UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería

Diseño e implementación de brazo robótico cartesiano para línea de empaque en el Laboratorio de Diseño de Procesos de la Universidad del Valle de Guatemala.

Trabajo de graduación en modalidad de trabajo profesional presentado por Daniel Marco Pablo González Orozco para optar al grado académico de Licenciado en Ingeniería Mecatrónica

Guatemala,

2025

UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería

Diseño e implementación de brazo robótico cartesiano para línea de empaque en el Laboratorio de Diseño de Procesos de la Universidad del Valle de Guatemala.

Trabajo de graduación en modalidad de trabajo profesional presentado por Daniel Marco Pablo González Orozco para optar al grado académico de Licenciado en Ingeniería Mecatrónica

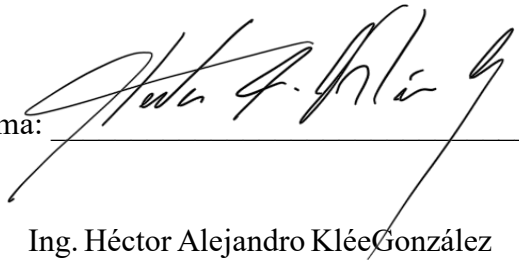
Guatemala,

2025

Vo.Bo.

Firma: 

Ing. Dulce María Chacón Muñoz

Firma: 

Ing. Héctor Alejandro KléeGonzález

Fecha de aprobación: 20 de noviembre de 2025

ÍNDICE

LISTA DE CUADROS

LISTA DE FIGURAS

LISTA DE ANEXOS

RESUMEN

En el presente trabajo se describió el proceso de diseño, desarrollo e implementación de un brazo robótico cartesiano con el cual se pretendió que los estudiantes aprendieran sobre los procesos de empaque en el Laboratorio de Diseño de Procesos de la Universidad del Valle de Guatemala. Este diseño de brazo robótico es una herramienta educativa y funcional, dentro de un entorno seguro y controlado, para eficientizar tareas repetitivas, incrementando la precisión, reduciendo el tiempo de operación y minimizando la intervención humana en procesos que requieren consistencia y rapidez.

El proyecto se desarrolló en etapas, con base en los requerimientos de funcionalidad y las necesidades operativas. Se elaboró el diseño mecánico utilizando el software Autodesk Inventor para la estructura del brazo con una base cuadrada desmontable para facilitar su montaje, transporte y mantenimiento dentro del laboratorio. Se emplearon materiales ligeros y duraderos, como perfiles de aluminio y elementos deslizantes con rodamientos, que garantizaron el desplazamiento en los tres ejes cartesianos (X, Y, Z), lo que facilitó el movimiento y la ubicación de productos durante el proceso de empaque. Asimismo, se seleccionaron motores apropiados, como servomotores y motores paso a paso, para cada caso de carga, velocidad y control de posición, buscando un balance entre eficiencia, costo y simplicidad de implementación. En cuanto al sistema de control, se utilizó Arduino Portenta *Machine Control*, seleccionado por su compatibilidad, robustez y soporte. Con este se programaron trayectorias simuladas en Matlab que representaran condiciones de trabajo reales. Por otro lado, durante este proyecto se aplicaron conceptos de diseño mecánico, electrónica, programación y control de sistemas, para realizar una integración de distintas áreas de la ingeniería Mecatrónica. Se aplicó una metodología organizada y se realizaron pruebas de validación, midiendo el rendimiento del brazo en aspectos como precisión, repetibilidad, rapidez de respuesta, adaptabilidad al entorno y estabilidad global en condiciones simuladas de funcionamiento.

Los resultados evidenciaron una mejora notable en la eficiencia de la línea de empaque del laboratorio, al disminuir errores humanos, reducir los tiempos de ciclo y aumentar la uniformidad del proceso. Por otro lado, el brazo robótico cartesiano demostró ser reproducible y escalable, facilitando su adaptación a diferentes contextos educativos, incluso en aplicaciones industriales. Además, sirvió como base para futuras investigaciones y proyectos de automatización más complejos dentro del entorno académico. El sistema se consolidó como una herramienta didáctica multidisciplinaria que contribuye al desarrollo de competencias técnicas y profesionales en los estudiantes universitarios, en concordancia con los objetivos académicos de la carrera de Licenciatura en Ingeniería Mecatrónica.

Palabras clave: brazo robótico cartesiano, automatización, diseño, sistemas de control, educación en ingeniería.

ABSTRACT

In the present work, the process of design, development, and implementation of a Cartesian robotic arm was described, with the aim that students could learn about packaging processes in the Process Design Laboratory at Universidad del Valle de Guatemala. This robotic arm design constitutes an educational and functional tool within a safe and controlled environment, aimed at improving repetitive tasks by increasing precision, reducing operation time, and minimizing human intervention in processes that require consistency and speed.

The project was developed in stages, based on functional requirements and operational needs. The mechanical design was created using Autodesk Inventor software for the arm structure, featuring a detachable square base to facilitate its assembly, transportation, and maintenance within the laboratory. Lightweight and durable materials were used, such as aluminum profiles and sliding elements with bearings, which ensured movement along the three Cartesian axes (X, Y, Z), facilitating the handling and positioning of products during the packaging process. Additionally, appropriate motors, such as servo motors and stepper motors, were selected for different load, speed, and position control requirements, seeking a balance between efficiency, cost, and ease of implementation. Regarding the control system, Arduino Portenta Machine Control was used, selected for its compatibility, robustness, and support. With this system, simulated trajectories were programmed in Matlab to represent real working conditions. Furthermore, throughout the project, concepts from mechanical design, electronics, programming, and control systems were applied to achieve an integration of different areas of Mechatronics Engineering. An organized methodology was applied, and validation tests were conducted to measure the arm's performance in terms of precision, repeatability, response speed, adaptability to the environment, and overall stability under simulated operating conditions.

The results showed a significant improvement in the efficiency of the laboratory's packaging line by reducing human errors, decreasing cycle times, and increasing process uniformity. Moreover, the Cartesian robotic arm proved to be reproducible and scalable, facilitating its adaptation to different educational contexts, including industrial applications. Additionally, it served as a foundation for future research and more complex automation projects within the academic environment. The system was established as a multidisciplinary teaching tool that contributes to the development of technical and professional competencies in university students, in alignment with the academic objectives of the Mechatronics Engineering program.

Keywords: Cartesian robotic arm, automation, design, control systems, engineering education.

I. INTRODUCCIÓN

La automatización de procesos industriales se ha convertido en una herramienta para la optimización de los sistemas de producción y empaque. Su objetivo es reducir el error humano, incrementar la productividad y mejorar la eficiencia de cualquier proceso. Actualmente, la integración de sistemas autómatas permite una mayor precisión y control en las operaciones industriales. La robótica industrial, como una de las ramas principales de la automatización, ha demostrado ser útil en entornos donde se requiere repetitividad, velocidad y confiabilidad operativa. En el Laboratorio de Diseño de Procesos de la Universidad del Valle de Guatemala, la incorporación de estas tecnologías mejoraría la experiencia educativa y haría posible la simulación de entornos industriales reales. La integración de herramientas automatizadas en el laboratorio no solo reforzaría el aprendizaje de los alumnos, sino que también permitiría poner en práctica conocimientos teóricos en proyectos reales que atiendan necesidades de la industria.

En este contexto, se propuso el diseño y desarrollo de un brazo robótico cartesiano, una solución mecatrónica que optimizó la línea de empaque automatizada del laboratorio. Un brazo cartesiano, definido por sus desplazamientos libres y rectos en los ejes X, Y y Z, proporciona una estructura sólida y exacta que simplifica la manipulación de elementos en trabajos repetitivos. Su estructura modular y su simpleza de programación lo hacen una elección excelente para aplicaciones educativas e industriales. El proyecto abarcó el desarrollo del sistema, contemplando el diseño mecánico de la estructura, la elección de motores y componentes electrónicos, la programación del sistema de control y su implementación. Priorizó el uso de tecnologías accesibles y versátiles que facilitaran su replicación en diferentes contextos, además de su funcionamiento en rutinas preestablecidas de forma autónoma.

Este trabajo aborda la explicación del proceso de diseño, construcción y evaluación del brazo cartesiano; el cual incluye los fundamentos teóricos, los objetivos y la metodología. Se buscó que, en su ejecución, desarrollaran habilidades técnicas, potenciando la investigación aplicada en el área de la automatización industrial y la robótica en el entorno académico.

II. OBJETIVOS

A. Objetivo general

- Diseñar e implementar un brazo robótico cartesiano, para su funcionamiento en la línea de empaque en el laboratorio de diseño de procesos de la Universidad del Valle de Guatemala.

B. Objetivos específicos

- Definir el diseño de un brazo robótico cartesiano para establecer las características estructurales y funcionales del sistema; mediante software de diseño asistido por computadora (CAD), criterios de funcionamiento mecánico, eléctrico y de control.
- Implementar el diseño de un prototipo físico de brazo robótico cartesiano, para ejecutar movimientos definidos en tres dimensiones; mediante la fabricación de componentes, ensamble de sistemas mecánicos, eléctricos y uso de microcontroladores.
- Evaluar el funcionamiento de prototipo de brazo robótico cartesiano, para verificar su precisión, velocidad y capacidad; mediante el uso y observación de tareas repetitivas en un sistema controlado.

III. JUSTIFICACIÓN

La incorporación de tecnologías de automatización en entornos educativos constituye una oportunidad fundamental para reforzar la capacitación práctica de los estudiantes en ingeniería. En particular, la utilización de sistemas robóticos como recursos educativos facilita la comprensión y aplicación de conceptos complejos vinculados al diseño, la operación y el control de procesos automatizados. Dentro de este contexto, la creación de un brazo robótico cartesiano emerge como una alternativa adecuada y coherente con las demandas del entorno académico contemporáneo.

Este proyecto adquiere una importancia particular al facilitar que los estudiantes se relacionen directamente con tecnologías que son ampliamente utilizadas en la industria, como motores paso a paso, controladores embebidos, software de diseño asistido por computadora (CAD) y entornos de simulación como Matlab. A través de esta interacción, se establece un vínculo entre la teoría enseñada en clase y la práctica en el terreno, ofreciendo a los estudiantes vivencias parecidas a las que enfrentarán en su carrera profesional.

Asimismo, el brazo cartesiano sugerido funciona como una plataforma interdisciplinaria de aprendizaje, donde se integran campos como el diseño mecánico, la electrónica, la automatización y la programación. Esta cualidad posibilita que los alumnos adquieran no solo habilidades técnicas, sino también competencias transversales como el trabajo colaborativo, el pensamiento crítico y la innovación.

Finalmente, la validación del sistema en condiciones simuladas de laboratorio permitirá evaluar su rendimiento, fiabilidad y flexibilidad. Esta vivencia no solo potenciará el proceso educativo, sino que también creará oportunidades para el avance de nuevas investigaciones, innovaciones tecnológicas e incorporación de funcionalidades futuras que aborden los desafíos de la automatización industrial.

IV. ALCANCE Y LIMITACIONES

A. Alcance

El presente proyecto tiene como objetivo el diseño, construcción e implementación de un brazo robótico cartesiano con fines académicos, destinado a mejorar la línea de empaque del Laboratorio de Diseño de Procesos de la Universidad del Valle de Guatemala. El sistema será implementado como una herramienta educativa que permita a los estudiantes experimentar de forma práctica con conceptos de automatización, control y diseño mecatrónico. El alcance del proyecto incluye:

- El diseño mecánico del brazo robótico cartesiano, considerando principios de robustez, modularidad y facilidad de mantenimiento.
- La selección e integración de componentes electrónicos y actuadores necesarios para su operación, incluyendo motores paso a paso, sensores y una unidad de control basada en Arduino Portenta *Machine Control*.
- El desarrollo del sistema de control básico que permita ejecutar movimientos punto a punto en los ejes cartesianos (X, Y, Z) y realizar tareas simples de manipulación.

