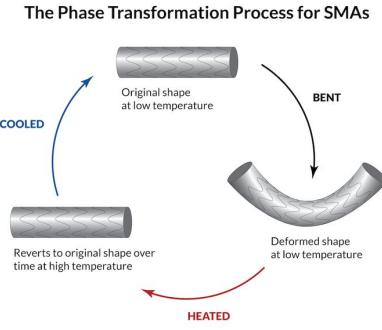


Figura 2.13: Materiales con memoria de forma

Gracias a esta propiedad, estos materiales han sido bastante estudiados en el campo de la robótica flexible. Cuando son utilizados como actuadores, normalmente se presentan con la topología de *muelles*, que se encuentran pretensados dentro de la estructura flexible del robot. Al hacer pasar una corriente por estos, se calentarán por la ley de Joule, llevando al cambio de fase y la posterior deformación del robot.

Un ejemplo de este tipo de actuación es el robot fruto del trabajo de Miguel Guzmán [11] que puede apreciarse en la figura 2.14.

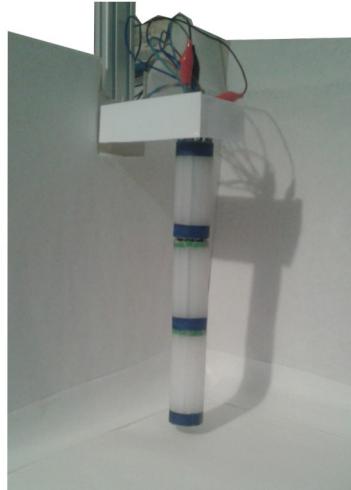


Figura 2.14: Robot blando actuado mediante SMA. Fuente: [11]

### 2.2.2.2 Twisted - Coiled Actuators (TCAs)

Otro mecanismo de actuación está basado en los *Twisted - Coiled actuators* o TCA. Este mecanismo de actuación se basa en la deformación de un hilo de nailon enrollado en un compacto muelle helicoidal [16]. Cuando el polímero se calienta, este se contrae y gracias a su topología de muelle, lleva a una deformación macroscópica, de forma parecida a los actuadores basados en SMA (figura 2.15).

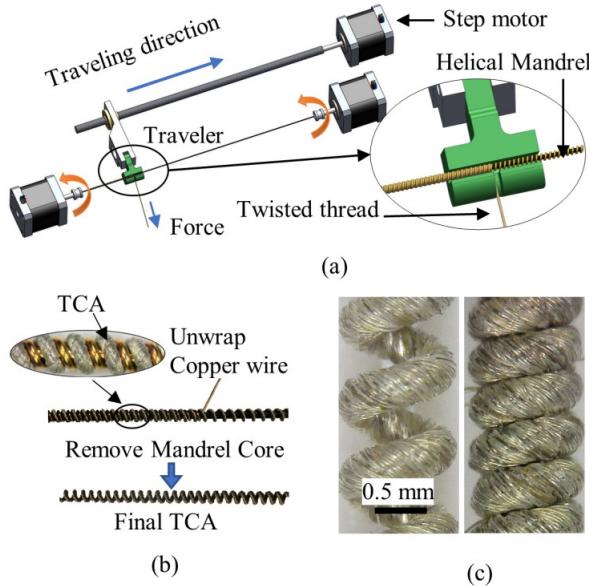


Figura 2.15: Fabricación de twisted-coiled actuators. Fuente: [16]

Para lograr este cambio de temperatura, normalmente se enrolla el hilo de nailon junto a algún otro elemento que produzca calor ante el paso de corriente, como puede ser el hilo de cobre o nicromo.

Dentro de este grupo de mecanismos de actuadores, se han hecho además avances que incluyen el uso del actuador en sí mismo también como elemento de sonorización, como en el trabajo realizado por Jiefeng Sun y Jianguo Zhao [18], donde por medio de la medida de la resistencia del hilo se puede estimar como de actuado está.

Un ejemplo de aplicación de este tipo de actuadores en la robótica es en el trabajo realizado en *Twisted-and-Coiled Actuators with Free Strokes Enable Soft Robots with Programmable Motions* [17], que se puede apreciar en la figura (2.16). En este *softbot* se han introducido las hélices de polímero en los cuerpos de silicona llevando a distintos tipos de grados de libertad (rotación, desplazamiento lineal, etc.).

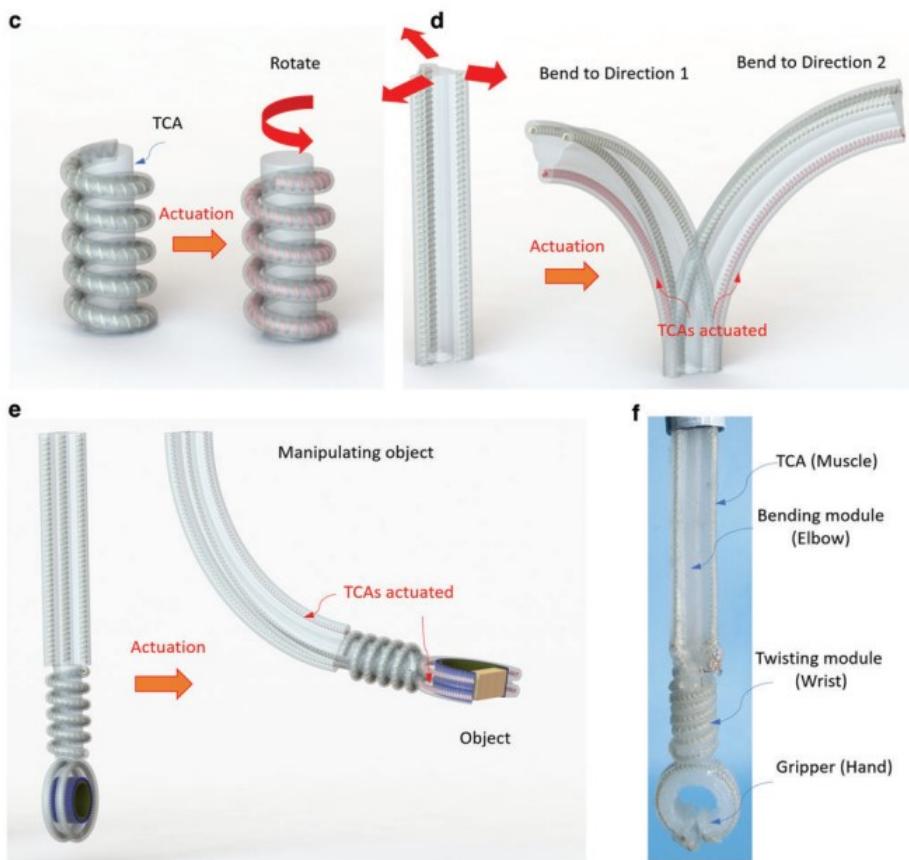


Figura 2.16: Robot basado en TCA. Fuente: [17]

### 2.2.2.3 Actuado por cables

Otro ejemplo de actuaciones extrínsecas son los robots actuados por cables. En este caso, el elemento que produce la deformación de la estructura son unos cables de los tiran unos motores. Un ejemplo de trabajo realizado que utiliza este tipo de actuación, es el *Softbot* quirúrgico realizado en *A cable-driven soft robot surgical system for cardiothoracic endoscopic surgery: preclinical tests in animals* [27], que puede observarse a continuación en la figura 2.17.

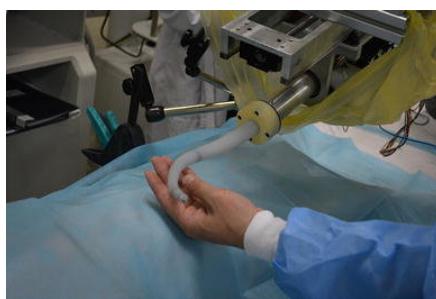


Figura 2.17: Robot blando actuado por cables para aplicaciones quirúrgicas. Fuente: [27]

### 2.3. Control de softbots

Una característica fundamental que diferencia a los *soft bots* de los robots tradicionales, es su estructura flexible. Esto lleva a que el modelado y posterior control de este tipo de robots sea una tarea muy compleja, que en muchos casos requiere soluciones distintas a las empleadas para el modelado de robots tradicionales.

Esta mayor complejidad reside principalmente en que, debido a esta estructura flexible, puede considerarse que los *soft bots* disponen de infinitos grados de libertad. En otras palabras, no existe una cadena cinemática rígida que modelar, lo que hace que muchos de los métodos empleados en robots tradicionales sean inservibles en la robótica blanda.

A parte de su cuerpo flexible, hay muchas otras variantes que hacen que los modelos de control se vuelvan muy complejos. En el caso de los *soft bots* neumáticos, por ejemplo, no solo es necesario el modelado del cuerpo flexible, sino también toda la fluidodinámica que lleva a la deformación del robot.

Sin embargo, han ido surgiendo distintas estrategias para controlar este tipo de robots. Estas se pueden dividir en dos grupos principales. Al primero de ellos, pertenecen las técnicas de control que se basan en modelar el comportamiento físico del robot, como, por ejemplo, por elementos finitos.

Al segundo grupo pertenecen las técnicas que se basan en métodos empíricos. Con esto nos referimos, a que en vez de modelar de forma completa el comportamiento físico del robot, que puede ser una tarea imposible en muchas ocasiones, se recurre a tomar datos experimentales. Estos datos experimentales pueden ser por ejemplo una relación entre los estímulos que se dan a los actuadores, y la pose alcanzada por el robot. Posteriormente, estos datos se pueden tratar de distintas maneras, por ejemplo, entrenando una red neuronal.

A continuación, se hace un resumen de las principales técnicas de modelado aplicadas en soft bots.

#### 2.3.1. Modelado mediante elementos finitos

Las herramientas basadas en elementos finitos han sido ampliamente utilizadas en el campo de los soft bots. Estas se basan en intentar replicar el comportamiento físico del robot en una simulación por ordenador, para poder estimar lo mejor posible como se comportará el robot en la vida real.

Estas herramientas, dentro del campo de los soft bots, se han utilizado tanto para modelarlos como para el control en sí mismo.

En el primer caso, el objetivo es simular el comportamiento del robot con el fin de disminuir el número de iteraciones experimentales hasta lograr un diseño que logre el movimiento óptimo. Esto es por ejemplo lo que se realiza en el trabajo realizad por Víctor Barroso [13] (figura 2.18)

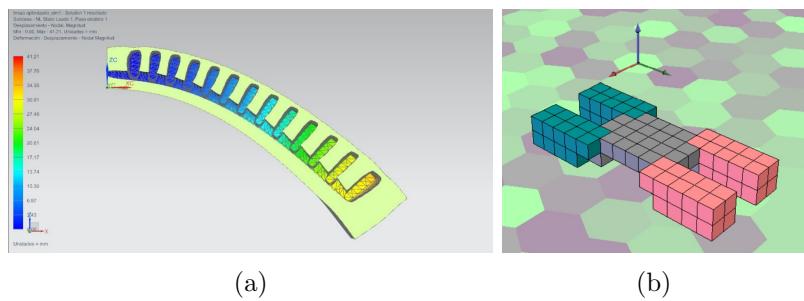


Figura 2.18: Elementos finitos - control en tiempo real

El otro uso que se le ha dado a esta herramienta en el campo de los robots blandos es en el control cinemático de estos. En este caso, el objetivo se trata de obtener un equivalente a lo que sería el modelo cinemático inverso y directo en robots rígidos tradicionales. Un ejemplo de esto es el trabajo realizado en *Dynamically Closed-Loop Controlled Soft Robotic Arm using a Reduced Order Finite Element Model with State Observer*

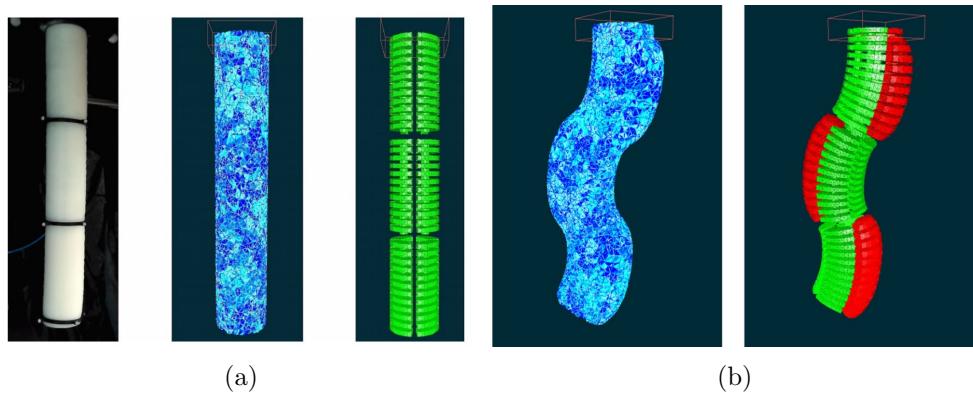


Figura 2.19: Elementos finitos - control en tiempo real

### 2.3.2. Control basado en curvatura constante

Otra forma de controlar este tipo de robots está enfocada en los *SoftBots* continuos. Se basa en asumir que los segmentos de los cuales estos se componen, al deformarse, lo hacen manteniendo una curvatura constante a lo largo del segmento. Esto permite hallar un sistema rígido equivalente y desarrollar el control sobre este (figura 2.20).

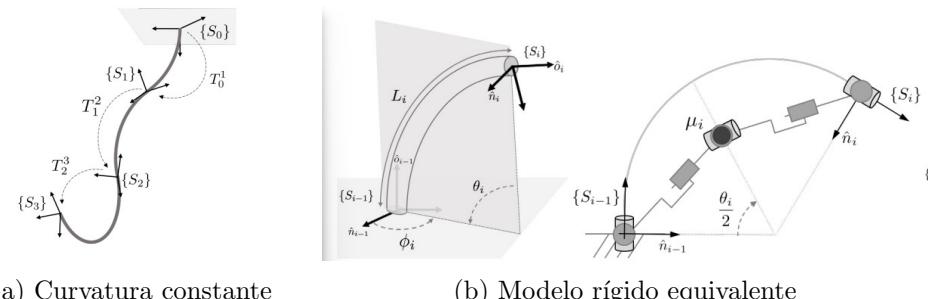


Figura 2.20: Control basado en curvatura constante

Esto permite simplificar significativamente el control, puesto que se reduce el sistema a uno que está ampliamente estudiado y se puede resolver de numerosas maneras. Un ejemplo de trabajo realizado utilizando esta técnica de control nos lo encontramos en *Dynamic Motion Control of Multi-Segment Soft Robots Using Piecewise Constant Curvature Matched with an Augmented Rigid Body Model* ([8]).

### 2.3.3. Entrenamiento mediante red neuronal

Finalmente, nos encontramos la otra vertiente que se ha desarrollado en los últimos años con respecto al control cinemático y dinámico de los denominados robots blandos. Estos métodos se basan en datos recogidos experimentalmente, a los cuales se denomina *datasets*. Estos se pueden emplear para entrenar distintas topologías de redes neuronales, que sean las que se encarguen del control del robot.

En este caso se entrena una red neuronal con datos del robot que se obtienen experimentalmente. Un ejemplo de esto es el trabajo realizado en *Learning the inverse kinematics of an octopus-like manipulator in three-dimensional space* [5], donde se consigue controlar un *softbot* actuado por cables mediante una red en el espacio tridimensional mediante una red neuronal. Esta red de tipo *feedforward* estima la relación entre la fuerza a aplicar en los cables y la posición en el espacio que se desea que alcance el robot. De esta manera no es necesario resolver de forma analítica las ecuaciones diferenciales no lineales que definen la cinemática inversa, encargándose de esta tarea el modelo resultante del entrenamiento. En este caso se trata de un entrenamiento supervisado.

## 2.4. Aplicaciones

Debido al escaso tiempo de desarrollo que tiene la robótica flexible, todavía no son un tipo de robots muy presentes en industria. Sin embargo, sí que se pueden observar distintas tendencias en las cuales empiezan a surgir aplicaciones de este tipo de tecnologías.

La primera de ellas es en el campo de la medicina. Gracias a que en muchos casos el elemento que produce el movimiento del *SoftBot* se encuentra en un lugar físico distinto, es posible asegurar de una forma más sencilla un espacio libre de contaminantes de lo que sería con un robot tradicional. En la referencia [5], se realiza un trabajo sobre un *softbot* que tiene este objetivo.

Dentro de las fábricas, un lugar en el que se empiezan a encontrar cada vez más elementos que hacen uso de la robótica blanda, es en las herramientas de los extremos de brazos robóticos tradicionales o de *cobots*. En este caso, se aprovecha la característica de los *SoftBots* de adaptarse a entornos desconocidos gracias a su estructura flexible. Un ejemplo de esto es el trabajo desarrollado por Miguel Arguelles Hortelano en *Desarrollo de herramientas de agarre basadas en robótica blanda para robots manipuladores colaborativos* [6], que se puede apreciar en la figura 2.21.

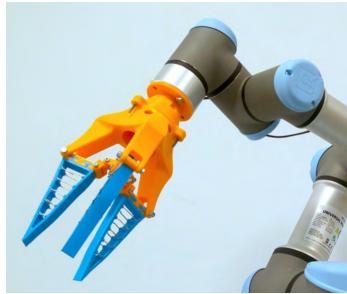


Figura 2.21: Herramienta basada en robótica blanda. *Fuente: [6]*

Sin embargo, gracias a los desarrollos que se han realizado en los últimos años, empiezan a aparecer robots manipuladores basados en su totalidad en la tecnología de la robótica blanda. Estos permiten una alta cooperación entre máquina y usuario, gracias a sus elementos flexibles que minimizan los daños en caso de impacto. Uno de los ejemplos comerciales más sobresalientes de esto es el brazo softbot manipulador desarrollado por la empresa Festo: *BionicSoftArm* (figura 2.22)

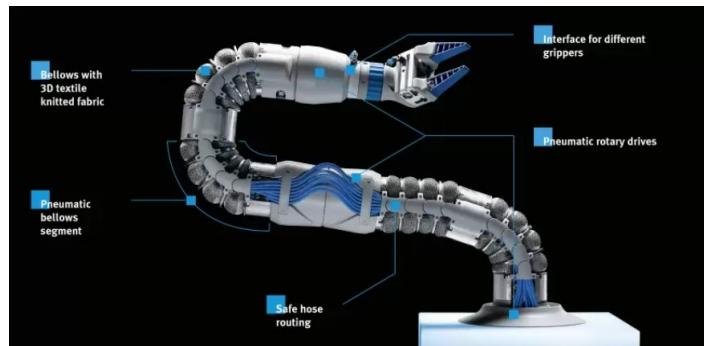


Figura 2.22: Festo BionicSoftArm. *Fuente: [15]*

Otra aplicación práctica en la que la robótica blanda está adquiriendo importancia es en la exploración de terrenos desconocidos. Aquí se aprovecha la capacidad de este tipo de robots de modificar su propia estructura para introducirse por huecos que un robot normal no podría, por ejemplo deshinchándose antes de entrar al hueco e hinchado después. Además, gracias a sus características constructivas son capaces de adaptarse a terrenos abruptos e imprevistos.

### 3. Diseño y fabricación

En el campo de los *soft bots* toma especial importancia el diseño, puesto que el cuerpo físico del robot hace a su vez de elemento de actuación. Esta actuación es fruto de la deformación provocada por la diferencia de presiones entre el aire exterior y el interior del cuerpo, conformándose así un sistema muy complejo de modelizar y del cual no se dispone de un modelo cinemático que rija su comportamiento. Acorde con esta circunstancia, la componente experimental toma una gran relevancia.

En el siguiente capítulo se abordan todas las fases de diseño y fabricación del robot resultante de este trabajo y el proceso de fabricación final puede consultarse en el siguiente enlace:

**link:** Proceso de fabricación final

#### Restricciones de diseño iniciales

Previo a comenzar el diseño y la posterior fabricación era necesario fijar algunas especificaciones dimensionales y geométricas impuestas por la funcionalidad buscada. Estos requisitos pueden resumirse en los siguientes puntos:

- El robot resultante debe constar de tres segmentos actuados de forma independiente contando cada uno de estos con tres grados de libertad
- La actuación de los segmentos que componen el robot debe ser neumática
- Los segmentos deben de estar compuestos de silicona flexible y sensorizados mediante gomas conductivas
- Estos segmentos deben de permitir el montaje y desmontaje sencillo, así como un diseño modular
- Los tubos neumáticos deben de quedar completamente embebidos en el cuerpo del robot para evitar roturas y permitir movimientos más complejos

De esta manera se decide que cada segmento tendrá una forma cilíndrica y contará con tres grupos de vejigas como las que se aprecian en la figura 3.1. Además, contará con un canal central hueco por el que puedan pasar los tubos de presión para los siguientes segmentos. El diseño se explicará de forma más detallada en el apartado 3.2

#### 3.1. Método de fabricación y materiales empleados

La búsqueda de un método de fabricación y los materiales adecuados para este trabajo ha supuesto un gran reto debido a la compleja geometría interior de los segmentos que componen el robot y los requisitos funcionales que tiene que cumplir. Este apartado contiene un breve resumen de los métodos de fabricación y materiales empleados.

Durante el transcurso de este trabajo se han utilizado dos métodos de fabricación basados en el moldeo, que difieren ligeramente en la manera en la que se consigue la geometría interior de los segmentos. Ambos comparten el uso de un molde impreso en 3D en el cual se colocan unas

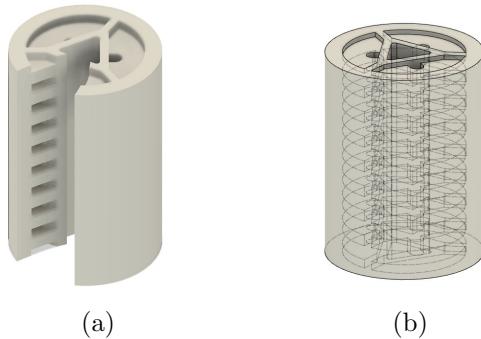


Figura 3.1: Detalle de las vejigas. *Fuente: elaboración propia*

piezas denominadas *machos*, los cuales algunos se pueden eliminar una vez curada la silicona. Este molde, una vez montado, se rellena de silicona, que una vez curada, se extrae y se procede a la eliminación de los *machos* que dan forma a las vejigas. En la figura 3.2 pueden observarse las distintas fases.

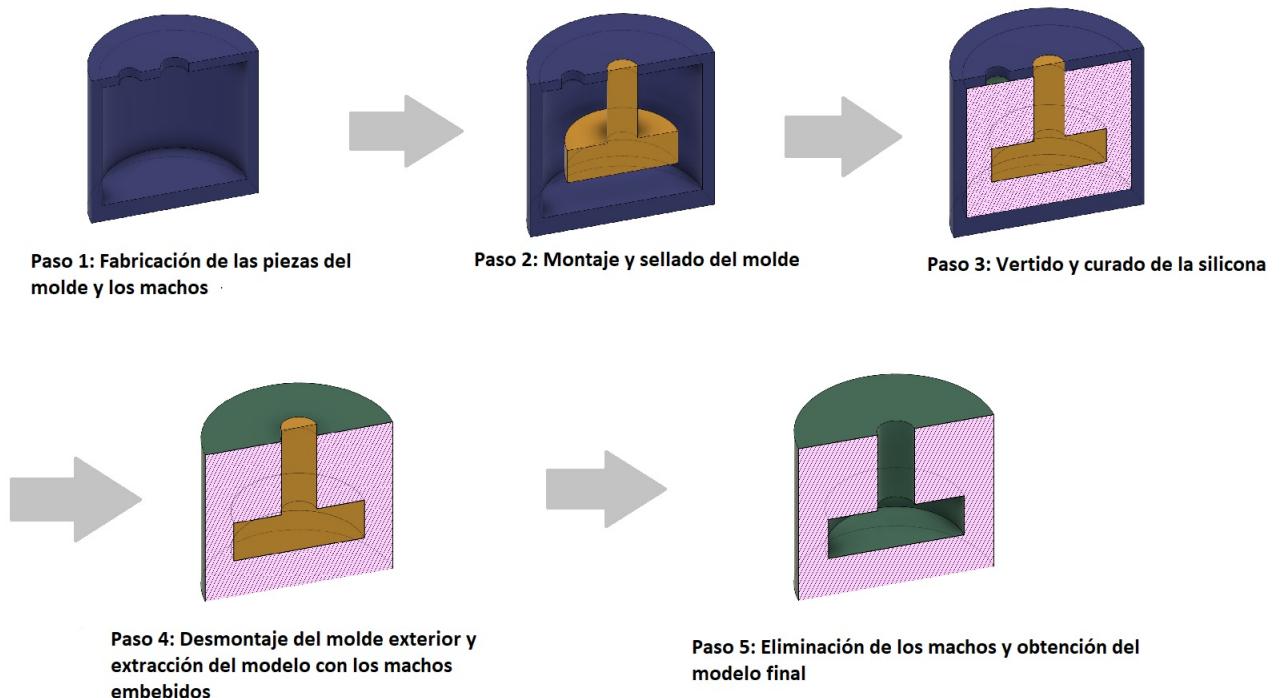


Figura 3.2: Método de fabricación empleado. *Fuente: elaboración propia*

Los dos métodos de fabricación empleados difieren en el paso 1 y en el paso 5. En el primero, los machos son fabricados con un polímero soluble en agua denominado PVA. Este se adquiere en forma de filamento apto para impresión 3D que permite imprimir en 3d los machos directamente desde el diseño CAD. Posteriormente, los machos son eliminados usando una bomba que recircula agua caliente para eliminar todos los restos.

El segundo está basado en el método de la cera perdida. En este, se fabrican los machos mediante impresión 3D utilizando un polímero rígido. Posteriormente, se crea un molde negativo de los

machos empleando silicona de platino. Este molde de silicona es usado para crear un modelo de los machos utilizando esta vez cera de parafina, que tiene la ventaja de poder eliminarse completamente únicamente usando calor.

El primer método de fabricación se ha utilizado en los primeros modelos debido a la facilidad de prototipado de las vejigas, al fabricarse directamente en el material soluble mediante impresión 3D. Para el modelo final se ha utilizado el basado en la cera perdida, debido a la facilidad de la eliminación de los machos y un mayor ajuste dimensional. La fabricación de cada modelo en concreto y la justificación del uso de un método de fabricación u otro se detallan con más profundidad en el apartado 3.2



Figura 3.3: Impresora 3D: Artillery Genius. *Fuente: Artillery*

### 3.1.1. Justificación de la elección de la silicona

Los segmentos tienen que ser lo suficientemente elásticos para poder deformarse, pero también lo suficientemente rígidos para mantener la rigidez estructural necesaria. Teniendo esto en cuenta se hizo una investigación para determinar el rango de durezas aceptable para las siliconas.

Para poder comparar las distintas siliconas disponibles se han caracterizado por su índice de dureza shore y su tiempo de curado. En la figura 3.4 se puede ilustrar dicha escala.

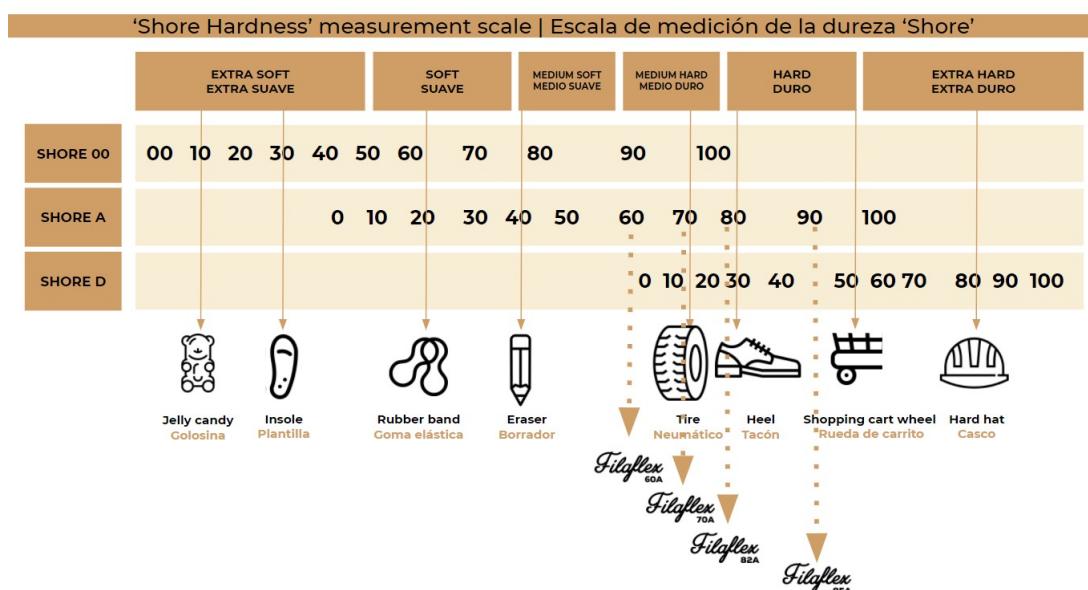


Figura 3.4: Escala Shore. *Fuente: recreus.com*

A su vez, las siliconas se pueden clasificar entre las de platino y las de estaño. Las primeras también se las denomina de adición, nombradas de este modo por la reacción que se lleva a

cabo. Su manejo es más sencillo debido a su contracción despreciable durante el proceso de canalización y a su nula toxicidad. Por otra parte, las siliconas de estaño, también denominadas de condensación, presentan como principal ventaja que son menos susceptibles a fallos de curado debido a impurezas, pero presentan contracción durante el curado ( $\approx 1\%$ ) debido a la producción de alcoholes.

Para establecer la cota inferior de dureza se tomó como referencia el trabajo Víctor Barroso [13] y Ángela Serrano [3] obteniéndose así shore 00-30. Para establecer la cota más alta de dureza se tomó referencia del trabajo de [25] obteniendo así Shore A 30.

Posteriormente, se realizó una búsqueda de siliconas de estas características, empleándose finalmente las que se reflejan en la siguiente tabla

Nombre comercial	Tipo	Dureza Shore	Tiempo de curado	Modelo (*)
PlatSil FS10	Platino	A13	12 minutos	Versión 1, Versión 2 Modelo sensores
EasyPlat 0030	Platino	00-30	4 horas	Versión final 1
TinSil 8015	Estaño	A15	24 horas	Versión final 2

Cuadro 3.1: Siliconas usadas en el proyecto

\* En el apartado 3.2 se explicará en detalle para qué se usó cada una de las siliconas y la justificación de los cambios.

### 3.2. Diseño y fabricación de los módulos

Para el diseño CAD de todos los modelos empleados se ha utilizado la herramienta *Autodesk Fusion 360*. Este software permite una gran flexibilidad a la hora de modificar diseños.

Para la impresión 3D de todas las piezas empleadas y de los machos se ha utilizado la impresora 3D *Artillery Genius Pro* con los siguientes parámetros de impresión.

	PLA (Molde)	PLA (Uniones mecánicas)	PVA
Altura de capa	0.1 mm	0.2 mm	0.1 mm
Relleno	5 %	14 %	0 %
Número de perímetros	2	3	1
Temperatura de extrusión	195 °C	195 °C	210 °C
Temperatura cama	200 °C	200 °C	200 °C
Laca	Si	Si	No

Cuadro 3.2: Parámetros de impresión utilizados

A continuación, se detalla el proceso de diseño y fabricación que se ha seguido hasta llegar a la versión final.

### 3.2.1. Caracterización de los sensores

**Esto falta y no sé si meterlo aquí, meterlo en un anexo o no mencionarlo**

### 3.2.2. Módulos: primera versión

Este primer modelo de los segmentos tiene como principal finalidad encontrar una geometría y número óptimo de vejigas que consigan un movimiento adecuado del segmento al deformarse.

#### 3.2.2.1 Diseño

Partiendo de la idea de que el cuerpo exterior de los segmentos fuera cilíndrico, los primeros parámetros de diseño a establecer fueron la altura y el diámetro exterior. La altura se ha establecido en 100 mm para todos los segmentos para posibilitar el empleo del sensor *Bendlabs Digital Flex Sensor* (figura 3.5). Para el diámetro se tomó como referencia que tuviera una proporción diámetro - altura similar al trabajo realizado en los artículos [25] y [11] estableciéndose de este modo en 47 mm.



Figura 3.5: Bendlabs digital Flex sensor. Fuente: *Bendlabs*

Una vez establecida la geometría primitiva, el siguiente objetivo era establecer la forma de la sección de dicho cilindro. Para esto era necesario establecer la geometría, número y distribución de las vejigas necesarias para la actuación. Uno de los requisitos establece que cada segmento debe tener tres grados de libertad desacoplados actuados independientemente. Acorde con esto se decide que cada segmento tendrá tres *hileras* de vejigas que pueden actuarse de forma independiente, llevando a que los segmentos tengan una simetría ternaria. Finalmente, para permitir que los segmentos tengan la menor anisotropía posible, entendiendo por ello que la resistencia que ofrece la silicona al curvado del segmento sea la misma en todas las direcciones espaciales, se les da una geometría de sectores circulares respecto a dicho eje ternario. Otra motivación para esta geometría de las vejigas es la posibilidad de imprimirlas en 3d sin la necesidad del uso de soportes.

Finalmente, para terminar de acotar la forma básica de la sección, solo resta establecer el grosor mínimo de las paredes, dependiendo este posteriormente de la resistencia mecánica de la silicona. Este se establece de forma arbitraria en 1.5 mm para seguir iterándose posteriormente en las siguientes versiones. En este punto del proyecto se desconocía el diámetro final de los tubos de conexión neumática, por lo que estableció de forma arbitraria en 4.5 mm, pudiendo acoplarse

de este modo tubos de jardinería para realizar las pruebas pertinentes. Por la misma razón se estableció el canal central (a lo largo del eje ternario) como un hexágono que pudiera alojar al sensor de Bendlabs. En la figura 3.6 puede observarse la sección diseñada en este punto del trabajo.

