



Escuela Superior de Ciencias Experimentales y Tecnología

**GRADO EN INGENIERÍA
DE TECNOLOGÍAS INDUSTRIALES**

Trabajo de Fin de Grado

**DISEÑO Y FABRICACIÓN DE HERRAMIENTAS PARA
UN BRAZO ROBÓTICO ABB IRB 120**

Marcos Vlietman Pelayo

Director: Felipe Machado Sánchez

Curso académico 2019/2020



Grado en Ingeniería de Tecnologías Industriales

Trabajo de Fin de Grado

El presente trabajo, titulado **DISEÑO Y FABRICACIÓN DE HERRAMIENTAS PARA UN BRAZO ROBÓTICO ABB IRB 120**, constituye la memoria correspondiente a la asignatura Trabajo Fin de Grado que presenta **D. Marcos Vlietman Pelayo** como parte de su formación para aspirar al título de Graduado en Ingeniería de Tecnologías Industriales.

Este trabajo ha sido realizado en la **Universidad Rey Juan Carlos**, en el **Departamento de Matemática Aplicada, Ciencia e Ingeniería de los Materiales y Tecnología Electrónica**, bajo la dirección de **Felipe Machado Sánchez**.

Móstoles, 8 de octubre de 2019

AGRADECIMIENTOS

A Diego Martín y José Carlos Hurtado que, sin tener que hacerlo, me han ayudado a la hora de entender el funcionamiento del robot y con la configuración de las conexiones de este.

A mi tutor, que me ha ayudado y guiado durante este proyecto.

A mis amigos, que han celebrado mis éxitos y han sufrido conmigo.

A mis padres y mis abuelos, que me han apoyado y seguido en las decisiones que he tomado a lo largo de esta etapa.

A mi abuela, por todo lo que ha hecho por mí, gracias.

ÍNDICE

ÍNDICE DE FIGURAS.	v
TABLA DE ABREVIATURAS.	vii
Resumen	1
1. INTRODUCCIÓN	2
1.1. Industria 4.0	2
1.2. Impresión 3D	4
1.3. Filosofía “Open Source”	6
1.4. Brazo robótico.	7
1.5. Microcontroladores.	9
1.6. Estado del arte: Herramientas para el brazo robótico ABB IRB120 basadas en open source.	10
1.7. Estructura de la memoria.	11
2. OBJETIVOS	12
3. Solución Técnica	13
3.1. Herramientas.	15
3.1.1. Porta-Rotulador.	15
3.2. Actuadores.	21
3.2.1. Motor paso a paso.	22
3.2.2. Servomotor.	24
3.3. Pinzas.	25
3.3.1. Pinza de motor paso a paso.	26
3.3.2. Pinza de servomotor.	32
3.4. Electrónica de potencia.	37
3.4.1. Electrónica de potencia del motor paso a paso.	37
3.3.2. Electrónica de potencia para la conexión con el Robot IRB120.	39
3.5. Programación en Arduino.	41
3.5.1. Programación del motor paso a paso.	41
3.5.2. Programación del servomotor.	42
3.6. Montaje final en el robot.	43
3.6.1. Simulación en RobotStudio.	43
3.6.2. Conexiones con Robot ABB IRB 120.	45
3.7. Materiales.	48
3.7.1. Software.	48
3.7.2. Hardware.	53
3.8. Metodología de trabajo.	63

3.9.	Comparación económica.....	65
4.	Conclusiones.....	69
5.	Bibliografía.....	71
6.	Anexo.....	75
6.1.	Anexo 1. Programación en Arduino.....	75
6.1.1.	Código motor paso a paso.....	75
6.1.2.	Código servomotor.....	79
6.2.	Anexo 2. Programación en RobotStudio.....	81
6.2.1.	Trayectoria del robot.....	81
6.2.2.	Programa de funcionamiento del robot.....	82

ÍNDICE DE FIGURAS.

Figura 1. Reglas de diseño DFMA.....	6
Figura 2. Morfología del brazo humano frente al brazo robótico. [Fuente: https://es.slideshare.net/leogeekec/morfologia-de-un-robot]	8
Figura 3. Esquema de componentes de un microcontrolador. [Fuente: Microbóticare]	9
Figura 4. "Porta-Rotuladores".....	13
Figura 5. Pinza motor paso a paso.....	14
Figura 6. Pinza servomotor	14
Figura 7. Estación de pruebas en RobotStudio.....	15
Figura 8. Porta-Rotulador 1.....	16
Figura 9. Robot ABB IRB con herramienta industrial	17
Figura 10. Porta-Rotulador 2.....	17
Figura 11. Vista axonométrica y planta de la tapa.	18
Figura 12. Posiciones de las herramientas durante la impresión.....	21
Figura 13. Secuencia de motores paso a paso bipolares.....	23
Figura 14. Motor paso a paso	24
Figura 15. Servomotor. [Fuente: https://www.monografias.com/trabajos60/servo-motores/servo-motores.shtml].....	25
Figura 16. Mismo sistema de agarre en ambas pinzas.	26
Figura 17. Pinza prototipo.....	26
Figura 18. Dimensiones de la deslizadera	27
Figura 19. Modelado del enganche. Vista trasera y frontal.....	28
Figura 20. Unión del enganche con la caja motor.....	28
Figura 21. Diseño final del enganche.....	29
Figura 22. Vista axonométrica frontal y trasera de la caja del motor.....	30
Figura 23. Dimensiones del dedo.	31
Figura 24. Sistema de agarre completo.	31
Figura 25. Vista frontal y trasera de la pieza final	31
Figura 26. Pinza diseñada por Obijuan	33
Figura 27. Modelo alámbrico.....	33
Figura 28. Base de la pinza modificada.	34
Figura 29. Vista de la base de la pinza completa.	34
Figura 30. Engranaje con hendidura.	35
Figura 31. Pieza de unión.....	35
Figura 32. Pieza final junto a servomotor.	36
Figura 33. Ensamblaje final de la pinza	36
Figura 34. Pinza con la solución para mejorar la fracción entre las piezas.....	37
Figura 35. Esquema de las conexiones del CNC y el controlador	37
Figura 36. Esquema del bobinado del motor paso a paso	38
Figura 37. Incompatibilidad de voltajes.....	39
Figura 38. Circuito esquemático.	40
Figura 39. Disposición de los componentes electrónicos.....	40
Figura 40. Modulación de los pulsos para las posiciones del servomotor SG90 de 0°, 90° y 180°.	43
Figura 41. Estación con Robot ABB120 y pinza servomotor.	44
Figura 42. Puertos de entrada y de salida del IRC5	45
Figura 43. Esquema de configuración de los puertos XS14 y XS15.....	46
Figura 44. Esquema de configuración del puerto XS16.....	47
Figura 45. Controlador IRC5 Compact, donde se señala la situación de la placa.....	47
Figura 46. Estantería porta-placa	47
Figura 47. Interfaz de DiYLC	49
Figura 48. Entorno de dibujo de circuitos esquemáticos de KiCAD.	50

Figura 49. Interfaz del perfil del autor del TFG en GitHub.	52
Figura 50. Placa Arduino UNO.	54
Figura 51. CNC shield.	55
Figura 52. Esquema de los pines requeridos del Arduino UNO respecto de la CNC.	55
Figura 53. Controlador Pololu A4988 con disipador de calor.	56
Figura 54. Cableado de conexión con el motor paso a paso y con la placa CNC shield.	56
Figura 55. Extrusor y cama de iDeator12.	58
Figura 56. Fallos en la fabricación de las piezas.	59
Figura 57. Optoacoplador.	62
Figura 58. Diagrama de Gantt del proyecto.	64
Figura 59. Programa en Arduino de la pinza con motor paso a paso	77
Figura 60. Programa en Arduino de la pinza con motor paso a paso (Continuación)	78
Figura 61. Programa en Arduino de la pinza con motor paso a paso (Continuación 2)	78
Figura 62. Código en Arduino de pinza con servomotor.	80
Figura 63. Código en Arduino de pinza con servomotor (Continuación)	81
Figura 64. Programa en RAPID de la trayectoria	82

TABLA DE ABREVIATURAS

<u>ABREVIATURAS</u>	<u>SIGNIFICADO</u>
TFG	Trabajo de fin de grado
RFID	Identificación por radio frecuencia – (Radio Frequency Identification)
API	Interfaz de programación de aplicaciones – (Application programming interface)
GPS	Sistema de posicionamiento global – (Global Positioning System)
IA	Inteligencia Artificial
M2M	Máquina a Máquina
M&M	Módulo conectado a una máquina remota
CAD	Diseño asistido por ordenador
FFF-FDM	Fabricación con Filamento Fundido
SLA	Estereolitografía
ELS	Sinterización selectiva por láser
DFMA	Diseño para la fabricación y el montaje
RIA	Asociación de Industrias Robóticas
ISO	Organización internacional de estándares
GDL	Grado de libertad
ALU	Unidad lógica aritmética
AD	Analógico - Digital
DA	Digital - Analógico
PWM	Modulación por ancho de pulsos
ESCET	Escuela Superior de Ciencias Experimentales y Tecnología
ITI	Ingeniería en Tecnologías Industriales
TCP	Punto central de la herramienta
CNC	Control numérico por ordenador
IDE	Entorno de desarrollo integrado
DIY	“Hazlo tú mismo” – (Do It Yourself)
A	Amperios
V	Voltios
€	Euro
μF	microFaradio
GND	Tierra
VDD	Fuente de tensión positiva de un transistor de efecto campo
VMOT	Voltaje del motor

LED	Diodo emisor de luz
Mx	Métrica
TCP	Punto Central de la Herramienta – “ <i>Tool Center Point</i> ”
ABB	Asea Brown Boveri
RIM	Robótica Industrial y Mecatrónica
BOM	Lista de materiales – “ <i>Bill Of Materials</i> ”

Resumen

La finalidad de este TFG es la creación de distintas herramientas finales diseñadas mediante herramientas CAD abiertas y controladas por hardware libre para realizar diversas actividades para un brazo robótico ABB IRB 120.

También se han creado dos herramientas para el desarrollo de las prácticas en la asignatura de Robótica Industrial y Mecatrónica de 4º de Ingeniería en Tecnologías Industriales (ITI). Se aplican conocimientos de diseño mecánico, programación de microcontroladores, electrónica y simulación de un entorno virtual de uso del robot creado por ABB.

Las herramientas utilizadas en el brazo robótico se han diseñado mediante un programa CAD de software libre (*FreeCAD*) y se han fabricado mediante el uso de dos impresoras 3D y el programa de impresión *Repetier* o *Cura*.

Dos de las piezas cuentan con un motor que permite el movimiento de las garras para poder coger materiales que se encuentren sobre la mesa de trabajo. Los motores se controlan mediante una placa de Arduino y el controlador integrado del brazo robótico.

1. INTRODUCCIÓN

En este TFG se desarrolla una parte de diseño mecánico de unos útiles que posibilitarán la interacción del brazo robótico con el ambiente en el que se encuentra y le permitirán realizar las actividades para las que ha sido diseñado. Por otra parte, el diseño junto con los programas de simulación y entornos generados permitirán el desarrollo de las prácticas de la asignatura Robótica Industrial y Mecatrónica, así como otras asignaturas que se impartan en el futuro que incluyan la programación y el uso de un robot como este.

En el proyecto se incluyen las vertientes del código abierto u “*Open Source*” así como la impresión 3D y en menor medida la industria 4.0 o “Industria Conectada”. Durante este TFG también se utiliza un microcontrolador Arduino que servirá para proporcionar movimiento a las pinzas de las herramientas.

A continuación, se describen las partes por las que está conformado este proyecto y posteriormente se explica el estado del arte donde se muestran los diseños estudiados y ejemplos de sistemas de agarre usados en brazos robóticos, tanto en ámbito industrial como pinzas que se pueden imprimir en 3D y cuyos diseños son fácilmente accesibles en internet.

1.1. Industria 4.0

La Industria 4.0 o *Cuarta revolución industrial* [1] es el proceso por el cual la industria se digitaliza para mejorar la eficiencia, la calidad y la seguridad. Esta industria se apoya en una serie de tecnologías que están en constante evolución y desarrollo.

Las tecnologías que permiten llevar a cabo la industrialización de las empresas y en las que se apoya la Industria 4.0 son:

- Internet de las cosas.
- Inteligencia artificial.
- Comunicación máquina-máquina.
- Big Data.

A continuación, se detallan los cuatro puntos mencionados:

- Internet de las cosas: El internet de las cosas es la conexión de distintos objetos físicos (automóviles, máquinas, electrodomésticos, etc.) que utiliza sensores, mediante el sistema *RFID* o “*Radio frequency identification*” (identificación por radiofrecuencia), y APIs (Interfaz de Programación de aplicaciones) para intercambiar datos y recopilar información.

Algunas de las numerosas aplicaciones de esta tecnología son:

- Edificios inteligentes conectados: Mejoras en la eficiencia energética y de seguridad, control de electrodomésticos y servicios de educación y salud.

- Ciudades inteligentes y transporte: Optimización del transporte, gestión de los procesos que intervienen en la seguridad de los usuarios y control de los recursos.
- Educación: Servicios de aulas virtuales y físicas.
- Electrónica de consumo.
- Salud: Transmisión del estado de los enfermos en tiempo real.
- Automoción: Monitorización del estado de los vehículos, vehículos autónomos, control sobre el estado del tráfico y localización GPS.
- Agricultura y medio ambiente: Monitorización del nivel de contaminación tanto acústica como atmosférica, pronosticar cambios climáticos y cálculos de nutrición.
- Servicios de energía: Información sobre el consumo de energía, control del estado de las redes suministradoras y transportadoras de energía y pronóstico de las tendencias y necesidades futuras.
- Compras: Control logístico a través de etiquetas electrónicas y lectores y compras inteligentes.

Se estima que en el 2020 [2] habrá conectados más de 22,4 mil millones de aparatos electrónicos.

• **Inteligencia artificial:** La inteligencia artificial se define como “la combinación de algoritmos planteados con el propósito de crear máquinas que presenten las mismas capacidades que el ser humano”. Según Stuart Russel, profesor de la universidad de California en Berkeley y San Francisco, y Peter Norvig, director de investigación en Google, se puede distinguir entre varios tipos de inteligencia artificial:

- Sistemas que piensan como humanos: las actividades como la toma de decisiones, el resolver problemas y el aprendizaje se automatiza (redes neuronales artificiales).
- Sistemas que actúan como humanos: Sistemas computacionales que realizan tareas de forma similar a como lo hacen las personas (robots).
- Sistemas que piensan racionalmente: Tratan de imitar la forma de pensar lógica racional de los humanos (sistemas expertos).
- Sistemas que actúan racionalmente: Intentan imitar de manera racional el modo de comportarse humano (agentes inteligentes).

Algunas de las aplicaciones principales de la inteligencia artificial son:

- Educación.
- Climáticas.
- Logística y transporte.
- Sanidad.

Se espera que para el año 2020 el [3] 85% de la interacción con los clientes sea gestionada por IA y que en el 2025 el mercado de IA estará en los 127 mil millones de dólares frente a los 2 mil millones de 2015.

- Comunicación máquina-máquina (Machine to Machine (M2M)): Se considera comunicación M2M [4] a todo tipo de tecnología que haga que los dispositivos en una misma red puedan intercambiar información y funcionar de manera completamente autónoma. Hoy en día el uso de esta tecnología se aplica mayormente en la supervisión de la máquina y el entorno en el que se encuentra.

Hay una serie de elementos fundamentales para poder llevar a cabo la conexión M2M, estos son:

- Dispositivo M&M: Permite la conexión entre una máquina remota y un servidor.
- Máquinas que gestionar: Sensores.
- Servidor: Es el ordenador que gestiona el envío y la recepción de información.
- Red de comunicación: Por cable o a través de una red inalámbrica.

Esta tecnología supone un elemento de grandes cambios para la gestión de almacenes, el tráfico, los servicios logísticos y la cadena de suministro, flotas de vehículos, robótica y más.

- Big Data: Gartner definió en 2001 el Big Data como [5] “datos que contienen una mayor variedad y que se presentan en volúmenes crecientes y a una velocidad superior”. Se incluyen tres términos, los cuales son los fundamentos del Big data:

- Volumen: Cantidad de información recopilada desde distintos orígenes.
- Velocidad: Rapidez con la que se producen los datos a través de las fuentes de información y la capacidad de procesar, tratar y analizar la información.
- Variedad: Existen una amplia variedad de fuentes de obtención de datos.

Recientemente se han añadido dos nuevos términos a la definición de Big Data los cuales son:

- Valor: El valor se crea por medio de estrategias de análisis de datos.
- Veracidad: Conocer, de los datos obtenidos, cuales son válidos, veraces y útiles.

1.2. Impresión 3D

Mediante la impresión tridimensional o fabricación por adición se pueden crear objetos que pueden ser productos finales o partes que posteriormente se ensamblarán para formar un producto final.

Estos productos se crean por la colocación de materiales en capas [6] según un modelo digital el cual se ha diseñado previamente con un programa de diseño asistido por ordenador o CAD por sus siglas en inglés.

Debido a la amplia variedad de materiales con los que se puede fabricar en la impresión 3D, se pueden crear de forma sencilla, a bajo coste, en función de las características que disponga la impresora 3D que se esté utilizando, toda clase de productos diferenciados.

Los métodos más comunes de fabricación aditiva son:

- Fabricación de filamento fundido (FFF) o modelado por deposición fundida (FDM). Mediante este método [7] la impresora calienta y extruye el material, normalmente plástico.
- Estereolitografía (SLA): En este método [8] se utiliza la luz ultravioleta para endurecer o curar la resina.
- Sinterización selectiva por láser (ELS): Es el método más común [9] en la industria. Los materiales se funden y se hacen polvo con el láser y capa a capa se crea el producto.

La primera impresora 3D fue creada por Charles Hull [10], cofundador de 3D Systems, en 1984 junto con el método de impresión por estereolitografía. Charles Hull creó también el formato de archivo de guardado STL el cual sigue siendo el más usado actualmente.

Durante 1988 se crearon los métodos de impresión de ELS por Carl Deckard [11] y de FDM por Scott Crump [12].

Posteriormente durante la década de los 90 [13] se desarrollaron nuevas aplicaciones CAD que facilitaban la creación de diseños y la posibilidad de imprimirlos.

En 1999 en “*Wake Forest Institute for Regenerative Medicine*” (Instituto de Medicina Regenerativa Wake Forest) crearon, mediante impresión 3D, el primer implante para una persona.

Finalmente, en 2004 surgió el movimiento Rep-Rap [14], el cual se basa en la filosofía de código abierto e impresión 3D, donde es posible crear piezas de impresora 3D para poder sustituir las piezas de la propia impresora o incluso crear impresoras de cero [15] para que otros pudieran usarlas, es decir, de autorreplicarse.

Durante el diseño de las piezas de este TFG se han tenido en cuenta ciertas limitaciones que existen a la hora de imprimir en una impresora 3D según el diseño para la fabricación y el ensamblaje o DFMA.

DFMA es un conjunto [16] de metodologías y técnicas para diseñar o rediseñar un producto de forma que se mejore su fabricación, la facilidad para el montaje y los costes de producción.

Las limitaciones encontradas han sido principalmente piezas angulosas que pueden provocar que el material se caiga al imprimirse la pieza, tolerancias a la hora de realizar taladros para la introducción de tornillos o para huecos en los que debían empezar otras piezas, problemas de dimensiones asociado a la reducción de la cantidad de material asociada al enfriamiento del objeto tras la impresión e intentar utilizar el menor número de elementos que compongan las herramientas finales.

La siguiente tabla (Figura 1) muestra algunas limitaciones de la impresión 3D con las soluciones para subsanar los posibles errores [17] que surgen durante la fabricación.







REGLAS DE DISEÑO DE IMPRESIÓN 3D						
	Paredes con soporte	Paredes sin soporte	Soporte y voladizos	Detalles en las piezas	Taladros	Tolerancia
	Paredes que se unen al resto de la figura por al menos dos caras	Paredes que se unen al resto de la figura por una cara	Máximo ángulo con el que se puede imprimir una cara sin requerir soporte	Detalles que se crean en la superficie del modelo	Tamaño mínimo al que se puede imprimir un taladro	Tolerancia a la que puede imprimir una tecnología específica
						
Modelado por deposición fundida	0,8 mm	0,8 mm	45°	0,6 mm ancho y 2 mm alto	Ø 2 mm	± 0,5% (límite inferior ± 0,5 mm)
Estereolitografía	0,5 mm	1 mm	Siempre se requiere soporte	0,4 mm ancho y alto	Ø 0,5 mm	± 0,5% (límite inferior ± 0,15 mm)
Sinterización selectiva del láser	0,7 mm			1 mm ancho y alto	Ø 1,5 mm	± 0,3% (límite inferior ± 0,3 mm)

Figura 1. Reglas de diseño DFMA.

1.3. Filosofía “Open Source”

Open Source, traducido literalmente como “Fuente Abierta”, es un movimiento por el cual cualquier producto, sea físico o no, pueda ser modificado y utilizado por cualquier persona. Que sea abierto no implica que sea gratuito puesto que el producto final puede venderse.

Seguir la filosofía Open Source permite el intercambio constante de ideas entre individuos con distintos enfoques, lo que puede proporcionar una rápida mejora de la tecnología y el conocimiento ya existente. Esta extensión del conocimiento permite que se creen comunidades con intención de participar y que pueden pertenecer o no a un tipo de conocimiento concreto.

Dentro de la filosofía de Open Source se encuentran dos vertientes, el “*software*” libre y el “*hardware*” libre:

- *Software* libre: Fue la primera en originarse y se basa, según *Free Software Foundation* y *Open Software Foundation*, en la creación, la distribución [18] y el uso del *software*. De esta manera el usuario tiene acceso, no solo al código fuente, sino que se le proporciona la libertad de ejecutar, copiar, estudiar, mejorar y cambiar el programa además de poder distribuirlo.

Siguiendo, como se ha explicado anteriormente, la filosofía de Open Source permitió la creación de comunidades de usuarios con distintos enfoques que ofrecen diverso conocimiento e información, en libros, foros, vídeos y otras fuentes de acceso libre y

gratuito para que cada usuario pueda ajustar los programas según sus necesidades y fomentar también el crecimiento y el interés de esta tecnología.

- *Hardware* libre: Extiende la vertiente del *Software* libre [19] hasta los aparatos físicos (sensores, las herramientas y sus diseños). De la misma manera que con el *software*, los diseños de *hardware* [20] están ideados para que sea posible que cualquier persona pueda fabricar, transformar, distribuir, vender y estudiar el producto original o los derivados de este.

Las comunidades que se forman en torno a este movimiento se conocen como “*Makers*” los cuales desarrollan proyectos tecnológicos desde el conocimiento libre. El microcontrolador Arduino es un ejemplo de este movimiento, nacido a partir de la necesidad, de unos investigadores, de tener un microcontrolador barato para realizar proyectos [21].

Estos dispositivos ofrecen una amplia variedad de tarjetas de desarrollo con las que se pueden hacer proyectos de innovación, de electrónica digital, con un entorno de programación sencillo para el usuario.

1.4. Brazo robótico.

El uso de los brazos robóticos se da principalmente en las industrias para realizar tareas complicadas o monótonas con largas y complejas líneas de producción abaratando los costes de fabricación.

La utilización de estas herramientas permite maximizar las ganancias reduciendo los tiempos de fabricación, incrementando la calidad de los productos finales o intermedios y realizando trabajos que pueden suponer un peligro para el ser humano.

Según la Asociación de Industrias Robóticas (RIA) [22], se puede definir robot industrial como:

“Manipulador multifuncional reprogramable con varios grados de libertad, capaz de manipular materias, piezas, herramientas o dispositivos especiales según trayectorias variables programadas para realizar tareas diversas.”

La Federación Internacional de Robótica [23], sin embargo, hace distinción entre el robot industrial de manipulación (brazo robótico) y el resto de robots apoyándose en la definición dada por la Organización Internacional de Estándares (ISO):

“Por robot industrial de manipulación se entiende una máquina de manipulación automática, reprogramable y multifuncional con tres o más ejes que pueden posicionar y orientar materias, piezas, herramientas o dispositivos especiales para la ejecución de trabajos diversos en las diferentes etapas de la producción industrial, ya sea en una posición fija o en movimiento.”

Un brazo robótico está compuesto por una serie de eslabones conectados por articulaciones que permiten el movimiento relativo [24] entre la pareja de eslabones.

Las articulaciones se pueden clasificar de la siguiente manera:

- Lineal: Un eslabón desliza sobre un eje solidario al eslabón anterior.
- Rotacional: Un eslabón gira alrededor de un eje solidario al eslabón que se encuentra antes.

El grupo de articulaciones y eslabones se califica como cadena cinemática. La cadena cinemática es abierta cuando cada eslabón está conectado, mediante una articulación, únicamente al eslabón anterior y siguiente con excepción del primero que se fija a un soporte y el último que tiene uno de sus extremos libres pudiéndosele acoplar una herramienta para ejecutar la actividad para la que ha sido diseñada.

Debido a la similitud entre los brazos robóticos y el brazo humano, las partes por las que está compuesto el brazo robótico reciben el mismo nombre que las del cuerpo humano. Esta comparación se puede ver en la imagen mostrada a continuación (Figura 2).

