CONSTRUCCIÓN Y PROGRAMACIÓN DE UN BRAZO ROBÓTICO

Yousra Benchikh, José Antonio Corral Palmero

Resumen– Este trabajo de Fin de Grado de ciudades inteligentes y sostenibles, incluye la construcción de un brazo robótico programado en Arduino. Este incluirá el cotejo de varios brazos robóticos en aras de seleccionar el que se ciña mejor a la máxima del proyecto, para acabar construyéndolo. Incluyendo desde la programación, hasta la exposición de las capacidades del brazo escogido con una serie de pruebas y pequeñas órdenes que este deberá obedecer.

Palabras clave– Impresión 3d, programación, Arduino, diseño, software, ciudad inteligente, robótica, campo de pruebas.

Abstract– This end-of-degree project for the degree in smart and sustainable cities includes the construction of a robotic arm programmed in Arduino. This will include the comparison of several robotic arms in order to select the one that best fits the maximum of the project, to finish building it. Including its programming with Arduino, to the exposure of the capabilities of the chosen arm with a series of tests and small commands that it must obey that, in the end, will be the final evaluation.

Keywords– 3d printing, programming, Arduino, design, software, smart city, robotics, proving ground.

1 INTRODUCCIÓN

UNA ciudad inteligente, por definición, es aquella que utiliza la tecnología y la innovación de una manera eficiente, el desarrollo sostenible con la finalidad de mejorar la calidad de vida de los ciudadanos. Para conseguirlo, requiere del Internet de las cosas (IoT), el big data, las aplicaciones móviles con el fin de contribuir en la mejora del medio ambiente, la vida de los ciudadanos, optimizar procesos como el transporte público, trabajar en la transparencia y la gestión pública administrativa e incluso, incentivar la participación ciudadana. Es por ello que, para poder llevar a cabo una 'Smart City', se requiere de medios que faciliten la vida y es aquí, donde entra el mundo de la robótica [37].

Podemos definir el significado de robótica como una ciencia que aglutina varias vertientes tecnológicas o disciplinas, con el objetivo de diseñar máquinas robotizadas que sean capaces de realizar tareas automatizadas o de simular el comportamiento humano o animal, en función de la capacidad del software [22]. También, puede considerar como la ciencia que tiene por objetivo diseñar máquinas que sean capaces de ejecutar tareas automatizadas o de simular el comportamiento humano o animal, en función de la capacidad del software.

Estas máquinas son los robots; podemos definir robot como sistema compuesto de mecanismos que le permiten hacer movimientos y realizar tareas específicas, programables y eventualmente inteligentes, valiéndose de conceptos de áreas del conocimiento como la electrónica, la mecánica, la física, las matemáticas, la electricidad y la informática, entre otras. Para el diseño de un robot se emplea ingeniería mecatrónica, la cual está compuesta por diversas ciencias, como son la mecánica, electrónica, la informática y la computación.[40].

Hoy, en nuestra sociedad actual, la robótica está teniendo un gran impacto y cada vez con mayor influencia en nuestra vida cotidiana, convirtiéndose incluso necesaria para satisfacer nuestras necesidades y realizar tareas que, podrían resultar prácticamente imposibles o dificultosas o muy repetitivas para el ser humano. Es por ello que la robótica llegó para realizar estas operaciones sin implicar la seguridad o la comodidad de la persona realizando todo tipo de tareas peligrosas, desagradables, repetitivas... Por consiguiente, sin duda, la robótica, hoy en día, es un miembro más en nuestra sociedad, ayudándonos en nuestros días. Además, la implementación de la robótica en nuestra sociedad obtiene mayor productividad, menor coste de implementación, la precisión y cumplimiento de los procesos, la flexibilidad en la capacidad y, sobre todo, en la reducción de los costes en todos los sectores existentes, ya sea desde el más local, como sería el hogar o desde el más complejo como sería en una empresa del sector de la automoción[44].

Cada vez más, las ciudades crecen más y con ellas, crecen las necesidades que se deben cumplir. Estas problemáticas complejas se pueden resolver a través de las TIC y es aquí donde entra la Robótica. Unas décadas atrás, se comenzaron

• E-mail: 1571461@uab.cat and joseantonio.corral@autonoma.cat
• Docente: Carles Pedret Ferrer, Departamento de Telecomunicaciones e Ingeniería de Sistemas
• Curso: 2021/2022

a realizar pruebas piloto con robots manipuladores como prueba de concepto en laboratorios de investigación y, más tarde, pasaron a ser usados para la industria manufacturera [18].

En los últimos 9 años, los gobiernos se han comenzado a dar cuenta de que el uso de la robótica, podrá ser una de las soluciones viables para la mejora de la calidad de vida de las personas, la contaminación medioambiental, la congestión del tráfico, la conectividad con la finalidad de dar soluciones efectivas y eficientes. Es por ello que, los gobiernos apostaron por su implementación en las ciudades. Grandes ciudades como Seúl, Tokio, Chenchén, Singapur, Dubai, Londres, San Francisco ya están realizando nuevas pruebas piloto para la robotización de las ciudades [33]. Algunos ejemplos, los podemos encontrar en: [21] :

- Japón donde usan la automatización con la finalidad de revitalizar la economía de la ciudad.
- Singapur, donde están realizando pruebas pilotos con el fin de mejorar los sistemas de gestión y control de la ciudad mediante la automatización de los servicios en hospitales y hoteles.
- Dubai que optó por una solución más ambiciosa mediante un prototipo de robotización de los servicios públcios con el objetivo de ser la ciudad 'más feliz del mundo' empleando los robots como vigilantes.
- Tokio que tiene como objetivo robotizar la ciudad para la revitalización de la economía, cultura y la demostración internacional.
- En Barcelona, se están realizando pruebas piloto del robot social para el cuidado de las personas mayores [20].

  La implantación de robots en prácticamente todos los ámbitos de la sociedad está avanzando vertiginosamente. Un selecto grupo de países lideran la instalación de robots que nos ofrecen un futuro apasionante, y son Estados Unidos, China, Japón, Corea del Sur y Alemania. Si bien hace unos años, únicamente podíamos ver la robótica implementada en el sector industrial automatizando puestos de trabajo, hoy en día los podemos encontrar en hoteles, bares, bancos, consultas médicas y entre otros [23].

1.1. Contextualización

Los orígenes remontan en la época de la Antigua Grecia, cuando Herón de Alejandría, en el 85 a.C, construyó el primer robot denominado autómata. Pero no fue hasta el 1937 cuando apareció Elektro que fue un robot humanoide de 2 metros de altura, 120 kg de peso, con capacidad de andar y pronunciar hasta 700 palabras[30].

La etimología del término robot, aparece por primera vez en una obra de teatro de Karel Capek para referirse a una empresa que él había inventado que fabricaba humanos artificiales. Actualmente, existen varios tipos de robots como los robots industriales, de servicio, domésticos o del hogar, médicos, militares, de entrenamiento, espaciales, educacionales, humanoides, estacionarios, de suelo, submarinos, aéreos, de microgravedad, artificieros y entre otros más[25].

1.1.1. La robótica industrial

George Charles Devol, nacido en 1912 en Kentucky, fue el primer inventor en crear un robot industrial. Años más tarde, junto a Joseph F. Engelberguer, originaron una empresa que se dedica al desarrolló de robots industriales. Y, en 1948, generó la primera máquina programable, fácil de usar y adaptable según convenio[17].

1.1.2. Robótica educativa

A través de la robótica educativa y el uso de referentes pedagógicos y didácticos, es posible apoyar los procesos de enseñanza y aprendizaje de la comunidad académica con herramientas tecnológicas. Desde la década de los setenta, se ha despertado un especial interés por los aportes que la robótica puede realizar a los procesos educativos. La "Robótica Pedagógica" es definida como una disciplina que permite concebir, diseñar y desarrollar robots educativos para que los estudiantes se inicien desde jóvenes en el estudio de las ciencias y la tecnología. Numerosas investigaciones demuestran el interés global por la inserción de herramientas robóticas en las aulas. Desde el año 1975, en la Universidad Du Maine, en Le Mans, Francia, donde aparece una primera utilización con fines educativos de la robótica; hasta la iniciativa que impulsó el fabricante líder de soluciones de impresión 3D, BCN3D Technologies en el año 2016, en colaboración con el Departament d'Ensenyament de la Generalitat de Catalunya, cuya estructura está impresa en 3D y su tecnología controlada por el software Arduino[40].

2 Objetivos y planificación

2.1. Objetivos generales

Los objetivos principales para llevar a cabo este proyecto están orientados a adquirir la familiarización de la robótica educativa, en concreto con brazos robóticos imprimibles y programables con librerías de código abierto, también conocidas como "frameworks"; así como aprender y conocer el funcionamiento del proceso de la impresión 3D, estrictamente necesario para el desarrollo del proyecto.

2.2. Objetivos específicos

Entre los designios referentes a la obtención de resultados demostrativos de la intencionalidad del proyecto, se encuentran:

- Impresión 3D de piezas de un brazo robótico manipulador Open Source.

- Obtención de los materiales requeridos para garantizar la correcta construcción y operatividad del brazo.
- Inspección de cada pieza asegurando que se han impreso correctamente y, en caso contrario, volver a elaborarla.
- Ensamblaje de las piezas impresas y la integración adecuada de los motores necesarios para garantizar la movilidad del mismo.
- Búsqueda y selección del Hardware más adecuado, las herramientas y librerías que implementaremos en parte de la programación.
- Diseñar un banco de pruebas y test para la comprobación del correcto funcionamiento del robot.

2.2.1. Planificación - Diagrama de Gantt

Para la organización de trabajo, se ha creado un diagrama temporal o diagrama de Gantt [31] que se podrá apreciar en la siguiente figura. Esta herramienta nos permitirá planificar las tareas exponiéndolas en un diagrama de manera gráfica.

- Cumplimiento de las tareas asignadas en un marco temporal fijado.

Fig. 1: Diagrama de Gantt

3 ESTUDIO DEL ESTADO DEL ARTE

3.1. Robots código abierto imprimibles en 3D

Existen diferentes tipos de robots imprimibles en 3D con código abierto. Algunos de ellos son los siguientes:

- **"Papa Bot"**: Tiene como función la transportación de pequeños objetos de un lugar a otro, pero, con muy poca carga. Incluye tres grados de libertad [26].

  - **Ventajas**: La principal ventaja es su bajo coste, alrededor de los treinta euros; especialmente habilidoso cuando se trata de objetos menudos.
  - **Desventajas**: Únicamente dispone de tres grados de libertad, lo que reduce notablemente su rango de movimiento y, en consecuencia, su funcionalidad.

Fig. 2: Papa Bot, por Elshasho, 2019. - Fuente: Thingiverse

- **EEZY BotMK2**: Tiene una carga útil aceptable y con 4 grados de libertad [28].

  - **Ventajas**: Relación calidad - precio balanceada, dotada de servos capaces de transportar elementos más pesados y con un mayor rango de movimiento, valiéndose de sus cuatro grados de libertad; muy accesible en cuanto a información se refiere.
  - **Desventajas**: Las piezas requeridas para su montaje son difíciles de encontrar, hasta tal punto de existir la necesidad de modificar ligeramente el diseño del brazo por cuestiones de cerraja.

Fig. 3: EEZY BotMK2, por daGHIZmo, 2016 - Fuente: Thingiverse

- **BCN3D Moveo** : Transporta objetos de hasta 0,5 kg y tiene cinco grados de libertad. Completamente, Open Source y, por ende, tendrá mayor accesibilidad [35].

  - **Ventajas**: Tiene la capacidad de transportar desde el objeto más ligero y con mayor dificultad de enganche, hasta objetos que logran alcanzar masas de hasta medio quilogramo; ostentando una movilidad y fluidez excelsa en el ámbito de la robótica.
  - **Desventajas**: Su principal desventaja es el coste, alcanzando los 400 euros; esta no es una desventaja al uso, sino que representa una barrera para los proyectos que dispongan de un presupuesto más ajustado.

Fig. 4: BCN3D Moveo, por BCN3D, 2022. - Fuente: BCN3D Moveo

- **Thor**: Fácilmente modificable y acorde a las necesidades que se tengan porque es de código abierto.[47].

  - **Ventajas**: Cuenta con seis grados de libertad y puede sostener hasta 750 gramos de carga; permite elevar sus cargas a grandes alturas y su precisión es excelente.

  - **Desventajas**: El gran número de piezas que lo componen son complejas de imprimir, lo que supone un gran adversidad a la hora de realizar el proceso de impresión.

Fig. 5: Brazo robótico Thor. Fuente: Escuela Ingeniería Electrónica

- **LittleArm**: Brazo robótico de pequeñas dimensiones con 3 grados de libertad [32].

  - **Ventajas**: Rápido de imprimir y sencillo a la hora de su montaje.

  - **Desventajas**: Demasiada sencillez lo que lo hace ser un modelo muy básico.

Fig. 6: Brazo robótico LittleArm. Fuente: Arduino Cloud

3.2. Tipos de software de edición y diseño

En el mundo del diseño y la edición, existen varios. Pueden ser gratuitos, de pago o incluso con licencia. A continuación, se mencionarán algunos de ellos:

Software privativo

- **Solidworks**, se trata de un software idóneo para aquellos usuarios que no sean expertos en el mundo del diseño 3D [43].

  - **Ventajas**: Perfecto para trabajar con figuras geométricas, valiéndose de herramientas que facilitan su uso y de una interfaz intuitiva.
  - **Desventajas**: Como todo software privativo, no es gratis, de hecho, la licencia más barata que se puede obtener es de 7000 euros; que se puede complementar con otros servicios de pago relativos a procesos de control y gestión, simulación, fabricación, marketing...

