ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE MECÁNICA

CARRERA INGENIERÍA INDUSTRIAL

IMPLEMENTACIÓN DE UN PROTOTIPO DE LÍNEA DE ENSAMBLE DE PARTES PLÁSTICAS BASADO EN UN SISTEMA ROBÓTICO CARTESIANO

Trabajo de Integración Curricular

Tipo: Propuesta Tecnológica

Presentado para optar al grado académico de:

INGENIERO INDUSTRIAL

AUTOR:

HUGO JHONNATAN BONILLA MENA

Riobamba-Ecuador

2023

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE MECÁNICA

CARRERA INGENIERÍA INDUSTRIAL

IMPLEMENTACIÓN DE UN PROTOTIPO DE LÍNEA DE ENSAMBLE DE PARTES PLÁSTICAS BASADO EN UN SISTEMA ROBÓTICO CARTESIANO

Trabajo de Integración Curricular

Tipo: Propuesta Tecnológica

Presentado para optar al grado académico de:

INGENIERO INDUSTRIAL

AUTOR: HUGO JHONNATAN BONILLA MENA

DIRECTOR: ING. EDUARDO FRANCISCO GARCÍA CABEZAS

Riobamba – Ecuador

2023

Yo, Hugo Jhonnatan Bonilla Mena, declaro que el presente Trabajo de Integración Curricular es de mi autoría y los resultados de este son auténticos. Los textos en el documento que provienen de otras fuentes están debidamente citados y referenciados.

Como autor asumo la responsabilidad legal y académica de los contenidos de este Trabajo de Integración Curricular; el patrimonio intelectual pertenece a la Escuela Superior Politécnica de Chimborazo.

Riobamba, 25 de septiembre de 2023

Hugo Jhonnatan Bonilla Mena
060475341-8

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE MECÁNICA

CARRERA DE INGENIERÍA INDUSTRIAL

El Tribunal del Trabajo de Integración Curricular certifica que: El Trabajo de Integración Curricular; Tipo: Propuesta Tecnológica, **IMPLEMENTACIÓN DE UN PROTOTIPO DE LÍNEA DE ENSAMLE DE PARTES PLASTICAS BASADO EN UN SISTEMA ROBÓTICO CARTESIANO**, realizado por el señor: **HUGO JHONNATAN BONILLA MENA**, ha sido minuciosamente revisado por los Miembros del Tribunal del Trabajo de Integración Curricular, el mismo que cumple con los requisitos científicos, técnicos, legales, en tal virtud el Tribunal Autoriza su presentación.

	FIRMA	FECHA
Dr. Miguel Ángel Pérez Bayas **PRESIDENTE DEL TRIBUNAL**		2023-09-25
Ing. Eduardo Francisco García Cabezas **DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR**		2023-09-25
Ing. Julio César Moyano Alulema **ASESOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR**		2023-09-25

DEDICATORIA

Dedico este trabajo con profundo agradecimiento y devoción, en especial a San Miguel Arcángel, cuya presencia en mi vida ha sido una fuente inagotable de protección y guía en los momentos más desafiantes. Tu luz ha iluminado los caminos oscuros de la incertidumbre y tu fortaleza ha sostenido mi espíritu en medio de la adversidad. En cada paso que doy, sé que tu energía y amor me acompañan, guiándome hacia la sabiduría y la determinación para superar obstáculos y alcanzar mis metas. Que esta dedicación sea un humilde reconocimiento de tu constante presencia en mi camino, y una expresión de gratitud por tu constante amparo y apoyo.

Hugo

AGRADECIMIENTO

Agradezco a mis padres por todo su amor, sacrificio y orientación, han sido mi roca, siempre impulsándome a dar lo mejor de mí, a mi hermano, tu compañía y apoyo son invaluables, eres mi aliado en la vida y fuente de inspiración. A ti, mi querida amada, gracias por tu amor incondicional, paciencia y alegría que traes a mi vida. Eres mi amor, confidente y compañera de vida.

Cada uno de ustedes ha dejado una huella imborrable en mi corazón y soy afortunado de tenerlos en mi vida. Les agradezco profundamente por su amor, apoyo y por hacer mi vida más significativa.

Hugo

TABLA DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE ANEXOS

RESUMEN

El propósito fundamental de este estudio es la creación e implementación de un prototipo de línea de ensamble de partes plásticas, utilizando un sistema robótico cartesiano. La investigación inicio desde un estudio bibliográfico para la determinación de parámetros en una línea de ensamble de partes plásticas. La herramienta SolidWorks, es esencial para modelar, diseñar y analizar la estructura del prototipo, lo que permitió su validación de manera eficaz y precisa. En el hardware se empleó una tarjeta de Arduino MEGA como microcontrolador del prototipo, así como motores paso a paso, drivers y sensores para generar el movimiento del robot cartesiano. Posteriormente los softwares de Arduino IDE y Visual Studio 2022 fueron esenciales para codificar algorítmicamente y crear la interfaz gráfica del prototipo. El resultado tangible de esta investigación es un prototipo exitoso de línea de ensamble de partes plásticas basado en un sistema robótico cartesiano, logrando una eficacia de ensamblaje del 92% a través de rigurosas pruebas de funcionalidad. El tiempo de ensamble varía entre 4,19 y 4,34 minutos, esta diferencia de tiempos existe debido al apilamiento de placas de Arduino en la línea de ensamble. Además, se demostró un consumo energético eficiente de 144 W, consolidando así su precisión y efectividad en el proceso de ensamblaje. Se concluye que el sistema implementado logra recrear las condiciones operacionales de una línea de ensamblaje de partes plásticas, proporcionando así un medio efectivo para llevar a cabo pruebas y validaciones. Se sugiere para futuros proyectos que se tome en consideración e integre de manera fundamental este proceso en la planificación y desarrollo de líneas de producción complejas.

Palabras clave: <ROBOT CARTESIANO>, <PARTES PLASTICAS (PLA)>, < LÍNEA DE ENSAMBLE>, <INTERFAZ GRÁFICA>, <ARDUINO>

17-10-2023

1834-DBRA-UPT-2023

SUMMARY

The main purpose of this research is the creation and implementation of a prototype assembly line for plastic parts, with the use of a Cartesian robotic system. The research started from a bibliographic study for the determination of parameters in a plastic parts assembly line. The SolidWorks tool is essential to model, design and analyze the structure of the prototype, which allowed its validation in an efficient and accurate way. An Arduino MEGA board was employed as the prototype's microcontroller, as well as stepper motors, drivers and sensors to generate the Cartesian robot's movement. Afterwards Arduino IDE and Visual Studio 2022 software were essential to algorithmically code and create the graphical interface of the prototype. The tangible result of this research is a successful prototype plastic parts assembly line based on a Cartesian robotic system, achieving 92% assembly efficiency through rigorous functionality testing. The assembly time ranges from 4.19 to 4.34 minutes, this time difference is due to the stacking of Arduino boards on the assembly line. In addition, an efficient power consumption of 144 W was demonstrated, thus consolidating its accuracy and effectiveness in the assembly process. It is concluded that the implemented system manages to recreate the operational conditions of a plastic parts assembly line, thus providing an effective means to carry out tests and validations. It is suggested for future research projects that this process be taken into a fundamental consideration and integrated into the planning and development of complex production lines.

Keywords: < CARTESIAN ROBOT>, <PLASTIC PARTS (PLA)>, <ASSEMBLY LINE>, <GRAPHIC INTERFACE>, <ARDUINO>.

Mgs. Mónica Paulina Castillo Niama
C.I. 060311780-5

INTRODUCCIÓN

En el escenario cambiante y desafiante de la revolución industrial, donde cada avance tecnológico redefine los estándares de eficiencia y productividad, surge la necesidad imperante de explorar e implementar soluciones innovadoras en el ámbito de la manufactura. Esta tesis se adentra en la vanguardia de la producción industrial, enfocándose en la creación y puesta en marcha de un prototipo de línea de ensamble de partes plásticas, cimentado sobre las capacidades de un sistema robótico cartesiano. La urgencia de optimizar procesos y permanecer competitivos en un mercado globalizado impulsa esta investigación hacia el corazón de la automatización.

En la actualidad, la producción a gran escala es el epicentro de la competitividad en el ámbito industrial. Empresas líderes a nivel mundial han trazado el camino hacia la eficiencia a través de la implementación de líneas de ensamble automatizadas, marcando un hito en la evolución de la manufactura. Sin embargo, en el tejido empresarial nacional, persiste un desafío considerable: el dilema entre la necesidad apremiante de adoptar tecnologías avanzadas para mantenerse a la par con los estándares internacionales y las barreras económicas que esto conlleva.

Este estudio se propone abordar una problemática palpable que afecta a numerosas empresas nacionales: el difícil equilibrio entre la demanda de automatización para mejorar la competitividad y los desafíos financieros asociados con la adquisición de tecnologías de última generación. A través de la presentación y análisis de un prototipo de línea de ensamble basada en un sistema robótico cartesiano, esta investigación se posiciona como una guía práctica y accesible para aquellas empresas que buscan dar el salto hacia la modernización de sus procesos sin comprometer su estabilidad económica

CAPITULO I

1. DIAGNÓSTICO DEL PROBLEMA

1.1 Antecedentes

La revolución industrial permite el desarrollo tecnológico, económico y social, abarca una infinidad de procesos, estos pasaron de ser manuales a industriales, hoy en día la producción dentro de las industrias es a gran escala por lo que se implementan una vasta variedad de líneas, facilitando la producción, mejorando tiempos y disminuyendo costos. Las más grandes empresas alrededor del mundo han estado a la vanguardia, mejorando sus procesos, líneas de ensamble y tiempos, estas empresas se mantienen en auge por su gran innovación tecnológica, la automatización de líneas a través del tiempo ha ido mejorando hasta el punto de solo necesitar un operario que supervise el proceso o hasta la línea de producción, muchas de las empresas nacionales han logrado progresar, colocándose por encima de la competencia, esto se lleva a cabo mediante la automatización de líneas de ensamble mejorando su proceso de producción pero existe un gran porcentaje de empresas nacionales que no se atreven a la implementación de las líneas de ensamble por el alto costo adquisitivo de la maquinaria y se mantienen realizando procesos con metodologías algo obsoletas.

1.2 Planteamiento del problema

En la industria de manufactura de partes plásticas, la eficiencia en la línea de ensamble representa un factor crítico para la productividad y la competitividad. En este contexto, la optimización de los procesos de ensamblaje adquiere una importancia central. Las líneas de ensamble convencionales enfrentan desafíos en términos de eficiencia operativa y consumo energético. La necesidad de mejorar estos aspectos ha llevado a explorar alternativas tecnológicas avanzadas.

Según (Machado, 2009, pp. 22-23) la única vía para garantizar éxito de las empresas especializadas en la industria metalmecánica y plástica es la creación de productos altamente tecnológicos que demanden procesos complejos de elaboración. Es crucial optimizar los métodos de producción a través de la automatización, lo que resultará en una reducción de gastos y una mejora en la calidad mediante la disminución de errores en el proceso y una mano de obra más eficiente.

Entre las opciones tecnológicas que han demostrado su potencial se encuentran los sistemas robóticos cartesianos, que ofrecen movimientos precisos y controlados. Estos sistemas, si se

aplican de manera efectiva, pueden mejorar significativamente la eficiencia en las operaciones de ensamblaje.

No obstante, la implementación de prototipos de líneas de ensamble basadas en sistemas robóticos cartesianos, especialmente diseñadas para el ensamblaje de partes plásticas, requiere una metodología estructurada y un análisis detenido de los requerimientos. Uno de los aspectos críticos es la selección adecuada de componentes y su configuración óptima para garantizar tanto la eficiencia en el ensamblaje como un consumo energético eficiente. Es necesario abordar estos aspectos para llevar a cabo una implementación exitosa y rentable de un sistema robótico cartesiano en una línea de ensamble de partes plásticas.

1.3 Justificación

La implementación de un prototipo de línea de ensamblaje de partes plásticas representa una valiosa contribución al ámbito de la automatización industrial y la ingeniería de sistemas. Al avanzar en esta propuesta tecnológica, estamos directamente contribuyendo al logro del objetivo 3 del eje económico del Plan de Creación de Oportunidades de Ecuador. Este objetivo se centra en "Fomentar la productividad y competitividad en los sectores agrícola, industrial, acuícola y pesquero".

La automatización se establece como un elemento esencial para potenciar la eficacia en la producción industrial. La adopción de tecnologías vanguardistas, como los sistemas robóticos cartesianos, coloca a las empresas en una posición altamente competitiva en el mercado. La innovación tecnológica, mejora la calidad de sus productos y ampliando su capacidad productiva. Este proyecto tiene como objetivo principal contribuir al progreso de la investigación en automatización industrial. Proporcionará un proceso detallado y fundamentado sobre la implementación de un sistema robótico cartesiano en la línea de ensamblaje de partes plásticas. Con esta investigación, buscamos avanzar hacia un futuro industrial más eficaz, innovador y competitivo.

1.4 Objetivos

1.4.1. Objetivo general

- Implementar un prototipo de línea de ensamble de partes plásticas basado en un sistema robótico cartesiano.

1.4.2. Objetivos específicos

- Realizar un estudio del arte con la finalidad de caracterizar el prototipo de la línea de ensamble.
- Seleccionar una herramienta CAD para el modelado y validación del diseño del sistema robótico para la línea de ensamble.
- Seleccionar hardware y software para la automatización de la línea de ensamble.
- Determinar la eficacia de la línea de ensamble mediante pruebas de funcionalidad.

CAPITULO II

2. MARCO TEÓRICO

2.1 Antecedentes de investigación

Las aplicaciones del sistema cartesiano de robots en la industria son diversas de acuerdo con las necesidades del usuario, por ejemplo, Jara y Pacheco (2013, pp. 17-22) diseñaron y desarrollaron un robot de tipo cartesiano, que sirve para colocar tapas y pasadores en las cajas, este fue implementado en el proceso de empaquetado, en el laboratorio de automatización industrial, Facultad De Informática y Electrónica de la ESPOCH. El Robot fue construido utilizando aluminio estructurado, además, cuenta con sensores magnéticos, inductivos y ópticos que funcionan juntamente con cilindros neumáticos sin vástago, lo que permite que el robot se mueva en los ejes (x, y, z), para sostener las piezas, se ha empleado una pinza neumática que permite una sujeción efectiva. Los motores del sistema fueron controlados por electroválvulas que se activan a través de un programa. Se empleó la herramienta TIA Portal V12 de Siemens junto con un PLC y Módulos de Entradas/Salidas Programables, para la programación del sistema se utilizó una pantalla táctil como interfaz de usuario de la máquina (HMI) que se utilizó para controlar el sistema a distancia. Los investigadores concluyeron que el uso de robots en los sistemas de envasado, además de mejorarlos, permite a los estudiantes poner en práctica sus conocimientos de automatización.

Arcos et al (2015; citado en Leal y Pérez, 2022, pp. 28-30) diseñaron y construyeron un robot con tres grados de libertad con propósitos educativos, que realiza varias funciones como transportar objetos (una caja de fósforos) y replicar figuras, además, cuenta con un sistema inteligente basado en redes neuronales que le permite al robot reconocer ciertas palabras en un archivo de audio.,

Parra (2017, pp. 8-86) desarrolló un prototipo de robot cartesiano para la manipulación de diversos elementos. El robot fue diseñado y construido para realizar las funciones de una estación de trabajo, levantando y colocando objetos mediante pinzas, consta de diferentes componentes, entre ellos electrónica, llantas, válvulas, controladores, etc., y se realizó la simulación de los diferentes componentes. en el software SolidWorks.

Aldaz (2018, pp. 1-35) diseñó y construyó un prototipo de robot que se asemeja a un robot cartesiano que ayuda a pintar, en el estudio se analizaron y seleccionaron los componentes del robot y el modelado en 3D se hizo en CAD. En el diseño de circuitos electrónicos, se emplearon sensores de proximidad y ultrasónicos para ubicar, pintar y verificar el procedimiento; los controladores se encargaron de los motores de paso y una válvula electroválvula activó la pistola de pintura. Con la ayuda de un procesador, se creó un sistema de control que se encarga de la máquina; a

través de la interfaz de usuario, el ser humano puede guiar al robot. Al construirse, se obtuvo el modelo a escala 1:2, que es capaz de pintar áreas de 500mm x 500mm, a una velocidad constante de 22.50mm/s, el cual puede realizar tres procedimientos de pintado: barrido, contorno y anguloso.

Rodríguez et al (2016; citado en Cruz et al., 2020, p. 1.8) afirman que el enfoque QFD (Quality Function Deployment) ha sido reconocido durante muchos años como una importante herramienta de apoyo para la toma de decisiones tempranas en el diseño de robots. A partir de ello, desarrollaron un robot cartesiano a modo de impresora 3D, enfocado en las necesidades y deseos del cliente, en este caso, estudiantes de la carrera de Ingeniería de la Universidad de Pamplona, se creó un diseño asistido por computadora (CAD) junto con él, además de construirlo y ponerlo en marcha.

En cuanto a investigaciones de línea de ensamble, los investigadores Rincón y Vesga (2019, pp. 14-16) diseñaron un prototipo robotizado cartesiano con el propósito de simular los procedimientos de armado y almacenamiento de un vehículo de plástico. Este prototipo fue hecho con piezas previamente ensambladas. El análisis arrojó una estructura con ranuras de 1200 x 800 x 400 mm en los ejes XYZ, cada una de ellas controladas por un PCL Siemens S7-300 a través de un sistema de Profibus, además, cada ranura contaba con un sistema de neumática para manipular piezas. Por último, el sistema cuenta con equipos y una estructura que le permiten adaptarse a procesos industriales más complejos, gracias a sus mecanismos de comunicación.

2.2 Referencias Teóricas

2.2.1. La robótica

La palabra "robótica" fue acuñada por Isaac Asimov, quien la describió como la especialidad que se encarga de los androides. Asimov también ideó las tres leyes de la robótica. En la ciencia ficción, los individuos han soñado con robots que visitarán otros planetas, tomarían el mando o, sencillamente, realizan labores domésticas para ellos. Según otros, los primeros robots se remontan al menos al siglo III a.C., sin embargo, los mecanismos más avanzados tienen un origen más mecánico que tecnológico (Euzcateguie, 2018, p.1)

El robot humanoide Elektro, creado por Westinghouse, es considerado el primero de su tipo. Era un robot de dos metros que podía moverse y tenía una grabación de 700 palabras que le permitía imitar el inicio de una conversación. Entre 1939 y 1940 estuvo expuesta en la Exposición Universal (Euzcateguie, 2018, p.1).

Tabla 1-2: Aplicaciones de la robótica

Transporte de materiales	Manipulación de plásticos y otros materiales

Montaje	Tareas peligrosas como soldaduras, implementación de sustancias inhalantes nocivas, transporte de materiales pesados
Corte mecánico, rectificado, desbardado y pulido	Medicina
Pintura	Reciclaje

Fuente: Euzcateguie, 2018.
Realizado por: Bonilla, H., 2023.

2.2.1.1 Leyes de la robótica

Isaac Asimov, bioquímico, científico, divulgador y autor, estableció estos tres principios de la robótica (Petrovichi, 2020, p. 1). En cualquier caso, se deben seguir las siguientes tres leyes de la robótica:

Tabla 2-2: Leyes de la robótica

Ley	Descripción
Primera ley	Un robot no puede lastimar a un ser humano, ni por inacción permitir que un ser humano sufra daño.
Segunda ley	Un robot debe obedecer a un ser humano, excepto si las órdenes entran en conflicto con la primera ley.
Tercera ley	Un robot debe proteger su existencia, siempre y cuando tal protección no entre en conflicto con la primera o segunda ley.

Fuente: Petrovichi, 2020.
Realizado por: Bonilla, H., 2023.

2.2.2. Articulaciones y grados de libertad de un robot

La estructura externa de un robot industrial se compone de una disposición en forma de cadena de piezas rígidas que están conectadas por juntas. Esta cadena comienza con una base de soporte que normalmente es fija y termina con un extremo móvil libre para conectar la herramienta de trabajo al robot. Las juntas permiten producir desplazamientos, rotaciones o una combinación de ambos entre los componentes que unen (también conocidos como ejes). Hay muchos tipos diferentes de articulaciones, pero las dos que más utilizan los robots son la prismática (P) y la rotacional (R).

Ambos permiten un único Grado de Libertad (GDL), o movimiento independiente entre las partes que unen. En términos generales, es posible afirmar que el número de grados de libertad (GDL) de un robot es igual al número de sus articulaciones o ejes. La razón es que los prismáticos sólo permiten el desplazamiento relativo en una sola dirección o eje, pero no el rotacional. Se puede mover cualquier objeto, elemento o herramienta alrededor del espacio moviéndose en los ejes X, Y o Z apropiados y girándose o balanceándose en estos mismos ejes. Los robots normalmente se posicionan orientando su elemento terminal o herramienta con las otras articulaciones y usando sus primeras tres articulaciones desde la base. En este contexto, no hay más de 6 GDF, ya que

algunas articulaciones proporcionan GDF falsos que se repiten de las que proporcionan otras. Por otra parte, se habla de robots con más de 6 articulaciones y GDL, los cuales permiten una mayor accesibilidad en zonas específicas (Segovia, 2018,p .6).

2.2.2.1 Tipos de articulaciones

La mecánica de un robot está formada por una serie de componentes o articulaciones que permiten el desplazamiento entre sí de cada enlace posterior. La mayoría de los robots industriales se asemejan a la anatomía del brazo humano, por lo que a veces se utilizan palabras como "cuerpo", "codo" y "muñeca" para describir sus diferentes componentes (Commons, 2021, p.6).

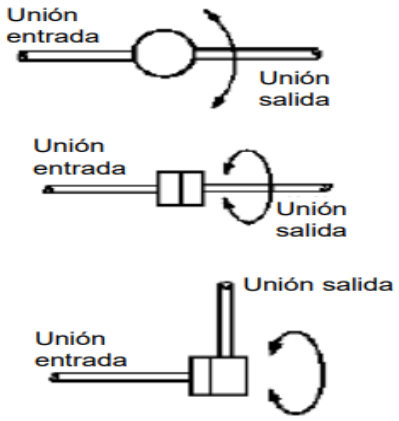


Ilustración 1-2: Tipo de articulación
Fuente: (Almeth, 2019, p.5).

Cada articulación tiene un rango de movimiento que incluye desplazamiento, rotación y combinaciones de los dos. Es posible lograr los seis tipos diferentes de articulación representados en la Ilustración 1-2 de esta manera, aunque solo la rotación y la articulación prismática se usan en robots en aplicaciones del mundo real (Commons, 2021, p.6).

El uso de varias combinaciones de juntas da como resultado varias configuraciones de robots, cada una de las cuales tiene propiedades que deben tenerse en cuenta durante el diseño y la construcción del robot, así como durante su uso. Las configuraciones más habituales son las llamadas configuraciones clásicas, que veremos en el siguiente apartado, donde solo se presta atención a las tres primeras articulaciones del robot, que son cruciales para posicionar su extremo en un punto del espacio (Commons, 2021, p.6).

- Articulación Rotacional: En este caso, los ejes de las dos articulaciones son perpendiculares entre sí.
- Articulación de Torsión: Esta junta gira entre las juntas de entrada y salida, con su eje de torsión corriendo perpendicular a los ejes de las dos juntas.

- Articulación de Revolución: En este patrón, el eje de rotación se encuentra perpendicular al de la unión de salida y se une al de entrada, por lo tanto, la unión de salida gira en torno a la de entrada (Almeth, 2019, p.5).

2.2.3. Tipos de robots

La estructura mecánica y el tipo de robot están asociados al tipo de las tres primeras articulaciones, que proporcionan una configuración específica de los tres ejes principales Ilustración 1-2 , que, como se explicó, determina la posición de la herramienta en el espacio y también el tipo de coordenadas que se emplean para determinar dicha o aquella posición. En consecuencia, tenemos los siguientes tipos básicos de robots: polares o esféricos, cilíndricos, cartesianos, SCARA y angulares o humanoides (Chávez, 2018,p .1).

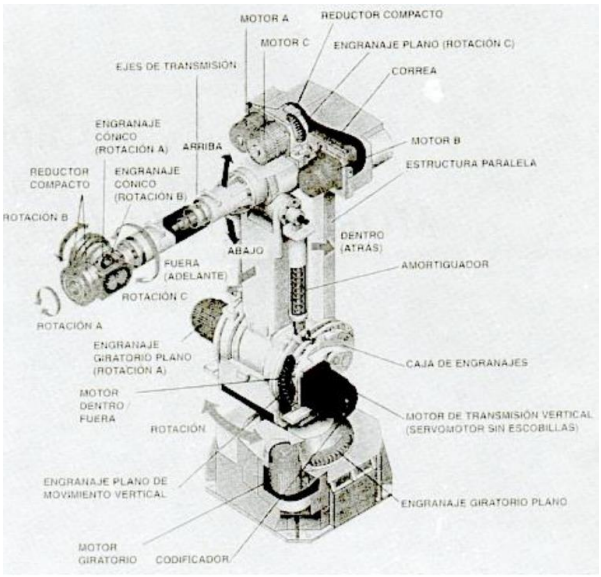


Ilustración 2-2: Estructura de robot
Fuente: (Almeth, 2019, p.5).

2.2.3.1 Tipos de robots cartesianos

Sus tres articulaciones más importantes son prismáticas, los ejes que las componen son perpendiculares entre sí y al realizar movimientos sobre ellos se genera la coordenada X, Y, Z de cada punto de acción. La forma de la construcción puede ser voladizo o pórtico. Es muy rápido y preciso, fácil de controlar, tiene un área de trabajo grande y una gran capacidad de carga, pero ocupa un espacio relativo y el elemento del cabezal de la herramienta no puede ajustarse especialmente. Se utilizan en aplicaciones que requieren un movimiento lineal de alta precisión principalmente en áreas de trabajo paralelas o de un solo plano. Si la precisión requerida no es alta, los concentradores controlados por PLC y las placas electrónicas superan con creces a este robot en términos de precio (Moreno y Chala, 2018, p.7).

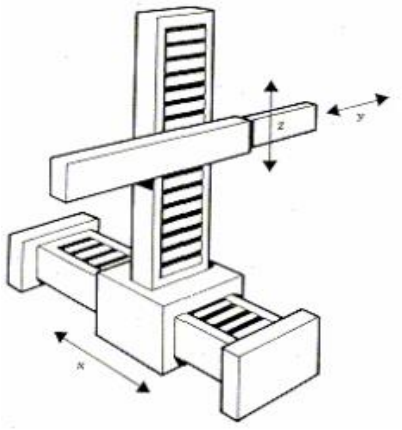


Ilustración 3-2: Robot cartesiano
Fuente: (Moreno y Chala, 2018, p.7).

Los robots tienen una estructura articulada como se muestra en la Ilustración 3-2, es decir, una serie de elementos o eslabones fácilmente móviles, que pueden trasladarse vertical, horizontal y lateralmente, y pueden combinarse con movimiento de rotación sobre un mismo eje (López et al., 2017, p.38).

En todos los robots se deben tener en cuenta los grados de libertad definidos por las siglas GDL, lo que significa que cada movimiento debe estudiarse de forma independiente, siendo posibles los movimientos articulares en el espacio tridimensional. Es por ello por lo que la cantidad de grados de libertad de un robot está compuesta por la suma de las articulaciones (López et al., 2017, p.39).

- La designación de puntos se realiza mediante coordenadas (Cartesiano: x, y, z).
- La precisión es consistente en todo el espacio de trabajo.
- Capaz de seguir una trayectoria previamente especificada.
- Estructura rígida: la distribución de carga no presenta problemas especiales.
- No funciona para puntos que están dentro de un espacio cerrado.

De acuerdo con lo expuesto en el informe del COBATAB (2022, pp. 84-86), se pueden identificar varios tipos de robots cartesianos, que incluyen modelos estándar, compactos, lineales y del tipo pórtico.

- **Robot estándar**

Puede configurarse utilizando variadas disposiciones de ejes (horizontal, vertical y de profundidad). Como ejemplo, es viable designar el eje Y como el eje vertical y el eje Z como el eje horizontal Ilustración 4-2. También, es posible integrar servomotores o actuadores neumáticos para gestionar las rotaciones en los ejes U, W y J (COBATAB, 2022, pp. 84-86).

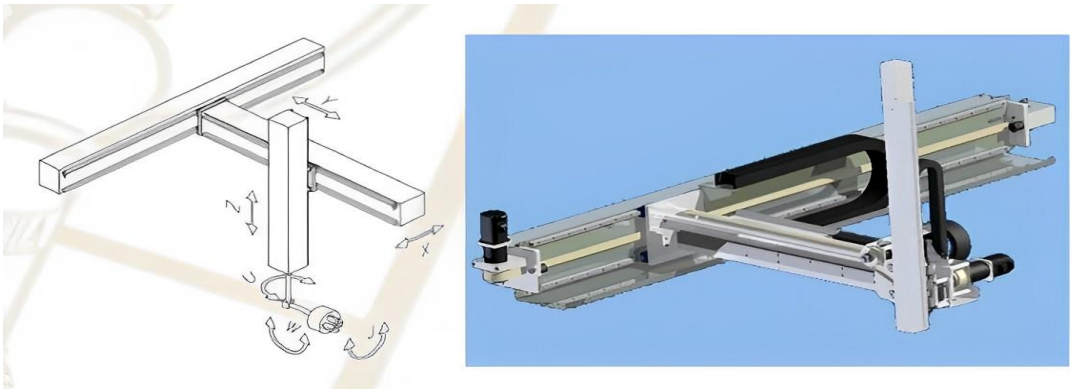


Ilustración 4-2: Robot cartesiano estándar: representación gráfica (izquierda) y ejemplo de aplicación (derecha)
Fuente: (COBATAB, 2022.)

- **Robot compacto**

Fundamentalmente, se trata de un robot estándar Ilustración 5-2, pero se distingue por tener una disposición de ejes diseñada especialmente para realizar rotaciones particulares (COBATAB, 2022, pp. 84-86).

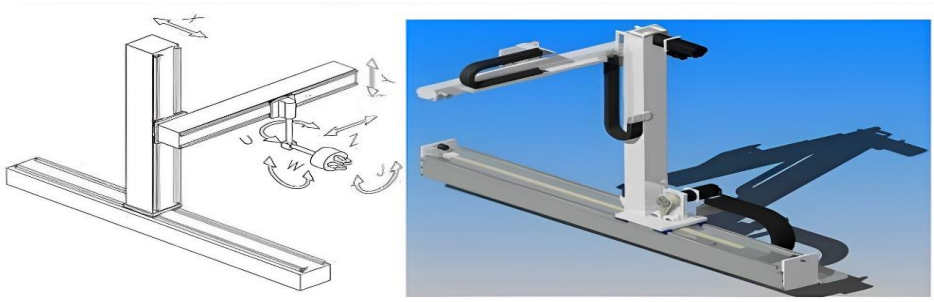


Ilustración 5-2: Robot cartesiano compacto
Fuente: (COBATAB, 2022.)

- **Robot lineal:**

Facilita la representación del desplazamiento en trayectorias comúnmente trazadas mediante rieles de conducción, Ilustración 6-2 (COBATAB, 2022, pp. 84-86).

Ilustración 6-2: Robot cartesiano lineal
Fuente: (COBATAB, 2022.)

- **Robot pórtico**

En esencia, se trata de un robot compacto en el que uno de sus ejes puede desplazarse verticalmente mediante una transmisión dual de apoyo (COBATAB, 2022, pp. 84-86).

Ilustración 7-2: Robot cartesiano pórtico
Fuente: (COBATAB, 2022.)

Por otro lado, según Muñoz (2023, pp. 1-5), en el ámbito de la robótica cartesiana Ilustración 7-2, se pueden encontrar diversos tipos que se caracterizan por sus distintos diseños constructivos y mejoras específicas adaptadas a una variedad de aplicaciones industriales. De acuerdo con el enfoque de este autor, la clasificación de los robots cartesianos se basa en consideraciones como su tamaño, capacidad de carga, velocidad y precisión. A continuación, se presentan los tipos más frecuentes de robots cartesianos:

- **Mini robots cartesianos**

Suelen tener dimensiones reducidas y se emplean principalmente en tareas que requieren alta precisión, como la microelectrónica, el ensamblaje microscópico y la fabricación a escala micro (Muñoz, 2023, pp. 1-5).

- **Robots cartesianos medianos**

Son de mayor envergadura y tienen la capacidad de gestionar cargas más pesadas. Son empleados en tareas como soldadura, ensamblaje de componentes automotrices y producción de dispositivos electrónicos (Muñoz, 2023, pp. 1-5).

- **Grandes robots cartesianos**

Son los de mayor tamaño y tienen la capacidad de gestionar cargas pesadas. Se emplean en aplicaciones que incluyen la manipulación de materiales en la industria del acero y la fabricación de maquinaria pesada (Muñoz, 2023, pp. 1-5).