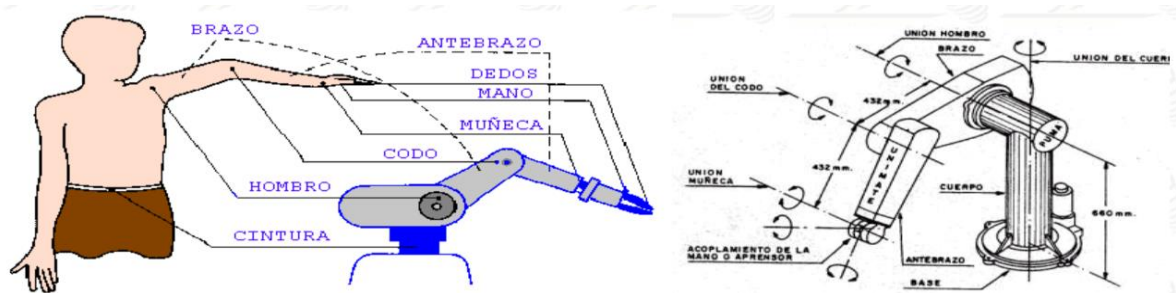


Figura 2. Morfología del brazo humano frente al brazo robótico. [Fuente: <https://es.slideshare.net/leogeekec/morfologia-de-un-robot/>]

Los grados de libertad o GDL que tiene una articulación es el número de movimientos independientes que una articulación puede realizar respecto a la anterior.

La configuración antes descrita (cadena cinemática abierta) hace que el número de GDL del robot coincida con la suma del número de GDL de las articulaciones que lo compone.

1.5. Microcontroladores.

Un microcontrolador es un circuito integrado con un computador completo alojado en su chip con prestaciones limitadas. Normalmente se encarga de realizar una tarea concreta y debido a sus pequeñas dimensiones y reducido peso se suele acoplar junto con la aplicación que controla.

Los microcontroladores son “sistemas cerrados” con todos los componentes de un computador interconectados. Estos se comunican con su entorno mediante los puertos de entrada y de salida (Figura 3).

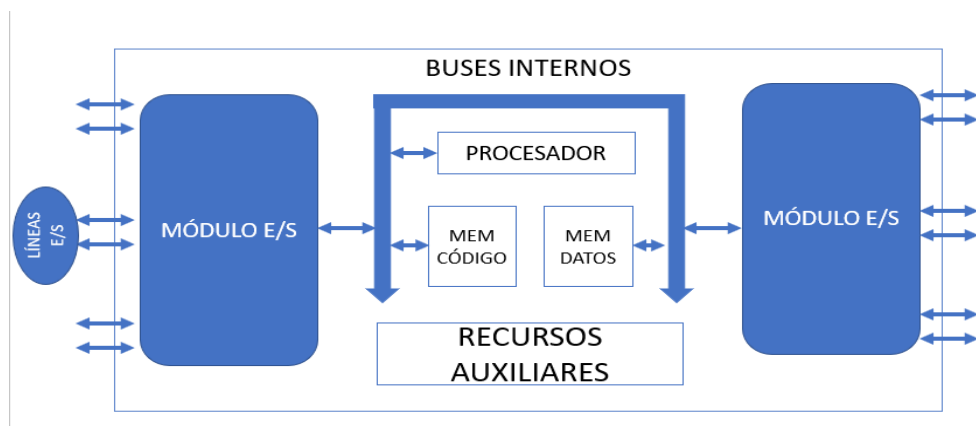


Figura 3. Esquema de componentes de un microcontrolador. [Fuente: Microbótica]

En la arquitectura interna del microcontrolador se pueden distinguir cinco partes principales:

- **Procesador.**

Formado por la “Unidad de control” y el “Camino de datos”. La Unidad de control se encarga de generar las señales de control para su ejecución, mientras que el Camino de datos está basado en una ALU (Unidad Aritmética Lógica) y se encarga de hacer las operaciones aritméticas y lógicas.

- **Memoria de código.**

Contiene las instrucciones del programa, la información que está en el programa debe mantenerse guardada en el circuito integrado de forma permanente para poder llevar a cabo el trabajo para el que está destinado.

- **Memoria de datos.**

Se utilizan para guardar datos por lo que son de lectura y de escritura.

- **Líneas de E/S.**

Los pines que se encuentran en el microcontrolador sirven como entrada y salida para los periféricos que permiten al dispositivo comunicarse con su entorno

- **Recursos auxiliares o periféricos.**

En este apartado entran elementos como conversores de AD (analógico-digital) y DA (digital-analógico), circuitos PWM o temporizadores y contadores.

1.6. Estado del arte: Herramientas para el brazo robótico ABB IRB120 basadas en open source.

La llegada de la filosofía “*open source*” y las impresoras 3D han permitido el diseño y la creación de multitud de proyectos donde se han construido distintos brazos robóticos y herramientas para usar en esos brazos robóticos perfectamente funcionales.


Debido a la compra por parte de la ESCET (Escuela Superior de Ciencias Experimentales y Tecnología) del robot ABB IRB120, surgió la necesidad de conseguir distintas herramientas para el desarrollo de las prácticas y la comprensión del funcionamiento de este robot. Los útiles que usa este tipo de manipulador pueden llegar a alcanzar un precio elevado con los fabricantes oficiales.

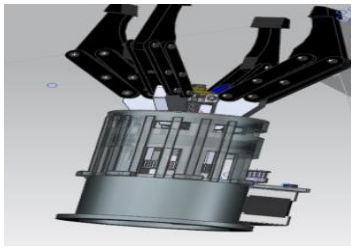
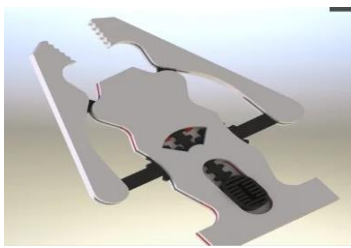

Para ello se han diseñado una serie de útiles, dos de ellos accionados mediante un motor. Las herramientas incluyen el controlador que permite su funcionamiento y así aprovechar las ventajas del “*software*” y “*hardware*” libre para disponer de una mayor variedad de útiles a un precio reducido.

Con el fin de llevar a cabo este objetivo se han estudiado distintos diseños de pinzas para la creación de una herramienta original y en el segundo caso basarse en un diseño ya existente y modificándolo para acondicionarlo en función de las especificaciones del robot y a los recursos disponibles.

Algunos ejemplos estudiados son los siguientes:

Tabla 1. Ejemplos estudiados

Ejemplos estudiados			
Pinza	Tipo	Creador	Comentarios
	Garra de cuatro dedos movidos por un tornillo sin fin. El actuador es un motor paso a paso	Tahamir hasan (GrabCAD) [26]	Mayor sujeción de los objetos a transportar. Tamaño excesivo para un funcionamiento eficiente del robot. Par aportado por el motor insuficiente.

	Garra de cuatro dedos movidos por un tornillo sin fin accionado por un motor paso a paso NEMA 14 situado en el exterior de la garra	Dan Gil (GrabCAD) [27]	Mejor sujeción Tamaño excesivo. Se acciona por motor paso a paso
	Pinza de dos dedos con accionamiento manual.	hectdiaf (Thingiverse) [28]	Ya que es de acción manual modificar el diseño para un motor sería muy complejo. Es un buen ejemplo de pinza paralela.
	Ejemplo de pinza paralela del fabricante Schunk para el agarre y desplazamientos de piezas pequeñas	Schunk [29]	Pinza comercial básica de funcionamiento similar a las diseñadas. Agarre paralelo ofreciendo una mejor sujeción.

1.7. Estructura de la memoria.

La memoria se ha estructurado en cuatro capítulos y dos anexos que se explican brevemente a continuación:

- Introducción: Se presenta el entorno en el que se basa este TFG y cómo se ve influenciado lo expuesto en los siguientes capítulos. También se dan algunos diseños actuales de ejemplos para la creación de las pinzas
- Objetivos: Se muestran los objetivos principales y secundarios que se van a tratar.
- Solución técnica: Se presentan las herramientas fabricadas y los pasos que se han seguido para la realización de cada una, así como, los resultados obtenidos tras realizar las pruebas. También se incluyen los elementos que se han utilizado a lo largo del trabajo. Finalmente, se muestra la metodología aplicada para la finalización de este TFG y una comparación económica de los diseños hechos y una pinza comercial con similares especificaciones.
- Conclusiones: Se presentan los resultados finales junto con líneas futuras que se pueden llevar a cabo con los diseños y configuraciones de electrónica que se han desarrollado en este TFG.
- Anexos: Se incluyen los códigos de los programas utilizados, así como su explicación detallada tanto en el entorno Arduino como en el de RobotStudio.

2. OBJETIVOS

La finalidad de este TFG es crear diversas herramientas funcionales, que se puedan usar en un entorno de aprendizaje como es el caso de las prácticas de la asignatura de Robótica Industrial y Mecatrónica de 4º de carrera de ITI, de bajo coste y con facilidad de reproducción, en caso de deterioro o rotura.

Se ha decidido diseñar e imprimir distintos útiles para que el robot ABB IRB120 desarrolle las actividades para las que ha sido programado sin depender de los actuadores de empresas fabricantes como SCHUNK, ya que tienen un precio elevado. Las herramientas se diseñan basadas en la corriente de *open source* y, en caso de ser necesario, controladas a través de un microcontrolador de Arduino. Para la realización del objetivo principal se ha de cumplir una serie de hitos secundarios:

- Replicar herramientas reales y completamente funcionales, usadas en la industria o en entornos de aprendizaje para poder llevar a cabo distintas actividades, mediante el uso de software y hardware libre.
- Diseñar un sistema de sustitución cómodo de las herramientas para agilizar el proceso y evitar que el robot cese su actividad durante un tiempo prolongado.
- Permitir el acceso a los diseños, programas y distinta información que conforma el proyecto.
- Que permita actualizaciones en el diseño como mejoras o variaciones que se ajusten mejor a las distintas casuísticas que se puedan ocasionar.
- Que sean compatibles con otros elementos complementarios al brazo robótico
- Aprendizaje del programa de simulación del fabricante (*RobotStudio*) utilizado para controlar el robot en un entorno ficticio para comprobar el correcto funcionamiento de las herramientas y que éstas se han diseñado apropiadamente al incluirlas en la estación de simulación desarrollada.

3. Solución Técnica

En este capítulo se presentan las distintas herramientas y las pinzas que se han diseñado durante este TFG, así como el proceso de diseño y elección de componentes.

En segundo lugar, se presentan la programación y los elementos necesarios para el control de los motores encargados de transmitir el movimiento a los dedos de las pinzas, la configuración para la comunicación entre el Arduino y el brazo robótico, las características básicas del robot ABB IRB120 y los distintos procesos para conseguir completar los hitos expuestos en el apartado 2 de este TFG.

Finalmente se presentan los resultados obtenidos durante el montaje, configuración y simulación junto con la metodología aplicada en el trabajo y un estudio comparativo estimado del coste final del proyecto frente al coste que resultaría la compra de las herramientas al fabricante.

Previa a la explicación en detalle de las herramientas y pinzas creadas y la realización de las configuraciones necesarias se presentan los productos finales durante el desarrollo de este TFG.

En primer lugar, se encuentran las herramientas “Porta-Rotuladores” las cuales sirven para almacenar un rotulador. Estas herramientas se han diseñado para las prácticas de la asignatura de Robótica Industrial y Mecatrónica (RIM). Los “Porta-Rotuladores” permiten al robot realizar dibujos mediante trayectorias previamente definidas.

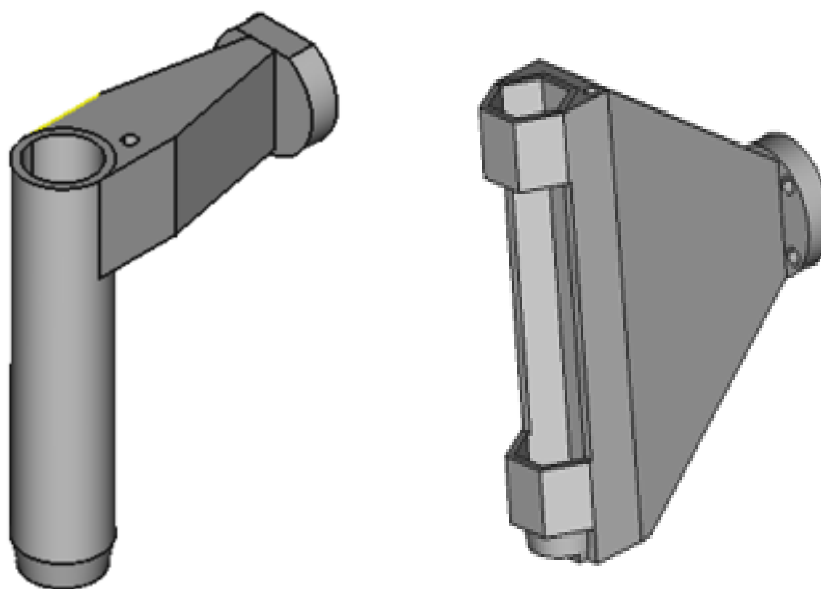


Figura 4. "Porta-Rotuladores".

En segundo lugar, se muestra cómo se ha diseñado la pinza motor paso a paso junto con las conexiones y cálculos realizados para su funcionamiento así como el programa de control del motor y finalmente los problemas encontrados durante esta fase del TFG y el por qué se tuvo que cambiar el diseño original.

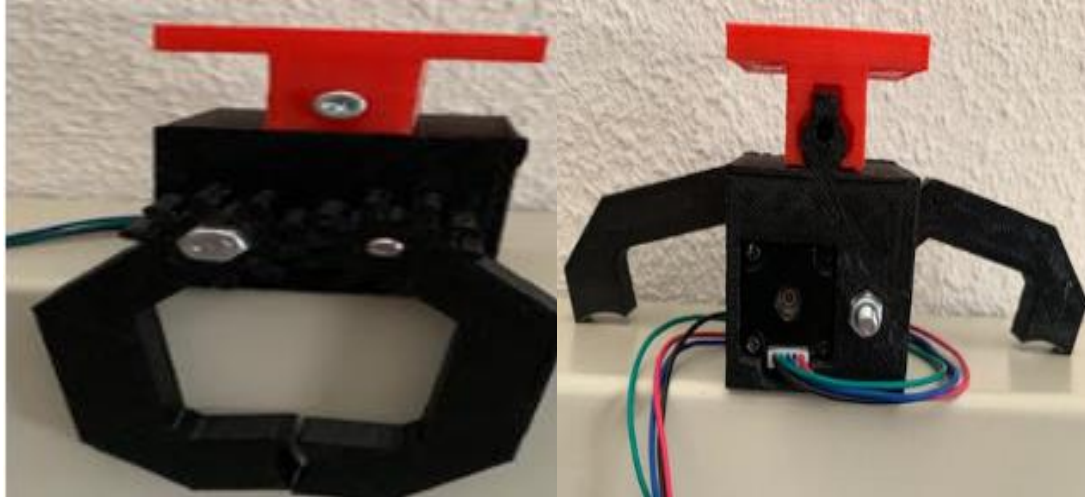


Figura 5. Pinza motor paso a paso.

En tercer lugar y de la misma manera que con la pinza motor paso a paso, se explican los pasos seguidos para la creación de la pinza servomotor, las conexiones que se han llevado a cabo y el programa de control del servo.



Figura 6. Pinza servomotor

En cuarto y último lugar se detalla cómo se ha realizado la simulación y escrito el programa que controla el brazo robótico ABB120 junto con la configuración electrónica necesaria para poder llevar la señal que activará la pinza hasta el Arduino.

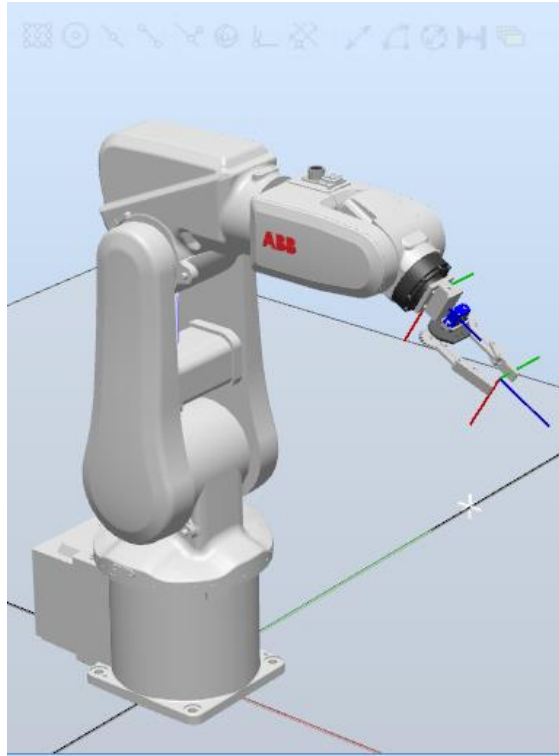


Figura 7. Estación de pruebas en RobotStudio

3.1. Herramientas.

Debido a la necesidad de diseñar unas herramientas para la realización de las prácticas de la asignatura de RIM se crearon dos herramientas que se han llamado “Porta-Rotuladores”. Para el diseño de estas se ha utilizado el “software” FreeCAD el cual se explica en el apartado 3.7.1.5.

3.1.1. Porta-Rotulador.

Estas herramientas se diseñaron con el fin de utilizarse en la primera práctica de la asignatura de “Robótica Industrial y Mecatrónica” del cuarto curso del grado de ITI.

Se crearon dos modelos distintos para obligar a los alumnos a modificar la resolución del problema en función del útil que tuvieran, además de ofrecer la posibilidad de abordar el problema desde otra perspectiva.

La función de estos “Porta-rotuladores” es la de servir de contenedor de un rotulador para que el robot ABB IRB120 pueda realizar una serie de trazos previamente programados.

La primera herramienta está diseñada de forma que al unirla a la brida del brazo robótico esté en una dirección perpendicular a esta, asemejándose al manejo de un rotulador que pueda tener un ser humano a la hora de dibujar.

Uno de los problemas que existen al imprimir en 3D es que no se pueden crear superficies abovedadas sin que se genere una estructura interna de apoyo o sin que las paredes del diseño se venzan.

La parte en la que se encuentra ubicado el rotulador se ha diseñado con forma hexagonal como solución para evitar la generación de la estructura interna y que se pueda dañar la pieza final durante el proceso de mecanizado necesario para retirar la estructura.

En la parte baja de la herramienta hay un cono truncado para conseguir que la punta del rotulador, parte sensible que se puede romper fácilmente, tenga una mayor sujeción.

Tras finalizar el diseño se observó que era posible quitar parte del material del contenedor del rotulador lo que permitía reducir el tiempo de impresión y el material utilizado durante la impresión. Obteniendo así el siguiente resultado final (Figura 8).

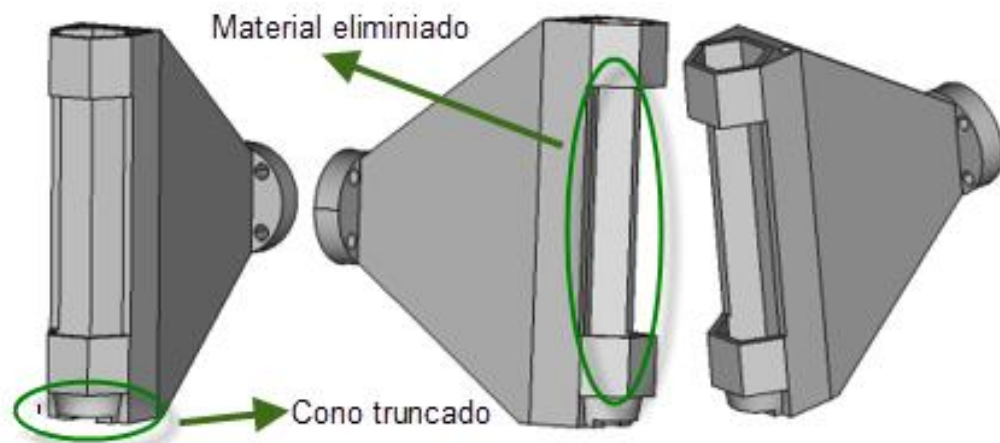


Figura 8. Porta-Rotulador 1.

La segunda herramienta de este tipo se diseñó para que existiera una mayor distancia entre la brida del robot y la punta del porta-rotulador haciendo que el brazo tenga mejor movilidad y reduciendo las posibilidades de que el robot se choque con la mesa o consigo mismo. Esta posición (Figura 9) se asemeja más a la que puede tener una herramienta que se acople a este tipo de robots en la industria.



Figura 9. Robot ABB IRB con herramienta industrial

El contenedor del rotulador es un cilindro terminado en un cono truncado, igual que la pieza anteriormente explicada, para que al imprimirse en dirección perpendicular al suelo no se generen estructuras internas ni exista ningún riesgo de que el material se venza. En esta herramienta para facilitar la impresión se ha hecho un corte en el enganche con la brida que permite su fabricación desde la cara superior de la cuña

Este diseño hace que se utilice una cantidad de material de impresión mucho menor debido a que la parte que sujeta el enganche a la brida y el contenedor del rotulador no es necesaria. Otra de las ventajas de usar esta forma es que al ser más sencilla es más probable que no haya problemas de impresión y se obtenga en menos tiempo. Más adelante hay una tabla con las dos piezas y sus características importantes (Tabla 3)

El resultado final obtenido es el siguiente (Figura 10):

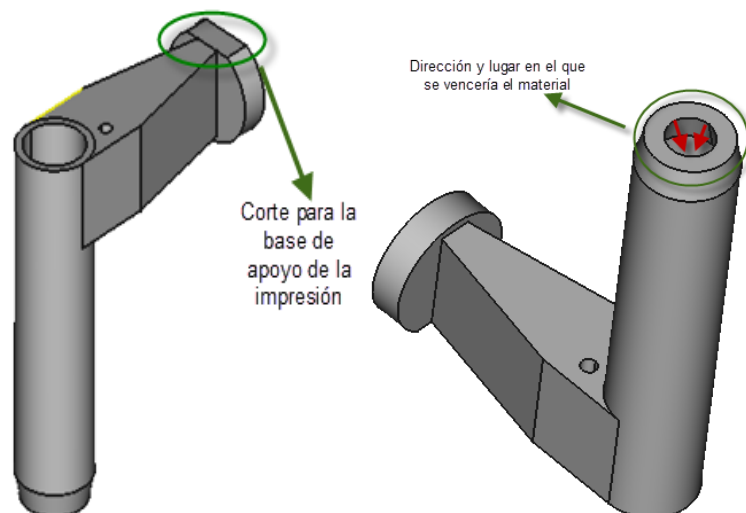


Figura 10. Porta-Rotulador 2

Las dimensiones del Porta-Rotulador están hechas para contener un rotulador de la marca BIC que es el que se ha usado durante las prácticas de la asignatura. Para evitar que el rotulador se salga al usar el robot, se ha diseñado una tapa que se une a la herramienta mediante unión roscada.

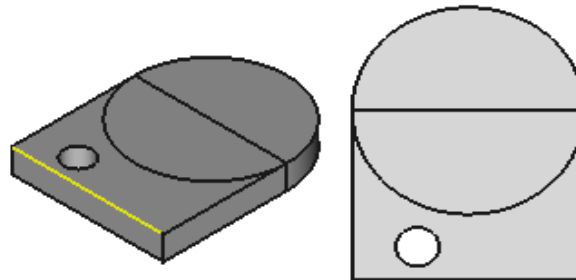
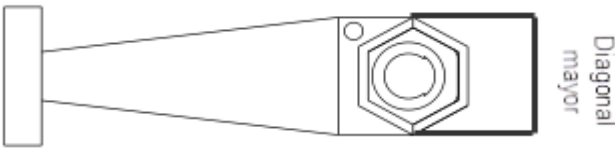
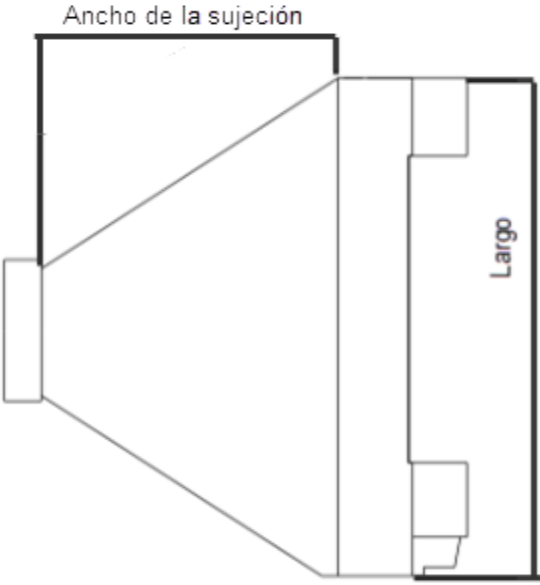


Figura 11. Vista axonométrica y planta de la tapa.

Algunas de las medidas más destacables de las dos piezas son las siguientes:

Tabla 2. Partes clave de Porta-Rotulador 2.