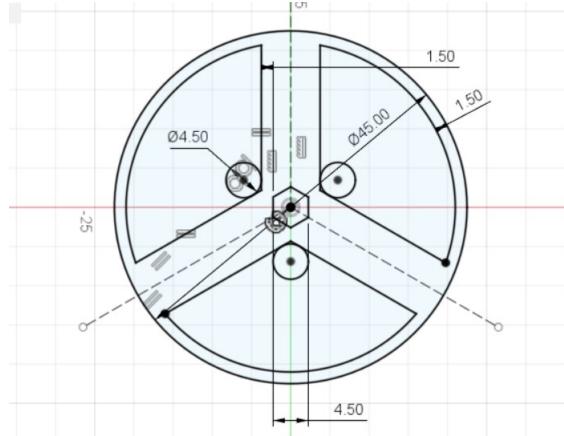


Figura 3.6: Segmentos V1. Sección básica. *Fuente: elaboración propia*

En esta versión, el grosor de las paredes entre vejigas se consideró demasiado grueso. Por ello, para facilitar la deformación en las direcciones de estas paredes se añaden unos huecos que disminuyen la cantidad de silicona y la deformación adecuada de las paredes. Parcialmente, para evitar el problema de deformación presente en el trabajo [11].

La distribución de los sensores elásticos se realiza basándose en los resultados obtenidos en el anexo C, en este caso distribuyendo los sensores en simetría ternaria respecto al eje del cilindro para poder compensar de forma trifásica las histéresis de estos. También se sigue el mismo método de montaje basado en dos moldes usado en C. Se obtiene así la sección mostrada en la figura 3.7.

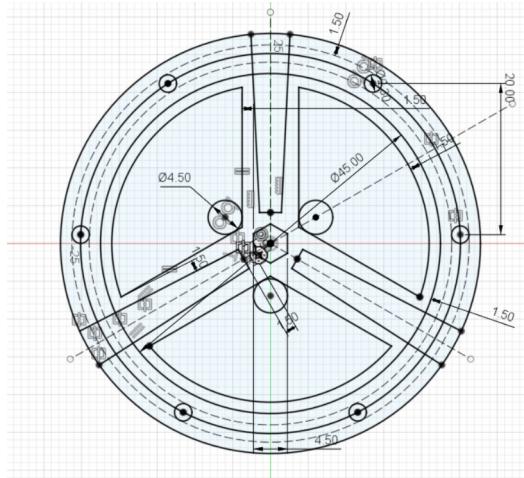


Figura 3.7: Segmentos V1. Sección. *Fuente: elaboración propia*

A continuación, empleando las operaciones correspondientes, se distribuyen volumétricamente estos elementos de la sección a lo largo del eje longitudinal. En este punto, el foco del diseño es la distribución, número, grosor e inter espaciado de las vejigas. Teniendo en cuenta que las hileras de vejigas tendrán una longitud fija, extendiéndose lo máximo posible a lo largo del

eje del cilindro, estas se pueden caracterizar únicamente mediante el espesor y espaciado de las vejigas. En esta primera versión se fija el espesor en 5 mm y el inter espaciado en 3 mm, de forma arbitraria para seguir iterando sobre ello. Además, se añaden los redondeos necesarios y algunos otros elementos auxiliares para disminuir la caída de presión debido a codos pronunciados y para facilitar el diseño del molde.

En este punto se diseñan los dos moldes, el primero para la forma básica del segmento y el segundo para recubrir de silicona los sensores una vez extraídos el modelo del primer molde. En esta versión de los segmentos los moldes consisten en seis piezas más los machos que dan forma a las vejigas. Estas piezas consisten en las tapas superior e inferior, un conjunto de tres piezas que rodean el perímetro del cilindro y un macho rígido que da forma al hueco central.

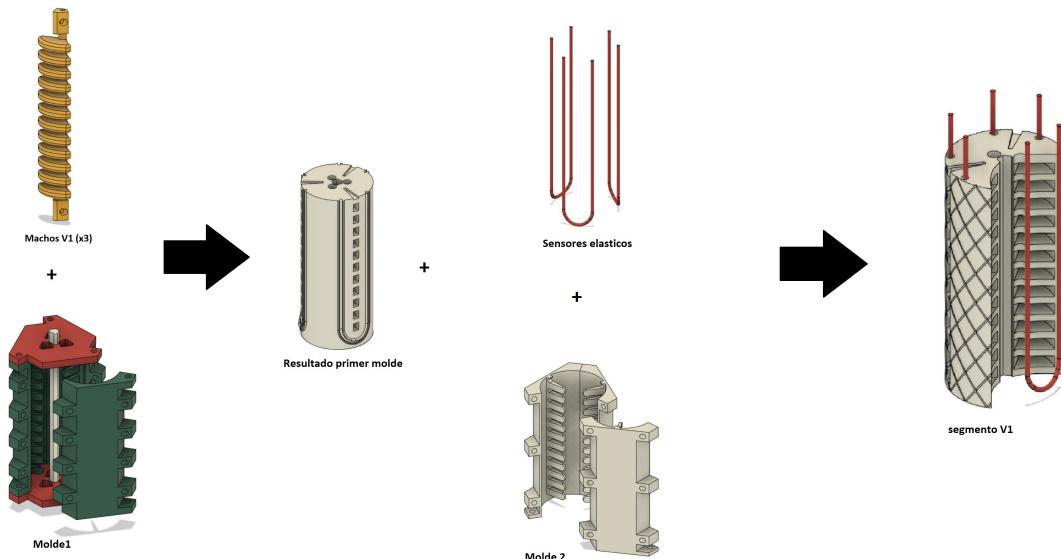


Figura 3.8: Segmentos V1. Moldes. *Fuente: elaboración propia*

En la figura (3.8) se puede apreciar el resultado de este primer diseño y el orden en el que se usan los distintos elementos. A continuación, se resumen los principales parámetros del diseño para poder compararlo con el siguiente.

- Altura = 100 mm
- Diámetro = 45 mm
- Grosor mínimo de las paredes = 1.4 mm
- Espesor vejigas = 5 mm
- Espaciado entre vejigas = 3 mm
- Conformado mediante dos moldes
- Huecos entre hileras de vejigas

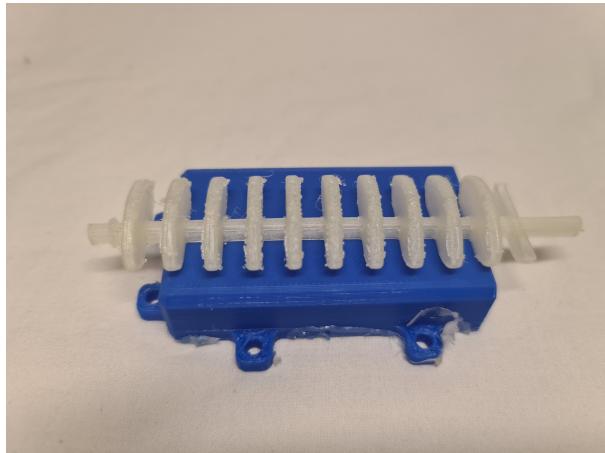
### 3.2.2.2 Conformado

Para la fabricación de esta primera versión se selecciona la silicona *Feroca FS10* de dureza shore A13. Esto motivado debido a su dureza intermedia entre las siliconas blandas posibles a adquirir. Esta silicona posee también como característica diferenciadora una resistencia al desgarro mayor.

Para comenzar el conformado primero se fabrican las piezas del molde mediante impresión 3D (figura 3.9.a)). Posteriormente, se fabrican los machos con la misma máquina, pero empleando esta vez un filamento de PVA (figura 3.9.b)). Una vez se dispone de todos los elementos, se procede al montaje del molde y su posterior sellado. Este sellado impide la posibilidad de que escape parte de la silicona por las juntas debido a posibles imperfecciones en la impresión 3D. Este sellado se realiza mediante un fino cordón de pegamento termofusible, que puede posteriormente se elimina de forma sencilla.



(a) Partes del molde impresas en PLA



(b) Machos impresos en PVA

Figura 3.9: Molde. *Fuente: elaboración propia*

Una vez sellado el molde se procede al vertido de la silicona. Esta se solidifica en un periodo aproximado de 20 minutos. Una vez solidificada la pieza, se desmonta el molde y se extrae el modelo. Este se introduce en una cubeta de agua con una bomba que la mantiene en movimiento, con el objetivo de disolver los machos interiores de PVA. Este proceso dura aproximadamente 24 horas y es necesario el uso de un cepillo para eliminar todos los residuos de PVA.

Una vez terminado el proceso, se procede a colocar los sensores en su lugar correspondiente y a montar el segundo molde para crear un recubrimiento, tal y como se detalla en C. Durante el vertido de la silicona en el segundo molde se pone de manifiesto un problema importante. La silicona cura demasiado rápido, no dando tiempo a llenar todos los huecos estrechos del molde y mostrando un fenómeno de llenado incompleto del molde. En la figura 3.9 puede observarse el módulo terminado. NOTA: foto realizada una vez hechas las siguientes iteraciones.



Figura 3.10: Segmento V1. *Fuente: elaboración propia*

#### 3.2.2.3 Ensayos y conclusiones

Una vez terminado el conformado de este molde se realizan las pruebas correspondientes. Para ello se unen los tubos mediante cianocrilato a la silicona para verificar el inflado de las vejigas. En este punto del trabajo todavía no se dispone del banco de actuación neumático 4, por lo cual se hace uso de un inflador automático para realizar las pruebas de llenado de las vejigas.

Una vez conectado el compresor y puesto en funcionamiento, se pone en evidencia la falta de fuerza de unión entre el modelo y los tubos neumáticos, que tienen que ser unidos forzosamente para mantenerse en su lugar. Una vez unidos los tubos, al hincharse las vejigas se pone de manifiesto el segundo problema, relacionado con la falta de estanqueidad de estas, al inflarse hasta un punto y luego escaparse el aire a presión. Tras investigarlo se encuentra que este problema se debe a dos factores fundamentales.

El primero se debe a que, durante el curado de la silicona, al ser este proceso demasiado rápido, las burbujas formadas en el vertido no disponen del tiempo suficiente para ascender a la superficie libre. Estas quedan embebidas en la silicona, provocando que en algunos puntos las paredes sean lo suficientemente finas para que el aire a presión las rompa y forme una fuga. El segundo está relacionado con el grosor de las paredes, que, debido a su grosor, permiten que se formen algunas fisuras durante el desmoldeo y la limpieza.

Finalmente, se concluye que este modelo presenta unos resultados no satisfactorios con los siguientes problemas a corregir:

- Problema de estanqueidad entre la unión tubo - silicona
- Problema de estanqueidad debido a burbujas formadas durante el curado
- Problema de estanqueidad debido a fisuras provocadas durante el desmoldeo y la limpieza

- Problema de llenado incompleto del segundo molde debido a la relación entre tiempo de curado de la silicona y las cavidades demasiado intrincadas que esta tiene que llenar

### 3.2.3. Módulos: segunda versión

En esta segunda versión el objetivo reside en corregir los problemas existentes en el primer módulo para poder terminar de validar el diseño de las vejigas. Para solucionar el problema del llenado a incompleto del segundo molde, se elimina el sistema de doble molde para unir los sensores.

#### 3.2.3.1 Cambios en el diseño

A continuación, se explican los cambios introducidos en el diseño para solucionar los problemas observados.

Respecto a la unión tubo - silicona, se opta por aumentar la distancia libre de vejigas, dentro de la silicona, para disponer de una mayor superficie de contacto que aumente el rozamiento y que, incluyendo cianocrilato como aglomerante, mantenga los tubos neumáticos en su lugar.

Respecto a la estanqueidad de las vejigas, se aumenta el grosor mínimo hasta 2.5 mm para reforzar las paredes y evitar fisuras. También se eliminan los huecos entre vejigas, al ser una fuente de fisuras.

Para sustituir el sistema de doble molde, en este, se aumenta grosor de los *nervios* para, una vez curado el primer molde, acomodar los sensores y recubrirlos de silicona de un pincel. Se puede observar la diferencia entre las dos versiones en la figura (3.11).

También se refina el diseño de las vejigas, disminuyendo el grosor a 3 mm y manteniendo el espaciado

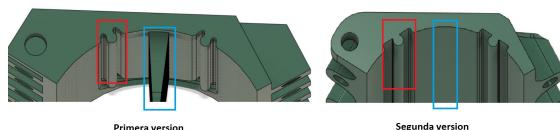
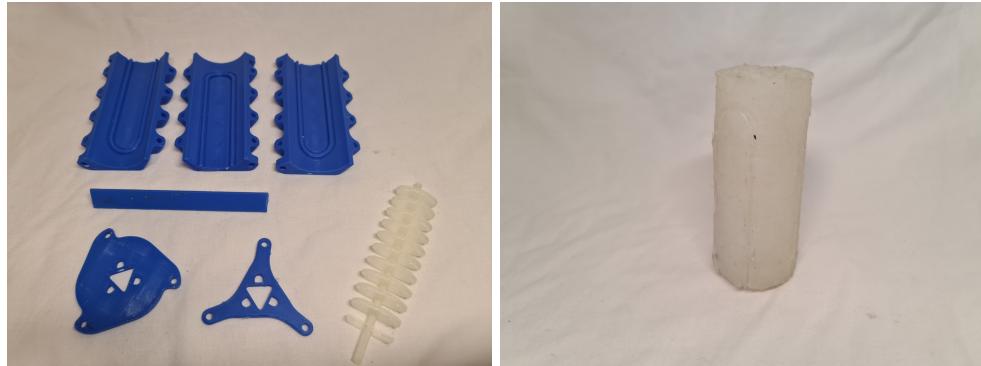


Figura 3.11: Comparación entre el molde de la primera versión y la segunda. *Fuente: elaboración propia*

### 3.2.3.2 Conformado

En este modelo el proceso de conformado es similar al del molde anterior, con la única diferencia en la forma en la que se unen los sensores elásticos al modelo. Se elimina el proceso del doble molde. A partir de este punto, la capa de silicona que mantiene los sensores en su lugar será aplicada mediante un pincel. En la figura 3.12 puede apreciarse el molde utilizado, junto a esta versión de los módulos terminada.



(a) Molde para V2

(b) Segmento terminado

Figura 3.12: V2: Molde. *Fuente: elaboración propia*

### 3.2.3.3 Ensayos y conclusiones

Esta vez se unen los tubos al modelo mediante cianocrilato. Se comprueba que la unión es estanca, pero al poco tiempo de estar las vejigas sometidas a presión, la unión falla y el tubo sale despedido.

Antes de ocurrir este fenómeno, se observa el comportamiento de las vejigas. Estas se inflan, deformando el robot en la dirección buscada. Sin embargo, en dos de los grupos de vejigas vuelven a aparecer los problemas de estanqueidad al poco tiempo.

### 3.2.4. Módulos: tercera versión

En esta tercera versión se introducen numerosos cambios después de observar que los principales problemas iniciales siguen presentes. Se comienza el diseño totalmente desde cero, utilizando la experiencia adquirida en las anteriores versiones. Se crea un fichero paramétrico nuevo en el que se sintetizan en una tabla los principales parámetros que condicionan el diseño, para facilitar iteraciones posteriores a través de la modificación de estos. En la figura 3.13 puede observarse dicha lista de parámetros junto al diseño final de esta versión y su molde.

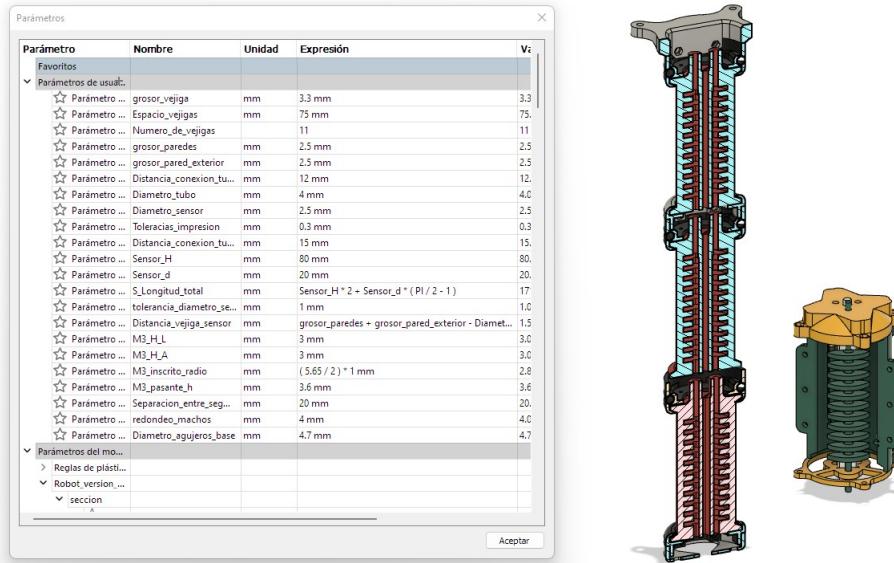


Figura 3.13: Lista de parámetros principales junto al diseño del robot y su molde. *Fuente: elaboración propia*

Para eliminar el defecto de las burbujas se realiza el primer cambio de silicona. Se pasa del modelo PlatSil FS10 al modelo EasyPlat 0030. Esto debido a que esta última tiene un tiempo de curado de 4 horas en vez de 12 minutos, permitiendo que las burbujas asciendan a la superficie. Cabe mencionar que la nueva silicona tiene una dureza SHORE menor, pasándose de shore A13 a shore 0030 (se pueden comparar ambas durezas usando la imagen 3.4).

Además, se diseñan nuevos elementos no presentes anteriormente, que permitan conectar los distintos módulos entre sí.

#### 3.2.4.1 Cambios de diseño en los módulos

A continuación, se explican los distintos cambios de diseño que han sido introducidos en los módulos en sí.

Debido al cambio de silicona, de menor dureza shore, se procede a aumentar de nuevo del grosor de las paredes hasta 4 mm, aumentando el tamaño general de los módulos. Además, se añaden dos nuevos elementos de diseño no presentes en las versiones anteriores. El primero de ellos (figura 3.14.a)), se trata en la adición de unas *rebabas*, cuya función será aumentar la superficie de contacto con los elementos de unión entre módulos que se explicarán en el apartado 3.2.4.2. El segundo cambio es el aumento de superficie módulo-tubo neumático mediante unos salientes

(figura 3.14.b) en los cuales se podrá colocar una abrazadera que asegure la estanqueidad del conjunto, resolviendo los problemas existentes en las versiones anteriores (3.2.3 y 3.2.2).

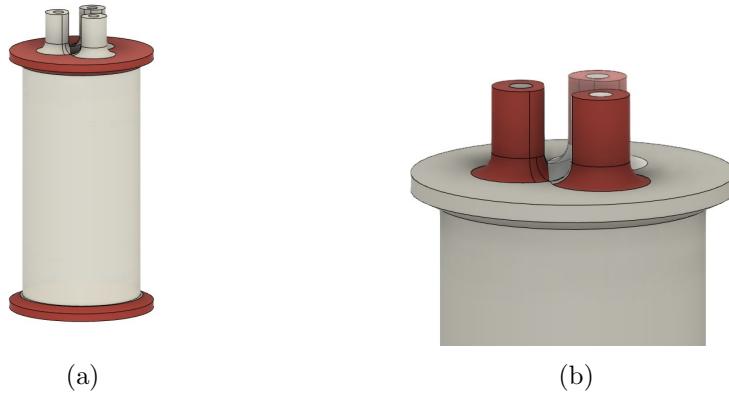


Figura 3.14: Diseño V3. Cambios en los módulos. *Fuente: elaboración propia*

Todos los elementos presentes en los módulos, así como su geometría interna, varían a medida que se modifican los parámetros principales de la tabla de parámetros.

#### 3.2.4.2 Diseño de nuevos elementos

A continuación, se explican el resto de nuevos elementos que se diseñan para satisfacer una de las especificaciones previas no tratadas hasta este punto. Esta se trata de que el robot final presente un diseño modular, permitiendo un montaje sencillo, así como la incorporación de distintos accesorios.

Para esto se diseñan unas piezas que harán de uniones entre los segmentos. Se trata de dos *anillos*, uno macho (3.15.a)) y otro hembra (3.15.b)) que estarán unidos a los extremos de los módulos haciendo que estos encajen entre sí.

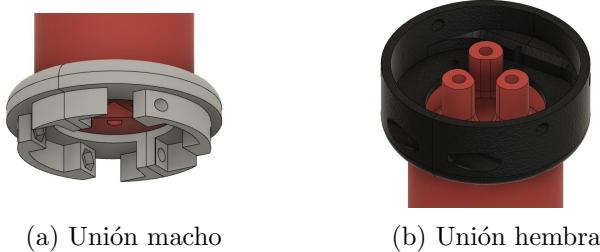


Figura 3.15: Uniones entre segmentos

Cada uno de estos anillos está diseñado para permitir su fabricación mediante impresión 3D en PLA. Además, para permitir su montaje al módulo de silicona, los *anillos* se pueden dividir en dos partes, que se atornillan juntas contra la silicona para realizar una unión por fricción. Estos elementos, además de cumplir con la función de elementos de unión entre segmentos, también hacen que el diseño del robot sea completamente modular, pudiendo añadir distintos accesorios, como el utilaje que puede observarse en la figura 3.16.

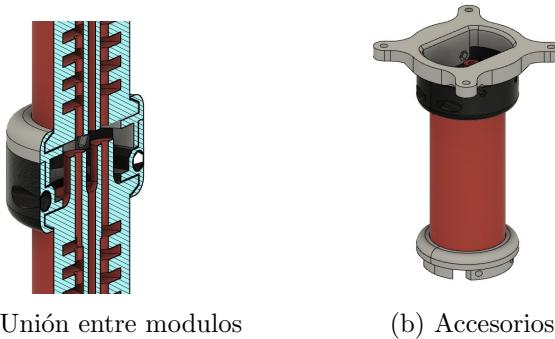


Figura 3.16: Detalle uniones entre segmentos. *Fuente: elaboración propia*

### 3.2.4.3 Conformado y montaje

En esta iteración del diseño de los módulos hay pocas variaciones en lo que al proceso de fabricación se refiere. Se continúa usando un sistema de un solo molde con machos impresos en PVA que son disueltos en una cubeta de agua una vez solidificada la silicona. El único punto para tener en cuenta es que, en este caso, debido al cambio de silicona, se debe tener especial precaución a la hora de limpiar los residuos de PVA, dada la poca dureza de esta. En la figura 3.17 se pueden apreciar las distintas partes del molde.



Figura 3.17: Molde V3. *Fuente: elaboración propia*

Además del segmento de silicona en sí mismo, en esta versión es necesaria la fabricación de los elementos auxiliares anteriormente explicados. En la siguiente figura se puede aprecia el módulo de silicona terminado, al que se le han unido dichos elementos de unión.



Figura 3.18: Segmento V3. *Fuente: elaboración propia*

#### 3.2.4.4 Ensayos y conclusiones

Una vez terminado el proceso de conformado de los módulos de siliconas, se les añaden los tubos neumáticos y se ajustan mediante el sistema de abrazaderas mencionado anteriormente. Por otra parte, en este punto del trabajo ya ha completado el montaje de la estación neumática que se tratará en el capítulo 4: Banco de actuación. Esto permite realizar unos ensayos más detallados.

Cuando se activa el sistema de actuación, se observa que las distintas vejigas se hinchan de forma satisfactoria, concluyendo que los cambios de diseño y de silicona han solucionado los problemas de estanqueidad. Después de terminar con las pruebas de un solo segmento, se procede, por primera vez, a montar el robot completo. Este se puede observar en la siguiente figura (3.19).

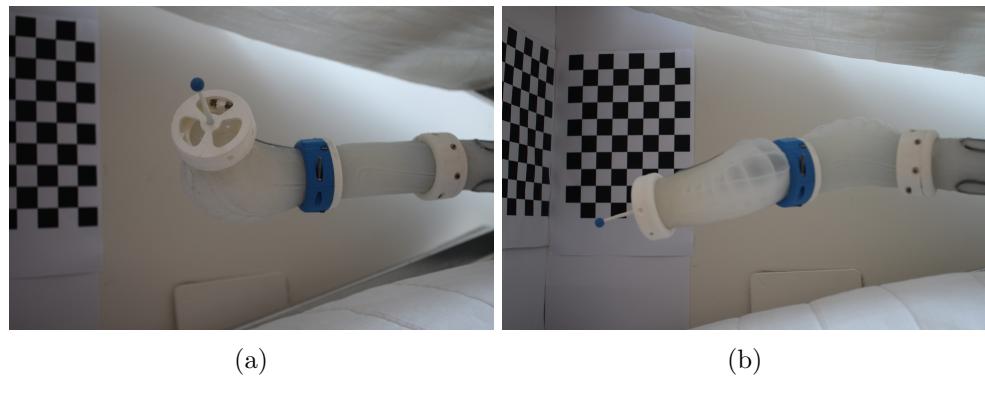


Figura 3.19: V3 final. *Fuente: elaboración propia*

Después de realizar distintas pruebas de funcionamiento se concluye que en esta iteración se satisfacen las especificaciones previamente establecidas. En el siguiente enlace se puede observar un video con la demostración del funcionamiento:

video demostración V1.

Por otra parte, empiezan a manifestarse otros problemas con los que no se contaba en las versiones anteriores. El primero está relacionado con el sistema de fabricación. Debido al material usado para los machos, la fabricación se alarga mucho, dada la gran cantidad de tiempo necesaria

para que estos se disuelvan en agua. Además, el PVA deja un residuo en los segmentos que es necesario eliminar con un cepillo, el cual, dada la poca dureza de la silicona, puede generar alguna fuga en estos.

Otro problema está relacionado directamente con el tipo de silicona escogida. Al ser tan blanda, requiere de unas paredes muy gruesas. Esto aumenta mucho el peso de los segmentos (**198 g**). Esto es un gran problema, ya que cada segmento tiene que aguantar el peso de los siguientes, lo cual sumado a las características físicas de la silicona, hace que el robot no se doble todo lo esperado (al menos el primer segmento, el que tiene que soportar más peso). Además del peso, la baja dureza shore de la silicona también provoca que los segmentos se hinchen más de lo esperado. A vista de todos estos factores se pueden resumir las conclusiones de esta iteración en los siguientes puntos:

- Mantener diseño y solo cambiar algunos parámetros. Se concluye que la geometría es correcta y que los sistemas de unión funcionan como se espera
- Necesidad de cambiar a una silicona más dura
- Necesidad de cambiar de sistema de fabricación para eliminar el uso de los machos de PVA
- Reducción de peso

### 3.2.5. Version final

En esta iteración, la cual será la versión final de este trabajo, se prioriza principalmente optimizar la versión anterior, incidiendo en los puntos que se han comentado en las conclusiones.

Respecto al sistema de fabricación, se mantiene el concepto de utilizar unos machos que puedan extraerse en fase líquida, pero se cambia el material del que estos están fabricados. Se pasa a utilizar cera de parafina, inspirado en los métodos de fabricación a la cera perdida.

El uso de este material para los machos conduce a un nuevo problema: la incompatibilidad con el tipo de silicona utilizada hasta este punto. Las siliconas utilizadas anteriormente consistían en siliconas de platino, que se basan en una reacción química de adición, la cual es inhibida en presencia de cera de parafina. Por ello se busca una alternativa, y para esta versión se pasa a una silicona de estaño, en concreto la *Feroxa TinSil 80-15*, de dureza **shore A15**. Esta ofrece una dureza ligeramente superior a la utilizada en V1 y V2.

#### 3.2.5.1 Cambios de diseño

En esta versión no se realizan cambios de diseño mayores. Principalmente, se ajustan algunos parámetros con el objetivo de disminuir el tamaño de las paredes, para aprovechar la mayor dureza de la silicona, y de esta manera reducir el peso. En la imagen 3.20 se pueden observar los parámetros que se han utilizado finalmente.

Nombre	Unidad	Expresión
grosor_vejiga	mm	3.3 mm
Espacio_vejigas	mm	75 mm
Numero_de_vejigas		11
grosor_paredes	mm	2.5 mm
grosor_pared_exterior	mm	2.5 mm
DistanciaConexion_tubos_interior_siloconas	mm	12 mm
Diametro_tubo	mm	4 mm
Diametro_sensor	mm	2.5 mm
Toleracias_impresion	mm	0.3 mm
DistanciaConexion_tubos_exterior_bridas	mm	15 mm
Sensor_H	mm	80 mm
Sensor_d	mm	20 mm
S_Longitud_total	mm	Sensor_H * 2 + Sensor_d * (PI / 2 - 1)
tolerancia_diametro_sensor	mm	1 mm
Distancia_vejiga_sensor	mm	grosor_paredes + grosor_pared_exterior - Diametro_sensor - tolerancia_dia...
M3_H_L	mm	3 mm
M3_H_A	mm	3 mm
M3_inscrito_radio	mm	(5.65 / 2) * 1 mm
M3_pasante_h	mm	3.6 mm
Separacion_entre_segmentos	mm	20 mm
redondeo_machos	mm	4 mm
Diametro_agujeros_base	mm	4.7 mm

Figura 3.20: Parámetros finales. *Fuente: elaboración propia*

Con respecto a la reducción de peso, además de disminuir el grosor de las paredes, se ha aumentado el número de vejigas para disminuir la densidad de los módulos. Con todos estos cambios, la herramienta *Fusion 360*, nos calcula un volumen de silicona de segmento de  $1,314 * 10^5 \text{ mm}^3$ . Por otra parte, gracias a la datasheet sabemos que la silicona tiene una densidad de  $1.1 \text{ Kg/m}^3$ . Por tanto, el peso predicho será:

$$\text{masa}_{\text{predicida}} = 1.1[\text{Kg/m}^3] * 1,314 * 10^{-6} * 10^5[\text{m}^3] = 0.1445[\text{Kg}] = 144.54[\text{g}] \quad (3.1)$$

Esto supondría una reducción del peso del 27.2 %

### 3.2.5.2 Conformado y montaje

En esta iteración del robot se cambia bastante el sistema de fabricación. El concepto es similar, pero los machos pasan de estar fabricados en PVA a estar fabricados de cera de parafina, para posteriormente, en vez de eliminarse mediante agua, eliminarse mediante calor. Para comenzar la fabricación, es necesario, previamente, fabricar unos moldes de silicona con la geometría de los machos para poder elaborar las piezas de cera. Para esto, mediante impresión 3D, se fabrica el *positivo* de los machos, que posteriormente se une a un pequeño recipiente. Una vez montado este conjunto, se vierte la silicona para crear el molde. En la figura 3.21 puede apreciarse el positivo de los machos junto al molde negativo elaborado.



Figura 3.21: Robot final

A continuación, ya se puede pasar al proceso de elaboración de los segmentos. Este proceso está explicado en este video, y se puede resumir en los siguientes pasos:

### 1. Fabricación de los machos de cera

Se vierte silicona en los moldes de silicona de los machos y se deja solidificar durante aproximadamente media hora. Para la fundición de la cera se utiliza una ola con la cual se funde la cera en aproximadamente diez minutos



Figura 3.22: Versión final. Fab 1

### 2. Montaje del molde

Se montan todas las piezas del molde, los machos de cera y se sella mediante un cordón de silicona caliente para evitar fugas de silicona, dado su largo tiempo de curado. A la hora del montaje, es importante revisar que no queden partículas de cera que hayan podido desprendérse de los machos. Además, previo al vertido de la silicona, se introduce el montaje en el congelador durante aproximadamente diez minutos para asegurar que los machos de cera sean los más sólidos posibles.



Figura 3.23: Versión final. Fab 2

### 3. Vertido de la silicona

En este paso se realiza la mezcla de la silicona y se procede al vertido. Debido a la toxicidad de la silicona de estaño, se da una especial importancia a las medidas de seguridad tales como ventilación o gafas de seguridad. Se llena el molde hasta que rebose ligeramente para compensar la contracción que se produce en la silicona durante el curado.



Figura 3.24: Versión final. Fab 3

### 4. Curado

Durante este proceso se deja curando la silicona durante mínimo **24 H**, en una estancia ventilada dada la emisión de alcoholes que se produce durante el proceso de vulcanización al usarse silicona de estaño.

### 5. Desmoldeo

En este punto se desmolda el bloque de silicona ya curado. Se utiliza un bisturí para romper los cordones de silicona mencionados anteriormente y facilitar el desmontaje del molde. El mismo bisturí se utiliza para cortar los excesos y rebabas de la silicona



Figura 3.25: Versión final. Fab 4

#### 6. Eliminado de cera

Este paso es necesario para eliminar los machos de cera interiores. Este proceso se lleva a cabo en dos pasos, que han sido inspirados en el trabajo realizado en la referencia [25];

Primero se introducen los modelos de silicona en un horno a 110 grados. La mayoría de la cera precipitará en un recipiente que se encuentra debajo. Para esto se ha hecho un sencillo utillaje con alambre de acero, que sostiene el módulo a la vez que permite el precipitado de la silicona

Después se da un baño en agua hirviendo durante unos 15 minutos que ayudará a eliminar cualquier resto de cera restante.



(a) Horno (b) Agua hirviendo

Figura 3.26: Versión final. Fab 5

#### 7. Sellado de los orificios de drenado

En este punto se sellan los orificios por los que ha drenado la cera en el paso anterior. Esto se hace por medio de la pieza mostrada en la figura 3.27. Sobre este *plato* se vierte un poco de silicona, posteriormente se monta sobre el molde y se deja solidificar durante **24 horas**.



Figura 3.27: Versión final. Fab 6

### 8. Montaje de los tubos neumáticos.

Finalmente, para terminar el conformado de un segmento, se unen los tubos neumáticos. Primero se da una pincelada de cianocrílico para asegurar la estanqueidad y después se ponen unas bridas para asegurar que los tubos no se salgan.



Figura 3.28: Versión final. Fab 7

#### 3.2.5.3 Ensayos y conclusiones

Una vez fabricado y ensamblado este modelo, se procede a realizar las pruebas necesarias para comprobar su correcto funcionamiento. Para ello se realiza el ensamblado del *SoftBot* completo y se conectan los tubos neumáticos al banco de actuación (capítulo 4).