B. Limitaciones

Para la realización del brazo robótico cartesiano, el proyecto presenta ciertas limitaciones derivadas del tiempo, los objetivos planteados en esta primera fase de implementación:

- Alcance funcional limitado: El brazo robótico está programado únicamente para ejecutar trayectorias básicas punto a punto. No se consideraron funciones avanzadas como visión artificial, control de fuerza o manipulación adaptativa.
- Capacidad de carga restringida: Debido al tamaño del prototipo y la capacidad de los actuadores seleccionados, el sistema solo puede manipular objetos de bajo peso, lo cual limita su uso a propósitos académicos o demostrativos.
- Falta de interfaz gráfica avanzada: Aunque se programó un sistema de control básico, no se desarrolló una interfaz de usuario completa ni conectividad industrial (por ejemplo, protocolo Modbus o comunicación con PLCs).
- No se realizó una caracterización dinámica completa: El análisis se centró en la cinemática directa e inversa, sin abarcar una modelación dinámica detallada ni análisis de vibraciones estructurales.
- Dependencia de una fuente de energía estable: El sistema requiere una alimentación eléctrica regulada, por lo que su uso en condiciones fuera del laboratorio puede verse

limitado sin la implementación de fuentes de respaldo o sistemas autónomos.

A futuro, se contempla abordar estas limitaciones mediante el desarrollo de nuevas versiones del prototipo, integrando tecnologías más avanzadas y expandiendo su capacidad funcional para uso en ambientes más exigentes o industriales.

V. MARCO TEÓRICO

La robótica cartesiana constituye una de las bases más simples y eficientes dentro de la automatización industrial y educativa. Su funcionamiento se basa en un sistema de coordenadas ortogonales (X, Y, Z), mediante el cual un actuador se desplaza linealmente sobre guías prismáticas. Esta arquitectura proporciona una relación directa entre el movimiento de los ejes y la posición del efector final, lo que facilita su análisis cinemático, su control y su integración en entornos industriales o académicos.

Los robots cartesianos se caracterizan por su facilidad de programación, precisión y rigidez estructural, siendo ampliamente utilizados en tareas de empaque, impresión, ensamblaje, soldadura o dosificación. Su sencillez mecánica y su bajo costo los hacen ideales para fines educativos, donde los estudiantes pueden comprender de manera práctica los fundamentos del diseño mecánico, la electrónica y el control de movimiento. Los motores utilizados en la automatización de sistemas robóticos pueden clasificarse en motores paso a paso, servomotores y motores de corriente continua. Los motores paso a paso son ideales para aplicaciones donde se requiere control preciso de posición, ya que permiten movimientos angulares discretos. Los servomotores, por otro lado, ofrecieron mayor velocidad y precisión mediante un sistema de retroalimentación que ajusta el torque y la posición en tiempo real. Finalmente, los motores de corriente continua son empleados en sistemas de menor precisión, donde se requiere un movimiento continuo sin posicionamiento exacto.

En cuanto a los sistemas de control, el uso de controladores programables (PLC) y microcontroladores como Arduino ha permitido la integración de algoritmos avanzados para la automatización de tareas. Estos sistemas permiten la programación de trayectorias, control de velocidad y respuesta a sensores en tiempo real, asegurando un funcionamiento eficiente del brazo robótico. El diseño de mecanismos robóticos también implica la selección de materiales adecuados para garantizar resistencia estructural sin comprometer la ligereza del sistema. Se utilizan perfiles de aluminio para la estructura debido a su combinación de rigidez y peso reducido, mientras que los componentes deslizantes se diseñan con cojinetes lineales para minimizar la fricción y mejorar la precisión del movimiento. Además, se revisó la literatura existente sobre el desarrollo de brazos robóticos cartesianos, analizando estudios previos sobre su implementación en líneas de ensamblaje y empaque. Se examinaron casos de éxito en la industria y en entornos académicos para identificar mejores prácticas y optimizar el diseño del presente proyecto.

A. Definición de robot

La idea de automatizar tareas ha acompañado al ser humano desde tiempos antiguos. A lo largo de la historia, distintas civilizaciones han desarrollado mecanismos que imitan movimientos humanos con fines prácticos o experimentales. En Grecia, durante el periodo helenístico, ya se construían autómatas con sistemas de poleas y pesos; siglos más tarde, en el

Renacimiento, relojeros e inventores europeos desarrollaron ingenios que podían simular movimientos animales o humanos. Sin embargo, fue hasta 1920 que el término "robot" se utilizó por primera vez, cuando el escritor checo Karel Čapek lo introdujo en su obra *Rossum's Universal Robots*, haciendo alusión a seres mecánicos diseñados para ejecutar tareas humanas (Čapek, 1920).

En el ámbito industrial, el concepto moderno de robot tomó forma con los trabajos de Joseph Engelberger y George Devol, quienes en 1956 fundaron la empresa Unimation e instalaron el primer robot en la planta de General Motors. Desde entonces, la robótica ha evolucionado rápidamente, pasando de simples manipuladores mecánicos a sofisticados sistemas con control numérico computarizado, visión artificial y capacidades de autoaprendizaje mediante algoritmos de inteligencia artificial (Val, 2009).

B. Clasificación de los robots industriales

Los robots industriales pueden clasificarse desde distintos criterios. Según su estructura mecánica, encontramos configuraciones como el robot cartesiano, cilíndrico, esférico, SCARA, articulado y paralelo. Cada una responde a necesidades específicas de espacio, precisión y tipo de tarea. El robot cartesiano, por ejemplo, está compuesto por tres ejes prismáticos ortogonales que permiten desplazamientos lineales a lo largo de los ejes X, Y y Z, lo que facilita su análisis y control cinemático. Esta configuración es ampliamente utilizada en tareas de posicionamiento preciso, corte, impresión 3D y ensamblaje automatizado (Val, 2009). También es posible clasificar los robots según el tipo de control: servo-controlados, que utilizan retroalimentación para ajustar su posición, y no servo-controlados, que operan con límites físicos; y por tipo de trayectoria: punto a punto, controlada y continua. Finalmente, su aplicación puede estar enfocada a soldadura, ensamblado, manipulación de piezas, pintura, entre otras tareas (Val, 2009).

C. Fundamentos de la robótica cartesiana

El robot cartesiano, también conocido como robot PPP (por tener tres articulaciones prismáticas), ofrece ventajas como alta precisión en trayectorias, estructura sencilla y facilidad para el análisis matemático. En la práctica, esta configuración de tres ejes lineales ortogonales permite movimientos rectilíneos en las tres direcciones del espacio, garantizando una correspondencia directa entre los parámetros articulares y las coordenadas del efector final. permite que el efector final se mueva de forma lineal en las tres dimensiones, simplificando el modelado cinemático y dinámico del sistema. Además, su diseño modular facilita la integración de herramientas como fresadoras, impresoras o cámaras de inspección (Val, 2009).

Desde el punto de vista didáctico, este tipo de robot permite la enseñanza práctica de conceptos fundamentales como cinemática directa e inversa, programación de trayectorias y control de motores, integrando conocimientos de mecánica, electrónica y software. En el proyecto actual, se plantea su uso como herramienta educativa dentro de una línea de empaque académica, combinando automatización industrial con procesos de enseñanza- aprendizaje.

D. Cinemática directa e inversa en robots cartesianos

El análisis cinemático constituye una de las bases fundamentales en el estudio de los manipuladores robóticos, ya que permite establecer la relación matemática entre los desplazamientos de las articulaciones y la posición del efector final dentro del espacio de trabajo. De acuerdo con (Craig, 2005; Tsai, 1999), la cinemática puede dividirse en dos partes principales: la cinemática directa, que determina la posición y orientación del efector final a partir de los parámetros articulares, y la cinemática inversa, que busca los valores de las articulaciones necesarios para alcanzar una posición deseada. En el caso particular de los robots cartesianos, el estudio se simplifica notablemente debido a que cada articulación prismática controla directamente uno de los tres ejes coordenados (X, Y y Z). Esto implica una relación lineal entre las variables articulares y las coordenadas del espacio cartesiano, lo cual elimina la necesidad de resolver sistemas no lineales o ecuaciones trigonométricas complejas.

Según Badillo Nájera (2016), en los robots de tipo PPP (prismático–prismático–prismático) la matriz Jacobiana corresponde a la matriz identidad, lo que indica que no existen singularidades dentro del espacio de trabajo en condiciones normales de operación. Este hecho convierte a la arquitectura cartesiana en una de las más estables y predecibles desde el punto de vista cinemático. Matemáticamente, la cinemática directa de un robot cartesiano puede expresarse como en la ecuación de la relación entre los desplazamientos articulares y las coordenadas cartesianas:

$$\begin{matrix} x \\ [y] \\ z \end{matrix} = \begin{matrix} q1 \\ [q2] \\ q3 \end{matrix}$$

Ya que q_1, q_2 y q_3 representan los desplazamientos lineales de los actuadores en los ejes X, Y y Z respectivamente (Tsai, 1999; Yoshikawa, 1990). Esta relación directa permite definir el espacio de trabajo como un volumen cúbico o paralelepípedo determinado por los límites físicos de cada eje. Por otro lado, la cinemática inversa busca calcular los valores articulares que permiten alcanzar una posición deseada (x_d, y_d, z_d). En un manipulador cartesiano, la solución es única y directa, ya que los desplazamientos requeridos en cada eje coinciden con las coordenadas objetivo, es decir:

$$q_i = x_i \; para \; i = 1,2,3$$

Como señala Val (2009), esta simplicidad en el cálculo de la cinemática inversa hace que los robots cartesianos sean ideales para aplicaciones educativas, pues permiten centrarse en la comprensión de trayectorias y control de movimiento sin la complejidad matemática de otros manipuladores. Estudios más recientes, como los de Osmanović et al. (2023) y Osmanović et al. (2025), han implementado modelos cinemáticos de robots cartesianos en entornos de simulación utilizando Matlab, confirmando que el comportamiento de estos sistemas puede predecirse con alta

exactitud incluso bajo perturbaciones externas. Dichas simulaciones permiten validar el desempeño del modelo y ajustar parámetros de control antes de la construcción física del prototipo.

Asimismo, Rojas et al. (2003) destacan que la correcta comprensión de la cinemática directa e inversa es esencial para el dimensionamiento de actuadores, el cálculo de torques requeridos y a programación de trayectorias punto a punto. En este sentido, la integración de herramientas computacionales como Matlab facilita la resolución numérica de las ecuaciones de movimiento y la visualización del comportamiento dinámico del efector final. Entonces la cinemática directa e inversa de un robot cartesiano se caracteriza por su simplicidad matemática y precisión geométrica, propiedades que la convierten en una excelente plataforma para la enseñanza de fundamentos de robótica, así como para el desarrollo de aplicaciones de automatización de bajo costo. La ausencia de singularidades, la relación lineal entre articulaciones y coordenadas, y la posibilidad de validar el modelo mediante simulación numérica refuerzan la utilidad de esta arquitectura tanto en el ámbito académico como en el industrial.

E. Componentes del sistema robótico

Un robot cartesiano está compuesto por una estructura mecánica que incluye guías lineales, husillos, motores y sistemas de transmisión. Los motores más utilizados en estos sistemas son los motores paso a paso, que permiten un control preciso del movimiento mediante trenes de pulsos. Estos motores requieren drivers que energizan sus bobinas de forma sincronizada, siendo comunes los controladores basados en los integrados DM542T (Val, 2009). Los sensores, por su parte, permiten detectar la posición del efector final, los límites de desplazamiento y condiciones del entorno. Entre los más comunes se encuentran los sensores ópticos de fines de carrera, sensores de efecto Hall, ultrasónicos y fotoeléctricos. Todos ellos están conectados a una unidad de control, en este caso, un microcontrolador como el Arduino Portenta, que procesa las señales y ejecuta las órdenes de movimiento mediante software embebido (Val, 2009).

F. Tipos de control aplicados en robótica cartesiana

El control es esencial para asegurar precisión y estabilidad en el movimiento de un robot. Los más empleados en sistemas cartesianos son:

- Control en lazo abierto: el controlador envía órdenes sin recibir retroalimentación. Es sencillo, pero propenso a errores por deslizamientos o inexactitudes.
- Control en lazo cerrado: incluye sensores o *encoders* que permiten ajustar la posición y velocidad.
- Control PID (Proporcional-Integral-Derivativo): combina tres acciones que corrigen el error entre la posición deseada y la real. El controlador PID se define como en la ecuación de término proporcional incrementando la respuesta ante errores de posición:

$$u(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{de(t)}{dt}$$

Donde K_p, K_i y K_d son las ganancias proporcional, integral y derivativa respectivamente. Este esquema es común en los sistemas basados en Portenta o PLCs por su estabilidad y facilidad de ajuste. La implementación de estos controles se puede simular en Matlab para ajustar las constantes antes del montaje físico.