Porta-Rotulador 1		
Parte de la pieza	Descripción	Medidas
	Parte interna de la pieza donde se introduce el rotulador	Diagonal: 28 mm
	Sujeción de la brida y el contenedor del rotulador	<ul style="list-style-type: none"> - Ancho de la sujeción: 80 mm - Largo de la sujeción y contenedor del rotulador: 141 mm -Ángulo que forma con la brida: 41,39°

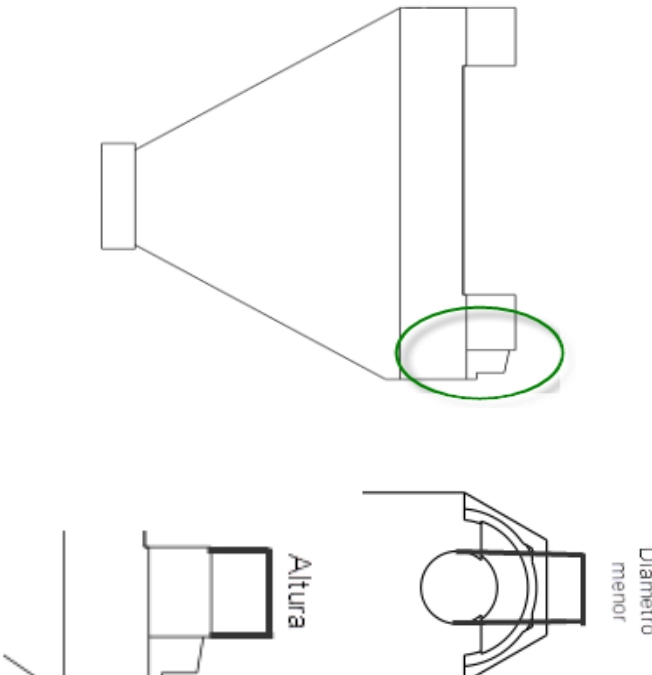
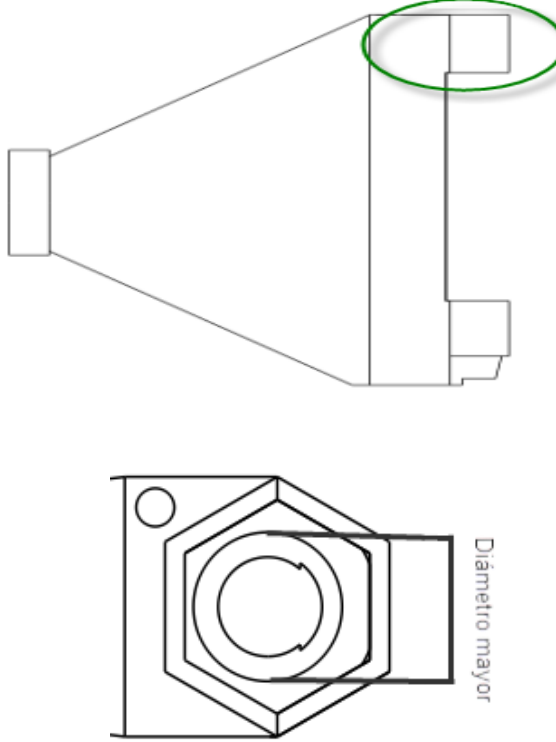
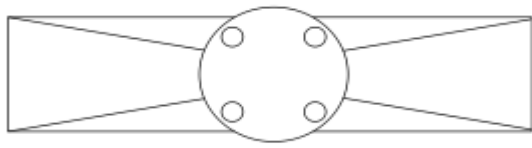
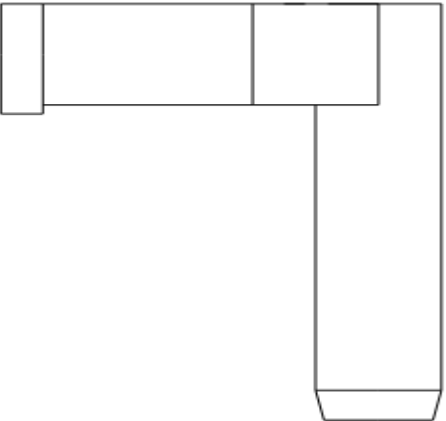
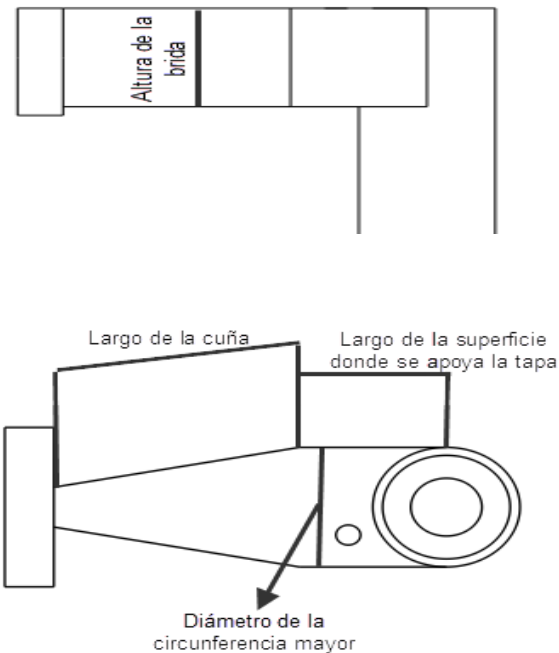
 <p>Altura</p> <p>Diámetro menor</p>	<p>Vista lateral e inferior de la punta del Porta-Rotulador</p>	<p>- Altura: 8,5 mm - Diámetro menor: 13 mm</p>
 <p>Diámetro mayor</p>	<p>Vista de la punta de la herramienta desde la parte superior del porta-rotulador</p>	<p>Diámetro mayor: 19,42 mm</p>
	<p>Enganche para la brida del robot</p>	<p>Diámetro de la brida: 40 mm</p>

Tabla 3. Partes clave de Porta-Rotulador 2

Porta-Rotulador 2		
Parte de la pieza	Descripción	Medidas
	<p>El contenedor del rotulador con las mismas dimensiones mostradas en la tabla anterior y según las medidas del rotulador</p>	<p>Largo del contenedor del rotulador: 141 mm</p>
	<p>Unión entre la brida del robot y el contenedor del rotulador.</p> <p>Proporciona una distancia de seguridad entre estos dos elementos para evitar que el brazo pueda golpear la mesa o a sí mismo</p>	<ul style="list-style-type: none"> - La circunferencia mayor del contenedor es de diámetro de 30 mm. - La altura de la cuña es de 34 mm. - El largo de la cuña es de 49,79 mm. - Largo de la superficie de apoyo de la tapa de 29,82 mm

La mayor parte del material de impresión se utiliza en las uniones a las bridas como se puede observar en las tablas que se encuentran arriba. Esto implica, como se ha comentado anteriormente, una gran diferencia de tiempo en la impresión de las dos piezas siendo mucho mayor el tiempo del primer Porta-Rotulador.

Un mayor tiempo junto con el diseño de partes complejas implica que existen más posibilidades de que se produzca algún error durante la fabricación por ello se aconseja usar el diseño de la segunda herramienta en caso de que se necesite replicar o modificar de alguna manera.

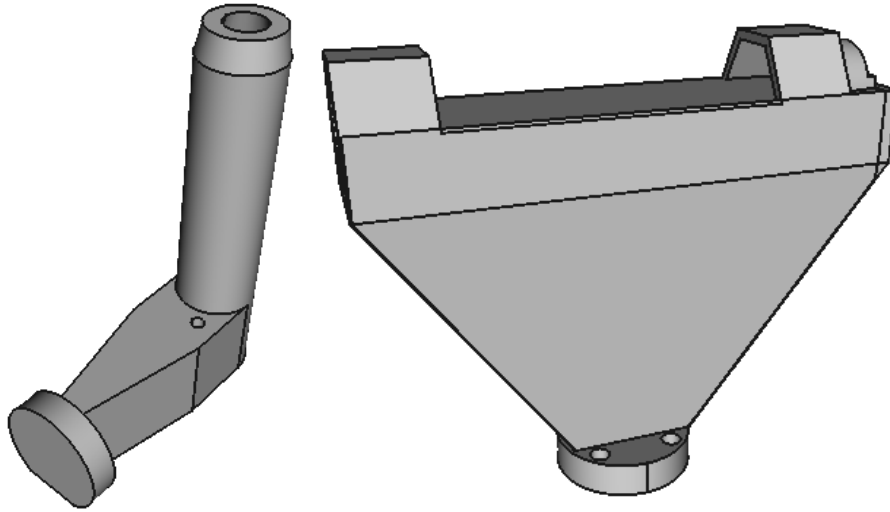


Figura 12. Posiciones de las herramientas durante la impresión

Por otro lado debido a la dificultad de la forma de la primera herramienta, el cálculo del centro de gravedad y del TCP son más complejos, ambos necesarios para la realización de las prácticas de la asignatura donde se usarán las piezas, esto obliga a los alumnos a usar distintos métodos para la obtención de estos datos.

3.2. Actuadores.

Los actuadores [30] son dispositivos que convierten la energía eléctrica, hidráulica o neumática en energía mecánica y están compuestos por los siguientes elementos:

- Sistema de transmisión: transmite el movimiento del actuador.
- Sistema de accionamiento: Produce el movimiento.
- Sistema reductor: Controla el par aportado y la velocidad del actuador a los valores necesarios.
- Sistema de control: Envía órdenes al actuador para que este se mueva como se requiere.

Los actuadores a su vez se clasifican según el tipo de energía que transformen:

- Hidráulicos: La fuente de energía es un fluido a presión como por ejemplo el aceite mineral
- Neumáticos: Su fuente de energía es el aire comprimido y aprovechan la compresión o descompresión de este aire para generar un movimiento rectilíneo o circular

- Eléctricos: Convierten la energía eléctrica en energía mecánica rotacional. Dentro de este grupo se encuentran los motores de corriente continua, los de corriente alterna y los motores paso a paso.

El actuador utilizado por defecto junto con el robot es de tipo neumático del fabricante *SCHUNK* mientras que en este TFG los actuadores diseñados son sistemas de agarre de tipo eléctrico.

Durante el desarrollo de este TFG se diseñó en primer lugar una pinza cuyo movimiento lo producía un motor paso a paso. Por razones que se explican en el apartado 3.2.1 tuvo que desecharse y cambiar el diseño para la utilización de una pinza accionada por un servomotor.

3.2.1. Motor paso a paso.

Un motor paso a paso es un actuador de tipo eléctrico [31] que convierte impulsos eléctricos en desplazamientos angulares. Es un motor que gira un ángulo determinado y cuenta con entre cuatro y seis terminales.

Este tipo de motores pueden ser de imán permanente, de reluctancia variable o híbrido, que es una mezcla de los dos tipos. Puede rotar en ambas direcciones, moverse en incrementos angulares precisos, conocidos como pasos, mantener el par a velocidad cero y puede ser controlado por circuitos digitales. El número y el rango de los pulsos controla la posición y la velocidad del eje del motor.

Los motores paso a paso pueden ser bipolares o unipolares, se alimentan mediante una fuente de corriente continua o alterna y requieren de circuitos digitales para energizar las bobinas y que el motor rote.

Las aplicaciones [32] donde se usan son generalmente aquellas que precisan de un gran control de posición y donde el coste o la complejidad de un sistema de retroalimentación es injustificado. Algunas de estas aplicaciones son:

- Máquinas CNC como las fresadoras.
- Cortadoras láser.
- Manipuladores robóticos.
- Discos duros.
- Impresoras 3D.

El motor paso a paso utilizado en la primera pinza creada es un motor NEMA 8.

Es de tipo bipolar híbrido que da 200 pasos por vuelta por lo que en cada paso se mueve $1,8^\circ$. El motor cuenta con 4 cables de colores, cada par conectado a una bobina distinta. Puede ser controlado por puentes H, uno para cada bobina, o por un controlador de paso a paso bipolar como el pololu A4988 explicado en el apartado 3.7.2.3.

Está formado por una parte fija, estátor, y una parte móvil, rotor. El sistema de transmisión de movimiento es el eje del motor.

La primera pieza que se explica a continuación es la pinza accionada por el motor paso a paso. Esta pinza está formada por tres partes independientes unidas mediante unión roscada que facilita la sustitución de las partes o si, por ejemplo, se quieren sustituir los dedos de la pinza por otros con mayor superficie de agarre se pueda hacer con facilidad. Se entrará en detalle de su diseño en el apartado 3.3.1.

La razón de usar un motor paso a paso en primer lugar es debido a que ofrece un buen control y precisión a la hora de seleccionar los pasos para los que se quiere mantener abierto o cerrado el sistema de agarre, su programación es sencilla, se puede variar la velocidad con la que se abre o cierra la pinza, el motor mantiene el par tanto en movimiento como en estado de parada y mantiene la posición sin que genere vibraciones. En un primer momento, dado que los trabajos a los que se le iba a destinar era el movimiento de cubos o piezas impresas en 3D, se pensó que con las características que tenía sería suficiente.

Otra de las razones para probar el uso de un motor paso a paso fue que la gran mayoría de brazos robóticos están accionados por servomotores y se quería comprobar si otro tipo de motor con una gran precisión en el movimiento y capaz de mantener el par en todo momento sería válido para desempeñar el trabajo del servo.

Para que el motor paso a paso gire es necesario que se invierta la corriente que circula por las bobinas según una secuencia determinada.

Paso	Bobina 1A	Bobina 1B	Bobina 2A	Bobina 2B
Paso 1	1	0	1	0
Paso 2	1	0	0	1
Paso 3	0	1	0	1
Paso 4	0	1	1	0

Invertidos
Invertidos

Figura 13. Secuencia de motores paso a paso bipolares.

La hoja técnica del motor escogido se encuentra en el apartado 3.7.2.4



Figura 14. Motor paso a paso

3.2.2. Servomotor.

El servomotor [30] es un actuador de tipo eléctrico que se utilizó como alternativa al motor paso a paso tras observar que el par que proporcionaba no era lo suficientemente alto como para que la pinza realizara las tareas para las que ha sido diseñada.

Es un dispositivo para el control de precisión de velocidad, posición y par motor. Un servo es un motor de corriente continua, de corriente alterna o de corriente continua sin escobillas (evita que el motor se caliente en exceso debido a la corriente que pasa por las escobillas y a la resistencia de estas) combinado con un dispositivo de sensor de posición como un “*encoder*” (convierte el movimiento mecánico en pulsos) digital.

El servo se maneja mediante un controlador programable que procesa la entrada del sensor y genera voltajes amplificados y corriente al motor para conseguir unos perfiles de movimiento específicos. A esto se le llama un sistema de control cerrado, ya que incluye un sensor de retroalimentación. Normalmente un servomotor es más caro, pero puede tener una respuesta mucho más rápida y fluida. Sin embargo, existen servos de bajo coste como el usado en este TFG.

Una de las diferencias que existen entre el motor paso a paso y el servo es que el servo puede moverse de forma continua en lugar de hacerlo en pasos discretos. En el caso del servo utilizado tiene el rango de movimiento limitado a 180° mientras que el motor paso a paso puede dar vueltas completas.

El servo utilizado tiene un par mucho más alto que el motor paso a paso siendo de un tamaño similar. Esto se debe a que el servo tiene reductora que aumenta el par mientras que se disminuye la velocidad, esto se puede ver en el apartado 3.7.2.5 en las hojas de características.

Un servomotor [33] está compuesto por una tarjeta controladora (sistema de control), un motor de corriente continua (sistema de accionamiento), una serie de engranajes entre los que se incluye el que se encarga de transmitir el movimiento directamente al eje del motor (sistema de transmisión y reductor) y un potenciómetro interno (sensor de posición).

El cableado está compuesto por tres cables de distinto color encapsulados en un terminal hembra. El cable rojo corresponde con la entrada de tensión, el cable naranja con la entrada de la señal de control desde el Arduino y el cable marrón que se conecta a tierra.

La siguiente imagen muestra un esquema del interior del servomotor.

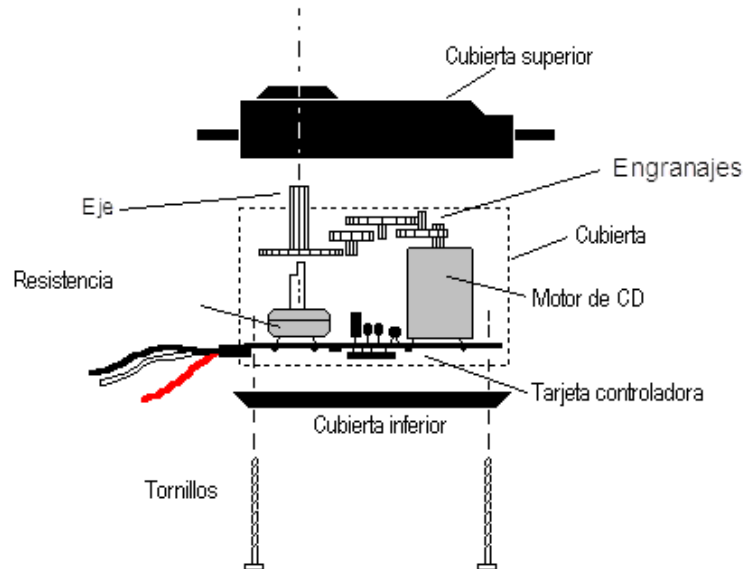


Figura 15. Servomotor. [Fuente: <https://www.monografias.com/trabajos60/servo-motores/servo-motores.shtml/>]

3.3. Pinzas.

Para llevar a cabo los objetivos expuestos en el TFG se han diseñado dos pinzas distintas.

La primera pinza está accionada por el motor paso a paso, mientras que la segunda se mueve gracias a un servomotor como se ha explicado anteriormente.

En este apartado se explica cómo se han creado estas pinzas y las características más importantes de las partes que las componen, los diversos problemas que han surgido a la hora de imprimirlas o los errores de diseño que han surgido obligando a rehacer algunas piezas.

Ambas pinzas utilizan elementos comunes como son el sistema de agarre y dos dedos de movimiento paralelo, cuyo movimiento se da con un engranaje conductor conectado al motor y enfrentado al engranaje del otro dedo.

Se ha aprovechado en ambas pinzas el mismo sistema de agarre a la brida.

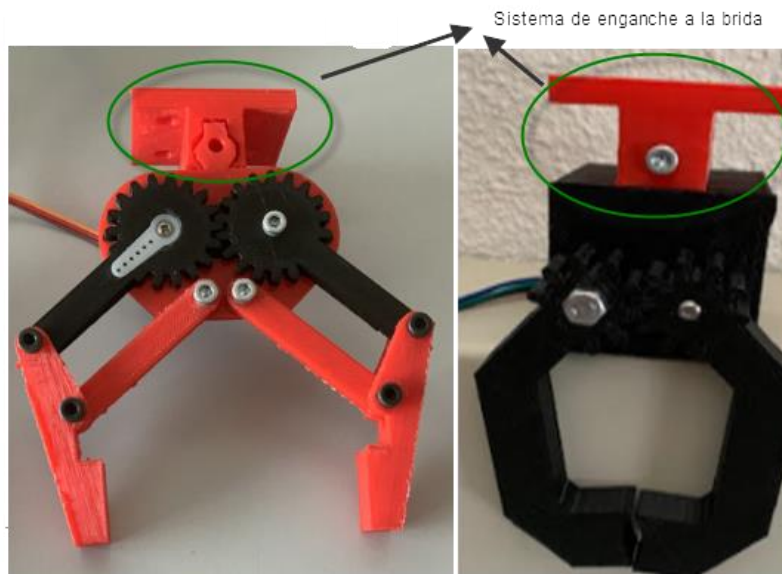


Figura 16. Mismo sistema de agarre en ambas pinzas.

Todas las piezas que se muestran en este apartado han sido diseñadas mediante FreeCAD y se encuentran en el GitHub del autor [34] junto con piezas que han servido de base para crear las siguientes.

3.3.1. Pinza de motor paso a paso.

La pinza del motor paso a paso fue la primera pinza diseñada. Esta herramienta consiste en un sistema de enganche, un contenedor para el motor y el sistema de agarre de las piezas que se deben transportar.

En un primer momento el útil se diseñó de forma que el enganche a la brida y la caja del motor fueran una parte entera.

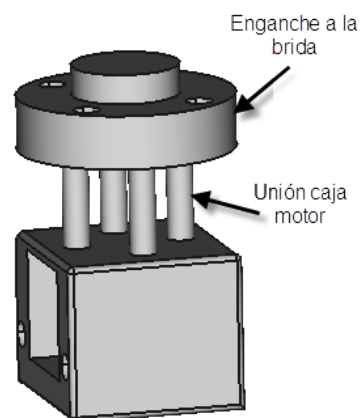


Figura 17. Pinza prototipo.

La parte superior es el enganche a la brida del brazo robótico que se une a la caja del motor mediante cuatro cilindros de pequeño tamaño. Se decidió utilizar este sistema en lugar de una única pieza maciza porque por la poca separación que hay entre los taladros uno de los extremos debería ser muy estrecho, convirtiéndose en un concentrador de tensiones provocando la posible rotura de la pieza con relativa facilidad.

La parte de la caja del motor está diseñada para contener un motor paso a paso con las dimensiones de un NEMA 8 y la posibilidad de asegurar el motor mediante una tapa sujeta por unión roscada para evitar que el paso a paso se caiga. Se usa un taladro para la sujeción del dedo que forma parte del sistema de agarre.

Finalmente, esta solución se acabó descartando. La unión entre el enganche de la brida y la caja del motor exigía que a la hora de imprimir se crearan una serie de estructuras de apoyo adicionales que con la impresora que se tenía disponible en ese momento habrían sido muy complicadas de retirar y había muchas posibilidades de dañar la pieza en el proceso.

Como evolución de esta solución se diseñó la pieza en tres partes individuales como se explica en este apartado.

A continuación, se detalla el diseño final de la pinza del motor paso a paso y los pasos que se han seguido hasta llegar al resultado esperado.

La primera parte es la unión a la brida del brazo robótico. Está diseñada de tal forma que sea una pieza que se quede en todo momento unida al robot. Puesto que el espacio que ocupa la brida es reducido y los taladros donde se roscan los tornillos para sujetar la pieza están muy juntos, como se puede ver en el apartado 3.7.2.7 donde se dan datos técnicos del brazo robótico.

El enganche (Figura 19) que se ha diseñado es una pieza en forma de “I” donde en la cara superior se han hecho unos taladros del mismo tamaño y disposición que los de la brida del brazo robótico. En el tronco de la pieza se ha hecho una hendidura a modo de carril y con un taladro en medio de esta para que puedan asegurarse las piezas una vez hecha la unión y que no se separen la una de la otra.

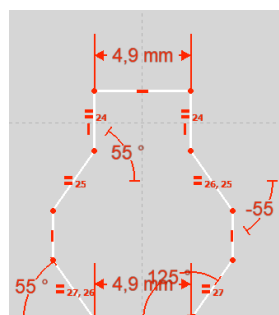


Figura 18. Dimensiones de la deslizadera

Puesto que el enganche se imprime sobre la cara que va unida a la brida del robot para evitar que los taladros donde van los tornillos de amarre se impriman con errores o se caiga el material de impresión por las limitaciones (vistas en el apartado 1.2) el ángulo, que debe ser mayor a 45° respecto de la horizontal, es el de la parte inferior.

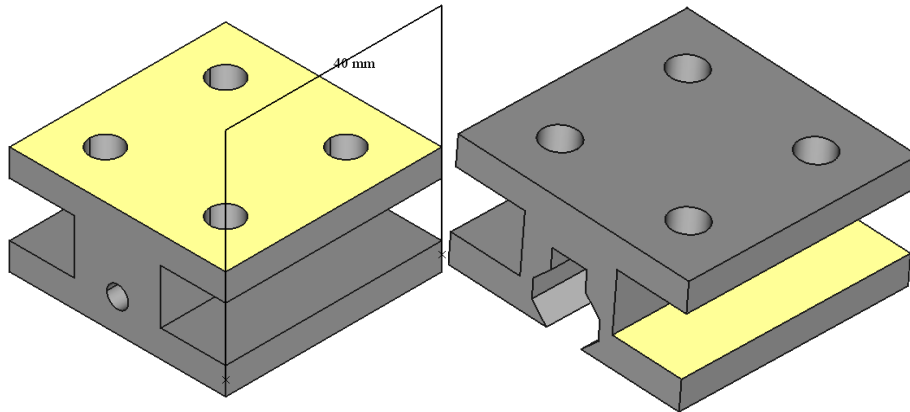


Figura 19. Modelado del enganche. Vista trasera y frontal

Las dimensiones del enganche están realizadas según la longitud del motor con una superficie cuadrada. La altura es de 2 cm (centímetros).

Una vez impresa y unida a la caja del motor el resultado final es el siguiente.



Figura 20. Unión del enganche con la caja motor.

En la figura anterior se muestran distintas vistas de las dos partes unidas, la pieza superior es el enganche a la brida y la inferior la caja del motor que se explica más adelante.

Posteriormente, esta pieza se ha modificado quitándole la cara inferior para permitir que los tornillos encajaran más fácilmente y en adición, al retirar material se ha reducido el tiempo de impresión.

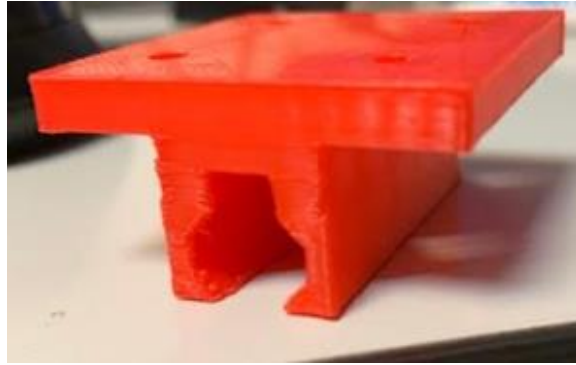


Figura 21. Diseño final del enganche.

Los taladros de estas piezas se han diseñado según las especificaciones del brazo robótico aportadas por ABB. Consiste en cuatro taladros de métrica cinco separadas a una distancia de 15,75 mm y un ángulo entre ellos de 90°. Puesto que al imprimirse el material se comprime al enfriarse se ha aumentado el radio de cada taladro en 0,2 mm por las características de impresión.

Esta información puede consultarse en la tabla de la Figura 1 del apartado 1.2.

La segunda parte de la pinza consiste en la “caja motora”. Esta pieza está diseñada de forma que contenga el motor paso a paso escogido y a la que se pueden unir, mediante unión roscada, los dedos de la pinza y el enganche a la brida del robot.

El motor paso a paso se ubica en un hueco dimensionado con las medidas del actuador con una cierta holgura para permitir que el paso a paso pueda ser introducido aplicando fuerza pero que sea complicado que se mueva una vez está instalado. El hueco se ha creado de forma que hay un espacio específico para el terminal del cableado del motor.