Fig. 7: Logo Solid Works - Fuente: SIEMENS, 2022

- **AutoCAD**: herramienta de diseño con una geometría digital excelente [15].

  - **Ventajas**: Se trata de un software muy preciso, además de ofertar una gran compatibilidad con un gran número de archivos.
  - **Desventajas**: Para poder sacar el mayor partido a este software, el usuario debe disponer de conocimientos previos relacionados con la geometría descriptiva; lo que hace que sea menos accesible.

Fig. 8: Logo AutoCAD - Fuente: Descargar Autocad 2022

- **Software de código abierto**
- **FreeCAD**: Software de modelado 3D de código abierto y gratuito[29].
  - **Ventajas**: Oferta compatibilidad con sistemas Windows, Linux y Mac; así como con un gran número de formatos como STEP, STL, SVG, DEX, etc.
  - **Desventajas**: Se puede llegar a no considerar una desventaja, sin embargo, este software debe iniciar el proceso que se vaya a realizar con un boceto en 2D, lo que puede ralentizar el trabajo.

Fig. 9: Logo FreeCAD - Fuente: 3D natives, 2022

- **Versiones online**
- **TinkerCAD**: De la família Autodesk, nació TinkerCAD y está basado en la geometría sólida como cubos y cilindros, entre otros. [45].
  - **Ventajas**: Viable tanto para aprendices como para educadores; con una interfaz intuitiva, se pueden construir estructuras de elevada complejidad y exportarlas en formatos como STL, OBJ y SVG; además de disponer de una biblioteca de archivos modificables.
  - **Desventajas**: A pesar de los excelentes resultados que se pueden obtener, este software carece de herramientas avanzadas para la realización de diseños igual de complejos que los que se pueden realizar con el resto de software; además de esto, al tratarse de una versión online, hay que trabajar con una permanente conexión a internet.

Fig. 10: Logo TinkerCAD. Fuente: Tinkercad, 2022.

3.3. Tipos de impresoras 3D

Existen diferentes tipos de impresoras. A continuación, se mostrarán las características, ventajas, desventajas y ejemplares:

Impresoras 3D por Estereolitografía (SLA): Se le debe introducir resina líquida, que, junto al láser, va creando capas. Suelen ser más limitados, ya que el filamento debe ser producto del fabricador de la empresa[38].

- **Ventajas**: Para piezas pequeñas y muy detalladas, esta, es la mejor opción, pues, permite una resolución muy elevada.
- **Desventajas**: Limitado, pues, requiere del filamento hecho por el fabricante. Proceso largo y requiere de limpieza posterior, lo que implica pérdida de tiempo. No resiste bien las altas temperaturas.
- **Ejemplos**: A continuación, veremos algunos modelos SLA: Form 3L Formlabs Impresora 3D SLA, Anycubic Photon Mono X UV resin SLA Impresora 3D, Anycubic Photon M3 Max Impresora 3D de Resina, Resolución 7K y Alta Precisión.

Fig. 11: Impresora 3D SLA. Fuente: Bitfab, 2022.

Impresoras 3D FDM: Extursion de Material: Similar al funcionamiento de una maquina de coser, va dando forma capa por capa durante el proceso de impresion y son las más usadas para usuarios amateur. Normalmente, suelen ser económicas y fáciles de manejar.

- **Ventajas**: Son económicas, fáciles de usar, adecuadas para cualquier lugar como una oficina, convellvan ahorro de costes y no son tan lentas en comparación a otras impresoras.
- **Desventajas**: Requiere una calibración minuciosa y los detalles de impresión suelen ser menores.
- **Ejemplos**: Prusa i3 MK3S y la Artillery Genius Pro-Impresora.

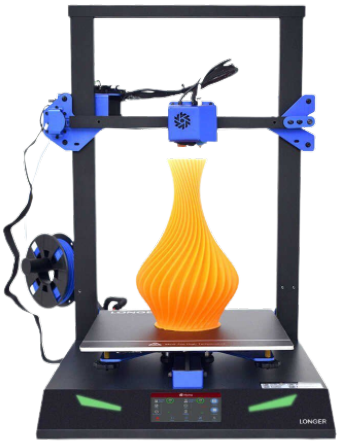

Fig. 12: Impresora 3D FDM. Fuente: Tresde

Impresoras 3D SLS: Sintetización Selectiva por Láser que lo utiliza para pequeñas partículas en polvo para convertirlas en estructuras sólidas que serán basadas en un modelo 3D.

- **Ventajas**: Para piezas pequeñas y muy detalladas, es la mejor opciñon pues permite una resolución muy alta en el detallado.
- **Desventajas**: Su impresión es limitada pues para funcionar, necesita filamento únicamente del propio fabricante. Los procesos son largos y requiere de limpieza con productos especializados una vez se haya completado la impresión. Lo que implica pérdida de dinero y tiempo. Por último, no resiste bien las altas temperaturas.
- **Ejemplos**: UniZ SLASH-Impresora 3D LCD, Fuse 1 Formlabs Impresora 3D SLS.

Fig. 13: Impresora 3D SLS. Fuente: DirectIndustry

3.4. Tipos de filamento

Se distinguen tres tipos de materiales de filamento [2]:

- **Básicos**: Considerados aquellos que son fáciles de imprimir y adecuados para cualquier impresora 3D. Además, suelen tener una buena adhesión entre las capas que se van imprimiendo y es por ello, que resultan ser la mejor opción para el ámbito educacional.
- **Exóticos**: Hace referencia a dichos filamentos que son relativamente difícil de encontrar y que, además, al no estar dentro del 'estándar' de las impresoras, añaden cierta dificultad a la hora de imprimir.
- **Alta ingeniería**: Son para un uso especializado en el sector industrial, como, por ejemplo, la automoción.

Asimismo, los filamentos más utilizados para cada clasificación [36]:

- **PLA**(ácido poli láctico), derivado de materias primas naturales y renovables como el maíz [19].
  - **Ventajas**: Fácil de usar a la hora de ejecutar la impresión, no requiere de una temperatura elevada para fundirse ni necesita que la cama de la impresora esté muy caliente, bastante estable, se podría considerar que la rapidez de su impresión, es más que otros materiales, tiene la capacidad de biodegradarse, ya que procede de materia orgánica e incluso, se puede reciclarlo.
  - **Desventajas**: Tiene problemas de resistencia térmica porque a partir de los 60 °C, ya se endeble. Tiene poca resistencia mecánica y es sensible a la humedad porque estropearía su estado [1].
  - **Consejos a tener en cuenta para la impresión**: La temperatura del extrusor debe estar aproximadamente cerca de los 198º, las piezas pequeñas y/o finas, deben ser impresas con ventiladores de capa para evitar su deformación. Se recomienda utilizar materiales para asegurar la adhesión de las piezas a la cama para garantizar que no se moverán y por ende, se imprimirán correctamente. Se pueden emplear colas especiales, lacas, cinta kapton... Y por último, para filamentos con color oscuro, aumentar la temperatura hasta 5 °C [16].
  - **Precio aproximado**: Dependiendo de la fuente, puede valer de entre 15€ hasta 25 € [3].

Fig. 14: Filamento PLA- Fuente: 3D Jake

- **PET** (Tereftalato de Polietileno): Es el tipo de plástico que se usa para los envases y botellas de agua, aceite...
  - **Ventajas**: Imprime con transparencia, resistencia elevada, impermeable y se puede hacer uso de él en el microondas sin que se derrita.
  - **Desventajas**: Tóxico, no biodegradable y endeble a partir de los 70 °C.
  - **Consejos a tener en cuenta para la impresión** : La temperatura debe estar entre (215 - 250)°C, si se quiere imprimir algo para uso doméstico, el filamento debe tener el certificado de uso. La impresión se puede proceder directamente sin esperar a que la cama se caliente y para los filamentos oscuros, se deberá dejar calentar un par de centígrados más.
  - **Precio aproximado**: Entre 15 € y 48 €.

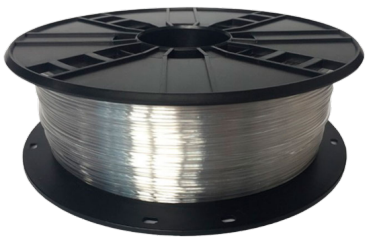

Fig. 15: Filamento PET- Fuente: Electrónica Embajadores

- **ABS** (acrilonitrilo butadieno estireno), polímero termoplástico común en el sector industrial [42].
  - **Ventajas**: Muestra capacidad de estabilidad en temperaturas altas, mantiene la tenacidad en temperaturas extremas, se puede lijar y/o forjar, lo cual lo hace flexible en el acabado final, muestra resistencia frente a productos, ataques químicos y/o impactos.
  - **Desventajas**: Se requiere de un cierto nivel de conocimiento y experiencia, dependiendo del objeto que se quiera imprimir, la dificultad puede ser elevada, durante la impresión, puede producirse el efecto Warping.
  - **Consejos a tener en cuenta para la impresión**: El extursor debe estar a 235 °C, la cama a 60 °C para pequeñas piezas y para mayores piezas a 80 °C. No usar ventiladores de capa. El proceso debe hacerse en una zona ventilada, pues, desprende gases que afectan nuestra salud. Y por último, para la impresión de filamentos oscuros, al igual que PLA, debe aumentarse unos pocos centígrados.
  - **Precio aproximado**: Entre 15 € y 20 €.

Fig. 16: Filamento ABS - Fuente: Amazon

- **PETG** (Tereftalato de Polietileno Glicol), termoplástico que combina la facilidad de impresión de PLA y la resistencia de ABS [27]
  - **Ventajas**: Duradero, las capas se adhieren con facilidad, no se produce Warping, no se presencian olores, fácil de usar, resistente y ofrece estabilidad térmica. Es la mejor opción si lo que se busca es transparencia.
  - **Desventajas**: A partir de los 80 °C, se reblance, suelta más hilo de lo que debería cuando se imprime una pieza, puede que las capas se desengancharan cuando los ventiladores de capa se ponen a la máxima potencia y se recomienda el uso de adhesivos como la laca.
  - **Consejos a tener en cuenta para la impresión**: La cama debe estar a 65 °C, el filamento se funde entre (220-250) °C dependiendo del fabricante. El ventilador de capa debe estar a una potencia baja-media. Y el flow, se debería reducir al 80-90 %.
  - **Precio aproximado**: Entre 16€ y 19€.

Fig. 17: Filamento PETG - Fuente: 3D filamentos Tienda Online

- **Nylon** (Polímero sintético), es una fibra textil elástica y resistente usado principalmente en la confección de tejidos, telas de puntos... [46]
  - **Ventajas**: El acabado de la impresión es suave, presenta buena capacidad de adherencia y resistencia. Además, el coeficiente de fricción es bajo.
  - **Desventajas**: Su conservación resulta ser costosa porque la humedad con facilidad lo puede estropear. Durante la impresión, los cambios de temperatura, puede provocar deformación en el material.
  - **Consejos a tener en cuenta para la impresión**: La temperatura debe estar a 240-260 °C (dependiendo de las indicaciones de la fábrica), la cama debe estar a 80 °C, recomendable usar adhesivos y no necesita de ventilador de capa.
  - Precio aproximado: Enter 30€ y 50€.

Fig. 18: Filamento Nylon - Fuente: 3D Print Filam

3.5. Tipos de archivos compatibles con impresoras 3D

Dependiendo del tipo de impresora 3D, acepta un tipo de formato u otro. A continuación se mencionarán algunos de ellos junto a sus ventajas y desventajas:

- ***STL***: Es el formato más común y que, además, fue de los primeros en aparecer. En él, se incluyen las coordenadas de los triángulos que forman la geometría 3D de la pieza que se quiere imprimir junto al color, forma y toda la información necesaria de la pieza.
  - **Ventajas**: El hecho de que sea tan conocido y popular, hace que muchas marcas lo incluyan y lo hagan compatible con sus máquinas impresoras 3D, es un formato ligero y fácil de compartir y es libre.
  - **Desventajas**: Una vez se abre el fichero para trabajar en él, pierde información relativa a la textura, color del diseño inicial, autor y materiales de uso.

- ***OBJ***: Es compatible con todos los software de diseño. Se usa mucho, pero no es tan conocido como STL. Tiene dos modos de funcionamiento:El preciso que conservará la geometría y el aproximado que generará una versión más simplificada del modelo.
    - **Ventajas**: Conserva la forma del modelo, la información sobre la geometría, textura e incluso de la malla original. El resultado es una malla lisa que simula mejor la superficie original.
    - **Desventajas**: Tarda más en la renderización, pues, el tamaño del archivo es más grande.
- ***3MF***: Se utiliza para diferentes programas de diseño de modelado en 3D. Incluye información relativa al modelo, el material, las propiedades de comprimidos y puede abrirse en diferentes programas como Microsoft 3D Builder, SolidWorks, Cura, CATIA, etc.
    - **Ventajas**: Permite almacenar colores, modelos y materiales. Los objetos son modificables y se les pueden añadir soportes
    - **Desventajas**: No contiene toda la información como la configuración individual [24].

3.6. Dispositivos Hardware

A continuación se darán a conocer las principales alternativas de dispositivos hardware que se encuentran en el mercado. Se ha decidido mostrar las opciones que presentaban mejor relación calidad-precio y compatibilidad, así como las que se adecuaban mejor a las necesidades que presentaba el proyecto. Algunos de los dispositivos Hardware existentes, se mencionarán a continuación: [34].