G. Aplicaciones educativas y proyección académica

El uso de robots cartesianos en el entorno educativo permite la formación práctica en automatización, robótica, diseño mecánico, y control. Su estructura abierta, bajo costo y facilidad de programación lo convierten en una plataforma ideal para el aprendizaje por proyectos, integrando disciplinas como electrónica, informática y mecánica en un solo sistema. Además, al emplear lenguajes de programación accesibles y herramientas como Inventor, Matlab y Arduino, los estudiantes pueden experimentar el ciclo de desarrollo de un sistema mecatrónico completo. Desde una perspectiva institucional, la implementación de un brazo robótico cartesiano en el laboratorio académico fortalece la infraestructura de enseñanza y permite replicar procesos industriales. Esto facilita la transición del estudiante al ámbito profesional, además de abrir la puerta a nuevas líneas de investigación, extensión y mejora continua en el campo de la automatización.

H. Robótica cartesiana y automatización industrial

La automatización industrial ha transformado los procesos productivos al permitir una mayor eficiencia, repetitividad y calidad en la ejecución de tareas. Dentro de este campo, la robótica cartesiana destaca por su simplicidad estructural y precisión, características que la han hecho popular en aplicaciones como ensamblaje, impresión 3D, y líneas de empaque. Un robot cartesiano opera sobre tres ejes ortogonales (X, Y, Z), lo que facilita el control de su cinemática directa e inversa, además de reducir la complejidad mecánica. Esta estructura permite un volumen de trabajo regular y una excelente capacidad de manipulación de objetos dentro de espacios delimitados (Sellés, 2013).

I. Aplicaciones educativas de la robótica y proyección académica

En entornos académicos, los robots cartesianos representan una herramienta didáctica valiosa para la enseñanza de principios de diseño mecánico, automatización y control. La construcción y la programación de estos sistemas permite a los estudiantes enfrentar problemas reales de ingeniería, desarrollando así competencias técnicas y multidisciplinarias (Universidad Libre, 2017). Además, su estructura modular facilita la implementación de mejoras y adaptaciones por parte de otros grupos estudiantiles, lo cual es útil en universidades que fomentan el aprendizaje

basado en proyectos y la continuidad de trabajos académicos.

J. Diseño mecánico y modelado CAD

El diseño asistido por computadora (CAD) es una herramienta fundamental para el desarrollo de sistemas robóticos. Con programas como Autodesk Inventor, es posible modelar piezas, realizar ensamblajes virtuales, verificar interferencias y simular el movimiento del robot antes de su fabricación. Este enfoque permite optimizar la selección de materiales, el peso del sistema, y validar la rigidez estructural mediante análisis de esfuerzos. En el presente proyecto, se utilizó Inventor para modelar la estructura del brazo, los ejes lineales y los componentes impresos en 3D, garantizando su compatibilidad y facilidad de montaje (Sellés, 2013).

K. Simulación y análisis con Matlab

Matlab se utiliza ampliamente para validar trayectorias, realizar análisis estructurales y desarrollar modelos matemáticos de sistemas robóticos. Sus herramientas permiten trabajar directamente con matrices, facilitando el análisis de cinemática y dinámica, así como la implementación de algoritmos de control. Además, su integración con Simulink permite el desarrollo de entornos de simulación interactivos (Quispe Palomino & Palomino Ticona, 2021).

L. Sistemas de control con Arduino Portenta

El control embebido de sistemas robóticos puede lograrse mediante plataformas como Arduino Portenta, que combina la simplicidad del entorno Arduino con capacidades avanzadas como doble núcleo, conectividad inalámbrica y programación en alto nivel. Estas características lo hacen adecuado para sistemas que requieren control de múltiples motores, lectura de sensores y ejecución de rutinas complejas, todo en tiempo real. En el brazo robótico cartesiano se utilizó para gestionar motores paso a paso, procesar señales de sensores y ejecutar rutinas automáticas. Su integración con Matlab y su entorno de programación abierta facilitaron la creación de un sistema flexible y escalable.

M. Sostenibilidad, replicabilidad y potencial académico

Uno de los objetivos clave de incorporar un brazo robótico cartesiano en un entorno universitario es establecer una plataforma educativa robusta, replicable y escalable. Al utilizar materiales accesibles, tecnologías de código abierto y una documentación técnica detallada, se garantiza que el sistema pueda mantenerse operativo a largo plazo y sirva como base para futuras investigaciones, mejoras o adaptaciones desarrolladas por nuevos grupos estudiantiles (Universidad Libre, 2017). En el caso particular de este proyecto, el sistema será implementado en el Laboratorio de Diseño de Procesos de la Universidad del Valle de Guatemala, lo cual permitirá

enriquecer la enseñanza de asignaturas relacionadas con la automatización, la mecatrónica y el diseño industrial. Su incorporación no solo brindará a los estudiantes la oportunidad de interactuar con tecnología aplicada, sino que también contribuirá a replicar condiciones reales de trabajo dentro de un entorno académico, fortaleciendo así la vinculación entre teoría y práctica.

N. Comparativa de arquitecturas robóticas

El robot cartesiano se posiciona como una excelente alternativa para ambientes académicos y de automatización ligera, dada su simplicidad de construcción, facilidad de programación y mantenimiento económico.

Cuadro 1. Comparativa de arquitecturas robóticas

Tipo de robot	Ejes	Complejidad de control	Precisión	Área de trabajo	Aplicaciones comunes
Cartesiano	X, Y, Z	Baja	Alta	Cúbica	Empaque, Impresión 3D, *pick/place*
SCARA	4	Media	Alta	Cilíndrica	Ensamblaje electrónico
Articulado	6	Alta	Alta	Esférica	Soldadura, pintura, CNC
Paralelo (Delta)	3	Alta	Muy Alta	Cónica	Embalaje rápido, *pick/place*[1i]

Nota. Elaboración propia.

O. Propuestas de mejora y evolución del sistema

Con base en la implementación actual, se identifican varias oportunidades de mejora y expansión del prototipo, las cuales pueden desarrollarse en futuras etapas de trabajos de cursos o tesis:

- Integración de visión artificial: con cámaras y algoritmos OpenCV para reconocimiento de objetos.
- Sistema HMI mejorado: mediante pantalla táctil o interfaz web para seleccionar trayectorias y monitorear en tiempo real.
- Retroalimentación de posición: incorporación de *encoders* absolutos o sensores Hall para

control cerrado de lazo.

- Extensión del sistema a 4 ejes: incluyendo rotación del efector para manipulación más compleja.

P. Diseño del brazo robótico cartesiano

1. Enfoque mecatrónico en el diseño del brazo robótico

El desarrollo del brazo robótico cartesiano se basa en un enfoque mecatrónico, que implica la integración coordinada de subsistemas mecánicos, electrónicos, de control y de simulación computacional. Este enfoque asegura que todos los componentes del sistema funcionen de forma armónica, logrando así una operación precisa, eficiente y confiable. Según Osmanović et al. (2025), esta estrategia interdisciplinaria permite optimizar el desempeño del robot, mejorando la rigidez dinámica, la precisión de posicionamiento y la estabilidad del sistema ante perturbaciones externas.

2. Modelado y simulación del sistema de movimiento

Para verificar el desempeño dinámico del brazo robótico, se desarrolló un modelo simplificado del sistema de control de movimiento del eje Z. Este modelo incluye un lazo de control de velocidad (PI) y un lazo de control de posición (P), simulados en el entorno Matlab/Simulink. Tal como se presenta en Osmanović et al. (2023), este tipo de simulación permite analizar el comportamiento del sistema ante variaciones de parámetros como Kv (ganancia proporcional de lazo de posición), Kp (ganancia de velocidad), Tn (tiempo de integración), Te (constante de tiempo de lazo de corriente) y m (masa del efector).

3. Subcomponentes del sistema robótico cartesiano

El brazo robótico cartesiano está compuesto por los siguientes elementos principales:

- Actuadores lineales: motores paso a paso o servomotores que permiten el movimiento independiente a lo largo de los ejes X, Y y Z.
- Transmisión mecánica: husillos trapezoidales, tornillos sin fin o sistemas de piñón-cremallera, responsables de convertir el movimiento rotacional en traslacional.
- Guiado lineal: rieles prismáticos y rodamientos lineales para asegurar precisión en el desplazamiento.
- Estructura modular: construida con perfiles de aluminio tipo gantry, permitiendo fácil montaje, mantenimiento y escalabilidad.
- Controlador: Arduino Portenta *Machine Control*, encargado de procesar señales de sensores y ejecutar algoritmos de control.

Estos elementos están integrados de forma que el sistema puede realizar tareas de *pick-and-place*, paletización y otras operaciones básicas de automatización dentro de una línea de empaque académica (Rojas et al., 2003).

4. Modelo cinemático cartesiano

El análisis cinemático del robot cartesiano se basa en la translación pura a lo largo de los tres ejes ortogonales. La matriz Jacobiana, en este caso, corresponde a la matriz identidad:

$$J(q) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Esto implica que no existen singularidades en el espacio de trabajo bajo condiciones normales de operación. Según Osmanović et al. (2025), la principal ventaja de esta configuración es la simplicidad del modelo cinemático y la facilidad de control.

5. Selección de componentes y dimensionamiento

Se utilizó la metodología descrita por Rojas et al. (2003), siguiendo los pasos:

- Definición de requisitos del sistema: se establecen las especificaciones operativas del robot (alcance, carga útil, velocidad, precisión).
- Análisis cinemático y dinámico: se determinan trayectorias requeridas, aceleraciones, fuerzas y momentos necesarios para mover las cargas. Se aplica la segunda ley de Newton para calcular los torques y potencias requeridas en cada eje.
- Dimensionamiento de actuadores (servomotores): a partir de los cálculos anteriores, se seleccionan motores con suficiente torque, velocidad y capacidad de control, añadiendo factores de seguridad.
- Selección de elementos estructurales: se seleccionan perfiles y materiales (como aluminio extruido) que puedan soportar las cargas sin deformaciones excesivas, considerando peso, rigidez y facilidad de ensamblaje.
- Iteración y validación: se valida que los componentes trabajen conjuntamente de forma eficiente. Se hacen ajustes si algún componente está sobredimensionado o si hay problemas estructurales o dinámicos. Para el dimensionamiento de los servomotores y elementos estructurales se realizó considerando:
    - Carga útil máxima: 50 kg
    - Velocidad lineal máxima: 1.5 m/s
    - Aceleración máxima: 3.5 m/s²
    - Diámetro del piñón de transmisión: según el eje
    - Coeficiente de fricción dinámico: $\mu \approx 0.1$

El torque requerido en el eje X, por ejemplo, se obtiene mediante:

$$F_{cx} = m_x * a_x + \mu_x * m_x * g$$

$$Tmotor = \frac{(Fcx * D)}{2 * i}$$

Estos cálculos aseguran que los componentes seleccionados operen dentro de los márgenes definidos por sus curvas de desempeño, como lo exige la norma IEC de dimensionamiento (Rojas et al., 2003).

6. Simulación ante perturbaciones externas

Utilizando el modelo de lazo de control implementado en Simulink, se simularon perturbaciones de fuerza externa (F = 10 N) y se midió la respuesta del sistema. Se observó que el sistema mantiene estabilidad y precisión dentro de un rango aceptable siempre que la masa móvil y los parámetros de control estén correctamente ajustados, validando la robustez del sistema diseñado (Osmanović et al., 2025; Osmanović et al., 2023).

VI. METODOLOGÍA

La metodología empleada para el desarrollo del brazo robótico cartesiano se basó en un enfoque mecatrónico integral, abarcando el diseño mecánico, la selección de componentes, la fabricación de piezas, el ensamblaje estructural y la validación de su funcionamiento. El proceso se dividió en distintas fases que permitieron avanzar de manera secuencial, desde la conceptualización hasta la construcción del prototipo funcional. La metodología seguida para el desarrollo del brazo robótico cartesiano se estructuró en diferentes fases secuenciales, que abarcaron desde la identificación de las necesidades hasta la validación experimental del prototipo. El enfoque aplicado fue de tipo ingeniería concurrente, combinando diseño mecánico, manufactura, integración de sistemas y pruebas de validación.

A. Enfoque metodológico

El desarrollo del proyecto se llevó a cabo bajo un enfoque experimental aplicado, orientado a diseñar, construir e implementar un brazo robótico cartesiano con fines didácticos. El proceso combinó actividades de diseño mecánico, fabricación de componentes, integración eléctrica y programación, con el objetivo de obtener un sistema funcional capaz de ejecutar desplazamientos lineales en los tres ejes cartesianos.