La cara frontal tiene dos agujeros. El primer agujero (Figura 22) es un taladro por el que se introduce un tornillo de métrica 4 (M4) y una longitud superior a los 40 mm de la caja para poder sujetar el cabezal de uno de los dedos que estará cerrado por una tuerca de seguridad de la misma métrica. La tuerca se deja lo suficientemente apretada para permitir el giro sin que se vea impedido por el rozamiento entre las piezas, pero sin que esta se separe demasiado de la caja del motor ni del otro engranaje.

El segundo agujero (Figura 22) es el correspondiente a la salida del eje del motor. En el eje del motor se coloca el otro dedo cuyo engranaje conducirá al dedo sujeto por el tornillo.

Finalmente, en la cara superior, está la deslizadera que se introducirá por el carril del enganche. Para modelar esta parte de la pinza se ha usado el carril creado en el enganche y a partir de un cubo mediante la herramienta “resta” de *FreeCAD* se ha generado el “negativo” de esta pieza y posteriormente se ha reducido el tamaño para que se pueda introducir en el enganche.

La deslizadera se ha diseñado con un taladro en el centro y dos milímetros más corta que la longitud de la caja para poder unir mediante un tornillo de M2 las dos piezas y poder asegurar que estas no se van a separar. No ha sido necesario usar un tornillo con una longitud superior a la de la deslizadera puesto que debido a la poca holgura dejada se ha podido roscar y asegurar una buena sujeción.

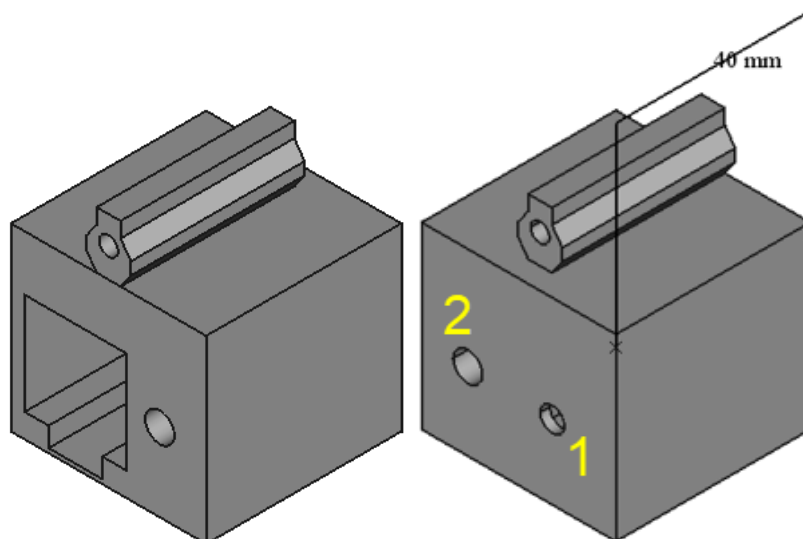


Figura 22. Vista axonométrica frontal y trasera de la caja del motor

El último componente de esta herramienta es el sistema de agarre para el transporte de las piezas.

Está formado por dos dedos simétricos, en uno de los extremos tiene el engranaje encargado de transmitir el movimiento y en el otro la parte de agarre de la pieza.

El dedo que se atornilla tiene en el extremo del engranaje una apertura del mismo tamaño que el tornillo de métrica 6 mientras que la apertura del dedo que se une al motor tiene el tamaño y forma del eje del paso a paso.

El engranaje está creado a partir de una de las herramientas de diseño incluidas en el programa de dibujo FreeCAD con un diámetro de 17,5 mm y con 10 dientes con lo que con la siguiente fórmula se puede calcular el módulo de los engranajes:

$$1. \quad \text{módulo} = \frac{\text{diámetro}}{\text{número de dientes}}$$

Que en este caso es de 1,75 mm

En la siguiente figura se muestran las dimensiones básicas de diseño de los dedos. Los engranajes están girados 45° uno respecto de otro para que puedan encajar y girar entre sí.

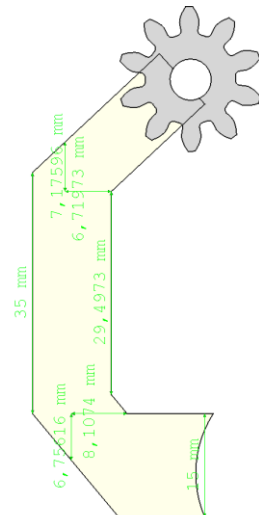


Figura 23. Dimensiones del dedo.

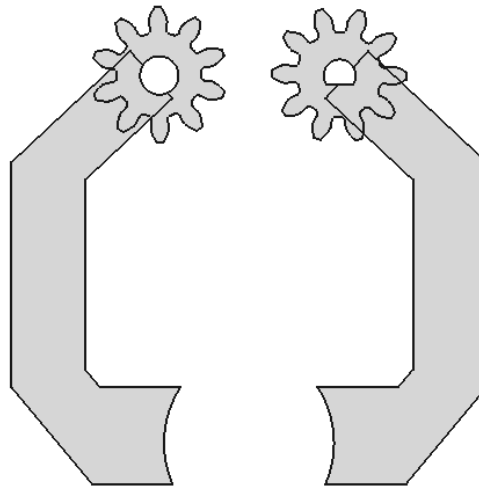


Figura 24. Sistema de agarre completo.

Tras ensamblar todas las piezas el resultado final es el siguiente:

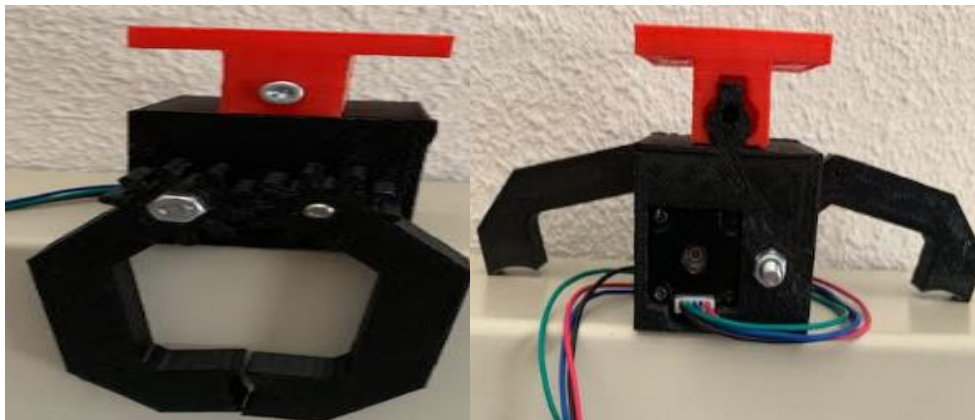


Figura 25. Vista frontal y trasera de la pieza final

Una vez la herramienta estaba ensamblada se realizaron diversas pruebas antes de conectarla al brazo robótico para comprobar su funcionamiento.

Se encontraron dos problemas que provocaban que el agarre de la pinza no fuera suficiente, aunque el movimiento de los dedos se produjera sin ningún inconveniente.

El primer problema era debido a que el par que proporciona el motor era demasiado pequeño como para mantener una fuerza suficiente para la sujeción de cualquier pieza, por muy ligera que fuera.

El segundo problema fue debido a que existía juego entre los dientes de los engranajes de los dedos.

Para solucionar este problema se decidió diseñar una nueva pinza y cambiar el motor por uno con mayor par para que el agarre fuera el suficiente y que así fuera una herramienta funcional.

Continuando con la idea de utilizar un motor paso a paso se comprobaron las especificaciones de otros motores NEMA como el motor NEMA 14 o el motor NEMA 17 que podrían ofrecer un par suficiente para mantener el agarre aun tratándose de piezas pesadas.

Estas opciones se desecharon por el peso y dimensiones de estos motores ya que reducían las capacidades de carga y de rango de movimiento del robot.

Esta modificación habría significado realizar el diseño de nuevo para poder contener el motor y un sistema de agarre mayor para que quedara en consonancia con el resto de la herramienta.

Por las razones expuestas anteriormente se decidió cambiar el motor y usar un servomotor ya que ofrecen un par mucho mayor al que aporta el motor NEMA 8 mientras que las dimensiones y peso son menores.

El cambio de actuador implica la modificación del programa de control, así como la geometría completa del sistema de agarre.

3.3.2. Pinza de servomotor.

Como se ha explicado en el apartado anterior, los errores de diseño y en la elección del motor obligaron al diseño de una nueva herramienta.

El útil se ha tenido que rediseñar por completo debido a que la forma y medidas del motor son completamente diferentes al anterior. Se ha intentado aprovechar partes ya creadas para agilizar el proceso y evitar el gasto de material de impresión.

De esta manera se pudo conservar el enganche a la brida, objetivo al que estaba destinada al principio, poder diseñar distintas piezas con diferentes características pero que mantuviesen el sistema de sujeción a la brida agilizando el intercambio de herramientas.

El diseño de la pieza está basado en la pinza del usuario de “*GitHub*” Obijuan [35] a partir de la cual se han hecho las modificaciones necesarias para adecuarlo a los materiales disponibles y a los requisitos necesarios para el cumplimiento del TFG.

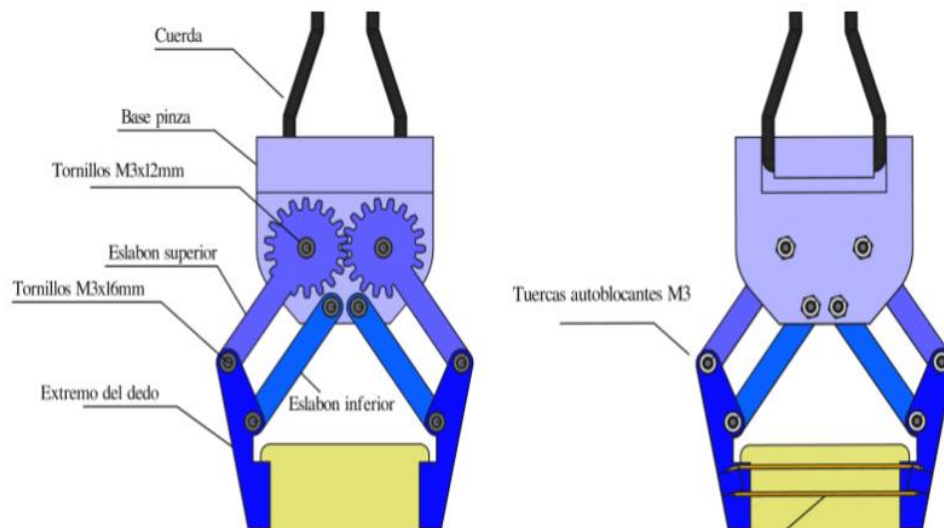


Figura 26. Pinza diseñada por Obijuan

Se trata de una pinza con dos dedos conectados a través de un engranaje, similar al diseño de la pinza anterior, de presión paralela con un mecanismo de cuatro barras que limita el movimiento de los dedos haciéndolo más estable y evitando el posible juego que exista entre los engranajes y su consecuente movimiento cuando el motor no esté en funcionamiento.

Para explicar el mecanismo de paralelogramo articulado se reduce al modelo alámbrico donde se simplifican las barras y nudos (Figura 27).

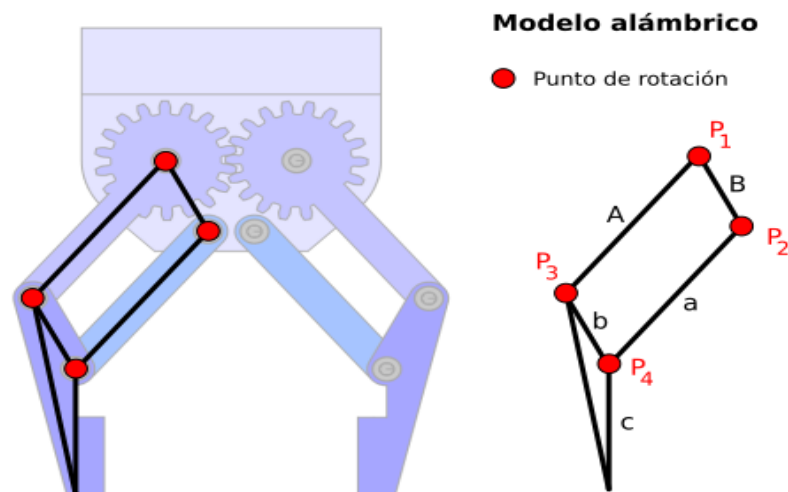


Figura 27. Modelo alámbrico.

Existen cuatro puntos de rotación (P1, P2, P3 y P4) y cinco barras. Las barras con la misma letra son paralelas. Los puntos P1 y P2 se unen a la base donde se colocará el servo por lo que no existe ningún desplazamiento de esos puntos. La barra c es la parte del dedo que realiza el agarre que, con este mecanismo, permanece siempre vertical.

Para poder utilizar el servo disponible se ha modificado la base de la pinza realizando un hueco con las medidas de este y modificando la posición en el espacio de la base donde permita el contacto entre los engranajes de los dedos una vez esté uno de ellos unido al servo (Figura 28). El motor se sujeta a la base gracias a dos tornillos de métrica uno (M1) que venían junto con el servo.

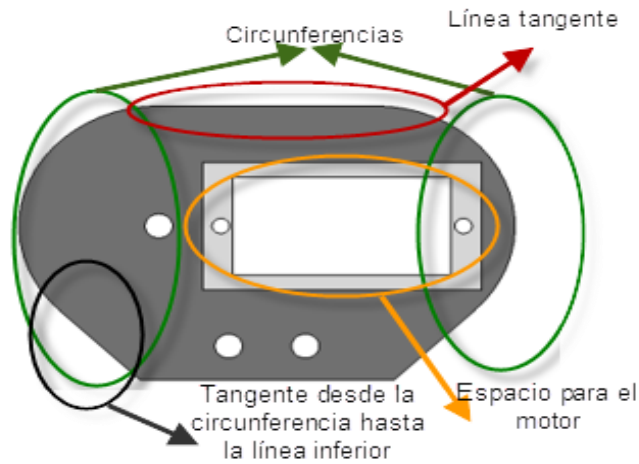


Figura 28. Base de la pinza modificada.

La base está formada por dos circunferencias y una línea tangente a ellas en la parte superior, mientras que en la parte inferior se ha creado una línea a 15 mm del hueco donde se colocará el servo y se ha unido una tangente a las circunferencias desde cada extremo de la línea inferior (Figura 28).

La base tiene unos taladros de M3 por los que se introducirán los tornillos que fijarán las partes de la pinza.

La parte superior de la base ha sido modificada para poder añadir la deslizadera previamente descrita en el apartado 3.3.1 en Figura 19.

La Figura 29 muestra el resultado final de la pieza.

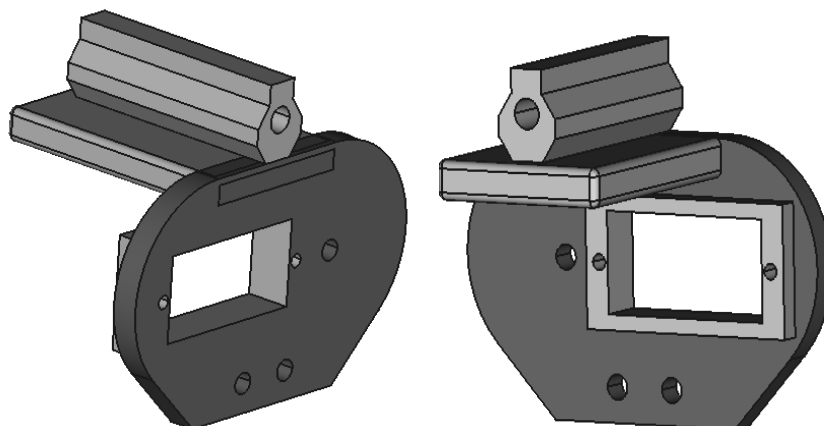


Figura 29. Vista de la base de la pinza completa.

El engranaje izquierdo del sistema de agarre ha sido modificado añadiéndole una hendidura con la forma de uno de los “cuernos” incluidos con el servo para poder transmitir el movimiento al dedo izquierdo. Las dimensiones de la hendidura en el engranaje y el resultado final es el siguiente (Figura 30):

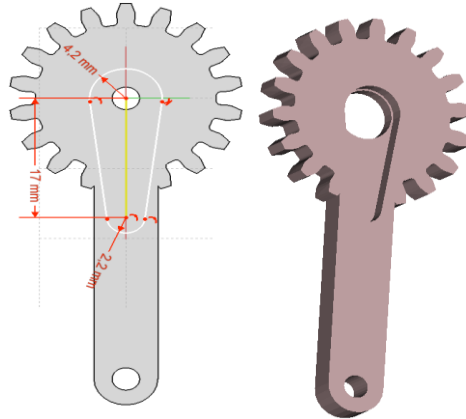


Figura 30. Engranaje con hendidura.

Los elementos de unión entre la superficie de agarre de los dedos, los engranajes y la base de la pinza no han sido modificados.

El módulo de este engranaje, siguiendo la ecuación 1. es:

$$m = 1,35 \text{ mm}$$

Los elementos de unión (Figura 31) entre la superficie de agarre de los dedos, los engranajes y la base de la pinza no han sido modificadas.

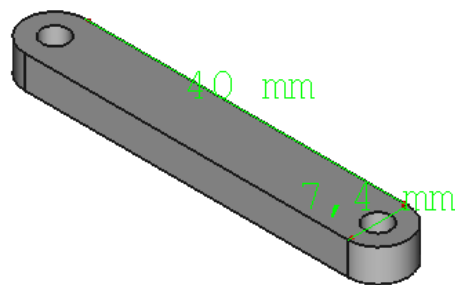


Figura 31. Pieza de unión

Para comprender mejor el montaje de la pinza a continuación se muestra la colocación de las piezas que forman parte de esta.

El sistema de agarre se ha modificado de forma que la superficie de contacto con los objetos que levante sea completamente lisa y se han quitado dos ganchos que se encontraban en la parte posterior de la pinza donde se sujetaría la goma en el modelo original.

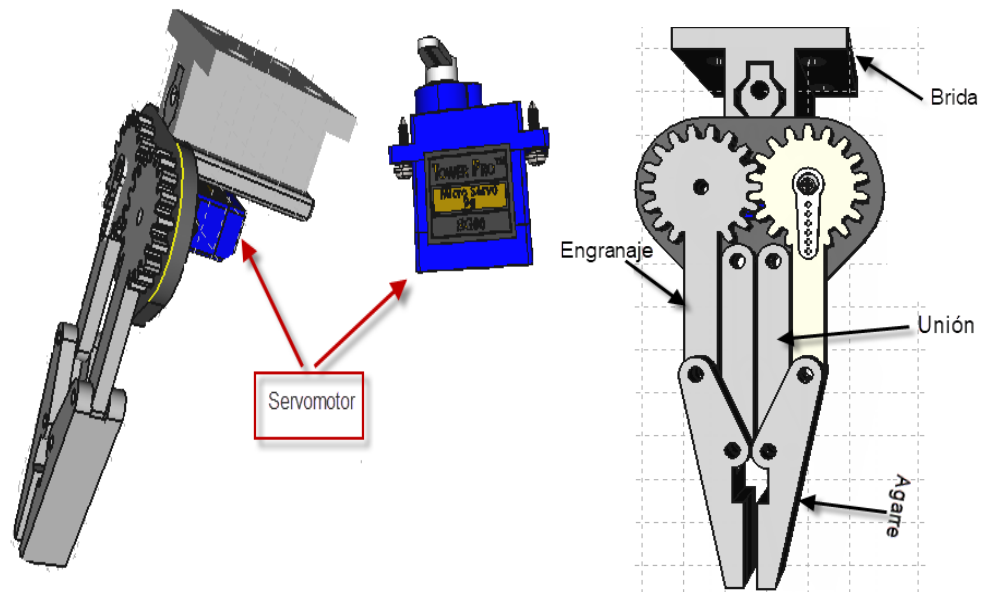


Figura 32. Pieza final junto a servomotor.

Tras el ensamblaje final de la pieza impresa el resultado es el siguiente:

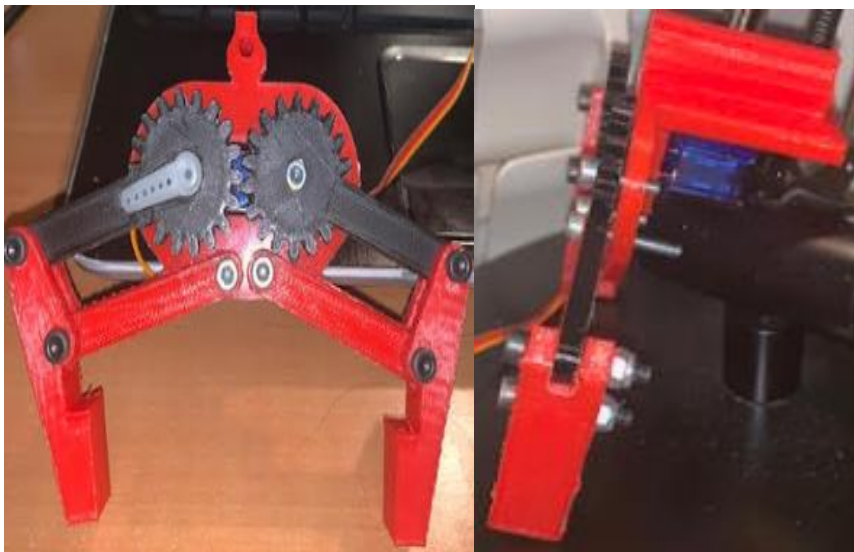


Figura 33. Ensamblaje final de la pinza

Esta pinza está accionada por un servomotor modelo SG90 de la marca “TowerPro” cuyas especificaciones se encuentran en la Tabla 7.

Puesto que el servo solo necesita una fuente de tensión, una toma de tierra y un pin PWM se puede conectar directamente a la placa Arduino Uno para su funcionamiento.

Una vez realizadas las pruebas de funcionamiento se ha observado que la superficie no proporciona el agarre suficiente por lo que se han añadido unas almohadillas de espuma para mejorar el grado de fricción entre las superficies.

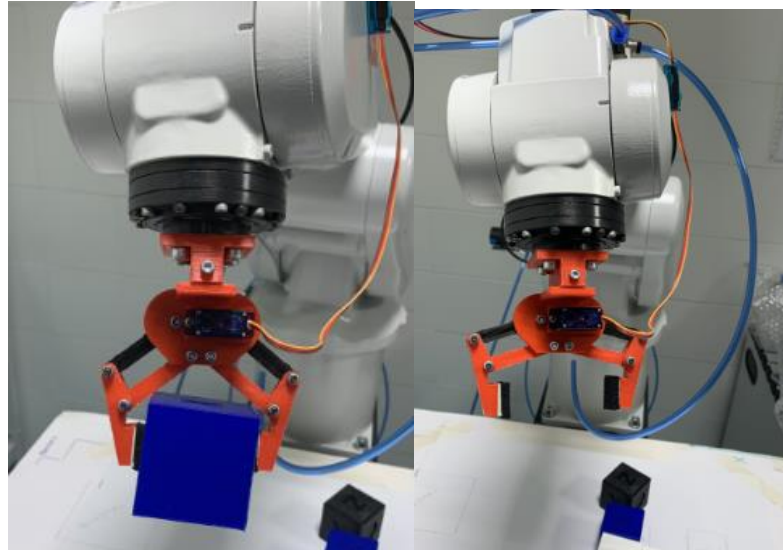


Figura 34. Pinza con la solución para mejorar la fracción entre las piezas.

3.4. Electrónica de potencia.

La necesidad de una electrónica de potencia viene dada por la incompatibilidad de niveles de voltaje entre el Arduino UNO y el controlador del brazo robótico IRC5 y por la necesidad de hacer funcionar el motor paso a paso a través de la placa CNC *shield* ya que el Arduino solo puede dar una corriente de 20 mA por pin.

3.4.1. Electrónica de potencia del motor paso a paso.

La electrónica de potencia utilizada en el motor paso a paso está conformada por una placa CNC *shield*, una fuente externa de alimentación de 12V y el controlador A4988. Las características de la CNC *shield* y el controlador se encuentran en el apartado 3.7.2.

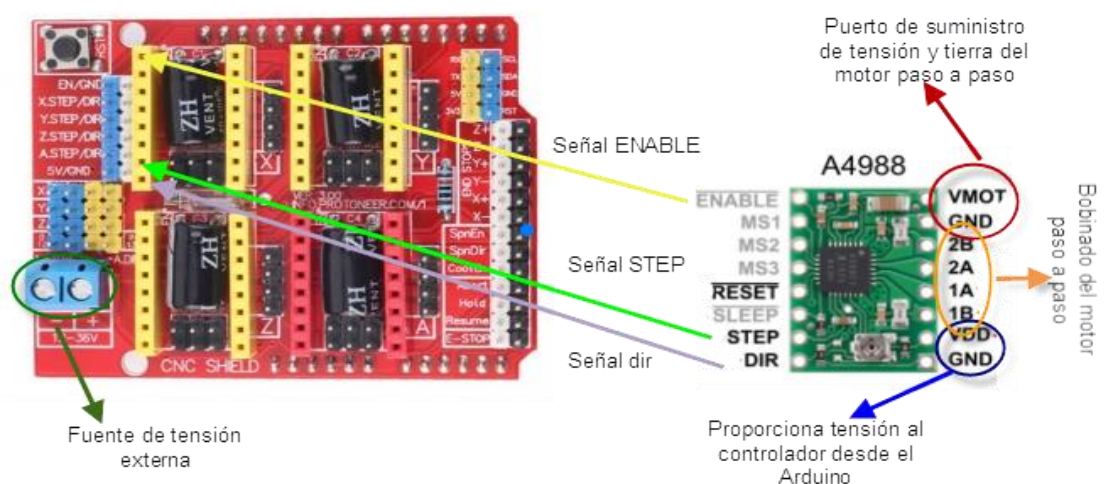


Figura 35. Esquema de las conexiones del CNC y el controlador

A través de la CNC *shield* el controlador recibe las señales digitales de *step*, *enable* y *dir* por los pines homónimos. Por otro lado, el controlador se conecta a la fuente de alimentación de 5V y el GND de la placa de Arduino por los pines VDD y GND.