- **Raspberry Pi**: Es un ordenador de placa reducida de bajo coste, fabricado con la intención de ser una herramienta de enseñanza escolar. Los complementos básicos para montar una Raspberry son: 1 cargador de 1A/1,5A, 1 pantalla con entrada HDMI / pantalla con RCA, Teclado y ratón, Cable Ethernet y la Tarjeta SD. Precio medio de 45€ [39].

Fig. 19: Raspberry Pi - Fuente: Wikipedia

- **Arduino**: Hardware libre, capaz de detectar y controlar objetos del mundo real que usa el lenguaje de programación C o C + +. Usan diversos microcontroladores y microprocesadores, normalmente Atmel AVR. Complementos básicos para montar un Arduino necesitamos de un Puerto USN o puerto Barrel Jack de 2.5 mm que pueden ser programadas por el puerto serie a través de Bootloader Precio medio de 25€. [14]

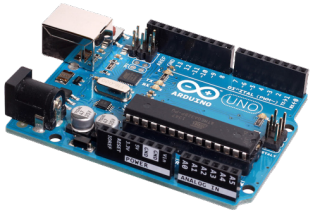

Fig. 20: Arduino - Fuente: Aprende a programar gratis

- **Netduino**: Plataforma de Microsoft para programación de microcontroladores, contiene un procesador Atmel (el mismo fabricante que los de Arduino). Esta placa es más potente que la de Arduino, pero esto se debe a que se programa en un lenguaje de más alto nivel (como C o VB.NET), se necesita más memoria y ciclos de procesador para el código. Esta plataforma integra un procesador Atmel SAM7X, 14 puertos digitales, 6 entradas analógicas y un PWM (Pulse Width Modulation).

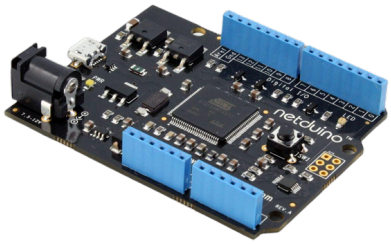

Fig. 21: Netduino - Fuente: OpenHacks

- **.NET Gadgeteer** es una plataforma basada en la idea de poder conectar de forma muy sencilla distintos dispositivos a nuestra placa base. A diferencia de las plataformas anteriores, para poder grabar los programas en el hardware, necesitaremos una placa llamada USB Client, conectada entre nuestro PC y el .NET Gadgeteer. Las placas más comunes son: GHI Electronics Fez Spider Mainboard, GHI Electronics Fez Hydra Mainboard y Sytech NANO Mainboard.

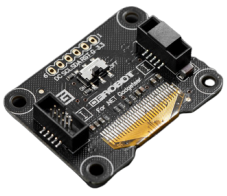

Fig. 22: NET Gadgeteer - Fuente: DFRobot

- **Teensy**: las capacidades de esta placa son impresionantes, con un procesador de 32 bits, 52 entradas/salidas totales, DAC, ADC, lector de tarjetas SD, etc. De hecho, los desarrolladores proporcionan un complemento software para una compatibilidad casi completa con los programas de Arduino. Precio medio de 50 €

Fig. 23: Teensy - Fuente: Amazon

4 METODOLOGÍA

A partir de la búsqueda exhaustiva sobre tipos de robots existentes imprimibles, el tipo de filamentos, las distintas impresoras 3D, los diferentes dispositivos Hardware, se elegirá una opción de cada grupo para construir y programar el brazo robótico.

Para cada elección, se elegirán opciones finalistas y se les puntuarán mediante criterios con el fin de seleccionar la mejor opción que sea eficiente, fácil y sobre todo accesible.

4.1. Elección del robot imprimible

La elección final se redujo a tres posibles finalistas que, al final, con una puntuación del 1 al 5, siendo 1 la puntuación más baja y 5, la puntuación más elevada, de acuerdo a nuestra evaluación propia.

En la siguiente figura se pueden ver los tres finalistas:

- **Little Arm** porque, el número de piezas era relativamente bajo en comparación a los otros y el precio es económico y bastante ajustable a nuestro presupuesto. Sin embargo, al presentar tanta sencillez en el diseño, tiene una movilidad muy reducida.
- **Bcn3D Noveo** Por el motivo de que es un modelo bastante complejo en cuanto a movimientos, diseño y funcionalidad, pero, el problema, la inversión que pide es muy elevada, al igual que la complejidad que muestra para construirlo.
- **EEZYbotARM MK2** por último, dicho robot presenta un grado de libertad elevado, aparentemente nos llamó la atención la facilidad de construirlo y, por último, el presupuesto para todas las funcionalidades que es capaz de realizar, obtiene muy buena nota.

De modo que, en la siguiente figura, se recogen las valoraciones:

	Little Arm	Bcn3D Noveo	EEZYbotARM MK2
nº piezas a imprimir	5	1	3
Grado de libertad	1	5	5
Precio aproximado	3	1	3
Puntuación	9	7	**11**

Fig. 24: Tabla que incluye puntaciones, fuente propia

De modo que, las calificaciones finales muestran que EEZYbotARM es el robot merecedor de ser construido por haber cumplido de manera equitativa nuestros requerimientos y condiciones.

4.1.1. Elección del material

Para la selección del tipo de filamento, se ha procedido a realizar de nuevo, el mismo proceso de selección: Se buscarán tipos de filamentos básicos y se hará una selección mediante puntuación según los criterios: Dureza, flexibilidad, durabilidad y facilidad de uso .

Los finalistas acabaron siendo:

- **PLA**:(ácido poli láctico), derivado de materias primas naturales y renovables como el maíz.
- **ABS**:(acrilonitrilo butadieno estireno), polímero termoplástico común en el sector industrial.
- **PETG**:(poliéster de glicol), termoplástico que combina la facilidad de impresión de PLA y la resistencia de ABS
- **Nylon**: (Polímero sintético), es una fibra textil elástica y resistente usado principalmente en la confección de tejidos, telas de puntos...
- **PC**:(policarbonato), uno de los filamentos más fuertes y resistentes, incluso con mayor nivel que Nylon. Utilizado en momentos en los que se necesita una gran resistencia, como por ejemplo un cristal a prueba de balas.

	PLA	ABS	PETG	Nylon	PC
Dureza	5	5	5	5	5
Flexibilidad	1	3	3	5	3
Durabilidad	3	5	5	5	5
Facilidad de uso	5	3	1	3	2
Total	14	16	14	**18**	15

Fig. 25: Tabla que incluye puntaciones, fuente propia

Según los resultados, Nylon, con un total de 18 puntos, es el que ha obtenido la puntuación máxima en comparación a los demás. Pese a ello, se descartó de inmediato, ya que la función de su filamento está mayoritariamente enfocada a otras distintas aplicaciones a lo que queremos construir. No obstante, se elegirá **PLA** y **ABS** pues, según varias fuentes y por experiencia propia, son los más fácil de manejar y los que requieren menos temperatura de calentamiento. Además, son los más adecuados en un ámbito educacional. En el OpenLab, se nos brinda hasta 90 gramos de filamento PLA de forma gratuita, a lo cuales, obviamente, deberemos añadirles más.

4.1.2. Elección de la maquinaria y software de impresión

La elección de la maquinaria, ha sido una dependencia total de la disponibilidad de las máquinas del espacio del OpenLab de la Universidad Autónoma de Barcelona. Por lo tanto, a lo largo de las sesiones en las que asistimos al OpenLAB, tuvimos la oportunidad de experimentar con 4 máquinas distintas:

- **Sigma R19**: Fabricada por BCN3D Technologies, es una impresora 3D fácil de usar con sistema de doble cabezal (arquitectura IDEX). Incluye interfaz gráfica por la cual se puede configurar y/o iniciar la impresión.
- **Sigmax R19**: Fabricada también por BCN3D Tecnhologies y es como el Sigma R19, pero con una versión más grande [**?**]. Estas dos últimas vienen dadas junto al Software gratuito BCNCura.

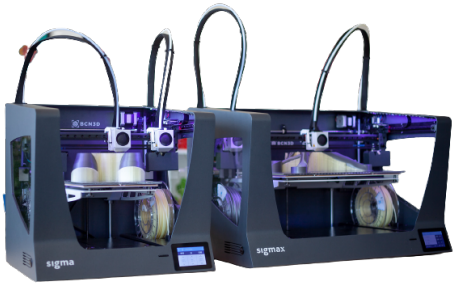

Fig. 26: Impresora Sigmax R19 Y Sigmax R19, 2022. Fuente:BCN3D Technologies

- **Prusa i3 MK3S+**, cuenta con un sensor SuperPINDA para mejorar la calibración de la primera capa [41]. La calibración y comunicación, se da a través del Software propio de la máquina llamado PrusaSlicer.

Fig. 27: Prusa i3 MK3S+ - Fuente:Prusa

- **Máquina cortadora láser**: Máquina que usa la energía térmica. Durante el corte, el láser concentra la luz sobre un punto levando su temperatura hasta que corta la pieza. Más adelante se explicará el porqué de la introducción de dicha máquina.

Fig. 28: Máquina cortadora láser - Fuente: Kaayak al Paint International

4.1.3. Elección del software de diseño

Se usará **TinkerCad** por su facilidad de uso y por el hecho de que sea online. Además, una vez se haya diseñado lo que hayamos querido realizar, el fichero final, puede ser un .STL que es el que necesitamos para poder usarlo en el software de la impresora para que posteriormente, se puedan modificar los parámetros más adientes a lo que queremos acabar imprimiendo.

4.1.4. Elección del Hardware

La elección se ha decantado por el hardware interno de **Arduino**; elección justificada al haber sido el propio Departamento de Electrónica de la Universidad el que nos la pudo proporcionar, siendo esta, una alternativa que nos permite la reutilización de un diseño ya existente y un sumo ahorro de costes. En nuestro caso, disponíamos de la placa Arduino Nano, cuyos elementos nos concedieron el suficiente número de pines para nuestra aplicación.

4.2. Impresión de las piezas

Para la impresión de las piezas, se ha recurrido a la red social llamada *'MakerBot Thingiverse'* en la que varios usuarios publican contenido acerca de creaciones propias de todo tipo de dispositivos. Navegando por la p agina web, nos encontramos a un diseñador mecánico llamado Carlo Franciscone, el creador de *EEZYbotARMMK2*. En su red social, incluía toda la información relativa al material necesario, la cantidad, las herramientas que necesitaremos, la configuración necesaria para la impresión. Asimismo, a continuación, se comentará con detalle el proceso de impresión, desde la descarga de los ficheros, hasta el montaje del brazo robótico.

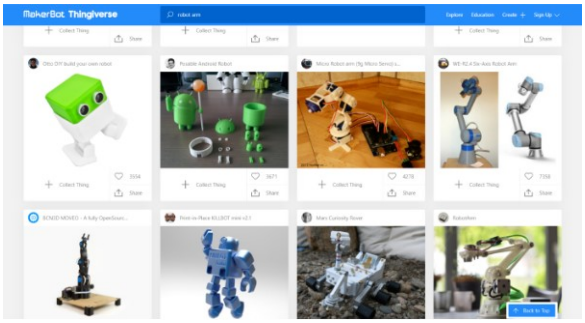

Fig. 29: Página oficila de makerbot Thingiverse

4.2.1. Programario y la descarga de ficheros

En la descarga de las piezas del brazo robótico EEZYbotARMMK2, se han encontrado un total de 22 ficheros STL. De tal forma que, lo único que deberemos hacer, será arrastrar los ficheros uno a uno distribuyéndolos en el espacio virtual del software. A continuación, discutiremos en qué clasificación se ha basado para realizar la correcta impresión pues no todas las piezas tienen tamaños iguales.

4.2.2. La impresión

Para esta etapa, se ha tenido que separar y clasificar según el tipo de pieza. Y por ello se ha tenido en cuenta las siguientes cuestiones:

- ¿La pieza necesita soporte para ser impresa?
- *¿La pieza es notablemente mayor?*
- *¿La pieza tiene una base lana?*

De modo que, a la hora de la impresión, las piezas que sean de un mismo género podrán ser impresas todas juntas sin causar ningún tipo de problema y en una impresora u otra dependiendo de la necesidad que requiera para imprimirse de forma correcta. Las que tengan soporte, mayoritariamente tardan más y no tendría sentido aplicarles soporte a aquellas que no lo necesitan pues sería una pérdida de tiempo y material.

En el OpenLab, tal y como se comentó anteriormente, para la impresión, se usarán tres impresoras diferentes: La Prusa, la Sigma R19 y la Sigmax R19. Y, dependiendo de si requerimos más precisión en el acabado, usaremos Prusa o Sigma R19 según si está libre cuando accedamos al OpenLab o no. Por último, se hará uso de la Sigmax R19 cuando se requiera de usar un espacio mayor en el que puedan caber todas las piezas con un tamaño más grande. Se ha hecho uso de tarjetas SD para poder realizar la inmediata conexión entre el software y la impresora.

En cuanto a la configuración, el fabricante nos indica la información que debemos tener en cuenta a la hora de la impresión. Pese a ello, se han modificado algunas, pues así la impresión tardaría menos en realizarse.

Llegados aquí, para poder realizar la correcta conexión entre los archivos que queremos imprimir y la impresora, se deberá adaptar y guardar en formato STL que es el que entenderá la impresora. Respecto a la configuración de la impresora, el fabricante recomienda dejar calentar la maquina hasta los 220º y en cama hasta los 60º, tal y como indica la siguiente imagen que a un está en proceso de calentamient.