- **Robots cartesianos de alta velocidad:**

Estos robots destacan por su capacidad de desplazamiento veloz y son especialmente adecuados para tareas que demandan altas velocidades y precisión, como el envasado (Muñoz, 2023, pp. 1-5).

- **Robots cartesianos de alta precisión**

Estos robots tienen la habilidad de ejecutar movimientos extremadamente precisos y se emplean en situaciones que demandan un alto grado de precisión, como la producción de elementos ópticos y la inspección de productos (Muñoz, 2023, pp. 1-5).

2.2.4. Tarjetas de desarrollo

Es un circuito electrónico que contiene un microcontrolador o un componente lógico y otros componentes como se muestra en la Ilustración 8-2, los puertos, los conectores y los reguladores, que permiten la incorporación de hardware externo. Su utilidad es la de poder conectarse a varios aparatos, tales como: sensores, módulos inalámbricos, tarjetas de memoria micro SD, etc. Están diseñados para tener entradas y salidas analógicas y digitales, así como puertos de conexión seriales, USB, reguladores de voltaje e indicadores de transmisión y recepción de datos. Todo ello depende de las funcionalidades de cada uno de los microcontroladores que existen en el mercado y de los distintos fabricantes (Ramón, 2020, p. 21). Sin embargo, los componentes esenciales de la placa son el microcontrolador, el puerto JTAG, conectores como USB, DC Jack, puertos de memoria como MicroSD y un regulador (Palma y Rodríguez, 2011, p.44-51).

2.2.4.1 Elementos comunes de las tarjetas de desarrollo

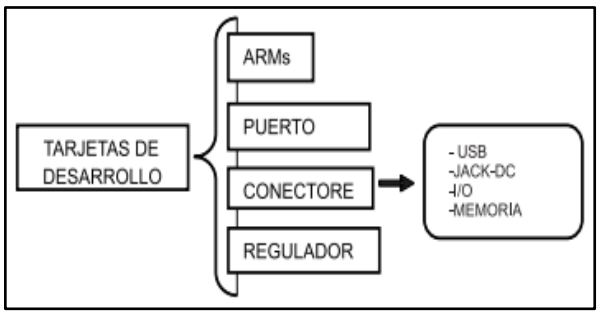


Ilustración 8-2: Elementos comunes en las tarjetas
Fuente: (Palma y Rodríguez, 2011, p.3).

- **Microcontrolador ARM**

Se basan en la arquitectura RISC, pero se les han hecho modificaciones para que el dispositivo sea más eficiente y, por lo tanto, consuma menos energía y tenga mayor capacidad de procesamiento, es decir, más instrucciones por segundo. Esto ha hecho que la arquitectura RISC se convierta en una opción viable para diseñar dispositivos como sistemas embebidos y dispositivos de uso general, como teléfonos celulares o consolas de videojuegos. Las arquitecturas ARM se caracterizan por tener un procesador "core" que se ha vuelto más pequeño y veloz con el paso del tiempo (Palma y Rodríguez, 2011, p.44-51).

- **Reguladores**

Los reguladores de voltaje reducen la tensión de entrada hasta valores apropiados para los componentes, para tarjetas de desarrollo, por ejemplo, pueden tener voltaje de 5V (estándar TTL), 3.3V que es el voltaje con el que operan los ARM, y en algunos casos de 2.5V y 1.8V para otro tipo de lógica (Palma y Rodríguez, 2011, p.44-51).

- **Conectores**

Los más comunes para tarjetas de desarrollo son los conectores:

- **USB y mini USB**

Permite la transferencia de datos a una velocidad de 12 Mbps y es ideal para conectar periféricos. Para tarjetas de desarrollo, el conector Mini-USB se usa en modo dispositivo y el USB-A se usa en modo host.

- **JACK- DC**

Este conector se utiliza para proveer energía a las tarjetas de forma externa, en caso de que no sea necesario conectarse a un computador.

- **Conectores I/O**

Normalmente están conectados a cada uno de los pines del ARM, lo que permite unirlos a otros componentes.

- **Memoria MicroSD**

La mayoría de las placas de desarrollo tienen interfaces SPI, lo que permite la compatibilidad con tarjetas de memoria SD, MMC o MicroSD, estas últimas son las más comunes porque son más sencillas de utilizar, ocupan menos espacio y son más económicas, por lo tanto, para facilitar la comunicación entre el ARM y la tarjeta (Palma y Rodríguez, 2011, p.44-51).

2.2.5. Tipos de tarjetas de desarrollo

Actualmente existen en el mercado muchos tipos de tarjetas de desarrollo, las cuales varían según su tamaño, velocidad de procesamiento y capacidades, así como plataformas desarrolladas por diferentes fabricantes que brindan aplicaciones y herramientas de prototipado de tarjetas de desarrollo en un entorno gráfico. Las tarjetas más populares son: Arduino, Raspberry Pi y Intel Galileo (Armas y Cerda, 2020, p.16).

- **Arduino**

Es una placa electrónica de hardware libre mírese la Ilustración 9-2, se encuentra basada en un microcontrolador reprogramable que varía según el modelo, además cuenta con una plataforma de software para desarrollar diversos proyectos interactivos de bajo costo y un ambiente amigable.

Las tarjetas Arduino son en realidad muy similares, todas tienen pines de salida y entrada que permiten leer el dispositivo, también envía datos a través de los pines de salida para activar o desactivar los diferentes actuadores conectados, ya sean analógicos o digitales (Armas y Cerda, 2020, p.16).

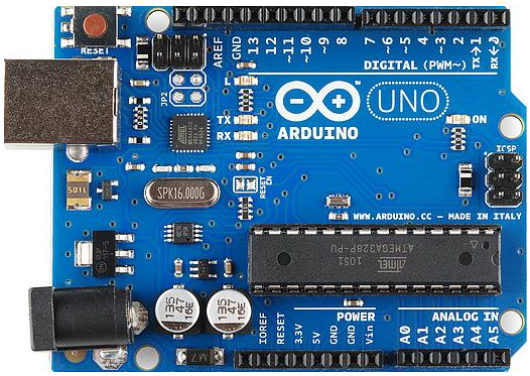


Ilustración 9-2: Tarjeta de desarrollo Arduino
Fuente: Arduino, 2020.

- **Raspberry Pi**

Se trata de un microcontrolador que ofrece múltiples facilidades para proyectos de electrónica, programación y monitoreo de actuadores y sensores, estos últimos con el fin de elaborar proyectos de tecnología Ilustración 10-2. Tienen una capacidad de almacenamiento de datos limitada que solo puede ser mayor que una tarjeta de crédito. Actualmente, los modelos A y B están disponibles.

Modelo A: Dado que requiere poca energía para funcionar es una buena opción para proyectos de energía solar, eólica o de almacenamiento. Además de que solo tenga 256MB de memoria interna, lo que limita la ejecución de programas y aplicaciones complejos (Ramón, 2020, p. 21).

Modelo B: Tiene un costo un poco mayor al del A, ya que incorpora más funciones, la primera de ellas es una memoria de 512MB, además cuenta con puertos USB adicionales que permiten conectar un teclado y un ratón (Ramón, 2020, pp. 22-23).

Ilustración 10-2: Tarjeta de desarrollo Raspberry Pi
Fuente: Ramón, 2020.

- **Intel Galileo**

Se basa en la arquitectura de 32 bits del sistema Intel Pentium, este dispositivo fue creado con el objetivo de funcionar con voltajes de 3.3V a 5V Ilustración 11-2, es similar a Arduino y se puede programar con el software de este último, se puede conectar a un computador, microcontroladores y Arduino; esta tarjeta contiene físicamente ranuras USB, puerto Ethernet, espacio para una tarjeta Micro SD y conectores de alimentación (González y Carrillo, 2019, p.15).

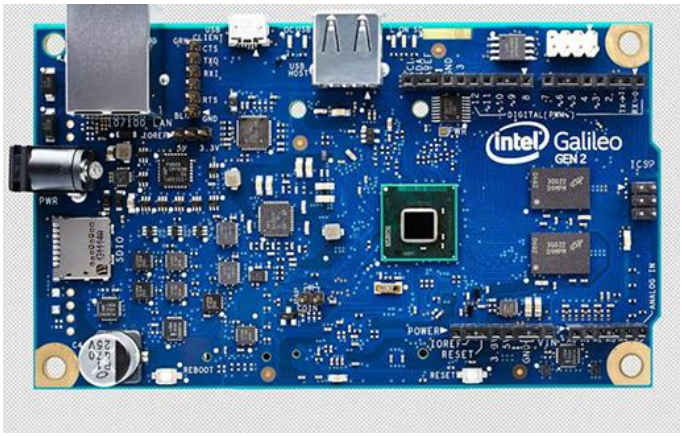


Ilustración 11-2: Tarjeta de desarrollo Intel Galileo
Fuente: Ramón, 2020.

2.2.6. Actuadores eléctricos

Un actuador eléctrico es una máquina que consta de un motor eléctrico y un reductor que permite la activación de cualquier mecanismo para realizar acciones o mover objetos. Solo requiere energía como fuente de energía y su estructura es sencilla en comparación con otros actuadores. Los actuadores eléctricos se utilizan para sistemas de posicionamiento y monitoreo remoto, movimiento de brazo articulado, encolado, rectificado, movimiento automático de objetos de laboratorio con alta precisión y velocidad, elevación vertical y movimiento de cargas. Los actuadores eléctricos no deben confundirse con los actuadores mecánicos, una diferencia notable entre ellos es que el actuador frontal transmite potencia a través de una acción electromecánica y la potencia del motorreductor transmite potencia a través de una operación mecánica precisa (Sdin, 2020, p.1).

2.2.6.1 Motores paso a paso

Este tipo de motores se los denomina motores de pasos Tabla 3-2, los bobinados que los componen son parte del estator, también es un imán estable y sus controladores están diseñados para mantener una posición fija y para girar en sentido horario o antihorario. La mayor parte de su progreso se da a través de frecuencias de audio, lo que les permite girar con mayor rapidez, un

control adecuado permite que avance o se detenga de forma instantánea y en una posición específica. Una de sus propiedades es la desaceleración del par motor o del par de giro, lo que provoca la firmeza de su detención cuando no está girando, al estar sumergido en este estado, su fuerza laboral continúa siendo aplicada en su eje (Ricaurte y Minda, 2017, pp. 34-35).

Características de un motor paso a paso

- Voltaje
- Resistencia eléctrica
- Grados por paso

Tabla 3-2: Tipos de motores paso a paso

Tipos	Características
Imán permanente	Tiene un imán que atrae el campo magnético que lo hace funcionar. Suele ser empleado en la fabricación de papel y cartulina para impresoras, así como en el cabezal de impresión de estas.
Reluctancia variable	Tiene un engranaje de hierro dulce que, cuando el estátor está excitado, y bajo la acción de su campo magnético, ofrece una menor resistencia para atravesarse por su flujo en la posición de equilibrio.
Unipolares	Los motores unipolares son relativamente fáciles de controlar porque tienen dos devanados.
Bipolares	Los motores bipolares requieren más circuitos para controlarlos y para proveer energía. La estructura de los motores bipolares requiere que las bobinas reciban corriente en ambos sentidos, no solo en un encendido y apagado como en los unipolares.

Fuente: Carletti, 2022, p.1
Realizado por: Bonilla, H., 2023.

2.2.7. Driver de motor

Un controlador de motor es un amplificador de baja corriente; su trabajo es tomar una señal de control de baja corriente y convertirla en una señal de corriente más alta que se puede usar para impulsar un motor (Ortiz, 2018, p. 1).

2.2.7.1 Driver de motor paso a paso

Un controlador de motor paso a paso es un componente electrónico que convierte varias señales de entrada en movimiento de motor paso a paso sin controlar directamente el motor paso a paso

en sí. En algunos casos, los controladores de motores de pasos son diseñados específicamente para trabajar con un protocolo de comunicación establecido, como el analógico o el CANbus, pero el concepto básico es sencillo: no tienen la capacidad de controlar por sí mismos el motor, a excepción de cuando se les indica que lo hagan por parte de un dispositivo de control, como un PLC o un ordenador (ZIKODRIVE, 2019, p.1).

2.2.8. Interfaz de potencia

Las interfaces de alimentación son circuitos de acoplamiento que se utilizan para alimentar grandes actuadores eléctricos que demandan corriente. Las interfaces de potencia incluyen electrónica de estado sólido como SCR, TRIAC, BJT, AOP y otros, así como dispositivos electromecánicos como relés y contactores (Gómez et al., 2016, p.45).

2.2.9. Finales de carrera

Un limit switch o sensor de finalización, es básicamente un interruptor que se coloca en puntos clave de la carrera de un objeto móvil para que este pueda informarle a la persona su ubicación. En términos sencillos, es un dispositivo que permite conocer la ubicación específica de cualquier objeto que se mueva (Mecafenix, 2021, p.1).

Dado que son muy simples, tienen pocas partes y cada una cumple una función específica.

- **Cabezal:** Es el componente responsable de convertir el movimiento del actuador en movimiento de contacto. Al final de la carrera, el mecanismo hace contacto eléctrico cuando el actuador se mueve correctamente.
- **Cuerpo del interruptor:** El bloque de contactos se encuentra dentro del cuerpo del interruptor.
- **Bloque de contactos:** Los contactos eléctricos del final de carrera están presentes.
- **Base:** El modo de conexión de los terminales del bloque de terminales se almacena en la base (Mecafenix, 2021, p.1).

2.2.10. Pistones neumáticos

Un cilindro o pistón es un dispositivo que transforma energía potencial en energía cinética o presión. La fuerza de empuje es proporcional a la presión del aire y al área del pistón (MICRO, 2019, p.1).

2.2.10.1 Características de los pistones neumáticos

El cilindro neumático se muestra en la Ilustración 12-2, están compuesto por diferentes componentes:

- Cilindro provisto de un embolo
- Pistón
- Vástago

Son numerosas las variedades de cilindros neumáticos, existen formas y tamaños, pero la mayoría son fabricados con estas especificaciones:

- Diámetro: entre 6 a 320 mm
- Carreras: de 1 a 2000 mm
- Fuerza: de 2 a 50000 N
- Velocidad del émbolo entre 20 mm/s y 1 m/s

(MICRO, 2019, p.1).

Ilustración 12-2: Pistones neumáticos
Fuente: Hernández, 2020.

2.2.10.2 Tipos de pistones neumáticos

Pistones neumáticos de simple efecto: Uno de sus movimientos se origina por el aire comprimido, mientras que el otro es causado por una acción opuesta, en este caso un resorte que se encuentra en el interior del cilindro. Este resorte se puede ubicar opcionalmente entre el pistón y la tapa delantera (con resorte delantero) o entre el pistón y la tapa trasera (con resorte trasero) (MICRO, 2019, p.1).

Pistones neumáticos de doble efecto: El cilindro se activa por el aire comprimido que ingresa en ambos lados, realiza una cantidad de trabajo aprovechable en ambos sentidos.

Pueden existir los siguientes tipos de cilindros de doble efecto:

- Cilindro neumático de impacto
- Cilindro neumático de fuelle
- Cilindro neumático multiposición
- Cilindro neumático guiado
- Cilindro neumático sin vástago (Peroni, 2018, p.3).

2.2.11. Electroválvulas

Constan de una válvula neumática como dispositivo que envía la señal de salida y una electroválvula como dispositivo de accionamiento. Cuando se aplica corriente eléctrica al solenoide, se crea una fuerza electromagnética que hace que se mueva la armadura unida a la leva de la válvula (Pazmiño, 2022, pp. 26-28).

- **Electroválvulas monoestables**

Tienen una bobina y regresan automáticamente a través de un resorte cuando el solenoide ya no está alimentado por electricidad (Pazmiño, 2022, pp. 26-28). Las válvulas monoestables son aquellas que cuentan con una posición de equilibrio definida, en la cual permanecen de manera continua a menos que se actúe sobre ellas mediante un dispositivo de control. Por lo general, el retorno a esta posición de equilibrio se logra mediante un resorte. En una válvula monoestable de dos posiciones, la posición estable se establece en función de la posición del resorte, que suele ubicarse convencionalmente en el lado derecho. En el caso de válvulas monoestables de tres posiciones, la posición de equilibrio se encuentra en el centro. Además, estas válvulas pueden ser de tipo normalmente abiertas (NA), permitiendo el flujo del fluido a presión a través de ellas en su posición de equilibrio, o normalmente cerradas (NC), bloqueando el flujo de fluido o permitiendo que fluya desde los elementos actuadores hacia el escape en su posición estable (QBPROFE, 2021, p.1).

- **Electroválvulas Biestables**

Las válvulas biestables son aquellas que no poseen una única posición de equilibrio estable en la Tabla 4-2 se muestran las electroválvulas más utilizadas. Esto significa que, aunque se anule la señal que la colocó en una posición determinada, la válvula permanecerá en esa misma posición hasta que se active una señal nueva que indique una posición diferente. Si se activan dos señales al mismo tiempo, prevalecerá la señal más antigua. La forma en que se controla cada posición en la válvula y el dispositivo de mando correspondiente se representa simbólicamente agregando el símbolo del accionamiento en el lateral del cuadrado que representa la válvula. A veces, también se denominan como "NA" o "NC". En estos casos, se considerará "NA" cuando se encuentre en la posición de la derecha en el caso de dos posiciones o en la posición central en el caso de tres posiciones, y de manera análoga para "NC" (QBPROFE, 2021, p.1).

Tabla 4-2: Electroválvulas más utilizadas

Tipo	Descripción
Electroválvula De 3/2 Vías Monoestable, Normalmente Cerrada	Esta válvula de asiento, normalmente cerrada (NC) es actuada directamente por un solenoide y devuelta a su posición de reposo por un muelle.

Electroválvula De 3/2 Vías Monoestable, Normalmente Abierta	Esta válvula es semejante a la normalmente abierta, excepto que su mecanismo de cierre es diferente, de este modo, permanece abierta en el estado de inactividad. En esta configuración, el cabezal está conectado a la fuente de alimentación 1. Al aplicar una corriente eléctrica a la leva, se eleva y se cierra el asiento superior, por lo tanto, se interrumpe la corriente. Al mismo tiempo, el asiento inferior libera el aire de la salida 2 hacia el escape 3.
Electroválvula De 5/2 Vías, Pilotada	La válvula de 5/2 vías tiene una función semejante a la de 4/2 vías. La peculiaridad de este modelo es que cuenta con dos escapes independientes, en contraste con el 4/2, que tiene un único conducto de escape. Al encontrarse en posición inicial, el resorte hace que la corredera 1 con 2 y 4 con 5 se conecten, mientras que 3 queda separado. Al activarse el solenoide, se abre la válvula auxiliar y el aire fluye del lado izquierdo de la corredera hacia el lado derecho, lo que hace que la jaula se mueva hacia el lado derecho. Como consecuencia, el escape 5 se interrumpe y el aire fluye ahora desde el 1 hacia el 4.
Electroválvula De 5/2 Vías, Biestable	Las válvulas descritas anteriormente utilizan un resorte para devolver la válvula a su estado original. Para solenoides dobles, el resorte se reemplaza por otro solenoide.

Fuente: Pazmiño, 2012, pp. 26-28
Realizado por: Bonilla, H., 2023.

2.2.12. Fuentes de poder

La fuente de poder convierte la corriente alterna que recibe de la red eléctrica en corriente continua, o en una variedad de corrientes continuas con la misma frecuencia y voltaje. También se utiliza para alimentar aparatos electrónicos como televisores, computadoras, routers, entre otros. (ConceptoABC, 2022, p.1).

- **Fuente AT**

La fuente AT tiene 3 tipos de tomas de salida como se muestra en la Ilustración 13-2.Los primeros dos tipos de procesadores se encuentran en la placa madre. Los dos últimos tipos, que son variables, provienen de los periféricos no enchufados que se encuentran en un puerto de la placa madre, como es el caso de las unidades de discos duros, CD-ROM, disqueteras, entre otros. La unión a la placa madre se realiza a través de dos clavijas de 6 pines cada una, que deben colocarse

de manera que los cables negros de ambos se encuentren unidos en el centro (Ilustración 14-2) (Rocha, 2019, pp. 1-72).

P-8					
Pin 1	Pin 2	Pin 3	Pin 4	Pin 5	Pin 6
Pwr gnd	key	+12 V	-12 V	Gnd	Gnd
Naranja		Amarillo	Azul	Negro	Negro

P-9					
Pin 1	Pin 2	Pin 3	Pin 4	Pin 5	Pin 6
Gnd	Gnd	-5 V	+ 5 V	+ 5 V	+ 5 V
Negro	Negro	Blanco	Rojo	Rojo	Rojo

Ilustración 13-2: Disposición de pines, colores y señales de ambos conectores
Fuente: Rocha, 2019.

- **Fuente ATX**

Evidencia en el interruptor de encendido, que en lugar de conectar o desconectar la corriente de 220VAC o 110VAC como lo hace el de la fuente AT , envía una señal a la fuente primaria, la cual indicará que se encienda o se apague, mientras que la secundaria permanecerá encendida y siempre se conectará la corriente de 220VAC o 110VAC, lo que permitirá realizar conexiones y desconexiones por software (Rocha, 2019, pp. 1-72).

La unión a la placa madre se realiza a través de dos clavijas de 6 pines cada una como se muestra en la Ilustración 14-2, que deben colocarse de manera que los cables negros de ambos se encuentren unidos en el centro (Rocha, 2019, pp. 1-72).

P-8					
Pin 1	Pin 2	Pin 3	Pin 4	Pin 5	Pin 6
Pwr gnd	key	+12 V	-12 V	Gnd	Gnd
Naranja		Amarillo	Azul	Negro	Negro

P-9					
Pin 1	Pin 2	Pin 3	Pin 4	Pin 5	Pin 6
Gnd	Gnd	-5 V	+ 5 V	+ 5 V	+ 5 V
Negro	Negro	Blanco	Rojo	Rojo	Rojo

Ilustración 14-2: Disposición de pines, colores y señales un conector ATX de 20 pines
Fuente: Rocha, 2019.

- **Características de las fuentes de poder**

La fuente ATX está compuesta por dos partes: una parte principal que es la antigua fuente AT (con algunos cambios), y una secundaria. La principal diferencia en cuanto al funcionamiento son sus características.

- **Versatilidad**

Las fuentes de energía se pueden controlar de forma mecánica o digital porque algunas de ellas tienen interruptores en las ranuras traseras de la carcasa de la computadora, mientras que otras se encienden y apagan mediante un software de computadora que se puede operar de forma manual o automática (SDI, 2022, p.1).

- **Eficacia**

Es un factor importante en el funcionamiento de las fuentes de alimentación porque durante la conversión de la corriente eléctrica ocurre un proceso termodinámico, lo que significa que la energía se pierde en forma de calor por efecto Joule o cualquier otro fenómeno (SDI, 2022, p.1).

- **Adaptabilidad**

Una fuente de alimentación debe poder ser parte de cualquier tipo de hardware de computadora moderno, sin importar a qué generación pertenezca, porque a medida que evoluciona la tecnología de computadoras de escritorio y portátiles, también lo hace el dispositivo (SDI, 2022, p.1).

- **Complejidad**

La arquitectura interna de una fuente de alimentación tiende a ser complicada por su mecanismo de conversión de corriente, que se basa en la física del estado sólido y consta de diferentes configuraciones de condensadores, transistores y diodos, junto con otros componentes electrónicos (SDI, 2022, p.1).

2.2.13. Lenguajes de programación

Un lenguaje de programación es una herramienta diseñada para representar algoritmos de manera lógica en una computadora. Aunque existen varios lenguajes de programación, el único entendido directamente por las computadoras es el lenguaje máquina, compuesto por secuencias de ceros y unos. Los demás lenguajes necesitan traducción o interpretación a lenguaje máquina (Zapata, 2016, pp. 14-15).

Tipos de lenguajes de programación

- **Lenguajes de máquina:** El más antiguo, requiere traducción a lenguaje binario para que las computadoras lo entiendan (Assembler, 2018, p.1).

- **Lenguaje ensamblador:** Diseñado para gestionar hardware de manera más eficiente, con una estructura que identifica los componentes de la computadora en la memoria como registros o campos (Zapata, 2016, pp. 14-15).

- **Lenguaje de medio nivel:** Ejemplos como C comparten características de bajo nivel, como gestión de memoria y registros, pero tienen una sintaxis más avanzada (Zapata, 2016, pp. 14-15).

- **Lenguaje de alto nivel y de muy alto nivel:** Los lenguajes de alto nivel se centran en expresar algoritmos de manera comprensible para los humanos la Tabla 5-2 muestra los lenguajes de programación más utilizados, mientras que los de muy alto nivel buscan resolver

problemas de procesamiento de datos de forma más sencilla y rápida para usuarios comunes (Zapata, 2016, pp. 14-15).

Tabla 5-2: Lenguajes de programación más utilizados

Lenguaje	Descripción
C	Es un lenguaje muy sencillo y claro una vez que se domina. El lenguaje C se utiliza principalmente para la creación de programas de computadora y aplicaciones de escritorio.
C++	Sus capacidades están diseñadas para ser más apropiadas para el desarrollo de aplicaciones y software más complejos, además, permite la implementación de patrones de diseño en el código.
C#	C# es muy utilizado para desarrollar aplicaciones para Windows, aplicaciones de escritorio, aplicaciones web y aplicaciones móviles. Además, C# es uno de los lenguajes de programación más populares para la programación de videojuegos y aplicaciones de realidad virtual.
Java	Es el idioma más popular a nivel mundial debido a que tiene tres características: es confiable, sencillo de aprender y escribir y cuenta con un gran número de hablantes. Adicionalmente, es un idioma que se puede utilizar para una variedad de propósitos.
Python	Python es un lenguaje de programación relativamente nuevo que tiene como objetivo ser de uso general. Se puede utilizar para realizar prácticamente cualquier tarea que requiera un lenguaje de programación.

Fuente: Assembler, 2018
Realizado por: Bonilla, H., 2023.

2.2.14. Diseño mecánico

El diseño mecánico implica la aplicación de la ingeniería mecánica a las dimensiones y formas del producto antes de que se fabriquen, con el objetivo principal de satisfacer los requisitos puramente mecánicos y las necesidades que requieren (Prisma, 2021, p.1).

2.2.15. Ensamble

El ensamblaje ha sido y seguirá siendo el mejor método de unificación, a lo largo de la historia de la evolución humana, en el proceso de construcción, carpintería y las herramientas que hicieron las cosas que todos conocemos. Los primeros montajes se basaban en cuerdas flexibles,

posteriormente cajas de madera y clavos, los materiales compuestos como el metal tuvieron gran importancia y hubo un gran desarrollo en ellos (Rodríguez, 2021, p.1).

2.2.16. Subensamble

Un subensamblaje es un grupo de piezas diseñadas para encajar posteriormente en un conjunto más grande. Se utilizan en la fabricación para crear mecanismos o circuitos, proporcionando portabilidad y precisión en las funciones. Estos subensambles son esenciales en dispositivos eléctricos o mecánicos, garantizando la realización de tareas específicas (Cad, 2020, p.5; Ferro, 2021, p.1).

2.2.17. Producto terminado

Un producto manufacturado es una mercadería que ha sido terminada y que ya está lista para ser vendida y distribuida. Es un producto que se consume directamente del empaque, por lo tanto, está destinado al público en general. Para que un bien sea ingerido, es necesario que se inicie un proceso de manufactura. La sustancia que se emplea como materia prima en este procedimiento es la materia prima, que es el punto de partida de cada línea de producción (Ceupe, 2020, p.2).
El producto resultante puede ser entonces un producto intermedio o un producto final, según el tipo de empresa. Los intermedios son preparaciones que no tienen un uso real por sí mismos, sino que sirven como complemento para la fabricación de productos listos para el consumo (Ceupe, 2020, p.4).

Características de un producto terminado:

- Es la etapa final dentro de la cadena productiva.
- Es un producto listo para su consumo.
- Se utilizan otros recursos, como materia prima o bienes intermedios se emplean para llevar a cabo su producción.
- Son productos que se elaboran en función de las necesidades del mercado.
- Las organizaciones que se especializan en elaborar productos terminados son denominadas empresas industriales.

2.2.18. Tiempos de ciclo

El tiempo de ciclo es el tiempo entre dos unidades consecutivas de un producto durante un período de tiempo, es decir, la tasa de producción promedio del producto durante ese período. Obviamente, el tiempo entre dos unidades de un producto no siempre es el mismo. Por lo que es necesario definirlo como tiempo promedio.

Se puede definir para equipos y para plantas completas. En el caso de una máquina, medirá la frecuencia con la que se revisa el equipo en esa máquina y, en el caso de una fábrica, medirá el tiempo promedio entre dos máquinas que están completamente cargadas cuando finalmente se retira el equipo. El tiempo de ciclo es un conjunto de parámetros para cada proceso. Este lapso está determinado por una variedad de factores que afectarán a todos los aspectos de la productividad y la administración de la producción (Paredes, 2019, p.3).

2.2.19. Productividad

La productividad se refiere a la capacidad de desarrollar una tarea dentro de un tiempo específico y utilizando una cierta cantidad de recursos asignados que tiene nuestro negocio. Evaluar la importancia de la productividad de nuestra compañía es crucial para optimizar nuestros procedimientos. La capacidad de producir más en menos tiempo es lo que se denomina productividad, por lo tanto, si una compañía puede incrementar su productividad, esto se traduce en que el valor de sus productos es mayor que el de sus materias primas. Este producto se está desarrollando, por lo tanto, hablaremos de la productividad de la planta (Miranda, 2018, p.65):

Mejorar la productividad en las industrias:

En cualquier sector industrial que se desempeñe una empresa es importante también medir la productividad (Miranda, 2018, p.65):

- Aliviar el trabajo manual.
- Integrar nuevas tecnologías y procesos de actualización.
- Disponer de una fuente con todos los datos necesarios agrupados.
- Invierta en la educación y capacitación de sus empleados.
- Invertir en mantenimiento

2.3 Bases conceptuales

- **Automatización:** Se denomina así a cualquier labor que es realizada por máquinas en lugar de personas, es decir, es la suplantación de procedimientos manuales por sistemas de computación (Gandhi, 2022, p.1).
- **Línea de ensamble:** Es el procedimiento que se emplea para armar un objeto o artículo (Gandhi, 2022, p.1).
- **Eje:** Cada una de las formas en las que el robot puede moverse o una sección de este. Pueden ser ejes o líneas de desplazamiento que se extiendan longitudinalmente sobre sí mismos (articulación prismática) o girar en ellos (rotación) (Gandhi, 2022, p.1).
- **Hardware:** La parte de una computadora o sistema informático que es física (Significados.com, 2020, p.1).

- **Robot cartesiano:** Es una mesa que puede moverse en cualquier dirección, combinando los ejes x, y, z (EduLAB. 2018.pp. 1-24).
- **Robot:** Manipulador mecánico, modificable y de aplicación universal. Se denomina un sistema híbrido que combina actividades de computación y físicas (EduLAB. 2018.pp. 1-24).
- **Robótica:** Se trata de una especialidad de la ciencia que se encarga de la creación y diseño de aparatos que sean útiles para el ser humano (EduLAB. 2018.pp. 1-24).
- **Sensor:** El programa informático toma la decisión de qué hacer a partir de la información que tiene y de las instrucciones que tiene (EduLAB. 2018.pp. 1-24).
- **Software:** Es un sistema de procedimientos que se encarga de que una computadora realice una sola función específicas (Leal y Pérez, 2022, pp.10-11).

CAPITULO III

3. MARCO METODOLÓGICO

3.1 Tipo de estudio

El presente trabajo es una propuesta tecnológica la cual requiere de una investigación aplicada de conocimientos teóricos y prácticos existentes en ingeniería, para abordar un problema específico el cual es el mejoramiento la línea de ensamble con la automatización de la línea de ensamble basándose en un robot cartesiano, se considera el desarrollo tecnológico nuevo y existente para adaptarlo al prototipo.

3.2 Tipo de Investigación

3.2.1. Investigación documental

La investigación documental es un método de investigación que se basa en el análisis crítico y sistemático de fuentes escritas y electrónicas disponibles. A través de la búsqueda y examen exhaustivo de libros, artículos, informes, tesis y otros documentos, se recopila, organiza y evalúa la información relevante para abordar un tema específico. Este enfoque permite comprender a fondo el estado actual del conocimiento sobre el tema, establecer fundamentos teóricos sólidos y proporcionar una base informativa crucial para investigaciones ulteriores

3.2.2. Investigación descriptiva

La investigación descriptiva tiene como objetivo principal describir las características, propiedades y fenómenos de un tema o situación particular. A través de la recopilación y análisis de datos, se busca brindar una representación precisa y detallada de cómo es o se manifiesta un fenómeno en específico. Este tipo de investigación es crucial para comprender las variables involucradas, sus relaciones y comportamientos. Además, sirve como base para generar hipótesis y diseñar investigaciones posteriores, proporcionando información valiosa que puede utilizarse en la toma de decisiones informadas en distintos ámbitos.