Mediante los puertos de alimentación y tierra de la placa CNC *shield* se alimenta al motor con una fuente externa de 12V por los pines VMOT y GND del controlador. Cada terminal de los controladores cuenta con un condensador de 100 μ F.

El motor se conecta a los pines 1A, 1B, 2A y 2B correspondientes a cada bobinado. La disposición de estos pines y el bobinado afectará a la dirección del giro invirtiéndola si se modifica el orden de conexión de los pines.

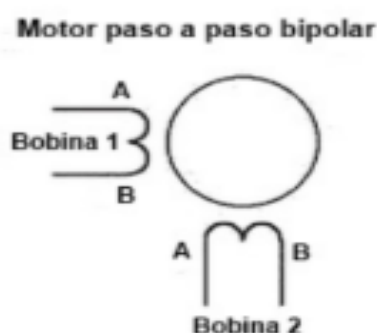


Figura 36. Esquema del bobinado del motor paso a paso

Finalmente, los pines MS0, MS1 y MS2 se utilizan para ajustar la resolución de los pasos del motor. Estos se conectan del controlador a la CNC *shield* a través de unos “jumpers”.

La resolución de los pasos escogida es de $\frac{1}{2}$ de pasos por lo que solo se ha conectado el primer pin como se muestra en la Tabla 4.

Tabla 4. Configuración de micropasos

MODE 0	MODE 1	MODE 2	Resolución de micropasos
Low	Low	Low	Pasos enteros
High	Low	Low	Medios pasos
Low	High	Low	1/4 de pasos
High	High	Low	1/8 de pasos
High	High	High	1/16 de pasos

Tras conectar el controlador y antes de empezar a trabajar con él, el controlador debe calibrarse para limitar la corriente del motor, así como hacer que los motores funcionen de manera óptima.

Conociendo la intensidad máxima del motor paso a paso a partir de los datos del fabricante, es de 600 mA, que la corriente debe limitarse a un 70% del total en caso de usar pasos enteros y que las resistencias R100 del controlador son de $0,1 \Omega$ es posible calcular el voltaje según la siguiente ecuación que debe pasar por el controlador y ajustar el potenciómetro:

$$V_{ref} = I_{m\acute{a}x} \cdot 8 \cdot R_s = 530 \cdot 10^{-3} A \cdot 8 \cdot 0,1 \Omega = 0,424 V$$

3.3.2. Electrónica de potencia para la conexión con el Robot IRB120.

Debido a que el controlador del robot IRC5 trabaja a una tensión de 24V y el microcontrolador Arduino Uno a 5V es necesario establecer una configuración electrónica que permita resolver la diferencia de tensión para permitir la comunicación entre el IRC5 y Arduino Uno.

Para conseguir evitar esta diferencia de tensión se ha escogido utilizar un optoacoplador, que es capaz de aislar eléctricamente el Arduino y el controlador del brazo robótico y así evitar que los distintos componentes se quemen por tensiones excesivamente altas.

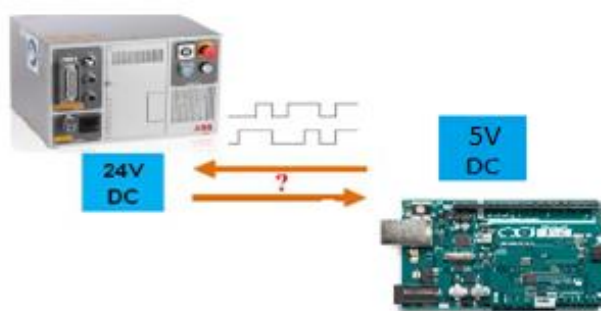


Figura 37. Incompatibilidad de voltajes.

El optoacoplador escogido es uno modelo 4N35 cuyas características se explican en el apartado 3.7.2.9.

La conversión debe realizarse de los 24V aportados por el controlador del robot a los 5V que soporta el Arduino UNO quedando el montaje de la siguiente manera:

Tabla 5. Configuración del circuito

Entrada	0-24V (IRC5)
Salida	0-5V (Arduino UNO)
Resistencia de entrada	1 k Ω
Resistencia de Pull down	10 k Ω
GND del diodo	GND del IRC5
GND del fototransistor	GND del Arduino
Intensidad de entrada	20 mA

El diseño esquemático del circuito necesario para la conexión entre el robot y el Arduino dibujado en KiCAD sería el siguiente:

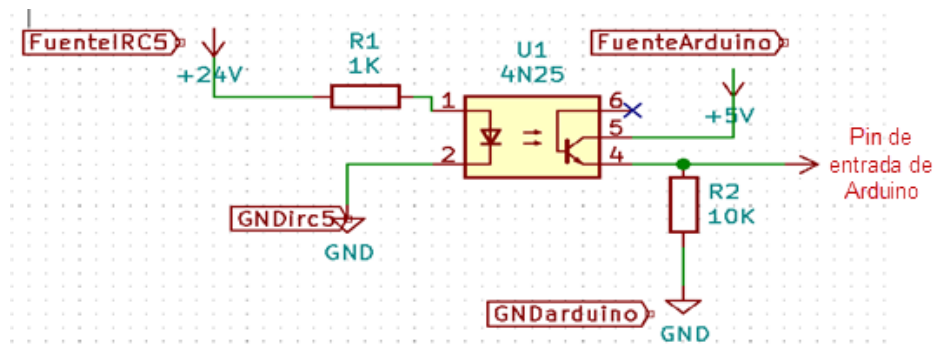


Figura 38. Circuito esquemático.

Los componentes electrónicos se han conectado a una placa mediante soldadura de estaño según el siguiente modelo creado en la aplicación DiYLC (Apartado 3.7.1.3) que además sirvió de guía durante la soldadura. También se coloca en esta placa el potenciómetro que se usará junto con el servo.

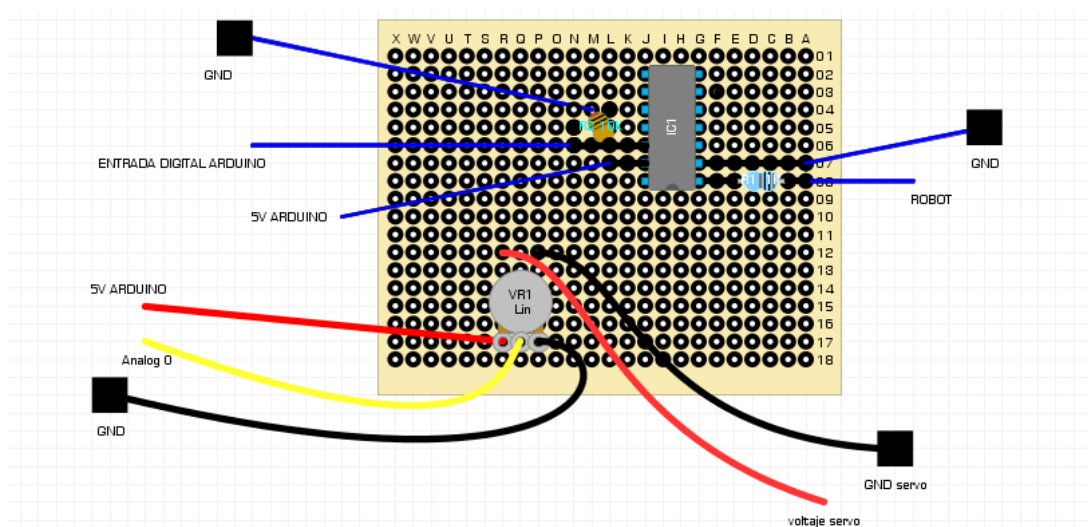


Figura 39. Disposición de los componentes electrónicos.

Como se puede observar el zócalo que se ha colocado es de 14 pines para permitir el acople de otro optoacoplador si en el futuro se quiere añadir algún elemento que se comunique desde el Arduino al IRC5 como puede ser un sensor de posición que provoque la parada del robot al encontrarse algún elemento en su camino.

El potenciómetro se suelda a la placa para poder tener todos los elementos de electrónica en un mismo sitio y así aprovechar el mínimo espacio del controlador del brazo robótico.

3.5. Programación en Arduino.

Para la programación de los motores se ha utilizado el *software* de Arduino “Arduino IDE” que está explicado en el apartado 3.7.1.1.

En primer lugar, se programó el motor paso a paso para que en función de la señal de entrada que recibiera el Arduino se abriera o se cerrara.

En segundo lugar, se hizo un programa con un funcionamiento similar, pero modificado para la utilización de un servo y la posibilidad de controlar la apertura y cierre del servo mediante el giro de un potenciómetro.

La señal de entrada es una señal digital enviada desde el controlador del brazo robótico hasta el circuito con el optoacoplador. Una vez esa señal es enviada se cierra el circuito interno del optoacoplador permitiendo que la entrada del Arduino reciba un 1 lógico, activandose el movimiento de la pinza.

El esquema del proceso del funcionamiento de ambos motores es el siguiente:

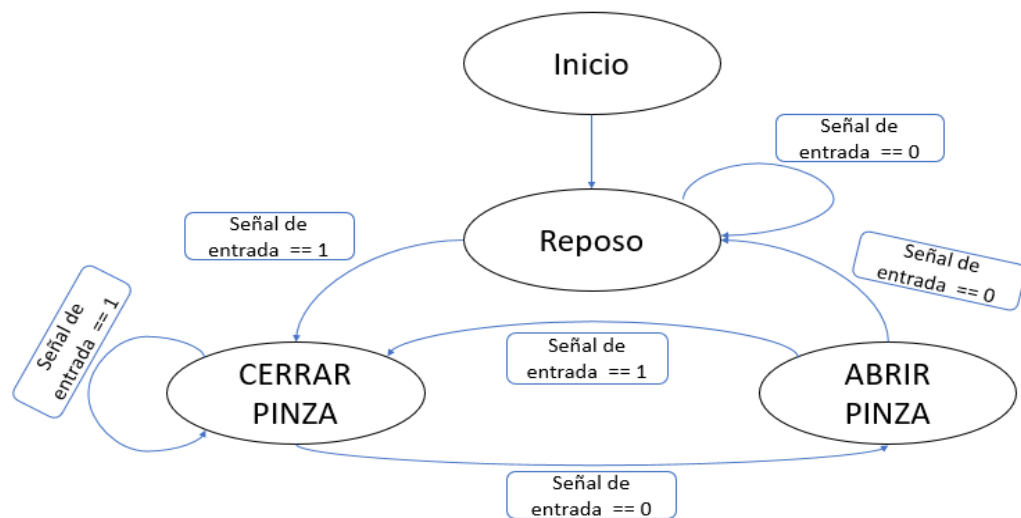


Diagrama 1. Esquemático del funcionamiento de la pinza

Este diagrama coincide para ambos casos realizando una serie de modificaciones que se explican a continuación.

3.5.1. Programación del motor paso a paso.

Como se ha explicado anteriormente (3.2.1), el motor paso a paso tiene dos pares de cables y cada par corresponde a una de sus bobinas. La secuencia que se envía a las bobinas permite el movimiento del motor, secuencias que se generan mediante el *Pololu* a través de las señales enviadas por el Arduino.

El giro del motor se controla mediante la señal STEP, que es una serie periódica (se enciende y apaga) según un valor temporal que se programe. Por cada pulso enviado el motor se mueve un paso.

Mediante la señal DIR se marca la dirección en la que gira y la señal ENABLE que habilita o inactiva el controlador. Esta última entrada está negada en el modelo del controlador usado, por lo que para que el controlador se habilite hay que poner un cero lógico.

- **Código de Arduino motor paso a paso:**

Para abrir la pinza el código entra en una función llamada “apertura” donde se le dice al motor que gire en sentido antihorario y dé una serie de pasos que se miden mediante un contador. De esta manera el motor termina de moverse cuando finaliza la cuenta.

Al contrario, para cerrar la pinza, el motor está configurado para girar en sentido horario y al igual que en la función de apertura hay un contador que determina cuantos pasos debe dar el motor hasta que finalice su movimiento.

Se puede ver la explicación en detalle del código junto con su código en el apéndice 6.1.1 de este TFG.

3.5.2. Programación del servomotor.

Como se ha explicado anteriormente el cambio de motor no solo obligó a realizar un nuevo diseño de la pinza, sino que también llevó a la creación de un nuevo programa de control del servomotor.

El servomotor se controla a través de una señal modulada por ancho de pulso o PWM por sus siglas en inglés. El PWM es una señal periódica que determina el ciclo de trabajo de un sistema electrónico como pueden ser un motor o un LED.

La función de la señal del PWM consiste en una onda cuadrada de valores 0 o 5V y que durante un periodo de tiempo de 20 ms, en caso del servo motor usado, (la frecuencia es de 50 Hz) la señal se puede encontrar en alto (activa) o en bajo (inactiva).

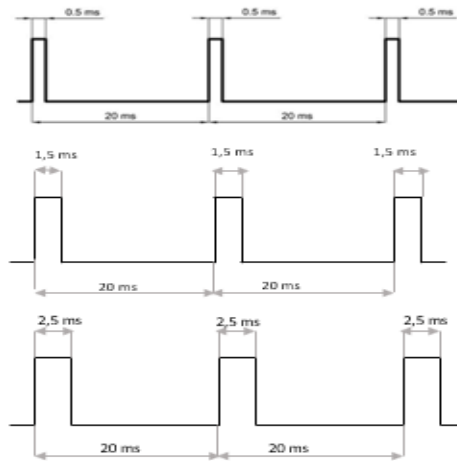


Figura 40. Modulación de los pulsos para las posiciones del servomotor SG90 de 0°, 90° y 180°.

El tiempo que se encuentra en el nivel alto se denomina “Ton” y es el que determina el ángulo de giro del motor.

El ángulo de giro del servo motor durante el proceso de cierre se puede modificar según un potenciómetro. El ángulo de apertura se fijará en 180° grados.

- **Código de Arduino servomotor:**

Para la programación del servo se ha utilizado la biblioteca disponible en Arduino IDE.

Arduino debe mapear el ángulo girado por el potenciómetro para que el motor se mueva un cierto ángulo.

El mapeo convierte el rango de valores de giro del potenciómetro (rango que va desde cero hasta 1023) al ángulo de giro del servomotor (rango desde cero hasta 180°).

Cuando se mapea ese valor se pasa a las funciones de apertura y cierre donde el valor límite de cierre de la pinza es el que se ha seleccionado con el potenciómetro.

Se puede encontrar la explicación con detalle en el apéndice 6.1.2

3.6. Montaje final en el robot.

Para poder llevar a cabo las conexiones y configuraciones necesarias para el montaje de la pinza del servomotor en el robot y pasar a comprobar el funcionamiento final de este, es necesario realizar una serie de pasos, tanto directamente con el robot como a través del programa RobotStudio, que se explican en los siguientes apartados.

3.6.1. Simulación en RobotStudio.

El *software* RobotStudio permite la simulación del comportamiento de las herramientas utilizadas y de las rutinas creadas para el funcionamiento del robot sin riesgo de provocar un accidente o algún daño en el robot.

Desde el *software* se puede crear la estación de trabajo con el robot y el controlador que se utilizará a la hora de hacer la simulación, así como la pinza que se ha diseñado previamente con un programa CAD. También se ha creado un cubo de pequeñas dimensiones que hace las veces de cubo de muestra de los que se usarán en las prácticas de laboratorio de la asignatura.

Al crear la herramienta se han configurado como un mecanismo con una serie de parámetros físicos como son: el peso de la pinza, su TCP (este término se explica un poco más adelante), los eslabones que forman el mecanismo y las acciones que realizarán estos eslabones como el recorrido que deben hacer al abrir y al cerrarse.

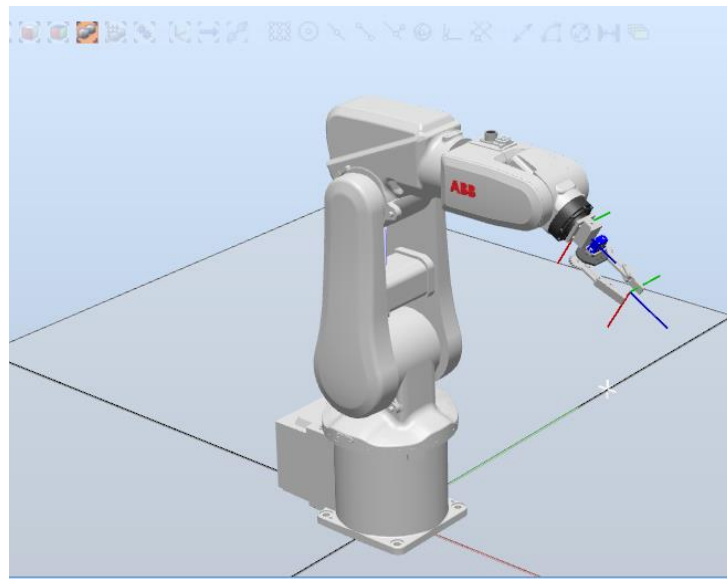


Figura 41. Estación con Robot ABB120 y pinza servomotor.

Desde esta estación de trabajo se pueden definir unos puntos en el espacio por los que debe pasar el TCP o “*Tool Center Point*” (Punto Central de la herramienta) de la pinza.

Tras definir los puntos de la trayectoria se ha de configurar la señal de salida que determina cuando se abre o se cierra la pinza y las acciones de abrir y cerrar la pinza.

Una vez se han definido los puntos y las señales digitales, se genera una trayectoria determinando la velocidad con la que la pinza se debe de aproximar hasta el objeto a coger, así como, el radio de tolerancia que se quiere usar para coger el objeto o dejarlo en un punto determinado. En el caso de la trayectoria definida no se ha permitido tolerancia alguna.

Otro parámetro que se debe configurar es el del tipo de movimiento, pudiendo ser lineal o “*Joint*” (de ejes). El movimiento lineal hace que el robot se tenga que mover obligatoriamente en la dirección y sentido que se han marcado generando una línea recta entre el punto de origen y el de destino mientras que el movimiento “*joint*” permite un movimiento más libre del robot para ir desde el punto inicia hasta el final.

Para realizar las pruebas de funcionamiento de la pinza se han creado los siguientes puntos:

- HOME: Posición de inicio al arranque del programa.
- Punto1: Punto de aproximación al objeto que debe recoger.
- Punto2: Punto en el que se encuentra el centro de gravedad del cubo que se debe recoger.
- Punto3_int: Es un punto intermedio en el espacio que hay entre el punto de origen y el punto del que se ha recogido el cubo y donde el robot hace un cambio de trayectoria
- Punto4: Punto de aproximación al destino donde se suelta el cubo.
- Puntofinal: Punto en el que se coloca el cubo.

La trayectoria descrita se encuentra en el apéndice 6.2.1 junto con el programa generado en el entorno *RAPID*. Este programa junto con la estación de simulación se puede encontrar también en el GitHub del autor.

3.6.2. Conexiones con Robot ABB IRB 120.

Es necesario hacer una serie de conexiones entre el controlador IRC5 Compact (Apartado 3.7.2.8) y la placa soldada, y entre el Arduino, el robot y la pinza.

En primer lugar, para conectar el controlador a la placa soldada, se deben asignar una entrada y una salida [36] para enviar la señal digital. Estas entradas y salidas se encuentran en la unidad de programación según la siguiente imagen Figura 42.

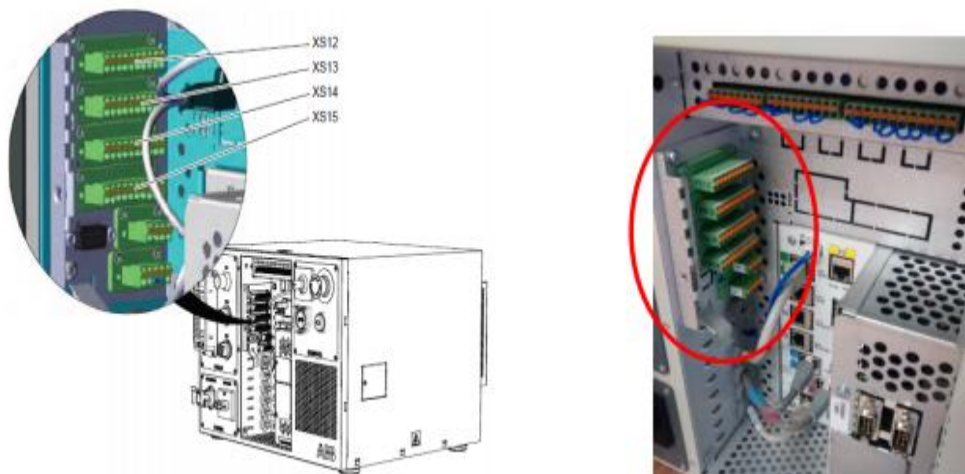


Figura 42. Puertos de entrada y de salida del IRC5

Los conectores mostrados en la imagen anterior, llamados XS, se asocian a la tarjeta de entradas y salidas digitales DSQC 652 de tipo DeviceNet. Este controlador cuenta con 16 entradas y 16 salidas a +24V DC que se distribuyen:

- XS12: Entradas (CH1/CH8)

- XS13: Entradas (CH9/CH16)
- XS14: Salidas (CH1/CH8)
- XS15: Salidas (CH9/CH16)
- XS16: Fuente de alimentación interna (+24V DC)

El controlador está configurado de forma que los conectores XS14 y XS15 en el pin 9 la tensión es de 0V y el pin 10 tiene +24V DC, sin embargo, no tiene tensión (Figura 43). Es necesario conectar una fuente a este pin. Esta fuente puede llegar desde el conector XS16 (si la corriente necesaria es inferior o igual a 6A) o a una fuente externa de 24V (si la potencia o corriente necesarias son mayores).

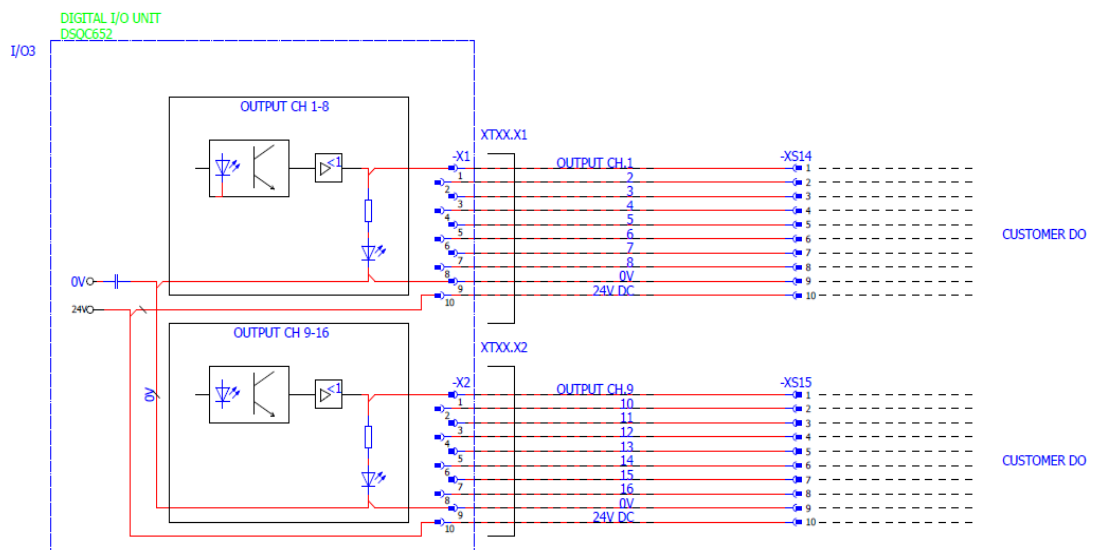


Figura 43. Esquema de configuración de los puertos XS14 y XS15.

Puesto que para este caso no es necesario utilizar una corriente mayor a 6A se usará el conector XS16 con la siguiente configuración.

- El pin 10 de los conectores X14 o X15 se conectan al pin 1 o al pin 3 del XS16 que tienen un voltaje de +24V. Solo es necesario hacer la conexión con uno de los conectores ya que estos están conectados internamente entre sí.
- El pin 9 de XS14 o XS15 se conecta a los pines 2 o 4 de XS16 conectados a tierra para así cerrar el circuito.

El esquema de las conexiones del puerto XS16 se puede ver en la Figura 44

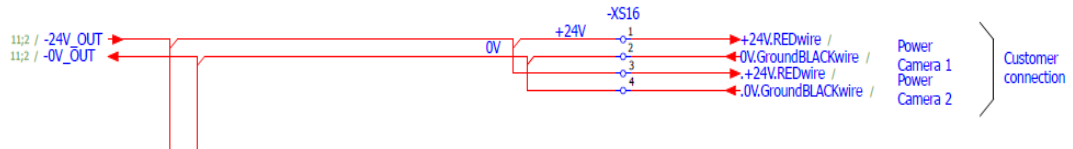


Figura 44. Esquema de configuración del puerto XS16.

Estando la electroválvula de la pinza neumática conectada al pin 2 del conector XS16 para la entrada y el pin 8 del conector XS15, se ha escogido conectar los pines 8 del XS14 y 4 del XS16.

Tras identificar los puertos de entrada y de salida que se van a utilizar se pasa al montaje y cableado de la placa de soldadura, el Arduino y la pinza del servomotor.