4.2.3. Impresión de las piezas

Durante la primera sesión en el OpenLab, se ha decidido comenzar la impresión con todas aquellas piezas que tienen una **base plana** por la sencillez que implicaban a la hora de imprimirlas. En este inicial proceso, se utilizó el modelo de impresora Sigma R19 con el filamento PLA para imprimir las piezas que eran relativamente pequeñas. En el Software BCN Cura, se ha configurado que se el proceso de impresión se haga con una resolución del 0,3 de relleno, una vez configurado, se han ido arrastrando las piezas por todo el espacio del software que simula la cama de la máquina.

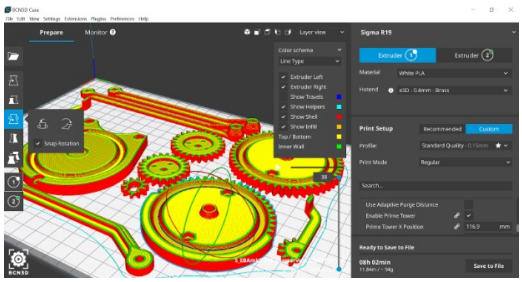

Fig. 30: Interfaz software BCN Cura y configuración de nuestras primeras piezas, fuente propia

Después de 8 horas de impresión, los resultados fueron los esperados que, a continuación se pueden apreciar:

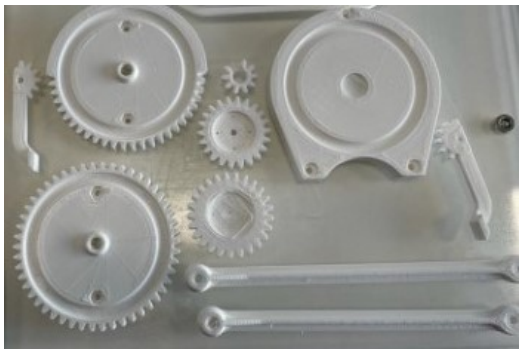

Fig. 31: Piezas impresas, fuente propia

Para la segunda sesión en el OpenLAB, se han elegido e impreso aquellas piezas que tienen base curvada o no sin base para que una vez hayan sido arrastradas al software, se las pueda configurar de manera que se impriman con base para que se estropee el filamento y por consiguiente, imprima mal la pieza. De modo que, se usará la máquina Sigmax R19 ya que tiene una cama mayor y en ella se podrá imprimir las piezas restantes que son las que pretenecen al cuerpo del robot.

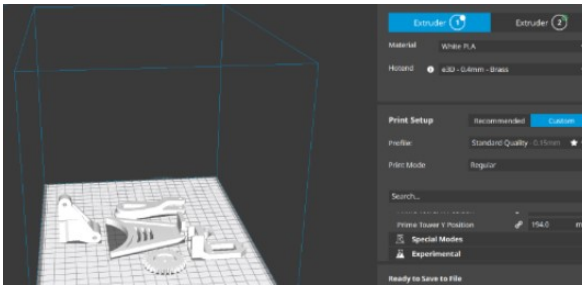

Fig. 32: Interfaz software BCN Cura y configuración del resto de piezas, fuente propia

Al cabo de 21 horas de impresión, el resultado fue el siguiente:

Fig. 33: Resto de piezas impresas, fuente propia

5 El Montaje del brazo

Para el montaje, se ha comenzado partiendo de la base que gran parte de ella viene ya viene impresa. De manera que, a esta base, se le ha añadido un servo en la parte inferior para que este tenga el control de movimiento en el ángulo de la totalidad del brazo. En esta parte, cabe destacar que se deberá abrir el servo desde la parte inferior para que pudiera caber:

Fig. 34: Colocación del servo, fuente propia

El siguiente paso consistirá en colocar la base sobre la base que sujetará el brazo. Se le añadirán dos servos que, por un lado, uno servirá para que el brazo se vaya moviendo hacia delante y hacia atrás mientras que el otro servirá para mover el brazo hacia arriba o hacia abajo.

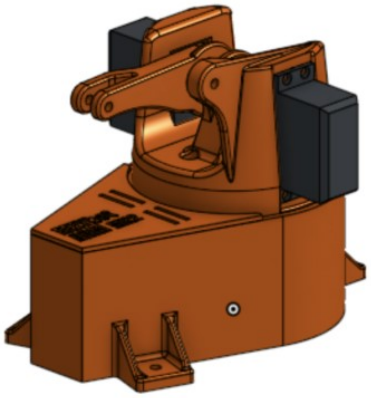

Fig. 35: Fijación de la posición de los dos servos a los laterales de la base - Fuente: Instructables, 2019

De manera que, el siguiente paso consistirá en colocar el cuerpo del brazo que accionará el movimiento.

Fig. 36: Montaje del brazo y antebrazo - Fuente: Instructables, 2019

5.1. Resultados del montaje

Después de tres horas de trabajo, el resultado del montaje del brazo robha sido el siguiente:

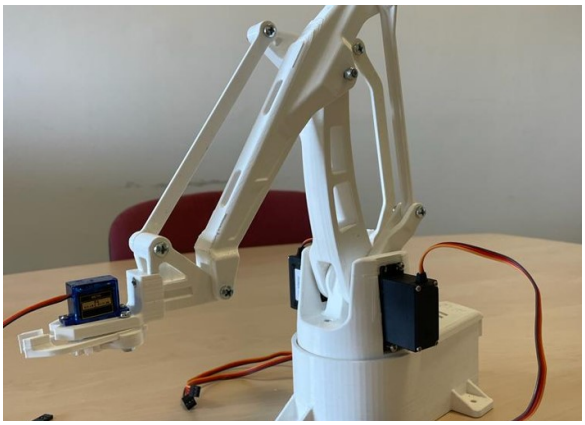

Fig. 37: Brazo montado, fuente propia

6 Diseño y modelaje del campo de pruebas

A partir del brazo robótico construido, se ha diseñado un campo de pruebas acorde a las capacidades de llegada y según su grado de libertad. El campo de pruebas consistirá en un tres en raya para probar que es capaz de mover piezas con exactitud según las ordenes que se le atribuyen. El tres en raya, es un juego en el cual su origen remonta a la lejana Persia harán unos 6 mil años. Este juego tan simple consiste en que cada jugador tiene cuatro fichas y deberá tratar de conseguir en diagonal o en horizontal tres fichas seguidas suyas antes que su contrincante.

Para el diseño del campo de juego, se ha usado Tinkercad pues, gracias a su facilidad de uso y de la biblioteca de figuras que tiene, agiliza aún más el proceso. Realizando pruebas de movimiento con el robot, nos fijamos en que no podría alcanzar nunca a coger las piezas si el tablero es totalmente plano de modo que sugerimos realizar un tablero con más altura. Pudimos comprobar, después de varios intentos, que el tablero debería tener al menos 6 centímetros de altura y que la distribución del cuadrado, debería ser de 12x12 para que todas las piezas cupieran a la perfección. De manera que el diseño creado con TinkerCAD ha quedado de la siguiente forma:

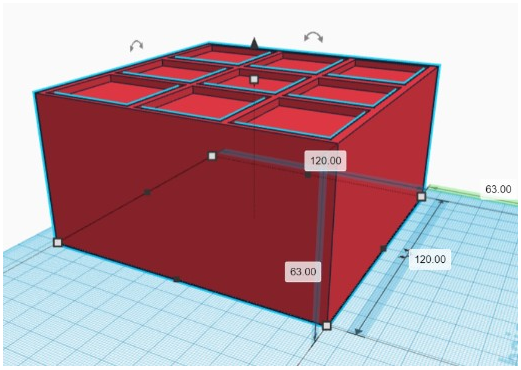

Fig. 38: diseño de campo de pruebas, fuente propia

La siguiente figura, tiene una altura de 6'3 cm y es porque, 6 cm hacen referencia a la altura de la figura mientras que los otros 3 milímetros restantes, son delimitar el campo de juego e irán en la parte superior a modo de dejar un pequeño hueco donde se colocarán las fichas. Pero, el hecho de imprimir una figura así de compacta y densa, se ha optado por imprimir solo la parte superior que delimita el posicionamiento de las fichas. El proceso de la parte inferior se explicará más adelante. A continuación, tenemos las rejas que delimitarán el campo de juego. Medirá 120 mm x 120 mm y de altura 3 mm:

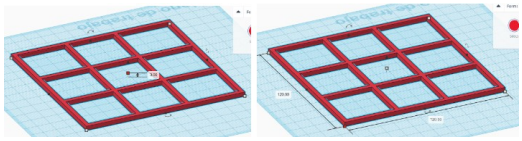

Fig. 39: diseño de las celdas separatorias, fuente propia

Las fichas por otro lado, al ver que la pinza no tendría mucha posibilidad de coger correctamente las piezas del juego, decidimos añadirle una base desde el centro hacia arriba de manera que, la pinza cogerá la ficha sin problema alguno. Como el juego será para dos jugadores, se usarán dos colores para distinguir equipos: El rojo para los círculos y el azul para las cruces. De modo que quedarán de la siguiente forma:

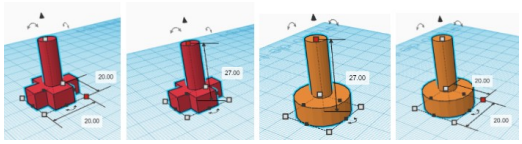

Fig. 40: Diseño de las fichas, fuente propia

6.1. Material necesario

Al principio se tenía la idea de imprimir todas las figuras en 3D, pero se ha elegido dividir el proceso. Para la base, se usará la cortadora láser y se aplicará en trozos de madera reciclados del OpenLab y que, a continuación, irán unidos. Pese a ello, la parte superior delimitadora se hará mediante la impresora 3D, pues requiere de una definición más acurada, al igual que las piezas restantes. Por lo tanto, el material necesario será:

- **Filamentos PLA** de color Rojo que se usará en la máquina Simga R19 junto software BCN Cura.
- **Filamento ABS** de color blanco que se usará en la máquina Prusa junto al software Prusa Slicer.
- **Cortadora láser** junto a su software.

6.2. Impresión de las piezas del campo de pruebas

Para la impresión de las piezas del campo de pruebas, se han usado dos máquinas diferentes para poder realizarlas a la vez. La primera máquina será la Sigma R19 que se encargará de imprimir las piezas del equipo de los circulos que se configurará el proceso de impresión a través de su propio software: BCN Cura con el filamento PLA. La segunda máquina, será la Prusa Slicer y se encargará de imprimir el equipo de las cruces y la reja del campo de pruebas con ABS que era el material que estaba disponible en ese entonces.

6.3. Corte de las piezas de madera

Para crear el cuadrado vacío, en el software se han dibujado un total de 5 piezas. De las cuales:

- La base medirá: *120 mm x 120 mm*
- Dos lados: *120 x 120 mm*
- Los otros dos lados restantes: *100 mm x 100 mm*

Las medidas respetan el modelo inicial, pero, en este caso, dos lados de la figura medirán menos pues, teniendo en cuenta que las maderas que nos han proporcionado miden 1 mm de ancho y, al ser dos, tendríamos que restar al total de la medición, 2 mm, por lo tanto = 120 mm – 20 mm = 100 mm. De modo que, una vez dibujadas las figuras en el programa, inciamos el proceso de corte en la cortadora láser:

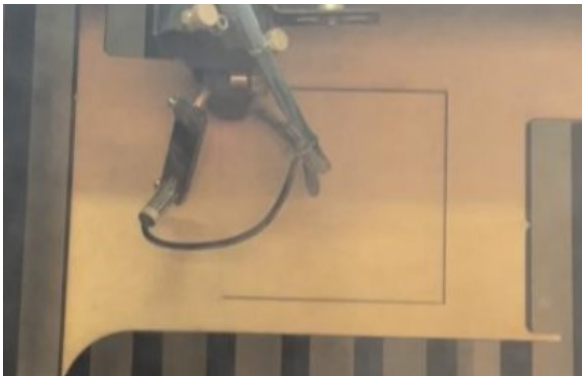

Fig. 41: Proceso de corte con cortadora láser, fuente propia

El proceso es muy rápido y duró aproximadamente poco más de un minuto. Finalizado el proceso de corte, el resultado ha sido el siguiente:

Fig. 42: Piezas cortadas, fuente propia

6.4. Resultado final

Para evitar que el robot una vez esté en funcionamiento se desvíe del campo de pruebas y por ende, obtener resultados no esperados, se le ha añadido una base çomún"de madera que une al robot a una distancia óptima al campo de pruebas.

El tablero, tal y como se puede apreciar a continuación, se le ha pintado la parte superior en blanco para dar un toque más profesional en el acabado.