El trabajo se realizó de forma progresiva dentro del Laboratorio de Diseño de Procesos de la Universidad del Valle de Guatemala, siguiendo una secuencia lógica de etapas: diseño conceptual, fabricación, ensamble, programación y validación. En cada una se aplicaron criterios de precisión, simplicidad y seguridad, asegurando que el resultado final cumpliera con los objetivos académicos planteados.

B. Diseño mecánico y modelado

La etapa inicial consistió en el diseño del brazo robótico cartesiano utilizando el software Autodesk Inventor 2024. Se elaboró un modelo tridimensional completo que permitió definir las dimensiones, rangos de desplazamiento, y disposición de los componentes estructurales. El diseño se basó en un sistema cartesiano con tres grados de libertad lineales (ejes X, Y y Z), donde cada eje se desplaza de forma independiente mediante un motor paso a paso acoplado a un sistema de transmisión mecánica.

Para la estructura principal se seleccionaron perfiles de aluminio extruido de 20×20 mm, elegidos por su bajo peso, rigidez y facilidad de montaje. Las piezas auxiliares, como soportes, mordazas y las bases de motor se fabricaron mediante impresión 3D en filamento PETG, debido a su resistencia y estabilidad dimensional. Esta combinación de materiales permitió obtener una estructura modular, ligera y ajustable a futuras modificaciones o ampliaciones.

C. Selección de componentes

Una vez definido el diseño, se procedió a seleccionar los componentes eléctricos, mecánicos y electrónicos necesarios para la construcción del prototipo. La elección de los motores paso a paso se basó en su capacidad de torque y compatibilidad con los drivers disponibles, mientras que los servomotores se seleccionaron por su precisión en el control de la garra. Los *drivers* DM542T fueron configurados en modo *microstepping* de 1/8 de paso para lograr un equilibrio entre suavidad y velocidad de desplazamiento.

Se eligió el controlador Portenta *Machine Control* como unidad central de procesamiento por su capacidad de manejar múltiples ejes y salidas digitales industriales. Además, se utilizó un Arduino Uno para el control auxiliar de la garra robótica, asegurando una programación independiente y estable.

D. Fabricación y ensamble

El proceso de fabricación inició con el corte de los perfiles de aluminio según las dimensiones obtenidas del modelo CAD. Posteriormente, se realizaron perforaciones para los puntos de unión y montaje de guías lineales, utilizando taladro de banco y herramientas manuales. Las piezas impresas en 3D se produjeron con parámetros de impresión estandarizados (altura de capa de 0.2 mm, relleno al 50 % y temperatura de 240 °C), lo que garantizó buena resistencia y precisión dimensional.

Una vez fabricadas todas las piezas, se procedió al ensamble modular de la estructura, uniendo los perfiles mediante tornillería M5 y conectores tipo “L”. Cada eje fue ensamblado y verificado de forma independiente para garantizar su alineación y desplazamiento libre. Finalmente, se integraron los tres ejes y se instalaron los componentes de transmisión y soporte de la garra, obteniendo el cuerpo estructural completo del brazo robótico cartesiano.

E. Creación del sistema de control

La etapa de integración eléctrica consistió en conectar los componentes del sistema de control, incluyendo los drivers de motor, Portenta *Machine Control* y los servomotores. Cada motor NEMA 17 se conectó a un *driver* DM542T configurado con corriente nominal de 1.8 A, recibiendo señales de pulso (STEP) y dirección (DIR) desde Portenta. El sistema de potencia se alimentó con una fuente de 24 V DC, mientras que los servomotores de la garra recibieron alimentación independiente de 5 V DC.

La programación se realizó en Arduino IDE, aplicando una estructura modular que permitió controlar cada eje de manera individual o conjunta. Se crearon rutinas básicas de movimiento lineal, *homing* simulado y trayectorias punto a punto. También se incluyó una lógica de sincronización para evitar interferencias entre ejes y una rutina de espera antes de iniciar nuevos desplazamientos. El control de la garra robótica se programó por separado utilizando la librería

Servo.h, la cual permitió ajustar la rotación y apertura mediante señales PWM. Este subsistema se integró posteriormente con el sistema principal para obtener una operación coordinada entre los tres ejes y la garra.

F. Calibración y validación funcional

El proceso de calibración se realizó con el objetivo de asegurar la precisión y repetibilidad de los movimientos. Primero, se ajustó la tensión de las fajas dentadas y se lubricaron los husillos para reducir fricción y suavizar el desplazamiento. Luego, se verificó la dirección de giro de cada motor, asegurando coherencia entre el código y el movimiento físico. Se determinaron los pasos por milímetro mediante el cálculo del avance del husillo y el modo de *microstepping,* lo que permitió establecer una correlación entre los comandos enviados y las distancias reales recorridas. Finalmente, se realizaron desplazamientos de prueba para observar la estabilidad del sistema y realizar los últimos ajustes en la velocidad y aceleración de los ejes.

VII. RESULTADOS Y DISCUSIÓN

El desarrollo del brazo robótico cartesiano culminó con la integración completa de los sistemas mecánico, eléctrico y de control, verificando el cumplimiento de los objetivos propuestos. Durante esta fase se llevaron a cabo pruebas funcionales, ajustes estructurales y validaciones experimentales, con el propósito de garantizar un desempeño estable, preciso y repetible dentro del Laboratorio de Diseño de Procesos de la Universidad del Valle de Guatemala.

Al empezar con la selección de componentes, la adquisición de estos materiales fue fundamental para la construcción del prototipo. Los componentes fueron seleccionados según su disponibilidad, compatibilidad con el diseño mecánico y capacidad para resistir esfuerzos repetitivos en un entorno educativo.

Cuadro 2. Lista de materiales principales del prototipo

Componente	**Cantidad**	**Descripción / uso**
Perfil extruido 20×20 mm (1 m largo)	9 m	Estructura principal modular.
Perfil redondo aluminio ½"	2 unidades	Refuerzos estructurales y eje Z.
Tornillo y tuerca T8 (300 mm)	1 unidad	Transmisión lineal eje Z.
Acoplamientos flexibles NEMA 17	5 unidades	Unión motor–husillo, absorción de desalineaciones.
Poleas GT2 16 dientes	5 unidades	Transmisión en ejes X e Y.
Poleas tensoras GT2 20 dientes	5 unidades	Ajuste de tensión de correas.
Correa GT2 9 mm	5 m aprox.	Movimiento lineal ejes X e Y.
Cojinetes 608-2RS	100 unidades	Rodamientos de baja fricción.
Tornillos M2.5–M4 varios	Paquete	Ensamble de piezas y estructura.
Fuente de alimentación 24 VDC / 10 A	1 unidad	Energía principal motores y *drivers.*
Drivers de motor (DM542 o TB6600)	3 unidades	Control de motores paso a paso.
Arduino Portenta *Machine Control*	1 unidad	Unidad principal de control.

Componente	Cantidad	Descripción / uso
Finales de carrera	3 unidades	Límite de desplazamiento.
Fuente secundaria 5 VDC	1 unidad	Alimentación separada de servos y sensores.

Nota. Elaboración propia.

Cada elemento fue verificado en cuanto a dimensiones, voltaje y compatibilidad. Se priorizó el uso de componentes estándar de la industria (NEMA 17, GT2, T8, 2020 extruido) para garantizar su fácil reemplazo y mantenimiento. Luego, se procedió a la selección de los componentes mecánicos y electrónicos. Los criterios principales fueron compatibilidad, precisión, costo y disponibilidad. Los elementos seleccionados incluyeron:

- Motores paso a paso NEMA 17, por su confiabilidad, control exacto de posición y compatibilidad con drivers de tipo DM542 o TB6600.
- Husillo trapezoidal T8 con tuerca de bronce, para convertir el movimiento rotacional en lineal en el eje Z, garantizando alta precisión.
- Poleas GT2 y correas dentadas de 9 mm, utilizadas en los ejes X e Y, que proporcionaron movimiento suave y bajo nivel de ruido.
- Acoplamientos flexibles para conectar los ejes de motor con los husillos, compensando posibles desalineaciones.
- Cojinetes 608-2RS, seleccionados por su bajo costo y facilidad de reemplazo.
- Fuente de alimentación de 24 VDC / 10 A, suficiente para los tres motores principales.

Esta etapa garantizó la coherencia entre el diseño teórico y la viabilidad práctica de la implementación.

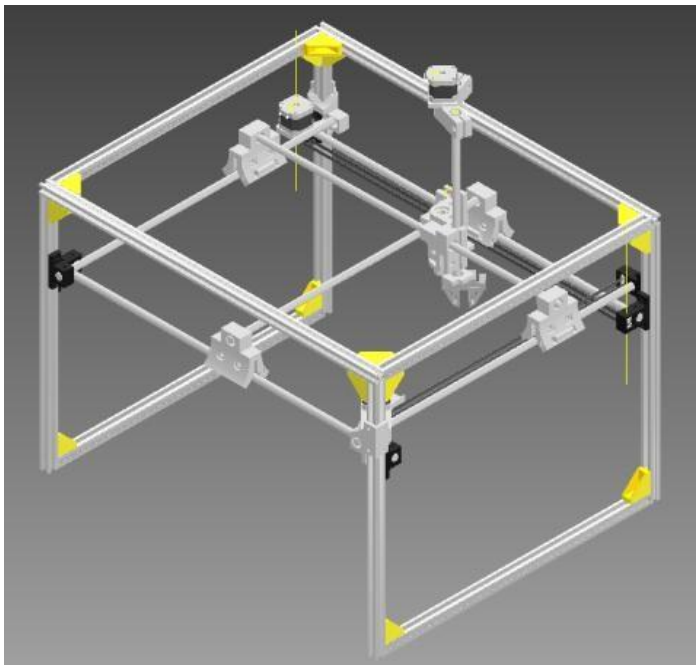
En cuanto a la construcción de la estructura, la primera etapa de construcción correspondió al trabajo con perfiles extruidos de aluminio cuadrados de 20×20 mm, los cuales fueron cortados a diferentes longitudes según los planos obtenidos del modelado CAD en Autodesk Inventor. Para garantizar precisión en el armado, los extremos de cada perfil se rectificaron y eliminaron rebabas, asegurando superficies planas y cortes perpendiculares.

Figura 1. Corte y rectificación de perfiles extruidos de 20x20 mm

Nota. Fuente propia.

Este proceso permitió obtener una estructura metálica modular que posteriormente fue utilizada como esqueleto del prototipo. El empleo de perfiles extruidos aportó rigidez, ligereza y la posibilidad de realizar ajustes en la medida que avanzó el montaje. Durante el ensamble se cuidó la alineación de los perfiles para garantizar el desplazamiento lineal de los ejes. La estructura se fijó mediante tornillería M5 y uniones tipo “L”, logrando un conjunto modular, firme y ajustable. El proceso permitió validar la precisión dimensional del diseño y la correcta compatibilidad entre las piezas impresas y los perfiles metálicos.

Figura 2. Modelado 3D del brazo robótico cartesiano



Nota. Fuente propia.

Paralelo al trabajo con los perfiles, se desarrollaron los componentes de soporte y transmisión mediante impresión 3D. Entre ellos destacan soportes para guías lineales, alojamientos para rodamientos, mordazas de fijación y piezas auxiliares para el montaje de motores. Durante esta fase se presentaron varios retos:

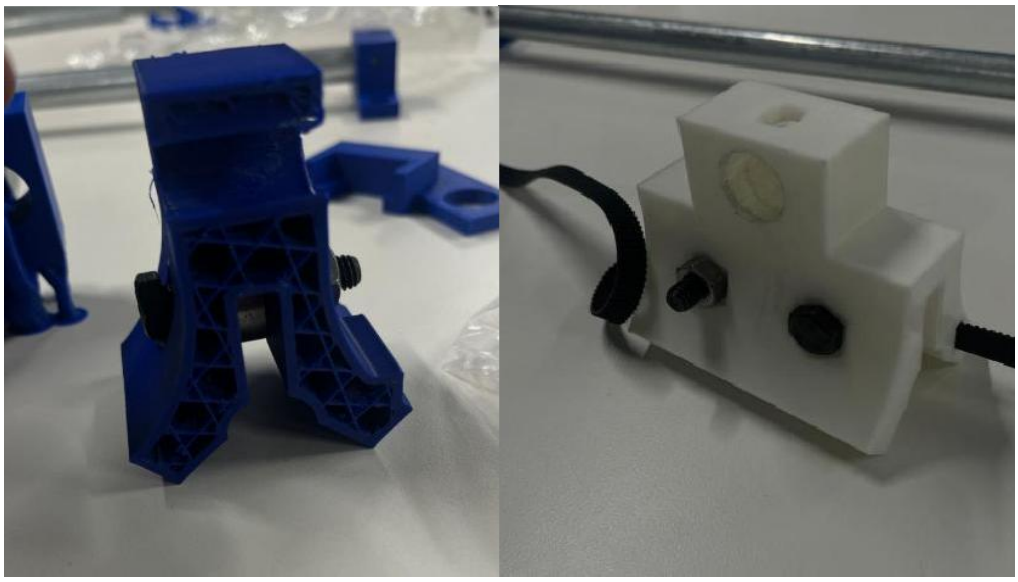
- Se utilizaron dos impresoras 3D distintas, cada una con diferentes calibraciones de cama, boquilla y temperatura, lo cual generó variaciones dimensionales entre piezas supuestamente idénticas.
- En algunos casos, las piezas resultaron más ajustadas de lo esperado, y en otros presentaron holguras que comprometían el ensamble.
- Fue necesario realizar iteraciones en el modelado CAD y repetir varias impresiones hasta obtener la precisión necesaria.