En primer lugar, se ha diseñado en FreeCAD una pieza en forma de estantería donde se colocarán tanto la placa de soldadura como el Arduino. Esta estantería está diseñada para ocupar el menor espacio posible que quepan las placas y una obertura por la que puedan pasar los cables que sean necesarios.



Figura 45. Controlador IRC5 Compact, donde se señala la situación de la placa

El diseño realizado es el siguiente:

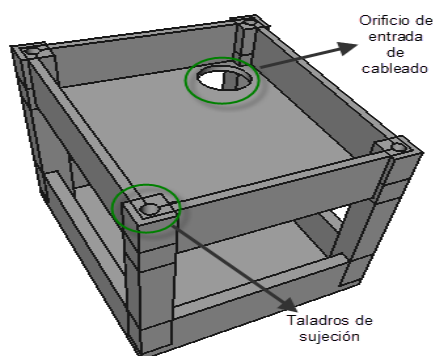


Figura 46. Estantería porta-placa

La estantería diseñada (Figura 46) cuenta con dos orificios por el que pasan los cables de una placa a otra y una serie de taladros para atornillar la parte superior e inferior y que la impresión sea más sencilla.

En segundo lugar, se conectan los pines explicados con anterioridad al optoacoplador por la parte en la que se encuentra el LED. El pin de salida 8 de XS14 se conecta a la entrada del circuito con el LED mientras que la salida se conecta al pin 4 del conector XS16.

Una vez se ha conectado, se pasa a conectar los cables que unen la placa de soldadura al Arduino según el esquema que se puede ver en la Figura 39.

Tras unir las dos placas, se hace la conexión del Arduino con la pinza y el potenciómetro.

El robot cuenta con una serie de canalizaciones internas que permite realizar las conexiones con el controlador o con elementos externos al robot evitando que el cableado se retuerza entre sí, que se enchanche en los ejes de giro y, en definitiva, que impida el correcto funcionamiento del robot.

Una vez se ha hecho este paso, se conecta en la parte superior del robot el cableado de la placa a la pinza.

3.7. Materiales.

A continuación, se exponen los materiales utilizados a lo largo del desarrollo de este TFG.

Este apartado se divide en las herramientas de software que se han utilizado para el diseño y control de los elementos involucrados en el trabajo y las herramientas de hardware o distintos componentes que forman parte de este.

3.7.1. Software.

3.7.1.1. Arduino IDE.

El entorno de programación más usado de las placas Arduino [37] es la aplicación multiplataforma (puede usarse en Windows, macOS y Linux) Arduino IDE las siglas en inglés de entorno de desarrollo integrado.

El lenguaje de programación del *software* de Arduino es C y C++ mediante el uso de reglas especiales de estructuración de código.

El código escrito por el usuario necesita de sólo dos funciones básicas para funcionar, una donde se inicia el “sketch” o código y el bucle principal del programa.

Arduino IDE utiliza un programa para convertir el archivo de texto en un código ejecutable de codificación hexadecimal que se carga a la placa de Arduino por un programa de carga que se

encuentra en el *firmware* de la placa. Cuenta con librería de *software* que proporciona operaciones comunes de entrada/salida.

3.7.1.2. Repetier.

Es una aplicación [38] que permite el análisis y la configuración de archivos “.stl” para su impresión en 3D. Este archivo “.stl” se genera a través del diseño generado en FreeCAD.

Para llevar a cabo la impresión, el *software* genera un código en formato “.gcode”, serie de comandos para la generación de las piezas a partir de los archivos “.stl” que se cargan en el programa y que, posteriormente, se envía a la impresora.

Repetier permite colocar los objetos de impresión según se quiera sobre el área de impresión y configurar los parámetros para la impresión como son la temperatura, la velocidad, la densidad del mallado y el patrón. Esta configuración previa de los parámetros puede evitar en la medida de lo posible fallos en la impresión que generarían aumento en los costes y en el tiempo de producción de la pinza.

Esta aplicación se encuentra instalada en el ordenador del laboratorio perteneciente al Área de Tecnología electrónica conectado a la impresora 3D con la que se fabrican las herramientas.

3.7.1.3. DiYLC.

Es un software [39] de dibujo desarrollado por una amplia comunidad de entusiastas de la electrónica y de la filosofía DIY “*Do it yourself*” (Hazlo tú mismo).

El objetivo es ofrecer una interfaz simple con potencia suficiente para permitir al usuario dibujar circuitos esquemáticos, esquemas de placas y diagramas de cableados de guitarras rápidamente y que no es difícil de aprender.

Se ha utilizado para el dibujo del mapa de los componentes electrónicos en la placa de inserción.

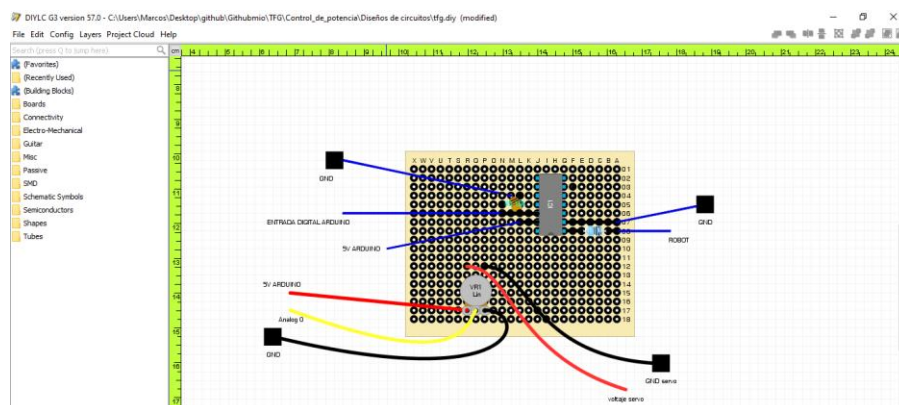


Figura 47. Interfaz de DiYLC

3.7.1.4. KiCAD.

Se trata de un paquete de *software* libre [40] que permite diseñar circuitos electrónicos de forma automática.

El programa permite el diseño de circuitos esquemáticos y su posterior conversión a una PCB o placa de circuito impreso. La herramienta de diseño incluye la posibilidad de crear listas de materiales, ilustraciones, archivos “Gerber” (archivos que contienen información para la generación de circuitos impresos) y vistas 3D de la PCB y sus componentes.

Se ha usado para el dibujo de los distintos circuitos esquemáticos mostrados en este TFG.

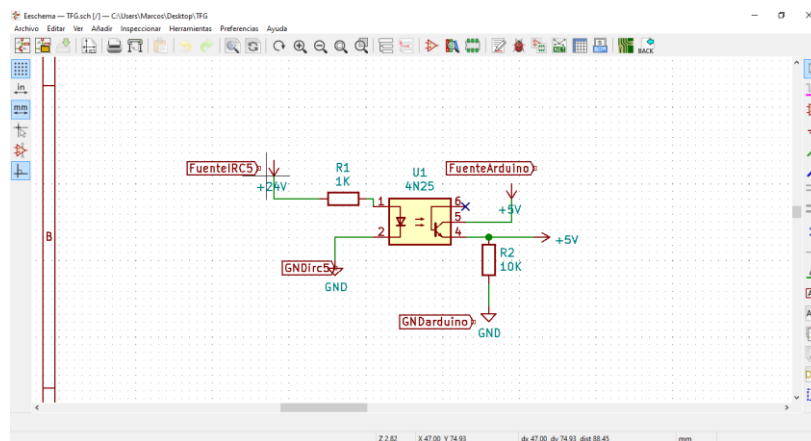


Figura 48. Entorno de dibujo de circuitos esquemáticos de KiCAD.

3.7.1.5. FreeCAD

Es un programa de diseño [41] asistido por ordenador de código abierto similar a AutoCAD, SolidWorks o CATIA que permite la creación de objetos en tres dimensiones.

Usa un modelado en 3D paramétrico que permite la edición del diseño de forma sencilla mediante la posibilidad de cambiar los parámetros del dibujo a través del historial de modelado.

Permite la creación de formas en dos dimensiones y usarlas como base para la creación de los modelos en 3D y a la vez permite extraer detalles de diseño de modelos 3D para crear dibujos en dos dimensiones de alta calidad.

Es un “*software*” flexible y accesible ya que está integrado en múltiples plataformas, es personalizable y permite leer un gran número de formatos de archivo.

Cuenta con numerosos entornos de trabajo con herramientas específicas para el desarrollo de distintas tareas. Al cambiar de un entorno a otro las herramientas de diseño varían entre sí. También incluye la posibilidad de realizar los diseños a través del lenguaje de programación Python.

Esta herramienta se ha usado en la creación de los distintos modelos de útiles que se han presentado en los apartados 3.1 y 3.3.

3.7.1.6. *GitHub*.

GitHub es una plataforma colaborativa en la nube [42] de desarrollo de *software* que permite el intercambio de proyectos, un control de versiones e información de manera gratuita y rápida.

Esta plataforma cuenta en la actualidad con más de 36 millones de usuarios y más de 100 millones de repositorios almacenados a los que se puede acceder a ellos de forma gratuita.

Se creó con el objetivo de trabajar en equipo dentro de un proyecto y la posibilidad de contribuir a mejorar el trabajo de los demás a través de las distintas herramientas que hay disponibles.

GitHub ofrece distintas herramientas para facilitar el seguimiento del avance de los proyectos, así como una biblioteca donde ver los cambios realizados. Estas son:

- Sistema de seguimiento de problemas: Permite a los usuarios que trabajen en un mismo proyecto aportar sugerencias o explicar problemas que se encuentren en el “*software*”.
- Revisión de código: Permite comentar puntos concretos de los ficheros y discutir los cambios que se han producido en una nueva entrada.
- Visor de ramas: Se pueden comparar los registros de los procesos que se han hecho en las distintas ramas del repositorio en la que se encuentre el trabajo.

“*GitHub*” usa “*Git*” para subir y organizar todas las versiones de los documentos que se cargan en la plataforma.

“*Git*” [43] es un sistema de control de versiones que se utiliza para controlar desde pequeños proyectos hasta los más grandes con velocidad y eficiencia.

Este sistema tiene una funcionalidad que permite separar los proyectos en distintas ramas que pueden ser completamente independientes las unas de las otras. La creación, unión, y borrado de estas líneas puede realizarse en unos pocos segundos.

La posibilidad de tener varias ramas de trabajo permite diversas acciones como:

- Cambio de contexto libre: Se puede crear una rama en la que probar un proyecto, añadir aportes, volver a la rama de partida, aplicar parches, volver a la rama donde se experimenta y unir las dos ramas.

- Separación basados en roles: Permite separar las ramas en función del uso que se le vaya a dar, por ejemplo, una que pase a productivo, otra para experimentar y otra para pequeños proyectos.
- Flujos de trabajo basados en funcionalidades: Separa las ramas de trabajo según se vayan desarrollando las distintas partes del trabajo para posteriormente unir las todas en la principal y borrar las secundarias.
- Ramas de experimentación: Se usan única y exclusivamente para hacer pruebas con los proyectos realizados.

El sistema de control de versiones es rápido ya que la gran mayoría de las operaciones se realizan de manera local teniendo así los sistemas centralizados.

Finalmente, al igual que GitHub, “Git” es gratuito y de código abierto.

Los resultados obtenidos y avances realizados durante el desarrollo de este TFG pueden verse en el GitHub del autor en el siguiente enlace <https://github.com/Marcos-vp/TFG>

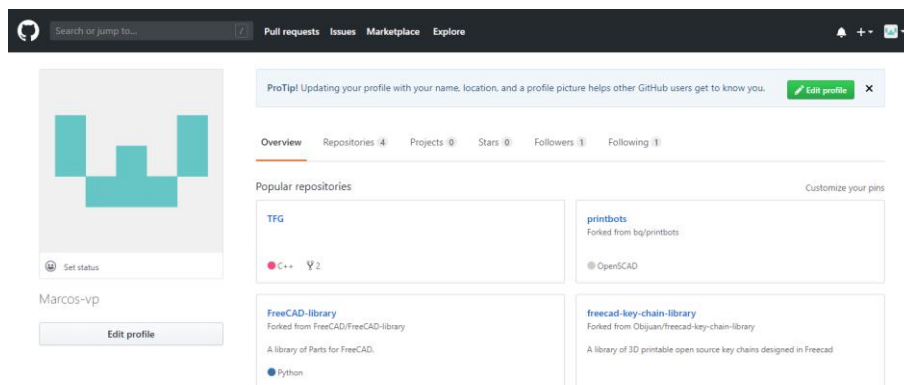


Figura 49. Interfaz del perfil del autor del TFG en GitHub.

3.7.1.7. RobotStudio.

RobotStudio [44] es el programa informático desarrollado por el fabricante del brazo robótico ABB que permite la programación fuera de línea y la simulación.

Desde el programa se pueden configurar los movimientos y rutinas que ejecutará el robot y que este se mueva de forma automática o utilizar un controlador llamado “Flexpendant” para controlarlo de manera manual.

En este TFG se usa el programa para simular el comportamiento del robot junto con la pinza desarrollada y posteriormente comprobar el funcionamiento real de ambas partes. A su vez, es necesario para configurar el sistema de comunicación entre el Arduino y el brazo robótico para poder activar el actuador en el momento requerido.

Esta comunicación se apoya en la electrónica de potencia que se explica en el apartado 3.3.2


3.7.2. Hardware.

En este apartado se presentan los componentes físicos que han intervenido durante el desarrollo del TFG desde el manipulador o brazo robótico hasta los componentes utilizados para llevar a cabo la electrónica de potencia en la comunicación entre el robot y microcontrolador Arduino.

3.7.2.1. Arduino.

Arduino es una plataforma electrónica de “*open source*” que se basa en *hardware* y *software*. Las placas de Arduino son placas electrónicas basadas en microcontroladores.

La placa escogida para este proyecto es la Arduino UNO debido a su pequeño tamaño y a que tiene las prestaciones suficientes para controlar el actuador de las herramientas que se puedan usar con el brazo robótico, así como la capacidad para controlar distintos sensores en caso de querer enviarle información de su entorno al manipulador. Las características técnicas del microcontrolador [45] son las siguientes:

Características técnicas Arduino UNO		
	Microcontrolador	ATmega328P
	Voltaje de operación	5V
	Voltaje de entrada (recomendado)	7-12V
	Voltaje de entrada (límites)	6-20V
	Pines digitales I/O	14
	Pines digitales PWM I/O	6
	Pines analógicos de entrada	6
	Corriente DC por pin I/O	20 mA
	Peso	25 g
	Memoria Flash	32 Kb
	SRAM	2 Kb
	EEPROM	1 Kb
	Reloj	16 MHz
	Dimensiones	68,6 x 53,4 mm

La placa puede alimentarse de diversas maneras como son el puerto USB, a través del pin “Vin” o a través de la entrada de potencia. El programa se carga en la memoria de código a través del terminal USB mediante el “*sketch*” generado en el “*software*” del fabricante Arduino IDE.

Algunos de los componentes usados en este TFG que conforman la placa son los siguientes:

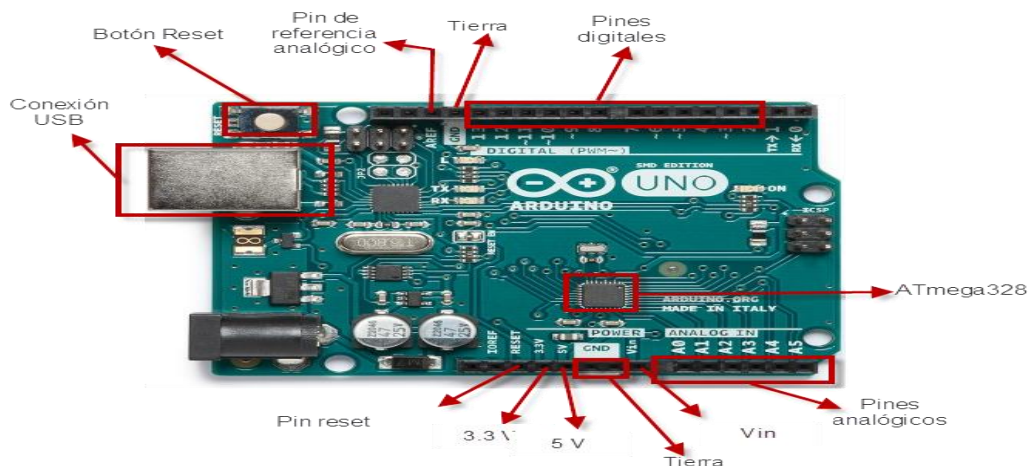


Figura 50. Placa Arduino UNO.

3.7.2.2. CNC Shield.

La placa Arduino CNC Shield ha sido utilizada para el control del motor paso a paso de la pinza diseñada en el apartado 3.3.1 y suele utilizarse en proyectos en los que hay que controlar una cortadora láser o máquinas de fresado.

Esta placa está diseñada para acoplar [46] hasta cuatro motores paso a paso y soporta cuatro controladores de potencia Pololu DRV8825 o A4988. Esta placa permite también el uso del firmware GRBL el cual debe instalarse en el Arduino previo a su utilización para realizar la programación y el control de los motores, aunque no se ha usado en este proyecto.

La CNC shield cuenta con 3 pares de pines que se conectan al controlador *Pololu A4988* mediante unos cierres llamados “Jumpers” en cada terminal que permite fraccionar el número de pasos que recorre el motor. (Figura 51).

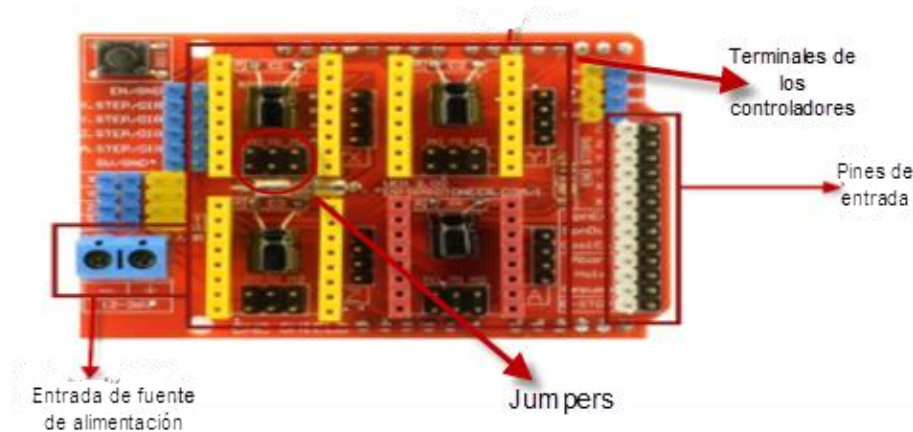


Figura 51. CNC shield.

Para programar el control del motor paso a paso es necesario conocer a qué pines corresponden cada terminal en la placa Arduino UNO, para conocer esto se puede ver la continuidad que hay entre cada pin o ver una hoja técnica de referencia donde vienen indicada la equivalencia de los pines.

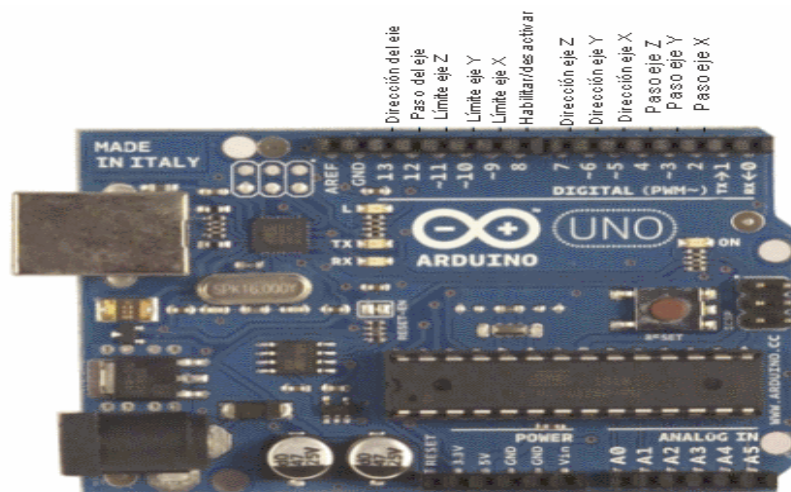


Figura 52. Esquema de los pines requeridos del Arduino UNO respecto de la CNC.

3.7.2.3. Controlador Pololu A4988.

El Pololu A4988 es un controlador de potencia [47] para los motores paso a paso, es decir, limita la corriente que circula a través del motor mediante un potenciómetro integrado. Genera las señales necesarias para que el motor funcione y proporciona protección frente a la temperatura y la corriente.

El controlador cuenta con protección térmica y de sobrecarga, sin embargo, es recomendable colocar un disipador de calor con pasta térmica de la siguiente manera:

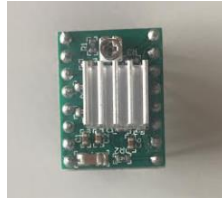


Figura 53. Controlador Pololu A4988 con disipador de calor.

Para que el conjunto del Arduino, la CNC *shield*, el controlador y el motor funcionen se deben alimentar el controlador y el motor mediante dos fuentes de alimentación independientes. Una de ellas con un voltaje de entre 3 y 5V que será para el controlador y es proporcionada por el Arduino y la otra para dar tensión al motor de 12V que se conectará a la placa CNC *shield* por el puerto de entrada como se ha explicado en el apartado 3.4.1..

El controlador del motor cuenta con dos pines para ambas alimentaciones.

Siguiendo la hoja de características técnicas del fabricante podemos saber cuál es el esquema de conexión para los motores paso a paso y cómo debe conectarse a la placa CNC *shield*.

Como se puede observar en el esquema el controlador (Figura 54) cuenta con cuatro pines a los que se les conecta las bobinas del motor paso a paso, un pin donde se aporta la tensión que solicita por el motor (*VMOT*) y las señales *STEP*, *DIR* y *ENABLE* que controlan la dirección y movimiento del motor que se conectan a la placa Arduino UNO.

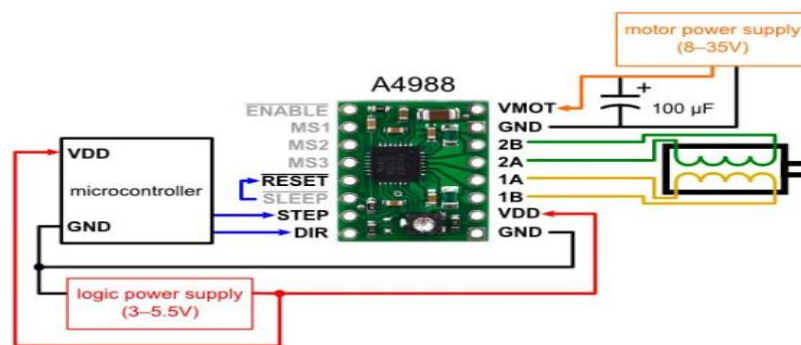


Figura 54. Cableado de conexión con el motor paso a paso y con la placa CNC *shield*.


En el caso de este TFG todo el cableado no es necesario ya que la conexión se realiza a través de la placa CNC *shield*.

Para ajustar la corriente que circula por el motor es necesario hacer un calibrado previo del controlador midiendo la caída de voltaje entre el pin *VMOT* y el pin *GND* según se especifica en la web del fabricante.

3.7.2.4. Motor paso a paso NEMA 8.

Las características técnicas que [48] se encuentran en la página del fabricante de este motor son las siguientes:

Tabla 6. Características técnicas del motor paso a paso.

Motor paso a paso NEMA 8		
Imagen	Característica	Valor
	Tamaño	20x20x30 mm sin incluir el eje
	Peso	60 g
	Diámetro del eje	4 mm en forma de "D"
	Pasos por vuelta	200
	Paso de corriente	600 mA por bobina
	Voltaje	6,5 V
	Resistencia	6.5 Ω por bobina
	Par motor	180 g·cm
	Inductancia	1.7 mH por bobina

3.7.2.5. Servomotor SG90.

El servo utilizado en este TFG es un microservo de la marca TowerPro modelo SG90 cuya hoja de datos [49] es la siguiente:

Tabla 7. Características técnicas del Servomotor

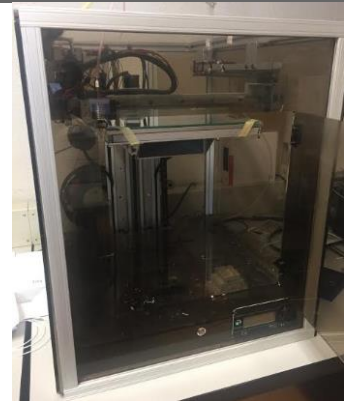
Servomotor SG90		
Imagen	Característica	Valor
	Ángulo de giro	360 grados
	Peso	9 g
	Dimensiones	23x12,x22 mm
	Par motor	1300 g·cm (4,V) – 1500 g·cm (6V)
	Velocidad de operación	110 rpm (4,8V) – 120 rpm (6V)
	Voltaje de operación	4,8V – 6V
	Rango de temperatura	0° - 55° centígrados
	Sistema de control	Digital

3.7.2.6. Impresora 3D.

Para la realización de este TFG ha sido necesario utilizar dos impresoras 3D, la primera se trata del modelo iDeator12 y la otra del modelo Ultimaker 3, esto es debido a que algunas de las piezas necesitan tener una tolerancia muy pequeña para la correcta impresión de las piezas y que estas encajen perfectamente.

La impresora iDeator 12 de tecnología FFF-FDM [50] (modelado por difusión fundida) cuenta con las siguientes características:

Tabla 8. Características técnicas de iDeator12.