3.3 Metodología

3.3.1. Método deductivo

El método deductivo es fundamental debido a su estructura lógica y rigurosa en el proceso de investigación. Partiendo de premisas generales, permite deducir conclusiones específicas y

coherentes relacionadas con el tema de estudio. Al aplicar este método se podrá establecer teorías, definir hipótesis y diseñar experimentos.

3.4 Técnicas de procesamiento de datos

Las técnicas de procesamiento e interpretación de datos se delinean a partir de la recolección de información esencial para el desarrollo del prototipo de línea de ensamble propuesto. Utilizando las herramientas del paquete Office, específicamente Excel y Word, se llevará a cabo la tabulación y análisis detallado de los datos en la sección de resultados. Este análisis será fundamental para extraer conclusiones precisas y generar recomendaciones sólidas en el marco de este estudio.

3.5 Parámetros del diseño del sistema

3.5.1. Requerimientos del prototipo

Con la fundamentación teórica se define la estructura del prototipo de una línea de ensamble, para la correcta selección de los recursos a necesitar y tecnologías a emplearse se propone requisitos propios que deberá realizar el prototipo:

- Almacenar componentes en instancias diferentes.
- Implementar un sistema robótico cartesiano electromecánico capaz de lograr movimientos tanto horizontales como verticales en todos los espacios del prototipo.
- Establecer un módulo automático de alimentación de componentes para el ensamble.
- Construir un módulo simple para transportar del producto en proceso mediante un juego simple de poleas.
- Utilizar finales de carrera para setear el robot cartesiano evitando posibles errores en la línea de ensamble.
- Aprovisionamiento de piezas que genere optimización del prototipo involucrado en la línea de ensamble.

3.6 Consolidación general del prototipo

La consolidación general del prototipo se muestra en la Ilustración 15-3, se consideró la colocación de un sistema electromecánico para conformar el modelo, describiendo el prototipo de una línea de ensamble de partes plásticas. La operación del prototipo de línea de ensamble empieza en la dosificación manual de las unidades de Arduino, bases y tapas respectivamente, las tarjetas de desarrollo de Arduino no contaran con una dosificación automática a comparación de las bases y tapas, estos componentes permiten regular la velocidad de salida de las bases y tapas,

los módulos realizaran su dosificación desde el momento que el robot cartesiano es puesto en posición inicial para realizar el ensamble.

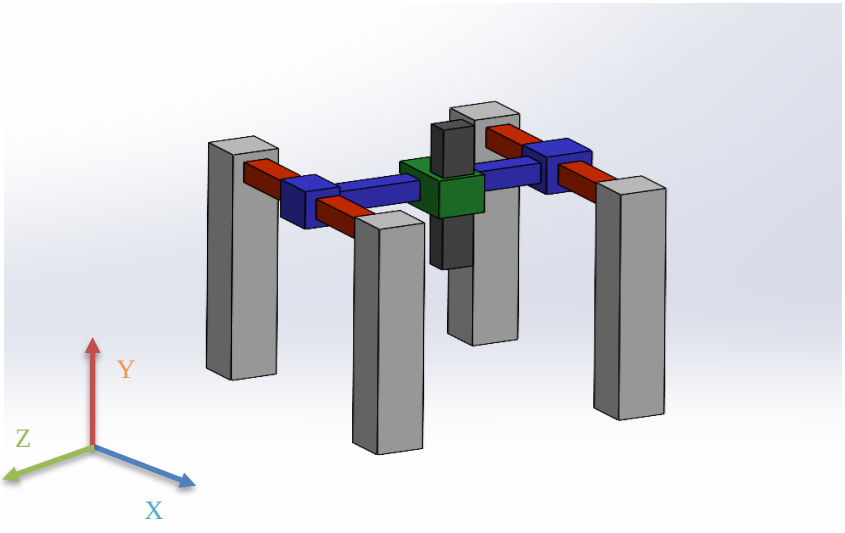


Ilustración 15-3: Consolidación del prototipo
Realizado por: Bonilla, H., 2023.

El prototipo se debe diseñar para ensamblar 5 unidades con apertura a la modificación para ensamblar un número mayor de piezas, el módulo simple para el transporte de unidades deberá cumplir con los requerimientos necesarios del prototipo, con un desplazamiento mayor a los 200 mm mediante un par de juego de poleas.

El robot cartesiano debe contar con un área de desplazamiento mayor a 500x100x500 mm para cumplir con los requerimientos necesarios, el prototipo de línea de ensamble de partes plásticas se encuentra sobredimensionada para posibles modificaciones dicho de otra manera el prototipo no se encuentra limitado por los componentes, características y diseños preliminares.

3.7 Arquitectura general del prototipo

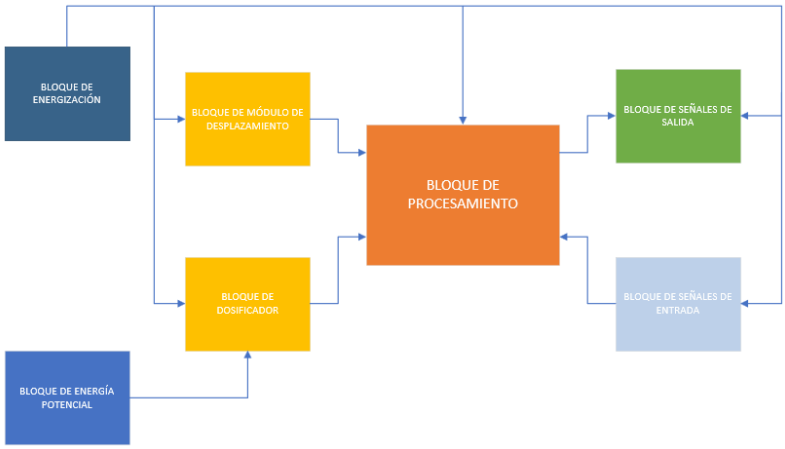


Ilustración 16-3: Arquitectura general del prototipo bloques
Realizado por: Bonilla, H., 2023.

- **Bloque de energización**

Hace referencia a la fuente de poder o alimentación, este es el suministro de energía que requieren los actuadores eléctricos y electrónicos que conforman el prototipo de línea de ensamble de partes plásticas.

- **Bloque de energía potencial**

Se refiere a una fuente de aire comprimido, con la funcionalidad de aumentar la presión de gases o vapores compresibles.

- **Bloque de módulo de desplazamiento**

Este bloque está relacionado con el módulo de desplazamiento siendo una unidad funcional que se integra en el bloque de procesamiento.

- **Bloque de dosificador**

Este bloque corresponde al módulo dosificador el cual es el encargado de proveer las bases y tapas respectivamente para cada unidad de tarjeta de desarrollo Arduino.

- **Bloque de procesamiento**

Se centraliza la gestión del prototipo de línea de ensamble de partes plásticas, la cual lanza las señales de salida a los actuadores eléctricos, las señales de entrada permite iniciar el seteo del robot cartesiano, da lugar al accionamiento del robot cartesiano como al bloque de desplazamiento continuando con la inicialización del bloque de dosificación el cual tiene dos tipos de energía que lo accionan, uno es eléctrico, permitiendo el paso al aire comprimido, transformando la energía potencial del aire comprimido en energía cinética.

- **Bloque de señales de salida**

Procesando la información proporcionada por el bloque de señales de entrada, se generan señales que interactúan con los actuadores eléctricos, esto da lugar al seteo del robot cartesiano, dando así inicio a poner en marcha el proceso de ensamble de las piezas.

- **Bloque de señales de entrada**

Este bloque permite obtener información fundamental sobre el prototipo, ingresa a un sistema, dispositivo o proceso para ser procesada, interpretada siendo de utilidad de alguna manera en una variedad de aplicaciones capturando datos o información.

3.7.1. Criterios de diseño del prototipo

- **Etapa 1: Abastecimiento y ubicación**

Esta etapa marca el comienzo fundamental del proceso, enfocándose en el seteo del robot cartesiano y abastecimiento de las partes plásticas. Se lleva a cabo un análisis detenido de las características individuales de cada pieza, se determina la cantidad necesaria para el ensamblaje y se establecen criterios precisos para la dosificación de las piezas plásticas. Este minucioso análisis es esencial para garantizar un ensamblaje preciso y eficaz en fases posteriores del proyecto, sentando así las bases para un proceso de producción fluido y de alta calidad.

- **Etapa 2: Dosificación y ensamblaje**

En la segunda etapa del proceso, se da el aprovisionamiento del prototipo con las piezas que serán ensambladas, lleva a cabo un paso crucial que implica la disposición precisa y estratégica de las piezas destinadas a ser ensambladas en el bloque de ensamblaje. Esta etapa se centra en la dosificación meticulosa de las piezas, asegurando que estén colocadas de manera óptima para facilitar su unión y ensamblaje posterior. Cada pieza es ubicada considerando su forma, tamaño y características específicas, garantizando así que se inicie proceso de ensamblaje de manera eficaz y acorde con las especificaciones requeridas. Esta etapa sienta las bases para una manufactura eficaz y un producto final de alta calidad

- **Etapa 3: Despachar**

En la etapa final, se tiene previsto la implementación de una rampa de despacho estratégicamente ubicada. Esta adición tiene como principal objetivo optimizar y agilizar significativamente el proceso de distribución de productos y mercancías. La rampa de despacho brindará una infraestructura funcional y eficaz, permitiendo una carga y descarga más expedita, contribuyendo así a mejorar la fluidez operativa y la puntualidad en la entrega de los productos.

3.7.2. Descripción de la operación prevista del prototipo

El esquema operativo previsto para el prototipo de la línea de ensamblaje de partes plásticas se encuentra representado en la Ilustración 17-3. Esta representación detallada ofrece una visión exhaustiva de cada etapa del proceso, permitiendo visualizar con claridad cómo se anticipa que funcione el sistema de ensamblaje. La Ilustración 17-3 actúa como una guía visual para comprender el flujo de operaciones, desde la recepción inicial de las piezas hasta su despacho, brindando una representación gráfica clave para comprender el diseño y operación proyectados

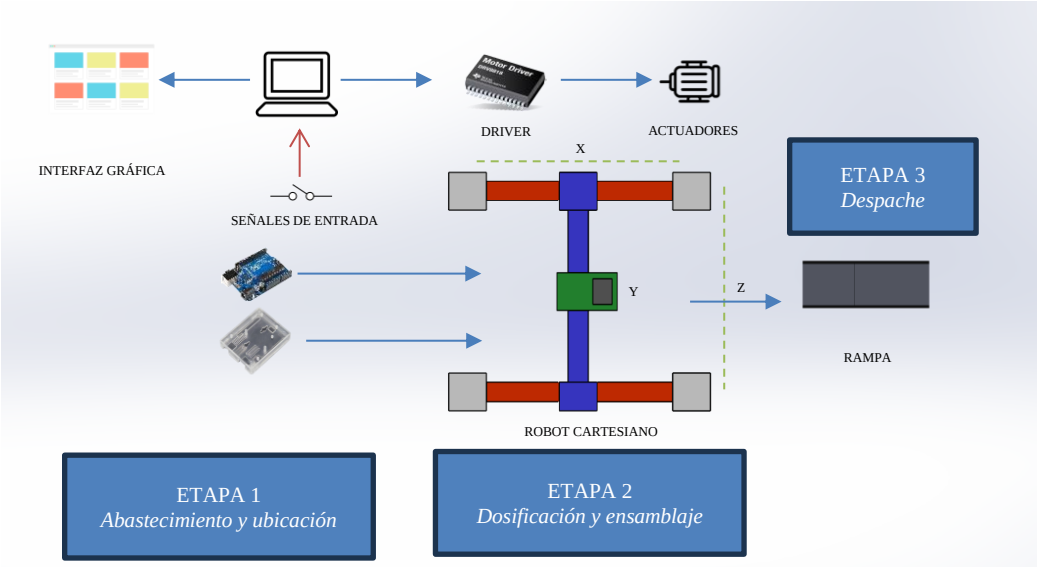


Ilustración 17-3: Arquitectura general del prototipo
Realizado por: Bonilla, H., 2023.

En la primera etapa, se iniciará la implementación de la interfaz gráfica, fundamental para poner en marcha el prototipo con éxito. Esta interfaz será la herramienta clave que permitirá la interacción y control del robot cartesiano. En este punto inicial, se llevará a cabo el seteo del robot en su posición de inicio, conocida como punto cero. Además, se procederá al abastecimiento de las partes plásticas de forma manual para garantizar un comienzo fluido en la operación del sistema. El proceso de activación incluirá la preparación de los módulos de dosificación, los cuales desempeñan un papel crucial en la etapa de ensamblaje. Cada módulo estará debidamente equipado con un sensor especializado diseñado para detectar la presencia de las partes plásticas necesarias en la producción. Además, se integrarán placas de Arduino en cada módulo para facilitar y coordinar el ensamblaje preciso de los componentes.

En la secuencia correspondiente a la segunda etapa, se llevará a cabo la crucial fase de dosificación y detección de las partes plásticas esenciales para el ensamblaje. En esta etapa, se verificará minuciosamente la existencia de una placa Arduino en el dispositivo dosificador, un componente clave en el control preciso del proceso. Una vez confirmada la presencia de la placa Arduino y las partes plásticas, se dará inicio al proceso de ensamblaje de las piezas, garantizando así un flujo operativo eficaz y coordinado dentro del sistema de la línea de ensamblaje.

En la etapa final, marcada como la tercera fase del proceso, se activará una vez concluido exitosamente el ensamblaje de las piezas. Este último tramo operativo será ejecutado de manera automática por el robot cartesiano, una pieza central en la operatividad del prototipo. La tarea encomendada al robot consistirá en liberar con precisión la pieza ensamblada, facilitando su

trayectoria a través de una rampa especialmente diseñada y situada al final de la línea de ensamble. La presencia de esta rampa asegura un proceso eficaz y continuo, permitiendo que el producto ensamblado alcance su destino final de manera óptima

3.8 Diseño del prototipo de línea de ensamble

3.8.1. Estructura del proceso del prototipo

El diseño del prototipo sigue una metodología estructurada que implica la ejecución de diversas etapas. Estas etapas están diseñadas meticulosamente para cumplir con los parámetros de diseño establecidos para el prototipo de línea de ensamble.

- **Análisis bibliográfico**: este proceso implica revisar investigaciones previas, literatura técnica y recursos relacionados con líneas de ensamblaje, robótica cartesiana, sistemas de automatización y tecnologías aplicables. Es fundamental para contextualizar y fundamentar la investigación.

- **Modelado CAD:** se refiere a la creación digital tridimensional de cada componente de la línea de ensamblaje y su interacción. El modelado CAD permite visualizar cómo se ensamblarán las piezas, cómo operará el robot cartesiano y cómo interactuarán todos los elementos de la línea.

- **Selección de hardware:** es el proceso de elegir y especificar los componentes físicos, como el robot cartesiano, sensores, sistemas de control y otros dispositivos, fundamentales para la operación eficaz de la línea de ensamblaje. Esta elección se basa en las necesidades del prototipo y la viabilidad técnica.

- **Conexión a hardware:** implica establecer las conexiones eléctricas y electrónicas adecuadas entre los distintos componentes de hardware, como sensores, actuadores, controladores, etc. Es crucial para garantizar que el sistema funcione de manera coordinada y eficaz

- **Programación:** es el desarrollo del software que controla las operaciones del robot cartesiano y la línea de ensamblaje en su conjunto. Esto implica definir algoritmos y lógica de control que permitan una interacción coordinada y precisa entre los componentes.

- **Construcción del prototipo:** se refiere a la materialización física y puesta en marcha del sistema propuesto. Incluye la fabricación de partes, ensamblaje y configuración precisa de cada componente siguiendo el diseño definido, para crear un prototipo funcional de la línea de ensamblaje.

- **Pruebas de funcionalidad:** estas pruebas están destinadas a validar la eficacia del prototipo en términos de ensamblaje de partes plásticas. Involucran verificar que el robot cartesiano siga las trayectorias adecuadas, que las piezas se ensamblen correctamente y que todo el sistema funcione sin problemas en un entorno controlado.

3.9 Diseño estructural del prototipo

El diseño estructural implica planificar y crear la disposición física y organizativa de una construcción, asegurando su estabilidad y resistencia, incluye la selección de materiales, distribución de elementos y análisis para garantizar que la estructura cumpla con funcionalidades necesarias.

3.9.1. Selección de perfiles para la estructura

La selección del perfil para la estructura del prototipo de línea de ensamble constituye una etapa importante en el desarrollo de este, tomando en cuenta factores fundamentales para su selección, se obtiene información preliminar de las propiedades de varios perfiles en un catálogo proporcionado por CNC PERFILES, una empresa que se encarga de la comercialización de productos, equipos para impresoras 3D, piezas para robots cartesianos y componentes para brazos robóticos. El perfil seleccionado para la estructura es un perfil de aluminio 20 V slot con un dimensionamiento de 20x40 mm para una mejor referencia mírese la Ilustración 18-3, este perfil este fabricado con aluminio 6063 T5, con un peso de 0.95 kg/m, superficie anodizada de color negro.

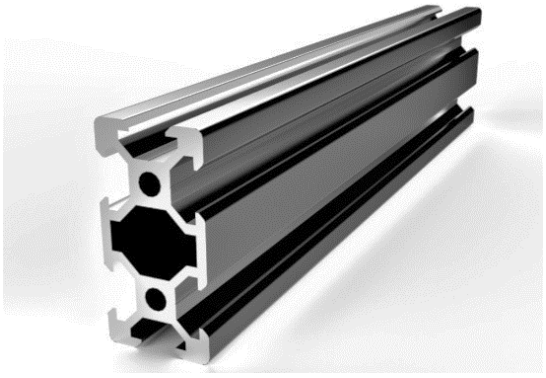


Ilustración 18-3: Perfil de aluminio 20 V slot
Realizado por: Bonilla, H., 2023.

3.9.2. Diseño de la estructura del prototipo

La estructura del prototipo de la línea de ensamblaje, esta meticulosamente diseñada utilizando el software SolidWorks, presenta dimensiones de 510x300x530 mm. Este diseño minucioso busca lograr la eficiencia óptima en la disposición y ensamblaje de las piezas plásticas. Para asegurar una movilidad óptima del robot cartesiano, se han implementado carros de desplazamiento en

cada eje con tensores de poleas conectador a los motores, garantizando una operación fluida y precisa en la ejecución de las tareas encomendadas.

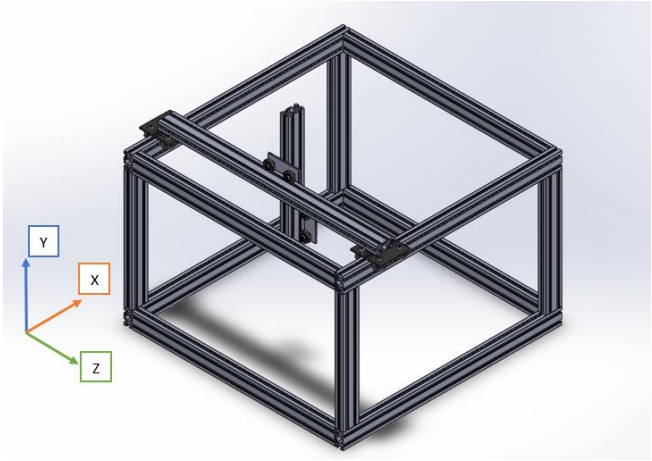
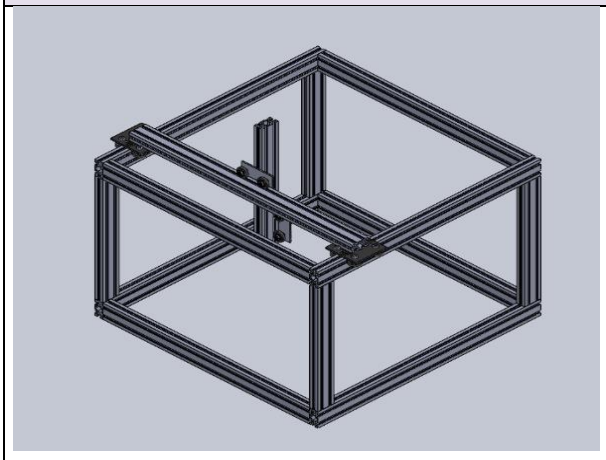


Ilustración 19-3: Estructura del robot cartesiano
Realizado por: Bonilla, H., 2023.

3.10 Análisis estático

El análisis estático realizado en la estructura del robot cartesiano tiene como finalidad el detectar errores, vulnerabilidades entre otros problemas potenciales para el prototipo de línea de ensamble de partes plásticas, la Tabla 6-3 muestra el modelado de la estructura del robot cartesiano tratado como solido con propiedades únicas del aluminio 6063 T5.

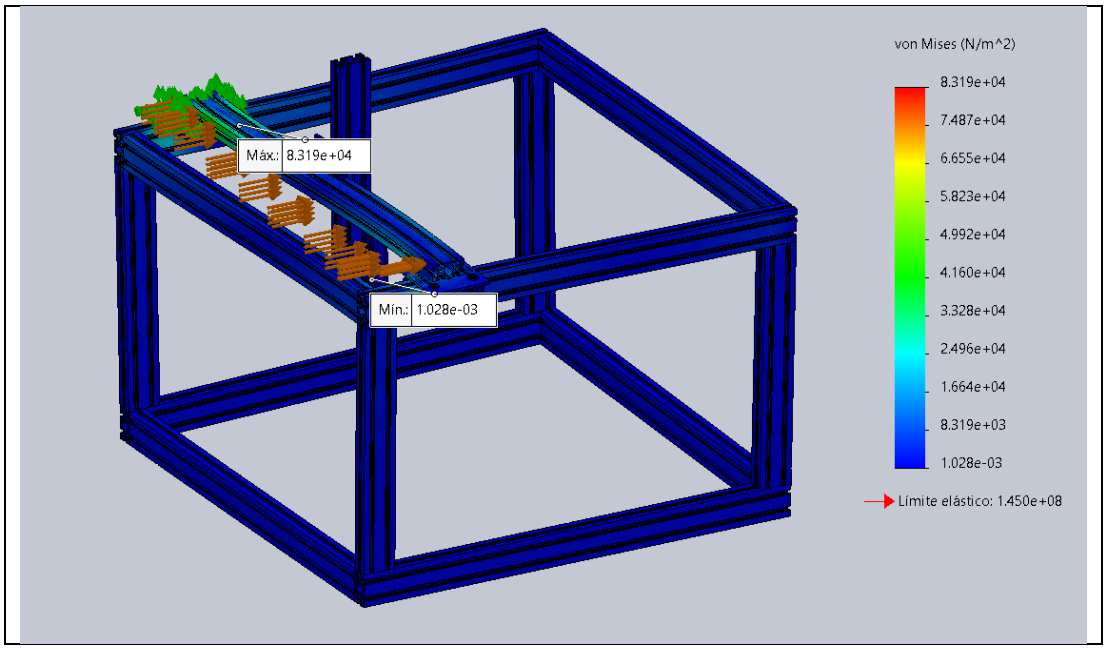
Tabla 6-3: Propiedades de la estructura del robot cartesiano

Referencia de modelo	**Tratado como**	**Propiedades volumétricas**
	Sólido	Masa: 2.29215 kg Volumen: 2292148.29 mm^3 Área de superficie = 2201701.93 mm^3 Peso:22.48 N
	Propiedades	**Componentes**
	Nombre	Estructura (Impacto alto)
	Módulo elástico	6.90E+10N/m^2
	Coef. de Poisson	0.33 N/D
	Módulo cortante	2.58E+10N/m^2
	Densidad de la masa	2700 kg/m^3
	Límite de tracción	185000000 N/m^2
	Límite elástico	145000000 N/m^2
	Coef. de exp. ter.	2.34E-05 /K
	Conductividad tér.	209 W/(m·K)
	Calor específico	900 J/(kg·K)

Fuente: SolidWorks, 2022
Realizado por: Bonilla, H., 2023.

Tabla 7-3: Resultado del estudio de tensiones – estructura del robot cartesiano

Nombre	**Tipo**	**Mínimo**	**Máximo**
Tensiones	VON: Tensión de von Mises	1.028e-03 N/m^2	8.319e+04 N/m^2

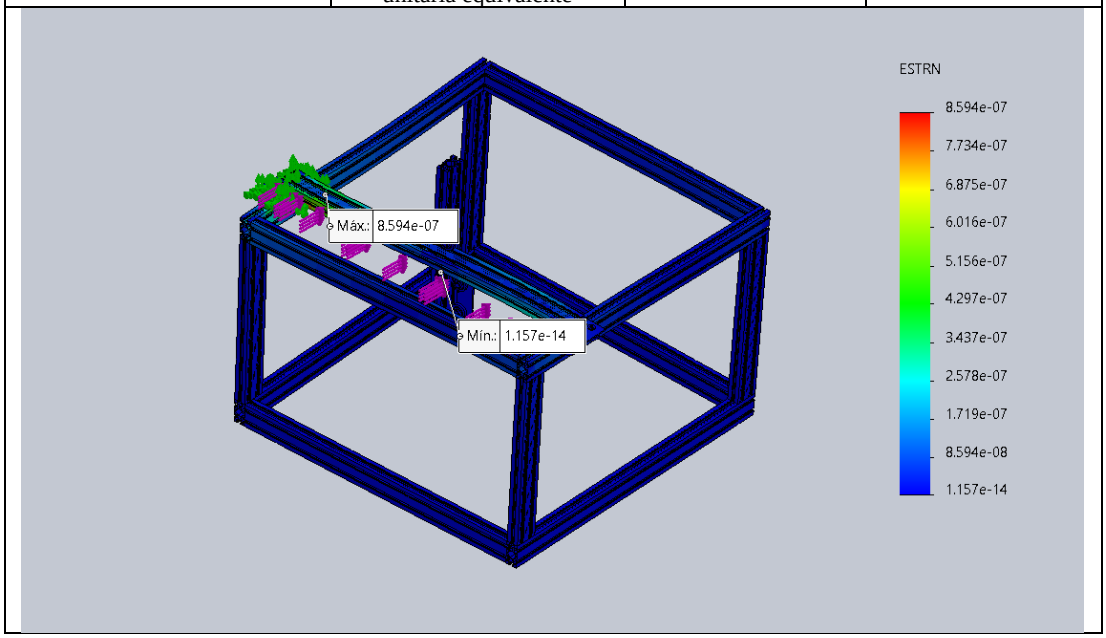


Fuente: SolidWorks, 2022
Realizado por: Bonilla, H., 2023.

En base a lo obtenido de la Tabla 7-3 se determinó que el límite elástico del aluminio es 1.45e+08 N/m^2, la estructura está sometida a una carga máxima de 1 N dando como resultado la tensión máxima de 8.319e+04 N/m^2 y la mínima de 1.028e-03 N/m^2, estas tensiones se encuentran por debajo del límite elástico de la estructura de aluminio por lo cual no existirá inconveniente en usar la estructura para el prototipo de línea de ensamble.

Tabla 8-3: Resultado del estudio de deformaciones unitarias – estructura del robot cartesiano

Nombre	Tipo	Mínimo	Máximo
Deformaciones unitarias	ESTRN: Deformación unitaria equivalente	1.157e-14 N/m^2	8.594e-07 N/m^2



Fuente: SolidWorks, 2022
Realizado por: Bonilla, H., 2023.

La Tabla 8-3 determina la deformación que se daría en caso de mantener un extremo del perfil anclado a la estructura del prototipo, dando a notar al aplicar la fuerza máxima de 1 N este tendrá a desplazarse sin problemas.

3.11 Diseño del producto

Para el desarrollo del prototipo de línea de ensamblaje, se diseñó una caja que consta de una base y una tapa. La base alberga la placa Arduino y otros componentes electrónicos, incorporando ranuras y aberturas estratégicas para asegurar y permitir conexiones de cables y acceso a puertos. La tapa se diseñó para encajar perfectamente, permitiendo fácil acceso y proporcionando áreas para visualizar indicadores LED.

3.11.1. Modelado de caja para Arduino Uno

Para el modelado de caja para Arduino Uno se realizan dos piezas, la primera será la base donde encajará de manera precisa permitiendo el correcto acoplamiento de la pieza con el Arduino, el Ilustración 20-3 representa el modelado de la base para Arduino, en su interior cuenta con 4 columnas para posicionar correctamente la placa.

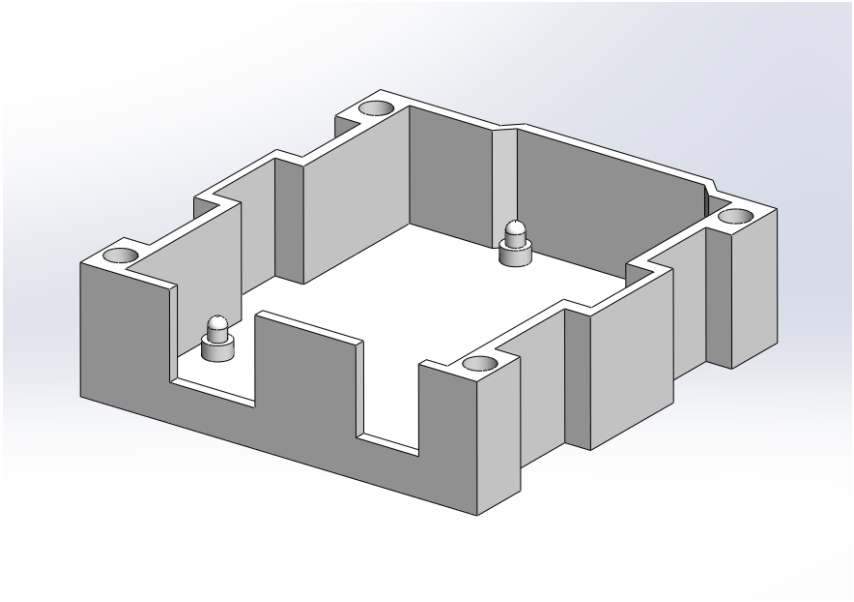


Ilustración 20-3: Base de caja para Arduino Uno
Realizado por: Bonilla, H., 2023.

La Ilustración 21-3 exhibe la vista frontal, superior y lateral de la base diseñada para albergar la placa Arduino, detallando sus dimensiones cruciales para la construcción. La base presenta paredes con un espesor uniforme de 1.89 mm y cuenta con cuatro orificios en las esquinas, estratégicamente ubicados para permitir un ensamblaje seguro con la tapa mediante presión.

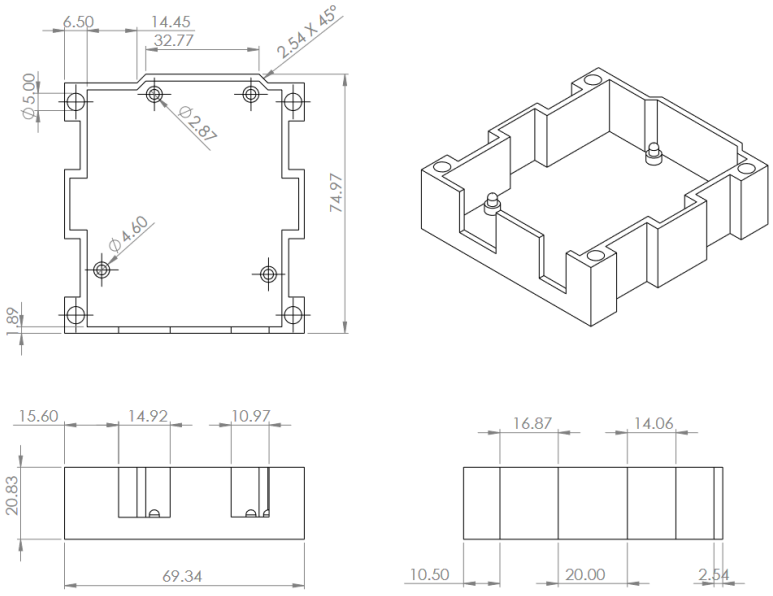


Ilustración 21-3: Dimensiones de la base para Arduino Uno
Realizado por: Bonilla, H., 2023.