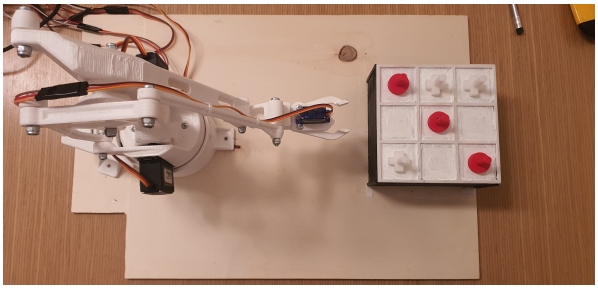

Fig. 43: Robot construido junto al campo de pruebas unido a través de la base comun. Fuente - Elaboración propia

7 La programación del robot

7.1. El Servomotor

Un servomotor, también denominado servo, consiste en un sistema compuesto por un motor eléctrico, un sistema de regulación que actúa sobre el motor y un sistema sensor que controla el movimiento de éste. Con él se pueden crear toda clase de movimientos de forma controlada y es la mejor opción para accionamientos rápidos y precisos. En este proyecto, se emplea para dotar a nuestro brazo robótico de la capacidad de moverse y trasladar objetos. Para nuestro caso de uso hemos seleccionado los siguientes servomotores, aquí detallamos también sus respectivas características técnicas:

Servomotores		
Características	**Servo MG946**	**Micro Servo SG90 9G**
Cantidad	1	3
Peso [g]	55	66
Dimensiones [mm]	40,7x19,7x42,9	12,4x8,7x3,4
Par	10,5kg/cm (4,8 V);13kg/cm (6V)	1kg/cm (4,8 V)
Velocidad	0,2 s/60º(4,8V);0,17s/60º(6V)	0,09s/60º(4,8V);0,08s/60º(6V)
Alimentación [V]	4,8-6,6	4,8-6
Recorrido	360º	120º ±

Fig. 44: Especificaciones del servo MG946 y Micro Servo SG90 9G,fuente propia

7.1.1. Funcionamiento del Servomotor

Como se ha especificado anteriormente, los servomotores se emplean para controlar de manera precisa la posición de distintos dispositivos; este control es de lazo cerrado, es decir, actúa como un bucle de su par, velocidad o posición del eje. Todo servomotor está compuesto por:

- **Motor eléctrico**: Genera el movimiento.
- **Sistema de control**: circuito electrónico que controla el movimiento mediante la técnica PWM (Pulse-width modulation).
- **Sistema de regulación**: formado por engranajes reductores; actúa sobre el motor para regular la velocidad y par del mismo.
- **Potenciómetro conectado al eje central**: Indica el ángulo en el que se encuentra el eje del motor.

El sistema de control recibe por el cable correspondiente a la señal de control los pulsos eléctricos enviados por la placa Arduino que determinan la nueva posición; A continuación, supervisa la posición actual a través del potenciómetro, y éste le devuelve una tensión proporcional al ángulo en el que se encuentra en ese momento, y calcula el error existente entre ambos voltajes (tensión de los pulsos y tensión de la posición). Si no hay diferencia entre estas dos tensiones, el motor está bien colocado y no recibe tensión; por el contrario, si la diferencia es distinta a cero, se le aplica al motor una tensión proporcional a la distancia que debe recorrer. Como hemos mencionado antes, se emplea la técnica PWM, un sistema de control basado en el ancho de pulsos que funciona de la siguiente manera:

- Cuando el ancho del pulso eléctrico es igual a 1,5 ms, el servo se encontrará centrado.
- Cuando sea inferior a 1,5 ms, girará hacia la izquierda.
- Cuando sea superior a 1,5 ms, girará a la derecha.

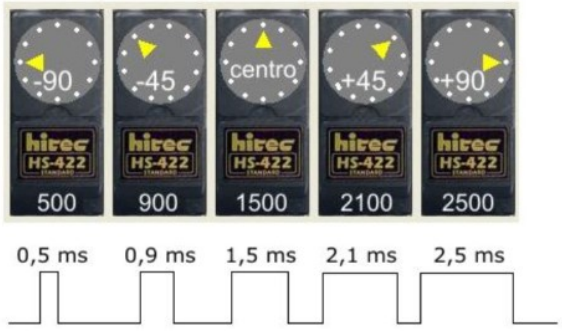

Fig. 45: Técnica PWM (Pulse-width modulation), por CRUZ-CARPIO, L. E. Ciencias de la Ingenieria y Tecnología T-VIII, 2018

7.2. Control del ángulo con Arduino y potenciómetro

El primer paso es seleccionar en la IDE de Arduino el tipo de tarjeta con el que vamos a trabajar, en nuestro caso, se trata de (Nombre placa); se selecciona el puerto USB donde se va a conectar la placa con el ordenador y se procede a realizar un breve código.

7.2.1. La explicación del código

A continuación se explicarán partes importantes en relación a la comprensión del código que se ha adjuntado más adelante en annexos:

- Se deben declarar variables globales que serán utilizadas en el programa, pero antes de eso, debemos llamar a un paquete dentro de la IDE de Arduino llamado Servo (#include Servo.h)[4], instrucción con la que se cargan comandos que trabajan solamente con este tipo de dispositivo; lo que recibe el nombre de **programación orientada a objetos.**
- Es necesario crear un objeto del tipo Servo y se declara una constante de tipo entero a la que se le asigna el pin analógico donde hayamos conectado nuestro potenciómetro; le sigue la declaración de dos variables de tipo entero: la que leerá el valor del potenciómetro y la que nos hará saber en qué posición está midiendo.
- Se debe indicar al programa donde está conectado el servo mediante la instrucción nombre_objeto_tipo_servo.attach(pin_cable_señal_servo).
- Para ejecutar la lectura de señal, se utiliza el comando variable_encargada_de_leer_el_valor_del_potenciómetro = analogRead (pin_potenciómetro)[5], cuyo valor se almacenará en esta variable.
- Posteriormente se empleará la función Serial.print()[6] para imprimir el valor medido.
- Para mostrar el valor del ángulo correspondiente con la posición en la que se encuentra el potenciómetro, se utiliza la rutina para "mapear"; leerá la señal de 0 a 1023 y un ángulo de 0 hasta 179; esto se mapea mediante el comando variable_ángulo = map(variable_potenciómetro, 0, 1023, 0, 179) y se utilizará nuevamente el comando Serialprint().
- Se debe indicar que los valores del ángulo los enviará al servo a través del comando nombre_objeto_tipo_servo.write(variable_ángulo)[7]; y se indica un delay para poder visualizar los datos.

7.3. Control de las articulaciones del robot

En este apartado trabajamos la antesala del objetivo del proyecto, es decir, comenzar a atisbar la funcionalidad del brazo robot. Para ello, hemos realizado una primera fase de pruebas en la que controlaremos el ángulo de los distintos servos mediante botones conectados a nuestra placa de pruebas, en consecuencia, tendremos el control de las 4 articulaciones de nuestro brazo; el proceso es similar al apartado anterior: Llamamos al paquete *Servo.h* y declaramos las variables globales de nuestros 4 servos. Declaramos las variables tipo *integer* para el almacenamiento de números enteros para los siguientes parámetros:

- Pines en los que hayamos conectado nuestros servos.
- Ángulos iniciales de los servos.
- Pines en los que hayamos conectado nuestros botones encargados de cambiar los ángulos.
- Estado de los botones – 2 botones por servo.

- Para especificar al microcontrolador los comandos que ejecutará en el momento de arranque debemos insertar la función void setup()[8] con las siguientes consignas:
  - Se debe indicar al programa donde están conectados los servos mediante la instrucción: nombre_objeto_tipo_servo.attach (variable_pin_cable_señal_servo).
  - Configuramos los pines digitales de los botones mediante la función pinMode(pin, mode)[9] para que se comporte como una entrada, concretamente con la etiqueta Input Pullup, la cual actúa usando la resistencia interna de Arduino para detectar la pulsación de nuestro botón, cuando esté presionado y cuando deje de estarlo.
  - Por último, emplearemos la función Serial.write()[10] para los ángulos de los servos que se reflejarán en nuestro puerto serie.
- Lo siguiente es definir la función void loop()[11], el punto de entrada de nuestro programa; aquí indicaremos los comandos que se ejecutarán mientras la placa Arduino esté habilitada; en nuestro caso estará compuesta por lo siguiente:
  - Igualamos los estados de los botones correspondientes a cada uno de nuestros servos a la función digitalRead(pin_digital_botón)[12], lo que hará que lea el valor del pin digital especificado y les asigne el valor a nuestras variables previamente declaradas.
  - Lo siguiente es iniciar una sentencia if para cada estado que determine lo siguiente: Mientras nuestro botón siga presionado, se incrementará el valor de nuestra variable a la que hemos declarado el valor del ángulo del servo; dentro de esta integraremos otra sentencia condicional que determinará los valores opuestos comprendidos en el rango de movimiento de los servos. * Con nuestros servomotores solo hemos alcanzado un recorrido completo de 70 grados. Finalmente, con la función serial.write(), escribimos el nuevo valor de los ángulos.
  - Y para finalizar nuestro programa, mediante la función serial.print(), especificamos las cadenas de caracteres que queremos asociar a los valores de los ángulos, los cuales también son imprimidos empleando la misma función, con el objetivo de que podamos visualizar de forma intuitiva en nuestro monitor serie toda la información relativa a los valores de los ángulos de cada uno de los servomotores.

- Las conexiones quedarían tal que así:

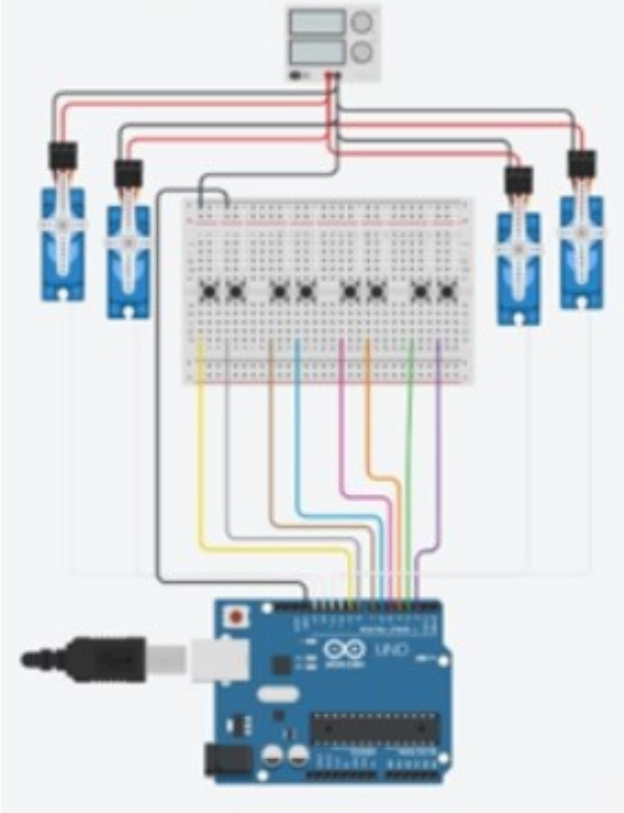

Fig. 46: Diagrama de conexionado, fuente propia

7.4. Programación de trayectorias

Este apartado se centra en mostrar el cálculo de trayectorias que seguirá nuestro brazo robot, alineado nuestro objetivo final, el de dotar de la capacidad de transportar los objetos que hemos diseñado a medida para las capacidades del mismo.

La programación en Arduino sigue el siguiente esqueleto:

- Llamamos al paquete Servo.h y declaramos las variables globales de nuestros 4 servomotores, especificando cada una de las articulaciones a la que corresponde cada uno de ellos, aspecto crucial para facilitar la tarea.
- Declaramos las variables tipo integer para los pines de nuestra placa Arduino donde hayamos conectado nuestros servomotores.
- Declaramos las variables tipo float, que disponen de una mayor resolución que las de tipo integer, para los siguientes parámetros: Variable “paso”: determina el número de “pasos” en los que se realizará el desplazamiento de cada uno de los servomotores; en nuestro caso, le asignamos el valor entero 20. Y la Variable “delta” para cada articulación: se encarga de almacenar la variación del ángulo de cada servomotor; las inicializamos a 0.
- Para especificar al microcontrolador los comandos que ejecutará en el momento de arranque, como ya hemos comentado en el apartado anterior, debemos insertar la función void setup(), esta vez con las siguientes consignas: Se debe indicar al programa dónde están conectados los servos mediante la instrucción nombre_objeto_tipo_servo.attach(variable_pin_cable_señal_servo), del mismo modo que en el apartado anterior. Configuramos el modo de los pines digitales de cada uno de los servos mediante la función pinMode(pin, mode) para que se comporten como salidas, esta vez especificándolo como OUTPUT.
- Establecemos la posición inicial de las articulaciones del brazo, para ello, llamamos cuatro funciones que se corresponden con sus respectivos servos y que se rigen por la siguiente lógica: Empleamos la palabra clave void para la declaración de cada una de las cuatro funciones, con lo que conseguimos que la función en cuestión no devuelva información a la función desde la que se le ha llamado; en nuestro caso hemos llamado a estas funciones: base, brazo, antebrazo y pinza, para los servos 1, 2, 3 y 4 respectivamente. A estas funciones les asignamos 2 variables internas de tipo integer[13] que determinarán la posición inicial y final del recorrido del servo en cuestión; estos son los parámetros que deberemos detallar en la función desde la que se llama; en nuestro caso han sido definidas como: “desde” y “hasta”.Dentro de cada función, con la sentencia for (inicialización; condición; incremento), declaramos que: para i = desde; i ¡= hasta; i++ –¿devuelva la posición (i) para cada servo con la función Serial.write() que ya conocemos.
- Definimos la función void loop(), donde indicaremos los comandos que se encargarán de traducir las trayectorias que seguirá el brazo, una vez se halle en la posición inicial:
- Llamaremos una función a la que le pasaremos las posiciones iniciales y finales de cada una de las articulaciones; con lo que lograremos que el brazo trace una trayectoria completa que no contemple brusquedades y que sea agradable al ojo humano, es decir, realizar un movimiento lo más parecido posible a como lo podría realizar una persona humana; este efecto se logra de la siguiente manera: Asignamos 8 variables internas a la función, referentes a las posiciones iniciales y finales que deberemos llamar de cada una de las articulaciones; nosotros hemos llamado a esta función con el nombre de “mover”. Dentro de esta función, definimos las cuatro variables que se encargarán de calcular las diferencias de posiciones (“delta”) y dividirlas entre la variable general “paso” que habíamos declarado con anterioridad, para cada uno de los servos que se le asignan a cada articulación. Con la sentencia for (inicialización; condición; incremento), declaramos que: para i = 1; i ¡= paso; i++) –¿devuelva la posición (posición inicial + delta), la escriba en cada uno de los servos con la función Serial.write(), y que actualice la posición inicial esta última; con esto lo que logramos es que cada servo realice el mo-

vimiento en el mismo número de pasos (20), movimiento que se logra con la escritura del ángulo inicial más la variable interna “delta”, lo que comporta la variación de grados de cada uno de ellos.