Impresora 3D iDeator12		
	Características	
	Tecnología	FDM-FFF
	Materiales	PLA, ABS
	Tamaño máximo de impresión	305x305x305 mm
	Nº de extrusores	2
	Base calefactable	Sí
	Tamaño de la impresora	568x520x622 mm
	Tipos de archivo	.gcode

El proceso de impresión que sigue esta impresora es el siguiente, el material que está introducido en el extrusor es calentado a una temperatura superior a la de fusión para permitir que este fluya en forma de hilo.

Los extrusores se mueven por la impresora gracias a motores paso a paso y a correas de transmisión siguiendo las instrucciones que se le han dado a través del código G (gcode).

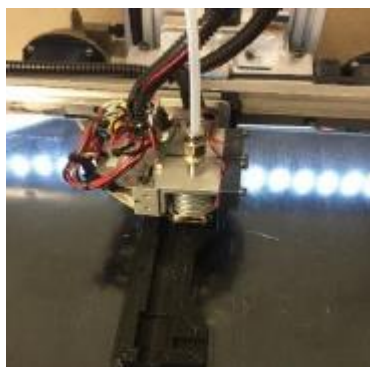



Figura 55. Extrusor y cama de iDeator12.

La impresora Ultimaker 3 [51] utiliza la tecnología de fabricación FFF y cuenta con las siguientes características:

Impresora 3D Ultimaker 3		
	Características	
	Tecnología	FDM-FFF
	Materiales	PLA, ABS
	Tamaño máximo de impresión	197x215x300 mm
	Nº de extrusores	2
	Base calefactable	Sí
	Tamaño de la impresora	342x505x588 mm
	Tipos de archivo	.gcode

Esta impresora utiliza el programa de impresión cura (el programa Repetier explicado anteriormente utiliza cura internamente) para transformar los archivos “.stl” a “.gcode”

El tiempo que se tarda en fabricar las distintas piezas puede ser muy variado, yendo desde los 5 minutos hasta las 5 horas. Esta diferencia de tiempo en la impresión se debe no solo al tamaño de la pieza que se está generando, sino también a la densidad que se le quiera dar a la pieza. La densidad de la pieza determina la robustez o lo maciza que será la pieza una vez esté terminada.

Durante la impresión de las piezas es muy posible que se produzcan errores debido a diversos motivos.

Estos errores pueden ser de diseño (reducción de tamaño de los taladros por el enfriamiento de la pieza, tolerancias demasiado pequeñas) o ser errores producidos por la impresora como que la pieza se despegue de la base por una mala calibración de la base o que exista suciedad.

Algunos errores que también se pueden dar es que se suelte uno de los extrusores por la vibración que existe durante su movimiento, que se produzcan atascos en los extrusores o que se curven partes de la base de la pieza debido a cambios de temperatura bruscos.



Figura 56. Fallos en la fabricación de las piezas.

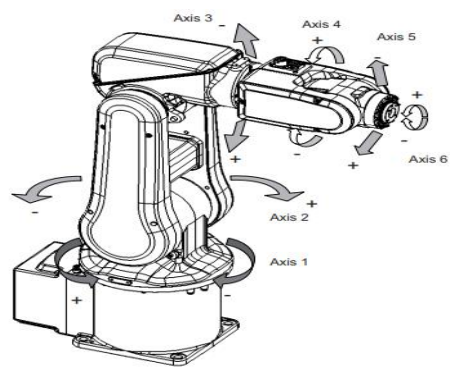
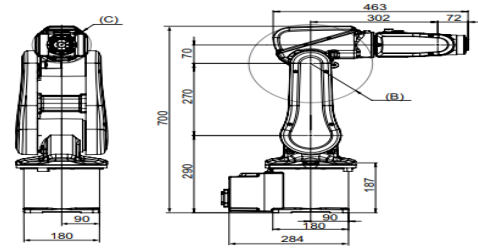
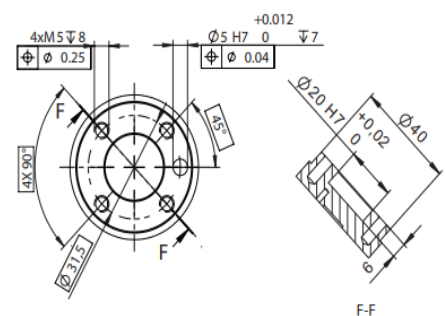
3.7.2.7. Robot ABB IRB 120.

El robot IRB 120 es un manipulador industrial [52] de 6 ejes que soporta una carga de 3 kg útiles llegando a los 4 kg con la posición de la muñeca en vertical.

El robot dispone del controlador *IRC5 Compact* y el *software RobotWare* que permite el control del movimiento del robot, la programación y ejecución de esos programas y la comunicación con sistemas externos.

Según la información técnica aportada por el fabricante en su página web:

Tabla 9. Características técnicas del Robot ABB IRB120.

ROBOT ABB IRB120	
Ejes del robot	
Dimensiones	
Capacidad de manejo	3 kg
Alcance	0,58 m
Peso	25 kg
Brida de las herramientas	

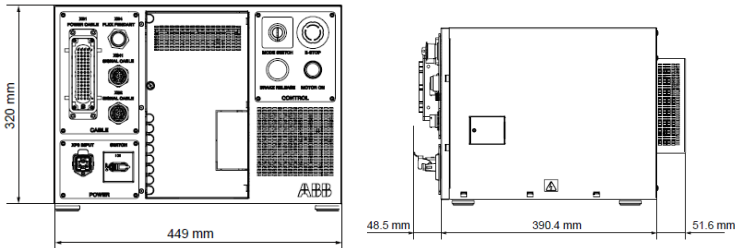
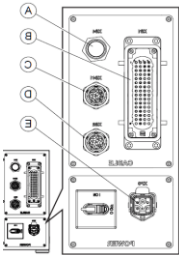
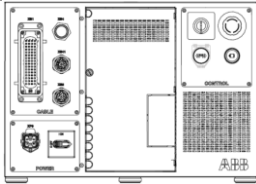
El robot dispone de unos soportes a lo largo de su estructura en los que se pueden conectar diversos dispositivos como electroválvulas para el funcionamiento de las pinzas cuyos actuadores son sistemas neumáticos o sensores para comunicarse con su entorno.

3.7.2.8. Controlador IRC5 Compact.

El controlador IRC5 Compact 74 [53] permite el control del robot y de los elementos externos a este que se le quieran añadir como es el caso de las herramientas diseñadas en este TFG.

De la misma manera que con el robot a continuación se exponen las distintas características básicas con las que cuenta el controlador

Tabla 10. Características técnicas del controlador IRC5 Compact

IRC5 COMPACT											
Medidas											
Interfaz de conexiones	 <table border="1"> <tr><td>A</td><td>Conexión de Flexpendant</td></tr> <tr><td>B</td><td>Alimentación del robot</td></tr> <tr><td>C</td><td>Tarjeta de medida serie</td></tr> <tr><td>D</td><td>Tarjeta de medida serie del robot</td></tr> <tr><td>E</td><td>Conexión eléctrica principal</td></tr> </table>	A	Conexión de Flexpendant	B	Alimentación del robot	C	Tarjeta de medida serie	D	Tarjeta de medida serie del robot	E	Conexión eléctrica principal
A	Conexión de Flexpendant										
B	Alimentación del robot										
C	Tarjeta de medida serie										
D	Tarjeta de medida serie del robot										
E	Conexión eléctrica principal										
Botonera del panel frontal	 <table border="1"> <tr><td>A</td><td>Interruptor de alimentación</td></tr> <tr><td>B</td><td>Liberación de frenos</td></tr> <tr><td>C</td><td>Selector de modo</td></tr> <tr><td>D</td><td>Motores ON</td></tr> <tr><td>E</td><td>Paro de emergencia</td></tr> </table>	A	Interruptor de alimentación	B	Liberación de frenos	C	Selector de modo	D	Motores ON	E	Paro de emergencia
A	Interruptor de alimentación										
B	Liberación de frenos										
C	Selector de modo										
D	Motores ON										
E	Paro de emergencia										
Peso del armario	28,5 Kg										
Potencia consumida	250 W										

3.7.2.9. Optoacoplador.

Un optoacoplador es un dispositivo electrónico formado por un LED y un fototransistor unidos de forma óptica por la luz que es emitida por el LED.

Una vez se enciende el LED, el fototransistor entra en la zona de conducción y permite circular la corriente como un transistor normal.

El optoacoplador utilizado tiene un encapsulado de tipo DIP de 6 pines modelo 4n35 de un único canal formado por un LED infrarrojo de Arseniuro de galio-aluminio y un fototransistor NPN de silicio.

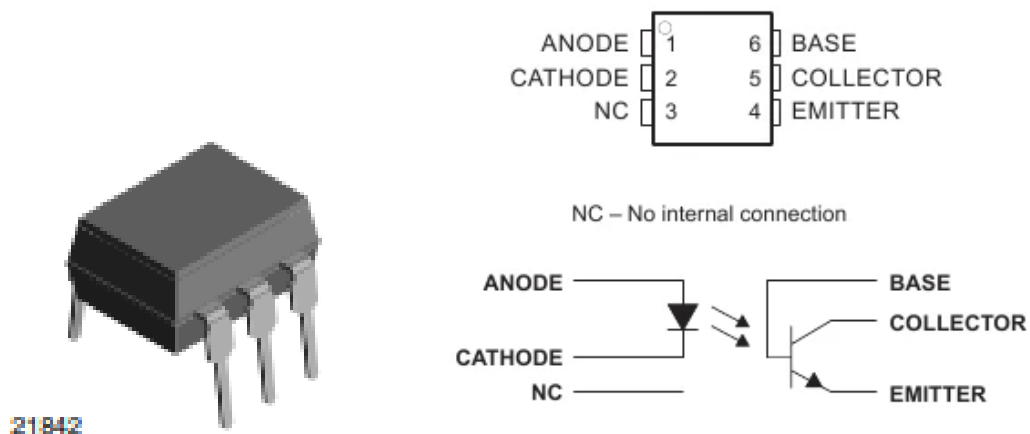


Figura 57. Optoacoplador.

Tabla de valores de entrada y de salida del optoacoplador utilizado dada por el fabricante.

Tabla 11. Valores de entrada y salida del optoacoplador utilizado

Valores máximos absolutos				
Parámetro	Condición de la prueba	Símbolo	Valor	Unidad
Entrada				
Voltaje inverso		V_R	6	V
Corriente directa		I_F	50	mA
Corriente de sobretensión	$t \leq 1 \mu s$	I_{FSM}	1	A
Potencia disipada		P_{DISS}	70	mW
Voltaje de ruptura fuente-emisor		V_{CEO}	70	V
Voltaje de ruptura emisor-base		V_{EBO}	7	V
Intensidad de la fuente		I_C	50	mA
	$t \leq 1 ms$	I_C	100	mA
Potencia disipada		P_{DISS}	70	mW

3.8. Metodología de trabajo.

La metodología de trabajo [54] sirve para definir las pautas y procesos a la hora de realizar cualquier tipo de proyecto. Esta permite optimizar los recursos, mejorar la calidad del proyecto y establecer las prioridades.

Actualmente las metodologías de trabajo más usadas son las siguientes:

- Cascada o “*waterfall*”: Es una metodología de gestión secuencial usada comúnmente en el desarrollo de software. Compuesta por una serie de fases estáticas (análisis de requisitos, diseño, prueba, implementación y mantenimiento). Ofrece un mayor control por cada fase, pero es difícil de readaptar si se da un cambio en el alcance de proyecto.
- Agile: Usada en proyectos que necesitan de gran velocidad y flexibilidad. Las fases de Agile se denominan “sprints” que son ciclos cortos de finalización de hitos. Este tipo de metodología permite ajustes rápidos durante el proyecto y facilita la identificación de problemas y la modificación de lo que sea necesario durante el proyecto.
- Híbrido: Mezcla el método de cascada junto con Agile. La planificación y los requisitos se hacen según el método cascada mientras que el diseño, el desarrollo y la implementación es Agile.
- Método de ruta crítica: Se utiliza cuando un proyecto tiene tareas interdependientes. Se utiliza una lista de actividades, fechas límite de consecución de trabajo y puntos con dependencias, hitos, etc.
- Gestión de proyectos de cadena crítica: Gestiona los recursos en vez de las actividades del proyecto.
- Six Sigma: Desarrollado por Motorola. Se basa en datos y cuenta con tres componentes de importancia donde los puntos más repetidos son definir el proyecto o problema, medir el alcance y analizar el proyecto.
- Scrum: Forma parte de las metodologías Agile. Se usan periodos que duran treinta días para definir las tareas de mayor importancia.

Al comienzo de este proyecto se quería usar una metodología de tipo Agile, definiendo las tareas que se debían hacer en periodos de tiempo limitados y separados para acotar los tiempos de entrega y debido a que es una metodología novedosa.

Finalmente, esta idea se desechó debido a las complicaciones encontradas a lo largo del TFG y la imposibilidad de dedicar el tiempo necesario que requería la culminación de los hitos definidos.

Por lo expuesto anteriormente se decidió utilizar una metodología híbrida.

Este modelo mezcla la metodología en cascada con la agile permitiendo aprovechar las ventajas de ambas formas de desarrollar un proyecto.

De esta manera se han podido establecer las siguientes fases de trabajo necesarias para afrontar con éxito este proyecto. Las fases que se han aplicado de ambas metodologías son las siguientes: “Requisitos”, “Diseño”, “Implementación”, “Verificación” y “Mantenimiento”.

- **Requisitos:** Es la etapa donde se estudian las necesidades o problemas que hay que afrontar para llegar al resultado esperado. Hay que ser extremadamente cuidadoso durante esta etapa ya que condicionará el desempeño del resto de fases.
- **Diseño:** Se define como serán las medidas tomadas para finalizar el proyecto y cuál puede ser la solución final.
- **Iteración:** Se realizan una serie de pruebas según se va avanzando en el diseño de las piezas fabricadas y en caso de ser necesario se modifican acciones anteriores hasta que el funcionamiento es el esperado. Esto permite un estudio de posibles prototipos y desde ahí elegir el producto final.
- **Puesta en marcha progresiva:** A medida que se iba alcanzando el producto final deseado se han ido desarrollando los programas de simulación y de control necesarios, así como una estación de pruebas físicas que simulaban el comportamiento del robot a la hora de enviar las señales digitales para el movimiento de la pinza.
- **Verificación:** Se comprueba que todo funciona como se ha esperado y que el diseño cumple con todos los requisitos.
- **Instalación y mantenimiento:** Se completa la instalación y se asegura que el producto funcione a lo largo del tiempo incluyendo alguna mejora o dispositivo complementario a lo creado.

Como resultado se obtiene un diagrama de Gantt como el siguiente:

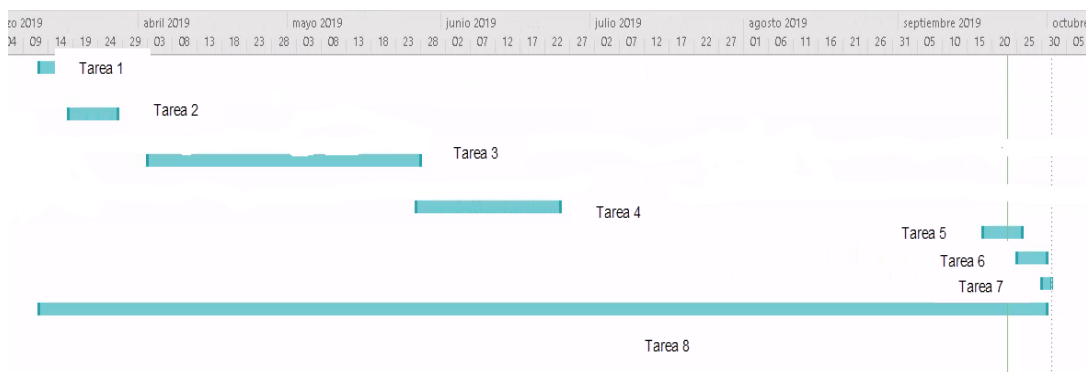


Figura 58. Diagrama de Gantt del proyecto.

Tarea 1: Conocimiento de los requisitos necesarios para diseñar las herramientas.

Tarea 2: Diseño y prueba de las herramientas Porta-Rotulador.

Tarea 3: Diseño y prueba de la pinza motor paso a paso.

Tarea 4: Diseño y prueba de la pinza servomotor.

Tarea 5: Simulación con RobotStudio.

Tarea 6: Conexiones del robot y la pinza servomotor.

Tarea 7: Pruebas finales.

Tarea 8: Redacción del documento. Esta tarea se ha ido completando y actualizando desde el comienzo del proyecto hasta su finalización.

3.9. Comparación económica.

Finalmente, se ha realizado una comparación entre el precio final del proyecto frente a la compra de una de las pinzas a la marca SCHUNK.

Esto se realiza para comprobar si, como se ha propuesto en los objetivos del TFG, resulta más rentable el diseño de las herramientas o la compra al fabricante por si en algún futuro fuera necesario rehacer alguna de las piezas o hacerla desde cero.

Para llevar a cabo la comparación se usará la pinza accionada por presión que tiene el laboratorio del área de Tecnología Electrónica frente a los materiales necesarios para la creación de las distintas herramientas antes expuestas. El resumen de los gastos se detalla en una tabla al final de este punto.

Para realizar el estudio económico se han supuesto una serie de escenarios que se irán viendo según se avance en la explicación

En primer lugar, se presenta el gasto económico que podría tener el diseño desde cero de cualquier útil que se pueda usar en este tipo de brazos robóticos.

Para que quede lo más completo posible y pueda ser comparable a una pinza de un fabricante comercial se ha supuesto que el creador de los útiles es un ingeniero recién graduado que cobra 2142,86€ brutos al mes.

Puesto que el TFG son 12 créditos, que en la URJC un crédito equivale a 10 horas y que el diseño, las pruebas y la configuración de las dos pinzas expuestas han sido aproximadamente el 67% del trabajo de los 12 créditos se obtiene que la mano de obra es de 1196,24€.

Se han utilizado una serie de componentes para el montaje de estas piezas de los cuales los más relevantes (tienen un precio significativamente más elevado que el resto de los materiales) son los siguientes:

- Motor paso a paso NEMA 8
- Servomotor SG90
- Placa Arduino
- CNC shield

Respecto a la pinza neumática de la marca SCHUNK con unas prestaciones similares a las de la pinza de este TFG tiene un valor de 370,60€ antes de impuestos.

La pinza de SCHUNK modelo KGG 60-40 es un útil de aluminio, por lo que es ligera, con un sistema neumático de dos pistones y de agarre paralelo similar al funcionamiento de la pinza del servo.

La compra de esta pinza viene con un sensor de presencia que indica si hay un objeto sobre el que cerrar la pinza o no.

Esto también podría llevarse a cabo en la pinza diseñada como se puede ver el en apartado 4 donde se incluyen líneas futuras de desarrollo del proyecto. También se han añadido otros elementos con los que se puede aprovechar la configuración realizada en la instalación del Arduino y la placa de soldadura.

Al precio de la pinza comprada a SCHUNK habría que añadirle el sistema neumático que permite el funcionamiento de la pinza mientras que mi pinza funciona de forma autónoma. El precio de diseñar las piezas necesarias para el enganche de la pinza con la brida y las piezas que comprenden al sistema de agarre o “dedos” de la pinza. En caso de no diseñarlos en la universidad sería necesario comprarlos junto con la pinza. Se estima en 10 horas el diseño y montaje de esta pinza y las piezas diseñadas al robot.

Para los cálculos se ha aplicado un IRPF del 30% para conocer el salario neto de un ingeniero recién graduado.

Puesto que la bomba del sistema neumático, el brazo robótico y las impresoras 3D (materiales implicados en el TFG) son activos de la universidad se ha decidido no tenerlas en cuenta.

Resumiendo lo expuesto anteriormente:

Tabla 12. Presupuesto TFG.

TFG			
Antes de impuestos			
Artículo	Precio unitario	Cantidad	Total (€)
Mano de obra	14,88(€/h)	90 (h)	1339,2
NEMA 8	27,77 (€/ea)	1 (ea)	27,77
SG90	2,9 (€/ea)	1 (ea)	2,9
Arduino	19 (€/ea)	1 (ea)	19
CNC Shield	5,45 (€/ea)	1 (ea)	5,45
TOTAL			1394,32
Después de impuestos			
Artículo	Precio unitario	Cantidad	Total (€)
Mano de obra	10,42(€/h)	90 (h)	837,37
NEMA 8	33,60 (€/ea)	1 (ea)	33,60
SG90	3,51 (€/ea)	1 (ea)	3,51
Arduino	22,99 (€/ea)	1 (ea)	22,99
CNC Shield	6,59 (€/ea)	1 (ea)	6,59
TOTAL			904,6

Tabla 13. Presupuesto SCHUNK

SCHUNK			
Antes de impuestos			
Artículo	Precio unitario	Cantidad	Total (€)
Mano de obra	14,88(€/h)	10 (h)	148,80
KGG 60-40	370,60 (€/ea)	1 (ea)	370,60
MMS 22-S-M8	52,32 (€/ea)	2 (ea)	104,32
PORTES	16,51 (€/ea)	1 (ea)	16,51
TOTAL			640,23
Después de impuestos			
Artículo	Precio unitario	Cantidad	Total (€)
Mano de obra	10,42(€/h)	10 (h)	104,2
KGG 60-40	448,43 (€/ea)	1 (ea)	440,43
MMS 22-S-M8	63,31 (€/ea)	2 (ea)	126,62

PORTES	19,97 (€/ea)	1 (ea)	19,97
TOTAL			699,22

Como se puede observar en un primer momento el precio de las pinzas diseñadas exceden al de la pinza comercial, sin embargo, gracias a la filosofía *open source* el diseño y la configuración es fácilmente replicable por lo tanto no será necesario volver a incurrir en el gasto de la mano de obra.

Otra de las ventajas es que los repuestos de las piezas se pueden obtener fácilmente y de manera inmediata, mientras que con la otra pinza se depende de la disponibilidad de esta y del tiempo de entrega por parte del fabricante. En el caso de las pinzas del TFG si se rompe uno solo de los componentes, como son piezas independientes, se pueden sustituir sin replicar la pieza al completo.

Se añade también una lista de materiales o BOM (*“Bill Of Materials”*) necesarios para llevar a cabo el montaje de las pinzas para futuros proyectos.

Tabla 14. Lista de materiales

Lista de materiales		
Material	Cantidad	Precio por unidad
NEMA 8	1	33,60 (€/ea)
SG90	1	3,51 (€/ea)
Arduino	1	22,99 (€/ea)
CNC Shield	1	6,59 (€/ea)
Tornillo M2x16	5	0,01 (€/ea)
Tornillo M2x8	2	0,01 (€/ea)
Cabezal servomotor	1	Incluido con la compra del servo
Tornillo M4x8	4	0,01 (€/ea)
Tornillo M4x50	1	0,05 (€/ea)
Tornillo M4x16	2	0,02 (€/ea)

4. Conclusiones.

A continuación, se presentan los resultados obtenidos tras finalizar este Trabajo de Fin de Grado, así como si se han cumplido los objetivos propuestos en el apartado 2.

En primer lugar, se ha conseguido hacer distintas herramientas funcionales que permiten el desarrollo de diversas actividades docentes en la asignatura de RIM cumpliendo así el objetivo principal del TFG.

En segundo lugar y a raíz del cumplimiento del objetivo principal el resto de los hitos se han ido completando también.

- Las herramientas diseñadas son similares a las que se pueden encontrar en entornos industriales.
- Con el sistema de agarre entre la pinza y la brida del robot la colocación y sustitución de las pinzas es más rápido y existen menos riesgos de dañar la brida en el proceso.
- Los diseños de las pinzas, esquemas de conexiones, programas y explicaciones de todo el proceso de diseño está disponible en GitHub por lo que todo el que quiera replicar las pinzas o continuar con el proyecto y tenga los medios para hacerlo tiene toda la información disponible.
- Los diseños de las pinzas admiten cambios y actualizaciones como puede ser las modificaciones de las dimensiones de los elementos que componen la herramienta, incluir unos sensores de presión en los dedos de las pinzas para saber si se está recogiendo algún objeto, de forma similar a los sensores comerciales.
- Se puede utilizar junto con elementos complementarios al robot como pueden ser cintas de transporte y el sistema de seguridad ya incluido.
- Se ha aprendido a utilizar el *software* del fabricante “*RoboStudio*” para realizar la simulación y las pruebas de la pinza.
- Además, se ha recordado y profundizado en los conocimientos obtenidos durante el grado sobre programación en Arduino.
- Se ha hecho la electrónica necesaria para poder comunicar el robot con el Arduino y como consecuencia se ha aprendido a soldar mediante soldadura de estaño y de diversos programas para la esquematización de los circuitos.

Para finalizar, quiero añadir que como líneas futuras de este proyecto una evolución en el diseño de las pinzas, podría ser la adición de un sensor de proximidad controlado por Arduino en el cuerpo del robot que al hacer que se aproxime a algo se pare y evitar, de esa manera, que se golpee

a sí mismo en un descuido durante el control o en una mala programación, o golpee a alguien que se pueda acercar demasiado. De la misma manera, una mejora, puede ser añadir el sensor de presión mencionado anteriormente en este apartado.