La Ilustración 22-3 representa la tapa de la caja para Arduino Uno modelada en 3D en el software SolidWorks, con características específicas para la correcta conexión de la tarjeta, la tapa fue diseñada para encajar de manera precisa, permitiendo un acceso sencillo y proporcionando áreas que permiten la visualización de los indicadores LED, su diseño está pensado

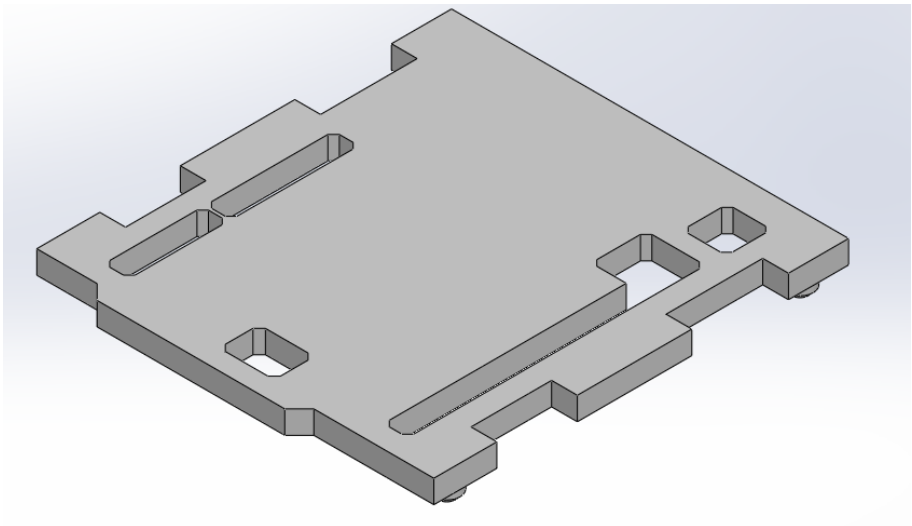


Ilustración 22-3: Tapa de caja para Arduino Uno
Realizado por: Bonilla, H., 2023.

La Ilustración 23-3 muestra la vista frontal, superior y lateral de la tapa para la placa Arduino con sus respectivas dimensiones a fin de construirla, el modelado de la tapa es una tarea fundamental, es necesario tener un conocimiento exhaustivo de las herramientas y técnicas disponibles en el software de modelado 3D SolidWorks.

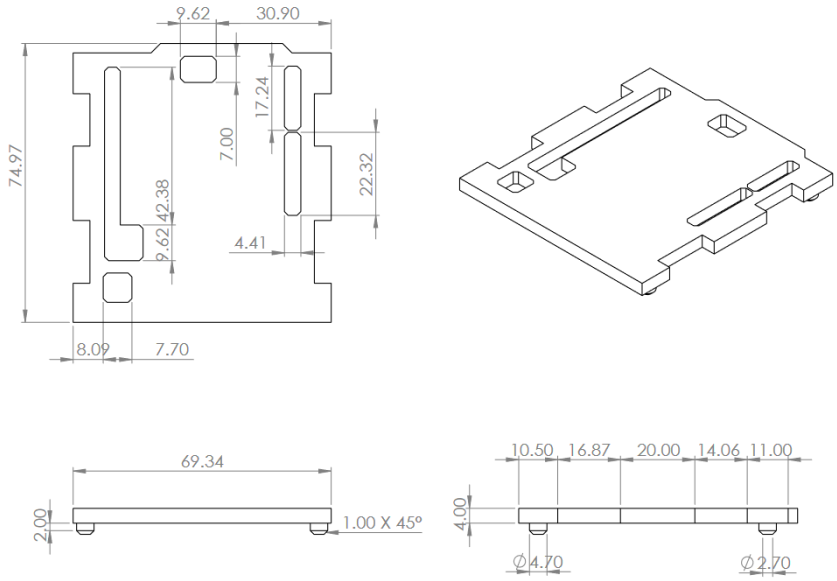


Ilustración 23-3: Dimensiones de la tapa para Arduino Uno
Realizado por: Bonilla, H., 2023.

La Ilustración 24-3 muestra una caja ensamblada diseñada específicamente para alojar una placa Arduino Uno. Se observa que la tapa encaja de manera precisa sobre la base, asegurando un cierre adecuado y protegiendo los componentes internos. La disposición interna permite un acomodo eficaz de la placa y otros elementos electrónicos, garantizando una organización limpia y facilitando el acceso a puertos y conexiones.

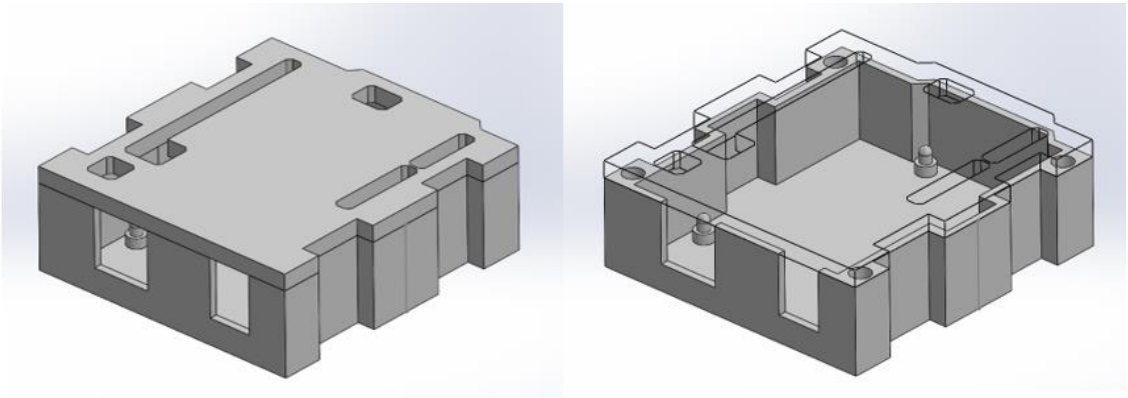


Ilustración 24-3: Caja de Arduino Uno ensamblada
Realizado por: Bonilla, H., 2023.

3.11.2. Selección de material para la impresión del producto

La selección de material para imprimir las piezas necesarias se lo hace en función de la durabilidad para alargar la vida útil del prototipo de línea de ensamble, temperatura máxima de servicio en grados Celsius con el fin de evitar posibles torceduras, la densidad y el precio de cada uno de los materiales, estos datos se los tomo para cuatro materiales de impresión, estos son ABS, PLA, HIPS, PETG, estos materiales se encuentran detallados en la Tabla 9-3.

Tabla 9-3: Características de materiales para imprimir

	ABS	PLA	HIPS	PETG
Durabilidad	8.0/10	9.0/10	4.0/10	8.0/10
Máxima Tem. Serv. (°C)	98	52	100	73
Densidad (g/cm^3)	1.04	1.24	1.03-1.04	1.23
Precios por Kg	10$ - 40$	10$ - 40$	24$ - 32$	20$ - 60$

Fuente: Liqcreate, 2022
Realizado por: Bonilla, H., 2023.

Mediante la Tabla 9-3 de características de materiales para imprimir destaca el PLA, este material tiene una alta durabilidad y una densidad mucho más elevada a comparación de los demás materiales, la temperatura máxima de servicio del PLA es la menor por lo cual se podría descartar, pero el prototipo de línea de ensamble no estará en contacto con temperaturas diferentes a las de ambiente, el precio se encuentra muy semejante entre los diferentes materiales para impresión 3d.

3.12 Diseño de módulos

- **Diseño del módulo dosificador de bases y tapas**

El modelado de un dispensador para las bases y tapas es un proceso esencial en el diseño del prototipo de línea de ensamble, ya que optimiza la eficacia y precisión en la distribución de estos componentes clave, el dispensador se configura cuidadosamente para garantizar un suministro adecuado y continuo de las bases y tapas durante el proceso de ensamblaje. por lo general los dispensadores estan diseñados a las necesidades requeridas, el módulo dosificador se plantea el uso de un pinston neumatico en conjunto con una electrovalvula para controlar el dispositivo hidraulico, en el Ilustración 25-3 representa el dispensador de tapas y en el Ilustración 26-3 se representa el modelado del dispensador de bases, cada uno de estos dispensadores estan diseñados para contener un maximo de 5 unidades por dispensador.

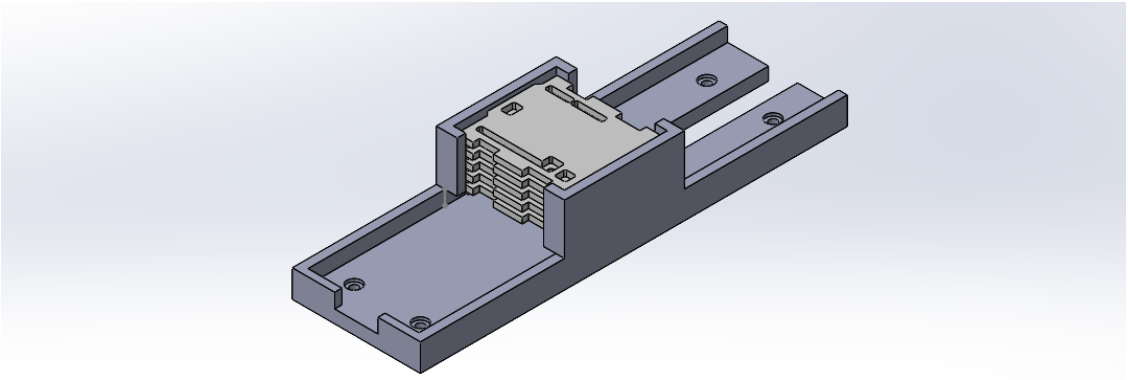


Ilustración 25-3: Dispensador de tapas
Realizado por: Bonilla, H., 2023.

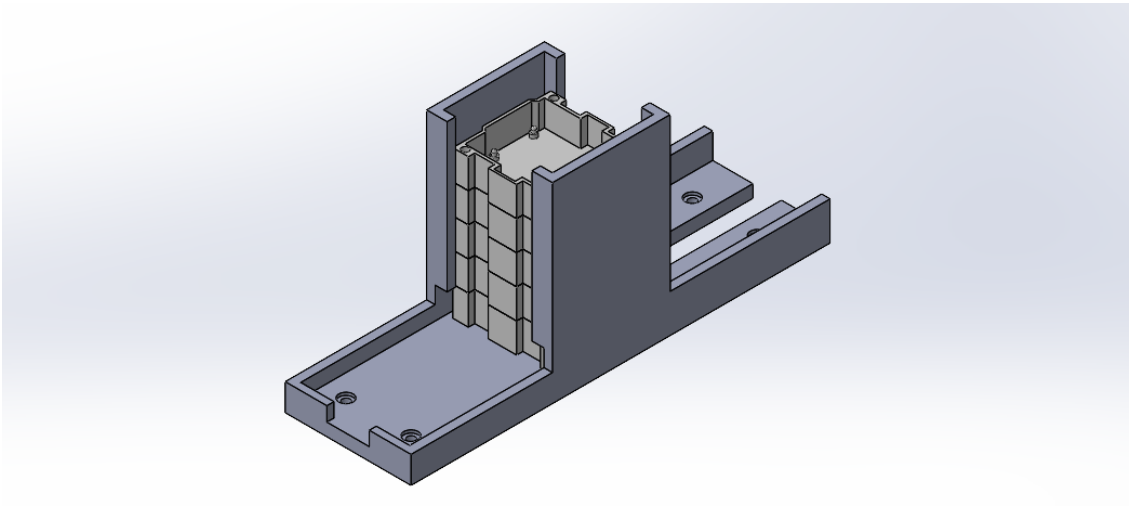


Ilustración 26-3: Dispensador de bases
Realizado por: Bonilla, H., 2023

- **Diseño del módulo porta placas de Arduino**

Para el modelado del módulo porta placas de Arduino es importante tomar en cuenta las dimensiones exactas de Arduino y los componentes que sobre salen de la misma, la Ilustración 27-3 representa el diseño 3d de la misma que alberga un máximo de 5 placas Arduino Uno, esta se sitúa sobre una base diseñada específicamente para el módulo porta placas con un espacio sobrante para colocar el relé y la electroválvula que controlara el pistón para el correcto funcionamiento del dispensador de bases y tapas.

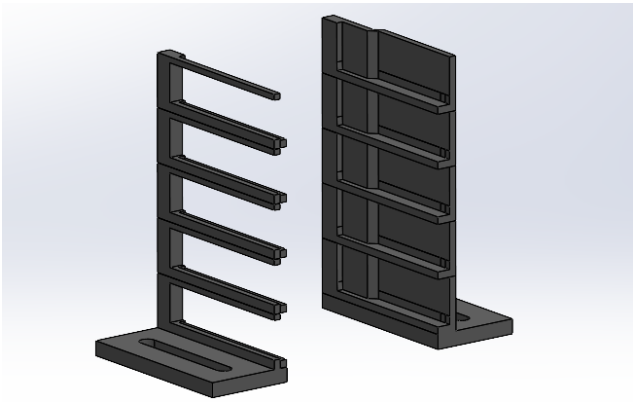


Ilustración 27-3: Base porta Arduino
Realizado por: Bonilla, H., 2023

- **Modelado estructural del módulo de desplazamiento**

En este apartado se detalla los criterios tomados para el diseño del módulo de desplazamiento, mediante el uso del software SolidWorks, por su versatilidad y gran variedad de recursos, permite la creación del modelo sólido en 3D como se muestra en la Ilustración 28-3.

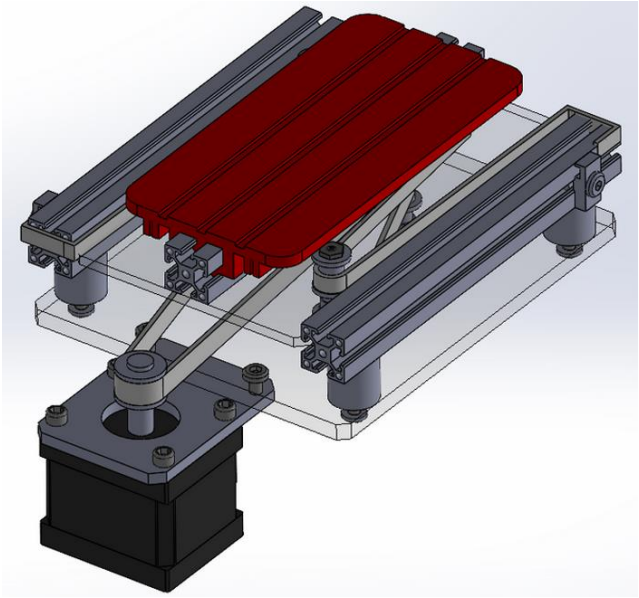


Ilustración 28-3: Modelado del módulo de desplazamiento en 3D
Realizado por: Bonilla, H., 2023

Este módulo necesita 2 placas para su desplazamiento, la placa base está sujeta a un perfil de aluminio 20x40 mm, a un par de poleas simples y a un motor nema 17, la placa inferior de desplazamiento encaja entre las ranuras de los perfiles para el correcto funcionamiento del módulo, en la parte superior de esta placa se encuentra un par dos poleas conectando dos relaciones de transmisión a la placa superior de desplazamiento, en la parte inferior de esta se encuentra un juego simple de poleas conectado al motor el cual dará el torque necesario para su correcto funcionamiento como se muestra en la Ilustración 29-3, generando el movimiento del conjunto a ensamblar.

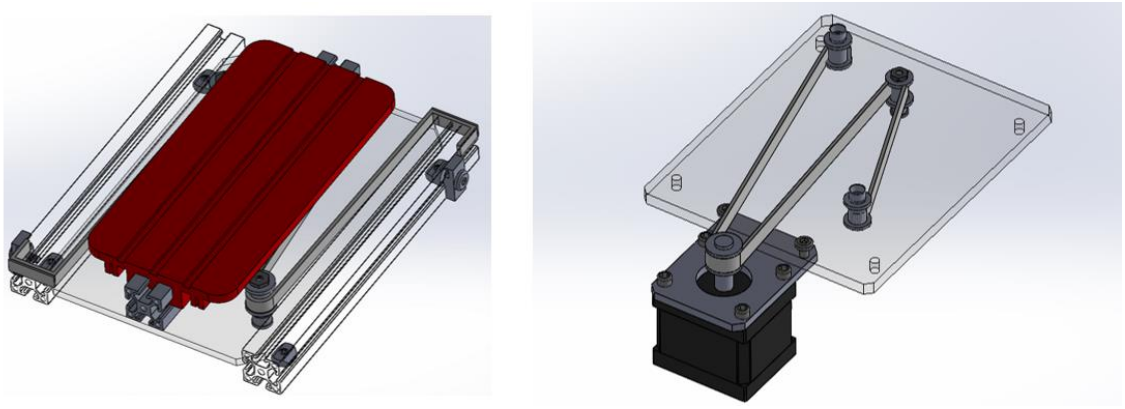


Ilustración 29-3: Módulo de desplazamiento dividido
Realizado por: Bonilla, H., 2023

En la Ilustración 30-3 se observa las dimensiones requeridas para el desplazamiento de la placa superior del módulo de desplazamiento, en la parte central de esta se tiene una extrusión de corte con las medidas de un perfil de aluminio 20x20 mm para encajar de una manera precisa.

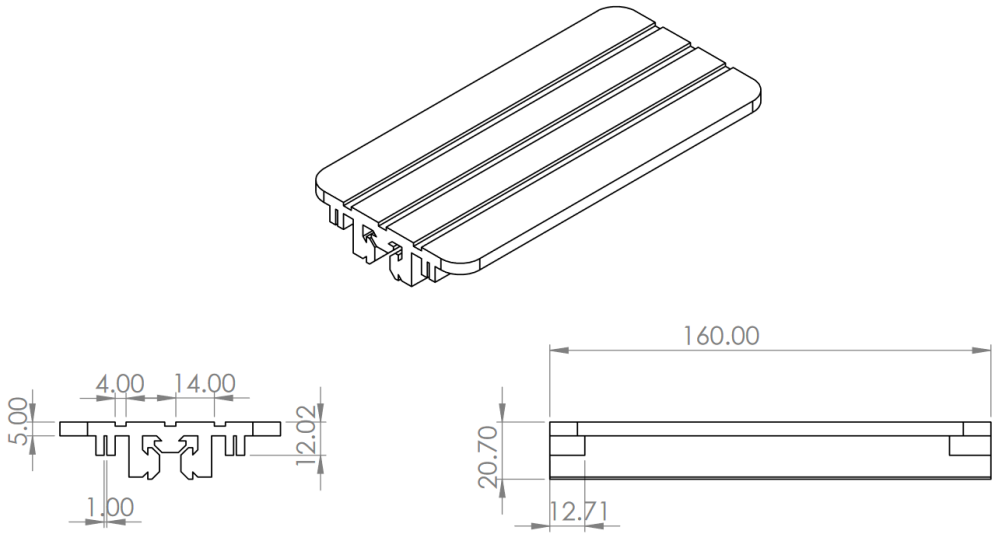


Ilustración 30-3: Placa de desplazamiento superior
Realizado por: Bonilla, H., 2023

La Ilustración 31-3, representa la placa inferior con dimensiones específicas para su funcionamiento, está localizado entre la placa superior y la placa base del módulo de desplazamiento, los 3 agujeros centrales estarán sujetos a un perfil de aluminio 20x20 mm y en los dos agujeros sobrantes se encuentran situado dos poleas simples con un par de rodamientos en su interior para su fácil rotación.

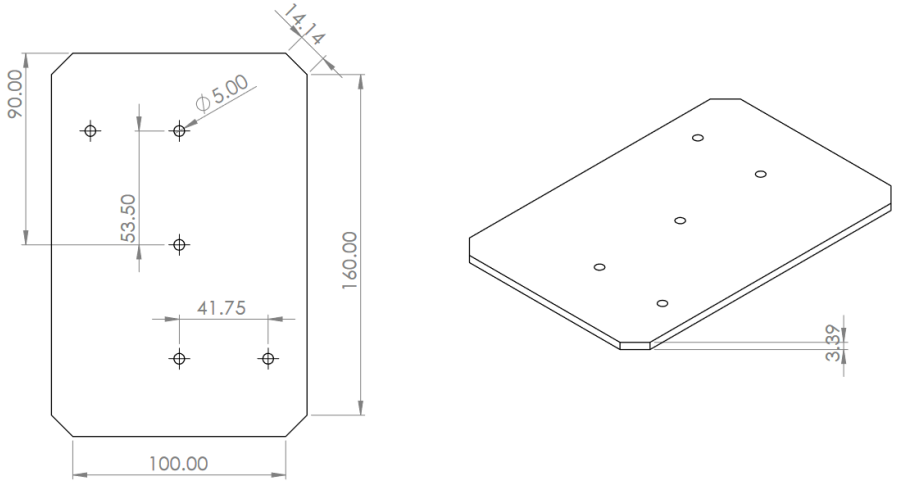


Ilustración 31-3: Placa de desplazamiento inferior
Realizado por: Bonilla, H., 2023

La Ilustración 32-3 representa la base del módulo de desplazamiento, en la que se muestra la cantidad y espacio entre agujeros, estos serán necesarios para el correcto acoplamiento de la base con los perfiles de aluminio, motor nema 17 y poleas para su correcto funcionamiento.

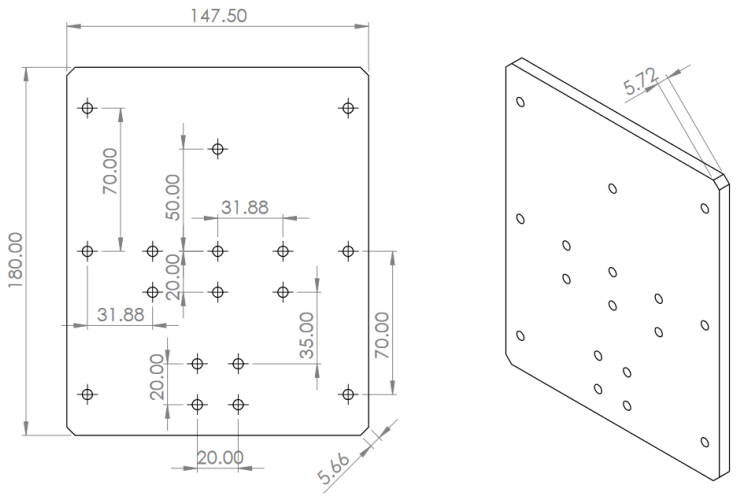


Ilustración 32-3: Placa base del módulo de desplazamiento
Realizado por: Bonilla, H., 2023

- ***Diseño del gripper***

Para diseñar el gripper se establecen los requisitos y especificaciones, definiendo aspectos cruciales como las dimensiones y precisión requerida. Se crean las piezas individuales del gripper, incluyendo las pinzas y el cuerpo, empleando herramientas como extrusión, revolución y corte para lograr las formas necesarias. Luego, se incorporan las correas dentadas diseñando las ranuras o superficies adecuadas en las pinzas, garantizando un ajuste óptimo y preciso. La Ilustración 33-3, se representan las distintas piezas que componen el gripper. Este consta de un par de pinzas, un cuerpo inferior destinado a alojar un motor, y la parte superior del cuerpo, que posibilita la integración del gripper con el robot cartesiano.

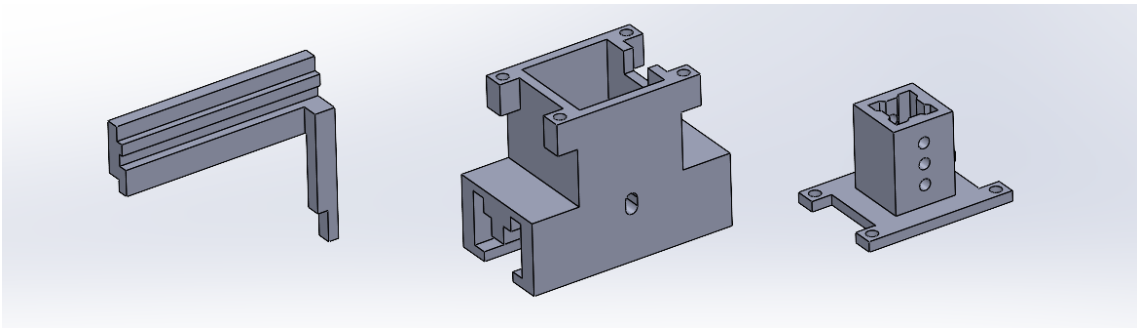


Ilustración 33-3: Partes del gripper
Realizado por: Bonilla, H., 2023

Es esencial utilizar correas dentadas en las pinzas del gripper, para obtener una mayor precisión y un menor descaste con el uso. A continuación, se procede a ensamblar las piezas en un ensamblaje de SolidWorks, asegurándose de colocar las correas dentadas en sus ubicaciones correctas dentro de las ranuras diseñadas previamente Ilustración 34-3.

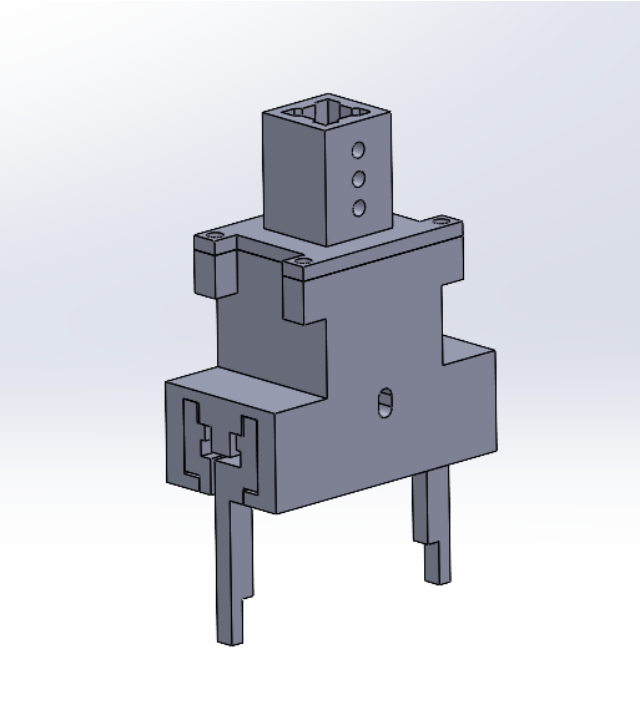


Ilustración 34-3: Gripper ensamblado
Realizado por: Bonilla, H., 2023

3.13 Asignación de sensores

Para el correcto funcionamiento del prototipo de linera de ensamble fue necesario la asignación de 8 sensores, estos estas distribuidos entre el robot cartesiano, dosificador de bases y tapas, módulo de desplazamiento y gripper.

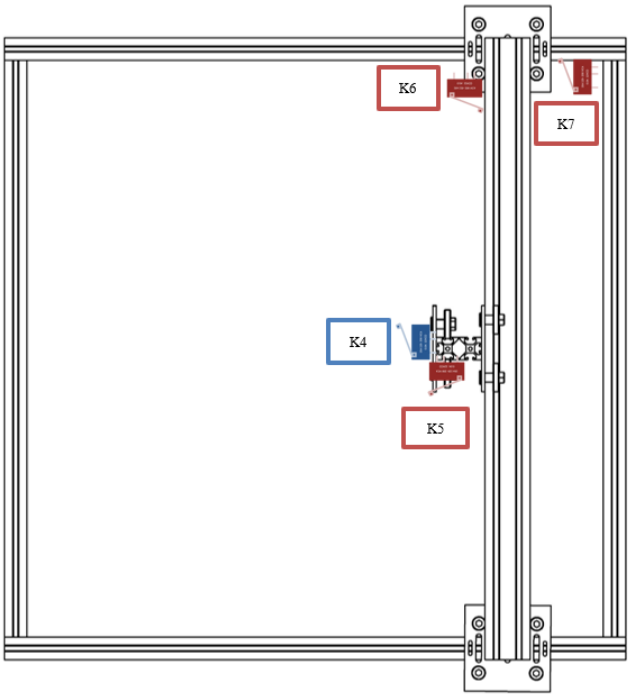


Ilustración 35-3: Asignación de sensores en el robot cartesiano
Realizado por: Bonilla, H., 2023

La Ilustración 35-3 muestra la disposición de los sensores para cada eje del robot cartesiano de un color rojizo y el gripper de un color turquesa, mientras que en el Ilustración 36-3 se muestran los sensores de color verde para el dosificador de bases-tapas, de un color azul para el módulo de desplazamiento respectivamente así pues permitiendo tener una visión más clara desde la vista superior del prototipo de la disposición de los sensores.

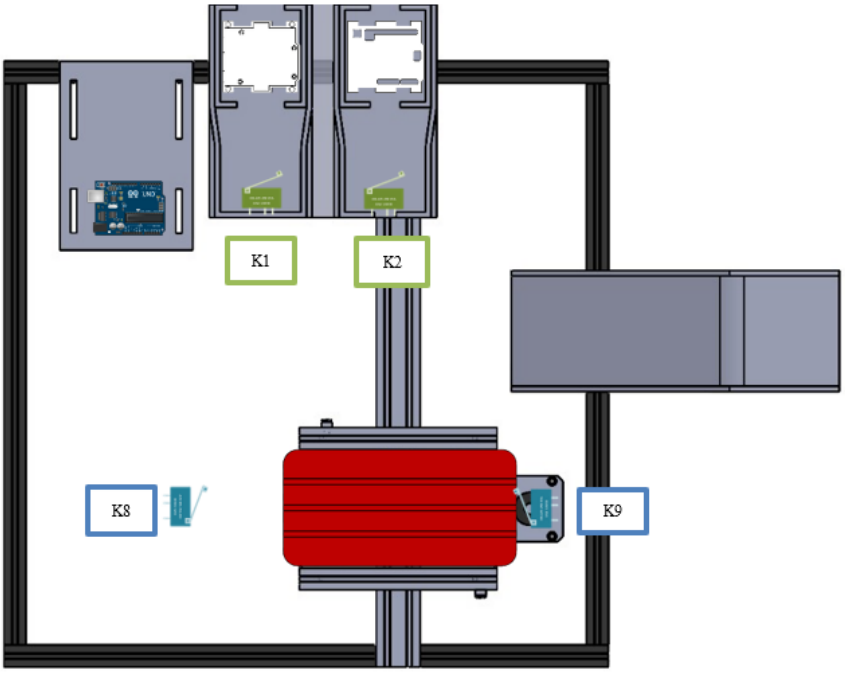


Ilustración 36-3: Asignación del resto de sensores
Realizado por: Bonilla, H., 2023

3.14 Etapa final del diseño

El diseño final del prototipo se puede observar en la Ilustración 37-3, se ha integrado un sistema modular que abarca varios componentes esenciales para el correcto funcionamiento y desempeño de la máquina. Este diseño incluye un módulo de desplazamiento, que consiste en un sistema que permite el movimiento preciso y controlado del producto ha ensamblar mediante un par de juego de poleas. Además, se ha incorporado un módulo de dosificación, diseñado para dispensar con precisión la cantidad adecuada de componentes requeridas en el proceso (base y tapa). Este módulo asegura una dosificación precisa y controlada, optimizando la eficacia y precisión en la producción, asimismo, se ha implementado un gripper, que es un dispositivo de sujeción y manipulación esencial en el ensamblaje de piezas y componentes. Este gripper es responsable de agarrar y mantener de manera segura las partes durante el proceso de ensamblaje, la incorporación de motores nema permiten determinar la posición del robot cartesiano manteniendo una idea clara para la implementación del prototipo, agregado a todo lo mencionado se incluye el módulo porta placas de Arduino, cada una de las instancias abarcaran un máximo de 5 componentes.

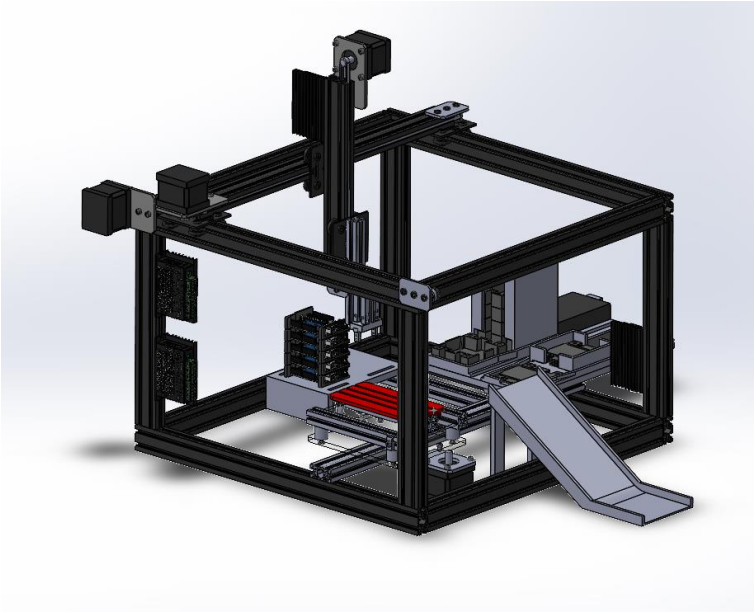


Ilustración 37-3: Diseño final del prototipo de línea de ensamble
Realizado por: Bonilla, H., 2023

3.14.1. Asignación de motores

La disposición de los motores paso a paso se lo realiza de manera estratégica para poder obtener un desenvolvimiento adecuado del prototipo de línea de ensamble de partes plásticas, se puede observar de una manera más detallada los motores en la Ilustración 38-3, los motores 1, 2 y 3 son motores paso a paso Nema 23 mientras que el motor 4 y 5 son motores Nema 17, el motor número uno se encarga del movimiento del eje Y, el motor dos del eje Z, el motor tres del eje X, agregado a esto el módulo de desplazamiento simple se traslada solo en el eje X y las mordazas del gripper se desplazan en el eje Z.