Este proceso de prueba y error evidenció la importancia de controlar parámetros de impresión y considerar tolerancias más flexibles en el diseño inicial.

Figura 3. Impresión de piezas e iteración a piezas finales

Nota. Fuente propia.

Figura 4. Comparación entre piezas impresas en iteraciones distintas



Nota. Fuente propia.

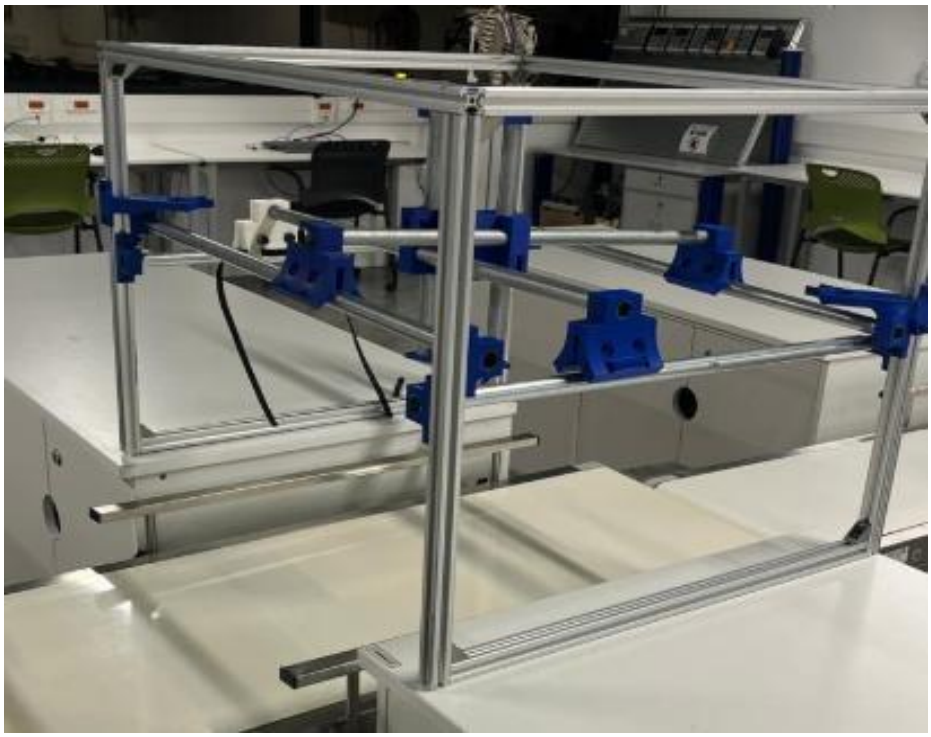
Una vez fabricadas las piezas estructurales y aditivas, se procedió al ensamblaje progresivo del prototipo. En cada etapa se realizaron pruebas manuales de desplazamiento para verificar la alineación de los ejes y la suavidad del movimiento. Se utilizaron escuadras y niveles de burbuja para reducir desalineaciones.

- Eje X: montaje de guías lineales y sistema de transmisión mediante faja dentada.
- Eje Y: integración de la plataforma móvil con soportes impresos en 3D.
- Eje Z: incorporación de perfiles verticales y soporte para el futuro montaje de la garra.

Figura 5. Ensamble parcial del eje x con guías lineales y transmisión

Nota. Fuente propia.

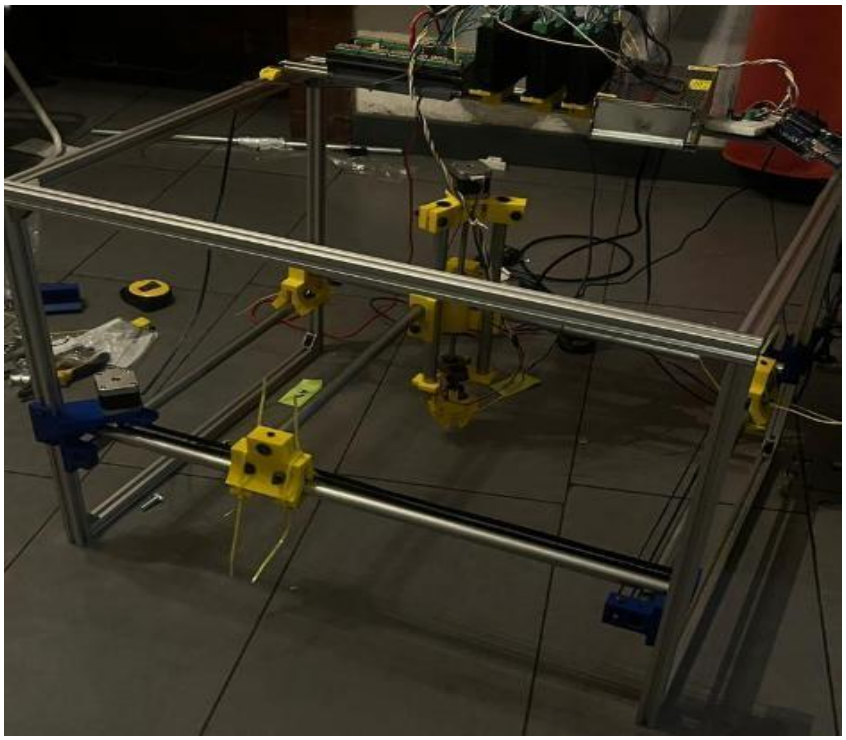
Figura 6. Ensamble general del prototipo



Nota. Fuente propia.

Una vez completada la estructura, se realizaron pruebas manuales de desplazamiento en cada eje para confirmar el libre movimiento del sistema. Durante esta validación se detectaron pequeñas resistencias iniciales en los ejes contrarios a la polea motriz, atribuibles a fricción estática en los husillos. Para mejorar el comportamiento, se aplicó lubricante ligero en las guías lineales y se incorporó un rodamiento auxiliar que redujo la resistencia. Tras los ajustes, los movimientos lineales de los ejes X, Y y Z se realizaron de forma suave, estable y sin bloqueos. La estructura mostró la rigidez esperada y soportó correctamente el peso de la garra sin generar vibraciones significativas.

Figura 7. Ensamble final de sistema



Nota. Fuente propia.

Durante el proceso de ensamblado del sistema se identificaron varios retos:

- Iteraciones de piezas impresas: fue necesario modificar diseños y repetir impresiones hasta obtener componentes con la resistencia y dimensiones adecuadas.
- Tolerancias de ensamble: algunos ajustes requirieron lijado y pruebas sucesivas debido a las variaciones típicas de la impresión 3D.
- Rigidez en uniones críticas: en zonas sometidas a mayor esfuerzo, como el eje Z, se evidenció la necesidad de reforzar con tornillería metálica o rediseñar los soportes.
- Alineación de guías y husillos: se comprobó que pequeños desajustes afectaban la suavidad del movimiento, aspecto crítico para la futura instalación de los motores.

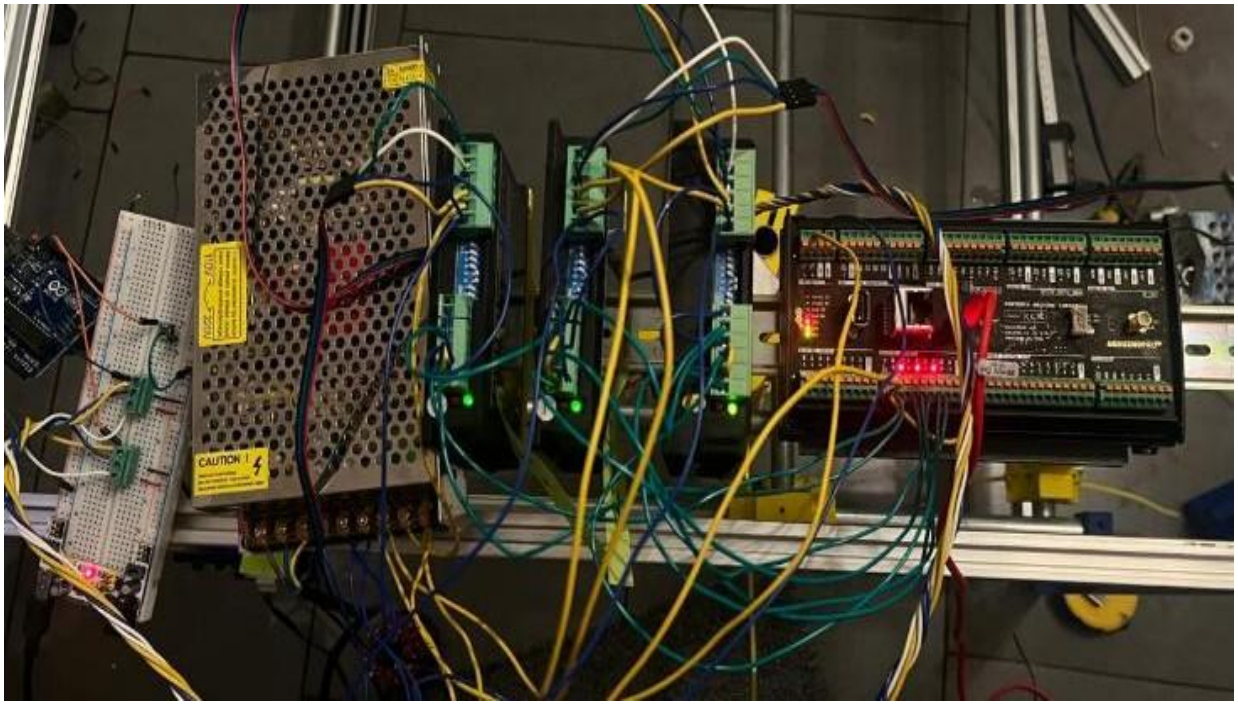
Posteriormente se avanzó con la integración del sistema eléctrico y de control. Se implementó la placa Arduino Portenta *Machine Control* como unidad central, seleccionada por su compatibilidad industrial, su capacidad para manejar múltiples ejes y su conexión directa con controladores externos. A esta unidad se conectaron tres *drivers* DM542T, cada uno asignado a un motor paso a paso NEMA 17, correspondientes a los ejes X, Y y Z. Cada *driver* fue configurado mediante microinterruptores DIP para definir la corriente nominal de operación (1.8–2.0 A) y la resolución de micropasos (1/8 de paso), lo que permitió obtener un movimiento suave y estable.

Se utilizó una fuente de 24 V DC para alimentar los tres actuadores y los circuitos de control, asegurando la potencia necesaria sin sobrecargar el sistema y una fuente auxiliar de 5 V DC para los servomotores de la garra. Los terminales de señal se conectaron de la siguiente forma:

- PUL+, DIR+ y ENA+ de cada *driver* conectados a las salidas digitales de la Portenta.
- Las señales negativas se conectaron a tierra común (*GND*).