5. Bibliografía.

- [1] **Del Val, José Luis**. Revista ingeniería Deusto. Industria 4.0: La transformación de la Industria, 2016 [En línea] [Consultado el: 13 de marzo de 2019] Disponible en: <https://revistaingenieria.deusto.es/tag/industria-4-0/>
- [2] **SAP**. Definición de internet de las cosas. [En línea] [Consultado el: 13 de marzo de 2019] Disponible en: <https://www.sap.com/spain/trends/internet-of-things.html>
- [3] **Iberdrola**. ¿Somos conscientes de los retos y principales aplicaciones de la Inteligencia Artificial? [En línea] [Consultado el: 14 de marzo de 2019] Disponible en: <https://www.iberdrola.com/innovacion/que-es-inteligencia-artificial>
- [4] **Equipo Editorial**. Reporte Digital. El mundo visto desde la tecnología M2M, 2019 [En línea] [Consultado el: 20 de mayo de 2019] Disponible en: <https://reportedigital.com/iot/infografia-el-mundo-visto-desde-la-tecnologia-m2m/>
- [5] **Oracle**. ¿Qué es el big data? [En línea] [Consultado el: 14 de abril de 2019] Disponible en: <https://www.oracle.com/es/big-data/guide/what-is-big-data.html>
- [6] **Autodesk**. ¿Qué es la impresión 3D? [En línea] [Consultado el: 16 de abril de 2019] Disponible en: <https://www.autodesk.es/solutions/3d-printing>
- [7] **Materialise**. Modelado por deposición fundida (FDM) [En línea][Consultado el: 16 de abril de 2019] Disponible en: <https://www.materialise.com/es/manufacturing/tecnologia-de-impresion-3d/modelado-por-deposicion-fundida>
- [8] **3D SYSTEMS**. Estereolitografía. [En línea] [Consultado el: 16 de abril de 2019] Disponible en: <https://es.3dsystems.com/on-demand-manufacturing/stereolithography-sla>
- [9] **Materialise**. Sinterización por láser. [En línea] [Consultado el: 16 de abril de 2019] Disponible en: <https://www.materialise.com/es/manufacturing/tecnologia-de-impresion-3d/sinterizacion-por-laser>
- [10] **3Dnatives**. Impresión 3D por estereolitografía, 2017 [En línea] [Consultado el: 17 de abril de 2019] Disponible en: <https://www.3dnatives.com/es/impresion-3d-por-estereolitografia-les-explicamos-todo/>
- [11] **3Dnatives**. Guía completa: Sinterizado selectivo por láser o SLS, 2019. [En línea] [Consultado el: 17 de abril de 2019] Disponible en: <https://www.3dnatives.com/es/sinterizado-selectivo-por-laser-les-explicamos-todo/>
- [12] **3Dnatives**. FDM o modelado por deposición fundida, 2017. [En línea] [Consultado el: 17 de abril de 2019] Disponible en: <https://www.3dnatives.com/es/modelado-por-deposicion-fundida29072015/>
- [13] **3Dfils**. Historia de la impresión 3D, 2018. [En línea] [Consultado el: 18 de abril de 2019] Disponible en: https://www.3dfils.com/de/blog/20_historia3d

- [14] **RepRap.org**. RepRap, 2019. [En línea] [Consultado el 18 de abril de 2019] Disponible en: <https://reprap.org/wiki/RepRap>
- [15] **JONES, R., HAUFE, P., SELLS, E., IRAVANI, P., OLLIVER, V., PALMER, C. and BOWYER, A.**, 2011. RepRap – the replicating rapid prototyper. *Robotica*, vol. 29, no. 1, pp. 177–191. DOI 10.1017/S026357471000069X. [En línea] [Consultado el: 18 de abril de 2019] Disponible en: https://www.cambridge.org/core/services/aop-cambridge-core/content/view/5979FD7B0C066CBCE43EEAD869E871AA/S026357471000069Xa.pdf/rep_rap_the_replicating_rapid_prototyper.pdf
- [16] **Borda Elejabarrieta, J.** Sisteplant. Diseño para fabricación y montaje (DFMA). [En línea] [Consultado el: 19 de abril de 2019] Disponible en: <https://www.sisteplant.com/compartiendo-conocimiento/articulos/disenio-para-fabricacion-y-montaje-dfma-unitarios-de-alto-valor-anadido-tecnologico-vat/>
- [17] **Brockotter, R.** 3D HUBS. Consideraciones de diseño claves para la impresión 3D. [En línea] [Consultado el: 19 de abril de 2019] Disponible en: <https://www.3dhubs.com/knowledge-base/key-design-considerations-3d-printing>
- [18] **Free Software Foundation**. ¿Qué es el software libre?, 2013. [En línea] [Consultado el: 25 de abril de 2019] Disponible en: <https://www.fsf.org/es/recursos/que-es-el-software-libre>
- [19] **BBVA**. ¿Qué es el “hardware” libre?, 2018. [En línea] [Consultado el: 25 de abril de 2019] Disponible en: <https://www.bbva.com/es/que-es-el-hardware-libre/>
- [20] **Open Source Hardware Association**. Declaración de principios 1.0. [En línea] [Consultado el: 29 de septiembre de 2019] Disponible en: <https://www.oshwa.org/definition/spanish/>
- [21] **Arduino**. About Us. [En línea] [Consultado el 25 de abril de 2019] Disponible en: <https://www.arduino.cc/en/Main/AboutUs>
- [22] **Inc. Robotics**. [En línea] [Consultado el: 29 de abril de 2019] Disponible en: <https://www.inc.com/encyclopedia/robotics.html>
- [23] **IFR Internacional Ferderation of Robotics**. International robot standarization within ISO [En línea] [Consultado el 29 de abril de 2019] Disponible en: <https://ifr.org/standardisation>
- [24] **Barrientos, Antonio, Peñín, Luis Felipe, Balaguer, Carlos, Aracil, Rafael**. *Fundamentos De Robótica*. 2da Edición. McGraw-Hill, 2007. ISBN: 978-84481-5636-7
- [25] **Angulo, Jose Mª. Romero, Susana. Angulo, Ignacio**. *Microbótica*. 1a Edición. Paraninfo, 2000. ISBN: 84-283-2597-9.
- [26] **Hasan, T.** GrabCAD. 4 claw gripper, 2018 [En línea] Disponible en: <https://grabcad.com/library/4-claw-gripper-1>
- [27] **Gil, D.** GrabCAD. Robotic Gripper Modify, 2019 [En línea] Disponible en: <https://grabcad.com/library/robotic-gripper-modify-1>

- [28] **Hectdiaf**. Thingiverse. Claw card, gripper, 2013. [En línea] Disponible en: <https://www.thingiverse.com/thing:151449>
- [29] **SCHUNK**. MPG. [En línea] Disponible en: https://schunk.com/es_es/sistemas-de-agarre/series/mpg/
- [30] **Alciatore, David, Histan, Michael**. *Introduction to Mechatronic and Measurement Systems*. 4ª Edición. McGraw-Hill, 2011. ISBN: 978-0-07-338023-0
- [31] **Bolton, William**. *Mecatrónica, Sistemas de control electrónico en ingeniería mecánica y eléctrica*. 2ª Edición. Alfaomega, 2009. ISBN: 978-6077854326
- [32] **Bertomeu, J., García, J., Gardonio, S., Gómez, S., Granados, F.** Motor Paso a Paso. [En línea] [Consultado el: 3 de mayo de 2019] Disponible en: http://www1.frm.utn.edu.ar/mielectricas/docs2/PaP/MOTOR_PaP_FINAL.pdf
- [33] **Menna**. COMOFUNCIONA. Como funciona un servomotor, 2018. [En línea] [Consultado el 4 de mayo de 2019] Disponible en: <http://como-funciona.co/un-servomotor/>
- [34] **Pelayo Vlietman, M.** GitHub. [En línea] Disponible en: <https://github.com/Marcos-vp>
- [35] **Gonzalez-Gomez, J. (Obijuan)**. GitHub. Pinza robótica porta acreditaciones. [En línea] [Consultado el 9 de mayo de 2019] Disponible en: <https://github.com/Obijuan/3D-parts/wiki/Pinza-rob%C3%B3tica-porta-acreditaciones>
- [36] **Martín, Diego**. Robótica Industrial. Entradas y salidas digitales del controlador ABB IRC5 Compact v1.0. Apuntes para la configuración de las señales de entrada y de salida. 1ª Edición. URJC, 2019.
- [37] **Arduino**. GitHub. Arduino. [En línea] [Consultado el: 13 de mayo de 2019] Disponible en: <https://github.com/arduino/Arduino>
- [38] **EducatiBot**. Repetier Host. [En línea] [Consultado el: 13 de mayo de 2019] Disponible en: <http://educatibot.com/impresion-3d/programas/repetier-host/>
- [39] **DIY Fever**. DIY Layout Creator. [En línea] [Consultado el: 14 de mayo de 2019] Disponible en: <http://diy-fever.com/software/diylc/>
- [40] **KiCad**. About KiCad. [En línea] [Consultado el: 14 de mayo de 2019] Disponible en: <http://kicad-pcb.org/about/kicad/>
- [41] **FreeCAD**. Acerca de FreeCAD. [En línea] [Consultado el: 15 de mayo de 2019] Disponible en: https://www.freecadweb.org/wiki/About_FreeCAD.
- [42] **GitHub**. GitHub. [En línea] [Consultado el: 15 de mayo de 2019]. Disponible en: <https://github.com/features>
- [43] **Git**. Control de versiones. [En línea] [Consultado el: 30 de septiembre de 2019]. Disponibles en: <https://git-scm.com/book/es/v2>
- [44] **ABB**. RobotStudio. [En línea] [Consultado el: 15 de mayo de 2019]. Disponible en: <https://new.abb.com/products/robotics/es/robotstudio>

- [45] **Arduino**. Arduino Uno. Especificaciones técnicas. [En línea] [Consultado el: 20 de junio de 2019]. Disponible en: <https://store.arduino.cc/arduino-uno-smd-rev3>
- [46] **BricoGeek**. CNC Shield v3. Características. [En línea] [Consultado el: 20 de junio de 2019] Disponible en: <https://tienda.bricogeek.com/shields-arduino/837-arduino-cnc-shield-v3.html>
- [47] **Pololu**. A4988 Stepper Motor Driver Current. [En línea] [Consultado el: 20 de junio de 2019] Disponible en: <https://www.pololu.com/product/1182>
- [48] **Pololu**. Stepper Motor: Bipolar, 200 Steps/Rev, 20×30mm, 3.9V, 0.6 A/Phase. [En línea] [Consultado el: 20 de junio de 2019] Disponible en: <https://www.pololu.com/product/1204>
- [49] **Bricogeek**. Micro Servo miniatura SG90. [En línea] [Consultado el : 20 de junio de 2019] Disponible en: <https://tienda.bricogeek.com/servomotores/968-micro-servo-miniatura-sg90.html>
- [50] **Imprimalia 3D**. iDeator 12. [En línea] [Consultado el: 21 de junio de 2019]. Disponible en: <http://imprimalia3d.com/services/ideator-12>
- [51] **Ultimaker**. Ultimaker 3 Impresora 3D de doble extrusor. [En línea] [Consultado el: 21 de junio de 2019] Disponible en: <https://ultimaker.tr3sdl.com/producto/ultimaker-3/>
- [52] **ABB**. Especificaciones del producto IRB 120. 2019. [Manual] [Consultado el: 22 de junio de 2019].
- [53]] **ABB**. Manual del producto. IRC5 Compact. 2018. [Manual] [Consultado el: 22 de junio de 2019].
- [54] **Gobierno TI**. CIO. Las metodologías de gestión de proyectos más populares. [En línea] [Consultado el: 3 de julio de 2019] Disponible en: <https://www.ciospain.es/gobierno-ti/las-metodologias-de-gestion-de-proyectos-mas-populares>
- [55] **Domínguez, P.** Open Classrooms. Gestiona tu proyecto de desarrollo. 2017. [En línea] [Consultado el: 3 de julio de 2019]. Disponible en: <https://openclassrooms.com/en/courses/4309151-gestiona-tu-proyecto-de-desarrollo/4538221-en-que-consiste-el-modelo-en-cascada>

6. Anexo.

6.1. Anexo 1. Programación en Arduino.

6.1.1. Código motor paso a paso.

Para las pruebas previas a la conexión con el robot de ABB y para la comprobación del diseño y del funcionamiento de la pinza se desarrolló un programa en Arduino en el que el motor giraba una cantidad determinada de pasos en una dirección en función de un valor dado en el puerto serie de Arduino.

Tras la comprobación del correcto funcionamiento del motor y tras haber realizado las conexiones necesarias en la electrónica de potencia se ha pasado a desarrollar el programa para el control del paso a paso simulando una señal de entrada enviada por el robot. El código se encuentra disponible en el GitHub del autor.

En el código se ha hecho el siguiente desarrollo:

- Configuración de las señales de salida del Arduino al motor:
 - dirPin: Señal de salida que consiste en un 0 o un 1 según el giro sea horario o antihorario respectivamente. Se ha asignado al pin digital 5. Según la disposición de la conexión del cableado del motor la dirección puede invertirse por lo que hay que conectar siempre el motor de la misma manera para evitar errores.
 - stepPin: Señal digital con la que el motor gira un paso o micropaso, dependiendo de si los pines M1, M2 o M3 estén activados. Se le ha asignado el pin 2 de la placa.
 - enable: Señal digital que activa o desactiva el controlador. Debido a que el controlador tiene una resistencia de pull-down no sería necesario asignarlo a ningún pin, sin embargo, por la referencia consultada se ha decidido incluirla. Se ha asignado al pin 8.

Para la asignación de los pines se ha utilizado el esquema de la CNC *shield* mostrado en el apartado 3.7.2.2.

- Posteriormente, se han declarado una serie de variables para el control del motor y la escritura del programa:
 - stepDelay: Tiempo que transcurre entre el envío de un 1 o un 0 para la activación de la señal STEP. Marca la velocidad a la que se moverá el motor.
 - entrada: Es una señal de entrada asignada al pin 9 y que sirve para determinar si el motor debe girar en una dirección u otra provocando que la pinza se abra o se cierre.

- valor: Variable que almacena el valor que se lee en la entrada.
 - steps: Variable que almacena el número de pasos que se deben dar hasta que el motor termine su recorrido. El número que se ha guardado es 100, es decir, dará 100 pasos hasta su detención. Como se ha activado uno de los pines de resolución de pasos equivale a 50 pasos enteros según la tabla mostrada en el punto Tabla 4. Configuración de micropasos y a 90° de giro teniendo en cuenta que en cada paso el motor se mueve 1,8° como se pueden ver en las especificaciones técnicas (Tabla 6).
- Se han definido dos estados distintos de la pinza para su funcionamiento, de esta forma una vez se cumpla la sentencia programada el motor dejará de girar, aunque le siga llegando una señal de activación:
 - m_cerrado: La pinza está en una posición donde los extremos de los dedos se encuentran en la dirección de la brida del robot.
 - m_abierto: La pinza está en una posición donde los extremos de los dedos están en una posición de 90 grados respecto la brida del robot.

Seguidamente se explican las funciones de programación del robot. Estas van precedidas por “void” que indica que son funciones que de las que no se espera que devuelva un valor.

- En “void setup” de Arduino IDE, función donde se configuran se han declarado las señales como entradas o salidas según se ha explicado antes y se ha guardado el estado inicial de la pinza como cerrado.
- En “void loop”, función bucle que contiene el funcionamiento del programa, se lee el valor que recibe el Arduino por el puerto “entrada” y se asigna a la variable “valor”. A continuación, se activa el estado en el que se encuentre la pinza en ese momento y se ponen las distintas condiciones de funcionamiento según este cerrada o abierta.

En caso de estar cerrado y si la señal recibida tiene un valor de 0, se llama a la función definida como “apertura” y la pinza se abrirá, y si recibe un valor de 1 se mantendrá en el estado de cierre sin moverse.

Si el estado es “abierto” y recibe un 1, se llama a la función “cierre” y la pinza se cerrará, y si recibe un 0 se quedará en el estado actual.
- Finalmente se han creado dos funciones, una de apertura y otra de cierre para determinar el control del motor:
 - Función apertura: Se define la dirección de giro del robot en sentido antihorario y una duración entre el envío de los pulsos eléctricos para realizar un paso de

25 ms (milisegundos). Posteriormente se inicia una cuenta para el control del número de pasos, esta variable se inicia con un valor igual a cero y se va sumando cada ciclo del bucle hasta llegar al número guardado en la variable “steps”. Finalmente se indica el nuevo estado en el que se encuentra la pinza.

- Función cierre: De la misma manera que en la función anterior se define la dirección de la pinza en sentido horario para el cierre de los dedos. El resto del código sigue la misma lógica que la función apertura. Finalmente se actualiza el estado en el que está la pinza.

```
const int dirPin = 5;
const int stepPin = 2;
int enable = 8;
int stepDelay;
int entrada = 9; //Entrada de corriente (1 o 0) para determinar si el motor abre o cierra
byte valor;
const int steps = 100;

// declaración de estados del motor

enum estado_pinza
{
    m_cerrado,
    m_abierto,
};

// e_motor: estado en el que está el motor.

estado_pinza e_motor = m_cerrado;

void setup() {
    // Marcar los pines como salida
    pinMode(dirPin, OUTPUT);
    pinMode(stepPin, OUTPUT);
    pinMode(enable, OUTPUT);
    pinMode(entrada, INPUT);
    digitalWrite(enable, LOW);
    e_motor = m_cerrado;
```

Figura 59. Programa en Arduino de la pinza con motor paso a paso

```

    }

    void loop() {

      delay (550);
      valor = digitalRead (entrada);

      switch (e_motor) {
        case m_cerrado:
          if (valor == LOW) {
            apertura ();
          }
          else ;{
          }
          break;
        case m_abierto:
          if (valor == HIGH) {
            cierre ();
          }
          else;{
          }
          break;
        }
      }
      void apertura () {
        digitalWrite(dirPin, HIGH);
        stepDelay = 25;
      }
    }
  }

```

Figura 60. Programa en Arduino de la pinza con motor paso a paso (Continuación)

```

      for (int x = 0; x < steps; x++) {
        digitalWrite(stepPin, HIGH);
        delay(stepDelay);
        digitalWrite(stepPin, LOW);
        delay(stepDelay);
        e_motor = m_abierto;
      }
    }
    void cierre () {
      digitalWrite(dirPin, LOW);
      stepDelay = 10;
      for (int x = 0; x < steps; x++) {
        digitalWrite(stepPin, HIGH);
        delay(stepDelay);
        digitalWrite(stepPin, LOW);
        delay(stepDelay);
        e_motor = m_cerrado;
      }
    }
  }
}

```

Figura 61. Programa en Arduino de la pinza con motor paso a paso (Continuación 2)

6.1.2. Código servomotor.

El código para el control del servo, de la misma manera que el programa del motor paso a paso, se encuentra disponible en el GitHub del autor.

- Para el control del servomotor, en primer lugar, se ha incluido la biblioteca predeterminada por Arduino para su programación que simplifica el código para el funcionamiento del servo.
- En segundo lugar, se ha definido el nombre del servo para que pueda ejecutar las órdenes de las funciones definidas más adelante.
- Posteriormente se han iniciado las distintas variables que se usarán en el código:
 - valor: Almacena el valor recibido por la entrada del Arduino, este puede tener un valor de 1 o 0.
 - entrada: Lee el valor obtenido por la señal enviada desde el brazo robótico. Se ha asignado al pin digital 5.
 - pos: Se inicia la variable con un valor igual a 0. Es un valor que se irá actualizando a medida que vaya girando el servo con la posición en la que se encuentre.
 - poten: Señal analógica de entrada que lee la posición en la que está el potenciómetro. Puesto que el rango de valores que alcanza el potenciómetro está comprendido entre 0 y 1023 es necesario realizar una conversión a los ángulos que tiene permitido girar el servo.
 - giro: Almacena el valor dado por la señal “poten”.
- Se han definido los estados en los que se encuentra la pinza en cada momento de la misma manera que se definieron los del motor paso a paso:
 - m_cerrado: Los dedos de la pinza toman la posición dada por el potenciómetro y una vez llegan a esa posición detienen su movimiento.
 - m_abierto: Partiendo del estado cerrado los dedos alcanzan una apertura de 180° y se mantienen en esa posición hasta que reciben una señal distinta de cero.
- En el “void setup” del programa solo se asigna el valor del pin al que estará conectado el servo para el envío de la señal PWM, en este caso es el pin número 9. Gracias a la biblioteca de Arduino no es necesario señalar si el resto de las señales son de entrada o de salida.
- En el “void loop” se asignan valores a las variables declaradas al inicio del código y se desarrolla el programa principal de movimiento de la pinza:

- valor: Lee la señal digital de entrada dada recibida a través del pin 5 de la placa Arduino.
- giro: Lee la señal analógica de entrada generada por el potenciómetro y posteriormente se mapea para convertir el rango de giro del potenciómetro al rango de giro de la pinza.
- Se activa el estado en el que se encuentra la pinza y se determina que si la pinza está cerrada y el Arduino recibe una señal de valor 0 los dedos se abrirán, en caso contrario seguirá cerrada.

Si la pinza está abierta y se recibe un 1 esta se abrirá.

La recepción de las señales 1 y 0 se leen a través del pin 5 de la placa Arduino.

- Finalmente se escriben las funciones de apertura y de cierre de la pinza:
 - La función apertura determina mediante un contador que la pinza pasará de una posición determinada por el giro del potenciómetro a una con un ángulo de 0° variando el ángulo en incrementos de 10° cada 15 ms. Cambia el estado de la pinza a abierto.
 - La función cierre provoca que la posición de los dedos varíe desde los 0° hasta la posición del potenciómetro a través también de un contador variando el giro en decrementos de 10° cada 15 ms. Cambia el estado de la pinza a cerrado.

```
#include <Servo.h>

Servo motor;
byte valor;
int entrada = 5;
int pos = 0;
int poten = A0;
int giro;
enum estado_pinza
{
    m_cerrado,
    m_abierto,
};
estado_pinza e_motor = m_cerrado;
void setup() {
    motor.attach (9);
    pinMode (entrada, INPUT);
}
void loop() {
    delay (100);
    valor = digitalRead (entrada);
    giro = analogRead (poten);
    giro = map (giro, 0, 1023, 0, 180);
    switch (e_motor) {
        case m_cerrado:
            if (valor == HIGH) {
                apertura ();
            }
    }
```

Figura 62. Código en Arduino de pinza con servomotor.

```

else :{}
break;
case m_abierto:
if (valor == LOW) {
  cierre ();
}
else: {}
break;
}
}
void apertura () {
  for (pos = giro; pos <= 180; pos +=10) {
    motor.write (pos);
    delay (15);
    e_motor = m_abierto;
  }
}
void cierre () {
  for (pos = 180; pos >= giro; pos -=10) {
    motor.write (pos);
    delay (15);
    e_motor = m_cerrado;
  }
}

```

Figura 63. Código en Arduino de pinza con servomotor (Continuación)

6.2. Anexo 2. Programación en RobotStudio.

6.2.1. Trayectoria del robot.

- MoveJ HOME: Movimiento de inicio en el arranque del robot. Es un movimiento tipo Joint.
- Reset DO_pinza: Se ordena al controlador que envía un cero lógico a la pinza para que esta se abra.
- MoveJ Punto1: Se mueve a la posición del punto 1 con un movimiento de tipo Joint.
- Move L Punto 2: Se mueve al punto 2, situado en la posición del centro de gravedad del cubo, con un movimiento lineal.
- Set DO_Pinza: Se envía un “1” para que la pinza se cierre.
- MoveL Punto1: El robot vuelve al punto 1 con el cubo cogido con un movimiento lineal.
- MoveJ Punto3_Int: El robot se mueve a un punto intermedio entre el punto HOME y el punto final mediante un movimiento Joint.
- MoveJ Punto 4: Se lleva al robot a un punto cercano al destino en el que se quiere dejar el cubo con un movimiento Joint.
- MoveL Puntofinal: Se mueve el robot hasta el punto en el que se quiere dejar el cubo con un movimiento lineal.
- Reset DO_pinza: Se envía un “0” al robot para que abra la pinza y suelte el cubo.
- MoveL Punto 4: Se vuelve al punto 4 con un movimiento Lineal.
- MoveJ HOME: Se lleva el robot a la posición de origen mediante un movimiento Joint.

6.2.2. Programa de funcionamiento del robot.

A continuación, se puede observar el código utilizado para describir la trayectoria del robot. Los valores “vx” especifican a la velocidad a la que debe moverse el robot y los “zx” los valores de tolerancia para encontrar los puntos en el espacio y que delimitan por donde puede pasar o donde puede terminar su recorrido.

```

1  MODULE Module1
2  □  CONST robtarget HOME:=[453.946750508,0,519.179491924],[0.5,0,0.866025404,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09];
3  □  CONST robtarget Punto1:=[500,-10,45],[0,0,1,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09];
4  □  CONST robtarget Punto2:=[500,-10,20],[0,0,1,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09];
5  □  CONST robtarget Punto3_Int:=[400,150,200],[0,0,1,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09];
6  □  CONST robtarget Punto4:=[500,90,45],[0,0,1,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09];
7  □  CONST robtarget Puntofinal:=[500,90,20],[0,0,1,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09];
8  □  PROC Path_10()
9      MoveJ HOME,v200,fine,Mi_mecanismo_1\WObj:=Workobject_1;
10     Reset DO_pinza;
11     MoveJ Punto1,v200,fine,Mi_mecanismo_1\WObj:=Workobject_1;
12     MoveL Punto2,v100,fine,Mi_mecanismo_1\WObj:=Workobject_1;
13     Set DO_pinza;
14     MoveL Punto1,v100,fine,Mi_mecanismo_1\WObj:=Workobject_1;
15     MoveJ Punto3_Int,v200,fine,Mi_mecanismo_1\WObj:=Workobject_1;
16     MoveJ Punto4,v200,fine,Mi_mecanismo_1\WObj:=Workobject_1;
17     MoveL Puntofinal,v100,fine,Mi_mecanismo_1\WObj:=Workobject_1;
18     Reset DO_pinza;
19     MoveL Punto4,v100,fine,Mi_mecanismo_1\WObj:=Workobject_1;
20     MoveJ HOME,v400,fine,Mi_mecanismo_1\WObj:=Workobject_1;
21     ENDPROC
22  ENDMODULE

```

Figura 64. Programa en RAPID de la trayectoria