8 TESTS Y ANÁLISIS DE LOS RESULTADOS

Para comenzar a realizar las pruebas, debimos hacer un trabajo previo relativo a la determinación de la posición correlativa de nuestro brazo y el tablero donde se encuentran las piezas que trata de transportar; este paso ha sido estrictamente necesario de definir antes de llevar a cabo el proceso de programación de trayectorias, ya que se trata de un componente definitorio sobre el que se arraiga la lógica que hemos aplicado en el apartado anterior. La distancia ortogonal entre el pie frontal de nuestro brazo y el centro de nuestro tablero son 150 mm.

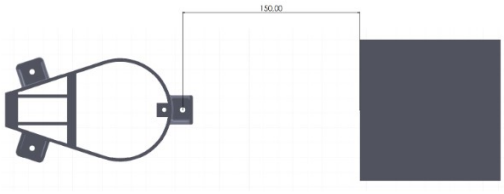

Fig. 47: Distancia entre brazo y tablero, fuente propia (AutoCAD)

Para comprobar si nuestro cálculo de trayectorias se alineaba con la respuesta que queríamos obtener, hemos decidido poner a prueba la capacidad de nuestro brazo robot, programando tres secuencias de movimiento distintas, que retan la operatividad del mismo, enfrentándolo a distintas fases en las que el nivel de exigencia aumenta para tal propósito.

Cada una de estas secuencias contiene un número distinto de desplazamientos (considerando un desplazamiento como el traslado de la ficha de una celda a otra de nuestro tablero); someteremos al robot a la realización de cada secuencia un total de 10 veces, variable fija que nos servirá de multiplicador para conocer el número total de desplazamientos realizados por cada bucle; el número que resulte de esta operación, será el número de desplazamientos que ha realizado en una única secuencia; este junto con el número de desvíos que hemos observado, son considerados como variable independiente y dependiente, respectivamente, y trasladados a un diagrama de dispersión, con el que valoramos qué tipo de relación subyace a estas dos variables, y cuantificar la intensidad de dicha relación.

Se han considerado desvíos, a los desplazamientos que ha efectuado el brazo, y como efecto, las fichas involucradas no han permanecido dentro de las celdas delimitadas en nuestro tablero de pruebas.

Para la representación de los desplazamientos, hemos contemplado a nuestro tablero como una matriz cuadrada de dimensiones 3x3, donde cada una de las nueve posiciones están referenciadas para explicitar el movimiento que asumirá la ficha en cada una de las siguientes secuencias.

8.1. Secuencia 1: desplazamiento lateral

La primera secuencia consiste en el desplazamiento de una única ficha desde la posición a1 a la posición a3 y regresar la ficha a la posición inicial (a1).

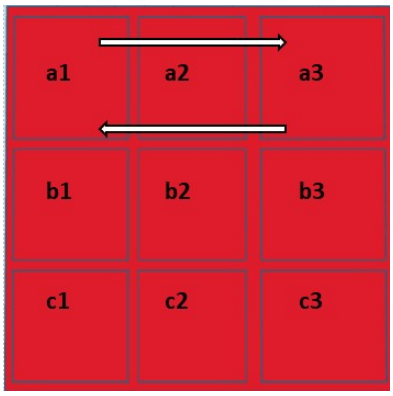

Fig. 48: Desplazamiento lateral a1,a3 / a3,a1, fuente propia

Este desplazamiento, como hemos comentado con anterioridad, se ejecuta en bucle hasta diez veces, con lo que obtenemos un total de 20 muestras de desplazamiento lateral. Cabe destacar que los datos que recogen las variables x e y, corresponden a la acumulación de desplazamientos y desvíos, respectivamente, por bucle; es decir:

$x_n = x^{n-1} + x, y_n = y^{n-1} + y$

n = número de bucle; x = número de desplazamientos realizados; y = número de desvíos observados

Los resultados obtenidos para esta primera secuencia son los siguientes:

Nº Bucle	Nº desplazamientos (x)	Nº desvíos (y)
1	2	1
2	4	1
3	6	2
4	8	2
5	10	2
6	12	2
7	14	2
8	16	2
9	18	2
10	20	2

Fig. 49: valores obtenidos de la primera secuencia, fuente propia

Con lo que obtenemos el siguiente diagrama de dispersión:

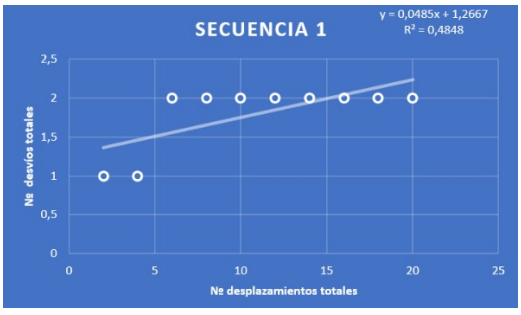

Fig. 50: Diagrama de dispersión secuencia 1, fuente propia

Para el análisis de la relación entre variables, hemos decidido integrar la recta de regresión cuya función que la aproxima es:

$$y = a + bx$$

Junto con el coeficiente de correlación lineal, determinado por la siguiente expresión:

$$r = \frac{n\sum x_i \sum y_i - \sum x_i \sum y_i}{\sqrt{n\sum x_i^2 - (\sum x_i)^2}\sqrt{n\sum y_i^2 - (\sum y_i)^2}}$$

Fig. 51: Expresión del coeficiente de correlación lineal,J.M.Domenech Roldan, Diagrama de correlación-dispersión

Los resultados de nuestro diagrama muestran una recta de regresión y = 0,0485x + 1,2667 y un coeficiente de correlación de R2 = 0,4848; con lo que interpretamos que existe una correlación positiva, ya que, a un crecimiento del número de desplazamientos, se observa una tendencia a crecer del número de desvíos, presentando, eso sí, un tipo de correlación débil, existiendo otras causas de dependencia, e incluso se puede llegar a considerar que no existe ninguna correlación evidente. Para justificarlo, nos apoyamos en el coeficiente obtenido, que toma valores comprendidos entre –1 y 1. Cuanto más próximo a 0 sea, menor es la relación entre datos, y cuanto más próximo a 1 (en valor absoluto), mayor será dicha relación. El valor que hemos obtenido nosotros (R2 = 0,4848), evidencia la conclusión que exponemos previamente, si bien el valor es más próximo a 0, no es concluyente.

8.2. Secuencia 2: desplazamiento diagonal

La segunda secuencia consiste en el desplazamiento de una única ficha desde la posición a1 a la posición c3 y regresar la ficha a la posición inicial (a1).

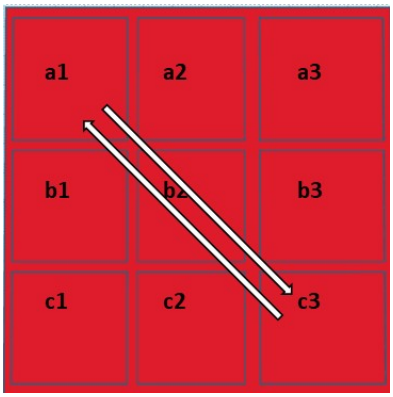

Fig. 52: Desplazamiento diagonal a1,c3 / c3,a1, fuente propia

Para este desplazamiento, obtenemos, al igual que en la secuencia anterior, 20 muestras de desplazamiento diagonal; los resultados obtenidos son los siguientes:

Nº Bucle	Nº desplazamientos (x)	Nº desvíos (y)
1	2	1
2	4	3
3	6	5
4	8	7
5	10	8
6	12	10
7	14	12
8	16	12
9	18	14
10	20	16

Fig. 53: Resultados obtenidos en la secuencia 2, fuente propia

Con lo que obtenemos el siguiente diagrama de dispersión:

Fig. 54: FDistancia brazo – tablero

La recta de regresión obtenida es y=0,8x, mientras que el coeficiente de correlación es de R2 = 0,9888; con esto observamos que, en este caso, existe una fuerte correlación positiva entre las dos variables, siendo el valor del coeficiente muy próximo a 1.

Esta secuencia, en comparación con la anterior, se muestra concluyente y expone valores que reiteran el aumento del número de desvíos acumulados, conforme aumenta el número de desplazamientos; no obstante, nuestra experiencia dicta convergencias entre los resultados y nuestra práctica. Esta conclusión nace de la observación, y si bien el diagrama muestra fielmente los desvíos que se han producido, no muestra la incongruencia de los mismos; en esta fase de pruebas el brazo no ha respondido de igual manera que en la fase anterior, haciendo caso omiso (cuando se cierra la pinza) del código ejecutado, moviéndose también otras articulaciones. Creemos que esto se debe a un **Following error**, basado en la diferencia entre la posición real que nuestro servo devuelve al controlador, y la trayectoria de posición y velocidad que este envía, es decir, un error de cálculo a nivel mecánico que afecta al comportamiento de nuestro brazo. Comprobamos esta teoría, ejecutando la función encargada de mover las articulaciones varias veces, con los mismos valores asignados; y observamos que, entre ejecuciones, el brazo realizaba desplazamientos que no se le había ordenado.

8.3. Secuencia 3: desplazamientos combinados

La tercera y última secuencia involucra, esta vez, a dos fichas, y consiste en combinar un desplazamien-

to diagonal, lateral y frontal de forma sucesiva; donde una de las fichas sufrirá dos de estos desplazamientos (diagonal y frontal) y la restante será trasladada definiendo una trayectoria horizontal. La secuencia es la siguiente: Ficha 1 de b3 a c2 – desplazamiento diagonal, Ficha 2 de b1 a b3 – desplazamiento lateral Y Ficha 3 de c2 a a2 – desplazamiento frontal

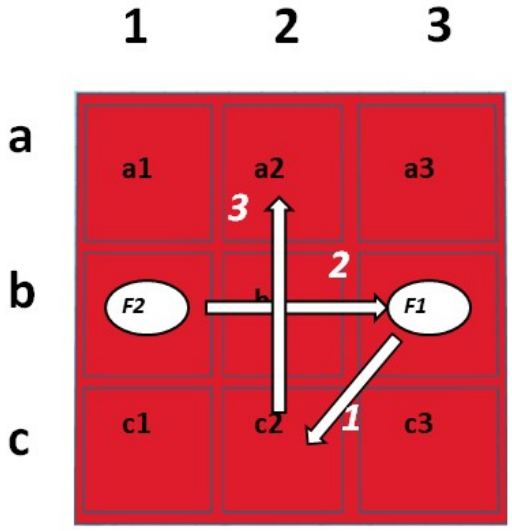

Fig. 55: desplazamiento diagonal (b3,c2), desplazamiento lateral (b1,b3), desplazamiento frontal (c2,a2), fuente propia

Al contrario que en las anteriores secuencias, para esta obtenemos 30 muestras de desplazamientos, diez para cada tipo de trayectoria; los resultados obtenidos son los siguientes:

Nº Bucle	Nº desplazamientos (x)	Nº desvíos (y)
1	3	2
2	6	3
3	9	4
4	12	4
5	15	4
6	18	4
7	21	5
8	24	5
9	27	6
10	30	6

Fig. 56: Resultados obtenidos en la secuencia 3, fuentre propia

Con lo que obtenemos el siguiente diagrama de dispersión:

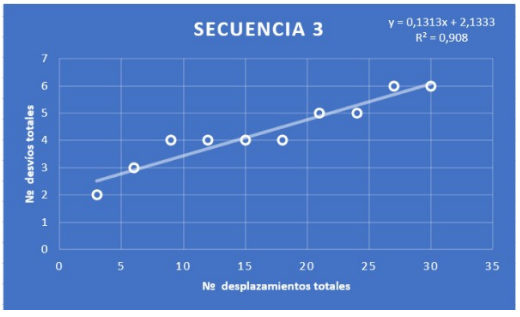

Fig. 57: Diagrama de dispersión de la secuencia 3, fuente propia

La recta de regresión obtenida esta ocasión es de y=0,1313x + 2,1333, y el coeficiente de correlación es de R2 = 0,908; apreciamos entonces, que existe una fuerte correlación positiva entre las dos variables, siendo el valor del coeficiente muy próximo a 1, al igual que en la secuencia anterior; no obstante, a pesar de realizar una combinación de tres trayectorias e involucrar a un mayor número de fichas, obtenemos un menor porcentaje de desvíos / desplazamientos (20), frente al porcentaje relativo al desplazamiento diagonal (80).

9 Conclusiones

Tras un análisis exhaustivo en relación a los robots existentes, los tipos de impresora, los tipos de material y filamentos, los distintos software de diseño y modelado, los diversos Hardware y entre otros temas que se han tratado, la impresión, pudo realizarse con resultados bastante buenos.