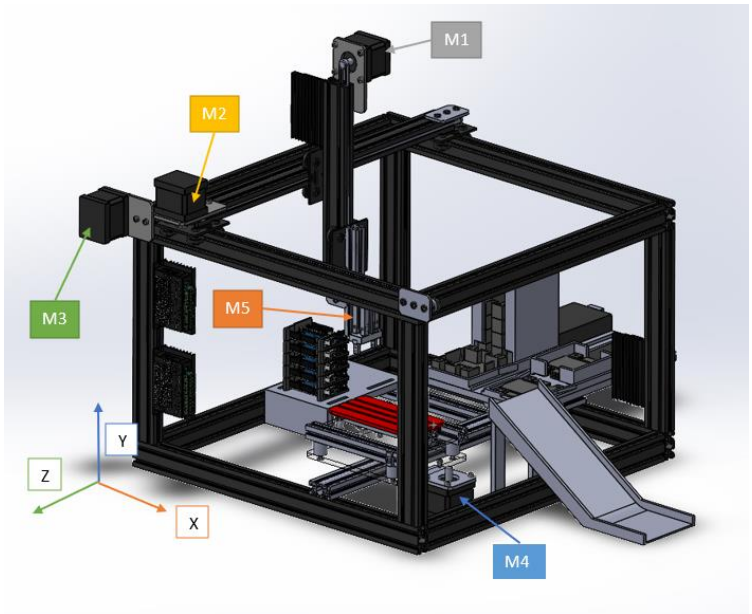


Ilustración 38-3: Disposición de los motores en la estructura
Realizado por: Bonilla, H., 2023

3.15 Hardware

Una vez definido los requerimientos para el prototipo de línea de ensamble, establecido la arquitectura y el diseño estructural que valida su funcionamiento, se da paso a la determinación del hardware para obtener un prototipo eficaz.

3.15.1. Selección para motor paso a paso

La selección apropiada de un motor es un paso crucial en el diseño de un sistema de control de movimiento. Primero, se debe comprender completamente los requisitos específicos del proyecto, tales como la carga a mover, la velocidad necesaria y el torque requerido. A continuación, en la Tabla 10-3 se analiza detalladamente las especificaciones de varios motores nema disponibles en el mercado.

Tabla 10-3: Características del motor paso a paso Nema 23

NEMA			
Motor	**Diámetro**	**Torque (Nm)**	**Revoluciones por minuto**
NEMA 8	20	0.01 - 0.04	0 - 1000
NEMA 11	28	0.06 - 0.12	0 - 1000
NEMA 14	35	0.05 - 0.5	0 - 1000
NEMA 16	39	0.1 - 0.25	0 - 1000
NEMA 17	43	0.2 - 1	0 - 1000
NEMA 23	57	0.5 - 3	0 - 1000
NEMA 24	60	1.2 - 4.6	0 - 1000
NEMA 34	86	3.0 - 12.0	0 - 1000
NEMA 42	102	12.0 - 20.0	0 - 1000

Fuente: Zikodrive, 2018
Realizado por: Bonilla, H., 2023.

Centrándose en aspectos clave como es el torque, permite elegir un motor que se ajuste óptimamente a las necesidades del proyecto. Es vital considerar la compatibilidad con el controlador correspondiente y la eficacia energética del motor para garantizar un rendimiento óptimo. Para determinar el motor a utilizar se toma en cuenta el peso del robot cartesiano que será de 4.0475 kg y el diámetro de la polea de 20 mm que estará conectada al motor, utilizando un factor de seguridad de 2.

$$\boldsymbol{F = m * g} \quad (1)$$

$$F = 4.0475\ kg\ * 9.91 \frac{m}{s}$$

$$F = 39.706\ N$$

$$\boldsymbol{T = F * D} \quad (2)$$

$$T = 39.706\ N * 0.01\ m$$

$$T = 0.4\ N\ m$$

$$\boldsymbol{T = F * Fc} \qquad (3)$$

$$T = 0.4\ NM * 2$$

$$T = 0.8\ N\ m$$

Mediante la aplicación de fórmulas específicas y cálculos detallados, se logró determinar que el torque requerido para la aplicación es de 0.8 Nm. Este valor es esencial para garantizar un funcionamiento óptimo del sistema, asegurando que la carga pueda ser manipulada de manera eficaz. Además, se ha aplicado un factor de seguridad 2 para mitigar posibles variaciones en la carga o condiciones inesperadas, garantizando una operación segura y confiable.

3.15.2. Motor paso a paso Nema 23

Tomando en cuenta que el torque necesario es de 0.8 Nm se determina según la Tabla 10-3 que el motor paso a paso Nema 23 es aquel que cumple con la exigencia del robot cartesiano independientemente para cada eje, este motor es uno de corriente continua en el cual su rotación se divide en un determinado número de pasos, comúnmente estos motores paso a paso se dividen en 200 los cuales individualmente tienen un ángulo de 1.8˚permitiendo tener una precisión exacta.

En la preselección del motor paso a paso se optó por un Nema 23 57BYGH76-3004, sus características se encuentran descritas en la

Tabla 11-3, este actuador eléctrico satisface las necesidades independientes de cada eje para el robot cartesiano la Ilustración 39-3 permite una comprensión visual del actuador a utilizar.

Ilustración 39-3: Motor paso a paso Nema 23

Realizado por: Bonilla, H., 2023.

Tabla 11-3: Características Motor Nema 23

MOTOR NEMA 23	
Características	**Rango**
Longitud:	76 mm
Corriente:	2.8A
Par:	1.89N / M 4 cables
Diámetro del eje:	6.35mm, 8 mm
Voltaje:	3V
Resistencia:	1,13 ohmios
Inductancia:	3,6 mH
Momentos estáticos:	18,9 kg / cm
Salida:	4
Momento de inercia:	480 g /cm²
Peso del motor:	1 kg
Par de retención:	0,68 KG / cm

Fuente: I2C Electronics, 2021
Realizado por: Bonilla, H., 2023.

3.15.3. Motor paso a paso Nema 17

Para el correcto funcionamiento del prototipo de línea de ensamble es fundamental tomar en cuenta los requerimientos básicos del módulo simple de desplazamiento y el gripper, por ende, se considera los motores paso a paso como una opción optima por su precisión y su fácil control de la posición del motor, su velocidad y rotación. El motor paso a paso a utilizar para el módulo de desplazamiento es un Nema 17 HS3001-B como se observa en laIlustración 40 -3, debido a que brinda características acordes a las necesidades del prototipo como se puede observar en la Tabla 12, tanto para el módulo de desplazamiento como para el gripper.

Ilustración 40-3: Motor paso a paso Nema 17
Realizado por: Bonilla, H., 2023

Tabla 12-3: Características motor Nema 17 HS3001-20B

Tabla de motor nema 17 HS3001- 20B

Angulo de giro:	1.8º
Longitud del motor:	42mm
Corriente de trabajo:	1.2A
Resistencia de la fase:	1.7Ohms
Inductancia de la fase:	4.5mH
Inercia del motor:	68 g*cm2
Peso del motor:	240g
Torque:	0.4 N.m

Fuente: Hetpro, 2022
Realizado por: Bonilla, H., 2023

3.15.4. Driver DM542

Para poder controlar los motores paso a paso Nema 23 es necesario de un Driver como complemento la Ilustración 41-3 muestra el microcontrolador, este componente digital tiene funciones de ajuste automático permitiendo controlar el amperaje y el número de pasos por revolución, mediante un interruptor de 6 pines se realiza la configuración adecuada para obtener 800 pasos a un amperio, se mantienen en ON el pin número 1, 4, 6 mientras que el pin 2, 3, 5 permanecen en OFF, para saber más acerca de sus características se incluye la Tabla 13-3.

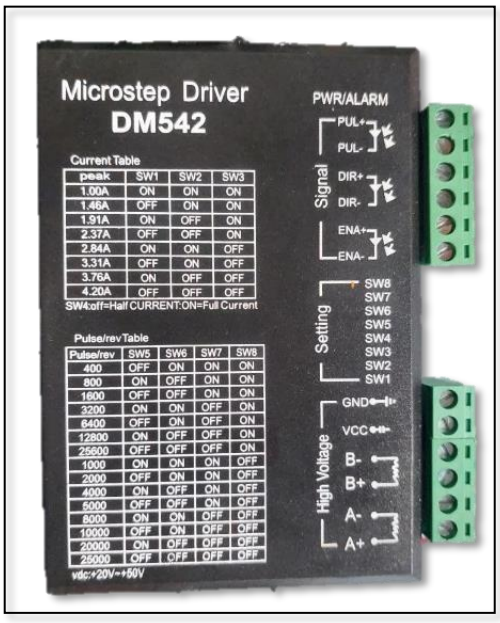


Ilustración 41-3: Driver DM542
Realizado por: Bonilla, H., 2023.

Tabla 13-3: Características del Driver DM542

DRIVER DM542	
Voltaje de operación:	20V – 50V DC
Corriente pico de salida:	1 ~ 4.2A
Corriente de señal lógica:	7 ~ 10 mA (típico 10 mA)

Frecuencia de entrada de pulso:	0~200kHz
Resistencia de aislamiento:	500MΩ
Vibración:	10-55Hz/0.15mm
Dimensiones:	118 x 76 x 34 mm

Fuente: Unit Electronics, 2022
Realizado por: Bonilla, H., 2023.

3.15.5. Driver TB6600

El Driver TB6600 fue seleccionado para controlar el motor Nema 17 debido a su capacidad para manejar corrientes y voltajes mírese la Ilustración 42-3, permitir microstepping para precisión de movimiento, ser fácil de configurar y compatible con microcontroladores como Arduino. Además, ofrece protecciones integradas y resulta coste-eficiente, convirtiéndolo en una opción óptima para garantizar un rendimiento eficaz y seguro en la línea de ensamble del prototipo.

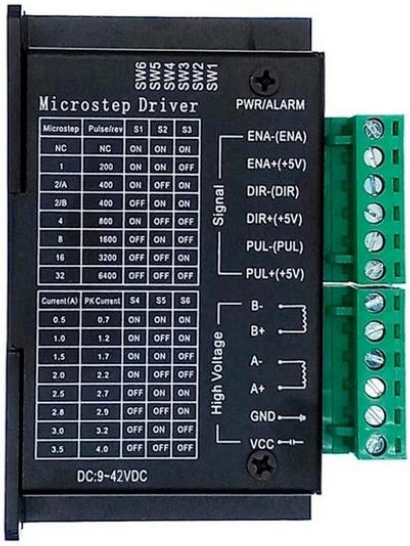


Ilustración 42-3: Driver TB6600
Realizado por: Bonilla, H., 2023.

3.15.6. Fuente de poder SPD3303X-E

Para el correcto funcionamiento del prototipo de linea de ensamble se necesita la alimenrtacion adecuada para propiciar el manejo del robot cartesiano, principalmente se necesita de una fuente regulable con 3 salidas para alimentar los controladores de los motores, el driver DM542 y TB6600 se alimentan con 24 voltios y 3 amperios.

La alimentacion sera la misma para los componentes del prototipo de linea de ensamble pero por cuestiones de conexcion de hardware se tendra un canal para los drivers que controlan los motores

NEMA 23 y otra para los motores NEMA 17, como referencia de la fuente de poder utilizada visualice la Ilustración 43-3 y para una mejor comprension de las caracteristicas la Tabla 14-3.

Ilustración 43-3: Fuente de Poder SPD3303X-E
Realizado por: Bonilla, H., 2023

Tabla 14-3: Características de la Fuente de Poder SPD3303X-E

Tabla Siglent SPD3303X-E	
Dispositivos compatibles	Ordenador personal
Tipo de conector	USB
Potencia de salida	220 vatios
Potencia	220 vatios
Método de refrigeración	Aire
Dimensiones del artículo LxWxH	16 x 14 x 11 pulgadas
Peso del artículo	17.6 Libras
Voltaje de entrada máximo	230 voltios
Voltaje de entrada mínimo	100 voltios

Fuente: Siglent, 2021
Realizado por: Bonilla, H., 2023.

3.15.7. Cilindro Neumático JELPC MSA25X100-S

El módulo automático de alimentación es esencial en el proceso de ensamblaje, utilizara un cilindro neumático de simple efecto con una carrera de desplazamiento de 100 mm para asegurar una dosificación precisa de las bases y tapas del conjunto. Este cilindro, regulado en velocidad por una electroválvula, brinda una sujeción óptima sin riesgos de sobrecalentamiento, manteniendo un desempeño estable en condiciones de temperatura ambiente y garantizando un dosificación eficiente y confiable para tener una idea más clara se tiene la Ilustración 44-3.
La combinación de tecnología neumática y un control preciso asegura un flujo de trabajo eficiente y sin contratiempos, facilitando así la correcta disposición de los elementos a ensamblar y contribuyendo al éxito global del sistema, en la Tabla 15-3 se muestran las características del pistón.

Ilustración 44-3: Pistón Neumático Jelpc MSA25X100-S
Realizado por: Bonilla, H., 2023

Tabla 15-3: Características del Pistón Neumático Jelpc MSA25X100-S

Tabla Pistón neumático MSA25 X100-S	
Operaciones	Simple Efecto
Medio de trabajo	Aire
Montajes	Básico LB FA FB SDB
Rango de presión de funcionamiento	0.1～1.0MPa
Presión de prueba	1.5 MPa
Rango de temperatura de funcionamiento	-20～80°C
Rango de velocidad de operación	50～800mm/s
Almohadón	Cojín de junta
Puerto	M5×0.8, G1/8

Fuente: Jelpc, 2022
Realizado por: Bonilla, H., 2023.

3.15.8. Tarjeta de desarrollo ARDUINO

Se considero la tarjeta de desarrollo Arduino por su plataforma abierta, amigable con el usuario, alta flexibilidad, costo bajo y una gran variedad de placas que permiten construir proyectos, prototipos, modelos y dispositivos autónomos funcionales, permitiendo interactuar tanto con el hardware como con el software.

Ilustración 45-3: : Tarjeta de desarrollo Arduino MEGA
Realizado por: Bonilla, H., 2023.

Para la correcta selección de la tarjeta de desarrollo se realizó una tabla comparativa de las características entre Arduino Mega, Arduino Uno y Arduino Leonardo, las cuales están descritas en la Tabla 16-3, entre los requisitos que debe tener la tarjeta de desarrollo para el prototipo de línea de ensamble es que debe tener un mínimo de 16 pines digitales de salida.

Tabla 16-3: Características del Arduino Mega 2560, Arduino Uno y Arduino Leonardo

	Tabla Arduino mega 2560	**Arduino Uno**	**Arduino Leonardo**
Tamaño de la placa	101.52×53.3 mm2	68.6mm x 53.4mm	68.6mm x 53.4mm
Microcontrolador	ATmega2560	ATmega328	ATmega32u4
Memoria FLASH	256 KB	32 KB	32 KB
Memoria SRAM	8 KB	2 KB	2.5 KB
Memoria EEPROM	4 KB	1 KB	1 KB
Nivel Lógico	5V	5 V	5 V
Entradas/Salidas Digitales	54	14	20
Salidas Analógicas (PWM)	15	6	7
Entradas Analógicas	16	6	12
Puertos Serial	4	1	1
Comunicación SPI	si	si	si
Comunicación I2C	si	si	si
Corriente por Pin E/S	20 mA	20 mA	20 mA
Corriente por pin de 3.3V	50 mA	50 mA	50 mA

Fuente: Luis Del Valle, 2021
Realizado por: Bonilla, H., 2023
.

El Arduino mega 2560 es la tarjeta de desarrollo seleccionada, este modelo es perfecto para satisfacer las necesidades del prototipo, por su amplio número de entradas y salidas digitales (54), su memoria flash de 256 KB y su memoria SRAM de 8 KB, por estas características se descartan a la tarjeta de desarrollo Arduino uno y Arduino leonardo.

3.15.9. Electroválvula AIRTAG

El prototipo de línea de ensamblaje incorpora una electroválvula fundamental para controlar con precisión el flujo de aire suministrado al pistón Ilustración 46-3, asegurando el correcto funcionamiento del módulo automático de alimentación de bases y tapas. Esta electroválvula, de configuración normalmente cerrada, garantiza un arranque sin complicaciones en el proceso de ensamblaje. Su versatilidad se refleja en la capacidad de manejar fluidos como aire, agua y aceite, sin requerir lubricación adicional, y está construida principalmente con aluminio para prevenir la corrosión.

La electroválvula desempeña un papel crucial al optimizar la eficiencia de la línea de ensamblaje por sus características, las mismas que están especificadas en la Tabla 17-3, facilitando un control fluido y seguro en la distribución de aire para el módulo de alimentación. Con su diseño robusto

y materiales resistentes, contribuye de manera significativa a la confiabilidad y durabilidad del sistema, asegurando un funcionamiento preciso y sin contratiempos en la producción.

Ilustración 46-3: Electroválvula neumática Airtac 3V1-06
Realizado por: Bonilla, H., 2023

Tabla 17-3: Características de la Electroválvula neumática Airtac 3V1-06

AirTac 3V1 - 06	
Fluido	Aire (a filtrar 40 μm elemento filtrante)
Actuación	Acción Directa
Tamaño del puerto	1/8"
Tipo de válvula	3 puertos y 2 posiciones
Lubricación	No se requiere
Presión de funcionamiento	0-0. 8mpa (0-114psi)
A prueba de presión	1,2 MPa(175psi)
Tamaño del orificio	Φ mm
Material del cuerpo	Aleación de aluminio

Fuente: Rome Co Indutrial, 2021
Realizado por: Bonilla, H., 2023.

3.15.10. Fin de carrera KW11-3Z

El prototipo de línea de ensamble consta de 3 finales de carrera que ayudan a posicionar el robot cartesiano de manera independiente en cada uno del eje (X, Y, Z), adicional a estos se agrega un final de carrera para posicionar la mordaza del gripper de modo que se pueda controlar de una mejor manera el posicionamiento de este con respecto a la pieza a tomar.

Dentro de la línea de ensamble se necesitará detectar la existencia de base y tapa para el ensamblaje, el módulo de desplazamiento contara con dos sensores los cuales indicaran el posicionamiento de la plaza de desplazamiento para realizar el ensamble, para un mejor entendimiento del sensor a utilizar mírese la Ilustración 47-3

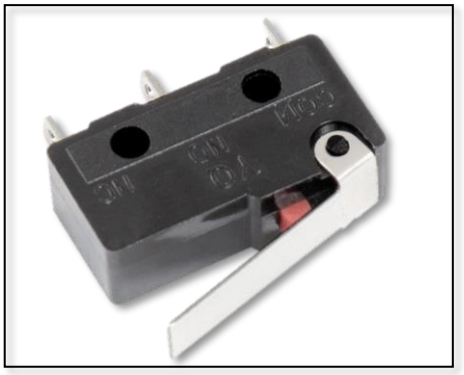


Ilustración 47-3: Fin de Carrera KW11-3Z
Realizado por: Bonilla, H., 2023.

3.16 Conexiones del Hardware

Se realizo el diseño de conexión de hardware para el prototipo de línea de ensamble, a fin de obtener un mejor entendimiento de estas con un total de cuatro gráficos para una mejor comprensión individual no obstante la tarjeta Arduino es el único componente que se tiene en común dentro de los mismos.

3.16.1. Conexión de motores nema

La Ilustración 48-3 representa el diagrama de conexión de los motores nema 23 para los ejes del robot cartesiano denominados como M1, M2 y M3.

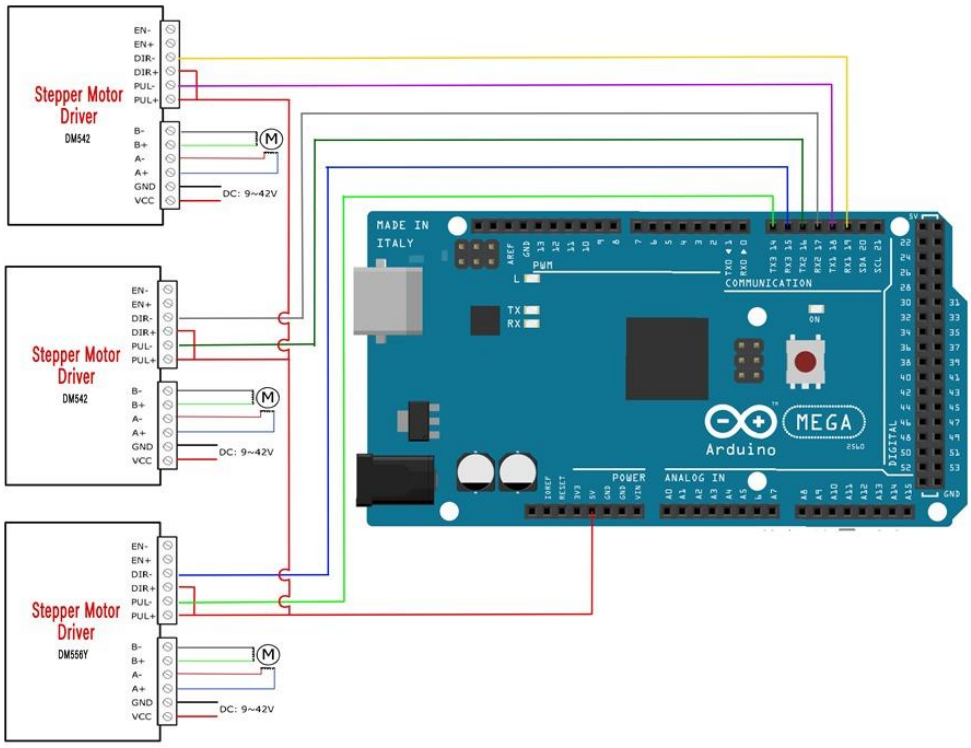


Ilustración 48-3: Esquema de conexión de motores nema 23
Realizado por: Bonilla, H., 2023.

El Ilustración 49-3 representan la conexión de los motores nema 17 lo cuales están denominados como M4, M5 para el módulo de módulo de desplazamiento y el gripper respectivamente.

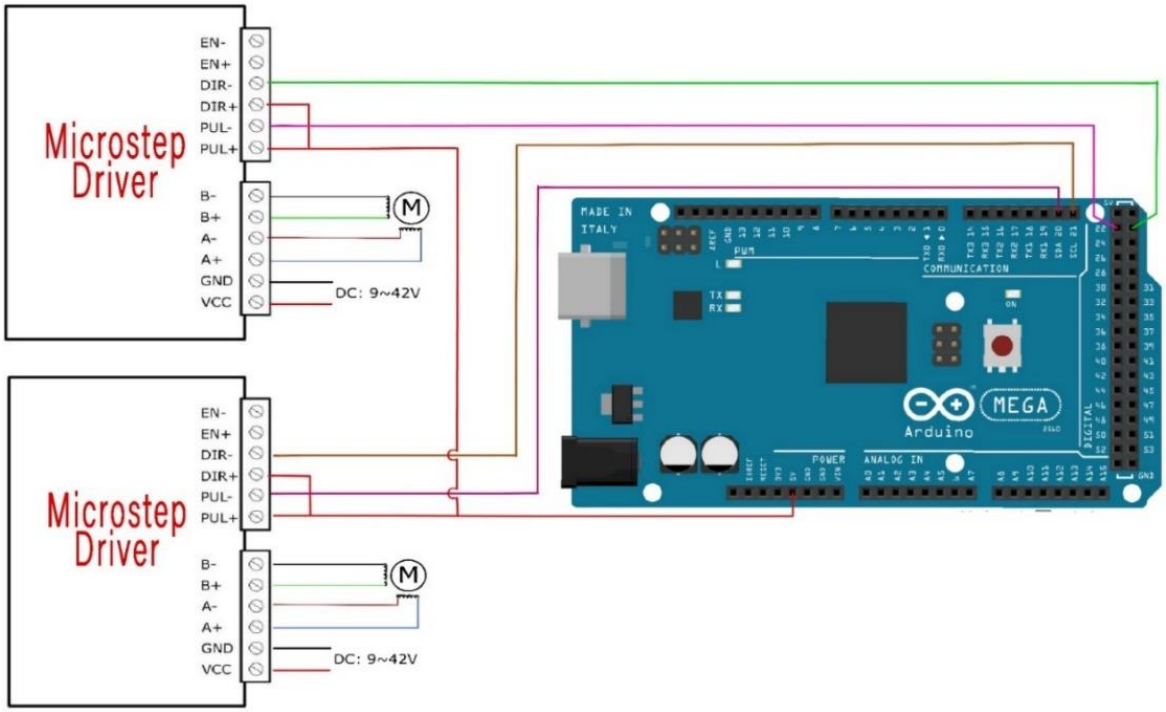


Ilustración 49-3: Esquema de conexión de motores nema 17
Realizado por: Bonilla, H., 2023.

La Tabla 18-3 representa la descripción de la conexión de los terminales de Arduino con las señales de pulso y dirección de los drivers para los motores paso a paso, cada driver tiene un pulso positivo y uno negativo, de la misma forma la dirección, conectados a los diferentes pines del Arduino Mega.

Tabla 18-3: Terminales de conexión del Arduino con señales de pulso y dirección

Arduino	DRIVER M1	DRIVER M2	DRIVER M3	DRIVER M4	DRIVER M5
Terminal 5v	PUL+ DIR+	PUL+ DIR+	PUL+ DIR+	PUL+ DIR+	PUL+ DIR+
Terminal 14	PUL-				
Terminal 15	DIR-				
Terminal 16		PUL-			
Terminal 17		DIR-			
Terminal 18			PUL-		
Terminal 19			DIR-		
Terminal 20				PUL-	
Terminal 21				DIR-	
Terminal 22					PUL-
Terminal 23					DIR-

Realizado por: Bonilla, H., 2023.

3.16.2. Conexión de sensores

En la Ilustración 50-3 muestra el esquema de conexión de los sensores para setear el robot cartesiano incluido el gripper los cuales están conectados al Arduino mega.

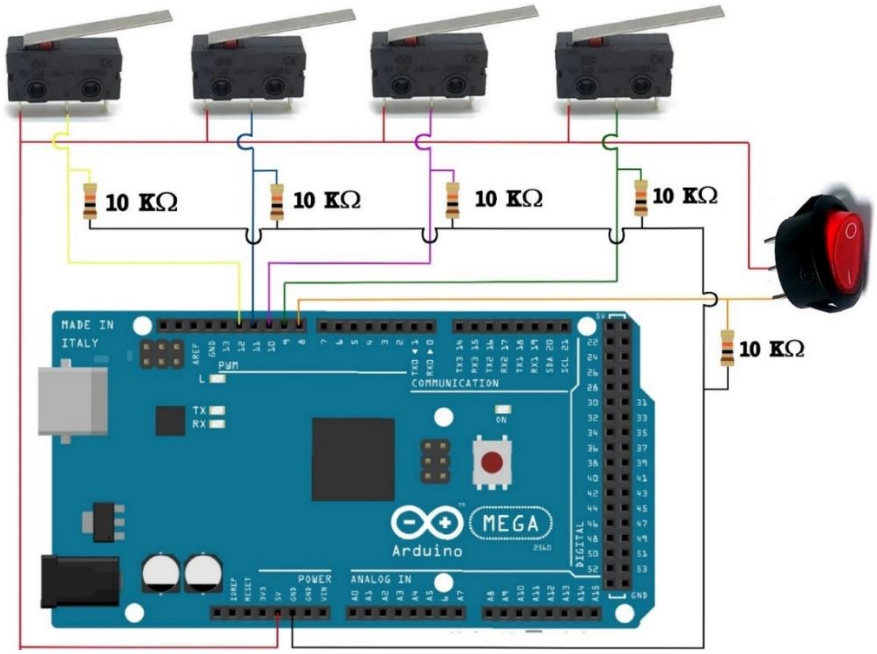


Ilustración 50-3: Esquema de conexión de finales de carrera
Realizado por: Bonilla, H., 2023.

La Ilustración 51-3 es el esquema de conexión de las señales de entrada como el sensor ultrasónico, relé y sensores para la línea de ensamble incluido el módulo de desplazamiento.

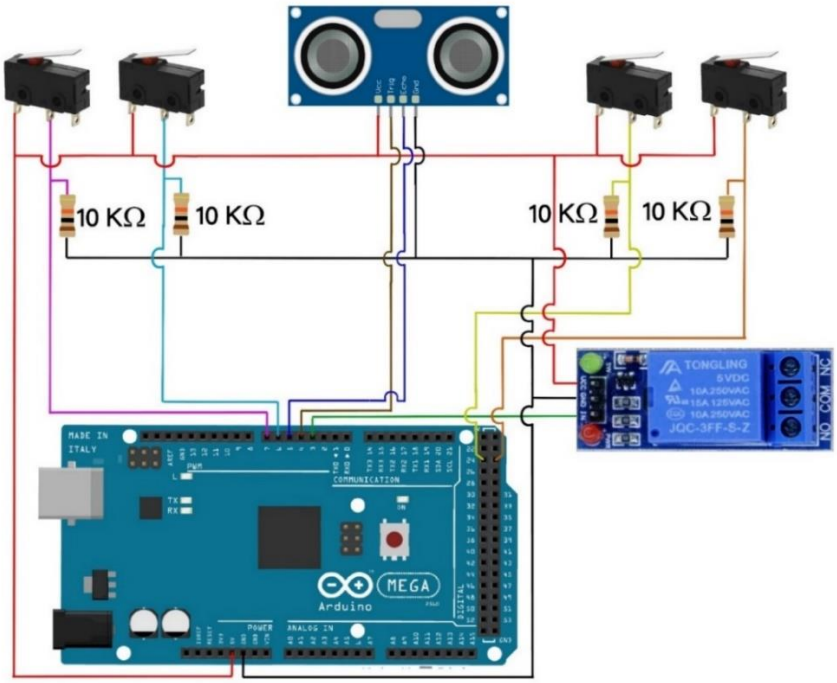


Ilustración 51-3: Esquema de conexión señales de entrada, sensor ultrasónico y relé.
Realizado por: Bonilla, H., 2023.

La Tabla 19-3 representa la descripción de la conexión de los terminales de Arduino con las señales de ingreso, relé y sensor ultrasónico del prototipo de línea de ensamble.

Tabla 19-3: Terminales de conexión del Arduino con señales de entrada

Arduino	Relé-pistón	Sensor ultrasónico	K1	K2	K3	K4	K5	K6	K7	K8	K9
Terminal 5v	VCC	VCC	COM	COM	COM	COM	COM	COM	COM	COM	COM
Terminal GND	GND	GND									
Terminal 3	IN										
Terminal 4		TRIGER									
Terminal 5		ECHO									
Terminal 6			NO								
Terminal 7				NO							
Terminal 8					NO						
Terminal 9						NO					
Terminal 10							NO				
Terminal 11								NO			
Terminal 12									NO		
Terminal 24										NO	
Terminal 25											NO

Realizado por: Bonilla, H., 2023.

- **Arduino Mega**

Esta placa de desarrollo de hardware es una versión ampliada del Arduino UNO, está diseñada para proyectos más complejos que requieren mayor cantidad de pines de entrada/salida y recursos de memoria convirtiéndolo en una opción popular para proyector de robótica, automatización entre otros complejos.

- **Motor Nema**

Es un motor paso a paso el cual tiene una amplia variedad de aplicaciones industriales y comerciales, este tipo de motor es más conocido por si alta precisión y capacidad de posicionamiento mediante la generación de movimientos angulares precisos

- **Final de carrera**

Este dispositivo es utilizado en diferentes aplicaciones industriales el cual se acciona mediante un mecanismo físico al entrar en contacto con un objeto que se encuentra en movimiento, al activarse el mecanismo se lanza una señal eléctrica para detener, cambiar de dirección o controlar un mecanismo dentro del prototipo.

- **Sensor ultrasónico**

Utiliza ondas de sonido de alta frecuencia para detectar la presencia, distancia, nivel o movimiento de objetos en el entorno situado, su función es la de detectar la presencia de placas de Arduino UNO en la porta placas del prototipo de línea de ensamble.

- **Relé**

Este dispositivo electromecánico controla el flujo de corriente eléctrica del circuito para la electroválvula, funciona como un interruptor para dar el accionamiento al pistón, accionando el dispensador de bases y tapas para su respectivo ensamble.

- **Electroválvula**

Empleado para controlar el flujo de aire en el sistema del dispensador de bases y tapas, es capaz de abrir y cerrar el paso del fluido en respuesta a la señal eléctrica del relé.

3.17 Software

Esta etapa es crucial en el desarrollo del prototipo, se enfoca en la integración estratégica de software para potenciar tanto la interfaz de usuario como el rendimiento operativo del prototipo de la línea de ensamblaje. Se trabaja en la programación de una interfaz de usuario intuitiva y funcional, facilitando la interacción fluida entre el usuario y el sistema al priorizar la navegabilidad amigable y la presentación clara de información relevante. Simultáneamente, se concentra en desarrollar un software que coordine de forma precisa cada fase del proceso, desde la alimentación de las piezas hasta el ensamblaje final, minimizando posibles errores en la manipulación y ensamblaje de los componentes.