Para verificar que se cumplen las especificaciones requeridas en el alcance de este proyecto, se hincha cada grupo de vejigas hasta la presión máxima que este admite. Se realizan prueba inflando cada grado libertad de distintas maneras. Se prueba a inflar cada hilera de vejigas de forma individual y también combinándolas entre ellas, para verificar que el robot se mueve entre posiciones intermedias de forma adecuada.

A continuación, en la figura 3.29, pueden apreciarse algunas de las poses que se han utilizado para realizar las pruebas.

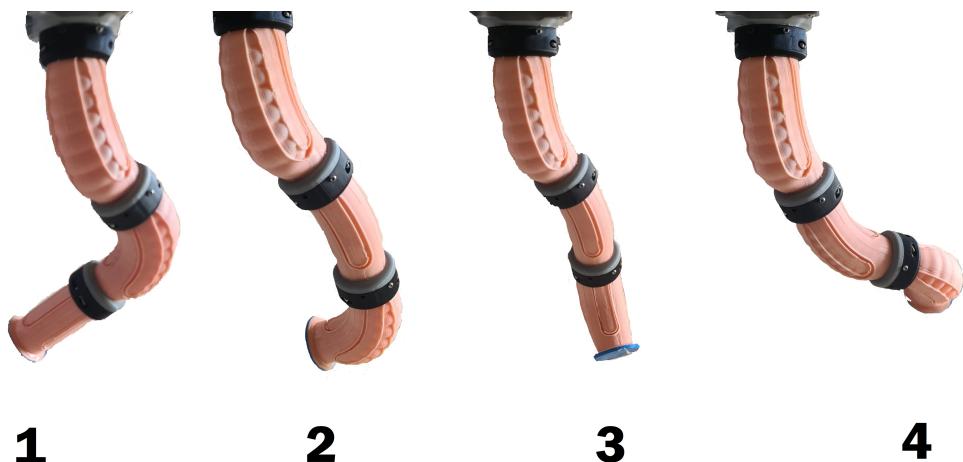


Figura 3.29: Robot final. *Fuente: elaboración propia*

Teniendo estos resultados en cuenta se puede concluir que esta será la versión final del robot con la que trabajará durante el resto del proyecto.

## 4. Banco de actuacion

Una parte relevante de este proyecto trata sobre la forma en la que se implementa la actuación del robot. Al haberse escogido la actuación neumática, aparte de la construcción de un prototipo funcional de silicona, es necesario disponer de un banco de actuación, que permita obtener distintas líneas controladas de aire comprimido. En este capítulo se tratará la estación neumática utilizada, así como la interfaz a través de la cual esta se comunica con el software.

Dentro de la actuación neumática se tuvo que escoger entre dos opciones: la actuación neumática diferencial o la actuación neumática directa, ambas explicadas en el punto 2.2: tipos de actuadores, en el estado del arte. Se ha escogido la neumática directa dada la disponibilidad del banco neumático fruto del trabajo de Ángela Serrano [3] y Víctor Barroso [13], que se puede apreciar en la figura (4.1).

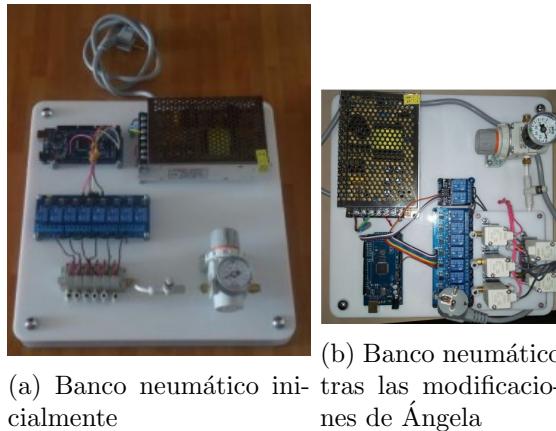


Figura 4.1: Banco de actuación de partida. *Fuente: a) [13], b) [3]*

### 4.1. Punto de partida y objetivos

El banco neumático utilizado se trata de un conjunto de electroválvulas y los elementos neumáticos necesarios, que permiten controlar un caudal de aire comprimido proveniente de un compresor, de acuerdo con unas señales de mando. Sin embargo, para cumplir las especificaciones necesarias del robot de este trabajo, ha sido necesario introducir ciertas modificaciones, como se explica en el siguiente punto.

Como elementos de partida, tras las modificaciones introducidas por Ángela Serrano, disponía de seis válvulas neumáticas 3/2, de las cuales cinco de ellas estaban dispuestas en serie con válvulas 2/2 y una de las válvulas 3/2 estaba sin utilizar. Esta configuración permite commutar cinco líneas neumáticas entre los estados de abierto al aire, a presión y cerrado, ocupándose las válvulas 3/2 de commutar entre los primeros dos estados y las válvulas 2/2 del estado cerrado (figura 4.2).

Además, el banco disponía de los elementos neumáticos auxiliares necesarios, tales como un presostato, un regulador de caudal y los racores necesarios.

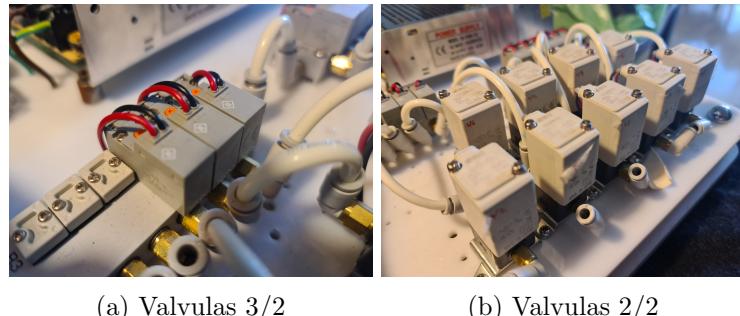


Figura 4.2: Válvulas neumáticas montadas en el banco de actuación. *Fuente: elaboración propia*

La señal eléctrica necesaria para conmutar las válvulas provenía de un banco de relés, los cuales hacían de interfaz entre la línea de tensión de 24v necesaria para el accionamiento de las válvulas y las señales de control de 5v provenientes de un microcontrolador. La línea de potencia de 12 V proviene de una fuente de alimentación de 102 W. En la siguiente tabla se muestra un resumen de las principales especificaciones de partida.

Cuadro 4.1: Banco neumático: especificaciones iniciales

Número de válvulas 2/2	5
Número de válvulas 3/2	6
Actuación de las válvulas	Mediante relés
Fuente de alimentación	102w (12 V, 8.5 A Max)
Presión máxima de trabajo	0.4 MPa (Impuesto por las válvulas 2/2)
Máximo número de líneas a controlar	5

Cuadro 4.2: Especificaciones iniciales del banco de actuación

Conocidas las especificaciones iniciales es necesario establecer las especificaciones requeridas para poder planificar las modificaciones a realizar al banco neumático.

Para este proyecto es necesario un banco neumático con nueve líneas neumáticas como mínimo, pero se dimensionará para doce para cubrir las necesidades de trabajos futuros. Respecto a la presión de trabajo, se realizan los ensayos oportunos que permiten establecer que esta será de un orden de magnitud menor, por lo que no representa una incompatibilidad con los elementos neumáticos existentes. Acorde con esto, se puede resumir que los cambios a realizar tienen que satisfacer los siguientes puntos

- Aumento de las líneas de neumáticas de cinco a nueve
  1. Dimensionamiento de elementos neumáticos
  2. Dimensionamiento de elementos electrónicos
- Elección de un microcontrolador e interfaz de comunicación

## 4.2. Cambios introducidos

A continuación, se explican de una forma más detallada los distintos cambios introducidos

#### 4.2.1. Dimensionamiento de elementos neumáticos

Para satisfacer la especificación de disponer de 9 líneas neumáticas para actuar los 9 grados de libertad del robot, es necesario añadir más válvulas. En la disposición de partida podían encontrarse cinco válvulas 2/2 (figura 4.3.a)) y seis válvulas 3/2 (figura 4.3.b)). Estas últimas precisan ser montadas en una placa base, que en este caso dispone de seis bahías, por lo que se tomó la decisión de adquirir otro similar para aprovechar el existente y, además, de esta manera, facilitar la opción de ampliar el banco a 12 líneas en un futuro.



Figura 4.3: Válvulas neumáticas. Modelos. *Fuente: SMC*

También se adquieren los racores necesarios (M4 a tubo 4 mm), así como los tubos neumáticos. De este modo, el conjunto neumático dispone de una presión máxima de trabajo de 0.4 MPa, limitado por las válvulas 2/2.

Otra cuestión que tuvo que ser solucionada era encontrar una fuente que supla del aire comprimido. Para ello se compararon distintos modelos de compresor y finalmente se escogió el modelo STANLEY FATMAX DST 101/8/6.



Figura 4.4: Compresor: STANLEY FATMAX DST 101/8/6. *Fuente: Stanley*

A continuación, en la figura 4.5 puede observarse el esquema neumático completo

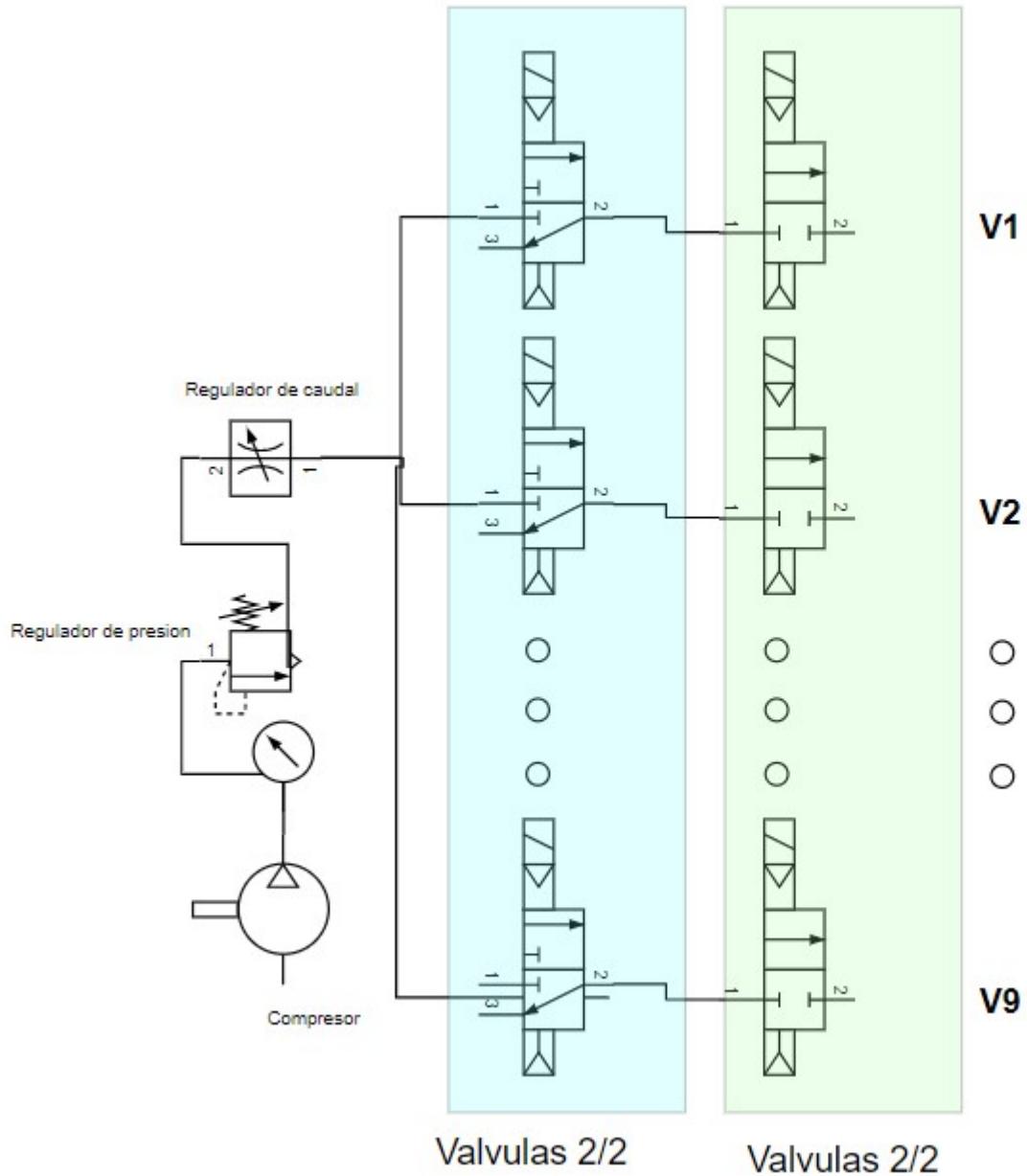


Figura 4.5: Esquema neumático. *Fuente: elaboración propia*

#### 4.2.2. Dimensionamiento de elementos electrónicos

Respecto al diseño electrónico, el primer cálculo que se realiza tiene como objetivo comprobar si la fuente de alimentación dispone de la potencia suficiente para actuar las 18 electroválvulas. Para ello se calcula la corriente máxima que pueden consumir las electroválvulas. Para ello se recurre a la datasheet. Se adjuntan los cálculos a continuación.

Válvula	Potencia	Corriente consumida( $\frac{P_{valvula}}{I_{valvula}} * N_{valvulas}$ )
Válvulas 2/2 (SMC VDW20BZ1D)	3 W	2.25 A
Válvulas 3/2 (SMC Y100)	0.5 W	0.375 A
	Corriente total	3

Cuadro 4.3: Cálculo de corrientes en banco de actuación

Por tanto:

$$I_{total\_valvulas} = 2.625 \ll I_{fuente} = 8.5A \quad (4.1)$$

De esta manera se establece que la fuente no tiene falta de potencia, y además se obtiene la corriente que circulará por los cables de alimentación lo que permitirá dimensionar su diámetro.

A continuación, es necesario seleccionar un microcontrolador adecuado para la tarea a realizar. Los factores para tener en cuenta son los siguientes:

- Mínimo nueve conversores analógico-digitales (ver anexo C)
- Dieciocho salidas digitales ( $2 * 9$  válvulas)
- Suficientes counters para poder gestionar las válvulas
- Dos puertos UART

Atendiendo a estas características se escoge la placa Arduino Due (figura 4.6), dadas sus características de E/S y su procesador de 32 bits.



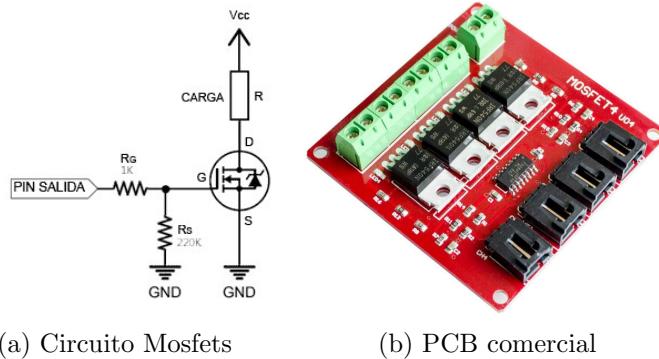
Figura 4.6: Microcontrolador utilizado: Arduino Due. *Fuente: RS*

Por otra parte, otra posible complicación está relacionada con los relés. Estos requieren una corriente para poder conmutar y esta, al aumentar los relés a 18 (1 por cada válvula a actuar), es mayor que la máxima.

$$I_{rele} = 80mA \Rightarrow I_{total\_reles} = 18 * I_{rele} = 1.44A \quad (4.2)$$

Además, gracias a la datasheet sabemos que la corriente máxima que puede suministrar la placa es  $I_{max\_arduino} = 800mA = 0.8A < I_{total\_reles}$  por lo que es necesario encontrar una alternativa al uso de los relés. Estos se sustituyen por transistores MOSFET, en concreto el modelo **IRF540** que cumple con las especificaciones requeridas. Se implementa mediante el circuito mostrado en la figura 4.7.a), pero debido a limitaciones temporales se adquiere en forma de una placa comercial

(figura 4.7.b)). Cada una de estas placas incluye cuatro mosfets, por lo que se adquieren 5 placas en total (capacidad para controlar hasta 20 válvulas). Estas placas de mosfet, además, incluyen un diodo flyback, el cual es idóneo para el trabajo, puesto que las bobinas que controlan las electroválvulas actúan como una carga inductiva



(a) Circuito Mosfets

(b) PCB comercial

Figura 4.7: Circuito utilizado en la interfaz de potencia del banco de actuación. *Fuente: a) luisllamas.es b) amazon.es*

#### 4.2.3. Pinout final

Una vez dimensionados los elementos neumáticos y electrónicos es necesario el correcto montaje físico de estos en el banco. Las 5 placas de mosfets se montan en el interior del banco de actuación y el resto de los elementos encima de este. La disposición de los elementos encima del banco se puede observar a continuación en la figura 4.8

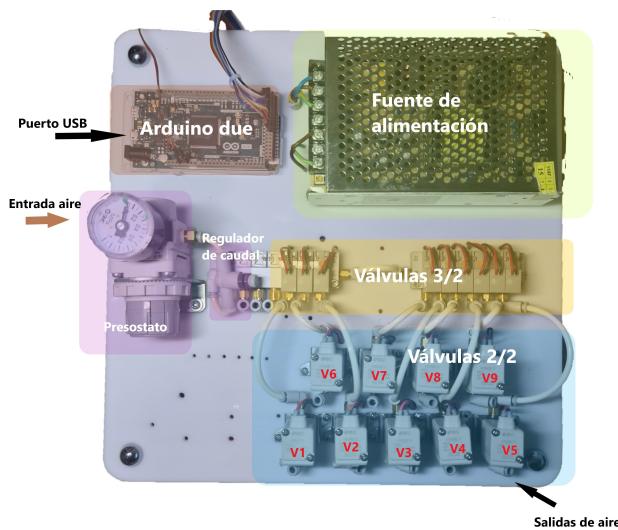


Figura 4.8: Banco de actuación: layout final. *Fuente: elaboración propia*

La gestión de los cables en el interior del banco neumático también ha supuesto un reto. La conexión entre el microcontrolador y la interfaz de potencia (las placas de mosfets) se realiza mediante dos cables, a los cuales se ha denominado **J1** y **J2**. Uno de ellos es de ocho pines y el otro de diez, y juntos se encargan de transportar las señales de control de las electroválvulas. Además de las señales de control, las placas de mosfet requieren una alimentación de baja tensión

que proviene de la toma de 3.3 V de la placa Arduino Due. Aparte de estas señales, al microcontrolador también se conecta la baliza luminosa, de la cual se hablará en el capítulo 5: Banco de medición. A continuación, en la figura 4.9 se ilustran dichas conexiones en el microcontrolador.

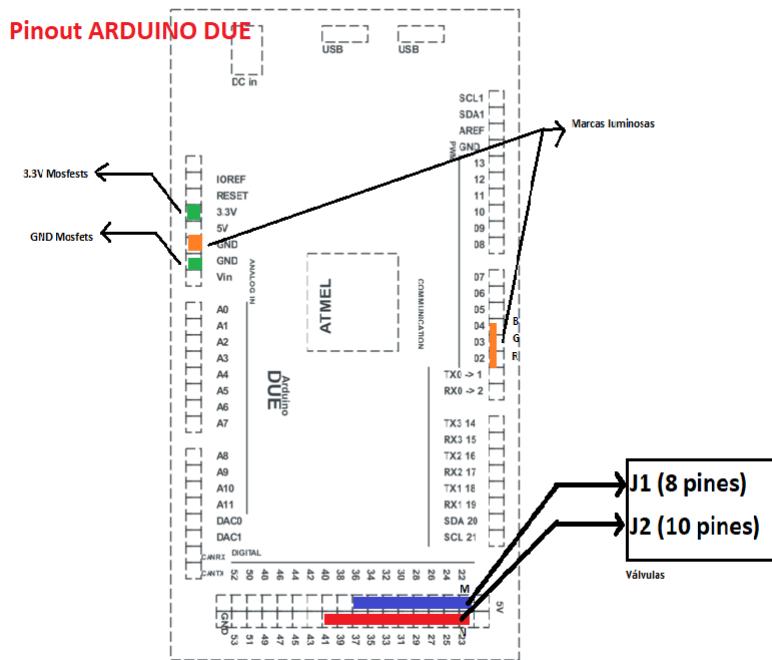


Figura 4.9: Conexiones Arduino Due. Fuente: adaptación de datasheet

Por otra parte, se encuentra la conexión entre la interfaz de potencia y las dieciocho electroválvulas. Para ello se agrupan los cables provenientes de las válvulas en cuatro grupos de cuatro y uno de dos, y se conducen hacia las salidas de las cinco placas de mosfets. Toda la relación de conexiones se puede consultar en la siguiente tabla:

	Conector J1 (8 pines)							
Color del cable	Marron	Rojo	Naranja	Amarillo	Verde	Azul	Morado	Gris
Válvula [ tipo (número) ]	32 (8)	32 (4)	32 (5)	32 (9)	32 (6)	32 (3)	32 (2)	32 (7)
Placa de mosfet	T1				T2			
Pin arduino	22	24	26	28	30	32	34	36

	Conector J2 (10 pines)									
Color del cable	Marron	Rojo	Naranja	Amarillo	Verde	Azul	Morado	Gris	Blanco	Negro
Válvula [ tipo (número) ]	22 (3)	22 (6)	22 (7)	22 (2)	22 (8)	22 (9)	22 (4)	22 (5)	32 (1)	22 (1)
Placa de mosfet	T3				T4				T5	
Pin arduino	41	39	37	35	33	31	29	27	25	23

Cuadro 4.4: Pinout

### 4.3. Arquitectura software

Para el control preciso del banco neumático se ha desarrollado una interfaz software de bajo nivel que realizará la comunicación entre el banco de actuación y el código de alto nivel en MATLAB. Este código es el que se encontrará cargado en el microcontrolador del banco neumático.

Para el desarrollo de este código se ha utilizado la herramienta *Platform IO* y se ha programado usando el lenguaje de programación *C++*.

#### 4.3.1. Explicación conceptual

A continuación, se va a resumir el funcionamiento de las principales funciones del código del banco de actuación. Este está basado en una rutina principal que se repite cíclicamente y se encarga de gestionar las siguientes tareas:

- Comunicación a través de UART
- Computo de las tareas de las válvulas
- Gestión de tareas de seguridad

Estas tareas tienen acceso a nueve objetos Válvula que se instancian al principio del programa.

```
1 // Array de valvulas
2 Valvula *misValvulas [NUM_VALVULAS];
```

##### 4.3.1.1 La clase Valvula

Para entender el resto del programa, primero se va a explicar cómo funcionan dichos objetos válvula. La clase de la cual se instancian estos objetos, cuya interfaz en UML es la mostrada en la figura 4.10, tiene como objetivo abstraer a nivel software cada conjunto de válvula 3/2 y 2/2 como una sola válvula 5/3. En otras palabras, esta clase permite tratar cada una de las nueve líneas neumáticas del banco como una sola válvula de tres posiciones, en vez de dos posiciones. Las tres posiciones de esta válvula son las correspondientes a:

1. línea al aire (mediante el método alAire() de la clase)
2. Línea conectada a presión (mediante el método presión())
3. Línea cerrada, para mantener la presión (mediante el método cerrada())

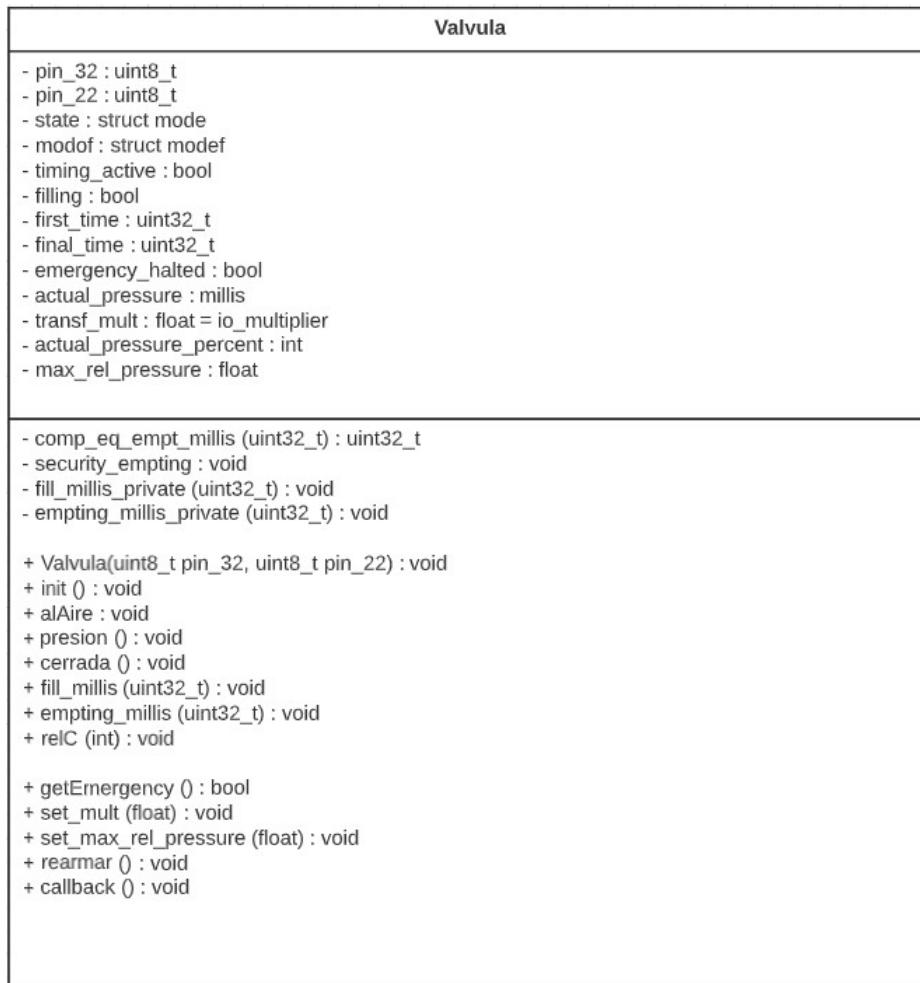


Figura 4.10: Interfaz UML de la clase valvula.

Además de estos tres métodos principales, un objeto válvula también puede responder a los siguientes mensajes:

- **fill\_millis(x):** Conecta la línea a presión durante x milisegundos, y posteriormente se cierra, manteniendo la presión.
- **emptying\_millis(x):** Deja al aire la línea durante x milisegundos, y posteriormente se cierra, manteniendo la presión
- **relC(x):** Enfocado al control de robots blandos. Se establece un tiempo máximo de llenado mediante el método *set\_max\_rel\_pressure()*. Posteriormente, al llamar al método, se hinchará el grupo de vejigas un x por ciento, siendo el 100 % el tiempo máximo de hinchado impuesto. El objeto mantendrá la cuenta de cuantos milisegundos ha hinchado y deshinchado. El objetivo de este método es poder controlar los actuadores de robots blandos mediante un *porcentaje de actuación*, en vez de por tiempos de llenado en bruto. Si a un objeto válvula se le pide que disminuya el porcentaje de presión a uno menor que el anterior, se pondrá en modo al aire, durante la cantidad de tiempo que lo haría a igual cambio de porcentaje al llenarse, multiplicado por un factor. Este factor se establece mediante el método *set\_mult()*, y existe debido a la diferencia de caudal de llenado y vaciado del banco.

También se integran en la clase funciones relacionadas con la seguridad. De esta manera, la válvula, siempre que se llame cíclicamente en el programa principal al método `callback()`, mantendrá la cuenta de milisegundos hinchados. Si estos superan un umbral máximo, definido en la interfaz de la clase, la válvula se pone en un *modo de emergencia*. En caso de que esto ocurra, el método `getEmergency()` devolverá 1, devolviendo normalmente 0 cuando la válvula no está en modo de emergencia. El método `rearmar()` devuelve la línea neumática a su estado normal.

#### 4.3.1.2 Comunicación UART

Una vez explicado el funcionamiento de las principales características de la clase válvula, procedemos a explicar el funcionamiento del resto del programa

Para la comunicación UART, en cada ciclo de ejecución del programa se consulta si se ha recibido un mensaje nuevo. La estructura de la trama es de la siguiente forma:

$$[C_{Op} \ Arg_1 \ Arg_2 \ \dots \ Arg_n] \quad (4.3)$$

Con  $n = 1, 2, \dots, k$ , siendo  $k$  el número de argumentos. El número de argumentos, así como el tipo de estos, dependen del código de operación introducido. El mensaje será en formato *string* y los argumentos separados por algún carácter especial. Los códigos de operación que se han implementado son los siguientes:

- $C_{Op} = "a"$ : Pone una válvula en modo al aire. La válvula se especifica en el argumento a continuación.
- $C_{Op} = "b"$ : Pone una válvula en modo cerrada. La válvula se especifica en el argumento a continuación.
- $C_{Op} = "c"$ : Pone una válvula en modo a presión. La válvula se especifica en el argumento a continuación.
- $C_{Op} = "f"$ : Este código de operación va seguido de dos argumentos de tipo *int*. El primero establece cuál de las nueve válvulas se va a conectar a presión durante los milisegundos indicados en el segundo argumento.

$$[f \ num_{valvula} \ t_{llenado}] \quad (4.4)$$

- $C_{Op} = "e"$ : Similar al código anterior, pero poniendo la línea en modo al aire.
- $C_{Op} = "w"$ : Este comando sirve para controlar todas las válvulas con un solo mensaje. En este caso la trama será de la siguiente manera:

$$[w \ modo \ argv_1 \ argv_2 \ \dots \ argv_9] \quad (4.5)$$

El argumento *modo*, establece si se desea controlar la posición de las válvulas (*modo* = 0) o tiempos de llenado/ vaciado (*modo* = 1). En el primer caso, los argumentos que siguen controlan la posición de cada válvula ( $argv_n = 1$ : presión,  $argv_n = 0$ : cerrada,  $argv_n = -1$ : alAire). En el segundo caso, los argumentos establecen el tiempo de llenado o vaciado, llenándose en caso de ser un número positivo, y vaciando en caso de ser negativo.

- $C_{op} = "v"$ : Cambia el brillo de las tres luces de la marca luminosa que se explicará en el capítulo 5. Se establece el valor de cada una en tres argumentos, siendo la trama:

$$[v \ r \ g \ b] \quad (4.6)$$

- $C_{op} = "l"$ : Cambia la presión máxima del modo de control relativo (relC)

$$[l \ p_{max}] \quad (4.7)$$

- $C_{op} = "m"$ : cambia el multiplicador de hinchado/ deshinchado (*set\_mult()*).
- $C_{op} = "x"$ : Para controlar una válvula en modo relativo. En el segundo argumento se establece el porcentaje de llenado
- $C_{op} = "R"$ : Rearma todas las válvulas. Sin argumentos adicionales

#### 4.3.1.3 Otras tareas del programa

A parte de la comunicación UART, el programa del banco de actuación también se encarga de tareas relacionadas con el control de las válvulas y tareas relacionadas con la seguridad. Para el primer punto, aparte de gestionar la comunicación serial, en cada ciclo de ejecución, se llama al método *callback* de todos los objetos válvula.

Para la segunda tarea, se ha implementado un sistema basado en una máquina de estados muy simple que dispone de dos estados. Normalmente, el banco neumático se encuentra en modo normal de funcionamiento. En caso de que se detecte que alguna de las válvulas ha estado en modo a Presión, más tiempo del máximo establecido, el banco entrará en modo de emergencia.

En este modo, todas las válvulas se ponen en modo alAire y el banco deja de aceptar las órdenes que acepta normalmente por uart. En caso de mandarse el comando de rearme, especificado en el apartado anterior, el banco volverá a su estado normal de funcionamiento.

#### 4.3.2. Resumen de ficheros y funciones

##### Hablo de los distintos ficheros???

```

1 #ifndef _PINOUT_H_
2 #define _PINOUT_H_
3
4 #include "Config.h"
5 #define NUM_VALVULAS 9
6
7 // ##### Valvulas #####
8
9 // valvulas 3/2
10
11 const uint8_t PIN_32_ARRAY[NUM_VALVULAS] =
12 {
13     25, // 1
14     34, // 2
15     32, // 3
16     24, // 4
17     26, // 5
18     30, // 6

```

```
19      36, // 7
20      22, // 8
21      28 // 9
22  };
23
24
25
26 // valvulas 2/2
27
28     const uint8_t PIN_22_ARRAY[ NUM_VALVULAS ] =
29     {
30         23, // 0
31         35, // 1
32         41, // 2
33         29, // 3
34         27, // 4
35         39, // 5
36         37, // 6
37         33, // 7
38         31 // 8
39     };
40
41     const uint8_t EMRGY_PIN = 9;
42     const uint8_t LED_PIN = 0;
43
44     const uint8_t PIN_LED_R = 2;
45     const uint8_t PIN_LED_G = 3;
46     const uint8_t PIN_LED_B = 4;
47
48
49
50
51 #endif // _PINOUT_H_
```

## 5. Sistema de adquisición de datos

Para el correcto desarrollo de este proyecto es necesario disponer de un sistema de medición objetivo, al cual nos referiremos a partir de este punto como *banco de medición*. Este consistirá en un conjunto de elementos que permitan rastrear la posición del extremo del robot respecto a un sistema de referencia fijo en el espacio.

El sistema tiene que ser capaz de reconocer la posición del extremo del robot en cualquier pose que este pueda realizar. En concreto, la posición y orientación de un sistema de referencia móvil situado en la punta del robot, con respecto a un base fija.

Este sistema es muy necesario para distintos puntos del trabajo, como por ejemplo medir la capacidad de repetibilidad del robot o servir para el control en el capítulo 6.

Para lograr este objetivo se diseña y construye el hardware necesario, y para la tarea de medición se hará uso de *Visión por computador*

Este capítulo tiene objetivo explicar el funcionamiento del sistema que se ha ideado para este proyecto, así como su implementación y las funciones que se han desarrollado para llevarlo a cabo.