El hecho de habernos informado previamente acerca de todo lo necesario para la impresión, nos ha ayudado a proceder a la impresión del robot con total garantía. Lo cierto, es que en algunas ocasiones se nos han presentado pequeños problemas en los que la pieza no se imprimía como se esperaba, no se configuraba correctamente la impresora y/o en el software de la impresora no se le incluía la información adecuada y en consecuencia de esta, los errores de la mala adhesión del filamento sobre la figura, se arrastraban y no salía como se esperaba. Además, una de las razones por las que los errores se arrastrabana fue el no esperar a ver a como la impresora realizaba la primera capa ya que, iniciábamos la máquina y nos ibamos para volver al cabo d un par de horas al openLAB y para cuando llegamos, la pieza ya se realizó de mala forma. Y por ello se tuvieron que imrpimir varias veces algunas de las piezas que se imprimieron mal por lo cual, se nos alargó bastante el proceso.

Por otro lado, para el montaje, se tuvieron ciertos problemas pues, cuando imprimimos las piezas, las que tenían mayor tamaño, para no hacer que tarden un tiempo excesivo, se configuró la máquina de forma que el filamento salga más grueso de modo que la capa al ser más gruesa, tardaría menos. El problema de engruesar la salida del filamento, hace que la capacidad de detallado, disminuya y por ende, los pequeños abujeros por los que deberán pasar los tornillos, se acabaron cerrando por el arrastre de los filamentos. Para solucionarlo, en vez de volver a realizar todas las piezas de nuevo, se tuvieron que realizar pequeños trabajos de bricolaje manuales para que finalmente las uniones mediante tornillos, se realizaran correcta. Es por ello que para futuras operaciones de impresión, se deberá tener más en cuenta la configuración de la impresión y adecuarla para cada tipología de piezas. Las piezas, si tienen base curva, requerirán de un soporte (lo cual, obviamente, supondrá más tiempo en la impresión) en cambio las planas, no. Y, solo se deberá aumentar el grosor del filamento cuando la pieza sea grande pero sin detalles que ese fue uno de los principales problemas en el montaje.

Pese a ello, la etapa de impresión y montaje, resultó ser la idónea y la esperada.

En lo relativo al código, se trata de una solución que requiere de una dinámica de ensayo-error para poder pulir las trayectorias que ejecuta el brazo. Nuestra conclusión es que no hemos resuelto la completa operatividad y funcionalidad que pretendíamos en un principio, sin embargo, nos ha servido para conocer y analizar el comportamiento de los servos y su consiguiente reproducción del movimiento en las articulaciones dotadas de estos, para esta solución en particular; lo que nos ha ayudado a comprender mejor el funcionamiento del servomotor y las carencias que ha mostrado en este caso práctico.

En cuanto a las pruebas finales, llegamos a la conclusión de que, no es el incremento del número de desplazamientos, fichas, o trayectorias, lo que refleja el aumento en el número de desvíos, sino que, más bien, se trata del ya mencionado **Following error**, que como hemos podido observar, se ve agravado en trayectorias de mayor recorrido como en el desplazamiento diagonal (aproximadamente 16 cm), mientras que en la secuencia 1 y 3, no supera los siete desvíos (trayectos comprendidos en el intervalo de 5 a 12 cm aproximadamente).

De los objetivos que se propusieron al inicio del proyecto, se cumplieron el 100 % de lo planteado e incluso, se consiguió algún que otro logro que no se tenía pensado. Como por ejemplo, cuando aprendimos y nos familiarizamos con la impresión 3D, nos lanzamos a diseñar nuestro campo de pruebas e imprimirlo. Lo que no se cumplió, fue el plazo en el que se tenía planeado realizar las tareas en el Diagrama de Gantt pues el material que se necesitó para el desarollo del proyecto no llegó en el tiempo estimado, se imprimieron varias veces las piezas de forma incorrecta. En cuanto a la pretensión de dotar de una completa funcionalidad a nuestro robot, estamos más que satisfechos; si bien solo es capaz de transportar cargas ligeras de un lugar a otro, consideramos que hemos alcanzado el paso previo al desarrollo de un brazo autómata. Pese a ello, se pudo lograr con éxito el propósito inicial del trabajo: Imprimir un brazo robótico y programarlo para que realice una tarea en concreto. En nuestro caso, jugar al tres en raya.

9.1. Trabajos futuros

Uno de los trabajos que hemos decidido postergar ha sido el referente al control del brazo robot; consistiría en la automatización del robot, objetivo que hemos alcanzado, pero esta vez, dotándolo de sensores que recaben información a tiempo real, y este sea capaz de modificar su trayectoria en función de los valores obtenidos de su entorno. Por otro lado, contemplamos el control remoto empleando tecnologías bluetooth, con las que a través de una aplicación que descargaríamos en nuestro smartphone, seríamos capaces de manejar la posición del robot. Para ambas propuestas, resultaría mucho más efectivo que el programa que ejecutara el brazo, estuviera respaldado por la técnica denominada cinemática inversa, con la que se podría considerar el volumen del espacio que el robot podría alcanzar y colocarlo en una determinada posición y orientación; para ello, emplearíamos un lenguaje de programación usado principalmente en automatización, se trata de Gcode, mediante el cual podríamos decirle a la máquina que ejecute instrucciones sobre la posición a la que se tiene que desplazar, la velocidad y la trayectoria a seguir.

9.2. Valoraciones personales

El proceso de impresión, montaje y diseño, ha sido una experiencia magnífica pues a medida que se iba avanzando en ello, aprendíamos cada vez más hasta llegar al punto de conocer tan bien el funcionamiento que ya pudimos trabajar de forma autónoma. Si que es cierto que resultaba algo frustrante dejar una pieza durante más de 20 horas imprimirse y que al día siguiente cuando pasas a recogerla te encuentras que el error se ha ido arrastrando y no se imprimió correctamente.

En lo referente al proceso de programación, ha sido un proceso tedioso, pues había múltiples factores que podían alterar su comportamiento, tales como: el desplazamiento que sufría el propio brazo respecto del tablero de pruebas, causado por la tosquedad con la que se mostraban ciertos movimientos; la desprogramada concurrencia con la que el brazo y el tablero completo han tenido que soportar, refiriéndonos a que cuando teníamos el robot completamente montado y operativo, tuvimos que recurrir a métodos precarios para poder seguir avanzando con la fase de pruebas, pues se disponía de las fichas, la programación estaba completa, pero el tablero, en cambio, no lo estaba, lo que ha repercutido en la ralentización del proceso; el hecho de que se aflojaran los tornillos que sujetaban la pinza durante el proceso de pruebas; la limitación del rango de movimiento que presentan los servos, apareciendo en sus especificaciones unos valores que no ha logrado alcanzar, etc.

AGRADECIMIENTOS

En primer lugar agradecernos mutuamente la implicación que tuvimos a la hora de realizar el proyecto pese a las complicaciones por falta disponibilidad, pudimos lograrlo. Esto, tampoco pudo haber sido posible sin la ayuda de nuestro tutor que nos brindó sobre todo, de su soporte principalmente en la etapa del montaje y en la etapa inicial de la programación. Cabe destacar que tampoco nada habría sido posible sin el OpenLAB de la Universidad Autónoma de Barcelona. Agradecer a Pablo y Ana que nos ayudaron en todo momento y nos resolvieron todas las dudas en relación a las máquinas de impresión, la máquina cortadora láser y entre otros.

REFERENCIAS

[1] ABAX. Pla y otros materiales de impresora 3d: Características de filamentos, 2022.

[2] ALL3DP. Los mejores filamentos para impresoras 3d de 2021, 2022.

[3] AMAZON. Filamentos pla, 2022.

[4] ARDUINO. Referencia idiomática, 2022.

[5] ARDUINO. Referencia idiomática, 2022.

[6] ARDUINO. Referencia idiomática, 2022.

[7] ARDUINO. Referencia idiomática, 2022.

[8] ARDUINO. Referencia idiomática, 2022.

[9] ARDUINO. Referencia idiomática, 2022.

[10] ARDUINO. Referencia idiomática, 2022.

[11] ARDUINO. Referencia idiomática, 2022.

[12] ARDUINO. Referencia idiomática, 2022.

[13] ARDUINO. Referencia idiomática, 2022.

[14] ARDUINO, A. Hardware arduino para la educación, 2022.

[15] AUTODESK. Autocad: el software de cad 2d y 3d en el que confían millones de usuarios para dibujar, crear y automatizar diseños en cualquier momento y lugar, 2022.

[16] BCN3D. Pla: Trucos y consejos, 2022.

[17] BIG, T. George devol, el creador de la robótica industrial, 2018.

[18] CASTILLO, J. I. "ciudades inteligentes y la robÓtica: Perspectivas". *IEEE 1*, 1 (2019), 1.

[19] CIENTÍFICOS, T. Ácido polilÁctico (pla), 2022.

[20] CUIDADORA, C. "se amplía la prueba piloto del robot social para el cuidado de las personas mayores". *Ajuntament de Barcelona 1*, 1 (2021), 1.

[21] DE FABRICANTES Y DISTRIBUIDORES, L. A. "ciudades robot: Tres prototipos urbanos para el futuro de las ciudades". *AECOC Innovation Hub 1*, 1 (2020), 1.

[22] DE ROBOTS, R. Robótica. qué es la robótica y para qué sirve, 2021.

[23] DE ROBOTS, R. Robótica. qué es la robótica y para qué sirve. *Revista de Robots 3* (2021), 1–2.

[24] DYNAPRO3D. ¿qué formato de archivo usan las impresoras 3d?, 2022.

[25] EINSROBOTICS. El origen de la palabra "robot", 2021.

[26] ELSHASHO. Brazo robotico / robotic arm - "papa-bot", 2019.

[27] FILM, D. P. Petg (copoliéster de polietilentereftalato glicol de extrusión), 2022.

[28] FRANCISCONE, C. Eezybotarm mk2, 2019.

[29] FREECAD. Freecad, 2022.

[30] GEOGRAPHIC, N. Inventos griegos, los autómatas de herón, 2018.

[31] HINOJOSA, M. A. Diagrama de gantt. *Producción, procesos y operaciones* (2003).

[32] INSTRUCTABLES. Littlearm v3 arduino robot arm, 2022.

[33] KOVACIC, M. "robot cities: Three urban prototypes for future living". *Singularity Hub 1*, 1 (2018), 1.

[34] LAZALDE, A., T. J. . V.-V. D. Modelos sostenibles y políticas públicas para una economía social del conocimiento común y abierto en el ecuador. *Buen Conocer-FLOK Society 10*, 1 (2015), 619–652.

[35] MOVEO, B. Bcn3d moveo – un brazo robótico de código abierto impreso en 3d, 2020.

[36] MÁSTONER. Los 10 mejores tipos de filamento para impresión 3d — 2020, 2022.

[37] NASER, A., AND CONCHA, G. Rol de las tic en la gestión pública y en la planificación para un desarrollo sostenible en américa latina y el caribe.

[38] NATIVES EL SITIO WEB DE LA IMPRESIÓN 3D, D. Sla: Impresión 3d por estereolitografía, ¡te explicamos todo!, 2022.

[39] PEDRET, C. Fonaments teórics i tècnics, 2022.

[40] PINTO-SALAMANCA, M. L., BARRERA-LOMBANA, N., AND PÉREZ-HOLGUÍN, W. J. Uso de la robótica educativa como herramienta en los procesos de enseñanza. *Ingeniería Investigación y Desarrollo 10*, 1 (2010), 15–23.

[41] PRUSA, P. R. J. Kit original prusa i3 mk3s+, 2022.

[42] RESINEX. Abs, acrilonitrilo butadieno estireno, 2022.

[43] SYSTEMS, D. Descubra solidworks®, 2022.

[44] TEIGENS, V., SKALFIST, P., AND MIKELSTEN, D. *Inteligencia artificial: la cuarta revolución industrial*. Cambridge Stanford Books, 2020.

[45] TINKERCAD. Autodesk tinkercad, 2022.

[46] TP. Tecnología de los plásticos - nylon, 2011.

[47] ÁNGEL L.M. Thor an open source 3d printable 6dof robotic arm, 2020.