3.17.1. Código Arduino

Para controlar de manera adecuada el prototipo de línea de ensamble se usa un Arduino mega por el gran número de pines que tiene, el proceso para elaborar el programa conlleva varios pasos fundamentales, desde la planificación y el diseño hasta la codificación, prueba y puesta en funcionamiento. Las siguientes condiciones nos ayudaran a delimitar de una mejor manera el código.

- Solo se efectuará el ensamble cuando el robot cartesiano se encuentre posicionado en un punto cero.
- Debe existir la presencia de base, tapa y placa para el ensamble, si no llegase a existir alguno de estos, el código, no dará paso a seguir ensamblando las cajas.
- Como condicional para evitar un mal posicionamiento del robot cartesiano se centrará el gripper, luego el eje Y, seguido por el eje X y por último el eje Z.

3.17.1.1 Control y monitoreo de variables del prototipo

La Ilustración 52-3 es una representación visual del algoritmo que sigue el prototipo de línea de ensamble, se /muestra de una manera clara y concisa la secuencia de las acciones y decisiones que se toman para llevar a cabo el ensamble del conjunto de piezas.

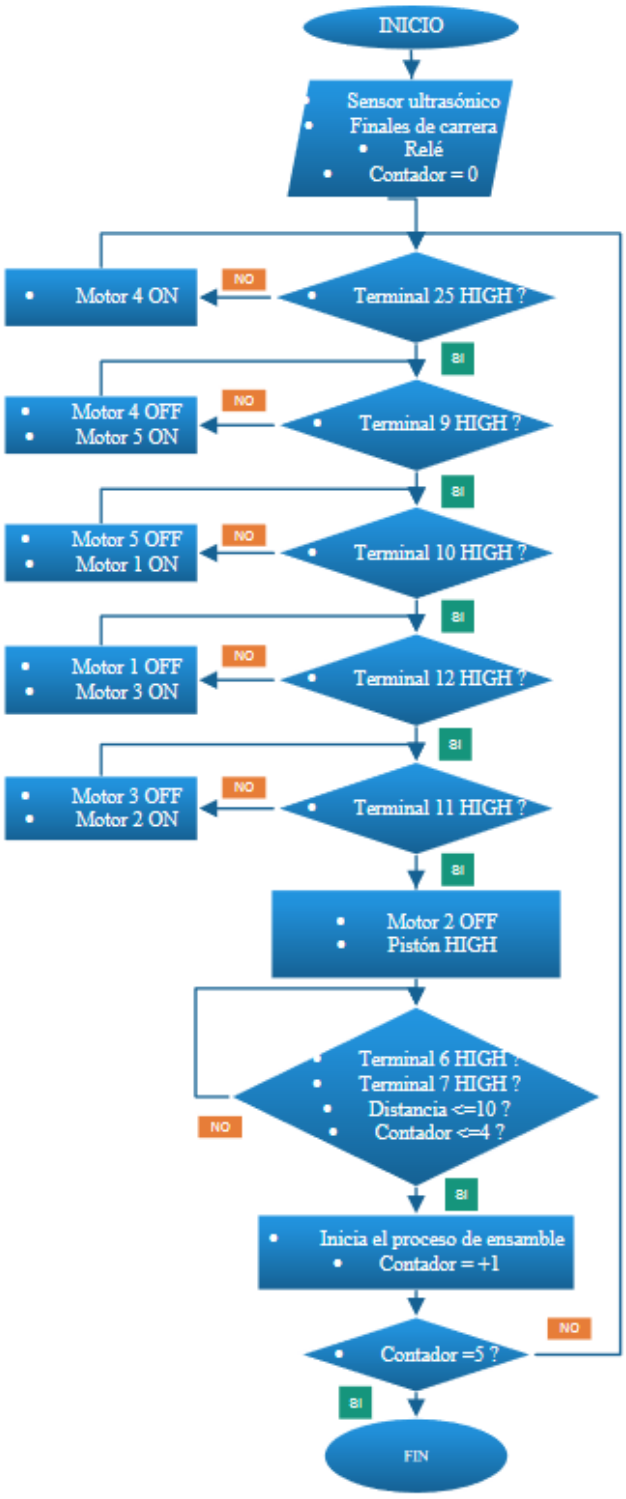


Ilustración 52-3: Diagrama de flujo del prototipo
Realizado por: Bonilla, H., 2023.

- El algoritmo inicia con una lectura de las señales de entrada y salida, como el relé, sensor ultrasónico y finales de carrera, al mismo tiempo que se inicia la interfaz gráfica para determinar las condiciones iniciales en las que se encuentra el prototipo.

- Adquirido los valores de las señales de los sensores se inicia el centralizado del robot cartesiano en un punto cero para evitar errores de ensamblaje y tener un punto de referencia al comenzar con los diferentes procesos.
- Al tomar en cuenta el valor del sensor ultrasónico este deberá ser menor o igual a 10, caso contrario significa que no existe la presencia de placas de Arduino para su ensamblaje, de la misma forma el terminal 6 y 7 determinara la existencia de base y tapa para iniciar con el proceso, cabe recalcar que el ensamblaje no iniciará hasta que exista la presencia de todas las piezas necesarias.
- En caso de cumplir con las condiciones especificadas se dará inicio al proceso de ensamble con pequeñas variaciones dependiendo del número de caja a ensamblar, caso contrario de no cumplir con las condiciones el prototipo no dará paso al ensamblaje de las piezas.
- Cuando el contador llegue a cinco la programación dará por finalizado el proceso de ensamblaje de las cinco cajas dando paso al abastecimiento de bases, tapas y tarjetas de Arduino para el prototipo.

El lenguaje en el cual se realizó el código de programación es C++ por su alto rendimiento y eficiencia al ser un lenguaje de bajo nivel, permite un control cercano al hardware, esto es beneficioso en prototipos que requieren un rendimiento optimo y por su compatibilidad con diferentes plataformas entre estas se encuentra la misma con la que esta echa la interfaz gráfica.

3.17.2. Interfaz gráfica del prototipo

La creación de la interfaz gráfica es un complemento esencial en el desarrollo del prototipo de línea de ensamble por la innovación constante de las empresas con sus maquinarias, la interfaz es intuitiva y atractiva mejorando la experiencia del operador, se encuentra escrita en el lenguaje de programación C# en el programa Visual estudio 2022 con formularios una de las herramientas de desarrollo proporcionada por Microsoft.

Para elaborar la interfaz gráfica, se procedió inicialmente con la creación de un diagrama de flujo como se muestra en la Ilustración 53-3. Este diagrama permitió visualizar y planificar de manera organización y secuencial las diversas etapas de la animación, mediante este proceso se establecieron las conexiones lógicas entre las diferentes funciones y componentes, lo que facilito una comprensión clara del flujo de interacción del usuario con la interfaz. El diagrama de flujo se convirtió en una guía fundamental para el desarrollo y diseño de la interfaz gráfica, garantizando que cumpliera los objetivos establecidos y proporcionando una base sólida para avanzar en el proceso de la implementación.

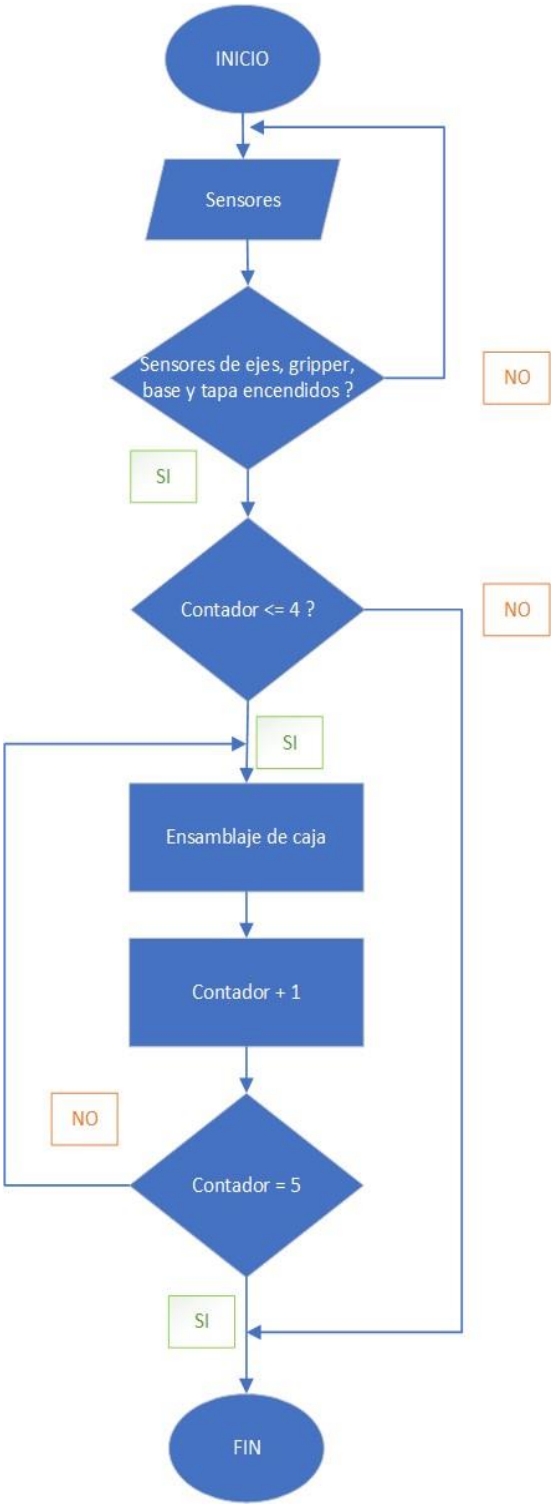


Ilustración 53-3: Diagrama de flujo de la interfaz grafica
Realizado por: Bonilla, H., 2023

Por el manejo local de la interfaz se implementó dos botones los cuales son inicio y reinicio, con la finalidad de poder controlar con eficacia el prototipo de línea de ensamble, agregado a esto dentro de la interfaz podemos notar la existencia de 5 botones los cuales son condicionales denominados como eje X, eje Y, eje Z, gripper y módulo de desplazamiento, cabe recalcar que tanto en el prototipo como en la interfaz se mantiene la condicional de no efectuar ningún

ensamble en el caso de no existir base, tapa o Arduino en la línea de ensamble, el Ilustración 54-3 describe la pantalla de inicio de la interfaz gráfica.

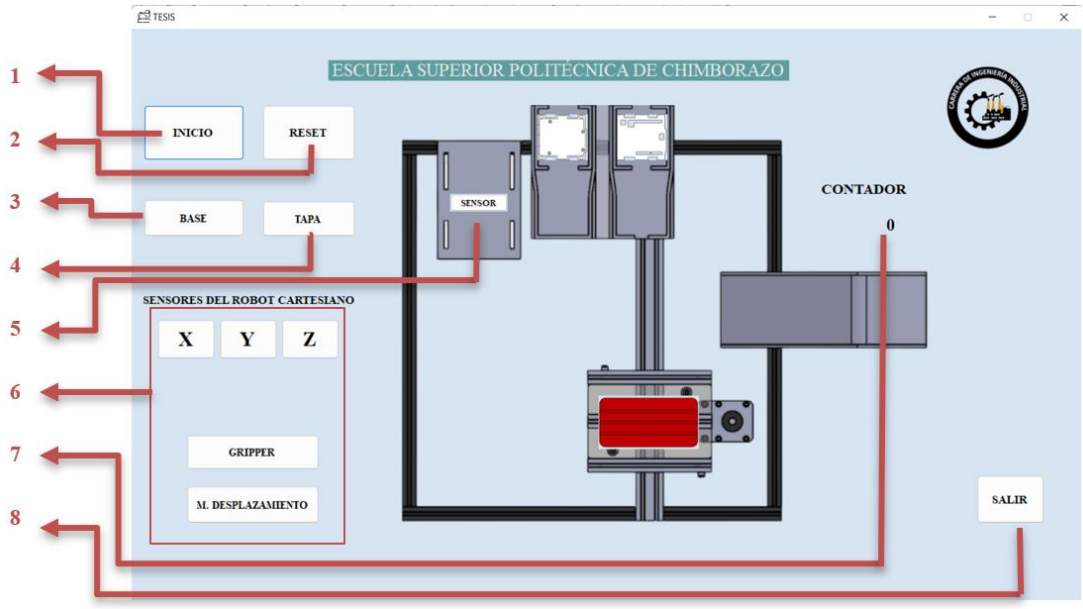


Ilustración 54-3: Pantalla de inicio
Realizado por: Bonilla, H., 2023

La interfaz tiene 7 recursos gráficos para la interacción con el prototipo de línea de ensamble, se encuentran al lado izquierdo:

I. **Inicio:** El botón inicio está diseñado para ejecutar el código de Arduino del prototipo enviando una señal de inicio al robot cartesiano, dependiendo de la posición de este se ejecutarán las acciones en el prototipo, en caso de encontrarse en el punto cero y cumple con las condiciones parta iniciar el ensamble se ejecutará el código correspondiente caso contrario no lo hará.

II. **Reinicio:** Incluido en la interfaz gráfica con la finalidad de restablecer o reiniciar el prototipo y contador a su estado original, punto de inicio, en caso de terminar el proceso de ensamblaje o presentar anomalías, permitiendo al operador comenzar de nuevo sin la necesidad de interactuar de forma física con el prototipo

III. **Base:** Este botón permite dosificar la base de manera manual en caso de ser necesario, esto quedaría a criterio del operador

IV. **Tapa:** Este botón permite dosificar la tapa de manera manual en caso de ser necesario, esto quedaría a criterio del operador

V. **Sensor:** De la misma forma como con el botón de base y tapa, permite detectar la existencia de Arduino en caso de existir algún fallo el operador puede dar señal para iniciar el ensamble

VI. **Sensores:** Permiten poder realizar el seteo de forma manual

VII. **Contador:** El contador nos permite contabilizar el número de ensamble que se ha efectuado, el prototipo de línea de ensamble se lo codificara para un máximo de 5 ensambles.

VIII. **Salir:** Este botón permitirá al usuario cerrar la interfaz gráfica.

3.17.2.1 Animación de la interfaz

La interfaz necesita de una animación grafica para mejorar la experiencia del operador, existen efectos de deslizamiento por parte de la base, tapa, placa de Arduino y el módulo de desplazamiento, en la Ilustración 55-3 se muestra la posición inicial del prototipo desde una vista superior del mismo y las trayectorias de la base, tapa.

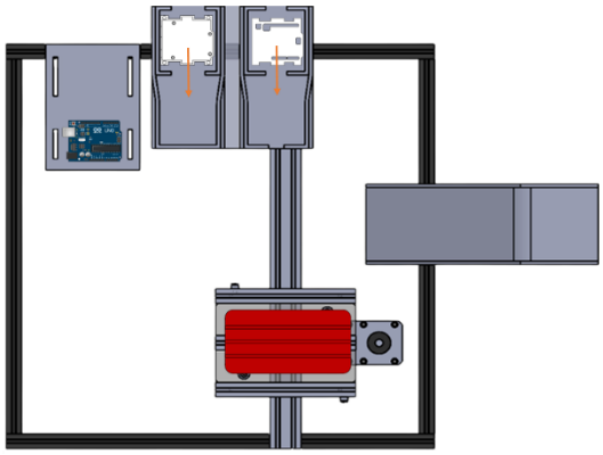


Ilustración 55-3: Posición inicial
Realizado por: Bonilla, H., 2023

La Ilustración 56-3 muestra cómo se desplaza la base y tapa a las posiciones requeridas dando paso al inicio del ensamblaje, desplazando la placa hasta la base y activando el módulo de desplazamiento.

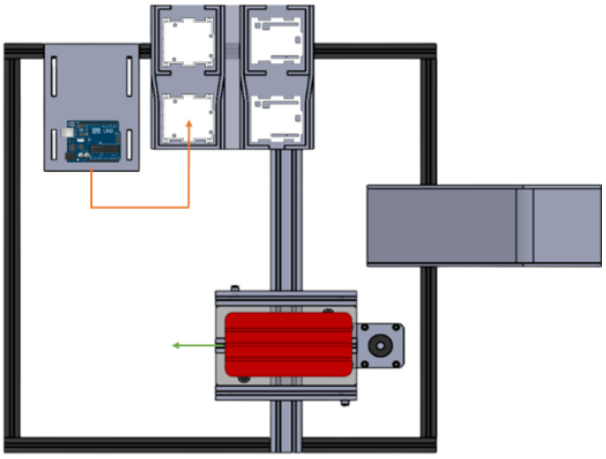


Ilustración 56-3: Posicionamiento del Arduino hasta la base
Realizado por: Bonilla, H., 2023

En la Ilustración 57-3 se verifica el posicionamiento de la placa de Arduino en la base de la caja e indica el trayecto del junto base-placa hasta el módulo de desplazamiento.

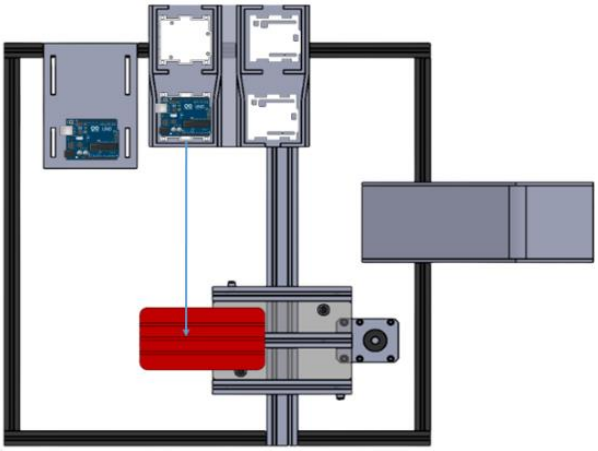


Ilustración 57-3: Desplazamiento del conjunto
Realizado por: Bonilla, H., 2023

La Ilustración 58-3 muestra la trayectoria a seguir el módulo de desplazamiento con el conjunto de base y placa para posteriormente colocar la tapa y ensamblar en conjunto

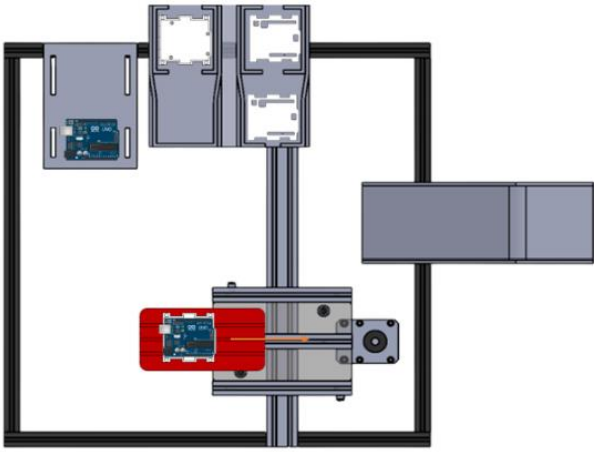


Ilustración 58-3: Módulo de desplazamiento
Realizado por: Bonilla, H., 2023

Se observa en la Ilustración 59-3 el posicionamiento adecuado del módulo de desplazamiento y la trayectoria a seguir la tapa del conjunto para ser ensamblada

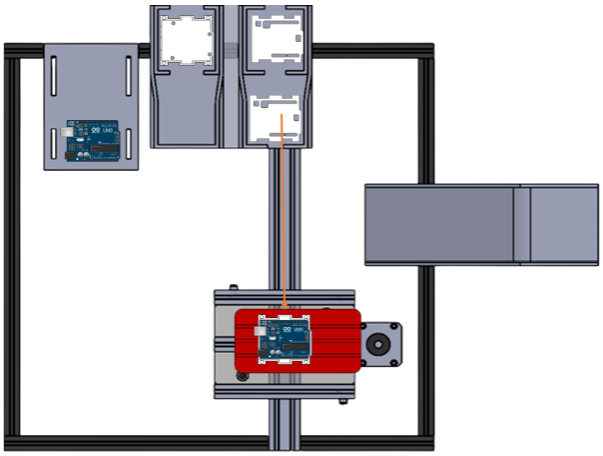


Ilustración 59-3: Ensamble del conjunto
Realizado por: Bonilla, H., 2023

Tras realizar el ensamble del conjunto base, placa y tapa se procede a despachar el conjunto por una ranfla ubicada en el costado derecho del prototipo como se muestra en la Ilustración 60-3.

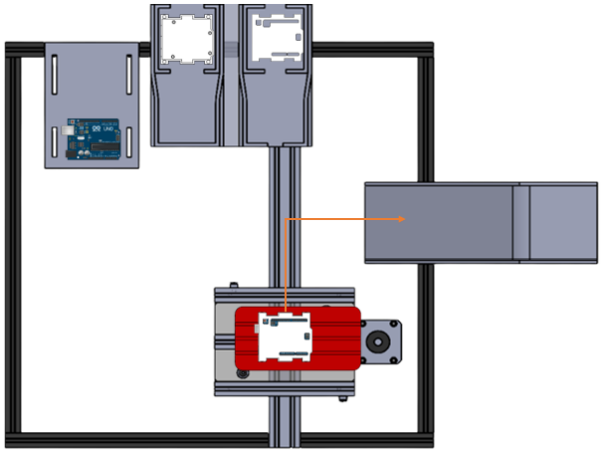


Ilustración 60-3: Despacho del ensamblaje
Realizado por: Bonilla, H., 2023

La Ilustración 61-3 muestra la trayectoria de despacho del conjunto ensamblado además de un contador para identificar el número de ensambles que lleva el prototipo.

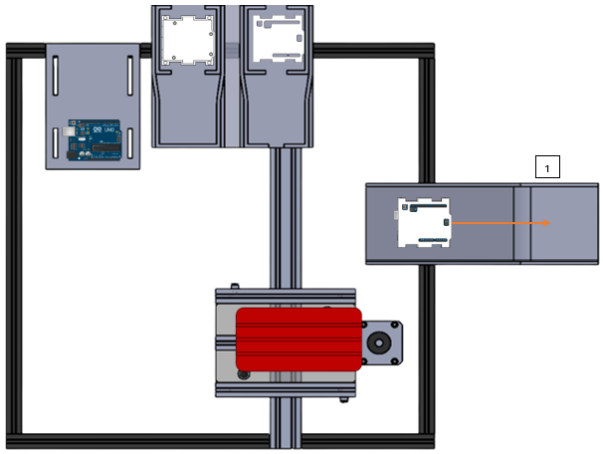


Ilustración 61-3: Conteo de número de cajas
Realizado por: Bonilla, H., 2023

3.18 Implementación del prototipo

Con el modelado de las diferentes piezas del prototipo se empezó con la construcción y ensamble de este, inicialmente verificando la existencia de todas las partes y componentes necesarios para el prototipo, así como las condiciones de estos.

En primer lugar, se ensambla la estructura del prototipo de línea de ensamble para posteriormente acoplar las diferentes partes y componentes del prototipo que se ensamblaran por separado. La Ilustración 62-3 muestra el ensamble de la estructura del prototipo de línea de ensamble de partes plásticas y el ensamble de los diferentes drivers que controlaran los motores paso a paso.

Ilustración 62-3: Ensamble de estructura para el prototipo
Realizado por: Bonilla, H., 2023

El módulo de desplazamiento se lo ensamblo por separado para una mayor comodidad, posteriormente se lo junto con la estructura del prototipo, para una mejor comprensión mírese la Ilustración 63-3 donde se observa el ensamble por separado. El módulo de desplazamiento cuenta con un juego de poleas donde se debe tensar de manera adecuada la banda dentada para evitar pivoteo o desplazamientos innecesarios al ejecutar la programación, esto se lo hace implementando una polea loca en la placa inferior del módulo de desplazamiento.

Ilustración 63-3: Ensamble del módulo de desplazamiento
Realizado por: Bonilla, H., 2023

La Ilustración 64-3representa el ensamble del módulo de desplazamiento con la estructura del prototipo de línea de ensamble.

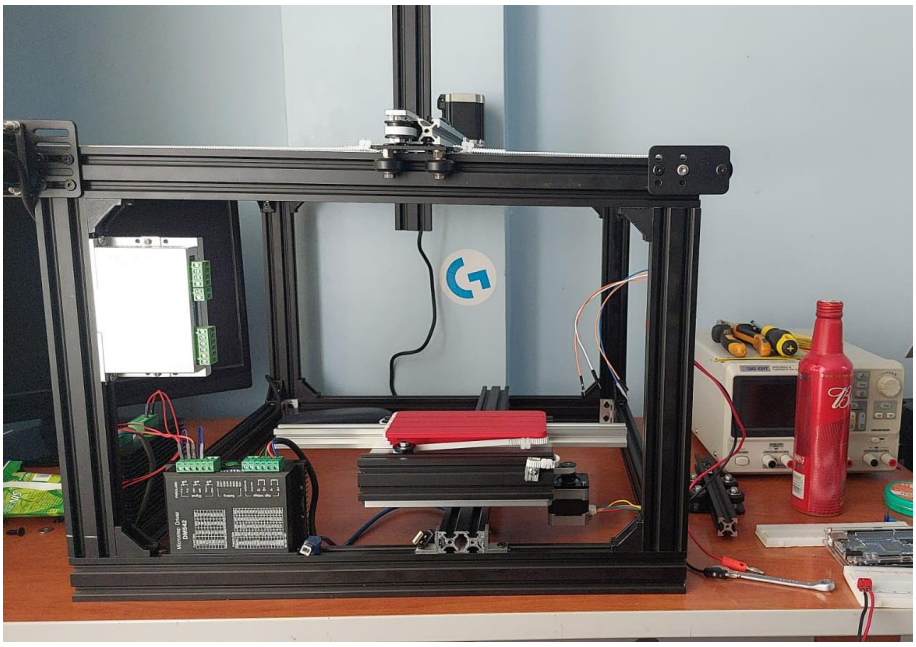


Ilustración 64-3: Ensamble del módulo de desplazamiento en la estructura
Realizado por: Bonilla, H., 2023

El dispensador de bases y tapas dosifica de manera mecánica, con la ayuda de un pistón para el desplazamiento de las partes y utilizando la fuerza de la gravedad para la recolocación de estas, la Ilustración 65-3muestra una capacidad de 5 unidades el dispensador dosifica de manera simultánea las bases y tapas para el ensamblaje de la caja.

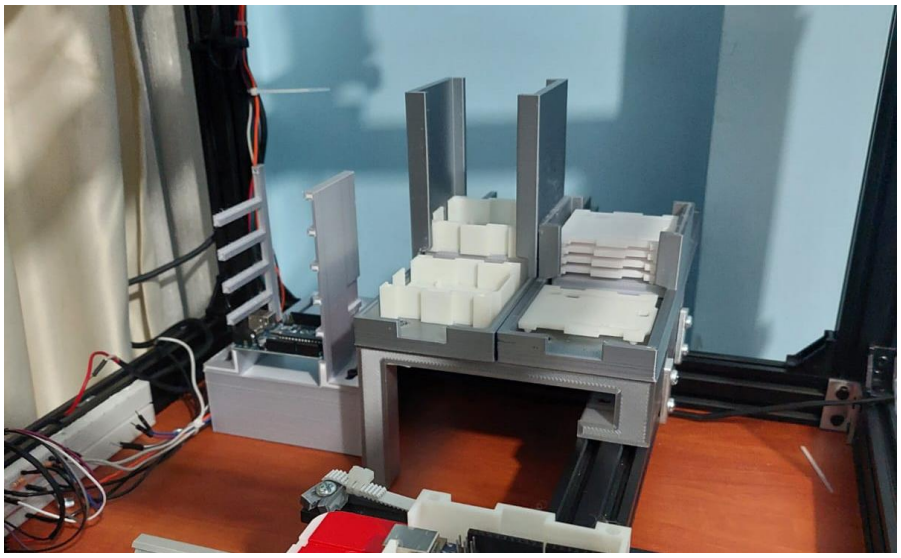


Ilustración 65-3: Ensamble de la porta placas de Arduino y dispensador de bases – tapas
Realizado por: Bonilla, H., 2023

Con la finalidad de evitar errores y garantizar el correcto funcionamiento del dispensador de bases y tapas se agregan dos finales de carrera para su detección, estos darán señales para iniciar el proceso de ensamble asegurando la detección de estas, Ilustración 66-3.

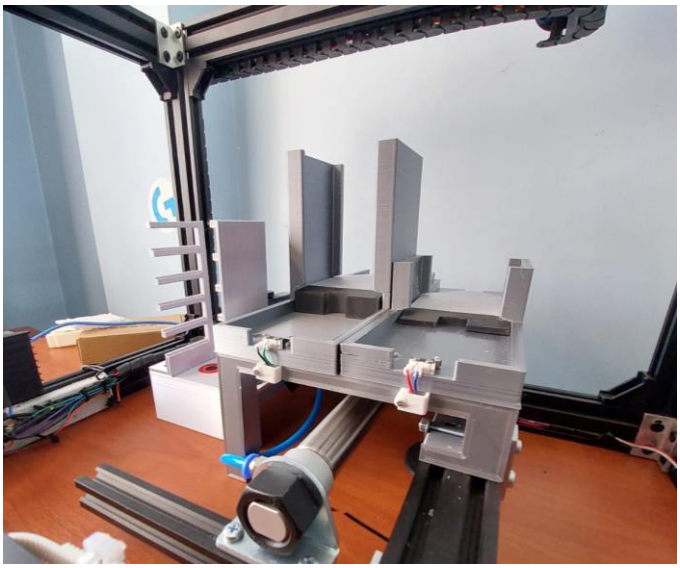


Ilustración 66-3: Ensamble de pistón en la estructura
Realizado por: Bonilla, H., 2023

Es necesario la protección física de los cables y conductores eléctricos para evitar enredamientos, desgaste o roturas durante el movimiento del robot cartesiano por ello se coloca cadenas porta cables, ayudando a mantener la integridad de los cables y garantizar una conexión eléctrica confiables, para una mejor visualización de lo mencionado mírese la Ilustración 67-3.

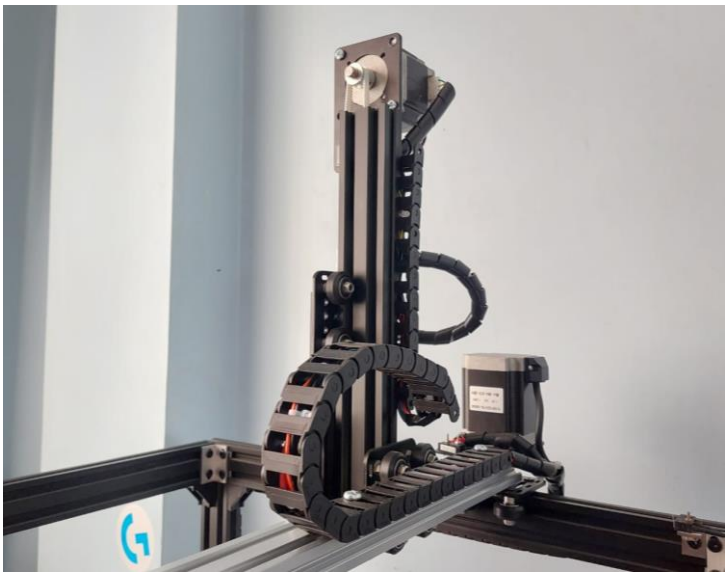


Ilustración 67-3: Acoplamiento de cadena porta cables
Realizado por: Bonilla, H., 2023

Para la manipulación de las piezas se implementó un gripper, este nos permite agarrar y mover los objetos a la posición deseada para su correcto ensamblaje, la pinza robótica está diseñada con medidas especifica, funciona con un motor paso a paso nema 17 en su interior, una pinza dentada y un final de carrera para tener un punto de referencia evitando errores, mejorando la eficacia de la pinza como se muestra en la Ilustración 68-3.

Ilustración 68-3: Ensamble del gripper en la estructura
Realizado por: Bonilla, H., 2023

Una vez ensamblado la estructura, dispensador de bases-tapas, módulo de desplazamiento, gripper y la conexión de los diferentes componentes eléctricos necesarios para el funcionamiento del prototipo de línea de ensamble obtenemos un prototipo sólido, funcional y eficaz que satisface nuestras necesidades y principios básicos de funcionamiento, Ilustración 69-3.