### 5.1. Hardware

Comenzaremos el capítulo explicando todos los elementos hardware de los que se compone el sistema. Estos cumplen la función de mantener el robot fijo en una posición y hacer posible la visión por computador

#### 5.1.1. Utilaje para el robot

Hacer fotos del utilaje, etc.

#### 5.1.2. Tabla de calibración

Este elemento (figura 5.1) se trata de una cuadrícula de calibración de tipo ajedrez, que cumplirá distintas funciones a lo largo del desarrollo de este trabajo. La primera de ellas es calibrar las cámaras tal y como se explicará en el apartado correspondiente (5.2.1). La otra es para hallar los coeficientes que determinan la posición de las cámaras con respecto al sistema de referencia fijo, tal y como se explicara en el mismo apartado.

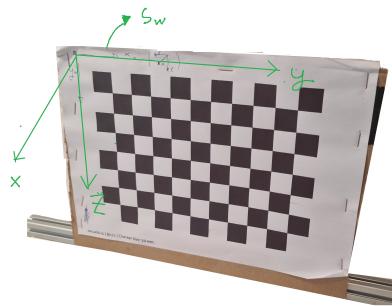


Figura 5.1: Tablero de calibración. *Fuente: elaboración propia*

### 5.1.3. Cámaras

Para la estereovisión se harán uso de dos cámaras. La primera es el modelo Logitech C270 (5.2.a)) a la cual se le denominara  $cam_{yz}$  en el resto del trabajo. La segunda es el modelo Krom 1080p Camera (5.2.b)), denominada  $cam_{xz}$  en el resto del trabajo. En la siguiente tabla se recogen las principales especificaciones de las dos cámaras.

Modelo	Resolución	Tasa de refresco	Campo de visión	alias
Logitech C270	1280 x 720	30 fps	55	$cam_{yz}$
Krom 1080p webcam	1920 x 1080	30 fps	110	$cam_{xz}$

Cuadro 5.1: Cámaras utilizadas



(a) Logitech C270 (yz)

(b) USB Camera (xz)

Figura 5.2: Cámaras utilizadas. *Fuente: a) Logitech, b) Krom*

### 5.1.4. Baliza luminosa

Es el objeto que será reconocido en el espacio para determinar la posición y rotación del sistema móvil con respecto al sistema fijo. La baliza (figura 5.3) se compone de tres esferas, fabricadas mediante impresión 3D en PLA, dentro de las cuales se han embebido tres diodos LED. Gracias a estos es posible variar la luminosidad de las esferas mediante software.

Estas esferas forman un triángulo, definiendo el denominado sistema de referencia móvil. Este elemento se encuentra unido al extremo del robot, permitiendo establecer así la posición y orientación de este.

La longitud de la varilla central, que aleja las esferas luminosas de la base del extremo del robot, permite que estas sean visibles por las cámaras en todas las poses que puede adoptar el robot. Si en caso contrario, las esferas estuvieran directamente unidas al extremo del robot, existirían numerosas poses en las cuales no sería posible determinar la posición, al quedar las esferas ocultas por el robot mismo.



Figura 5.3: Baliza luminosa. *Fuente: elaboración propia*

## 5.2. Visión por computador

A continuación, se explica todo lo referente a la visión por computador. Esta será posible utilizando las dos cámaras mencionadas anteriormente en una configuración de *estereovisión*. Además de esto, se han desarrollado distintas funciones de Matlab que se detallarán en este capítulo, y se ha utilizado la toolbox *Matlab Computer Vision Toolbox*. Para esto se ha partido en parte de la referencia [1].

Antes de explicar detalladamente todos los procesos que se siguen para hallar la posición - orientación en el espacio, se va a explicar brevemente el fundamento en el que se basa.

Lo primero, es tener presentes los distintos sistemas de referencia de los que se tratará en el siguiente capítulo. Parte de esta información ha sido recogida de la fuente

1. Sistema de referencia fijo  $S_w$ , o sistema del mundo real.
2. Sistema de referencia móvil  $S_m$ , situado en el extremo del robot.
3. Sistemas 3D cámaras (uno para cada cámara,  $S_{c1}$  y  $S_{c2}$ ), sistemas de referencia sólidos a las cámaras.
4. Sistema 2D cámaras (uno por cada cámara,  $(u_1, v_1)$  y  $(u_2, v_2)$ ), sistemas de referencia 2D solidarios a las cámaras (píxeles de la imagen.)

Para que quede definido el sistema móvil  $S_m$  basta con conocer la posición de tres puntos de este sistema, que serán las 3 marcas luminosas de la baliza mencionada en el apartado 5.1. Por esto el problema de la visión se reduce en hallar la posición de estas tres marcas respecto al sistema fijo  $S_w$ .

Para ello se hará uso de dos cámaras en configuración de *estereovisión*. Se tomarán dos imágenes, de las dos cámaras. Posteriormente, se reconocerá cada una de las marcas en las dos imágenes, lo cual nos dará sus coordenadas  $(u_1, v_1)$  y  $(u_2, v_2)$ . Para conseguir esto se hace uso del **color thresholding** (5.2.2.1) y el reconocimiento de patrones (5.2.2.2).

Por otra parte, la relación entre el sistema  $S_w$  y los sistemas  $S_{c1}$  y  $S_{c2}$ , relativos a las cámaras, viene dada por los denominados **parámetros extrínsecos**. Además, se puede realizar la transformación entre los sistemas  $S_{c1}$  y  $S_{c2}$  y los  $(u_1, v_1)$  y  $(u_2, v_2)$  mediante los denominados **parámetros intrínsecos**. La obtención de estos parámetros se detalla en el apartado 5.2.1.

Por ello, una vez conocidos estos parámetros, y conocida la posición de un punto en los sistemas  $(u_1, v_1)$  y  $(u_2, v_2)$  es posible reconstruir la posición del punto en el sistema del mundo real  $S_w$ , utilizando la triangulación estéreo, que se explica en el apartado 5.2.3.

### 5.2.1. Calibración de cámaras

Lo primero que hay que tener en cuenta a la hora de implementar técnicas de visión por computador es que hay que disponer de una llamada *cámara calibrada*. Con esto nos referimos a hallar los denominados **parámetros intrínsecos** y **extrínsecos** de la cámara.

Los parámetros intrínsecos modelan las características internas de la cámara, como puede ser la distancia focal, y se utilizan para eliminar la distorsión de la imagen provocada por la geometría de la lente. Nos sirven para pasar de los sistemas solidarios 3D de las cámaras  $S_{c1}$  y  $S_{c2}$  a los sistemas 2D  $(u_1, v_1)$  y  $(u_2, v_2)$ ). La relación entre ambos sistemas es la siguiente:

$$Z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = M_i \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} \Rightarrow Z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} \quad (5.1)$$

Siendo  $M_1$  y  $M_2$  las matrices de parámetros intrínsecos de las cámaras.

Por otra parte, tenemos los **parámetros extrínsecos**, que se componen de una matriz de rotación y un vector de traslación, que definen la posición de la cámara respecto a un sistema fijo. Dicho de otra manera, transforman coordenadas del sistema del mundo real  $S_w$  al sistema de una cámara  $S_{ci}$

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = M_i \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R_{3x3} & t_{3x1} \\ 0_{1x3} & 1_{1x1} \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad (5.2)$$

Combinando las expresiones 5.2.1 y 5.2.2 se llega a la siguiente expresión, que relaciona las coordenadas del mundo real de un punto en el sistema  $S_w$  al sistema en coordenadas de la imagen de una de las cámaras  $(u_1, v_1)$  y  $(u_2, v_2)$

$$Z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = P_i \begin{bmatrix} R_{3x3} & t_{3x1} \\ 0_{1x3} & 1_{1x1} \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad (5.3)$$

Ahora que se ha expresado la necesidad de hallar los parámetros intrínsecos y extrínsecos de las cámaras, se va a explicar el procedimiento y las funciones que se han desarrollado para esto.

Primero, para hallar los parámetros intrínsecos se ha hecho uso de la herramienta *Camera Calibrator* perteneciente al paquete *Computer Vision Toolbox* de Matlab. Mediante esta herramienta y la tabla con el tablero explicada en el apartado 5.1 es posible estimar los parámetros extrínsecos e intrínsecos de las cámaras. Este proceso habrá que realizarlo dos veces, para hallar los parámetros de ambas cámaras.

El proceso que se realiza en dicha herramienta es, primero, tomar una serie de imágenes en las que sea visible la cuadrícula en distintas posiciones. Posteriormente, la herramienta hará una primera estimación de los datos y mostrará la información estadística necesaria para descartar las imágenes no adecuadas.

Una vez estimados los parámetros, se han almacenado las dos variables generadas, *cameraParams* y *cameraParams\_xz* en (de *cam<sub>yz</sub>* y *cam<sub>xz</sub>* respectivamente), en un fichero denominado **par.m**.

Aunque la herramienta también estime los parámetros extrínsecos, para hallar estos se ha desarrollado una función propia denominada **findCameraPose.m**, ya que la herramienta **Camera Calibrator** devuelve los parámetros extrínsecos de la última imagen utilizada en la calibración, y en el trabajo real, el sistema fijo puede variar a lo largo del tiempo. En resumen, interesa tener una función para encontrar rápidamente los parámetros extrínsecos para implementar una función de calibración.

### La función **findCameraPose.m**

```
1 function [rotation,translation] = findCameraPose(cam, cameraPar)
```

Esta función recibe como parámetros un objeto cámara de Matlab, y un struct *cameraParams*, como los devueltos por la herramienta *Camera Calibrator* y devuelve los parámetros extrínsecos de la cámara en ese momento respecto al sistema fijo *S<sub>w</sub>* definido por la cuadrícula. Tomará una imagen con la cámara pasada como parámetro, y detectará la cuadrícula en ella. Posteriormente, usando *cameraParams.Intrinsics*, distorsionará la imagen. Finalmente, usando la función de Matlab *extrinsics* encontrará la matriz de rotación y el vector de posición, de los cuales se componen los parámetros extrínsecos. Además, mostrará en una nueva figura la posición de la cámara respecto al sistema fijo de la cuadrícula

El tamaño de la cuadrícula se puede establecer internamente en la función.

```
9 % Detection del patron
10 [imagePoints,boardSize] = detectCheckerboardPoints(im);
11 squareSize = 20; % Tamano del tablero en mm
12 worldPoints = generateCheckerboardPoints(boardSize, squareSize);
```

### 5.2.2. Reconocimiento del objeto en la imagen

Para poder posicionar un punto del espacio respecto al sistema de referencia fijo  $S_w$  previamente es necesario reconocer dicho punto en dos imágenes tomadas por  $cam_{yz}$  y  $cam_{xz}$ . Para ello se utilizarán las regiones MSER, tal y como se explica en el punto 5.2.2.2. Este método tiene como objetivo reconocer y posicionar una esfera en los sistemas  $(u_1, v_1)$  y  $(u_2, v_2)$ .

El problema reside en que en el caso de este trabajo hay posicionar varias esferas al mismo tiempo: las tres marcas luminosas azul, verde y roja. Para ello, previo a utilizar el procedimiento del punto 5.2.2.2 es necesario filtrar las imágenes por colores. Esto se hará mediante el proceso de color Thresholding explicado a continuación en el punto 5.2.2.1

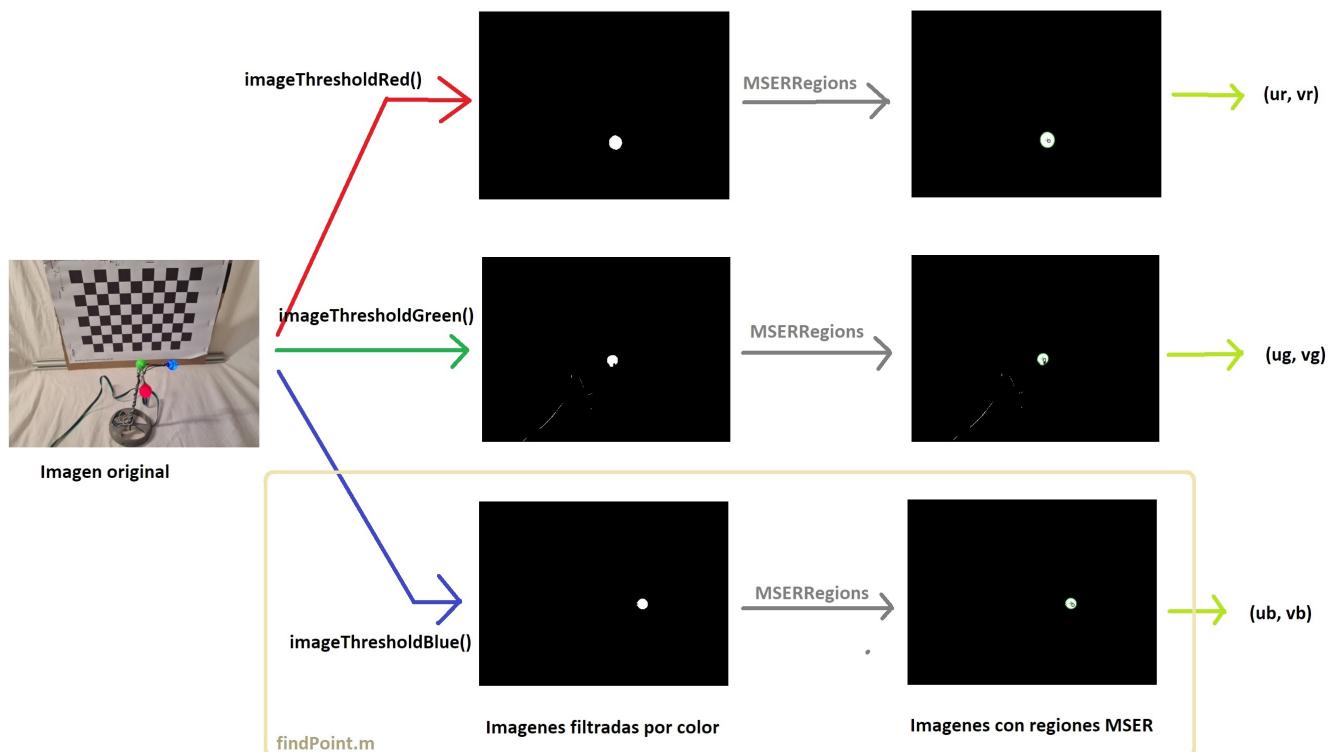


Figura 5.4: Proceso del reconocimiento de los puntos en la imagen: FindPoint. Fuente: elaboración propia

#### 5.2.2.1 Color Threshold

Para filtrar las imágenes por colores se ha utilizado la herramienta *Color Thresholder*, del paquete *Computer Vision Toolbox*.

Esta herramienta permite, dada una imagen, mostrar todos sus píxeles en los espacios de color RGB, HSV, YCbCr y Lab (figura 5.5).

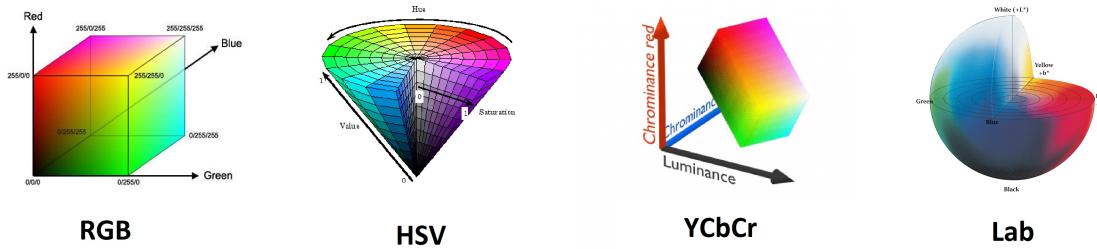


Figura 5.5: Distintos espacios de color. *Fuente: elaboración propia*

El objetivo es, primero, seleccionar el espacio de color en el que más se pueda diferenciar el objetivo a aislar. Dependiendo de la imagen puede convenir más trabajar con un espacio de color u otro (los que más se han utilizado a lo largo de este trabajo son el RGB y HSV). Una vez seleccionado el espacio de color más adecuado, se aplica un filtro que acota una región de dicho espacio, eliminando el resto.

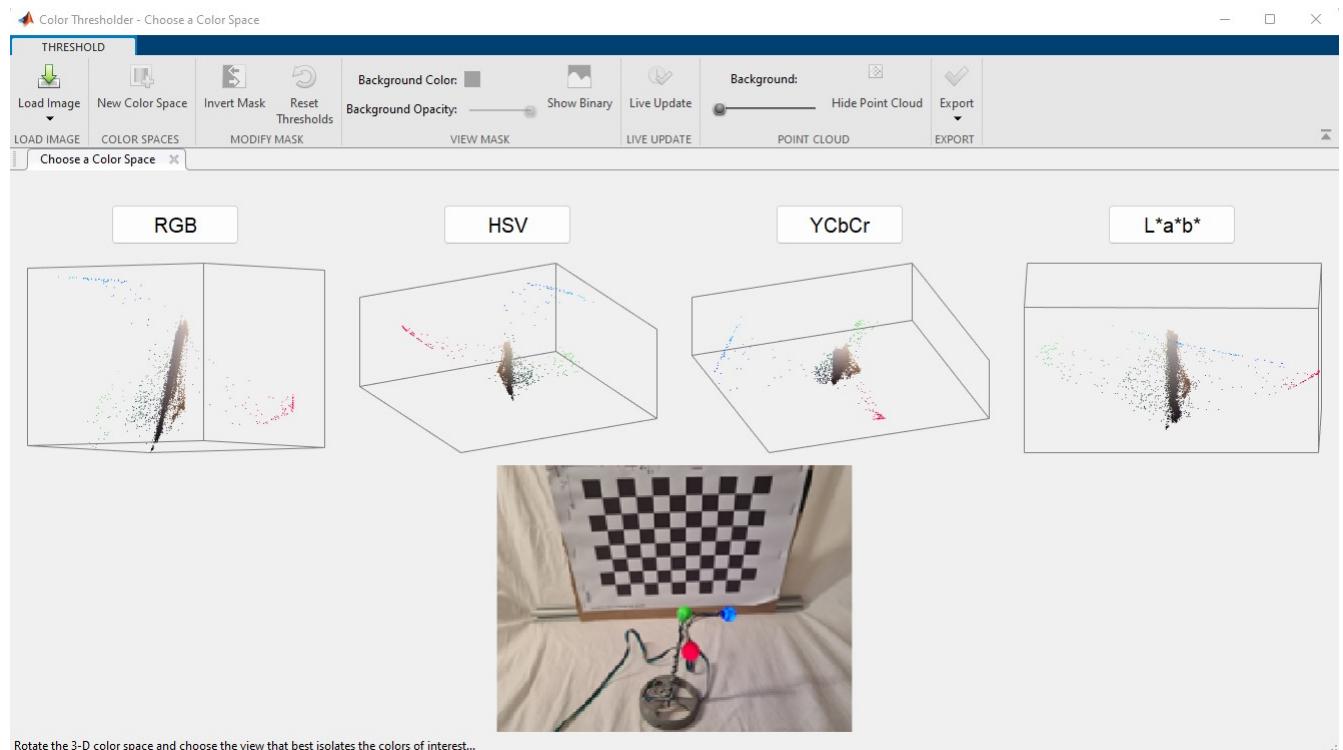


Figura 5.6: Matlab ColorThresholder. *Fuente: Matlab*

La herramienta *colorThresholder* permite generar una función para aplicar el filtro a muchas imágenes de forma sistemática. De esta manera, en el trabajo se han calibrado las cámaras para reconocer las marcas luminosas azul, rojo y azul, en las condiciones de luminosidad del espacio de trabajo. Si las condiciones lumínicas varían, es necesario volver a calibrar dichas funciones. Por tanto, el procedimiento que se ha seguido:

1. Montar las cámaras en el entorno de trabajo con las condiciones de luminosidad que van a estar presentes.

2. Tomar una imagen con cada una de las cámaras y abrirla en la herramienta *color Thresholder*
3. Seleccionar el espacio de color en el que se diferencien más las marcas luminosas.
4. Calibrar un filtro para cada marca luminosa y cada cámara
5. Generar las funciones.

Aplicando los criterios anteriores se han generado las funciones de Matlab necesarias. Consisten en las siguientes seis:

- imageThresholdBlue\_yz.m
- imageThresholdBlue\_xz.m
- imageThresholdOrange\_yz.m
- imageThresholdOrange\_xz.m
- imageThresholdGreen\_yz.m
- imageThresholdGreen\_xz.m

Cada una de estas funciones tiene una interfaz similar a la siguiente:

```
9 function [BW,maskedRGBImage] = imageThresholdOrange_yz(RGB)
```

Se acepta como parámetro la imagen a color, y se devuelve la misma imagen en blanco y negro, siendo blanca la zona de interés y negra el resto de la imagen que no ha pasado el filtro

### 5.2.2.2 MSER regions

Una vez pasada la imagen por el filtro de color, es necesario reconocer la marca en esta, que consistirá aproximadamente en un círculo blanco. Para lograr esto se empleará un método basado en las *Maximally stable extremal regions* o regiones MSER. Una región MSER de una imagen es una región de esta caracterizada por una intensidad aproximadamente constante rodeada de un fondo respecto al cual existe un alto contraste. Estas regiones se caracterizan por un elipsoide (figura 5.7).

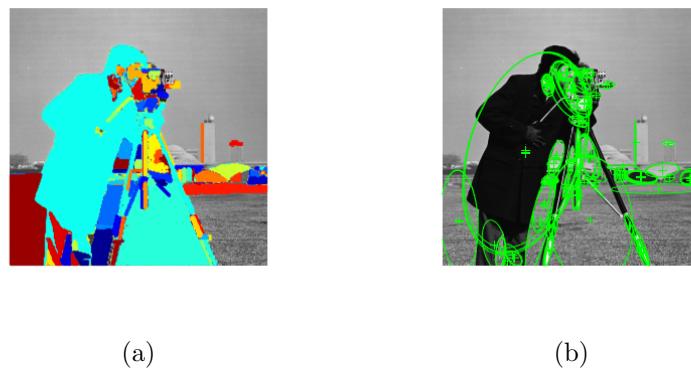


Figura 5.7: Regiones MSER. Fuente: Matlab

De esta manera, en nuestra imagen a analizar, para reconocer el objeto deseado se detectarán las regiones MSER utilizando la función *detectMSERfeatures* de *Matlab*. Posteriormente, se filtran dichas regiones por excentricidad, por si en caso de haber más de una, solo se contabilice la de la marca del robot. Esta, al tratarse de una esfera, siempre proyectará un círculo sobre la imagen, dando lugar a una región MSER con excentricidad cercana a uno.

Una vez realizado esto, se determinará la posición de la marca en la imagen como la posición del centroide del elipsoide de su región MSER correspondiente.

### 5.2.2.3 Integración: función *findpoint.m*

Los métodos explicados en los puntos 5.2.2.1 y 5.2.2.2 se han integrado en una función de *Matlab* para automatizar de forma sencilla el proceso de reconocer las marcas luminosas en las imágenes. Esta es la función *findPoint.m*

```
1 function [BW,maskedRGBImage] = imageThresholdOrange_yz(RGB)
```

Esta función devuelve la posición del centro de la marca esférica de la imagen y acepta los siguientes parámetros:

- *rgb*: imagen a analizar
- *color*: color de la marca esférica a buscar ("r", "g", "b")
- *camera*: con qué cámara se ha tomado la imagen ("yz" o "xz")
- *debug*: en caso de ser "y", se muestra en una nueva figura el proceso de encontrar el punto en la imagen

Primero se filtra por color la imagen utilizando la función correspondiente:

```
13 switch color
14 case 'b'
15     if camera == "yz"
16         [BW, ~] = imageThresholdBlue_yz(rgb);
17     else
18         [BW, ~] = imageThresholdBlue_xz(rgb);
19     end
20 case 'p'
21     [BW, ~] = imageThresholdPurple(rgb);
22 case 'g'
23     if camera == "yz"
24         [BW, ~] = imageThresholdGreen_yz(rgb);
25     else
26         [BW, ~] = imageThresholdGreen_xz(rgb);
27     end
28 case 'o'
29     if camera == "yz"
30         [BW, ~] = imageThresholdOrange_yz(rgb);
31     else
32         [BW, ~] = imageThresholdOrange_xz(rgb);
33     end
34 case 'n'
35     [BW, ~] = imageThresholdBlack(rgb);
36 end
```

Posteriormente, se detectan las regiones MSER de la imagen filtrada y se filtran por excentricidad para hallar la posición del punto deseado:

```

38 % Detectamos las regiones
39 [misRegiones,mserCC] = detectMSERFeatures(BW);
40
41 % Filtramos por excentricidad
42 stats = regionprops('table',mserCC,'Eccentricity');
43 eccentricityIdx = stats.Eccentricity < 0.99;
44 misRegiones = misRegiones(eccentricityIdx);
45
46 % Comprobamos numero de regiones detectadas
47 if misRegiones.Count > 0
48     x = misRegiones.Location(1,1);
49     y = misRegiones.Location(1,2);
50 else
51     x = -1;
52     y = -1;
53 end

```

### 5.2.3. Posicionamiento en el espacio: triangulación estéreo

Una vez halladas las coordenadas del punto deseado en los sistemas de referencia 2D de las cámaras  $(u_1, v_1)$  y  $(u_2, v_2)$ , y conociendo las matrices de parámetros intrínsecos y extrínsecos, es posible hallar la posición de dicho punto respecto al sistema de referencia fijo  $S_w$ . Esto se hará mediante un método denominado *triangulación estéreo*, tal y como se explica en los trabajos realizados en [29] y [10]

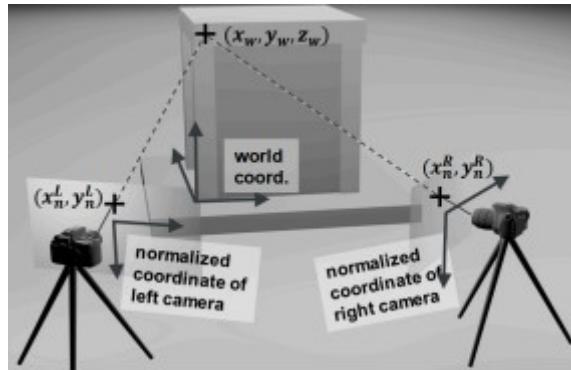


Figura 5.8: Triangulación estéreo. Fuente: [29]

A continuación, se explicará en que consiste el proceso. Primero, se asume que la distorsión de las imágenes ha sido eliminada previamente mediante los parámetros intrínsecos. De esta manera se puede escribir la información de la ecuación 6.3 de la siguiente manera (refiriéndonos con los superíndices a la imagen de  $cam_{yz}$  o  $cam_{xz}$ ):

$$\begin{bmatrix} u_1 Z_c^{yz} \\ v_1 Z_c^{yz} \\ Z_c^{yz} \\ 1 \end{bmatrix} = \begin{bmatrix} R_{3x3}^{yz} & t_{3x1}^{yz} \\ 0_{1x3} & 1_{1x1} \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad (5.4)$$

$$\begin{bmatrix} u_2 Z_c^{xz} \\ v_2 Z_c^{xz} \\ Z_c^{xz} \\ 1 \end{bmatrix} = \begin{bmatrix} R_{3x3}^{yz} & t_{3x1}^{xz} \\ 0_{1x3} & 1_{1x1} \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad (5.5)$$

Al tratarse las coordenadas  $(X_w, Y_w, Z_w)$  de las ecuaciones (5.4) y (5.5), de las mismas en ambas cámaras, pueden reordenarse y simplificarse las ecuaciones de la siguiente manera:

$$A \begin{bmatrix} Z_c^{yz} \\ Z_c^{xz} \end{bmatrix} = B \quad (5.6)$$

Siendo A y B:

$$A = (R_{3x3}^{yz})^T \begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix} - (R_{3x3}^{xz})^T \begin{bmatrix} u_2 \\ v_2 \\ 1 \end{bmatrix} \quad (5.7)$$

$$B = (R_{3x3}^{yz})^T t_{3x1}^{yz} - (R_{3x3}^{xz})^T t_{3x1}^{xz} \quad (5.8)$$

Ambas matrices, A y B, se tratan de matrices ortonormales, por lo que sus inversas son iguales a sus transpuestas. Estas matrices se pueden calcular mediante los parámetros extrínsecos conocidos. Una vez calculadas, podemos obtener  $Z_c^{yz}$  y  $Z_c^{xz}$  mediante mínimos cuadrados de la siguiente manera:

$$\begin{bmatrix} Z_c^{yz} \\ Z_c^{xz} \end{bmatrix} = (A^T A)^{-1} A^T B \quad (5.9)$$

Una vez hallados los parámetros  $Z_c^{yz}$  y  $Z_c^{xz}$ , se pueden obtener las coordenadas  $(X_w, Y_w, Z_w)$  mediante las ecuaciones (5.4 y 5.5).

### La función getCoordinates.m

Una vez explicado el fundamento, procederemos a explicar la manera en la que esto se ha implementado en Matlab. Para ello se ha creado la función *getCoordinates.m*, que devolverá la posición del punto deseado respecto al sistema fijo.

```
1 function [pointReal] = getCoordinates(yz, xz, matyz, matxz, dx, dy,
dz, zoffset)
```

Esta función acepta los siguientes parámetros:

- $yz$ : coordenadas  $(u_1, v_1)$  del punto en la imagen de  $cam_{yz}$
- $yz$ : coordenadas  $(u_2, v_2)$  del punto en la imagen de  $cam_{xz}$
- $matyz$ : Matriz con las coordenadas extrínsecas de  $cam_{yz}$  halladas mediante la función *findCameraPose.m*

- $\text{matzz}$ : Matriz con las coordenadas extrínsecas de  $\text{cam}_{yz}$  halladas mediante la función *findCameraPose.m*
- $dx$ : traslación respecto al eje x (cero antes del homing)
- $dy$ : traslación respecto al eje y (cero antes del homing)
- $dz$ : traslación respecto al eje z (cero antes del homing)
- $zoffset$ : traslación respecto al eje z (Para posicionar la altura del sistema fijo)

Al comienzo del desarrollo se optó por realizar el proceso de triangulación programado desde cero, pero finalmente se realiza utilizando la función *triangulate.m* de Matlab. Esta función acepta como parámetros las coordenadas de los puntos en las imágenes (*yz* e *xz*) y las matrices de parámetros extrínsecos de las cámaras (*matyz* y *matzz*). Devuelve como resultado el punto de triangulación.

Este punto, antes de ser utilizado, es rotado para que su posición esté correctamente definida respecto al sistema  $S_w$ . Además, se aplican unos coeficientes ( $dx$ ,  $dy$ ,  $dz$  y  $zoffset$ ) cuya función se explicará a continuación.

### 5.3. Implementación en Matlab

Hasta este punto, en este capítulo se ha explicado el desarrollo teórico y práctico de las siguientes funciones de *Matlab*:

- **findCameraPose.m**: Función para calcular la posición de una cámara respecto al sistema fijo
- **findPoint.m**: Función que devuelve la posición en la imagen de una de las marcas luminosas deseadas.
- **getCoordinates.m**: Función que, dadas las posiciones de un punto en las imágenes de dos cámaras, y dados los parámetros extrínsecos de estos, devuelve la posición de dicho punto respecto al sistema fijo.

En resumen, el conjunto de estas funciones permite la detección de un punto del espacio y su posicionamiento respecto a un sistema fijo. Sin embargo, para lograr el objetivo principal de posicionar en traslación y rotación un sistema de referencia móvil  $S_m$  con respecto al sistema fijo,  $S_w$  es necesario tener en cuenta algunos detalles adicionales.

El trabajo que se ha explicado hasta este punto permite obtener las coordenadas de los tres puntos que definen el sistema móvil con respecto al sistema fijo definido por el tablero de calibración, tal y como se muestra en la imagen a continuación:

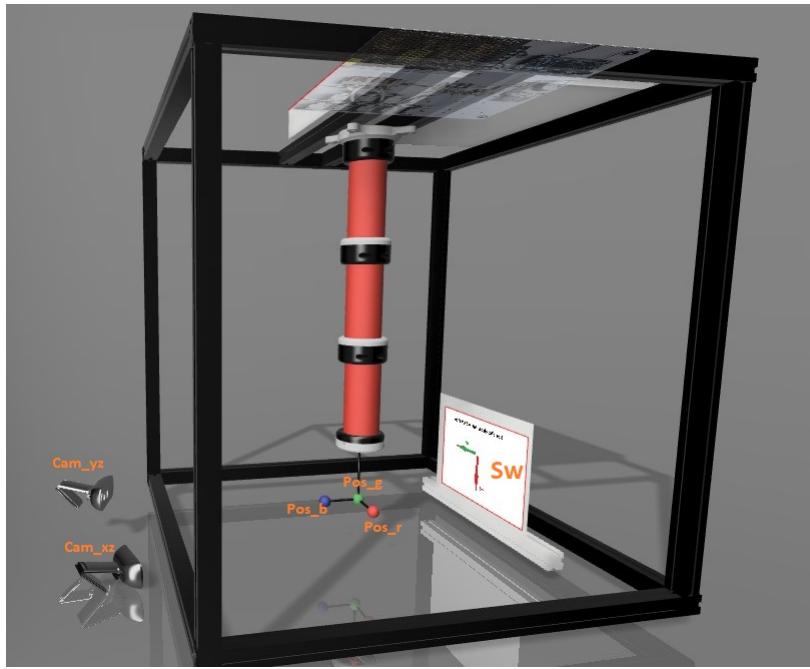


Figura 5.9: Posicionamiento inicial. *Fuente: elaboración propia*

Por tanto, serán necesarias dos operaciones adicionales que se integrarán en la interfaz que se explicará en el capítulo 6.