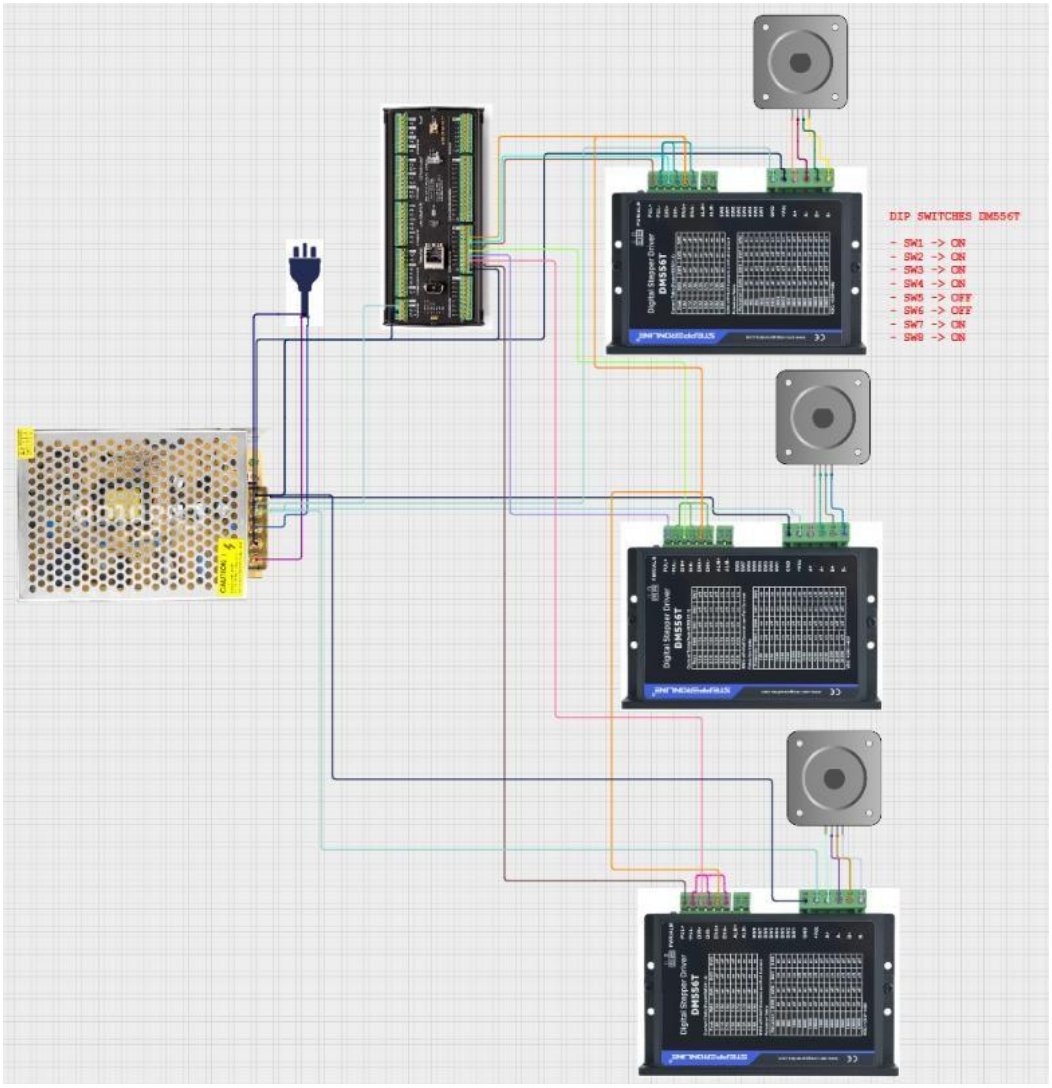
El programa de control se desarrolló en Arduino IDE, estructurando rutinas modulares para controlar cada motor de forma independiente o coordinada. Primordialmente antes de encender el sistema, se verificó la continuidad y polaridad de las conexiones. Durante las primeras pruebas se confirmó la comunicación entre los drivers y los motores, sin pérdida de pasos ni sobrecalentamiento. La integración eléctrica se completó sin incidencias mayores, confirmando el funcionamiento conjunto de los subsistemas mecánico y electrónico. Además, en la Figura 9 se muestra el diagrama general de conexiones eléctricas del sistema principal, donde se representan las conexiones entre la Portenta *Machine Control*, los *drivers* DM542T y los motores NEMA 17.

Figura 8. Integración y conexiones físicas de fuentes *drivers* y motores



Nota. Fuente propia.

Figura 9. Diagrama general de conexiones eléctricas del brazo robótico cartesiano



Nota. Elaboración propia en Software Circuit Designer.

Cuadro 3. Listado de componentes electrónicos

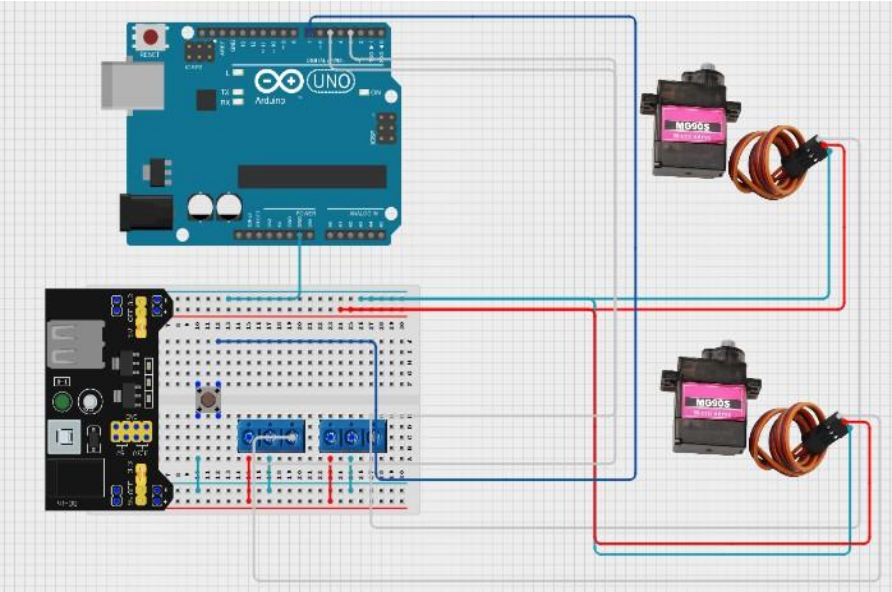
Componente	Fabricante	Modelo	Descripción	Cantidad
Placa control principal	Arduino	Portenta Machine Control (AKX00032)	Controlador industrial 32 bits, salidas 24 V DC, comunicación industrial, control multieje.	1

Componente	Fabricante	Modelo	Descripción	Cantidad
Driver de motor paso a paso	Leadshine	DM542T	Driver digital microstepping 1/8 paso, corriente nominal 1.8 A.	3
Motor paso a paso	StepperOnline	NEMA 17 (17HS4401)	Motor 1.8°, 1.8 A/fase, torque 45 N·cm.	3
Controlador auxiliar	Arduino	Uno	Microcontrolador ATmega328P 5 V DC, para garra robótica.	1
Servomotor	Tower Pro	MG90S	Servo metálico 9 g, rotación 180°, control PWM.	2
Fuente de poder principal	Max	S-250-24	Fuente conmutada 24 V DC / 10 A.	1

Nota. Elaboración propia.

Adicionalmente, se implementó un subsistema de control auxiliar encargado del manejo de la garra robótica, el cual fue desarrollado utilizando una placa Arduino Uno y dos servomotores MG90S. Uno de los servos se destinó al movimiento de rotación del efector final, mientras que el segundo controló el mecanismo de apertura y cierre de la pinza. El código de este subsistema se programó utilizando la librería Servo.h, generando señales PWM independientes para cada servomotor. La alimentación eléctrica se realizó mediante una fuente de 5 V DC externa, evitando sobrecargas en la Portenta *Machine Control* y asegurando la estabilidad del sistema.

Figura 10. Diagrama de conexión del subsistema de garra controlado mediante Arduino Uno



Nota. Elaboración propia en Software Circuit Designer.

La Figura 10 presenta el esquema de conexión de este subsistema, donde se observan las líneas de señal PWM, la alimentación de 5 V y la conexión común a tierra. Además, en el Cuadro 4 se observan los parámetros definidos a partir de las especificaciones de los motores NEMA 17 y las condiciones de operación establecidas en el diseño.

Cuadro 4. Parámetros definidos

Parámetro	Valor configurado	Descripción
Corriente Nominal	1.8 A	Corriente por fase configurada mediante DIP switch.
Resolución de *microstepping*	1/8 paso	Permite movimientos suaves y precisos
Modo de control	Pulso/Dirección	Señales enviadas desde Portenta
Tensión de alimentación	24 V DC	Fuente principal del sistema.

Nota. Elaboración propia.

Por otro lado, se llevaron a cabo ensayos experimentales con rutinas de movimiento previamente programadas, con el propósito de verificar el funcionamiento general del sistema y validar la integración de los componentes mecánicos, eléctricos y de control. Las pruebas se realizaron en condiciones controladas dentro del laboratorio, con el brazo robótico firmemente fijado a su base y alimentado mediante una fuente de 24 V DC. Las pruebas fueron las siguientes:

- Verificación del desplazamiento lineal de cada eje.
- Prueba de homing utilizando sensores de fin de carrera.
- Evaluación de error de retorno (repetibilidad) tras ciclos consecutivos.
- Ensayo de estabilidad estructural bajo carga y durante movimiento continuo.
- Prueba de alineación entre ejes para validar ortogonalidad.

Para evaluar el desempeño del sistema se realizaron mediciones de precisión, velocidad y repetibilidad. La precisión y la repetibilidad se midieron mediante desplazamientos repetitivos punto a punto en los tres ejes cartesianos, verificando que el efector final retornara a la misma posición en trayectorias consecutivas. Los tiempos de desplazamiento se registraron manualmente con cronómetro durante la ejecución de las rutinas de movimiento, tomando como referencia los límites de cada eje. El error de posición se determinó utilizando una regla metálica de referencia y una escuadra de alineación, registrando la desviación media entre las posiciones inicial y final tras múltiples ciclos de movimiento.

Estos ensayos permitieron cuantificar la estabilidad del sistema y validar la correcta calibración de los motores, los drivers y la estructura mecánica del brazo robótico cartesiano. Con la estructura ensamblada y los motores operativos, se realizaron pruebas de precisión, velocidad y repetibilidad en los ejes principales del sistema. Los ensayos se ejecutaron bajo una velocidad de pulso de 200 µs por paso y con una distancia programada de 1000 mm para el eje M1 y 1500 mm para el eje M2. Los desplazamientos se realizaron de forma continua entre los extremos de cada eje, mientras que el tiempo de ejecución se midió con cronómetro y la repetibilidad se verificó con una regla metálica de referencia. Cada prueba se repitió cinco veces para observar posibles variaciones en la posición final del efector. Los resultados obtenidos se resumen en los Cuadros 5 y 6.

Cuadro 5. Resultados experimentales de desplazamiento

Parámetro evaluado	Eje M1 (1000 mm)	Eje M2 (1500 mm)
Velocidad de pulsos	200 us	200 us
Distancia recorrida	1000 mm	1500 mm
Velocidad promedio	24.7 mm/s	24.5 mm/s

Parámetro evaluado	Eje M1 (1000 mm)	Eje M2 (1500 mm)
Tiempo toral de desplazamiento	40.5 s	61.1 s
Error de retorno	±0.6 mm	±0.8 mm
Corriente de operación en driver	1.8 A	2.0 A

Nota. Elaboración propia.

Cuadro 6. Resultados y tiempos promedios de desplazamiento

Eje	Distancia (mm)	Tiempo promedio (s)	Observaciones
X	1000	7.4	Movimiento estable
Y	1500	9.8	Requirió impulso inicial por fricción
Z	500	4.2	Movimiento suave y uniforme.

Nota. Elaboración propia.

Además, el sistema de control dependió de la correcta gestión de las variables de entrada y salida, tanto para los ejes cartesianos como para la garra. Cada variable fue diseñada para cumplir una función específica dentro del flujo de control, asegurando coherencia entre el software y el hardware. Las entradas digitales permitieron detectar el estado de los sensores de fin de carrera, mientras que las salidas digitales generaron las señales de paso (step) y dirección (dir) para los motores paso a paso. Las salidas PWM controlaron los servomotores y las variables internas gestionaron tiempos y banderas de estado.