GLOSARIO

- **Arduino**: Plataforma basada en una placa electrónica de hardware libre que incluye un microocontrolador.
- **Autómata**:Máquina que imita los movimientos de un ser animado
- **Automatización**: Procedimientos automáticos que se realizan en un proceso industrial
- **Biodegradable**: Capacidad de descomponerse en elementos químicos naturales mediante agentes
- **Bootloader**: Herramienta del software que determina la prioridad de los procesos
- **Brazo robótico**: Dispositivo programable que tiene la función de realizar tareas que se asemejan al brazo de un humano.
- **Cable Ethernet**: Es un cable de red que se encarga de conectar dispositivos con cable local a la red.
- **Diseño**: Acción que tiene la finalidad de crear algo de forma creativa que proyecta algo de forma estética.
- **Dispositivo**: Elemento que se encagra de realizar funciones.
- **Efecto Warping**: En el momento de la impresión 3D, cuando la pieza empieza a curvarse hacia arriba, a esto, se le llama efecto Warping.
- **Electricidad**: Parte de la física que estudia la electricidad. Es la forma de energía que produce efecto luminoso.
- **Electrónica**: Parte de la física que estudia los movimientos electrones y la acción de las fuerzas electromágneticas para usarlos en aparatos que reciben y transmiten informaicón.
- **Entrada analógica**: Magnitud que lee tensión o intensidad.
- **Entrada HDMI**: Conexión usada para la transmisión de datos no comprimidos. Además, permite la conexión entre diferentes aparatos electrónicos.
- **Etimología**: Ciencia que estudia el origen de las palabras.
- **Física**: Ciencia que estudia las propiedades de la materia y de la energía. Establece leyes que dan explicación a los fenómenos naturales.
- **Following error o error de seguimiento**: Medida que indica cuanto se ha desviado una lista de valores.
- **Frameworks**: Esquema que ofrece la base estructurada para la elaboración de un proyecto con objetivos específicos.
- **Geometría**: Rama de las matemáticas que estudia la forma, los puntos, las líneas, ángulos, planos de las figuras con la finalidad de medirlas.
- **Grado de libertad**: Son cada una de las coordenadas independientes que se requieren para la descripción del estado del robot.
- **Hardware**: Elementos físicos que forman un sistema informático.
- **Industrial**: Actividad económica relacionada con la industria.
- **Informática**: Conocimientos que se ocupan de la trata automática de la información a través de computadoras.
- **Inteligencia Artificial**: Programa basado en el aprendizaje continuo.
- **Interfaz gráfica**: Sistema que se puede controlar de manera intuitiva.
- **Internet de las cosas (IoT)**: Descripción de la red de objetos que incluyen sensores, software y entre otras tecnologías.
- **Matemáticas**: Ciencia que estudia la propiedad de los números y la relación que conllevan.
- **Materia orgánica**: Conjunto de células animales y vegetales descompuestas por la acción de microorganismos.
- **Materias primas**: Materiales obtenidos directamente de la naturaleza.
- **Mecánica**: Rama de la física que estudia el movimiento y equilibrio de los cuerpos.
- **Medio ambiente**: Conjunto de factores físicos y biológicos que rodean los seres vivos.
- **Microprocesador**: Pequeño procesador que integra un solo circuito.
- **Open Source** o código abierto: Software cuyo código fuente se encuentra disponible y accesible para todo el mundo de forma gratuita.
- **Partículas**: Cuerpo de dimensiones diminutas que forma la materia.
- **Pedagógico**: Ciencia que estudia el cómo educar y enseñar.
- **Pin analógico**: Permite conextar una tensión externa de referencia.
- **Placa base**: Componente que interconecta los demás componentes de un dispositivo.
- **Polímetro termoplástico**: Plástico que a altas temperaturas permite deformarlo para dar la forma que guste pues cuando se enfríe, se endurecerá.

- **Potencia**: Capacidad de realizar una acción para producr un efecto en concreto.
- **Potenciómetro**: Herramienta que se encarga de la medición del potencial eléctrico.
- **Procesador Atmel**: Procesador con tecnología avanzada para la memoria y la lógica.
- **Programación C**: Lenguaje de programación que indica a un computador cómo debe realizar una tarea en concreto.
- **Puerto digital**: Interfaz por la cual se puede enviar y recibir información.
- **Pulsos eléctricos**: Tecnología 'no térmica' que genera efectos.
- **PWM** (Pulse Width Modulation): Tiene la finalidad de variar la energía recibida por un dispositivo como apagar/encender un dispositivo.
- **Sensor**: Dispositivo que capta información de las magnitudes físicas como la variación de luz, la temperatura y entre otras.
- **Software**: Programas que permiten al ordenador realizar tareas concretas.
- **Tarjeta SD**: Tarjeta extríble con el fin de almacenar información.
- **Ventilador de capa**: Ventilador que se coloca en el cabezal de la máquina de impresión 3D.

A.1. Anexo

Presupuesto total del robot MK2 RoboticArm

Pressupost EEZYbotARM MK2			
Material	**Unitats**	**Preu/unit**	**Total**
Filamento PLA (ácido poliáctico)	1	17.35	17.35
BOM: MG946 metálico de 360º servo	3	11.28	33.84
BOM: Servo SG90	1	12.99	12.99
Tuercas autoblocantes M3 x 2, M4 x 9, M6 x 1	1	17.99	17.99
Tornillos M3X20 (X2), M6X25 (X1), M3X10, M4X40/30	1	15	15
Varilla roscada M4X60/32 (120MM, corta medida)	1	9.49	9.49
Rodamiento 606zz	10	5.02	1.99203187
Ball spheres 6 mm de diámetro	25	1.563477173	15.99
Arandelas M4	100	0.0499	4.99
		Total	**129.632**

Fig. 58: Presupuesto del robot EZZYbotArm

A.2. Anexo

Código Arduino

```
#include <Servo.h>
Servo servo1; //base
Servo servo2; //brazo
Servo servo3; //antebrazo
Servo servo4; //pinza
int pinservo1 = 10;
int pinservo2 = 11;
int pinservo3 = 12;
int pinservo4 = 13;
float paso = 10; // número de pasos que voy a establecer para el desplazamient
float delta_p = 0; // calculo de la variacion del angulo para cada servomotor
float delta_a = 0;
float delta_br = 0;
float delta_b = 0;
void setup() {
  servo1.attach(pinservo1);
  servo2.attach(pinservo2);
  servo3.attach(pinservo3);
  servo4.attach(pinservo4);
  pinMode(pinservo1, OUTPUT);
  pinMode(pinservo2, OUTPUT);
  pinMode(pinservo3, OUTPUT);
  pinMode(pinservo4, OUTPUT);

  // Posición incial
  mover(35, 35, 35, 35, 35, 35, 35, 35);

  delay(5000);
  Serial.begin(9600);

}
void loop() {
  // desplazamiento_lateral
  mover(25, 35, 20, 20, 40, 50, 33, 8); // Acercarse 1ra posición
  delay(1000);
  mover(35, 25, 20, 20, 50, 67, 8, 8); // Acercarse 1ra posición
  delay(1000);
  mover(25, 5, 20, 20, 67, 67, 8, 8); // Cerrar pinza
  delay(1000);
  mover(5, 5, 20, 20, 67, 40, 8, 33); // Regresar de la 1ra posición
  delay(1000);
  mover(5, 5, 20, 20, 40, 68, 33, 33); // Acercarse a la 2da posición
  delay(1000);
  mover(5, 25, 20, 20, 68, 68, 33, 33); // Cerrar pinza
  delay(1000);
  mover(25, 25, 20, 20, 68, 35, 33, 33); // Regresar de la segunda posición
```

Fig. 59: Código Arduino

```
delay(1000);
mover(25, 25, 20, 20, 35, 68, 33, 33); // Acercarse a la 2da posición
delay(1000);
mover(25, 5, 20, 20, 68, 68, 33, 33); // Cerrar pinza
delay(1000);
mover(5, 5, 20, 20, 68, 35, 33, 33); // Regresar de la segunda posición
delay(1000);
mover(5, 5, 20, 20, 35, 50, 33, 8); // Acercarse 1ra posición
delay(1000);
mover(5, 5, 20, 20, 50, 67, 8, 8); // Acercarse 1ra posición
delay(1000);
mover(5, 25, 20, 20, 67, 67, 8, 8); // Abrir pinza
delay(1000);
mover(25, 25, 20, 20, 67, 40, 8, 33); // Regresar de la 1ra posición
delay(10000);

// desplazamiento_diagonal

mover(25, 35, 20, 20, 40, 50, 33, 7); // Acercarse 1ra posición
delay(1000);
mover(35, 25, 20, 20, 50, 67, 7, 7); // Acercarse 1ra posición
delay(1000);
mover(25, 5, 20, 20, 67, 67, 7, 7); // Cerrar pinza
delay(1000);
mover(5, 5, 20, 20, 67, 40, 7, 33); // Regresar de la 1ra posición
delay(1000);
mover(5, 5, 20, 10, 40, 30, 33, 35); // Acercarse a la 2da posición
delay(1000);
mover(5, 5, 10, 12, 30, 40, 35, 38); // Acercarse a la 2da posición
delay(1000);
mover(5, 5, 12, 5, 40, 50, 38, 38); // Acercarse a la 2da posición
delay(1000);
mover(5, 25, 5, 5, 50, 50, 38, 38); // abrir pinza
delay(1000);
mover(25, 25, 5, 5, 50, 40, 38, 38); // Regresar de la segunda posición
delay(1000);
mover(25, 25, 5, 5, 40, 50, 38, 38); // Acercarse a 2da posición
delay(1000);
mover(25, 5, 5, 5, 50, 50, 38, 38); // cerrar pinza
delay(1000);
mover(5, 5, 5, 20, 50, 35, 38, 33); // acercarse a 1ra posición
delay(1000);
mover(5, 5, 20, 20, 35, 67, 33, 7); // acercarse a la 1ra posición
delay(1000);
mover(5, 25, 20, 20, 67, 67, 7, 7); // abrir pinza
delay(1000);
```

Fig. 60: Código Arduino

```
delay(1000);
mover(25, 25, 20, 20, 67, 40, 7, 33); // regresar a posición inical
delay(10000);

// desplazamiento en cruz

//desplazamiento 1 --> b3, c2 // diagonal

mover(35, 35, 35, 10, 35, 53, 35, 38); // acercarse a posición b3
delay(1000);
mover(35, 5, 10, 15, 53, 54, 38, 38); // acercarse a posición b3 y cerrar
delay(1000);
mover(5, 5, 15, 15, 54, 48, 38, 30); // trasladar ficha de b3 a c2
delay(1000);
mover(5, 5, 15, 15, 48, 43, 30, 27); // trasladar ficha de b3 a c2
delay(1000);
mover(5, 5, 15, 8, 43, 43, 27, 25); // trasladar ficha de b3 a c2
delay(1000);
mover(5, 25, 8, 8, 43, 43, 25, 23); // abrir pinza en c2
delay(1000);


// desplazamiento 2 --> b1, b3 // lateral


mover(25, 35, 8, 8, 43, 30, 23, 8); // acercarse a posición b1
delay(1000);
mover(35, 35, 8, 8, 30, 55, 8, 8); // acercarse a posición b1
delay(1000);
mover(35, 5, 8, 18, 55, 57, 8, 8); // cerra pinza en posición b1
delay(1000);
mover(5, 5, 18, 35, 57, 53, 8, 38); // trasladar ficha de b1 a b3
delay(1000);
mover(5, 5, 35, 15, 53, 53, 37, 37); // trasladar ficha de b1 a b3
delay(1000);
mover(5, 25, 15, 15, 53, 53, 37, 37); // abrir pinza en b3
delay(1000);


// desplazamiento 3 --> c2, a2 // frontal

mover(25, 25, 15, 15, 53, 43, 37, 27); // acercarse a posición c2
delay(1000);
mover(25, 25, 15, 8, 43, 45, 27, 23); // trasladar ficha de b3 a c2
delay(1000);
mover(25, 5, 8, 8, 45, 45, 23, 23); // cerrar pinza en c2
```

Fig. 61: Código Arduino

```
delay(1000);
mover(25, 5, 8, 8, 45, 45, 23, 23); // cerrar pinza en c2
delay(1000);
mover(5, 5, 8, 8, 45, 40, 23, 23); // levantar posición
delay(1000);
mover(5, 5, 8, 25, 40, 55, 23, 22); // trasladar ficha de c2 a a2
delay(1000);
mover(5, 5, 25, 20, 55, 67, 22, 22); // trasladar ficha de c2 a a2
delay(1000);
mover(5, 25, 20, 20, 67, 67, 22, 22); // abrir pinza en a2
delay(1000);
mover(25, 35, 20, 35, 67, 35, 22, 35); // regresar a posición inicial
delay(10000);


}

void mover(int pi, int pf, int ai, int af, int bri, int brf, int bi, int bf)
{
  delta_p = (pf - pi) / paso; // posicion incial - posicion final / paso --> pinza
  delta_a = (af - ai) / paso; // antebrazo
  delta_br = (brf - bri) / paso; // brazo
  delta_b = (bf - bi) / paso; // base
  //Serial.println("delta_p: ");
  //Serial.println(delta_p);
  //Serial.println("delta_a: ");
  //Serial.println(delta_a);
  for (int i = 1; i <= paso; i++) { // que haga los 10 pasos de 1 en 1
    servo4.write(pi + delta_p); // escribe en los servos el angulo incial mas el delta,
    pi = pi + delta_p; // actualiza la posicion del servo
    delay(10);

    servo3.write(ai + delta_a);
    ai = ai + delta_a;
    Serial.println("ai:");
    Serial.println(ai);
    delay(10);

    servo2.write(bri + delta_br);
    bri = bri + delta_br;
    Serial.println("bri:");
    Serial.println(bri);
    delay(10);

    servo1.write(bi + delta_b);
    bi = bi + delta_b;
```

Fig. 62: Código Arduino

```
    servo3.write(ai + delta_a);
    ai = ai + delta_a;
    Serial.println("ai:");
    Serial.println(ai);
    delay(10);

    servo2.write(bri + delta_br);
    bri = bri + delta_br;
    Serial.println("bri:");
    Serial.println(bri);
    delay(10);

    servo1.write(bi + delta_b);
    bi = bi + delta_b;
    delay(10);
  }
}
// posiciones iniciales de cada articulacion --> finalmente no se usa
void base(int desde, int hasta) {
  for (int i = desde; i <= hasta; i++) {
    servo1.write(i);
    delay(50);
  }
}
void brazo(int desde, int hasta) {
  for (int i = desde; i <= hasta; i++) {
    servo2.write(i);
    delay(50);
  }
}

void antebrazo(int desde, int hasta) {
  for (int i = desde; i <= hasta; i++) {
    servo3.write(i);
    delay(50);
  }
}

void pinza(int desde, int hasta) {
  for (int i = desde; i <= hasta; i++) {
    servo4.write(i); // Abrir
    delay(25);
  }
}
```

Fig. 63: Código Arduino