La primera de ellas consiste en realizar una transformación al sistema  $S_w$  para que en vez de estar situado en el lugar del espacio en el que se encontraba el tablero de calibración en el momento en el que se llama a la función *findCameraPose.m*, se encuentre en el extremo del robot, con el objetivo de medir el cambio de posición respecto a este punto (figura 5.10). En este momento entran en juego los parámetros  $dx$ ,  $dy$  y  $dz$ , que se introducen como parámetros en la función *getCoordinates.m*. Estos definen una traslación en el sistema  $S_w$  para situarlo en el lugar adecuado. Para hallar dichos parámetros se integrará una funcionalidad de homing en la interfaz. Esta función lo que hará es tomar la posición de la marca verde en el momento en el que se llame la función y las coordenadas obtenidas corresponderán a dichos parámetros. Esto se justifica teniendo en cuenta que la marca verde coincide con la posición del origen del sistema de referencia fijo estando el robot en reposo, pudiendo aplicarse de forma adicional una traslación respecto al eje z que se define mediante el parámetro *zoffset*.

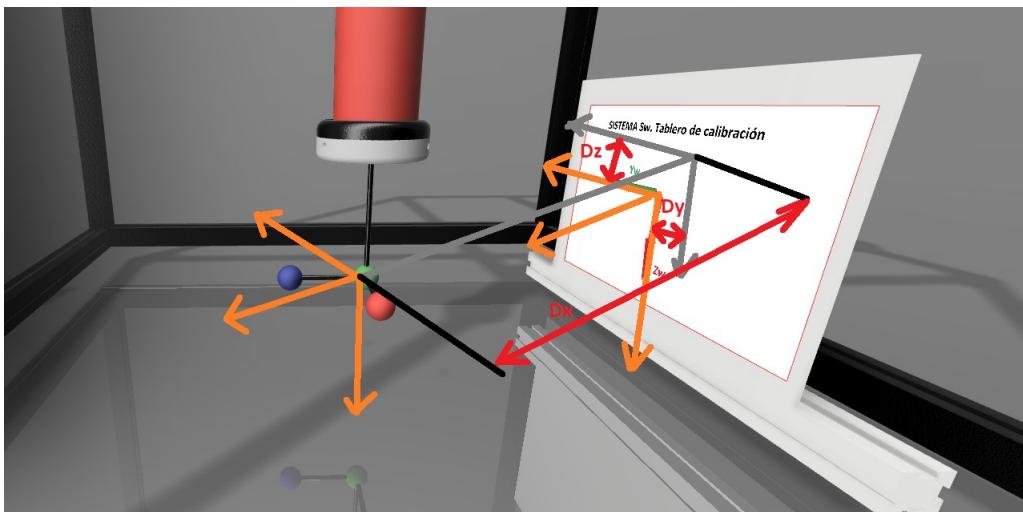


Figura 5.10: Transformación adicional 1. *Fuente: elaboración propia*

La segunda operación que se tiene que realizar está relacionada con, una vez obtenidas las posiciones de los puntos r, g y b (marcas roja, verde y azul respectivamente), obtener las rotaciones del sistema móvil con respecto al sistema fijo. Para ello se desarrolla la función **getRotations.m**, que recibe como parámetros las posiciones de dichos puntos y devuelve las rotaciones deseadas en forma de matriz de rotación y ángulos de Euler.

```
1 function [rotm,euler] = getRotations(r,g,b)
```

Para ello, en la función, una vez comprobado que se trata de puntos válidos, se proceden a calcular los vectores directores del sistema móvil:

```
1 if r(1) == -1 || r(2) == -1 || r(3) == -1 || g(1) == -1 || g(2) == -1 || g
2   (3) == -1 || b(1) == -1 || b(2) == -1 || b(3) == -1
3   euler = [-1 -1 -1];
4   rotm = [-1 -1 -1 ; -1 -1 -1 ; -1 -1 -1];
5 else
6   gv = [g(1), g(2), g(3)];
7   rv = [r(1), r(2), r(3)];
8   bv = [b(1), b(2), b(3)];
9
10 % Calculamos vectores directores del sistema móvil
11 xr = rv - gv ;
12 xr = xr / norm(xr);
13 yr = bv - gv;
14 yr = yr / norm(yr);
```

Estos vectores coinciden con los mostrados en la siguiente figura:

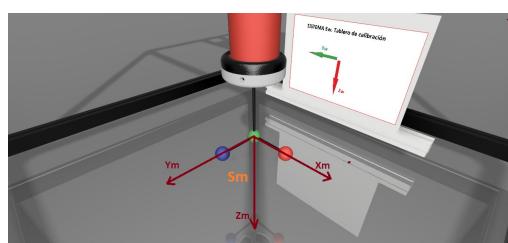


Figura 5.11: Transformación adicional 2. *Fuente: elaboración propia*

Una vez se dispone de dichos vectores directores ya se puede calcular la matriz de rotación y los ángulos de Euler:

```
1 % Calculamos matriz rotacion
2 v = cross(xr,yr);
3 ssc = [0 -v(3) v(2); v(3) 0 -v(1); -v(2) v(1) 0];
4 rotm = eye(3) + ssc + ssc^2*(1-dot(xr,yr))/(norm(v))^2;
5
6 % Pasamos a angulos de Euler
7 euler = rotm2eul(rotm);
```

Toda esta metodología es la que se integrará en la interfaz de usuario tratada en el capítulo 6

## 6. Diseño experimental y recogida de datos

En este capítulo se explicará el procedimiento experimental previo necesario para el control del robot.

### 6.1. Introducción

Debido a la complejidad del robot, se decide descartar las metodologías de control más tradicionales en el alcance de este trabajo. En vez de ellos se barajan distintas estrategias de control basadas en el manejo de parámetros que definen el comportamiento del robot. Todas las opciones de control barajadas tienen en común la necesidad de disponer de una gran cantidad de datos empíricos, lo cual lleva a la necesidad de desarrollar un diseño experimental para sistematizar la recogida de estos datos.

Estos grandes conjuntos de datos se almacenarán en distintos ficheros denominados *datasets*, que disponen de distintos campos para organizar la información que se detallarán más adelante en este capítulo. Debido a que la captura de esta información se realiza en distintas fases y que los datasets tienen que representar el comportamiento del robot de forma objetiva, la reaplicabilidad del experimento toma especial importancia.

Los datos más relevantes que se capturarán serán la posición del extremo del robot y el conjunto de presiones que logran esta configuración. Debido a la redundancia de grados de libertad y la propia geometría del robot, existirán distintos conjuntos de presiones que lleven a una misma posición del extremo.

En este capítulo, convergen los distintos elementos desarrollados hasta este punto. Gracias al desarrollo del sistema de visión por computador es posible automatizar esta recogida de datos y obtener datasets con una gran cantidad de muestras. Esta automatización se integrará en la interfaz de usuario que se explicará en el capítulo 6, para permitir la recogida de datasets adicionales en unas condiciones experimentales similares.

El proceso de recogida de datos consistirá, de forma resumida, en primero establecer un número de muestras. Posteriormente, para cada muestra se enviará desde Matlab al banco de actuación una combinación aleatoria de nueve presiones, una para cada grado de libertad del robot. Una vez hinchadas las vejigas de este a las presiones enviadas, se tomarán imágenes con las dos cámaras del sistema de visión, y se establecerá la posición y orientación del extremo del robot. Este procedimiento se repetirá el número de veces indicado y al finalizar el proceso se almacenarán los datos deseados en un fichero de tipo struct

### 6.2. Datos a recoger y condiciones experimentales

En este apartado se recoge el desarrollo, conclusión y estructura de los datos recogidos, así como las condiciones experimentales en las que se han tomado los datos

#### 6.2.1. Datos a recojer

El primer punto que resolver se trata de decidir cuáles serán los datos que se recojan, para establecer una estructura de datos fija y poder comparar los datos tomados a lo largo del

tiempo.

Los resultados más importantes será la combinación de presiones en cada iteración y la posición del sistema de referencia móvil en cada momento.

Respecto a la combinación de presiones, en este caso se medirán como *porcentaje de actuación*, siendo el cien por ciento lo máximo que puede hincharse una vejiga y cero por ciento totalmente desinchada. Para ello es necesario establecer previamente las presiones máximas relativas enviadas. Este concepto es el explicado en el capítulo 4: Banco de actuación. Además de la combinación de presiones, en los metadatos del dataset también debe figurar la presión máxima relativa establecida (la presión equivalente al 100 por ciento de actuación), para permitir la comparación de distintos databaseis. En caso de que dos datasets se hayan tomado con distintas presiones máximas relativas, no serían comparables, puesto que un cien por ciento de actuación de las vejigas equivaldría a curvaturas distintas.

Una vez realizadas las primeras pruebas se pone de manifiesto la necesidad de realizar la recogida de datos en distintas fases. En cada fase se establece el porcentaje de actuación máximo y mínimo que se enviarán al robot, por lo que este dato también quedará almacenado en el dataset.

Con respecto a la posición y orientación del sistema de referencia móvil con respecto al fijo, esto se hará mediante el sistema de visión por computador explicado en el capítulo 5. Los datos que se almacenarán serán, en primer lugar, la traslación del sistema móvil con respecto al fijo, definida por la posición de la marca verde, que coincidirá con el origen del sistema móvil. Además, se almacenará la rotación en forma de ángulos de Euler.

Además, se almacenarán otros datos que, aunque no sirvan directamente para el control, si cumplen la función de metadatos que permitan comparar distintos datasets entre sí. Estos son algunos como la presión de la línea neumática, la temperatura ambiente, el número de muestras, etc. Finalmente, también se guardan datos que puedan permitir en un futuro depurar alguna inconsistencia en los datos, como pueden ser las posiciones de las distintas marcas luminosas en las imágenes tomadas por las dos cámaras en cada iteración.

Todos estos datos que se organizarán en estructuras con los siguientes campos:

- **pmax**: Vector con tres elementos que definen los porcentajes máximos de actuación enviados en este dataset

$$[P_{maxS1} \ P_{maxS2} \ P_{maxS3}] \quad (6.1)$$

- **pmin** Vector con tres elementos que definen los porcentajes mínimos de actuación enviados en este dataset

$$[P_{minS1} \ P_{minS2} \ P_{minS3}] \quad (6.2)$$

- **temp\_amb**: Temperatura ambiental en grados centígrados a la cual se han recogido las muestras.
- **presion\_presostato** Presión de la línea neumática, determinada por la presión máxima que se indica en el presostato del banco de actuación en [Mpa]
- **maxRpressure** Máxima presión relativa que se ha configurado en el banco neumático previo a comenzar la recogida de datos

- **nummuestras** Número de muestras que contiene el dataset
- **img\_points** Puntos encontrados en las distintas imágenes, en forma de estructuras
- **points** Distintos puntos del espacio correspondientes a las distintas marcas luminosas identificadas en cada iteración.
- **inputs** Presiones enviadas. Matriz con *nummuestras* filas y nueve columnas, en las que cada fila es un vector que contiene las nueve presiones enviadas en cada iteración.

$$[V_1 \ V_2 \ V_3 \ V_4 \ V_5 \ V_6 \ V_7 \ V_8 \ V_9] \quad (6.3)$$

#### ▪ **outputs**

Posiciones alcanzadas. Matriz con *nummuestras* filas y 6 columnas. Cada fila contiene seis elementos, de los cuales los primeros tres corresponden a la posición del origen del sistema móvil y las siguientes tres a los tres ángulos de Euler

$$[X_m \ Y_m \ Z_m \ \alpha \ \beta \ \gamma] \quad (6.4)$$

#### ▪ **index**

Índice de cada muestra en la tabla. Se utiliza en distintas funciones de la *interfaz de usuario*

#### ▪ **elapsed\_time**

Tiempo en segundos que ha tardado en recogerse el dataset.

En la imagen a continuación se puede observar un ejemplo de dataset recogido:

1x1 struct with 12 fields	
Field	Value
pmax	[50,50,50]
pmin	[20,20,20]
temp_amb	20
presion_presostato	0.1000
maxRpressure	1200
nummuestras	20
inputs	21x9 double
img_points	1x1 struct
points	1x1 struct
outputs	21x6 single
index	1x21 double
elapsed_time	122.6156

Figura 6.1: Ejemplo dataset. Fuente: elaboración propia

### 6.2.2. Condiciones experimentales

A parte de definir de forma objetiva los datos que se van a recoger, es necesario definir bien las condiciones experimentales en las que se van a recoger los datos. Para ello, lo primero que se realiza es decidir una nomenclatura para referirse a los distintos segmentos y grupos de vejigas

del robot. Teniendo esto en cuenta se escoge la siguiente nomenclatura para referenciar los distintos grados de libertad:

$$S_{nm} \quad (6.5)$$

Donde  $n$  se refiere el segmento al que pertenece el grado de libertad en cuestión ( $n = 1$  para el segmento superior,  $n = 2$  para el intermedio, y  $n = 3$  para el inferior). Por otra parte,  $m$  se refiere a uno de los tres grupos de vejigas del segmento  $n$ . Esto se puede observar de forma gráfica en la imagen a continuación:

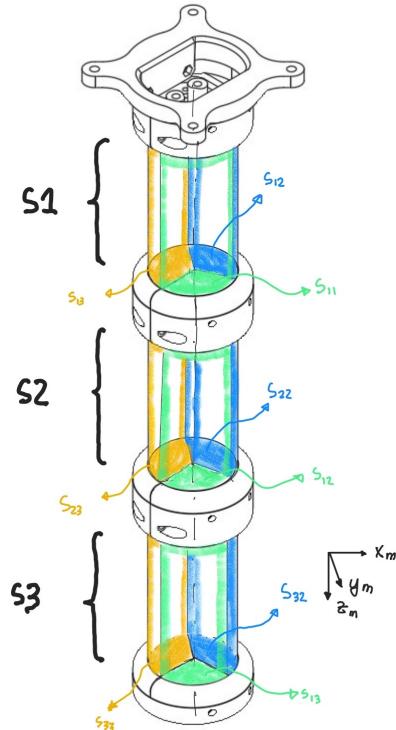


Figura 6.2: Nomenclatura, grados de libertad. *Fuente: elaboración propia*

Una vez establecido un sistema para denominar los distintos grados de libertad del robot, es necesario establecer de forma inequívoca la posición de montaje de este en el banco de pruebas. Esto se debe a que la posición del sistema de referencia móvil siempre debe ser la misma con respecto a la posición de las distintas vejigas del robot, para mantener la coherencia en los dataset. El montaje se realiza de manera que el eje y del sistema móvil esté alineado con las vejigas  $S_{1n}$ , tal y como se muestra en la figura 6.3

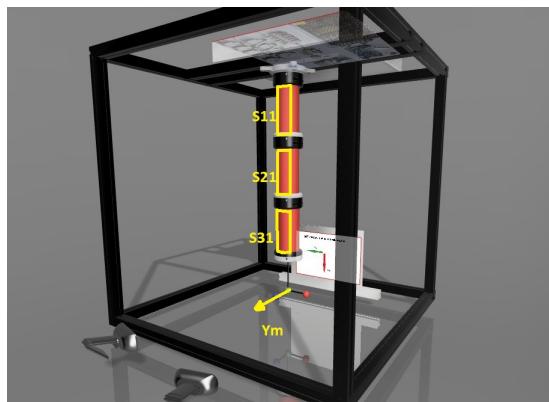


Figura 6.3: Montaje final. *Fuente: elaboración propia*

Además del montaje físico del robot, se intenta mantener una temperatura ambiental de aproximadamente 21 grados centígrados en todas las sesiones de toma de datos, para procurar que la silicona de la cual está fabricado el robot mantenga unas propiedades similares.

Por otra parte, también se establecen unas condiciones lumínicas concretas, para el correcto funcionamiento del sistema de visión. Un defecto de este sistema es que al calibrarse los filtros de color (ver apartado 5.2.2.1) para unas condiciones de luz determinadas, el sistema es muy susceptible al cambio de condiciones ambientales que puedan producirse. Debido a esto, se determina que la recogida de datos se realizará en ausencia de luz. Por ello se cubre todo el montaje con una lona opaca, de tal modo que las cámaras únicamente captén la luz proveniente de las balizas luminosas del robot, eliminando así cualquier incertidumbre relacionada con condiciones lumínicas anómalas. Se calibran las cámaras para funcionar en esta situación.

### 6.3. Procedimiento

A continuación, se explica el procedimiento a través del cual se recogen los datos que formarán los *datasets*. Todo este proceso se ha automatizado en Matlab, y se ha integrado en la interfaz de usuario que se tratará en el capítulo 6.

El procedimiento general consistirá en una serie de operaciones que se realizarán para cada muestra tomada. Para cada muestra que se desea tomar, primero se deshinchan todas las vejigas del robot para que este se encuentre en la posición de reposo. Posteriormente, se generará una combinación aleatoria de presiones que se enviará al robot. Estas presiones se encuentran entre el rango mínimo y máximo establecido previamente. A continuación, se envían dichas presiones al robot.

Esto causa que el robot adopte una pose aleatoria. De esta manera, cada combinación de presiones lleva ligada una pose concreta del robot. Una vez el robot ha terminado de moverse, lo cual se determina esperando el tiempo de inflado de la vejiga que más se tiene que inflar, se toma una imagen con cada una de las cámaras.

De estas imágenes se extrae la posición y orientación del extremo tal y como se explica en el capítulo 5.2, y se almacena en un fichero.

Un aspecto que destacar es que en este trabajo no se ha contemplado la posibilidad de inflar tres vejigas del robot al mismo tiempo. Se han realizado algunas pruebas, y al darse este caso el robot simplemente se *estira* una longitud que no se ha considerado lo suficientemente grande

como para valorar el caso de inflar tres vejigas de un mismo segmento al mismo tiempo. Debido a esto, previo a calcular la combinación de presiones aleatoria que se enviará al robot, es necesario calcular que vejigas de cada segmento se quedan sin hinchar en cada caso, lo cual se realiza de forma aleatoria.

En la figura 6.4 puede observarse un diagrama que ilustra el proceso descrito.

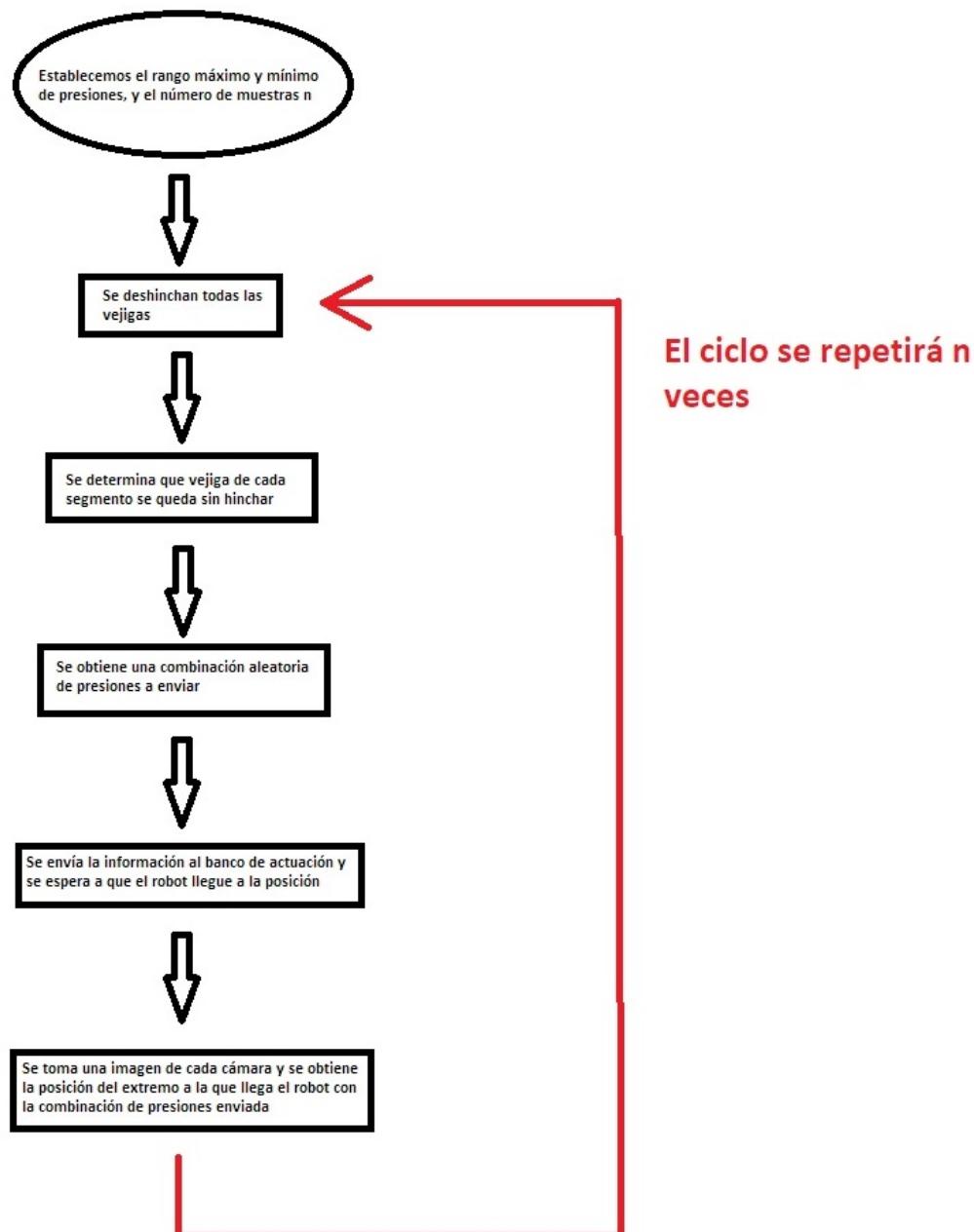


Figura 6.4: Proceso de toma de datos. *Fuente: elaboración propia*

Hay algunos aspectos relacionados con **tiempos** que vale la pena destacar. El primero está re-

lacionado con el hinchado y deshinchado de las vejigas. Este se realiza de forma porcentual, es decir, tal y como se explica en el capítulo 4: *Banco de actuación*, existirá una presión máxima relativa, equivalente al cien por ciento de actuación y un multiplicador, que compensa la diferencia entre el tiempo de hinchado y deshinchado de las vejigas. Este multiplicador se obtiene de forma experimental y es aproximadamente **0,45** para una presión de trabajo de **0,1 MPa** e hinchando y deshinchando **un solo grupo** de vejigas al mismo tiempo. Algo parecido ocurre con el tiempo equivalente al cien por ciento de actuación. Este, en el contexto de este trabajo, se refiere a que un grupo de vejigas estará actuado al cien por ciento cuando se llene durante esta cantidad de tiempo, hinchando únicamente ese grupo de vejigas.

Teniendo esto en cuenta, se ha determinado que la mejor opción en el momento de enviar las presiones al banco neumático, es realizarlo de una en una, esperando entre envíos el tiempo de llenado de la presión que se ha enviado anteriormente en vez de hinchar todas las vejigas al mismo tiempo.

#### 6.4. Filtros

¿Hablo sobre las funciones de filtros?

## 7. Interfaz de usuario y control

En el siguiente capítulo se tratará todo lo referente al control cinemático del *softbot* fruto de este trabajo, así como el software que se ha desarrollado para ello. Este software, al cual se ha denominado *SoftControl*, se encarga de sintetizar todo el trabajo realizado en este proyecto, en una sola aplicación amigable a un usuario no experto. Esta se ha realizado mediante la herramienta *App Designer* de Matlab, y posteriormente se ha exportado en forma de un paquete instalable.

### 7.1. Control

En este apartado se va a realizar una introducción del método de control que se ha utilizado para realizar el control cinemático del robot fruto de este trabajo para cumplir los objetivos establecidos. Estos objetivos se centran en la obtención de un robot que sea capaz de llevar su extremo a un punto definido previamente, tanto en posición como en orientación. Además, debe realizarlo con un mínimo de precisión y repetibilidad. Para ello se ha pensado en un método de control en lazo abierto que se ajuste a los objetivos a alcanzar.

Tal como se ha explicado en capítulos anteriores, en este trabajo se ha optado por métodos de control basados en datos experimentales, siendo así estos métodos empíricos, frente a la obtención de un modelo teórico que simule las características físicas del robot en cuestión. Dentro de esta clase de métodos nos encontramos por ejemplo los que se basan en redes neuronales. Estos se han considerado dentro de los métodos de control barajados, pero por el alcance del proyecto no se ha implementado.

Para el control del softbot desarrollado, se ha optado por un control mediante tabla. Esta tabla relaciona los distintos puntos del espacio de trabajo a los que el robot puede acceder, con el conjunto de señales que hay que enviar a los actuadores en el espacio de las articulaciones. De esta forma el modelo cinemático inverso estaría de alguna manera almacenado en la tabla. En el contexto de este trabajo, esta tabla consiste en los datasets que se recogen, de acuerdo con los criterios establecidos en el capítulo 6. De esta manera, precisión de posicionamiento del robot, dependerá de cómo cercano sea el punto deseado al más cercano que pertenezca a la cuadrícula de puntos por los cuales es posible controlar el robot.

A lo que se ha denominado cuadricula, es en realidad el conjunto de puntos que hay disponibles en dicha tabla. De esta manera, la precisión del posicionamiento aumentará con el número de muestras del dataset. En el momento que se desea que el robot vaya a una posición determinada, se busca el punto de la cuadricula que minimice la distancia euclídea a este.

A pesar de tratarse de un método de control algo rudimentario, se ha observado en las pruebas experimentales que da resultados satisfactorios, por lo que se reafirma la decisión de no haber implementado una red neuronal en el contexto de este proyecto.

De forma paralela, y a modo experimental, se ha implementado un método cuyo objetivo es disminuir el número de grados de libertad a controlar, de nueve a seis. A esto se le ha denominado control por *ángulo - curvatura* y se explica más adelante en este mismo capítulo. Reducir los parámetros de control del robot de nueve a seis, puede favorecer la utilización de una red neuronal en un desarrollo futuro.

## 7.2. Interfaz de usuario: SoftControl

Una parte de este proyecto, en la que más esfuerzo se ha invertido, ha sido en el desarrollo de un programa que incorpore todas las funcionalidades de este proyecto, y permita la reutilización de estas en otros trabajos futuros. Esta herramienta que se ha desarrollado, a la cual se ha denominado *SoftControl*, se trata de una aplicación con interfaz gráfica de usuario, que busca facilitar la interacción con el *SoftBot* resultante de este trabajo, así como sistematizar los procesos de adquisición de datos y las pruebas conceptuales.

Este programa que se ha desarrollado está asociado principalmente al banco de actuación usado en este proyecto. Se encarga de mandar las órdenes de control a alto nivel mediante la interfaz UART al microcontrolador que se encuentra en el banco. De este modo, la aplicación podrá utilizarse en otros proyectos que hagan uso del mismo banco neumático.

Aparte del control del banco de actuación, se han integrado todos los algoritmos relacionados con el sistema de visión por computador que se encarga de la adquisición de datos. Esto incluye desde la calibración de las cámaras, hasta los algoritmos que obtienen la posición y orientación de las marcas luminosas tratadas en el apartado 5.2.

El control cinemático basado en una tabla y el tratamiento de los datos recogidos también se puede realizar a través de la interfaz.

Durante el desarrollo de la interfaz se ha tenido especialmente presente el objetivo de que sea una aplicación reutilizable para otros proyectos que hagan uso del banco de actuación de este proyecto.

Todo esto busca facilitar el control del robot, así como de otros robots neumáticos que puedan realizarse posteriormente. Teniendo en cuenta este objetivo, de que sea un software que se pueda reutilizar, se ha tenido especialmente en cuenta la experiencia de usuario.

Para el desarrollo de la herramienta se ha utilizado *Matlab* y posteriormente *Matlab compiler* para empaquetar la aplicación en un instalable.

### 7.2.1. Estructura general

El programa se divide en los siguientes menús principales que cubren todas las funcionalidades desarrolladas:

- **Config:**

Pestaña dedicada principalmente a gestionar aspectos relacionados con la comunicación por puerto serie con el banco neumático, así como de ajustar distintos parámetros de este.

- **Visión por computador:**

Este menú tiene varias funcionalidades relacionadas con la visión por computador. Permite calibrar de forma sencilla los parámetros extrínsecos de las cámaras. Es decir, la posición de estas con respecto al sistema de referencia fijo. También permite realizar el *homing*, es decir, ajustar la posición del sistema de referencia que se desea usar para la adquisición de datos, con respecto a la posición del sistema de referencia del tablero de calibración.

Aparte de funciones de calibración, en esta pestaña se pueden obtener fácilmente la posición y orientación del sistema de referencia móvil, definido por las tres marcas luminosas.

**■ Control manual:**

Este menú sirve para el control directo del banco de actuación, es decir, de los distintos grados de libertad del robot en el contexto de este trabajo.

En otras palabras, permite controlar las distintas líneas neumáticas de las distintas maneras que se han desarrollado.

**■ Recogida de datos:**

Esta ventana sirve para configurar los parámetros de una sesión de recogida de datos. Es decir, establecer las presiones máximas, mínimas, número de muestras, ..., etc.

**■ Control mediante tabla:**

Esta pestaña permite cargar un dataset con el objetivo de llevar el extremo del robot a distintos puntos del espacio. Se podrán filtrar los datos del dataset deseado, así como enviar los datos al banco de actuación para que se ejecuten los movimientos.

Estas funciones se han programado mediante la herramienta *App Designer de Matlab*, pero también se han utilizado las distintas funciones desarrolladas en este proyecto.

A continuación, se explicarán estas ventanas de una forma más detalladas. Se detallará como usarlas, así como su funcionamiento interno.

### 7.2.2. Configuración

Como se ha dicho anteriormente, esta pestaña sirve principalmente para la comunicación por serial con el banco neumático. En la siguiente figura se puede apreciar el layout, y a continuación se detallan los elementos que aparecen en este

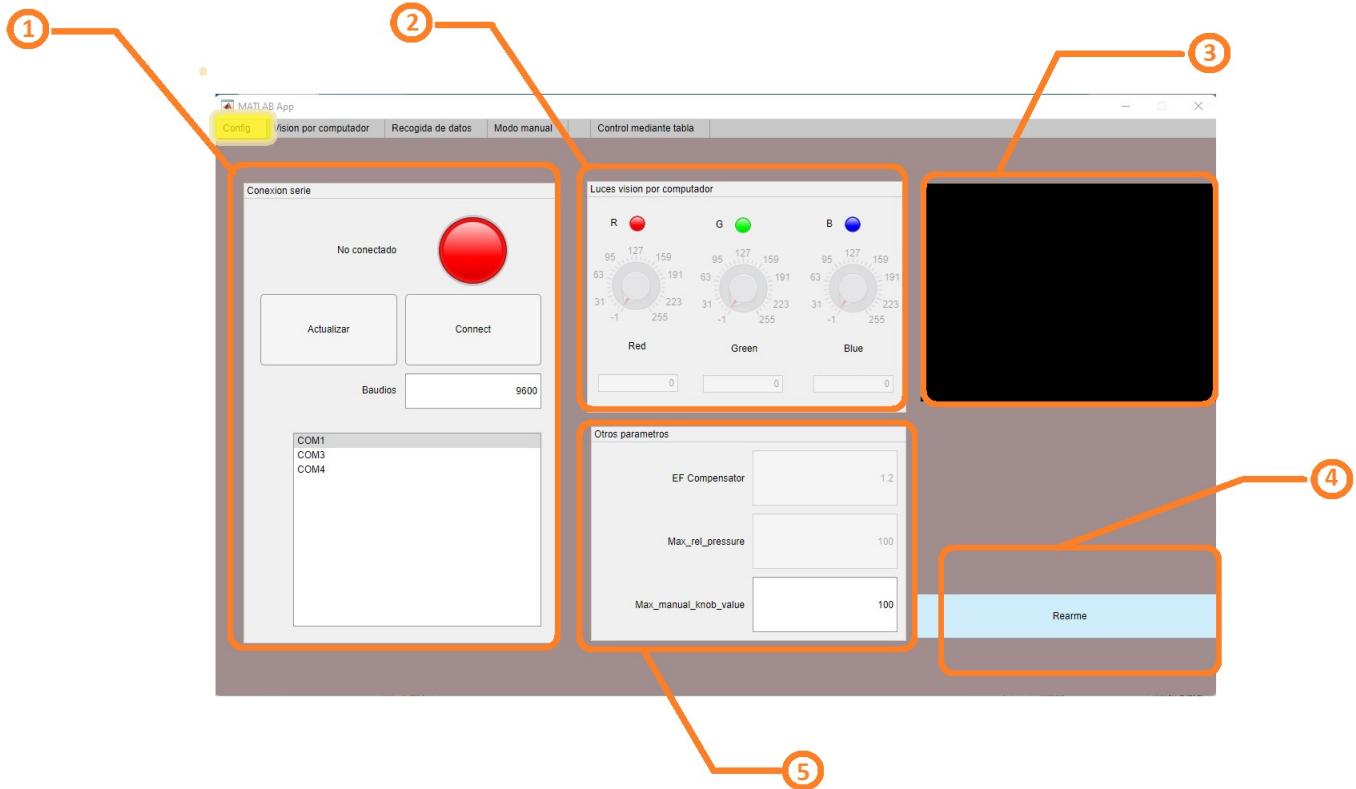


Figura 7.1: SoftControl: Configuración. *Fuente: elaboración propia*

### 1. Configuración del puerto serie:

Este apartado sirve para establecer la comunicación en el puerto serie. En el recuadro de texto se introduce la frecuencia a la que se realizará la comunicación en baudios. Posteriormente, al pulsar el botón *Actualizar*, se mostrarán los distintos puertos serie disponibles en la lista que hay a continuación. Para establecer la conexión con el banco de actuación, se selecciona el puerto serie correspondiente a este y se pulsa el botón *Conectar*.

Una vez realizado esto se intentará establecer la comunicación serial.

### 2. Configuración de las marcas luminosas

En este recuadro se puede configurar el brillo de cada una de las tres marcas luminosas que utiliza el sistema de adquisición de datos. Para apagarlas, basta con ponerlas en cero.