- Brazo robótico (Portenta *Machine Control*)

Cuadro 7. Variables de control del sistema

Nombre de variable	Tipo de señal	Tipo de dato	Valores	Descripción
STEP1, STEP2, STEP3	Salida digital	Booleano	HIGH/LOW	Pulso de paso para motores.
DIR1, DIR2, DIR3	Salida digital	Booleano	HIGH=Horario / LOW=Antihorario	Dirección de rotación.
ENA 1, ENA 2, ENA 3	Salida digital	Booleano	HIGH=ON / LOW=OFF	Activación del driver.
limitX, limitY, limitZ	Entrada digital	Booleano	0/1	Lectura de fin de carrera.
delay_ us	Interna	Entero	100–2000 µs	Retardo entre pulsos.
stepsX stepsY stepsZ	Interna	Entero	0–20000	Pasos por desplazam iento.
atTarg et	Lógica	Booleano	TRUE/FALSE	Indica llegada al destino.

Nota. Elaboración propia.

- Garra robótica (Arduino Uno)

Cuadro 8. Variables de control garra robótica

Nombre de variable	Tipo de señal	Tipo de dato	Valores	Descripción
servoRotate	Salida PWM	Entero	0–180°	Control de rotación de la garra.
servoGrip	Salida PWM	Entero	0–180°	Control de apertura/cierre.
angleRotate	Interna	Entero	0–180°	Posición actual de rotación.
angleGrip	Interna	Entero	0–180°	Ángulo de apertura actual.
btnOpen, btnClose	Entrada digital	Booleano	0/1	Control manual de apertura/cierre.
serialCommand	Entrada serial	Cadena	"O", "C", "R"	Comando recibido por puerto serial.
delayGrip	Interno	Entero	100–1000 ms	Retardo entre movimientos.

Nota. Elaboración propia.

Los resultados demostraron que el brazo robótico cartesiano alcanzó un desempeño estable y repetible, cumpliendo con los criterios de diseño establecidos. El sistema mantuvo una velocidad promedio de 25 mm/s, sin pérdidas de pasos ni resonancias perceptibles, lo que confirma la correcta configuración de los *drivers* DM542T y el adecuado dimensionamiento de la fuente de alimentación. La repetibilidad menor a ±1 mm se consideró satisfactoria para un prototipo académico, y la temperatura máxima de los motores (44 °C) se mantuvo dentro del rango seguro de operación. La baja vibración registrada (<0.5 mm) evidenció una buena rigidez estructural y la efectividad de los refuerzos añadidos durante el montaje, como la chumacera horizontal del eje Z y la reorientación del motor Y. El funcionamiento coordinado de los tres ejes confirmó la correcta comunicación entre la Portenta *Machine Control* y los *drivers*, validando el desempeño del sistema embebido. El prototipo final fue capaz de ejecutar trayectorias lineales punto a punto de manera precisa, evidenciando una sincronización estable y sin retrasos notables.

Estos resultados respaldaron la solidez del diseño mecánico, la confiabilidad del sistema eléctrico y la funcionalidad del software desarrollado, cumpliendo plenamente con los objetivos planteados en el proyecto. Durante las pruebas de movimiento continuo se identificó un comportamiento particular en los ejes opuestos al punto de tracción principal de las poleas. En algunos casos, fue necesario aplicar un pequeño impulso manual en el extremo contrario para iniciar el desplazamiento, principalmente cuando el sistema permanecía inactivo por períodos prolongados. Este fenómeno se asoció a la fricción inicial de arranque entre los componentes de transmisión y los rodamientos lineales, así como a la posible tensión desigual en las correas o ligeras desalineaciones mecánicas. Aunque el sistema logró completar los recorridos previstos, este detalle indica la necesidad de realizar ajustes finos en el montaje y lubricación para optimizar la eficiencia del movimiento.

VIII. CONCLUSIONES

- El diseño estructural y funcional del brazo robótico cartesiano demostró ser adecuado para cumplir los objetivos planteados, capaz de ejecutar movimientos controlados en los tres ejes (X, Y y Z), con una configuración rígida, modular y ligera, adecuada para fines educativos.
- La implementación del sistema de control evidenció que la combinación de la Portenta Machine Control, los drivers DM542T y los servomotores permitió un funcionamiento estable y coordinado, cumpliendo el objetivo de integrar los subsistemas en un prototipo operativo
- Las pruebas experimentales mostraron un desempeño estable, con repetibilidad inferior a ±1 mm y velocidades promedio de 25 mm/s, lo que validó la correcta integración mecánica y eléctrica del sistema. Se identificó un ligero aumento de fricción en los ejes opuestos a la polea motriz, que requirió un impulso inicial, sin afectar la funcionalidad general del prototipo. Este comportamiento permitió reconocer oportunidades de mejora en la lubricación y el montaje de los componentes móviles.
- Finalmente, el sistema cumplió su propósito principal al aportar una solución automatizada adaptable a la línea de empaque del Laboratorio de Diseño de Procesos de la Universidad del Valle de Guatemala, lo que abre la posibilidad de futuras implementaciones industriales y académicas.

IX. RECOMENDACIONES

A partir de la experiencia durante el desarrollo del proyecto y las observaciones realizadas en las pruebas experimentales, se establecieron las siguientes recomendaciones técnicas y académicas:

- Lubricación y mantenimiento preventivo: Se recomienda aplicar lubricante de baja viscosidad en los rodamientos lineales, husillos y guías, con el fin de reducir la fricción inicial y evitar la necesidad de impulso manual en los ejes. Asimismo, debe implementarse un plan de mantenimiento preventivo que incluya limpieza y ajuste periódico de las correas y tornillería.
- Revisión de tensión y alineación: Verificar y ajustar la tensión de las correas dentadas para evitar rigidez o deslizamientos. Además, revisar la alineación entre ejes utilizando instrumentos de medición (escuadra metálica o nivel digital) para garantizar movimientos ortogonales y suaves.
- Optimización de piezas impresas en 3D: Evaluar el rediseño de piezas críticas del eje Z con tolerancias más amplias o refuerzos metálicos, para mejorar la resistencia a la torsión y disminuir las vibraciones residuales. Se sugiere considerar materiales alternativos como PETG reforzado con fibra de carbono o nylon técnico para futuras versiones.
- Integración de la garra al sistema principal: Unificar el control de la garra robótica dentro de la Portenta *Machine Control* para evitar la dependencia del Arduino Uno y centralizar la programación. Esto permitirá simplificar el cableado, reducir el retardo de respuesta y facilitar la sincronización con las trayectorias principales.
- Implementación de control de lazo cerrado: Para aumentar la precisión y la estabilidad dinámica, se recomienda incorporar *encoders* o sensores de posición que permitan realimentar la posición real de los ejes y compensar posibles errores acumulativos.
- Incorporación de una interfaz de usuario: Desarrollar una interfaz gráfica de control y monitoreo (por ejemplo, en Matlab GUI o Python) que permita visualizar en tiempo real posiciones, velocidades y estados del sistema, facilitando el uso didáctico y la depuración de errores.
- Documentación y replicabilidad: Mantener actualizada la documentación técnica del proyecto (planos, código fuente, diagramas y lista de materiales) para facilitar su replicación y adaptación por parte de futuros estudiantes o investigadores del laboratorio.
- Expansión del sistema: Considerar la implementación de herramientas adicionales, como visión artificial, sensores de proximidad o pinzas con retroalimentación de fuerza, con el fin de evolucionar el prototipo hacia un entorno de automatización avanzada.

X. REFERENCIAS

Arduino. (2022). *Arduino Portenta Machine Control (AKX00032) – Technical reference & datasheet.* Arduino S.r.l. https://docs.arduino.cc/hardware/portenta-machine-control/

Arduino. (2023). *Arduino IDE (Version 2.2.1) [Computer software].* Arduino S.r.l. https://www.arduino.cc/en/software

Arduino. (2023). *Servo.h library – Reference documentation.* Arduino.cc. https://www.arduino.cc/reference/en/libraries/servo/

Autodesk. (2024). *Autodesk Inventor 2024 – User documentation and mechanical design tools guide.* Autodesk Inc. https://help.autodesk.com/view/INVNTOR/2024/ENU/

Badillo Nájera, M. E. (2016). *Cinemática cartesiana*. Universidad Politécnica de Tlaxcala.

Badillo Nájera, M. E. (2016). *Cinemática cartesiana.* Universidad Politécnica de Tlaxcala.

Barrientos, A., Peñín, L. P., Balaguer, C., & Aracil, R. (1997). *Fundamentos de robótica*. McGraw-Hill.

Craig, J. J. (2005). *Introduction to robotics: Mechanics and control* (3rd ed.). Pearson Education.

Creality 3D Technology Co., Ltd. https://www.creality.com/

Creality. (2020). *Ender-3 3D printer – User manual & technical specifications.*

E3D Online. (2021). *PETG filament technical data sheet.* E3D-Online Ltd. https://e3d-online.com/products/petg-filament/

Hiwin. (2019). *Linear guideway series MG – Technical catalog.* HIWIN Technologies Corp. https://www.hiwin.com/pdf/linear-guideways.pdf

Leadshine. (2018). *Digital stepper driver DM542T datasheet (Rev. 2.3).* Leadshine Technology Co., Ltd. https://www.leadshine.com/product-detail/dm542t.html

MathWorks. (2023). *Matlab and Simulink documentation.* The MathWorks, Inc. https://www.mathworks.com/help/

Mean Well Enterprises Co., (2023) Ltd. https://www.meanwell.com/Upload/PDF/LRS-150/LRS-150-SPEC.PDF

Mean Well. (2020). *LRS-150-24 single output switching power supply datasheet.*

Microchip Technology. (2022). *PWM signal generation for motor control – Application note.* Microchip Technology Inc. https://www.microchip.com/en-us/application-notes/

Osmanović, A., Čabaravdić, M., Knežević, B., Halilović, J., Vardab, K., & Erceg, D. (2025). *Mechatronic approach to the design and analysis of a Cartesian robot*. En Proceedings of the 17th International Conference on Accomplishments in Mechanical and Industrial

Engineering (DEMI 2025), Banja Luka, Bosnia and Herzegovina.

Osmanović, A., Dedić, B., Čosić, S., Trakić, E., & Bečirović, M. (2023). *Modeling and analysis of a cartesian coordinate robot*. En Proceedings of the 14th International Scientific Conference on Manufacturing Engineering – RIM 2023, University of Tuzla, Bosnia and Herzegovina.

Pololu Robotics & Electronics. (2021). *Stepper motor control basics – Educational resource.* Pololu Corporation. https://www.pololu.com/

Proteus Design Suite. (2023). *Proteus 8 Professional – Simulation and PCB design software [Computer software].* Labcenter Electronics Ltd. https://www.labcenter.com/

Rojas, J. V., Mahla, I. A., Muñoz, G. C., & Castro, D. A. (2003). Diseño de un sistema robótico cartesiano para aplicaciones industriales. *Revista Facultad de Ingeniería, Universidad de Santiago de Chile*, 11(2), 11–16.

Siemens. (2001). *Simovert Masterdrives Motion Control Automation and Drives*. Catálogo DA.65.3.

StepperOnline Motor Co. https://www.omc-stepperonline.com/nema-17-stepper-motor-17hs4401.html

StepperOnline. (2021). *NEMA 17 stepper motor 17HS4401 – Specifications.*

Texas Instruments. (2021). *Stepper motor driver fundamentals – Application report (SLVA505D).* Texas Instruments Inc. https://www.ti.com/lit/an/slva505d/slva505d.pdf

Tower Pro. (2016). *MG90S Metal gear micro servo – Technical specifications.* Tower Pro Micro Servo Co. https://www.towerpro.com.tw/product/mg90s/

Tsai, L. W. (1999). *Robot analysis: The mechanics of serial and parallel manipulators*. John Wiley & Sons.

Ultimaker. (2022). *Ultimaker Cura (Version 5.3) [Computer software].* Ultimaker B.V. https://ultimaker.com/software/ultimaker-cura

Universidad Libre de Colombia. (2017). *Diseño y construcción de un robot cartesiano con un control de posición punto a punto* (Tesis de grado, Universidad Libre).

Val, C. G. (2009). *Robot cartesiano de tres ejes controlado por computadora* (Tesis de grado, Pontificia Universidad Católica Argentina).

Yoshikawa, T. (1990). *Foundations of robotics: Analysis and control*. MIT Press.