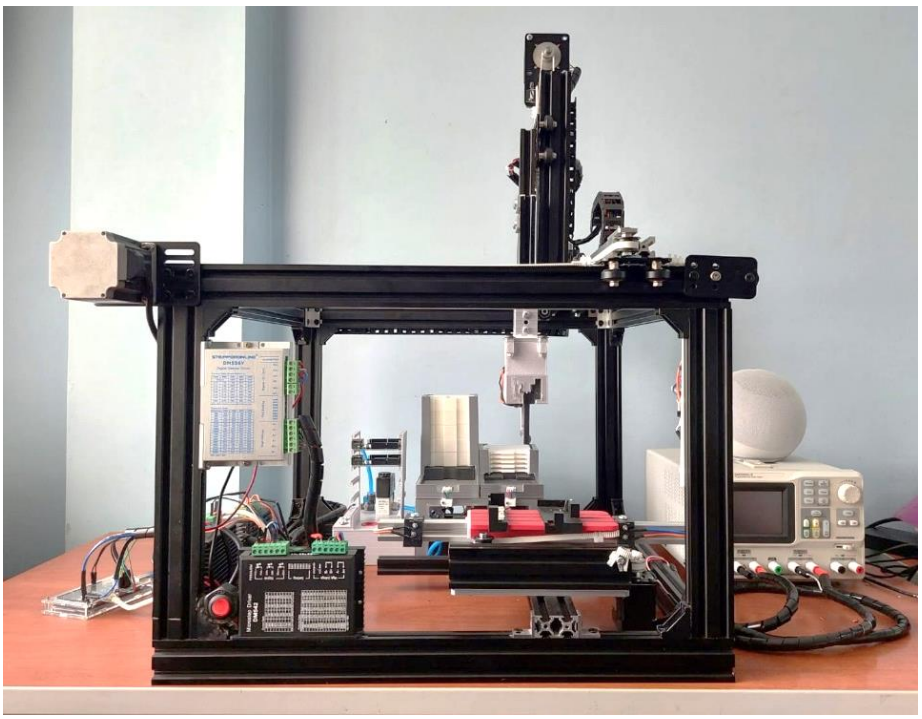


Ilustración 69-3: Prototipo de línea de ensamble
Realizado por: Bonilla, H., 2023

3.18.1. Desplazamiento secuencial en pasos

En este apartado se presenta el desplazamiento secuencial y detallado de ensamble que utiliza los diferentes motores paso a paso. Estos motores están diseñados para llevar a cabo un número especifico de pasos que, en conjunto, conducen al ensamble de 5 cajas hasta un punto final común, el anexo G muestra de una manera detallada la dirección, el actuador, la caja y el número de pasos necesarios para ejecutar el desplazamiento del robot cartesiano permitiendo determinar el punto final del mismo, existe un variación de pasos dependiendo del número de caja a ensamblar debido a la diferencia de distancia que existe entre las placas a ensamblar.

La Tabla 20-3 muestra el desplazamiento secuencial del robot cartesiano en pasos, se toma solo en cuenta los tres motores que están destinados al sistema de desplazamiento del robot cartesiano.

Tabla 20-3: Desplazamiento secuencial del robot cartesiano en pasos

Actuadores	EJE	N. Pasos
Motor 2	Z	1300
Motor 3	X	4180
Motor 2	Z	400
Motor 1	Y	900
Motor 2	Z	200
Motor 1	Y	20
Motor 2	Z	1500
Motor 3	X	2060
Motor 2	Z	1300
Motor 1	Y	640
Motor 2	Z	180
Motor 1	Y	250
Motor 1	Y	1000
Motor 2	Z	5430
Motor 1	Y	680
Motor 2	Z	130
Motor 2	Z	30
Motor 1	Y	680
Motor 2	Z	5000
Motor 3	X	1400
Motor 1	Y	900
Motor 2	Z	350
Motor 1	Y	230
Motor 1	Y	1000
Motor 2	Z	5520
Motor 1	Y	200

Motor 3	X	70
Motor 1	Y	100
Motor 2	Z	200
Motor 3	X	340
Motor 1	Y	170
Motor 2	Z	800
Motor 1	Y	140
Motor 1	Y	140
Motor 2	Z	800
Motor 1	Y	170
Motor 1	Y	250
Motor 2	Z	120
Motor 1	Y	650
Motor 2	Z	2800

Realizado por: Bonilla, H., 2023

La

Tabla ***21***-3 muestra la posición final del robot cartesiano luego de realizar el ensamble, se observa que no importa la caja sé que ensamble el punto inicial y final será el mismo para evitar errores demostrando la eficacia y precisión del proceso de ensamble.

Tabla 21-3: Posición final del robot cartesiano en pasos

Actuador	Posición en pasos
Motor 1	800
Motor 2	3740
Motor 3	-310
Motor 4	3900

Realizado por: Bonilla, H., 2023

3.19 Cálculo de tiempo teórico

El tiempo teórico se refiere al tiempo que se estima que tomara el prototipo en realizar el ensamble en condiciones ideales y sin considerar ningún tipo de interrupción, retraso o variación. El tiempo teórico se calculó tomando en cuenta los datos esenciales, como el número de pasos secuenciales utilizados en la programación de Arduino, los segundos por paso de cada motor y el tiempo utilizado para el módulo de desplazamiento y electroválvula.

La Tabla 22-3 muestra los datos necesarios para calcular el tiempo teórico al ensamblar una caja sin tomar en cuenta el desfase de tiempo existente por la porta placas, obteniendo un tiempo teórico de 251.965 segundos, este tiempo rara vez se puede llegar a cumplir en práctica debido a

diversos factores, como retrasos imprevistos, interrupciones, ajustes necesarios dentro del proceso.

Tabla 22-3: Tiempo teórico

Tiempo teórico				
Actuadores	Dirección	Pasos	Tiempo/paso	Tiempo total (s)
Motor 2	1	1300	0.004	5.254
Motor 3	0	4180	0.004	16.931
Motor 2	0	400	0.004	1.617
Motor 4	1			5.500
Motor 5	1	2000	0.003	6.643
Motor 1	1	900	0.004	3.648
Motor 2	1	200	0.004	0.808
Motor 5	0	2000	0.003	6.643
Motor 1	0	20	0.004	0.081
Motor 2	1	1500	0.004	6.062
Motor 1	0		0.004	0.000
Motor 3	1	2060	0.004	8.344
Motor 2	0	1300	0.004	5.254
Motor 1	1	640	0.004	2.594
Motor 5	1	3400	0.003	11.294
Motor 3	1		0.004	0.000
Motor 2	0	180	0.004	0.727
Motor 1	1	250	0.004	1.013
Electroválvula				10.000
Motor 5	0	600	0.003	1.993
Motor 1	0	1000	0.004	4.054
Motor 2	1	5430	0.004	21.945
Motor 1	1	680	0.004	2.756
Motor 5	1	1000	0.003	3.322
Motor 2	0	130	0.004	0.525
Motor 2	1	30	0.004	0.121
Motor 1	0	680	0.004	2.756
Motor 4	0			5.500
Motor 2	0	5000	0.004	20.207
Motor 3	1	1400	0.004	5.671
Motor 1	1	900	0.004	3.648
Motor 5	0	1300	0.003	4.318
Motor 2	0	350	0.004	1.415
Motor 1	1	230	0.004	0.932
Motor 5	0	1600	0.003	5.315
Motor 1	0	1000	0.004	4.054
Motor 2	1	5520	0.004	22.309
Motor 1	1	200	0.004	0.811
Motor 3	1	70	0.004	0.284
Motor 1	1	100	0.004	0.405
Motor 5	1	3000	0.003	9.965

Motor 2	0	200	0.004	0.808
Motor 3	1	340	0.004	1.377
Motor 1	0	170	0.004	0.689
Motor 2	1	800	0.004	3.233
Motor 1	1	140	0.004	0.567
Motor 1	0	140	0.004	0.567
Motor 2	0	800	0.004	3.233
Motor 1	1	170	0.004	0.689
Motor 1	1	250	0.004	1.013
Motor 2	1	120	0.004	0.485
Motor 5	0	1600	0.003	5.315
Motor 1	0	650	0.004	2.635
Motor 2	0	2800	0.004	11.316
Motor 5	1	1600	0.003	5.315
			T. Total	251.965

Realizado por: Bonilla, H., 2023

3.20 Cálculos de la distancia máxima del robot cartesiano

Se llevará a cabo el cálculo de la distancia para cada eje de desplazamiento del robot cartesiano, lo que nos permitirá determinar el número máximo de pasos necesarios y el tiempo requerido para cada movimiento. Este análisis es esencial para planificar de manera precisa y eficaz la trayectoria y los desplazamientos del robot en el espacio, asegurando así un funcionamiento óptimo y coordinado en la línea de ensamble. Al comprender la relación entre la distancia y los pasos, así como el tiempo asociado a cada desplazamiento, podremos optimizar el rendimiento del robot y garantizar una producción eficaz y precisa en el proceso de ensamblaje.

Para el cálculo de la distancia máxima recorrida en cada eje por el robot cartesiano se escribe un código funcional el cual permite contabilizar el número se pasos en cada eje desde su punto más alejado hasta el punto cero el cual está delimitado por sensores que detectaran la llegada del robot cartesiano en los diferentes ejes, la Tabla 23-3 muestra las características individuales para cada eje del robot cartesiano incluido el número de pasos obtenidos desde la ejecución del código contabilizar de pasos como se observa en el Anexo H.

Tabla 23-3: Características individuales del robot cartesiano

	Eje X	Eje Y	Eje Z	Gripper
Pasos	6735	3214	8027	7463
Tiempo	27.28	13.028	32.441	24.79
Diámetro	16	16	16	13
Pasos/ Rev	800	800	800	6400

Realizado por: Bonilla, H., 2023.

Para determinar la distancia recorrida en un número determinado de pasos es necesario tomar en cuentas las características individuales, empleando la siguiente ecuación podemos determinar la distancia recorrida por cada revolución del motor paso a paso.

$$\boldsymbol{P = d * \pi} \quad (4)$$

$$P = 16 * \pi$$

$$P = 50.2655\ mm$$

$$\boldsymbol{P = d * \pi}$$

$$P = 13 * \pi$$

$$P = 40.84\ mm$$

Realizando una regla de tres podemos determinar el número de revoluciones máximas para cada eje del robot cartesiano y el gripper.

$$\frac{800\ Pasos}{1\ rev} = \frac{6735\ Pasos}{rev_x}$$

$$rev_y = 8.42$$

$$\frac{800\ Pasos}{1\ rev} = \frac{3214\ Pasos}{rev_x}$$

$$rev_y = 4.02$$

$$\frac{800\ Pasos}{1\ rev} = \frac{8027\ Pasos}{rev_z}$$

$$rev_z = 10.03$$

$$\frac{6400\ Pasos}{1\ rev} = \frac{7463\ Pasos}{rev_g}$$

$$rev_g = 1.16$$

A continuación, se realiza el producto de la distancia recorrida por revolución del motor paso a paso y las revoluciones máximas por eje para obtener la distancia máxima a recorrer en cada eje delo robot cartesiano.

$$\boldsymbol{D = P * rev_{xyz}} \quad (5)$$

$$D = 50.27\frac{mm}{rev} * 8.42\ rev$$

$$D = 423.17\ mm$$

$$\boldsymbol{D = P * rev_y}$$

$$D = 50.27\frac{mm}{rev} * 4.02\ rev$$

$$D = 201.94\ mm$$

$$\boldsymbol{D = P * rev_z}$$

$$D = 50.27\frac{mm}{rev} * 10.03\ rev$$

$$D = 423.17\ mm$$

$$\boldsymbol{D = P * rev_g}$$

$$D = 40.84\frac{mm}{rev} * 1.16\ rev$$

$$D = 47.62\ mm$$

La Tabla 24-3 es un resumen de las características que tienen los ejes de desplazamiento del robot cartesiano, incluido el gripper, permite determinar el tiempo que se tomara el moverse de un punto a otro, el número de pasos necesarios y la distancia recorrida.

Tabla 24-3: Tabla resumen de los ejes de desplazamiento

	Eje X	Eje Y	Eje Z	Gripper
Pasos	6735	3214	8027	7463
Tiempo	27.28	13.028	32.441	24.79
Diámetro	16	16	16	13
Pasos/ Rev	800	800	800	6400
Distancia	423.17	201.94	504.35	47.62

Realizado por: Bonilla, H., 2023.

Mediante los cálculos pertinentes, se logró determinar las capacidades de desplazamiento para cada eje del sistema. El eje X tiene una amplitud de movimiento de 423 mm, mientras que el eje Y puede desplazarse hasta 201 mm. En cuanto al eje Z, se ha calculado que su recorrido máximo es de 504 mm. Por último, el gripper, elemento crucial en la manipulación, tiene la capacidad de movimiento más reducida, con un rango de desplazamiento de 47 mm. Estos datos son fundamentales para garantizar un diseño y operación precisos del robot cartesiano, permitiendo una planificación efectiva de las tareas y trayectorias en la línea de ensamble.

3.21 Diagrama de la metodología del prototipo

La Ilustración 70-3 ofrece un diagrama representativo de la metodología clave para la realización del prototipo.

Ilustración 70-3: Diagrama de la metodología del prototipo
Realizado por: Bonilla, H., 2023

Definición de Requerimientos.- Se inició con un análisis exhaustivo del problema planteado para identificar los requerimientos esenciales del proyecto. Esto incluyó definir las funciones, características y objetivos que el prototipo debía cumplir para abordar el problema de investigación de manera efectiva.

Diseño Estructural del Prototipo.- Basándose en los requerimientos definidos, se procedió a diseñar la estructura y arquitectura general del prototipo. Esto implicó planificar la disposición de los componentes, la interconexión entre ellos y la integración de los elementos necesarios para lograr un funcionamiento coordinado y eficaz.

Hardware - Selección y Configuración.- Luego de establecer la estructura, se procedió a la selección cuidadosa de los componentes hardware necesarios para el prototipo. Se consideraron factores como especificaciones técnicas, costos, disponibilidad en el mercado y compatibilidad. Posteriormente, se configuraron estos elementos de acuerdo con las necesidades del diseño.

Software - Codificación del Algorítmico:.- Se dedicó una fase a la codificación del algoritmo fundamental que regiría el comportamiento del prototipo. Esto involucró la escritura y depuración de código para garantizar que el software estuviera en sintonía con los requerimientos establecidos, permitiendo así el funcionamiento adecuado del sistema.

Implementación del Prototipo.- Una vez establecido tanto el diseño estructural como el desarrollo del software, se procedió a la implementación física del prototipo. Se ensamblaron y conectaron los componentes siguiendo el diseño previamente establecido, garantizando la correcta integración y funcionamiento de cada parte del sistema.

Pruebas de Funcionalidad.- Finalmente, se llevaron a cabo pruebas exhaustivas para evaluar la funcionalidad y desempeño del prototipo. El análisis de resultados es fundamental para verificar si el prototipo cumple con los requerimientos definidos.

CAPITULO IV

4. Marco de análisis e interpretación de resultados

En este capítulo se detalla cómo se obtuvieron los resultados y el procedimiento para obtenerlos, cumpliendo con los requerimientos y condiciones necesarias del prototipo.

4.1 Pruebas de funcionalidad del prototipo de línea de ensamble.

La validación de la funcionalidad es una parte esencial en el proceso de desarrollo del prototipo ya que ayuda a identificar posibles problemas y áreas de mejora es decir que mediante pruebas de ensamblaje se obtiene la eficacia y la precisión del prototipo en términos de como realiza una función específica y cuánto tiempo lleva realizarla.

Antes de comenzar con las pruebas es importante definir los parámetros que se desean evaluar, entre estos se encuentra el tiempo requerido para realizar el ensamblaje de la caja, tiempo de desplazamiento desde que termino el proceso hasta el posicionamiento del robot cartesiano en el punto cero, precisión del prototipo y la eficacia del proceso de ensamble, tomando en cuenta el desface del tiempo por el porta placas de Arduino el cual retardara el ensamble dependiendo del número de Arduino a ensamblar. La Tabla 25-4 detalla las pruebas que se realizaron en el prototipo, se encuentra dividido entre 5 cajas con 10 corridas cada una, dando un total de 50 corridas, se registró el tiempo que le tomo al robot cartesiano en realizar el proceso individual de ensamble, el tiempo de desplazamiento desde que termino el proceso hasta el punto cero agregado a esto cada caja tienen una marca ya sea de verificación o de rechazo, las cajas que están verificadas significa que no tuvieron ningún inconveniente en el proceso y aquellas que por el contrario tienen una marca de rechazo tuvieron inconvenientes dentro del proceso al realizar el ensamble.

Tabla 25-4: Tiempo de ensamblaje por caja

PRUEBA DE ENSAMBLE				
ENSAMBLE	CORRIDA	TIEMPO	ENSAMBLE	DESPLAZAMIENTO
1	1	259.75	✓	27.56
	2	259.77	✓	27.71
	3	259.86	✓	29.41
	4	259.26	✓	26.95
	5	259.30	✓	30.02
	6	259.41	✓	30.37
	7	258.18	✓	28.2
	8	258.32	✓	28.11

	9	258.26	✓	30.51
	10	258.32	✓	29.25
2	1	263.79	✓	26.22
	2	263.28	✓	26.75
	3	262.85	✓	30.91
	4	263.39	✓	27.35
	5	263.48	✓	28.61
	6	263.54	✓	27.52
	7	263.64	x	27.2
	8	263.97	✓	24.88
	9	263.41	✓	28.1
	10	262.76	✓	29.64
3	1	266.85	✓	28.69
	2	265.39	✓	28.68
	3	265.32	x	28.53
	4	264.81	✓	29.8
	5	267.38	x	27.8
	6	266.71	✓	27.78
	7	266.35	✓	26.56
	8	267.79	✓	29.99
	9	267.10	✓	27.83
	10	264.36	✓	30.76
4	1	273.48	✓	29.45
	2	271.36	✓	28.48
	3	271.34	✓	27.94
	4	270.09	✓	30.31
	5	270.41	✓	29.89
	6	270.32	✓	29.05
	7	270.33	x	28.38
	8	270.36	✓	30.17
	9	270.32	✓	31.27
	10	270.82	✓	27.02
5	1	273.82	✓	29.37
	2	274.30	✓	27.59
	3	275.64	✓	29.74
	4	275.43	✓	27.82
	5	275.28	✓	30.09
	6	274.64	✓	27.28
	7	274.95	✓	27.77
	8	275.22	✓	26.93
	9	275.32	✓	27.68
	10	275.36	✓	27.06

Realizado por: Bonilla, H., 2023

Analizando los resultados de la Tabla 25-4 se determinó que de un total de 50 muestras que fueron ensambladas se obtuvo un resultado satisfactorio ya que 46 de ellas fueron ensambladas correctamente representado una tasa de éxito del 92% demostrando un alto nivel de precisión en el proceso de ensamble.

Durante el proceso se detectaron varias muestras las cuales no se ensamblaron correctamente, representando el 8%, aunque los resultados de las pruebas de ensamblaje son satisfactorios se reconoce la importancia de realizar un análisis de las 4 muestras restantes que representa un problema dentro del proceso brindando oportunidades de mejora y desarrollo de estrategias para evitar futuros errores.

Por ello se realizó un análisis exhaustivo de las muestras que representaron problemas identificando la causa general y el área de mejora en el prototipo, la falla común que tuvieron las 4 muestras se encuentra en el mal posicionamiento de la placa en la base debido al porta placas de Arduino, se detectó un pequeño desplazamiento de los soportes de las placas 2, 3 y 4 por ende el error solo se encuentra en las muestras tomadas de las mismas.

La Tabla 26-4 contiene información importante sobre el tiempo promedio de ensamble por caja, la cual es mayor al tiempo teórico ya que se incluyen interrupciones, desfaces de tiempos, retrasos y variaciones dentro del ensamble, agregado a esto se encuentra en la tabla la probabilidad de ensamblaje individual de las cajas. Esta información es crucial para comprender el rendimiento y la eficacia del proceso de ensamblaje en un contexto específico.

En la tabla se muestra el número de ensamble, la única peculiaridad que diferencia a los ensambles es en el apartado de las placas de Arduino, debido a que estas se encuentran montadas una sobre otra para su ensamble posterior, crean una diferencia de tiempos entre las mismas, aunque todas conservan las mismas piezas para su ensamble el tiempo promedio varía dependiendo el número de placa de Arduino uno a ensamblar llegando a ser más tardado ensamblar la quinta placa con respecto a la primera.

Tabla 26-4: Tabla resumen de eficacia

Tabla resumen			
N° de caja	Tiempo promedio por caja	Probabilidad	Desfase de tiempo
1	259.043	1	0
2	263.411	0.9	4.368
3	266.206	0.8	7.163
4	270.883	0.9	11.84
5	274.996	1	15.953
	Total	0.92	39.324

Realizado por: Bonilla, H., 2023

El prototipo de línea de ensamble tiene una eficacia del 92% al ensamblar 5 cajas seguidas sin parar. Esto quiere decir que es capaz de ensamblar correctamente el 92% de las cajas producidas, demostrando que el prototipo tiene una alta eficacia ya que solo un pequeño porcentaje representa

problemas durante el proceso. Para calcular la eficacia se identifica el número de cajas ensambladas correctamente, que es de 46. Luego, dividimos este número para el total de cajas ensambladas, que son 50 y multiplicamos por 100 para obtener la eficacia en porcentaje del prototipo

Existe una columna adicional en la tabla resumen la cual muestra como la disposición de las placas de Arduino en las bases crea una diferencia de tiempo de ensamble de cada caja con relación a la primera. Por ejemplo, el ensamble número 3 requiere de 7.16 segundos más con respecto a la primera, es importante tomar en cuenta esta diferencia de tiempos para poder planificar y optimizar el ensamble ya puede afectar la producción y la eficacia de la línea. El tiempo promedio perdido es de 39.32 segundos

4.2 Consumo del prototipo

El consumo dentro del prototipo juega un papel fundamental en la evaluación y optimización, entender el consumo de energía garantiza su eficacia y la viabilidad a largo plazo.

4.2.1. Consumo individual

Se explora y analiza el consumo de energía de los actuadores a profundidad, usando una fuente la cual es un suministro de energía programable de corriente continua ajustable a los diferentes componentes y dispositivos del prototipo de línea de ensamble, tiene un total de 3 salidas de corriente que se puede configurar de manera independiente, se utilizan dos salidas las cuales están denominadas como CH1 Y CH2.

La Tabla 27-4 muestra el consumo individual de los motores nema 23 los cuales se utilizaron en el eje x, y, z respectivamente. El voltaje y el amperaje con el cual trabaja el canal uno es el mismo con el que trabaja el canal número 2 sin embargo se hace el uso individual de los mismos para obtener mejor información del consumo y obtener graficas de consumo general mucho más detalladas.

Tabla 27-4: Consumo individual CH1

CH1	V	A	W
Seteo	24	3	72.00
Eje x	24	1.5	36.00
Eje y	24	2.6	62.40
Eje z	24	1.5	36.00

Realizado por: Bonilla, H., 2023

La Tabla 28-4 muestra el consumo individual del canal número 2 en el cual se encuentra conectado el motor del módulo de desplazamiento, gripper y la electroválvula con su respectivo consumo.

Tabla 28-4: Consumo individual CH2

CH2	V	A	W
Seteo	24	3	72.0
M. Desplazamiento	24	1.75	42.0
Gripper	24	1	24.0
Electroválvula	24	0.3	7.2

Realizado por: Bonilla, H., 2023

4.2.2. Consumo general

Es esencial medir y comprender el consumo energético general del prototipo de línea de ensamble, para lograrlo de manera precisa y completa, se llevó a cabo un proceso de toma de mediciones del amperaje que involucro la recopilación de datos a lo largo del tiempo de ensamble. La Ilustración 71-4 muestra la gráfica que se obtuvo de la toma de mediciones de amperaje del canal número uno en el transcurso de 273 segundos.

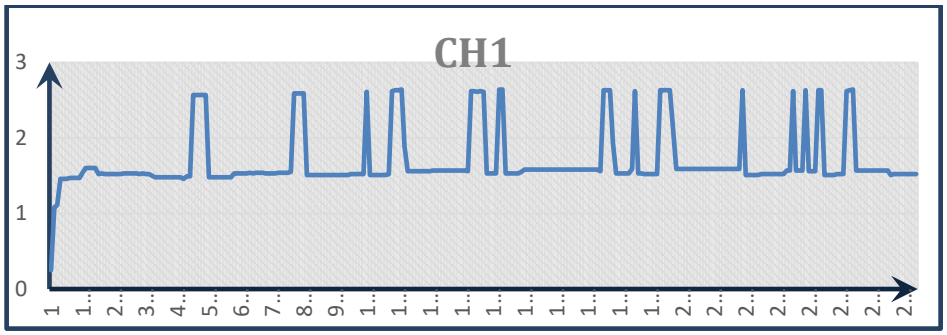


Ilustración 71-4: Consumo recopilado CH1
Realizado por: Bonilla, H., 2023

La Ilustración 72-4 fue tomado de la fuente de poder, comparándolo con la gráfica anterior podemos notar la similitud y precisión de los datos tomados para la realización de la misma, el actuador que tiene una mayor exigencia de amperaje es el motor paso a paso nema 23 utilizado en el eje Y el cual demanda un amperaje promedio de 2.6 A, esto se debe a su configuración, mientras que los dos motores restantes se mantiene en un amperaje similar el cual se encuentra en 1.5 A.

Ilustración 72-4: Consumo generado por la fuente CH1
Realizado por: Bonilla, H., 2023

La Ilustración 73-4 muestra el consumo recopilado de la segunda salida de la fuente de poder también conocida como CH2, la variación más alta es debido al módulo de desplazamiento el cual tiene una exigencia aproximada de 1.75 A, el gripper mantiene una exigencia constante de 1 A, en la gráfica podemos notar una variación a partir del segundo 103 debido a la activación de la electroválvula la cual exige un consumo constante de 0.3 A desde su activación para desactivar el flujo de aire mediante un mecanismo electromagnético.

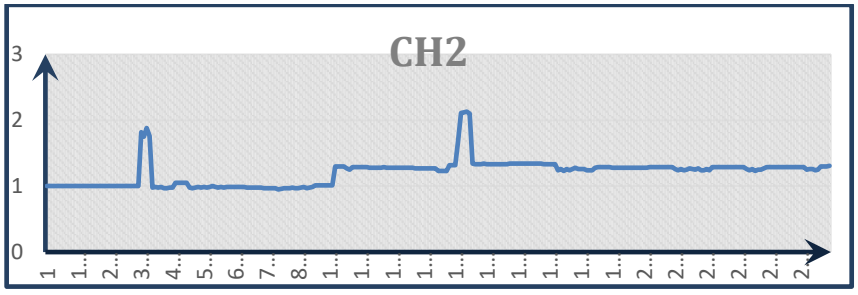


Ilustración 73-4: Consumo recopilado CH2
Realizado por: Bonilla, H., 2023

Una manera para comprobar que la toma de datos de la segunda salida de la fuente de poder es correcta, es mediante la comparación con la gráfica arrojada por la fuente la misma que se muestra en la Ilustración 74-4 podemos notar la similitud en su comportamiento y en las diferentes exigencias de los actuadores que se encuentran conectados a la segunda salida CH2.

Ilustración 74-4: Consumo generado por la fuente CH2
Realizado por: Bonilla, H., 2023

CAPITULO V

5. EVALUACIÓN ECONÓMICA

La evaluación económica es un apartado importante en el análisis y sección de conclusiones que tiene como objetivo evaluar los aspectos financieros y económicos relacionados con el prototipo proporcionando una visión clara de la vialidad económica del proyecto y permite tomar decisiones informadas sobre su implementación o aplicación en la vida real.

5.1 Análisis de costos de equipos

En la Tabla 29-5, se muestra el desglose completo de los costos asociados con los equipos utilizados en la creación del prototipo de líneas de ensamble.

Tabla 29-5: Costos de equipos del prototipo de línea de ensamble

#	Cantidad	Medida	Descripción	Costo Unitario	Costo Total
1	2	un	Motor NEMA 17	$18.75	$37.50
2	3	un	Motor NEMA 23 57BYGH76	$49.11	$147.33
3	2	un	TB6600	$14.73	$29.46
4	3	un	Driver DM542	$28.57	$85.71
5	1	un	Arduino mega 2560	$27.00	$27.00
6	1	m	Cilindro neumático JELPC MSA 25X100-S	$55.00	$55.00
7	1	un	Electroválvula	$35.00	$35.00
				Total	$417.00

Realizado por: Bonilla, H., 2023

5.2 Análisis de costos de materiales

La Tabla 30-*5* proporciona un análisis detallado de los materiales empleados en la elaboración del prototipo de línea de ensamble.

Tabla 30-5: Costos de materiales del prototipo de línea de ensamble

#	Cantidad	Medida	Descripción	Costo Unitario	Costo Total
1	7.1	m	Perfil de Aluminio V 20x40 serie 20 Black	$16.70	$118.57
2	1.07	m	Perfil de Aluminio V 20x20 serie 20 Black	$8.72	$9.33

3	4	un	Set V2	$23.71	$94.84
4	16	un	Soporte de esquina serie 20 Negros reforzados	$1.00	$16.00
5	8	un	Soporte universal en L doble serie 20	$1.83	$14.64
6	8	un	Soporte universal en L simple serie 20	$1.00	$8.00
7	94	un	Tuerca T simple serie 20	$0.09	$8.46
8	126	un	Perno Allen cabeza plana M5x08	$0.09	$11.34
9	5	m	Banda GT2 6mm con hilo de acero	$4.49	$22.45
10	3	un	Placa soporte motor Nema 23	$8.48	$25.44
11	1	un	Polea GT2-6mm 32 dientes D-5	$8.00	$8.00
12	3	un	Polea GT2-6mm 20 dientes D-8	$2.46	$7.38
13	40	un	Tuerca T rectangular M5 serie 20	$0.18	$7.20
14	10	un	Perno Allen cabeza plana M5x10	$0.09	$0.90
15	10	un	Perno Allen cabeza plana M5x15	$0.09	$0.90
16	10	un	Perno Allen cabeza plana M5x25	$0.09	$0.90
17	10	un	Perno Allen cabeza plana M5x35	$0.13	$1.30
18	10	un	Perno Allen cabeza plana M5x40	$0.13	$1.30
19	6	un	Placa de unión de dos agujeros serie 20	$0.71	$4.26
20	6	un	Placa de unión de tres agujeros serie 20	$0.98	$5.88
21	7	un	Polea con rodamientos Delrin	$2.68	$18.76
22	10	un	Tuerca T rectangular M5 serie 20	$0.18	$1.80
23	8	un	Resistencia 10 ohmios	$0.10	$0.80
24	1	un	Cables para Arduino macho a macho	$3.50	$3.50
25	1	un	Cables para Arduino macho a hembra	$3.50	$3.50
26	24	m	Cables AWG 22	$0.35	$8.40
27	145	horas	Piezas 3D	$1.00	$145.00
28	5	un	Cable conector de motor paso a paso	$2.25	$11.25
29	10	un	Final de carrera	$0.50	$5.00
30	8	m	Estaño 0.5 mm	$0.45	$3.60

31	5	un	Termo encogible 6 mm	$0.45	$2.25
32	3	un	Termo encogible 4 mm	$0.40	$1.20
33	2	un	Termo encogible 2 mm	$0.35	$0.70
34	2	m	Manguera neumática	$3.50	$7.00
35	4	un	Perno allen avellanado M3X06 inox	$0.07	$0.28
36	1.5	m	Cadena porta cables	$13.39	$20.09
37	2	un	Acople para cilindro neumático	$2.00	$4.00
				Total	$604.22

Realizado por: Bonilla, H., 2023

5.2.1. Costo total del prototipo de línea de ensamble de partes plásticas.

Después de evaluar los gastos asociados con los equipos y materiales, tal como se detalla en la Tabla 31-5, se presenta un costo global de $1021.22.

Tabla 31-5: Costos totales del prototipo de línea de ensamble

#	Cantidad	Descripción	Costo Unitario	Costo Total
1	1	Costos de equipos	$417.00	$417.00
2	1	Costos de materiales	$604.22	$604.22
			Total	$1 021.22

Realizado por: Bonilla, H., 2023

CONCLUSIONES

- Mediante el análisis exhaustivo del arte, se logró detallar y definir con precisión las características que debe cumplir el prototipo de línea de ensamble para satisfacer las exigencias del proyecto. Esta investigación permitió la unificación de todas las particularidades del prototipo, lo que a su vez facilitó la identificación de los elementos clave que deben integrarse en el mismo las cuales fueron la incorporación de un gripper, robot cartesiano tipo pórtico, microcontrolador.
- Se selecciono el software SolidWorks como herramienta CAD para modelar y validar el diseño del sistema robótico para la línea de ensamble, por su versatilidad y amplio conjunto de funcionalidades avanzadas. Su interfaz intuitiva y amigable facilita la adopción y el uso eficaz, permitiendo un diseño tridimensional detallado, análisis estructural, simulación de movimiento y generación de planos técnicos. Además, la compatibilidad con estándares industriales, su capacidad de simulación y el apoyo de una licencia proporcionada por la ESPOCH para todos los estudiantes brindan un entorno propicio para el desarrollo exitoso del sistema robótico, lo cual brindo un diseño y validación precisa.
- Se optó por los motores paso a paso Nema 23 y 17 debido a su alta eficacia y precisión. La elección de la tarjeta Arduino Mega fue fundamental para conectar todos los pines necesarios y garantizar el óptimo funcionamiento del prototipo. Además, se incorporó una fuente con tres canales de salida para mantener un suministro eléctrico constante. Para el módulo dosificador de bases y tapas, se integró un pistón neumático de simple efecto, junto con una electroválvula para controlar el pistón. Asimismo, se emplearon los drivers TB6600 y DM542 para lograr un control preciso de los motores Nema 17 y Nema 23, respectivamente. En cuanto al software, se seleccionó ARDUINO IDE, un entorno de desarrollo integrado que facilita la programación y carga de código en microcontroladores Arduino, utilizando el lenguaje C++. Visual Studio fue cuidadosamente seleccionado para la creación de la interfaz gráfica en este proyecto debido a su poderoso conjunto de herramientas integradas, diseñadas para facilitar el desarrollo de interfaces atractivas e intuitivas.
- Mediante el enfoque práctico y experimental, se llevaron a cabo un total de 50 pruebas de ensamble, logrando un éxito del 92%. El 8% restante de los ensambles resultaron fallidos al intentar colocar el Arduino en la base. Tras una investigación minuciosa, se determinó que el problema radica en el soporte para placas de Arduino. Específicamente, los soportes numerados 2, 3 y 4 presentan una leve inclinación debido a imperfecciones en la impresión 3D

- Se logro implementar un prototipo de línea de ensamble de partes plásticas basado en un sistema robótico cartesiano mismo que resulto exitoso por su alta eficacia, con una tasa de ensamblaje del 92%. Esta eficacia destaca la viabilidad y utilidad del enfoque robótico en la automatización de procesos de ensamble, lo puede resultar en ahorro recursos humanos para la producción industrial.