### 3. Salida

En este recuadro de texto aparecerán distintos mensajes al interactuar con distintos elementos de la pestaña configuración. Como por ejemplo si se ha efectuado correctamente la comunicación serial o si ha habido algún error.

### 4. Configuración de parámetros del banco de actuación.

En este recuadro se ajustan tres parámetros del banco de actuación relacionados con el control de las válvulas neumáticas.

- **EF\_Compensator:** Parámetro para compensar la diferencia de caudal dependiendo del sentido del aire comprimido en la línea neumática. Al hinchar hay un caudal

mayor que al deshinchar. Experimentalmente, se ha comprobado que la relación entre la diferencia de estos caudales es aproximadamente lineal. Para el robot desarrollado en este trabajo se ha obtenido que el valor de este parámetro que mejor funciona es aproximadamente 1.45.

- **Max\_rel\_pressure:** Este parámetro entra en juego cuando se utilizan las válvulas en modo relativo. Es decir, en vez de enviar el tiempo de apertura o cierre, se envía un porcentaje, que indica un *porcentaje de hinchado*, tal y como se explica en el capítulo 4. En este modo, el valor que se indica en este recuadro son los milisegundos de llenado equivalentes al cien por ciento de actuación.
- **Max\_manual\_knob\_value:** Simplemente es un parámetro que establece el valor máximo de los knobs que aparecen en el control manual.

### 5. Botón de rearme

En el capítulo 4 se ha explicado que el banco de actuación tiene un sistema de seguridad a bajo nivel, por el cual las válvulas nunca van a abrir una línea neumática a presión más de un tiempo establecido. En caso de alcanzar este valor, el banco neumático se pone en modo de emergencia y no permite realizar ninguna acción.

Este botón devuelve el banco al modo normal.

Para la comunicación serial con el microcontrolador se ha utilizado la función *serialport*. De este modo, al pulsar el botón connect, se creará un objeto serial a nivel interno, que se almacena en un atributo del programa.

Las órdenes se enviarán en forma de *string* mediante la función *writeline*.

### 7.2.3. Visión por computador

Esta pestaña de la interfaz está dedicada a configurar todos los parámetros necesarios para el sistema de visión por computador que se ha utilizado en este proyecto. Esto incluye desde la calibración de los parámetros extrínsecos de las cámaras hasta el ajuste de la posición del sistema de referencia fijo respecto al cual se calcula la posición. A continuación, en la figura 7.2, se puede apreciar el layout de esta ventana.

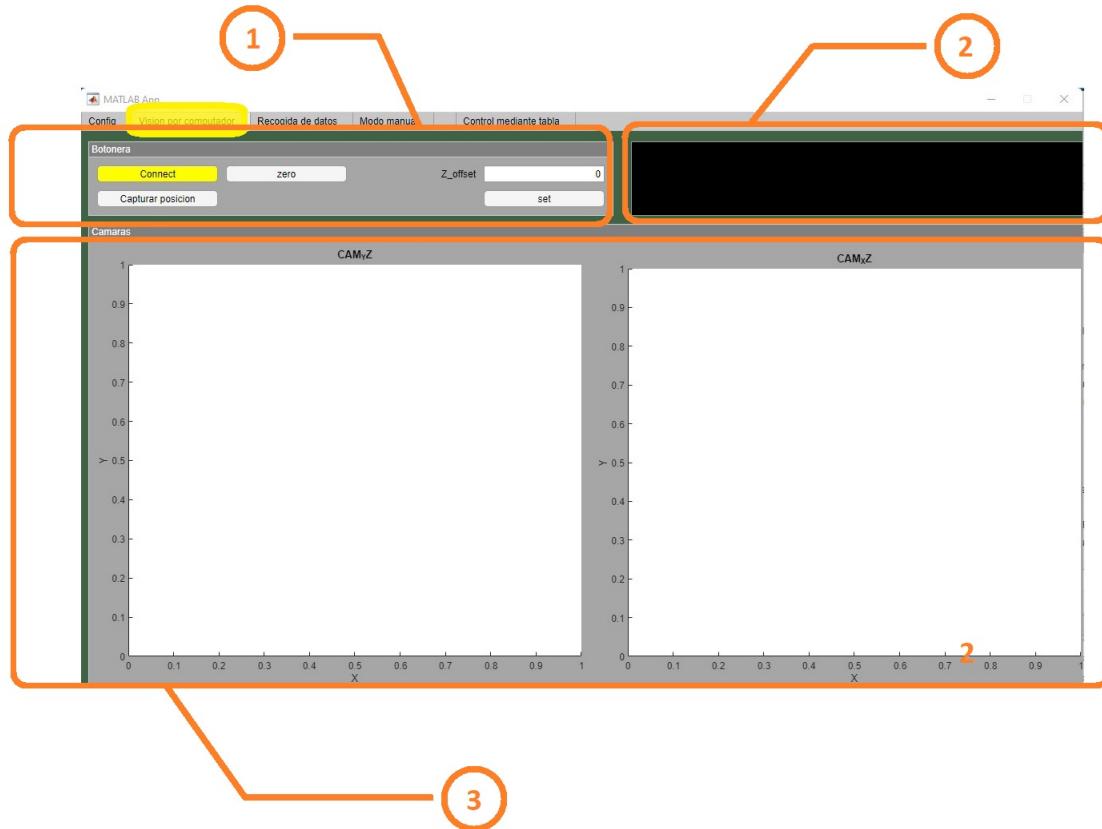


Figura 7.2: SoftControl: Visión por computador. *Fuente: elaboración propia*

#### 1. Botonera de control

Elementos para interactuar con esta parte de la interfaz

- **Connect**

La función de este botón es de inicializar todos los elementos del sistema de visión por computador. En el momento que se desee realizar esta inicialización, deberán conectarse las dos cámaras al ordenador mediante USB y colocar el tablero de calibración en frente de estas. Es importante que la cuadrícula quede completamente en el campo de visión de ambas cámaras. También es importante que la orientación del tablero con respecto a las cámaras sea similar a la del sistema de referencia fijo que se desea utilizar.

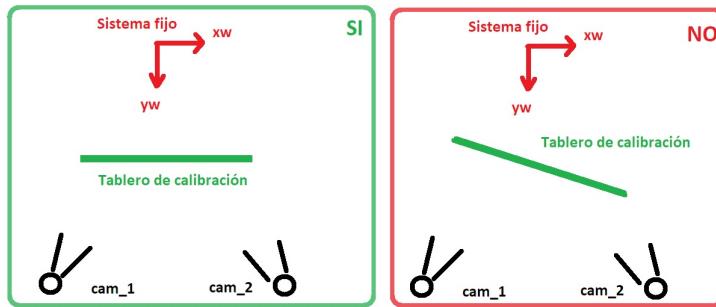


Figura 7.3: SoftControl: Orientación tablero. *Fuente: elaboración propia*

Una vez colocado de forma correcta el tablero, al pulsar el botón connect, se realizará la conexión de las cámaras de forma automática. Estas tomarán una imagen y realizarán la calibración de sus respectivos parámetros extrínsecos. Se mostrará en una nueva ventana la orientación de las cámaras con respecto al tablero que se ha calculado.

Una vez realizado todo esto, se podrán utilizar el resto de los botones.

#### ■ Zero

Al presionar este botón, las cámaras tomarán dos imágenes que se mostrarán en las ventanas correspondientes. En estas el programa busca la marca luminosa verde, la cual es el origen del sistema de referencia fijo. Respecto a esta se calculan los parámetros dx, dy y dz explicados en el capítulo 5.2. Se realizará la transformación correspondiente al sistema de referencia del tablero para que coincida con el sistema de referencia deseado para tomar los datos.

#### ■ Capturar posición

Este botón permite identificar la posición del sistema de referencia móvil con respecto al fijo que se ha ajustado previamente. Para que esta función haga su trabajo correctamente, es necesario ajustar el brillo de las marcas luminosas mediante los diales de la pestaña *Config*. Cuando se presiona este botón, las cámaras, junto a las funciones que se han desarrollado, identifican estas marcas luminosas y calculan la posición y orientación del sistema de referencia móvil

#### ■ Z\_offset

Este recuadro permite introducir una desviación para *subir o bajar* el sistema de referencia fijo con respecto a su eje z.

#### ■ Set

Configura la distancia de desviación en z, indicada en el recuadro correspondiente.

## 2. Output

Este cuadro de texto cumple la función de mostrar la salida de esta parte del programa. Al pulsar el botón *Capturar posición*, se mostrará la posición identificada en milímetros y la orientación en ángulos de Euler en grados. Además, al presionar el botón *zero*, se mostrarán los parámetros dx, dy y dz calculados.

## 3. Visión de las cámaras

En estas dos ventanas se muestran las imágenes tomadas por las cámaras sobre las cuales se superponen unas marcas sobre los elementos de la imagen identificados.

#### 7.2.4. Modo manual

Esta parte del programa se enfoca al control del *SoftBot* o de las válvulas del banco de actuación, sin ningún tipo de control cinemático. Esta parte del programa es fácilmente reutilizable para otros trabajos futuros que hagan uso del mismo banco neumático.

La principal función que se le ha dado a esta pestaña ha sido la de ensayar las diferentes versiones de los segmentos que se han desarrollado. Gracias a las funcionalidades que se han implementado es posible un control sencillo del banco de actuación a alto nivel que agiliza sustancialmente la realización de los experimentos.

Esta ventana (figura 7.4) dispone a su vez de tres sub-modos que se corresponden a distintas maneras de controlar las nueve líneas neumáticas de las que dispone el banco.

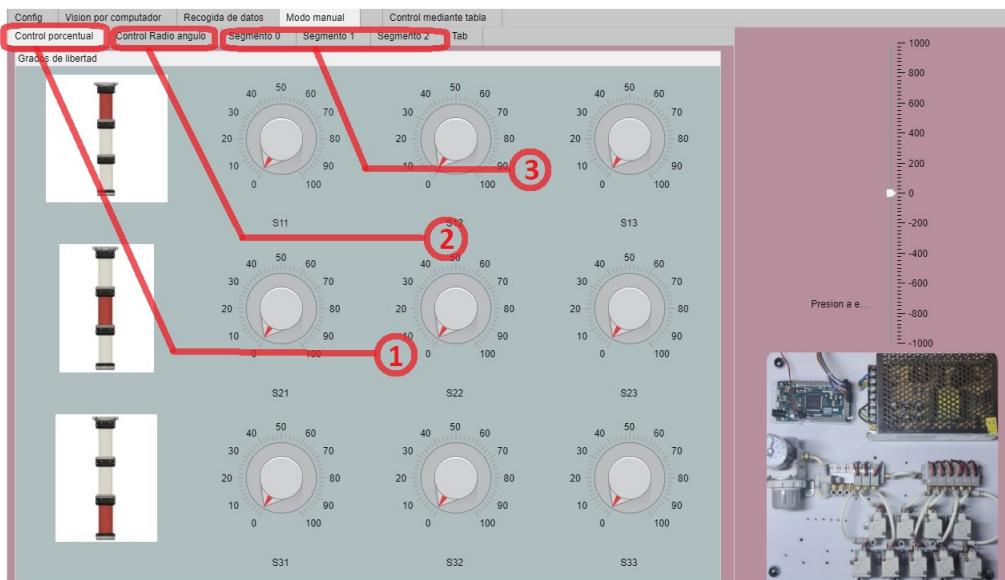


Figura 7.4: SoftControl: Modo manual. *Fuente: elaboración propia*

##### 1. Control porcentual

Este modo de funcionamiento corresponde al control porcentual de las líneas neumáticas que se ha explicado en el apartado 4. En vez de mandar los milisegundos de llenado/vaciado directamente, se envía el porcentaje de actuación deseado. El cien por ciento se corresponde al tiempo de llenado que hace que un segmento del robot se hinche lo máximo posible. Este valor se determina experimentalmente. El desarrollo de esta forma de controlar las válvulas permite un funcionamiento mucho más seguro del robot al estar acotado el tiempo máximo que pueden hinchar las válvulas.

##### 2. Control radio - ángulo

Esta es una forma experimental de control que se ha desarrollado paralelamente en el transcurso de este trabajo. Se explica de una forma más detallada en el apartado correspondiente.

##### 3. Control directo

Permite abrir o cerrar las válvulas de forma bruta. Se envían directamente los milisegundos que se desea que cada una de las válvulas esté a presión, al aire o cerrada. No se recomienda

su uso para el robot desarrollado en este trabajo, dada la facilidad de pasarse de tiempo llevando a la rotura de algún segmento. Sin embargo, se trata de una funcionalidad muy práctica para darle otros usos al banco de actuación.

A continuación, se explicará cada uno de estos tres subapartados en mayor detalle.

#### 7.2.4.1 Control porcentual

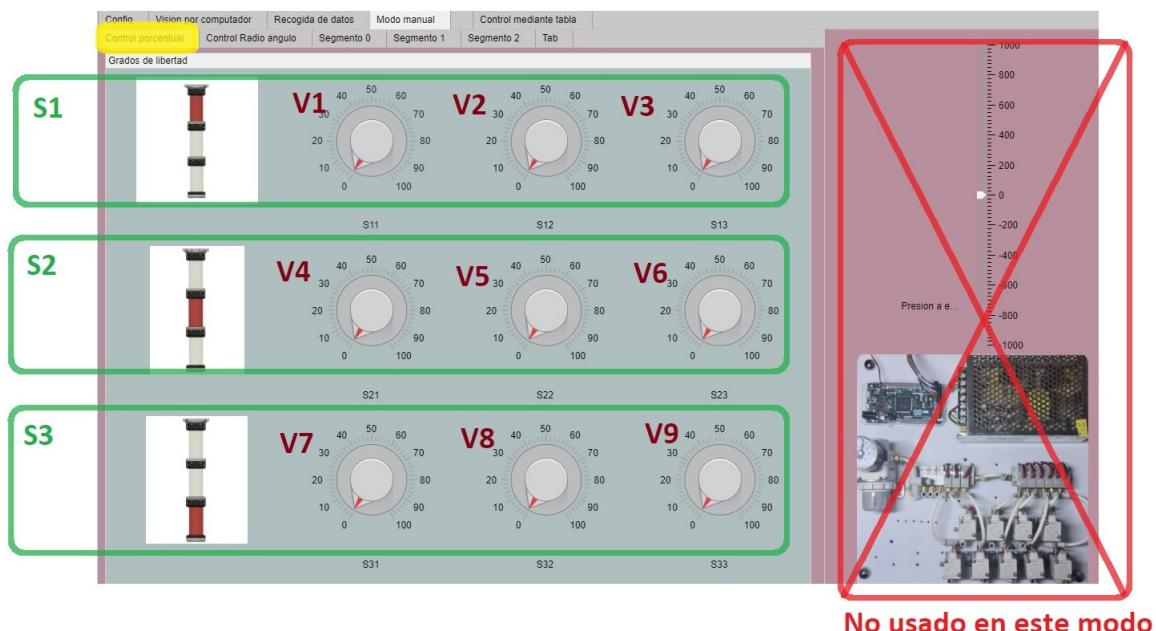


Figura 7.5: SoftControl: Modo manual - Control porcentual. *Fuente: elaboración propia*

Este modo, tal y como se ha explicado anteriormente, sirve para el control de las líneas en modo porcentual. En la figura 7.5 puede apreciarse el layout de esta parte de la interfaz. Este consiste esencialmente en nueve diales, uno para cada grado de libertad, en los cuales se puede indicar el porcentaje de actuación de cada uno de estos. Los diales están ordenados por orden de válvulas del banco neumático y cada fila se corresponde a un segmento del robot.

Los diales normalmente permiten seleccionar un rango continuo de valores entre cero y cien, pero admiten una limitación de este rango para mayor seguridad. Este valor máximo se establece en la pestaña *config* de la interfaz.

En el momento en el que se modifica el valor de uno de estos diales, se envía la orden correspondiente al banco de actuación mediante la función *writeline*.

```

1 % Value changed function: S23Knob
2     function S23KnobValueChanged(app, event)
3         value = app.S23Knob.Value;
4         writeline(app.mi_arduino, "x,5," + num2str(value));
5     end

```

#### 7.2.4.2 Control curvatura - ángulo

Esta forma de controlar el robot se ha desarrollado de forma paralela al desarrollo de este trabajo a modo experimental. Se basa en asumir que para cada pose del robot siempre quedará un grupo de vejigas sin actuar. Partiendo de esta hipótesis, se ha buscado una forma de reducir los grados de libertad del robot de nueve a seis. Esto sería posible en caso de que fuera posible determinar de alguna manera que vejigas se quedan, se hinchan y cuáles no. De esta manera se busca una forma de caracterizar cada segmento del robot por únicamente dos parámetros, en vez de los tres valores de las presiones de los grupos de vejigas. Uno de estos parámetros define el radio de curvatura del segmento y el segundo la dirección de esta curvatura en grados. En la siguiente imagen se puede apreciar la interfaz de este modo de funcionamiento.

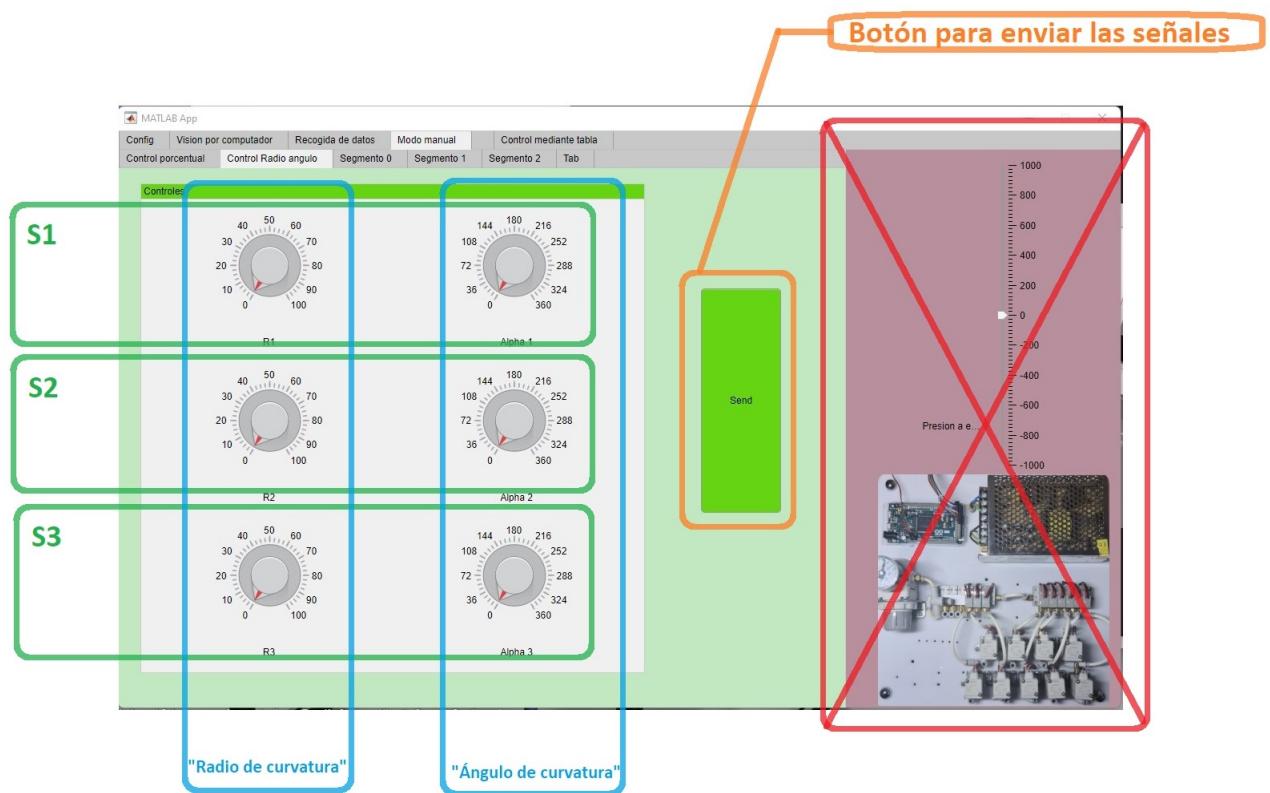


Figura 7.6: SoftControl: Modo manual - Control Radio ángulo. Fuente: elaboración propia

Para cada segmento del robot, se puede seleccionar el valor para la curvatura y el ángulo deseados. En el momento en el que se pulsa el botón *send* se enviarán las órdenes al banco neumático y el robot adoptará la pose correspondiente a los valores introducidos.

A continuación, se procede a explicar el fundamento en el que se basa esta forma de controlar el robot. Como se ha dicho anteriormente, este procedimiento se basa en controlar cada segmento mediante un factor que define *cuanto se dobla* el segmento, y otro que define *hacia donde se dobla*. Esto se puede observar de forma gráfica en la figura 7.7.

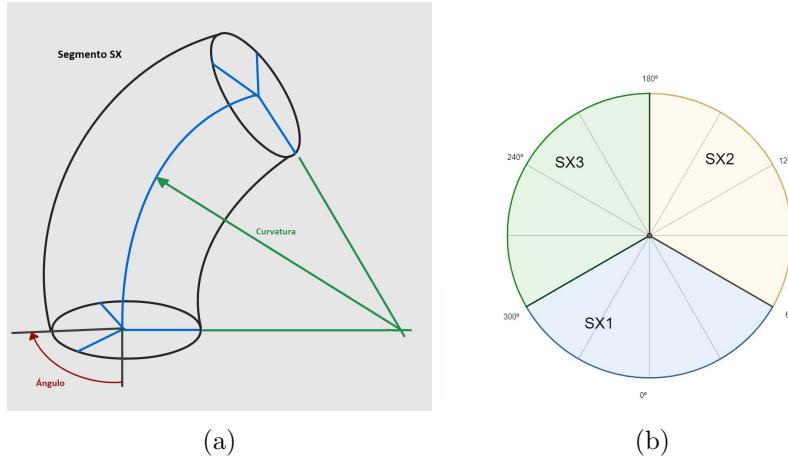


Figura 7.7: SoftControl: Control curvatura - ángulo. Fuente: elaboración propia

El primer parámetro, al que se denominará curvatura, define cuantos flecta el segmento. Este sistema de control hará uso del sistema de control porcentual de las vejigas, de este modo una curvatura de cien equivale al cien por ciento de actuación. De este modo, en un segmento del robot, se envía a cada grupo de vejigas la curvatura deseada multiplicada por un factor multiplicador. Este factor multiplicador lo define una función a la cual se ha denominado *gamma*. Esta función devuelve un valor comprendido entre cero o uno que depende del ángulo dado como parámetro.

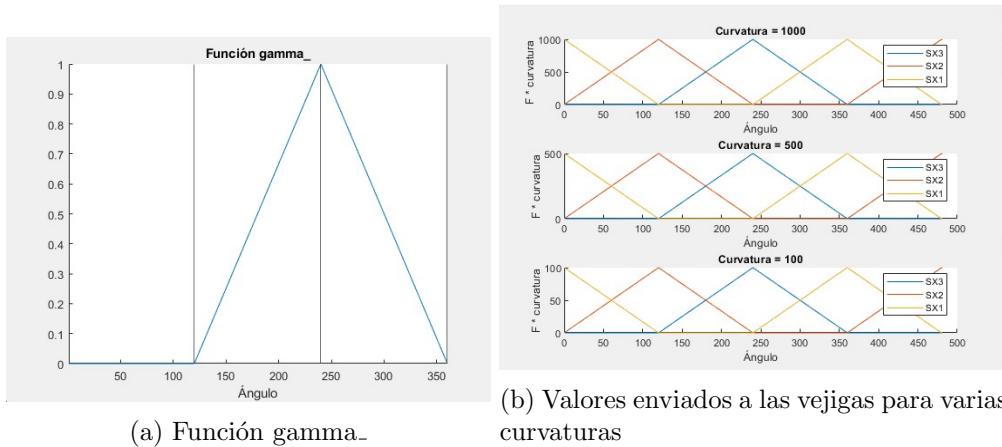


Figura 7.8: SoftControl: Control curvatura - ángulo: función gamma. Fuente: elaboración propia

De este modo, las presiones que se enviarán a las vejigas de un segmento *SX* se determinan de la siguiente manera:

$$\begin{cases} \text{Segmento } SX1: & presion\_a\_enviar = curvatura * gamma_(ang + 240) \\ \text{Segmento } SX2: & presion\_a\_enviar = curvatura * gamma_(ang + 120) \\ \text{Segmento } SX3: & presion\_a\_enviar = curvatura * gamma_(ang) \end{cases} \quad (7.1)$$

#### 7.2.4.3 Control directo

Este sub-modo de la pestaña de control manual del software desarrollado, sirve para controlar las válvulas del banco de forma directa. Con esto nos referimos a enviar directamente cuantos milisegundos se desea que estas estén en un modo u otro.

Este modo de controlar el banco neumático dispone de tres pestañas, una para cada segmento del robot. En cada una de estas hay tres botones de enviar y tres indicadores que marcan el total de presión en cada válvula, en milisegundos. En la parte derecha del programa nos encontramos un deslizador lineal que permite seleccionar cuantos milisegundos se desea hinchar o deshinchar. Si el valor es positivo, las válvulas hincharán, es decir, se pondrán a presión durante los milisegundos especificados. Si el valor es negativo, las válvulas se abrirán durante el tiempo indicado, deshinchando la vejiga a la que estén conectadas.

En el momento que se pulse el botón enviar de alguna de las válvulas, se enviará a dicha válvula el tiempo especificado en el deslizador lineal. Además, en este momento se actualizará el indicador asociado. Estos indicadores funcionan en lazo abierto, es decir, no reciben realimentación del banco de actuación. El valor que indican es una estimación a partir del histórico de presiones que se ha enviado.

Es importante aclarar que en este modo no hay ninguna función de seguridad a alto nivel que impida sobre hinchar algún segmento ocasionando la rotura de este. Debido a esto se recomienda usar este modo con cautela en el momento que haya algún elemento conectado a las válvulas que sea sensible a sobrepresiones.

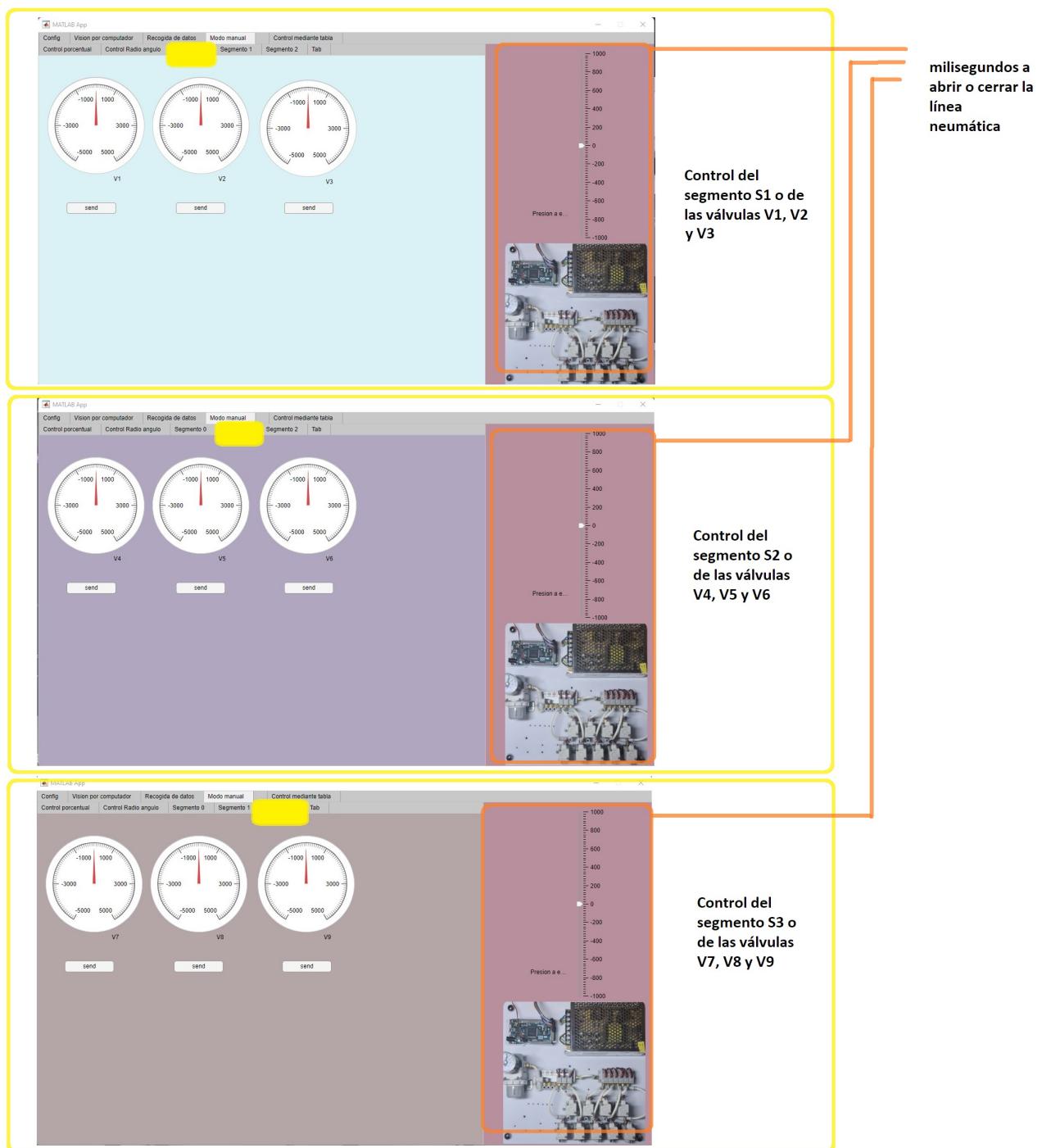


Figura 7.9: SoftControl: Modo manual - Control directo. *Fuente: elaboración propia*

### 7.2.5. Recogida de datos

Esta pestaña de la interfaz tiene la función de automatizar la recogida de los *datasets* que se han explicado en el capítulo 6 *Diseño experimental*. A continuación, se van a explicar los diferentes elementos de los que consta el layout que se observa en la figura 7.10.

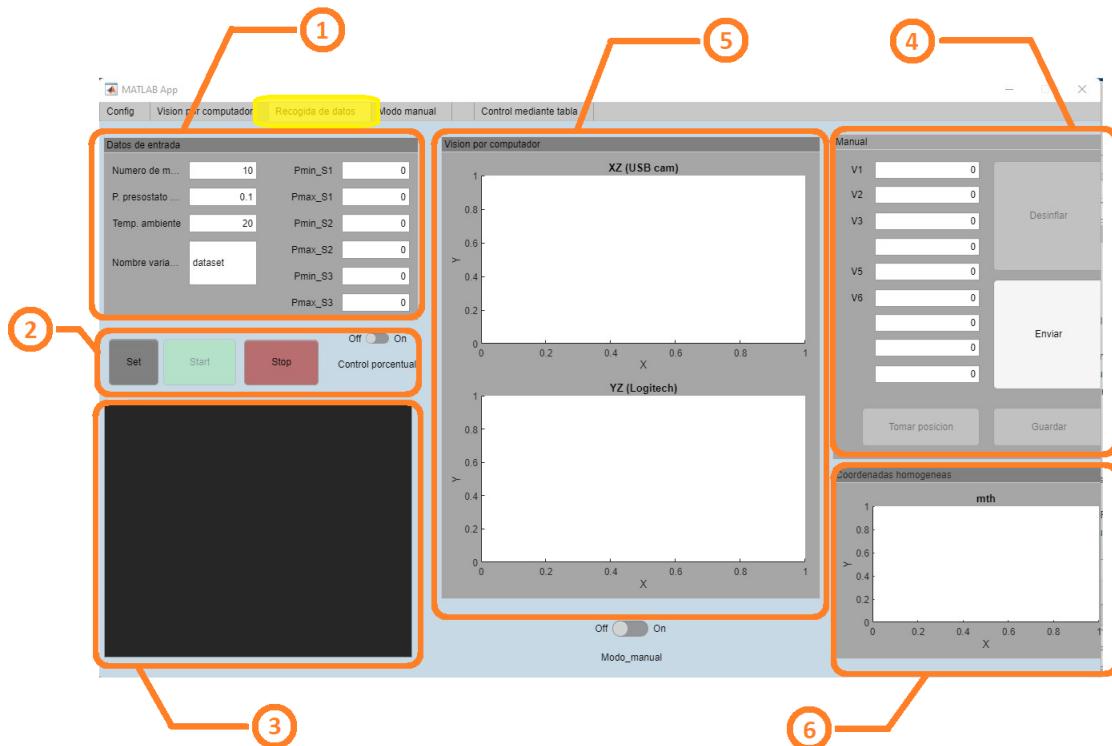


Figura 7.10: SoftControl: Recogida de datos. *Fuente: elaboración propia*

### 1. Parámetros del dataset

Este apartado de la ventana sirve para configurar las condiciones bajo las cuales se desea que se recojan los datos. Se pueden configurar todos los parámetros de los que debe disponer cada dataset, tal y como se ha especificado en el capítulo 6 de esta memoria. A continuación, se hace un breve repaso del significado de cada uno de estos:

- **Número de muestras:**

Número de muestras de las que dispondrá el dataset. Determina el número de iteraciones que realizará el algoritmo de recogida de datos.

- **P. presostato:**

Presión que marca el presostato del banco de actuación en el momento que se va a iniciar la recogida de datos. Se introduce en unidades de MPa.

- **Temp. ambiente:**

Temperatura ambiental del recinto en el que se encuentra el robot medida en grados centígrados.

- **Nombre dataset:**

Nombre que se desea dar al dataset.

- **Pmax\_Sn:**

Presiones máximas que se enviarán a las vejigas en cada iteración. En porcentaje en caso de realizarse en modo porcentual, y en milisegundos en caso contrario.

- **Pmin\_Sn:**

Presiones mínimas que se enviarán a las vejigas en cada iteración. En porcentaje en caso de realizarse en modo porcentual, y en milisegundos en caso contrario.

## 2. Botonera de control

Botones que sirven para inicializar la recogida de datos automática.