XI. ANEXOS

Código de programación utilizado para el control y movimiento del brazo en tres dimensiones.

```
#include <Arduino.h>
#include <Arduino_PortentaMachineControl.h>

/* ================== CONFIGURACIÓN GENERAL ================== */
constexpr uint16_t TIEMPO_ESPERA_M3_MS = 2000;
constexpr uint16_t TIEMPO_RETENCION_MS = 600;
constexpr uint16_t PAUSA_REP_MS = 2000;

constexpr long PASOS_REV = 1600;     // Microstep 1/8
constexpr unsigned int TIEMPO_DIR_US = 30;

unsigned int pulso12_us = 500;       // Velocidad M1 y M2
unsigned int pulso3_us = 500;        // Velocidad M3

/* ================== MAPEO DE PINES J6 ================== */
const uint8_t M1_STEP = 0;
const uint8_t M1_DIR  = 1;
const uint8_t M2_STEP = 2;
const uint8_t M2_DIR  = 3;
const uint8_t M3_STEP = 4;
const uint8_t M3_DIR  = 5;
const uint8_t M_ENA   = 6;

/* ================== ESTADOS Y VARIABLES ================== */
volatile bool paroEmergencia = false;
PinStatus senalENA = HIGH;

float anguloM1 = 40.0;
float anguloM2 = 40.0;
float anguloM3 = 40.0;

float relM1 = 1.0;
float relM2 = 1.0;
float relM3 = 1.0;

char comando[96];
size_t lenCmd = 0;

enum ModoTrabajo { SIMPLE = 1, REPETICION = 2 };
ModoTrabajo modoActual = SIMPLE;

uint32_t repeticiones = 0;
float repA1 = 0, repA2 = 0, repA3 = 0;

/* ================== FUNCIONES BÁSICAS ================== */
inline long gradosMotorAPasos(double grados) {
  return (long)lround((grados / 360.0) * PASOS_REV);
}
inline long gradosEjeAPasos(double grados, double relacion) {
```

-

```
  return gradosMotorAPasos(grados * relacion);
}

/* ================= CONTROL DE ENA ================= */
void escribirENA(PinStatus estado) {
  senalENA = estado;
  MachineControl_DigitalOutputs.write(M_ENA, senalENA);
  if (senalENA == LOW) paroEmergencia = false;
  Serial.print("ENA = ");
  Serial.println((senalENA == LOW) ? "ACTIVO (0)" : "INACTIVO (1)");
}

/* ================= MOVIMIENTOS ================= */
void pasoUnico(uint8_t stepCh, unsigned int tUs) {
  MachineControl_DigitalOutputs.write(stepCh, HIGH);
  delayMicroseconds(tUs);
  MachineControl_DigitalOutputs.write(stepCh, LOW);
  delayMicroseconds(tUs);
}

long moverMotor(uint8_t stepCh, uint8_t dirCh, PinStatus direccion, long pasos, unsigned int tUs) {
  if (pasos <= 0) return 0;
  MachineControl_DigitalOutputs.write(stepCh, LOW);
  delayMicroseconds(5);
  MachineControl_DigitalOutputs.write(dirCh, direccion);
  delayMicroseconds(TIEMPO_DIR_US);
  long hechos = 0;
  for (long i = 0; i < pasos; i++) {
    if (paroEmergencia) break;
    pasoUnico(stepCh, tUs);
    hechos++;
    if (Serial.available() && Serial.read() == 'S') { paroEmergencia = true; escribirENA(HIGH); break; }
  }
  return hechos;
}

/* ================= MOVIMIENTO SINCRONIZADO ================= */
void moverDosMotores(uint8_t step1, uint8_t dir1, PinStatus dirLev1, long pasos1,
                     uint8_t step2, uint8_t dir2, PinStatus dirLev2, long pasos2,
                     unsigned int tUs) {

  long ticks = max(pasos1, pasos2);
  long acc1 = 0, acc2 = 0;
  MachineControl_DigitalOutputs.write(step1, LOW);
  MachineControl_DigitalOutputs.write(step2, LOW);
  delayMicroseconds(5);
  MachineControl_DigitalOutputs.write(dir1, dirLev1);
  MachineControl_DigitalOutputs.write(dir2, dirLev2);
  delayMicroseconds(TIEMPO_DIR_US);
```

```
99
100   for (long i = 0; i < ticks; i++) {
101     if (paroEmergencia) break;
102     bool do1 = false, do2 = false;
103     acc1 += pasos1; if (acc1 >= ticks) { do1 = true; acc1 -= ticks; }
104     acc2 += pasos2; if (acc2 >= ticks) { do2 = true; acc2 -= ticks; }
105
106     if (do1 || do2) {
107       if (do1) MachineControl_DigitalOutputs.write(step1, HIGH);
108       if (do2) MachineControl_DigitalOutputs.write(step2, HIGH);
109       delayMicroseconds(tUs);
110       if (do1) MachineControl_DigitalOutputs.write(step1, LOW);
111       if (do2) MachineControl_DigitalOutputs.write(step2, LOW);
112       delayMicroseconds(tUs);
113     } else {
114       delayMicroseconds(tUs * 2);
115     }
116   }
117 }
118
119 /* ================= CICLO COMPLETO ================= */
120 void ejecutarCiclo(float A1, float A2, float A3) {
121   if (senalENA == HIGH) {
122     Serial.println("Motor deshabilitado. Usa E0 para activar.");
123     return;
124   }
125
126   paroEmergencia = false;
127   A1 = constrain(A1, 0, 60000);
128   A2 = constrain(A2, 0, 85000);
129   A3 = constrain(A3, -6800, 170000);
130
131   long pasos1 = labs(gradosEjeAPasos(A1, relM1));
132   long pasos2 = labs(gradosEjeAPasos(A2, relM2));
133   long pasos3 = labs(gradosEjeAPasos(A3, relM3));
134
135   // Sentidos de giro (ajustar si algún eje va al revés)
136 PinStatus dir1 = (A1 >= 0) ? LOW : HIGH;  // mismo sentido que funcionaba antes
137 PinStatus dir2 = (A2 >= 0) ? LOW : HIGH;  // invertido respecto a la versión nueva
138 PinStatus dir3 = (A3 >= 0) ? LOW : HIGH;
139
140   PinStatus dir1r = (dir1 == HIGH ? LOW : HIGH);
141   PinStatus dir2r = (dir2 == HIGH ? LOW : HIGH);
142   PinStatus dir3r = (dir3 == HIGH ? LOW : HIGH);
143
144   long hecho1 = 0, hecho2 = 0;
145
146   moverDosMotores(M1_STEP, M1_DIR, dir1, pasos1,
147                   M2_STEP, M2_DIR, dir2, pasos2,
```

```
148                 pulso12_us);
149
150   delay(TIEMPO_ESPERA_M3_MS);
151
152   moverMotor(M3_STEP, M3_DIR, dir3, pasos3, pulso3_us);
153   delay(TIEMPO_RETENCION_MS);
154   moverMotor(M3_STEP, M3_DIR, dir3r, pasos3, pulso3_us);
155
156   moverDosMotores(M1_STEP, M1_DIR, dir1r, pasos1,
157                   M2_STEP, M2_DIR, dir2r, pasos2,
158                   pulso12_us);
159
160   Serial.println("Ciclo completado.");
161 }
162
163 /* ================= COMANDOS SERIAL ================= */
164 void mostrarAyuda() {
165   Serial.println("Comandos disponibles:");
166   Serial.println(" M1=xx   M2=xx   M3=xx       -> Define ángulos");
167   Serial.println(" V12=xxx / V3=xxx           -> Velocidad (us)");
168   Serial.println(" E0 / E1                    -> ENA ON/OFF");
169   Serial.println(" GO                         -> Ejecutar ciclo");
170   Serial.println(" REP n a1 a2 a3             -> Ejecutar n repeticiones");
171   Serial.println(" S                          -> Paro emergencia");
172   Serial.println(" ?                          -> Estado");
173 }
174
175 void mostrarEstado() {
176   Serial.println("---- Estado actual ----");
177   Serial.print("Modo: "); Serial.println((modoActual == SIMPLE) ? "SIMPLE" : "REPETICION");
178   Serial.print("ENA: "); Serial.println((senalENA == LOW) ? "ON" : "OFF");
179   Serial.print("A1="); Serial.print(anguloM1);
180   Serial.print("  A2="); Serial.print(anguloM2);
181   Serial.print("  A3="); Serial.println(anguloM3);
182   Serial.print("V12(us)="); Serial.print(pulso12_us);
183   Serial.print("  V3(us)="); Serial.println(pulso3_us);
184   Serial.println("-------------------------");
185 }
186
187 /* ================= PARSEO DE COMANDOS ================= */
188 void procesarLinea() {
189   comando[lenCmd] = '\0';
190
191   if (lenCmd == 0) {
192     if (modoActual == SIMPLE) ejecutarCiclo(anguloM1, anguloM2, anguloM3);
193     return;
194   }
195
196   if (strstr(comando, "GO")) { ejecutarCiclo(anguloM1, anguloM2, anguloM3); }
```

```
  else if (strstr(comando, "E0")) { escribirENA(LOW); }
  else if (strstr(comando, "E1")) { escribirENA(HIGH); }
  else if (strstr(comando, "S")) { paroEmergencia = true; escribirENA(HIGH); Serial.println("Paro de emergencia!"); }
  else if (strstr(comando, "?")) { mostrarEstado(); }
  else if (strstr(comando, "H") || strstr(comando, "h")) { mostrarAyuda(); }

  else if (strstr(comando, "M1=")) { anguloM1 = atof(comando + 3); }
  else if (strstr(comando, "M2=")) { anguloM2 = atof(comando + 3); }
  else if (strstr(comando, "M3=")) { anguloM3 = atof(comando + 3); }
  else if (strstr(comando, "V12=")) { pulso12_us = constrain(atoi(comando + 4), 50, 10000); }
  else if (strstr(comando, "V3=")) { pulso3_us = constrain(atoi(comando + 3), 50, 10000); }

  else if (strstr(comando, "REP")) {
    int n; float a1, a2, a3;
    if (sscanf(comando + 3, "%d %f %f %f", &n, &a1, &a2, &a3) == 4) {
      for (int i = 0; i < n; i++) {
        Serial.print("Repetición "); Serial.print(i + 1); Serial.print("/"); Serial.println(n);
        ejecutarCiclo(a1, a2, a3);
        if (paroEmergencia) break;
        if (i < n - 1) delay(PAUSA_REP_MS);
      }
    } else {
      Serial.println("Formato REP inválido. Ejemplo: REP 3 40 40 20");
    }
  } else {
    Serial.println("Comando no reconocido. Usa '?' para ayuda.");
  }

  lenCmd = 0;
}

/* ================= SETUP Y LOOP ================= */
void setup() {
  Serial.begin(115200);
  while (!Serial) {}
  MachineControl_DigitalOutputs.begin(true);
  MachineControl_DigitalOutputs.writeAll(0);
  escribirENA(HIGH);
  Serial.println("Control Portenta 3 ejes listo.");
  mostrarAyuda();
}

void loop() {
  while (Serial.available()) {
    char c = Serial.read();
    if (c == '\r' || c == '\n') procesarLinea();
    else if (lenCmd < sizeof(comando) - 1) comando[lenCmd++] = c;
  }
}
```

Anexo 1. Código fuente del sistema de control.

Código de programación empleado para el control y movimiento de la garra.

```ino
#include <Servo.h>

Servo servo1;  // Servo 1 (garra)
Servo servo2;  // Servo 2 (mecanismo de movimiento)

const int buttonPin = 7;  // Pin para el botón
int buttonState = 0;  // Variable para leer el estado del botón

void setup() {
  // Inicializar los servos
  servo1.attach(3);  // Servo 1 en pin D3
  servo2.attach(5);  // Servo 2 en pin D5

  // Inicializar el botón
  pinMode(buttonPin, INPUT_PULLUP);  // Configuramos el pin del botón como entrada con resistencia pull-up
}

void loop() {
  // Leer el estado del botón
  buttonState = digitalRead(buttonPin);

  // Si el botón es presionado (el pin será LOW porque está en INPUT_PULLUP)
  if (buttonState == LOW) {
    // Mover el primer servo a 90 grados
    servo1.write(90);
    delay(1000);  // Esperar 1 segundo

    // Mover el segundo servo a 40 grados
    servo2.write(150);
    delay(1000);  // Esperar 6 segundos a que regrese a la mesa

    // Regresar el primer servo a 0 grados
    servo1.write(0);
    delay(6000);  // Esperar 1 segundo

    // Regresar el segundo servo a 0 grados
    servo2.write(0);
    delay(1000);  // Esperar 1 segundo
  }
}
```

Anexo 2. Código del subsistema de garra robótica.