- **Set**

Guarda los datos que se han especificado en el apartado *Parámetros del dataset*, y desbloquea el botón iniciar.

- **Start**

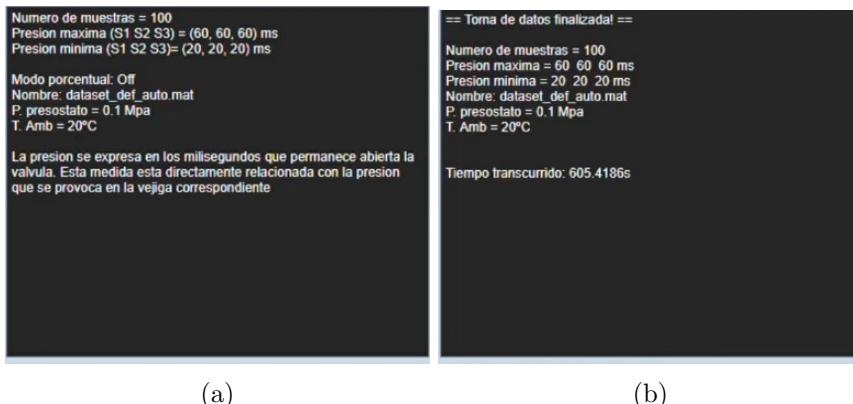
En el momento que se presiona este botón comenzará la toma de datos. Se realizarán el número de iteraciones indicadas en el parámetro *número de muestras..*. Es importante, que, una vez terminado este proceso, si se desea que el dataset generado se almacene, se pulse el botón *Guardar* del apartado *Modo manual*

- **Stop**

Botón que detiene la toma de datos automatizada

## 3. Salida

Este elemento del layout se trata de un cuadro de texto que muestra toda la información que de interés al usuario. Algunos ejemplos de esto son, por ejemplo: los datos que se han establecido para la recogida del dataset, los datos obtenidos en cada iteración, o el tiempo que ha tardado en tomarse la muestra. Se muestra a continuación un ejemplo de ello:



(a)

(b)

Figura 7.11: SoftControl: Ejemplo salida toma de datos. *Fuente: elaboración propia*

## 4. Modo manual:

Permite llevar el robot a una posición determinada por el usuario, en vez de una que genere aleatoriamente el algoritmo. Para ello, se introducen las presiones deseadas en porcentaje y se pulsa el botón *Enviar*. Una vez que el robot alcanza la pose deseada, se puede pulsar el botón *Tomar muestra*. Esto hará que se capten los datos necesarios y se añadan de forma automática al dataset.

## 5. Visión de las cámaras

Son dos gráficos en los que se pueden visualizar la imagen captada por las cámaras en cada iteración, con una marca superpuesta sobre los elementos identificados en las imágenes (7.12). Es un elemento de gran utilidad debido a que la toma de datos, tal y como se ha detallado en el capítulo 6, se realiza a oscuras. De esta manera se puede montar el ordenador en un lugar aparte y verificar que no hay ningún problema durante el proceso.

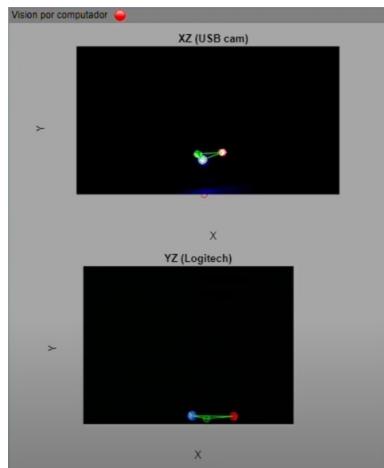


Figura 7.12: SoftControl: Ejemplo visión de las cámaras. *Fuente: elaboración propia*

## 6. Rotaciones

En este gráfico aparecerá la orientación del sistema móvil con respecto al fijo en cada iteración del sistema. Es una forma más de verificar que el sistema de visión está funcionando correctamente.

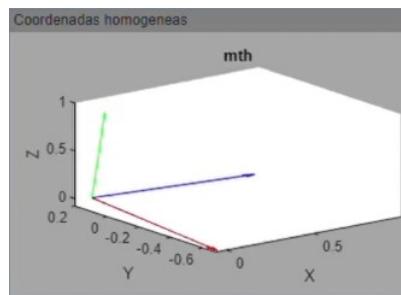


Figura 7.13: SoftControl: Ejemplo rotaciones. *Fuente: elaboración propia*

### 7.2.5.1 Procedimiento de uso

A modo de facilitar al uso de esta herramienta, se va a detallar brevemente la forma sistemática de usar esta parte de la interfaz para recoger datos de forma satisfactoria. El primer paso es realizar y configurar correctamente la conexión con el banco neumático en la ventana de configuración. Una vez realizado esto, se conectan y calibran las cámaras en la pestaña de visión por computador.

Una vez configurado el hardware correctamente, ya se puede pasar a esta ventana y configurar los datos con los que se desea realizar el proceso de recogida de datos. Cuando este paso se ha realizado, se pulsa el botón *set* y se verifica en la salida que todos los datos son correctos. En caso de que así sea ya se puede pulsar el botón *Start* y comenzar el proceso.

### 7.2.5.2 Funcionamiento interno

### 7.2.6. Control mediante tabla

El resto de las ventanas del programa explicadas hasta este punto tenían una finalidad más centrada en la parte experimental de este trabajo. Por otra parte, esta parte del software se ha diseñado con la funcionalidad de, usando los datasets que se generan con los demás elementos de la interfaz, mover el robot de una forma coherente.

Tal y como se ha explicado en el apartado 7.1: Control, se ha optado por un control mediante tabla. Esto se refiere a disponer de una tabla que barra todas las posiciones del espacio de trabajo con una resolución determinada, y relacione estas con las coordenadas del espacio de las articulaciones que han de enviarse al robot para que alcance dicha posición.

En esta ventana, se pone a disposición del usuario una forma sencilla de realizar este control en el robot fruto de este trabajo. Para ello se utilizan los datasets generados mediante la herramienta de recogida de datos explicada en el apartado anterior.

Debido a las características de esta parte del programa, se va a explicar, centrándonos más en el uso que en los elementos de la interfaz.

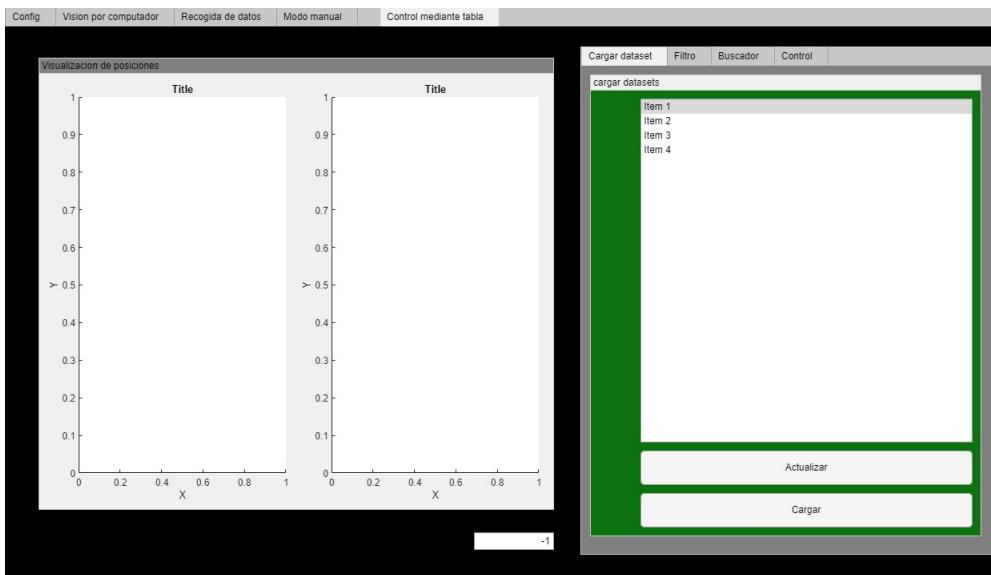


Figura 7.14: SoftControl: Control. *Fuente: elaboración propia*

Esta parte de la interfaz (figura 7.14), se divide en dos partes principales. A la izquierda, dos gráficas en las que se muestran los puntos del dataset seleccionado, antes y después de filtrarlo. A la derecha, distintas pestañas que se encargan de realizar las operaciones necesarias. Estas son: Cargar dataset, Filtro, Buscador y Control.

A continuación, se explicará cómo realizar un control entre varios puntos utilizando estas distintas pestañas:

### 7.2.6.1 Carga de un dataset

El primer paso consiste en cargar los datasets en la herramienta de control. Para ello, estando en la pestaña *Cargar dataset*, se pulsa el botón *Actualizar*. Una vez realizado esto, se mostrarán en la lista todos los datasets disponibles. Para que estos se muestren, sus ficheros correspondientes tienen que encontrarse en el mismo directorio que la interfaz.

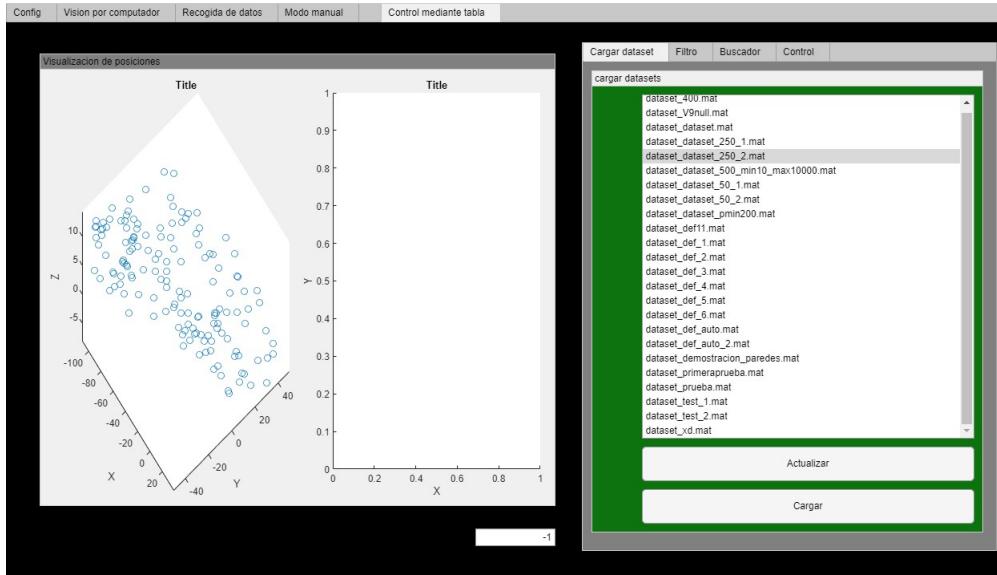


Figura 7.15: Control: Carga de un dataset. *Fuente: elaboración propia*

Una vez seleccionado el dataset deseado, se pulsa el botón *Cargar*. Una vez hecho esto, la herramienta cargará de forma interna el dataset, y se mostrarán en el primer gráfico todos los puntos de los que este dispone, tal y como se puede observar en la figura 7.15.

### 7.2.6.2 Filtrado del dataset

En este paso se procede a filtrar los puntos del dataset para eliminar los que no interesan. Este filtrado permite eliminar los índices en los cuales alguna coordenada del punto se salga de un rango, que se puede especificar en la interfaz mediante los deslizadores lineales que se aprecian en la figura 7.16. Estos incluyen tanto las coordenadas x, y, z, que se muestran en los gráficos, como también los ángulos de Euler que definen las rotaciones del extremo. De esta manera se puede filtrar el dataset para, por ejemplo, solo incluir puntos en los que el extremo del robot se encuentra paralelo al plano del suelo. El filtrado, además de descartar los puntos que quedan fuera de los rangos establecidos por el usuario, también descarta de forma automática los puntos que por una razón u otra no se han identificado correctamente. Estos son lo que alguna de sus coordenadas es -1.

Una vez establecidos los rangos deseados se pulsa el botón **Filtrar**. En este momento se mostrará en la segunda gráfica el conjunto de los puntos resultante después del filtrado. En la figura 7.16 se puede observar un ejemplo de esto. En la esquina inferior derecha se ha incluido un recuadro en el que figura el número de puntos disponibles después del filtrado

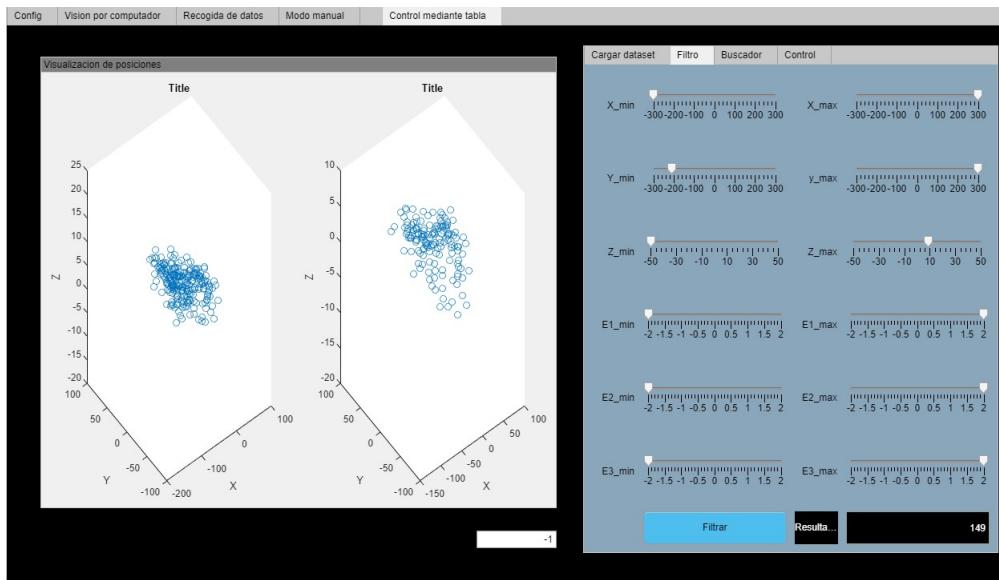


Figura 7.16: Control: Filtrado del dataset. *Fuente: elaboración propia*

Para filtrar los datasets se ha desarrollado la función **datafilter.m**

```
1 function [data_filtered] = datafilter(data, x,y,z,e1,e2,e3)
```

Esta función recibe como parámetros el dataset a filtrar, y los seis rangos correspondientes a las distintas coordenadas.

### 7.2.6.3 Selección de los puntos de interés

Una vez filtrado el dataset, ya se pueden especificar los puntos a los que se desea que vaya el robot. Para ello se ha desarrollado una funcionalidad de búsqueda (7.17). Esta parte de la interfaz permite buscar distintos puntos del espacio a los que se desea que el robot vaya y la herramienta buscará el punto del dataset que minimice la distancia euclídea al punto especificado.

Para especificar estos puntos que se desean buscar, se han implementado dos opciones. La primera de ellas consiste en introducir las coordenadas manualmente en los recuadros disponibles, y ajustar mediante los controles rotativos los ángulos de Euler deseados. La segunda opción es especificar los puntos mediante la visión por computador. De este modo, el usuario puede colocar la baliza luminosa en el punto del espacio deseado y pulsar el botón *Captar mediante CV*. En este momento, en caso de estar las cámaras configuradas correctamente, se captará el punto de forma automática utilizando las funciones de visión desarrolladas.

Una vez especificado el punto deseado se pulsa el botón *Buscar*. En este momento se realizará la búsqueda entre todos los puntos del dataset, y se mostrará en el cuadro de salida el punto más cercano encontrado. Si este punto satisface las necesidades del usuario, se pulsa el botón, *añadir*. Esto añadirá el punto a una lista de puntos a los que ir, que se explicará en el siguiente apartado.

## 7. Interfaz de usuario y control

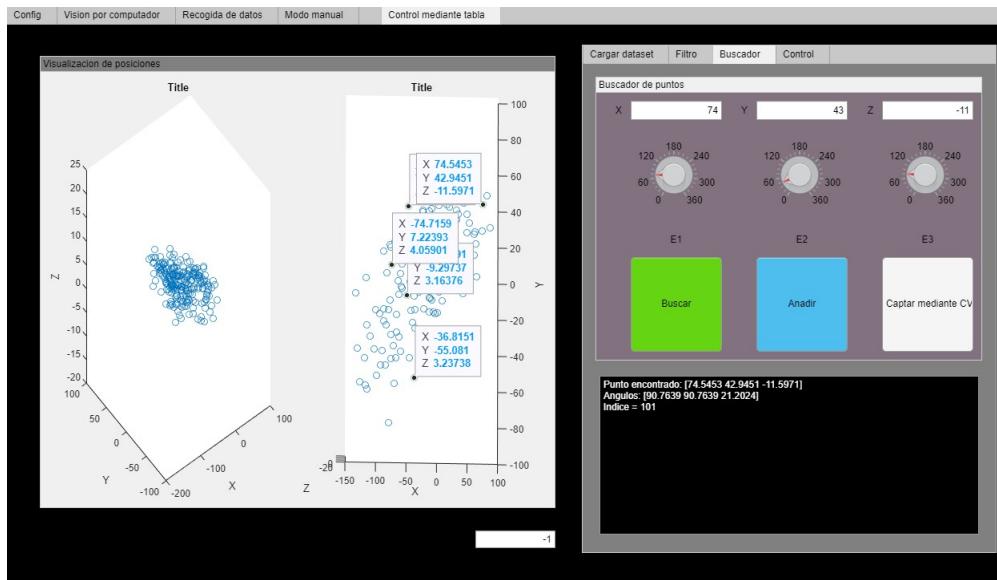


Figura 7.17: Control: Búsqueda de los puntos de interés. *Fuente: elaboración propia*

Para realizar esta búsqueda se ha desarrollado la función **buscador\_posición.m**, cuyo código se puede ver a continuación:

```

1 function [punto, angulos, presiones, idx_] = buscador_posicion(data, punto,
2   euler, index)
3
4 if euler ~= -1
5   busqueda_angulo = true;
6 else
7   busqueda_angulo = false;
8 end
9
10 if index ~= -1
11   busqueda_indice = true.
12 else
13   busqueda_indice = false;
14 end
15
16 if busqueda_indice == false
17   if busqueda_angulo == false
18     for i = 1:data.nummuestras
19       arr = [data.outputs(i,1), data.outputs(i,2),data.outputs(i,3) ;
20 punto];
21       distArr(i) = pdist(arr, 'euclidean');
22     end
23     [~,idx] = min(distArr);
24     idx_ = idx;
25     punto = [data.outputs(idx,1), data.outputs(idx,2),data.outputs(idx
26 ,3)];
27     angulos = [data.outputs(idx,4), data.outputs(idx,5),data.outputs(idx
28 ,6)];
29     presiones = [data.inputs(idx,1),data.inputs(idx,2),data.inputs(idx
30 ,3),data.inputs(idx,4),data.inputs(idx,5),data.inputs(idx,6),data.inputs(idx
31 ,7),data.inputs(idx,8),data.inputs(idx,9) ];
32   else
33     punto_(1:3) = punto;
34     punto_(4:6) = euler;
35     for i = 1:data.nummuestras
36       arr = [data.outputs(i,1), data.outputs(i,1),data.outputs(i,1),
37 data.outputs(i,4), data.outputs(i,5),data.outputs(i,6) ; punto_];
38     end
39   end
40 end

```

```

31         distArr(i) = pdist(arr, 'euclidean');
32     end
33     [~,idx] = min(distArr);
34     idx_ = idx;
35     punto = [data.outputs(idx,1), data.outputs(idx,1),data.outputs(idx
36     ,1)];
36     angulos = [data.outputs(idx,4), data.outputs(idx,5),data.outputs(idx
37     ,6)];
37     presiones = [data.inputs(idx,1),data.inputs(idx,2),data.inputs(idx
38     ,3),data.inputs(idx,4),data.inputs(idx,5),data.inputs(idx,6),data.inputs(idx
39     ,7),data.inputs(idx,8),data.inputs(idx,9) ];
39     end
40 else
41     for idx = 1:data.nummuestras
42         if data.index(idx) == index
43             angulos = [data.outputs(idx,4), data.outputs(idx,5),data.outputs(idx
44             ,6)];
44             presiones = [data.inputs(idx,1),data.inputs(idx,2),data.inputs(idx
45             ,3),data.inputs(idx,4),data.inputs(idx,5),data.inputs(idx,6),data.inputs(idx
46             ,7),data.inputs(idx,8),data.inputs(idx,9) ];
46             punto = [data.outputs(idx,1), data.outputs(idx,1),data.outputs(idx
47             ,1)];
47             idx_ = idx;
48         end
49     end

```

#### 7.2.6.4 Control del robot entre los puntos de interés buscados

Finalmente, ya se puede controlar el robot para que se desplace entre los puntos especificados. Los puntos que se han buscado y añadido mediante la pestaña de búsqueda de la interfaz aparecen ahora en la lista que se puede apreciar en la figura (7.18). El usuario podrá hacer que el robot se mueva entre estos puntos con total libertad. Para ello basta con seleccionar uno de la lista y pulsar el botón *ir*. Esto hará que el robot se mueva automáticamente a dicha posición.

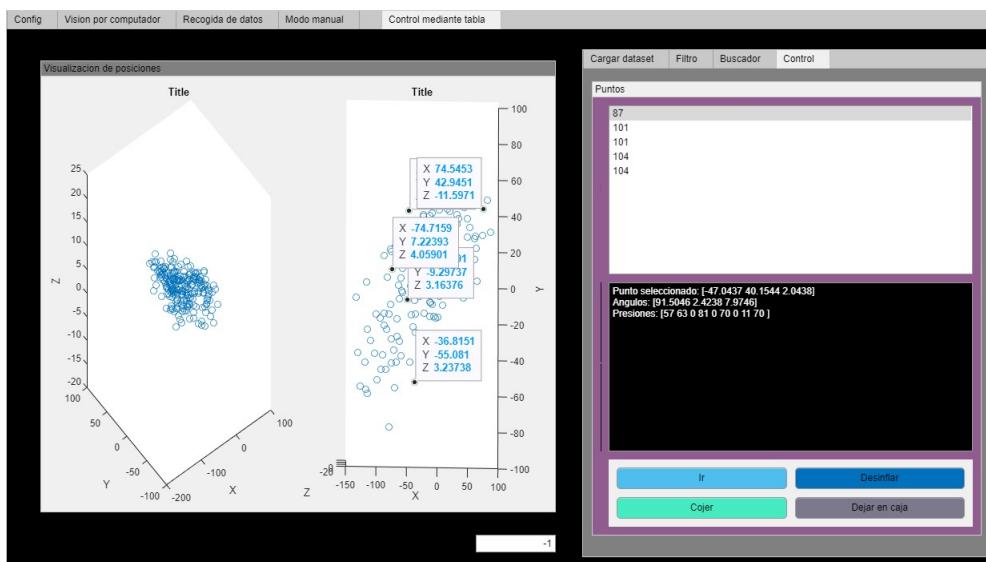


Figura 7.18: Control: Movimiento a las posiciones indicadas. *Fuente: elaboración propia*

De esta manera, la resolución de la cuadrícula del espacio de trabajo sobre la cual se puede controlar el robot aumenta con el número de muestras del dataset, siendo este el principal parámetro que define la precisión de posicionamiento.

## 8. Experimentos y resultados

En este punto del trabajo se ha diseñado y ensamblado un *SoftBot* manipulador, así como las herramientas software necesarias para realizar el control de este. En este capítulo se mostrarán los resultados obtenidos al someter a dicho robot a distintas pruebas. Estos experimentos tienen como principal objetivo demostrar la capacidad de posicionamiento del robot, así como su repetibilidad. Para poder evaluar de una forma más precisa el lugar al que llega el extremo del robot, se le acopla un puntero láser que se dirigirá a los distintos puntos.

Previo al comienzo de los experimentos, es necesario recoger distintos *datasets* entre los cuales buscar los puntos a los que se desea que se mueva el robot. Para esto se realizan varias fases de toma de datos, con distintos rangos de presiones. Un dato interesante es el tiempo medio que se tarda en recoger una muestra de un dataset, para poder estimar la duración de una recogida de datos. Con esto se puede evaluar la eficacia del sistema automático de adquisición de datos.

En la tabla 8.1, pueden observarse los tiempos transcurridos en distintas sesiones de toma de datos, junto al número de muestras tomadas en cada una de ellas. Con esto se obtiene un tiempo medio de muestra de  $t_{mm} = 6,76s$ . Esto muestra la gran ventaja que supone automatizar la recogida frente a realizarse de forma manual.

Dataset:	1	2	3	4	5	6	7
N. Muestras	50	50	50	250	3	100	100
Tiempo (s)	347.66	348.69	356.32	1942.23	19.42	605.41	6.00E+02
Tiempo/Muestra	6.9532	6.9738	7.1264	7.76892	6.473333333	6.0541	5.9988
Tiempo medio por muestra:	6.764079048						
Varianza:	0.399164461						

Cuadro 8.1: Tiempo que tardan en recogerse distintos datasets. *Fuente: elaboración propia*

### 8.0.0.1 Experimentos

Una vez se dispone de estos datos ya es posible realizar los experimentos. Se trabaja con aproximadamente mil muestras, lo que nos da una resolución de posicionamiento en el campo de trabajo de aproximadamente cinco milímetros. Sin embargo, esto solo se cumple en los espacios cubiertos por el robot durante la toma de datos, existiendo zonas vacías. Esto se puede observar en la nube de puntos que se muestra en la interfaz, tal y como se explica en el punto 7.15

Para demostrar la capacidad que tiene el robot de llegar a estos puntos, se han realizado tres experimentos.

El primero consiste en guiar el extremo del robot sobre distintos puntos del plano horizontal, de modo que el extremo del robot quede paralelo al plano del suelo. Para esto, en la fase de selección de los puntos se indican los ángulos de rotación de tal modo que el manipulador del robot esté en la orientación deseada. En la siguiente imagen se pueden ver los puntos seleccionados y como el robot se mueve entre cada uno de ellos

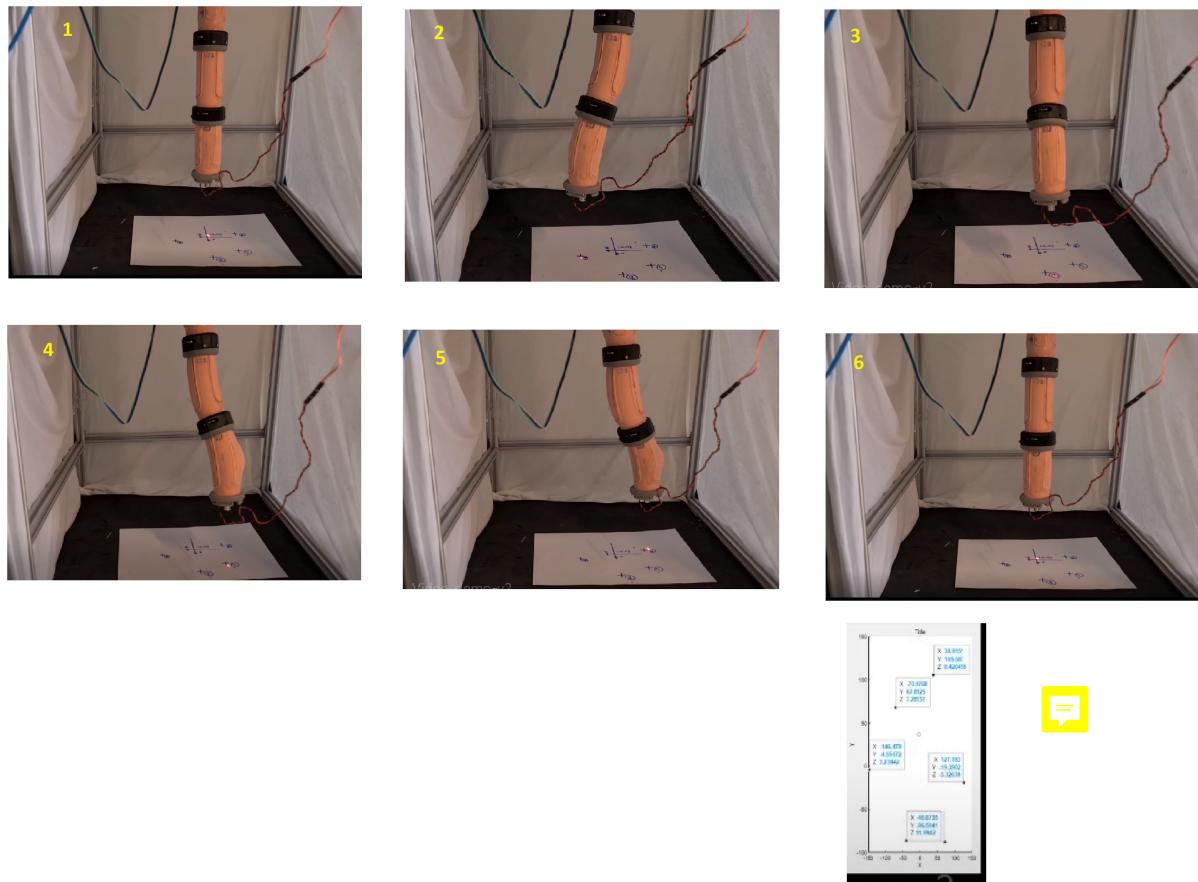


Figura 8.1: Experimentos 1. *Fuente: elaboración propia*

En el segundo experimento se han seleccionado puntos distintos del plano horizontal. Pero en este caso, no se ha impuesto que la cara del robot que tiene el puntero quede horizontal al suelo. Tal y como puede observarse en la figura 8.2, el robot es capaz de apuntar a las distintas dianas de forma satisfactoria. Para medir la repetibilidad, se dibujan en la hoja de las dianas, unos círculos de 2 cm de diámetro. En todas las pruebas que se han realizado se concluye que el robot es capaz de apuntar a las dianas sin salirse de dicho círculo. Sin embargo, a pesar de esto, hay un factor que sí que llama la atención y no puede observarse en las imágenes. Debido a la propia geometría y material del que está constituido el robot, el momento en el que llega a la posición deseada y se detiene, tiende a adquirir un movimiento de oscilación. Este fenómeno ha sido complicado de disminuir, ya que es muy intrínseco al robot.

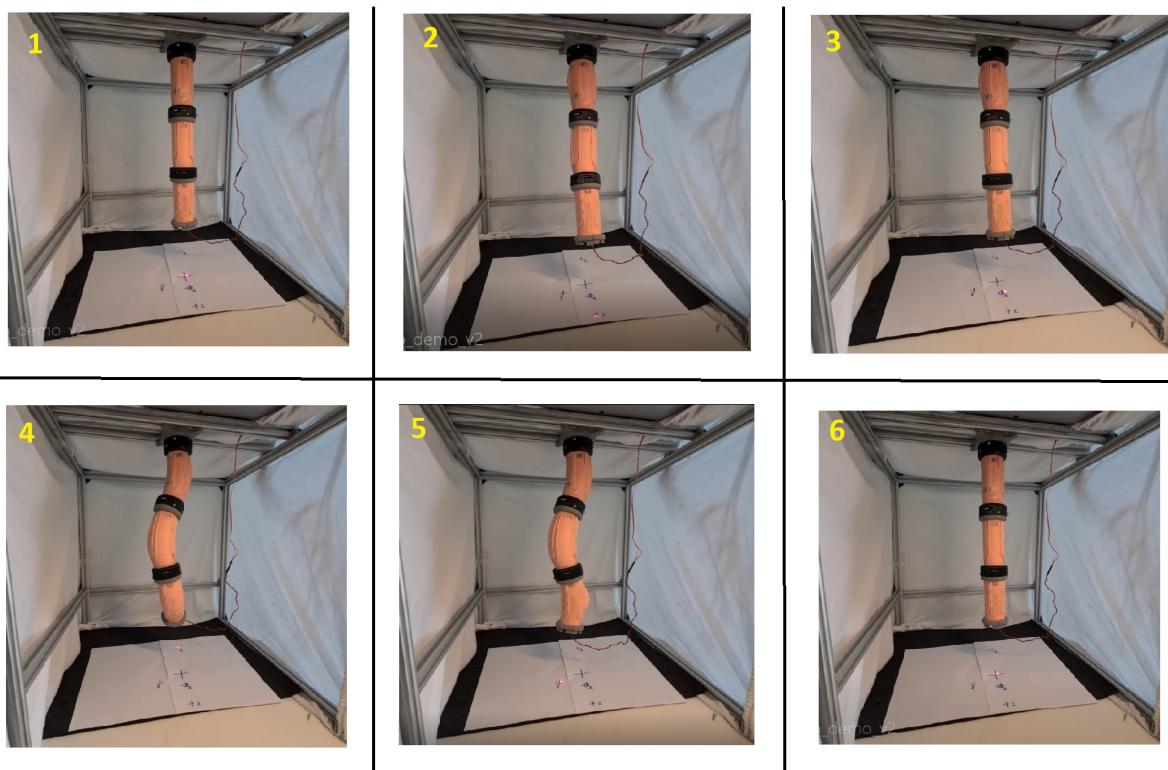


Figura 8.2: Experimentos 2. Fuente: elaboración propia

En el tercer experimento se buscan posiciones que lleven al robot a apuntar a dianas que se encuentren en las paredes laterales en vez del plano del suelo. La finalidad de esto es demostrar la capacidad que tiene el robot de flexionarse gracias a su construcción flexible. En la figura 8.3 se puede apreciar como el *Softbot* llega a los distintos puntos.

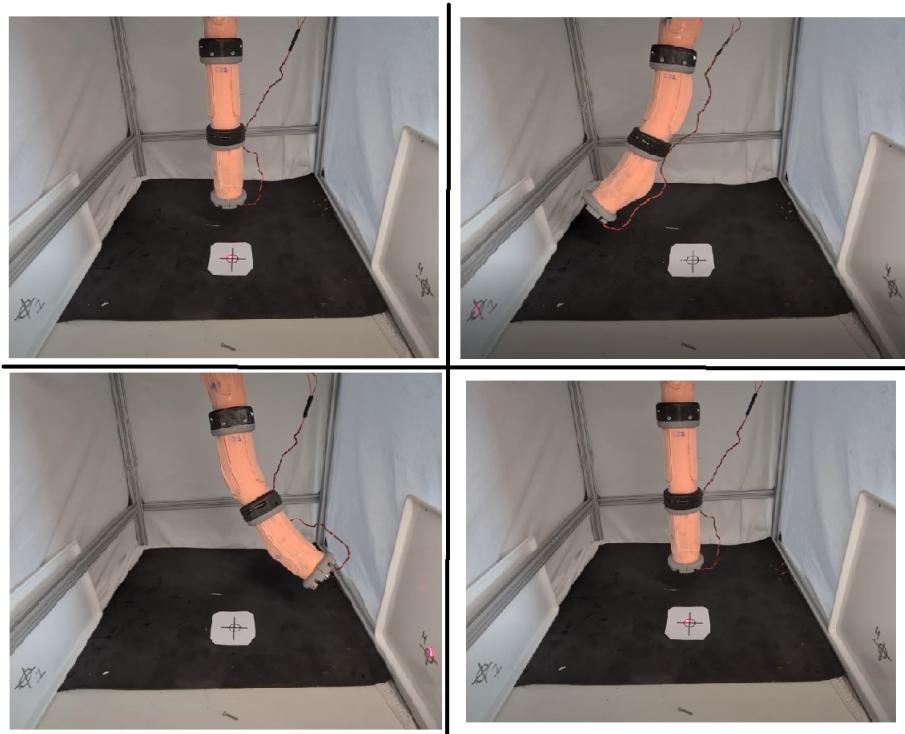


Figura 8.3: Experimentos 3. *Fuente: elaboración propia*

Todo este proceso de toma de datos, experimentos y demostración se puede encontrar en el siguiente video: Video demostración

#### 8.0.0.2 Resultados

No sé muy bien cómo organizar esto



## 9. Sostenibilidad e impacto social

Durante el desarrollo de un proyecto en la actualidad es necesario tener una visión global del impacto social y medioambiental que este tendrá. De esta manera, en el momento que se toman decisiones sobre algún aspecto de este, aparte de atender a los criterios técnicos relacionados directamente con el proyecto, se atiende también a criterios relacionados con el impacto que este tendrá en la sociedad y en el medioambiente.

Con respecto al impacto ambiental, en proyectos como este, que tratan sobre el desarrollo de un *producto* tangible, no es solo necesario tener en cuenta el impacto que el proyecto tiene en el momento de su uso, sino durante todo su ciclo de vida. A esto se le denomina análisis del ciclo de vida, que se diferencia de un estudio de impacto ambiental, por ejemplo, en el espacio de tiempo que se abarca y en que se tienen en cuenta todos los flujos energéticos y de materia de todas las fases del proyecto.

En el contexto de este proyecto, a pesar de que no se ha realizado un análisis de ciclo de vida completo, debido al alcance de este, sí que se han analizado ciertos aspectos medioambientales de por ejemplo el proceso de fabricación o el momento en el que el robot se vuelve inservible.

Con respecto al proceso de fabricación, principalmente se ha tenido en cuenta el PLA del que están hechos los distintos moldes y la cera que se utiliza durante el conformado. Debido a la necesidad del proyecto de realizar numerosas iteraciones con distintos diseños, han sido necesarios muchos moldes distintos. Estos están realizados de PLA, que es un material que presenta la ventaja de ser fácilmente reciclable al tratarse de un polímero termoplástico.

A parte de esto, se ha prestado atención a que la mayor cantidad posible de componentes de los moldes sea reutilizable. Para esto se ha intentado que estos sean lo más modulares posible.

Por otra parte, en el proceso de fabricación, se utiliza cera de parafina para los machos solubles. Para disminuir el consumo de este material, en el momento que esta se funde y se extrae del modelo, se ha recogido y se ha reutilizado para otros moldes.

Por otra parte, hay un factor muy importante presente en este proyecto difícil de controlar. Este es el uso de la silicona, tanto de platino como de estaño. Ambas se tratan de dos materiales poco reciclables y no biodegradables. Por lo que se ha intentado minimizar el uso de esta por cada segmento del robot. Sin embargo, a pesar de tratarse de materiales no biodegradables, se consideran materiales con menos impacto que los plásticos tradicionales. Esto se debe a que son totalmente inertes, por lo que no producen toxinas ni se descomponen en otros elementos (no se presenta el problema de los *microplásticos*).

De todos modos, en el momento que se compara el impacto de un *SoftBot* manipulador como el resultante de este trabajo, con un robot manipulador tradicional de las mismas características, saltan a la vista muchos aspectos relacionados con el impacto. El robot tradicional, durante su proceso de fabricación y debido a sus características de actuación, utiliza una gran cantidad de materiales y componentes. Desde un gran número de motores, la electrónica necesaria para el control de estos, ..., hasta todos los procesos de conformado de los elementos rígidos de este. Teniendo en cuenta esto, al menos de una forma cualitativa, se puede prever que el impacto medioambiental de un robot blando como el que se ha desarrollado, en el momento en el que se perfeccione, puede tener un impacto medioambiental mucho menor que un robot tradicional.

Además, se ha intentado que el diseño del robot sea lo más modular posible. Esto permite modificar las características de este, añadiendo o quitando algún módulo, en vez de fabricar el

robot completo de nuevo. Lo mismo aplica al banco de actuación desarrollado, que se puede aprovechar fácilmente para otros proyectos futuros que se desarrolle.

Todo esto se refiere a la parte del impacto medioambiental. Cuando se tratan robots como el que se ha ideado en este proyecto, también se pueden matizar varias ideas con respecto al impacto social.

Debido a su construcción mediante elementos blandos, estos robots suponen una sustancial diferencia con respecto a los robots industriales tradicionales. Gracias a que en caso de colisión no provocan ningún daño, muy compatibles con tareas en las que tengan que convivir con seres humanos. Solo se pueden comparar con los modernos cobots. Estos también permiten la cooperación con el ser humano. Sin embargo, estos siguen siendo rígidos, confiando los aspectos relacionados con la seguridad principalmente el control y la programación. Teniendo esto en cuenta, se considera que los *softbots* siguen siendo superiores a estos en lo que a seguridad se refiere, al residir está directamente en su esencia flexible y no a programaciones susceptibles a fallar si no se realizan correctamente.

Finalmente, se quiere mencionar, que gracias a su proceso de fabricación y los materiales que se emplean, en el momento en el que esta clase de robots se perfeccionen, pueden convertirse en una alternativa muy económica a otros tipos de robots. Esto puede llevar a una mayor presencia de robots en muchos ámbitos al existir opciones más accesibles, llevando a que una mayor cantidad de personas puedan aprovecharse de las ventajas de estos.

## 10. Planificación y presupuesto

### 10.1. Planificación

En este capítulo se detalla la planificación adoptada en el transcurso de este proyecto. Este se ha realizado mediante una estructura de descomposición de proyecto (EDP) y un diagrama de Gantt. También se tratarán los aspectos económicos y el presupuesto.

La estructura de descomposición de proyecto cumple la función de segmentar el proyecto completo en tareas más sencillas para facilitar la planificación de estas en el tiempo. Para la planificación temporal del trabajo se ha utilizado un diagrama de Gantt que se ha realizado mediante la herramienta *Microsoft Project*.

### 10.1.1. Estructura de descomposición de proyecto

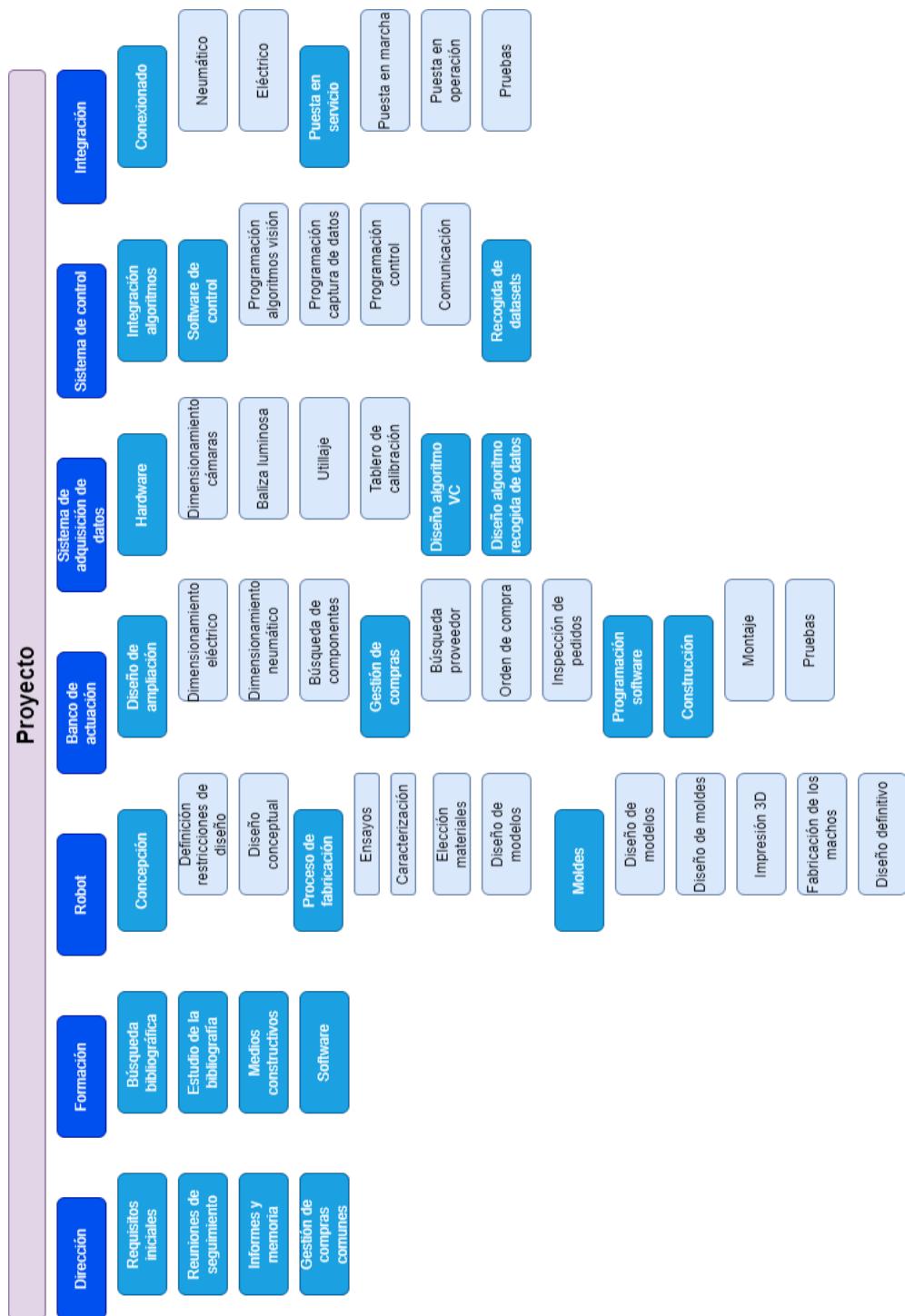


Figura 10.1: Estructura de descomposición de proyecto. *Fuente: elaboración propia*

### 10.1.2. Diagrama de Gantt

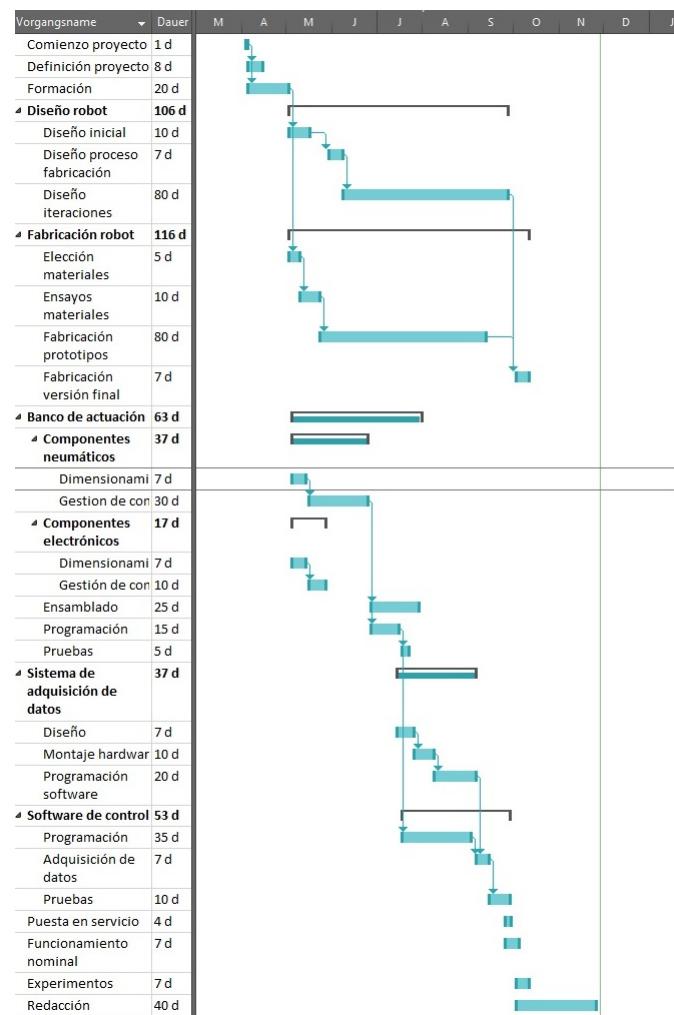


Figura 10.2: Diagrama de Gantt. Fuente: elaboración propia

## 10.2. Presupuesto

En la siguiente tabla se recoge la descomposición del coste de los distintos elementos utilizados en este proyecto. Se han organizado en distintas partidas atendiendo a la finalidad de estos.

Partida	Concepto	Unidades	Precio unidad (€)	Importe
1	Bibliografía			
	Lectura de documentos	40	15	600
2	Robot			
2.1	Silicona			
	Silicona TinSil 80-15	2	42.96	85.92
	Silicona Platsil FS-10 kit 900 gr.	2	58.69	117.38
	Silicona EASYPLAT 00-30 kit 2 kg.	1	58.08	58.08
2.2	Impresión 3D			
	PLA BQ 1.75 mm 1kg	2	20	40
	Smartfil PVA 1.75 mm 750 gr.	1	45.46	45.46
	Horas impresión	90	0.1	9
2.3	Fabricación			
	Cera de parafina	5	2	10
	Guantes latex (caja)	1	12.99	12.99
	Tornillería	1	21.99	21.99
	Mano de obra	90	15	1350
3	Banco de actuación			
3.1	Componentes neumáticos			
	Compresor y manguera neumática	1	144.99	144.99
	Tubo neumático TS0425BU-20	1	15.74	15.74
	Electroválvula 2/2 VDW20BZ1D	4	16.25	65
	Racor M5 a 4mm KQ2L04-M5A	20	2.13	42.6
	Placa base SS3Y1-S41-06-M5-Q	1	40.47	40.47
	PLACA CIEGA SY114	3	1.37	4.11
3.2	Componentes electrónicos			
	Arduino due	1	49.78	49.78
	Placa de mosfets IRF540	5	9.21	46.05
	Cableado	1	5	5
3.3	Montaje			
	Mano de obra	45	15	675
4	Software			
	Mano de obra	90	15	1350
5	Ensayos y recogida de datos			
	Mano de obra	40	15	600
6	Memoria final			
	Redacción	85	15	1275
			<b>TOTAL</b>	<b>6664.56</b>

## 11. Conclusiones y trabajos futuros

Este trabajo ha supuesto la creación de un *SoftBot* manipulador de nueve grados de libertad actuado neumáticamente. Se han abordado las fases de diseño conceptual, diseño básico, fabricación y control. Además, se ha desarrollado un sistema de adquisición de datos basado en visión por computador, se han realizado tareas de diseño neumático para controlar el robot y de desarrollo software.

Gracias a esto se han mejorado numerosas competencias como la programación en Matlab y c++, diseño CAD y se han adquirido otras totalmente nuevas, como el diseño de procesos de fabricación con polímeros elastómeros.

En este capítulo se recoge una síntesis de los distintos desarrollos y aprendizajes realizados en este trabajo. También se hace una propuesta de posibles líneas futuras para continuar este proyecto.

### 11.1. Conclusiones

Este trabajo ha servido para poner de manifiesto muchos aspectos relacionados con el campo de los robots flexibles manipuladores. Se ha comprobado que la manera de trabajar con estos difiere completamente con la de los robots tradicionales. Debido a sus características flexibles y los métodos de actuación que utilizan, no es posible seguir los mismos pasos que se seguirían, en caso de desarrollar un robot prototipo basado en cadena cinemática rígida. A todo esto, se suma el relativamente breve tiempo que ha tenido este campo de la robótica blanda para desarrollarse, existiendo poca documentación al respecto.

En este proyecto se ha comprobado que uno de los aspectos más críticos a la hora de desarrollar un prototipo de este tipo, es encontrar los materiales y un proceso de fabricación adecuado. Especialmente, en un proyecto como este, en él se perseguía el objetivo de desarrollar un robot blando, de nueve grados de libertad, el diseño ha supuesto un enorme reto debido a distintos factores.

El primero de ellos, está relacionado con una de las características más fundamentales que se buscaban en el robot fruto de este trabajo: la capacidad de disponer de segmentos que se puedan actuar en tres grados de libertad. Esto ha obligado a que la geometría interior de los segmentos tenga una geometría compleja, que no se puede conseguir mediante muchos de los métodos de fabricación actualmente utilizados en este campo. A esto se suman diversos factores constructivos como la búsqueda de un diseño modular o que los tubos neumáticos queden embebidos en el propio cuerpo.

Otro factor está relacionado con los materiales. Tal y como se ha explicado en el capítulo correspondiente, se ha observado la sensibilidad que tienen las siliconas a inhibir su correcto curado en presencia de diversos materiales, lo que limita aún más la búsqueda de un proceso de fabricación adecuado.

Pese a estas dificultades, y después de diversas iteraciones, se ha desarrollado un prototipo funcional que satisface los requerimientos de diseño establecidos al principio del proyecto. Además, se ha realizado mediante un proceso de fabricación acotado en el presupuesto, y cuyos resultados pueden obtenerse para la fabricación de softbots en futuros proyectos.

A pesar de que el diseño y la fabricación de las distintas iteraciones del robot ha sido una de

las partes más importantes de este proyecto y que más tiempo ha consumido, no es la única dificultad que se ha enfrentado. Otro de los aspectos más importantes ha sido la forma en como los distintos grados son controlados. Para esto, se ha modificado un banco de actuación acorde a las necesidades del proyecto. Estas modificaciones han incluido la sustitución de relés por mosfets, lo que se concluye que ha sido un gran paso hacia adelante gracias a su menor tiempo de respuesta y menor consumo eléctrico.

Otra dificultad que emerge de las características intrínsecas de este tipo de robots es la complejidad a la hora realizar un control de posición. Tal y como se ha explicado a lo largo de esta memoria, en este proyecto se ha apostado por técnicas de control basadas en datos empíricos. Para esto, se ha desarrollado un sistema de adquisición de datos, basado en visión por computador, que permite obtener de forma sencilla datos experimentales que pueden utilizarse para distintos algoritmos que realicen el control de posición.

Uno de los aspectos en los que se habría incidido más, pero de alguna manera escapaban el alcance del proyecto, es en el uso de estos datos para entrenar una red neuronal que se encargue del control de posicionamiento. Sin embargo, sí que se ha conseguido realizar un control de posicionamiento utilizando los datos captados mediante el sistema de visión, poniendo de manifiesto las capacidades que tiene el robot de fruto de este trabajo y las posibilidades de futuros desarrollos sobre este. Este último punto se motiva debido a que todos lo que se ha realizado en este proyecto se ha integrado en un software fácilmente adaptable y sencillo de utilizar.

Con todo esto se concluye, que a pesar de que habría algunas líneas que se hubieran desarrollado más en otras circunstancias, el robot que se ha construido, junto a la programación que lo acompaña, satisface los requisitos establecidos al inicio del proyecto.

## 11.2. Líneas futuras

A pesar de que se considera que se han cumplido los objetivos propuestos del trabajo, no quita que haya lugar para posibles trabajos futuros. Se detallan a continuación las que se han considerado más importantes:

### ■ Relacionados con el robot

Mejoras que se relacionan con el robot en sí mismo

#### • Mejora en el diseño:

A lo largo del trabajo realizado con el prototipo se han encontrado distintos aspectos del diseño que se pueden mejorar. Estos están relacionados con las zonas de rotura de los segmentos. A medida que se ha utilizado el robot, se han ido detectando zonas de los segmentos, por los que tienden a romperse. Estas son, principalmente, las zonas pensadas para la incorporación de los sensores y las paredes interiores que alojan los tubos neumáticos. Por ello se propone como posible mejora, simular el diseño y reforzar estas zonas débiles.

#### • Incorporación de sensores:

Tal y como se explica en los ANEXOS y como se puede observar en el diseño del robot, este está preparado para la incorporación de unos sensores resistivos. Debido al alcance y cuestiones temporales, no ha sido viable incorporarlos en el contexto de este proyecto, pero se propone como una posible mejora. Estos permitirían un control en lazo cerrado.

- **Adición de una herramienta:**

Para los experimentos que se han realizado, se ha incorporado en el extremo del robot un puntero láser con el que el robot es capaz de apuntar a distintas marcas. Sin embargo, sería interesante que este dispusiera, de, por ejemplo, una pinza para demostrar de una forma más eficaz las posibilidades de manipulación.

- **Relacionadas con el banco de actuación**

Mejoras que se ha detectado que pueden realizarse en el banco de actuación:

- **Incorporación de sensores de presión:**

Un factor que ha limitado en gran medida este proyecto es no poder controlar directamente la presión de las distintas vejigas, habiéndose hecho por este motivo un control basado en tiempos. Sería muy interesante, que cada línea neumática tuviera un sensor de presión analógico. De esta forma, empleando técnicas de control muy sencillas, sería posible controlar en lazo cerrado la presión de cada línea.

- **Diseño de una PCB para los mosfets:**

Debido a la limitación de alcance del proyecto, se han adquirido los Mosfets que realizan la función de interfaz de potencia del banco de actuación, en unos módulos comerciales. Se propone, como propuesta, diseñar una pcb que incluya todos los elementos necesarios del banco neumático. Esta podría incorporar la interfaz de potencia, la electrónica necesaria para los sensores de presión mencionados en el punto anterior, o incluso el microcontrolador.

- **Relacionadas con el control**

- **Control en cadena cerrada:**

Una vez incorporada la sensorización necesaria, un control en cadena cerrada podría disminuir significativamente el error de posicionamiento. Además, quizás, mediante un control adecuado, pueda incluso disminuirse la oscilación que realiza el robot al llegar a un nuevo punto.

- **Aplicación de aprendizaje automático:**

Sin duda esta es la mejora que más importante se considera. A lo largo del proyecto se han investigado las posibilidades que el aprendizaje automático puede ofrecer a la hora de realizar el control cinemático de un robot de estas características.

## Referencias

- [1] A. Anwar. «What are Intrinsic and Extrinsic Camera Parameters in Computer Vision?» (), dirección: <https://towardsdatascience.com/what-are-intrinsic-and-extrinsic-camera-parameters-in-computer-vision-7071b72fb8ec>.
- [2] G. Chen, X. Yang, X. Zhang y H. Hu, «Water hydraulic soft actuators for underwater autonomous robotic systems,» *Applied Ocean Research*, vol. 109, pág. 102551, abr. de 2021. DOI: 10.1016/j.apor.2021.102551.
- [3] Á. S. Curiel, «Generacion de patrones de marcha para soft robot caminante,» TFG, ETSII UPM, 2016.
- [4] T. Dayyoub y A. Maksimkin, «Electroactive Polymer-Based Composites for Artificial Muscle-like Actuators: A Review,» *Nanomaterials*, vol. 12, pág. 2272, jul. de 2022. DOI: 10.3390/nano12132272.
- [5] M. Giorelli, F. Renda, M. Calisti et al., «Learning the inverse kinetics of an octopus-like manipulator in three-dimensional space,» *Bioinspiration biomimetics*, vol. 10, pág. 035006, jun. de 2015. DOI: 10.1088/1748-3190/10/3/035006.
- [6] M. A. Hortelano, «Desarrollo de herramientas de agarre basadas en robotica blanda para robots manipuladores colaborativos,» TFG, ETSII UPM, 2020.
- [7] T. Kalinsky, D. Drotman, B. Shih y E. Aronoof-Spencer, «Differential pressure control of 3D printed soft fluidic actuators,» en *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017.
- [8] R. K. Katzschatmann, 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.
- [9] W.-B. Li, W.-M. Zhang, Q.-H. Gao et al., «Electrostatic field induced coupling actuation mechanism for dielectric elastomer actuators,» *Extreme Mechanics Letters*, vol. 35, pág. 100638, 2020, ISSN: 2352-4316. DOI: <https://doi.org/10.1016/j.eml.2020.100638>. dirección: <https://www.sciencedirect.com/science/article/pii/S2352431620300134>.
- [10] A. Masiero y A. Cenedese, «Reconstruction error in a motion capture system,» *CoRR*, vol. abs/1203.3230, 2012. arXiv: 1203.3230. dirección: <http://arxiv.org/abs/1203.3230>.
- [11] M. G. Merino, «Robot continuo actuado con SMA,» TFG, ETSII UPM, 2020.
- [12] F. A. Mohd Ghazali, C. K. Mah, A. AbuZaiter, P. S. Chee y M. S. Mohamed Ali, «Soft dielectric elastomer actuator micropump,» *Sensors and Actuators A: Physical*, vol. 263, págs. 276-284, 2017, ISSN: 0924-4247. DOI: <https://doi.org/10.1016/j.sna.2017.06.018>. dirección: <https://www.sciencedirect.com/science/article/pii/S0924424716311967>.
- [13] V. B. Moreno, «Desarollo y control de un soft - bot caminante con accionamiento neumatico,» TFM, ETSII UPM, 2015.

- [14] T. Ranzani, M. Cianchetti, G. Gerboni et al., «A modular soft manipulator with variable stiffness,» sep. de 2013.
- [15] F. A. S.A.U. «BionicSoftArm.» (), dirección: [https://www.festo.com/es/es/e/sobre-festo/investigacion-y-desarrollo/bionic-learning-network/bionicsoftarm-id\\_68209F](https://www.festo.com/es/es/e/sobre-festo/investigacion-y-desarrollo/bionic-learning-network/bionicsoftarm-id_68209F).
- [16] A. N. Semochkin, «A device for producing artificial muscles from nylon fishing line with a heater wire,» en *2016 IEEE International Symposium on Assembly and Manufacturing (ISAM)*, 2016, págs. 26-30. DOI: 10.1109/ISAM.2016.7750715.
- [17] J. Sun, B. Tighe, Y. Liu y J. Zhao, «Twisted-and-Coiled Actuators with Free Strokes Enable Soft Robots with Programmable Motions,» *soro*, 2021.
- [18] J. Sun y J. Zhao, «Integrated Actuation and Self-Sensing for Twisted-and-Coiled Actuators with Applications to Innervated Soft Robots,» en *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, págs. 8795-8800. DOI: 10.1109/IROS45743.2020.9340647.
- [19] M. Sun, C. Tian, L. Mao et al., «Reconfigurable Magnetic Slime Robot: Deformation, Adaptability, and Multifunction,» *Advanced Functional Materials*, vol. 32, n.º 26, pág. 2 112 508, 2022. DOI: <https://doi.org/10.1002/adfm.202112508>. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/adfm.202112508>. dirección: <https://onlinelibrary.wiley.com/doi/abs/10.1002/adfm.202112508>.
- [20] S. Terrile, «Soft Robotics: Applications, design and control,» Thesis, ETSII UPM, 2021.
- [21] S. R. Toolkit. «Pneumatic Artificial Muscles.» (), dirección: <https://softroboticstoolkit.com/book/pneumatic-artificial-muscles>.
- [22] Y. Toshimitsu, K. Wong, T. Buchner y R. Katzschmann, *SoPrA: Fabrication Dynamical Modeling of a Scalable Soft Continuum Robotic Arm with Integrated Proprioceptive Sensing*, mar. de 2021.
- [23] D. Trivedi y C. Rahn, «Shape Sensing for Soft Robotic Manipulators,» vol. 7, ene. de 2009. DOI: 10.1115/DETC2009-87598.
- [24] R. L. Truby, L. Chin y D. Rus, «A Recipe for Electrically-Driven Soft Robots via 3D Printed Handed Shearing Auxetics,» *IEEE Robotics and Automation Letters*, vol. 6, n.º 2, págs. 795-802, 2021. DOI: 10.1109/LRA.2021.3052422.
- [25] R. L. Truby, C. D. Santina y D. Rus, «Distributed Proprioception of 3D Configuration in Soft, Sensorized Robots via Deep Learning,» *IEEE Robotics and Automation Letters*, vol. 5, n.º 2, págs. 3299-3306, 2020. DOI: 10.1109/LRA.2020.2976320.
- [26] V. Wall, G. Zöller y O. Brock, «A method for sensorizing soft actuators and its application to the RBO hand 2,» en *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, págs. 4965-4970. DOI: 10.1109/ICRA.2017.7989577.

- [27] H. Wang, R. Zhang, W. Chen, X. Wang y R. Pfeifer, «A cable-driven soft robot surgical system for cardiothoracic endoscopic surgery: preclinical tests in animals,» *Surgical endoscopy*, vol. 31, ago. de 2017. DOI: 10.1007/s00464-016-5340-9.
- [28] 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, 2021, ISSN: 2296-9144. DOI: 10.3389/frobt.2021.730330. dirección: <https://www.frontiersin.org/articles/10.3389/frobt.2021.730330>.
- [29] Y.-S. Yang, «Measurement of Dynamic Responses from Large Structural Tests by Analyzing Non-Synchronized Videos,» *Sensors*, vol. 19, n.º 16, 2019, ISSN: 1424-8220. DOI: 10.3390/s19163520. dirección: <https://www.mdpi.com/1424-8220/19/16/3520>.

## ANEXOS

### C. Primer anexo: Caracterización de sensores elásticos

A pesar de no haberse incluido como tal dentro del desarrollo del robot de este trabajo, se considera interesante incluir en esta memoria los resultados que se han obtenido con sensores elásticos de cara a futuros trabajos.

Estos sensores tenían el objetivo de permitir un control en cadena cerrada del robot. Estos consisten en filamentos de goma a los que se les ha incorporado carbono durante su conformado. De esta manera, al estirarse, disminuye su sección cambiando la resistencia del filamento. De este modo se pueden tratar estos sensores como resistencias variables. Además, al estar basados en una goma elástica, son óptimos para embeberse en la estructura de un robot de estas características.

Al comienzo de este trabajo se han realizado pruebas con el modelo que proporciona el fabricante Adafruit. Para medir la señal, se conecta dicho sensor en configuración de divisor de tensión y se mide la caída de tensión en la resistencia fija.

Las primeras pruebas realizadas con el osciloscopio pusieron de manifiesto que este tipo de sensores presentan un acusado fenómeno de histéresis. Con esto nos referimos, que una vez estirado el sensor, al soltarlo, tarda un cierto tiempo en recuperar su forma (y resistencia eléctrica) inicial.

Para poder solventar este problema se ha ideado un sistema de medición diferencial. En esta configuración, en vez de obtener la señal de un solo sensor, se obtiene procesando la señal de varios sensores. Este procesamiento matemático se realiza vía software, obteniendo previamente la señal de cada sensor de forma separada utilizando una placa *Arduino Mega* como interfaz analógico - digital.

Para demostrar la viabilidad de esta solución se ha realizado un prototipo físico con el que se han realizado varios experimentos. Este prototipo consiste en un modelo de silicona, cuyo objetivo es fletarse en un plano. Dentro de este, están embebidos dos sensores elásticos formando dos lazos, de forma que uno pueda cancelar la señal del otro. En la figura C.1, puede observarse el modelo de silicona que se ha fabricado.



Figura C.1: Anexo 1: Modelo realizado para la caracterización de sensores elásticos. *Fuente: elaboración propia*

Una vez terminado el proceso de fabricación del modelo de silicona se procede al conexionado. Para ello, se utiliza un divisor de tensión, con una resistencia fija de un megaohmio. Además, se calcula un sencillo filtro RC con un condensador de cien microfaradios y una resistencia de doscientos ohmios. Todo el sistema de medición se alimenta con la tensión de 5 V del Arduino. En cuanto a procesamiento, se realiza una resta ponderada entre las señales que se obtienen de los sensores, para que en el momento en el que el modelo este sin deformar, la señal sea cero. También se acota la señal entre ciertos valores para eliminar el ruido. A continuación, en la figura C.2, puede observarse el resultado obtenido.

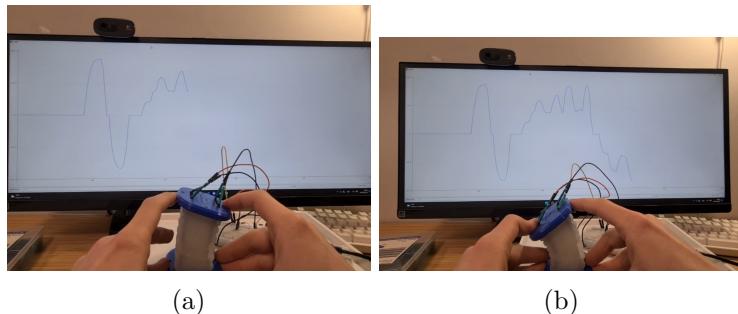


Figura C.2: Anexo 1: Resultados medición diferencial. *Fuente: elaboración propia*

Después de realizar algunos experimentos más se concluye que la disposición diferencial de los sensores es óptima, y se tiene en cuenta a la hora de diseñar los segmentos del robot.

**D. Segundo anexo**

Planos?