RECOMENDACIONES

- Se recomienda considerar, investigar y evaluar diferentes materiales para los componentes impresos en 3D y para las partes que componen el prototipo en general como el módulo de desplazamiento, gripper, dispensador de bases-tapas y soporte de placas para Arduino. Explorar si el cambio en el diseño o en los materiales podrían reducir los costos de fabricación, mejorar la durabilidad o incluso aumentar la eficacia del proceso de ensamble.

- Se recomienda que en caso de ensamblar componentes diferentes a los detallados se deberá investigar como seria la adaptación de estos en el prototipo de línea de ensamble, existen soportes los cuales están diseñados con la finalidad de adaptarse a otros componentes impresos en 3D mientras se consideren las dimensiones en las que encajaran las piezas.

- Con el fin de fortalecer aún más la operatividad y el control del proceso de ensamblaje, es recomendable avanzar en la mejora de la interfaz gráfica del prototipo. Esto podría incluir la implementación de alertas visuales en caso de problemas o fallos, así como la posibilidad de realizar ajustes en tiempo real.

- Dado que el prototipo implementado ha demostrado la eficacia de utilizar equipos fácilmente sustituibles, se recomienda encarecidamente que esta práctica sea adoptada en futuros proyectos de automatización y robótica industrial. Es vital priorizar la elección de componentes y dispositivos que no solo cumplan con las necesidades operativas específicas, sino que también estén ampliamente disponibles en el mercado

BIBLIOGRAFIA

MACHADO, J. Automatización de los procesos Productivos en la planta II División Partes y Piezas para la Empresa Indurama S.A[En línea]. (Tesis de maestría). Universidad de Cuenca. Cuenca. 2009. pp. 1-195. Consulta: [24 de enero de 2023]. Disponible en: http://efaidnbmnnnibpcajpcglclefindmkaj/http://dspace.ucuenca.edu.ec/bitstream/123456789/2627/1/tm4288.pdf

ALDAZ, M. Diseño y construcción de un prototipo de robot cartesiano asistente de pintura[En línea]. (Trabajo de Titulación) (Tesis de pregrado). Universidad Tecnológica Equinoccial. Quito. 2018. pp. 1-60. Consulta: [25 de enero de 2023]. Disponible en: http://repositorio.ute.edu.ec/bitstream/123456789/18147/1/70341_1.pdf

ALMETH, N. *MORFOLOGÍA*. S.l.: ISA. 2019. p.5.

ARMAS, J.; & CERDA, A. Implementación de un robot móvil para desplazamientos en ambientes no estructurados empleando visión artificial [En línea]. (Trabajo de Titulación) (Tesis de pregrado). Escuela Superior Politécnica de Chimborazo. Riobamba-Ecuador. 2020. pp. 1-110. [Consulta: 7 febrero 2023]. Disponible en: http://dspace.espoch.edu.ec/bitstream/123456789/16378/1/108T0331.pdf

ASSEMBLER. Tipos de lenguaje de programación. *Institute of Technology* [En línea]. [Consulta: 7 febrero 2023]. Disponible en: https://assemblerinstitute.com/blog/tipos-lenguaje-programacion/

CAD. *Subensamblajes*. S.l.: s.n. 2020. p.5.

CAMARILLO, A. ¿Qué es el puerto JTAG? *Noticias en 330* [En línea]. 2021. [Consulta: 7 febrero 2023]. Disponible en: https://blog.330ohms.com/2021/06/15/que-es-el-puerto-jtag/#:~:text=JTAG%20es%20una%20interfaz%20de,de%20circuito%20impreso%20(PCB).

CARLETTI, E. Motores paso a paso: características básicas. *Robots didácticos* [En línea]. 2022. [Consulta: 7 febrero 2023]. Disponible en: https://robots-argentina.com.ar/didactica/motores-paso-a-paso-caracteristicas-basicas/

CARTER FRÍO, Ramiro Rodrigo. Diseño y construcción de un robot paralelo tipo delta para mejora de la productividad en las industrias. [En línea] (Trabajo de Titulación). (Titulación) Universidad Antonio Nariño, La Guajira, Colombia. 2018. p. 5. [Consulta: 2023-01-27]. Disponible en: http://repositorio.uan.edu.co/bitstream/123456789/2581/10/2020JoséAntonioCastañedaJaramillo.pdf

CEUPE. ¿Qué es un Producto terminado? Concepto, características y ejemplos. *CEUPE* [En línea]. 2020. Consulta: [25 de enero de 2023]. Disponible en: https://www.ceupe.com/blog/producto-terminado.html

CHÁVEZ HERNÁNDEZ, Pedro A. Diseño y Desarrollo de un Robot de Inspección de Tuberías [En línea] (Trabajo de Titulación). (Titulación) Universidad Politécnica de Madrid, Madrid, España. 2018. p. 1. [Consulta: 2023-01-27]. Disponible en: https://oa.upm.es/14776/1/MARIA_ALEJANDRA_URDANETA_LIMA.pdf

COMMOMS. *Morfología de los robots industriales*. S.l.: s.n. 2021. p.6.

CONCEPTO ABC. Fuente de poder. [En línea]. 2022. [Consulta: 7 febrero 2023]. Disponible en: https://conceptoabc.com/fuente-de-poder/#Caracteristicas

CRUZ, E.; et al. *Diseño y construcción de brazo robótico para línea de producción entrenadora* [En línea]. 2020. pp. 1-8. Consulta: [27 de enero de 2023]. Disponible en: http://somim.org.mx/memorias/memorias2020/articulos/A1_67.pdf

EDULAB. *Glosario de términos y conceptos robóticos*. España: S.n. 2018. pp.1-24.

ELINDAR. Drivers para motores paso a paso. [En línea]. 2018. [Consulta: 7 febrero 2023]. Disponible en: http://www.elindar.com.ar/ver-drivers-pap.html

ESCUDERO ARROYO, Natalia Ana. Robot delta para cultivo en hileras [En línea] (Trabajo de Titulación). (Titulación) Universidad Autónoma de Occidente, Cali, Colombia. 2019. p. 4. [Consulta: 2023-01-27]. Disponible en: https://red.uao.edu.co/bitstream/handle/10614/9708/T07376.pdf?sequence=1&isAllowed=y

EUZCATEGUIE, J. Robótica. *Ferrovial* [En línea]. 2018. Consulta: [25 de enero de 2023]. Disponible en:

https://www.ferrovial.com/es/innovacion/tecnologias/robotica/#:~:text=El%20término%20robótica%20fue%20acuñado,aliviándolo%20de%20las%20labores%20caseras.

FERRO. Subensamble: en la industria metalmecánica. *Industrial Ferro* [En línea]. 2021. Consulta: [25 de enero de 2023]. Disponible en: https://industrialferro.com.mx/subensamble-en-la-industria-metalmecanica/

GANDHI, M. Línea de ensamble: cómo beneficia los procesos de manufactura. *Atycom* [En línea]. 2022. Consulta: [25 de enero de 2023]. Disponible en: https://www.autycom.com/linea-de-ensamble/#:~:text=¿Qué%20es%20una%20línea%20de,hasta%20tener%20el%20ensamble%20final.

GÓMEZ, J. "Interfaz de potencia de estado sólido y relevadores, controlada mediante dispositivos móviles". *Revista Ingienantes* [En línea]. 2016, 1(2). pp.45-46. . Consulta: [27 de enero de 2023]. Disponible en: https://citt.itsm.edu.mx/ingeniantes/articulos/ingeniantes3no2vol1/7.%20Interfaz%20de%20potencia%20de%20estado%20sólido%20y%20relevadores%20controlada%20mediante%20dispositivos%20móviles.pdf

GONZALES, H.; & CARRILLO , M. Implementación de un prototipo de robot sembrador de papa en terrenos sin inclinación para pequeños productores [En línea]. (Trabajo de Titulación) (Tesis de pregrado). Escuela Superior Politécnica de Chimborazo. Riobamba-Ecuador. 2019. pp. 1-102. [Consulta: 7 febrero 2023]. Disponible en: http://dspace.espoch.edu.ec/bitstream/123456789/13686/1/108T0313.pdf

JARA, M.; & PACHECO, M. Diseño e implementación de un robot cartesiano, para el montaje de tapa y/o pasador, en el proceso de paletizado [En línea]. (Trabajo de Titulación) (Tesis de pregrado). Escuela Superior Politécnica de Chimborazo. Riobamba. 2013. pp. 1-211. Consulta: [25 de enero de 2023]. Disponible en: http://dspace.espoch.edu.ec/handle/123456789/3265

LEAL, A.; & PÉREZ, J. Control de posición en el espacio articulado para el robot cartesiano del GIESCV [En línea]. (Trabajo de Titulación) (Tesis de pregrado). Universidad El Bosque, Facultad de Ingeniería Electrónica. 2022. pp. 1-173. Consulta: [25 de enero de 2023]. Disponible en:https://repositorio.unbosque.edu.co/bitstream/handle/20.500.12495/9449/Documento%20de%20Proyecto%20Grado%20AI-036-08_09.pdf?sequence=1&isAllowed=y

LÓPEZ. J.; et al. Robot cartesiano: seguimiento de trayectorias irregulares arbitrarias mediante computadora [En línea]. (Trabajo de Titulación) (Tesis de pregrado). Universidad Autónoma Del Estado De Hidalgo. México. 2017. pp. 1-171. Consulta: [27 de enero de 2023]. Disponible en: https://www.uaeh.edu.mx/docencia/Tesis/icbi/licenciatura/documentos/Robot%20cartesiano%20seguimiento%20de%20trayectorias.pdf

COBATAB. *Robótica móvil* [en línea]. Tabasco-México: Colegio de Bachilleres de Tabasco. 2022. [Consulta: 18 septiembre 2023]. Disponible en: https://www.cobatab.edu.mx/servalumnos/GuiasCap/6to/ROBoTICA.pdf

MUÑOZ, David. Robots cartesianos: qué son y cómo funcionan. *Tecnología del plástico* [en línea]. [Consulta: 18 septiembre 2023]. Disponible en: https://www.plastico.com/es/noticias/robots-cartesianos-que-son-y-como-funcionan

MARTÍNEZ, A. La eficiencia en la industria. *Predictiva* [En línea]. 2020. Consulta: [25 de enero de 2023]. Disponible en: https://predictiva21.com/eficiencia/

MECAFENIX. ¿Qué es un final de carrera? *Ingeniería Mecafenix*. 2021. [Consulta: 7 febrero 2023]. Disponible en: https://www.ingmecafenix.com/electronica/final-de-carrera/

MENDOZA SIGCHA, Ángel Wilson. Robot delta para cultivo en hileras [En línea] (Trabajo de Titulación). (Titulación) Universidad Politécnica de Madrid, Madrid, España. 2019. p. 3. [Consulta: 2023-01-27]. Disponible en: https://oa.upm.es/69186/2/TFG_VICTOR_ORIHUEL_ARRIBAS.pdf

MICRO. Pistones neumáticos. *Micro automación* [En línea]. 2019. [Consulta: 7 febrero 2023]. (Disponible en: https://ar.microautomacion.com/es/que-es-un-cilindro-neumatico/

MIRANDA, S. *Productividad.* S.l.: s.n. 2018. p.65.

MORENO LÓPEZ, Gabriela E.; & CHALA CHOTA, Jessica Abigail. Robot delta para cultivo en hileras [En línea] (Trabajo de Titulación). (Titulación) Universidad Central del Ecuador, Quito, Ecuador. 2018. p. 7. [Consulta: 2023-01-27]. Disponible en: https://ebin.pub/information-and-communication-technologies-8th-conference-ticec-2020-guayaquil-ecuador-november-2527-2020-proceedings-1st-ed-9783030628321-9783030628338.html

ORTIZ, A. Driver de motor – Electrónica de potencia. *DADEMUCH* [En línea]. 2018. Consulta: 7 febrero 2023]. Disponible en: https://dademuch.com/2018/04/26/driver-de-motor-dc-electronica-de-potencia/

PALMA, C; & RODRÍGUEZ, S. *Tarjetas de Desarrollo: Herramientas para el diseño*[En línea]. .S.l.: Escuela Tecnológica Instituto Técnico Central. 2011. p.3. Consulta: [27 de enero de 2023]. Disponible en: https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwjIiYWGiPv8AhVPSTABHXvsBkkQFnoECBIQAQ&url=https%3A%2F%2Frevistas.itc.edu.co%2Findex.php%2Fletras%2Farticle%2Fdownload%2F104%2F100&usg=AOvVaw3OT-orf6BkOSwcJh3Nr0AY

PAREDES, P. Tiempo de proceso. *Organización de la producción* [En línea]. 2019. Consulta: [25 de enero de 2023]. Disponible en: http://www.organizaciondelaproduccion.com/tiempo-proceso.php

PARRA, P. *Diseño, construcción y evaluación de un robot cartesiano XYZ electroneumático*.' Valencia: Universidad Politécnica de Valencia, 2017. pp. 1-35.

PAZMIÑO, D. Estudio, y selección de elementos básicos de un módulo mecatrónico para diagnosticar, ejecutar y evaluar procesos electroneumáticos en el laboratorio de automatización industrial de la FICM. Trabajo de Titulación) (Tesis de pregrado). Universidad Técnica de Ambato. Ambato-Ecuador. 2012. pp. 1-121. [Consulta: 7 febrero 2023]. Disponible en: https://repositorio.uta.edu.ec/bitstream/123456789/2153/1/Tesis%20I.%20M.%20144%20-%20Pazmiño%20Beltrán%20Diego%20Alejandro.pdf

PETROVÍCHI, L. Isaac Asimov, creador de las tres leyes de la robótica y fundador del imperio galáctico. *La voz de Galicia* [En línea]. 2020. Consulta: [25 de enero de 2023]. Disponible en: https://www.lavozdegalicia.es/noticia/cultura/2020/01/03/asimov-creador-tres-leyes-robotica-fundador-imperio-galactico/0003_202001H3P29991.htm#:~:text=Isaac%20Asimov%2C%20creador%20de%20las,y%20fundador%20del%20imperio%20galáctico

PRISMA. ¿Qué es el Diseño Mecánico CAD? *PRISMACIM* [En línea]. 2021. Consulta: [25 de enero de 2023]. Disponible en: https://prismacim.com/diseno-mecanico/#:~:text=El%20diseño%20mecánico%20consiste%20en,las%20que%20han%20sido%20concebidos.

RAMÓN, A. Diseño y construcción de un prototipo de incubadora con supervisión inteligente para la eclosión de huevos [En línea]. (Trabajo de Titulación) (Tesis de pregrado). Escuela Superior Politécnica de Chimborazo. Riobamba-Ecuador. 2019. pp. 1-112. [Consulta: 7 febrero 2023]. Disponible en: http://dspace.espoch.edu.ec/bitstream/123456789/13693/1/108T0315.pdf

RICAURTE, A.; & MINDA, I. Diseño e implementación de un brazo robótico industrial con 5 grados de libertad guiado por Kinect [En línea]. (Trabajo de Titulación) (Tesis de pregrado). Escuela Superior Politécnica de Chimborazo. Riobamba-Ecuador. 2017. pp. 1-124. [Consulta: 7 febrero 2023]. Disponible en: http://dspace.espoch.edu.ec/bitstream/123456789/8957/1/108T0221.pdf

RINCÓN, F.; & VESGA, J. Diseño, construcción y puesta en marcha de un prototipo robotizado cartesiano para prácticas de procesos industriales de almacenamiento y ensamble [En línea]. (Trabajo de Titulación) (Tesis de pregrado). Universidad Pontificia Bolivariana. Bucaramanga. 2019. pp. 1-169. Consulta: [27 de enero de 2023]. Disponible en: https://repository.upb.edu.co/bitstream/handle/20.500.11912/1047/digital_18513.pdf?sequence=1&isAllowed=y

ROCHA, A. *Fuente de poder*. 2°ed. S.l.: s.n. 2019, pp. 1-72

RODRÍGUEZ, O. *Ensambles* [En línea]. México: Universidad Nacional Autónoma de México. 2021. pp. 1-18. Disponible en: http://132.248.9.195/ptd2013/enero/0688067/0688067_A12.pdf

SDI. Fuente de poder. [En línea]. 2022. [Consulta: 7 febrero 2023]. Disponible en: https://sdindustrial.com.mx/blog/fuentes-de-poder/#:~:text=Las%20fuentes%20de%20poder%20sirven,dispositivos%20que%20se%20conectan%20por

SDIN. Introducción a los actuadores eléctricos: Motores eléctricos. *SDI* [En línea]. 2020. Consulta: [25 de enero de 2023]. Disponible en: https://sdindustrial.com.mx/blog/introduccion-a-los-actuadores-electricos-motores-electricos/#:~:text=Los%20actuadores%20eléctricos%20son%20un,en%20comparación%20con%20otros%20actuadores.

SEGOVIA MEDINA, Jonathan Eduardo. Construcción del prototipo e implementación de un sistema de control de un robot delta para el laboratorio de mps de la universidad politécnica

salesiana [En línea] (Trabajo de Titulación). (Maestría) Universidad Politécnica Salesiana, Cuenca; Azuay, Ecuador. 2018. p. 6. [Consulta: 2023-01-27]. Disponible en: https://dspace.ups.edu.ec/bitstream/123456789/23497/1/UPS-CT010096.pdf

SERRANO,A. *Control y robótica.* 4° ed. S.l.: s.n. 2018, pp. 1-46

SIEMENS. Diseño asistido por ordenado (CAD). *SIEMENS* [En línea]. 2021. Consulta: [25 de enero de 2023]. Disponible en: https://www.plm.automation.siemens.com/global/es/our-story/glossary/computer-aided-design-cad/12507

SIGNIFICADOS. Hardware. *Significados.com.* [En línea]. 2021. Consulta: [25 de enero de 2023]. Disponible en: https://www.significados.com/hardware/

ZAPATA, C. *Fundamentos de programación, Guía de auto enseñanza* [En línea]. Colombia: Alfaomega Ra-Ma. 2006. pp. 1-298. [Consulta: 7 febrero 2023]. Disponible en: https://ridum.umanizales.edu.co/jspui/bitstream/20.500.12746/202/2/180_Zapata_Ospina_Carlos_Andres_2006%20file2%20libro.pdf

ZIKODRIVE. Motor de control de pasos. [En línea]. 2019. [Consulta: 7 febrero 2023]. Disponible en: https://www.zikodrive.com/es/ufaqs/principal-diferencias-stepper-motor-driver-stepper-motor-controller/#:~:text=Un%20controlador%20de%20motor%20de%20pasos%20es%20un%20componente%20electrónico,control)%20el%20motor%20de%20pasos.

ZIKODRIVE. *Marco de motor NEMA y especificaciones típicas para motores paso a paso* [En línea]. 2018. Consulta: [13 de febrero de 2023]. Disponible en: https://www.zikodrive.com/es/ufaqs/nema-motor-marco-tamanos-media/

I2C ELECTRONIC. *Motor paso a paso nema 23 57BYGH76-3004A* [En línea]. 2021. Consulta: [13 de febrero de 2023]. Disponible en: http://i2celectronica.com/paso-a-paso/841-motor-paso-a-paso-nema-23.html

HETPRO. *Motor a pasos NEMA 17HS3001-20B* [En línea]. 2022. Consulta: [13 de febrero de 2023]. Disponible en: https://hetpro-store.com/motor-a-pasos-nema-17hs8401-78-oz-in/?_gl=1*a99kee*_ga*LUM2NHc3NnlUZVBYSmdEYmllR3hucWU1alpPcWs5Rzl6ZjVOWlpTVDk0clF0SE9KaF9vOFQxbm9YdjdpWkhUaw.

UNIT ELECTRONICS. *DM542 Driver para motora pasos DC20-50V 4.2A* [En línea]. 2022. Consulta: [13 de febrero de 2023]. Disponible en: https://uelectronics.com/producto/dm542-driver-para-motor-a-pasos-dc-20-50v-4-2a/

STEPPERONLINE. *Driver para motores paso a paso digital serie Y 1.7-5.6ª DC20V-50V para motor paso a paso NEMA 23, 24, 34* [En línea]. 2021. Consulta: [13 de febrero de 2023]. Disponible en: https://www.omc-stepperonline.com/es/controlador-paso-a-paso-digital-serie-y-1-7-5-6a-dc20v-50v-para-motor-paso-a-paso-nema-23-24-34-dm556y

LUIS LLAMAS. *Motores paso a paso con Arduino y Driver A4988 o DRV8825* [En línea]. 2022. Consulta: [13 de febrero de 2023]. Disponible en: https://www.luisllamas.es/motores-paso-paso-arduino-driver-a4988-drv8825/

SIGLENT. *SPD3303X/SPD3303X-E Series Programmable DC Power Supplies* [En línea]. 2021. Consulta: [13 de febrero de 2023]. Disponible en: https://siglentna.com/power-supplies/spd3303x-spd3303x-e-series-programmable-dc-power-supply/

JELPC. *MA Series Mini Stainless Steel Cylinder* [En línea]. 2022. Consulta: [13 de febrero de 2023]. Disponible en: https://www.jelpc.com/product/cylinder/mini-cylinder/ma-series-stainless-steel.html

LUIS DEL VALLE. *Arduino Mega 2560, ArduinoUno y Arduino Leonardo* [En línea]. 2021. Consulta: [13 de febrero de 2023]. Disponible en: https://programarfacil.com/blog/arduino-blog/arduino-mega-2560/

TRIMATEC. *Airtac 2VC250-25: 2 Way Solenoid Valve – 2V25025BIT* [En línea]. 2022. Consulta: [13 de febrero de 2023]. Disponible en: https://trimantec.com/products/airtac-2v250-solenoid-fluid-control-valve-22-way-2v25025bit

ROME CO INDUSTRIAL. *3V1-06 Electroválvula neumática 3/2 puertos 1/8" NC compacta* [En línea]. 2021. Consulta: [13 de febrero de 2023]. Disponible en: https: https://www.romecoindustrial.com/producto/3v1-06-electrovalvula-neumatica-3-2-puertos-1-8%E2%80%B3-nc-compacta/

AV ELETRONICS. *Fin de Carrera KW11-3Z* [En línea]. 2022. Consulta: [13 de febrero de 2023]. Disponible en: https://avelectronics.cc/producto/fin-de-carrera-kw11-3z-5a-250v/

LIQCREATE. *Comparación de las propiedades de loes plásticos de impresión 3D FDM, SLS y resina* [En línea]. 2022. Consulta: [13 de febrero de 2023]. Disponible en: https://www.liqcreate.com/es/art%C3%ADculos-de-apoyo/propiedades-resina-fdm-sls/

ANEXOS

ANEXO A: PROGRAMACIÓN DE ARDUINO PARA EL PROTOTIPO DE LÍNEA DE ENSAMBLE

```
String inputString = "";
bool stringComplete = false;

int PULUNO=14;
int DIRUNO=15;
int PULDOS=16;
int DIRDOS=17;
int PULTRES=18;
int DIRTRES=19;
int PULCUATRO=20;
int DIRCUATRO=21;
int PULCINCO=22;
int DIRCINCO=23;
int cont = 0;
int PISTON = 3;
int TRIG = 4;
int ECO = 5;
int DISTANCIA;
int DURACION;
int BINICIO=8;


void setup() {
  pinMode (PULUNO, OUTPUT);
  pinMode (DIRUNO, OUTPUT);
  pinMode (PULDOS, OUTPUT);
  pinMode (DIRDOS, OUTPUT);
  pinMode (PULTRES, OUTPUT);
  pinMode (DIRTRES, OUTPUT);
  pinMode (PULCUATRO, OUTPUT);
  pinMode (DIRCUATRO, OUTPUT);
  pinMode (PULCINCO, OUTPUT);
  pinMode (DIRCINCO, OUTPUT);
  pinMode (PISTON, OUTPUT);
  pinMode (8, INPUT);
  pinMode (9, INPUT);
  pinMode (10, INPUT);
  pinMode (11, INPUT);
  pinMode (12, INPUT);
  pinMode (6, INPUT);
  pinMode (7, INPUT);
  pinMode (24, INPUT);
  pinMode (25, INPUT);
  pinMode (TRIG, OUTPUT);
  pinMode (ECO, INPUT);
  Serial.begin(9600);
}

void loop() {
 digitalWrite(TRIG, HIGH);
 delay(1);
 digitalWrite(TRIG, LOW);
 DURACION = pulseIn(ECO, HIGH);
 DISTANCIA = DURACION / 58.2;
```

```
 Serial.println(DISTANCIA);
 delay(200);

 if (stringComplete)
 {
  inputString.trim();
  Serial.println(inputString);

 if (inputString.equals("BINICIO"))
   {
    if((digitalRead(6) == LOW) && (digitalRead(7) == LOW)){
    digitalWrite(PISTON, HIGH);
    delay(5000);
    digitalWrite(PISTON, LOW);
    }
    while (digitalRead(25) == LOW){
    digitalWrite(DIRCUATRO,0);
    digitalWrite(PULCUATRO,HIGH);
    delayMicroseconds(4000);
    digitalWrite(PULCUATRO,LOW);
    delayMicroseconds(4000);}
     // griper
     if(digitalRead(25) == HIGH){
     while (digitalRead(9) == LOW){
     digitalWrite(DIRCINCO,0);
     digitalWrite(PULCINCO,HIGH);
     delayMicroseconds(200);
     digitalWrite(PULCINCO ,LOW);
     delayMicroseconds(200);}
      // eje y
      if(digitalRead(9) == HIGH){
      while (digitalRead(10) == LOW){
      digitalWrite(DIRUNO,0);
      digitalWrite(PULUNO,HIGH);
      delayMicroseconds(2000);
      digitalWrite(PULUNO,LOW);
      delayMicroseconds(2000);}
      //eje x
      if(digitalRead(10) == HIGH){
      while (digitalRead(12) == LOW){
      digitalWrite(DIRTRES,1);
      digitalWrite(PULTRES,HIGH);
      delayMicroseconds(2000);
      digitalWrite(PULTRES,LOW);
      delayMicroseconds(2000);}
       //eje z
       if(digitalRead(12) == HIGH){
       while (digitalRead(11) == LOW){
       digitalWrite(DIRDOS,0);
       digitalWrite(PULDOS,HIGH);
       delayMicroseconds(2000);
       digitalWrite(PULDOS,LOW);
       delayMicroseconds(2000);}


if(inputString.equals("BINICIO")){
    if((digitalRead(9) == HIGH) && (digitalRead(10) == HIGH) && (digitalRead(11) == HIGH) &&
(digitalRead(12) == HIGH) && (cont <= 4) && (digitalRead(7) == HIGH) && (digitalRead(6) ==
HIGH) && (DISTANCIA <= 10)){
```

```
for (int i = 0; i <= 1300 ; i++){
digitalWrite(DIRDOS,1);
digitalWrite(PULDOS,HIGH);
delayMicroseconds(1000);
digitalWrite(PULDOS,LOW);
delayMicroseconds(1000);}

for (int i = 0; i <= 4180 ; i++){
digitalWrite(DIRTRES,0);
digitalWrite(PULTRES,HIGH);
delayMicroseconds(2000);
digitalWrite(PULTRES,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 400 ; i++){
digitalWrite(DIRDOS,0);
digitalWrite(PULDOS,HIGH);
delayMicroseconds(2000);
digitalWrite(PULDOS,LOW);
delayMicroseconds(2000);}

// desplazamiento de el M.D

while (digitalRead(24) == LOW){
digitalWrite(DIRCUATRO,1);
digitalWrite(PULCUATRO,HIGH);
delayMicroseconds(4000);
digitalWrite(PULCUATRO,LOW);
delayMicroseconds(4000);}

for (int i = 0; i <= 2000 ; i++){
digitalWrite(DIRCINCO,1);
digitalWrite(PULCINCO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULCINCO,LOW);
delayMicroseconds(2000);}

            if(cont==0){
            for (int i = 0; i <= 900 ; i++){
            digitalWrite(DIRUNO,1);
            digitalWrite(PULUNO,HIGH);
            delayMicroseconds(2000);
            digitalWrite(PULUNO,LOW);
            delayMicroseconds(2000);}
            }
            if(cont==1){
            for (int i = 0; i <= 1350 ; i++){
            digitalWrite(DIRUNO,1);
            digitalWrite(PULUNO,HIGH);
            delayMicroseconds(2000);
            digitalWrite(PULUNO,LOW);
            delayMicroseconds(2000);}
            }
            if(cont==2){
            for (int i = 0; i <= 1800 ; i++){
            digitalWrite(DIRUNO,1);
            digitalWrite(PULUNO,HIGH);
            delayMicroseconds(2000);
            digitalWrite(PULUNO,LOW);
            delayMicroseconds(2000);}
```

```
            }
            if(cont==3){
            for (int i = 0; i <= 2250 ; i++){
            digitalWrite(DIRUNO,1);
            digitalWrite(PULUNO,HIGH);
            delayMicroseconds(2000);
            digitalWrite(PULUNO,LOW);
            delayMicroseconds(2000);}
            }
            if(cont==4){
            for (int i = 0; i <= 2700 ; i++){
            digitalWrite(DIRUNO,1);
            digitalWrite(PULUNO,HIGH);
            delayMicroseconds(2000);
            digitalWrite(PULUNO,LOW);
            delayMicroseconds(2000);}
            }

for (int i = 0; i <= 200 ; i++){
digitalWrite(DIRDOS,1);
digitalWrite(PULDOS,HIGH);
delayMicroseconds(2000);
digitalWrite(PULDOS,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 2000 ; i++){
digitalWrite(DIRCINCO,0);
digitalWrite(PULCINCO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULCINCO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 20 ; i++){
digitalWrite(DIRUNO,0);
digitalWrite(PULUNO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULUNO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 1500 ; i++){
digitalWrite(DIRDOS,1);
digitalWrite(PULDOS,HIGH);
delayMicroseconds(2000);
digitalWrite(PULDOS,LOW);
delayMicroseconds(2000);}
            if(cont==1){
            for (int i = 0; i <= 450 ; i++){
            digitalWrite(DIRUNO,0);
            digitalWrite(PULUNO,HIGH);
            delayMicroseconds(2000);
            digitalWrite(PULUNO,LOW);
            delayMicroseconds(2000);}
            }
            if(cont==2){
            for (int i = 0; i <= 900 ; i++){
            digitalWrite(DIRUNO,0);
            digitalWrite(PULUNO,HIGH);
            delayMicroseconds(2000);
            digitalWrite(PULUNO,LOW);
            delayMicroseconds(2000);}
```

```
            }
            if(cont==3){
            for (int i = 0; i <= 1350 ; i++){
            digitalWrite(DIRUNO,0);
            digitalWrite(PULUNO,HIGH);
            delayMicroseconds(2000);
            digitalWrite(PULUNO,LOW);
            delayMicroseconds(2000);}
            }
            if(cont==4){
            for (int i = 0; i <= 1800 ; i++){
            digitalWrite(DIRUNO,0);
            digitalWrite(PULUNO,HIGH);
            delayMicroseconds(2000);
            digitalWrite(PULUNO,LOW);
            delayMicroseconds(2000);}
            }

for (int i = 0; i <= 2060 ; i++){
digitalWrite(DIRTRES,1);
digitalWrite(PULTRES,HIGH);
delayMicroseconds(2000);
digitalWrite(PULTRES,LOW);
delayMicroseconds(2000);}

              if(cont==3){
              for (int i = 0; i <= 20 ; i++){
              digitalWrite(DIRTRES,0);
              digitalWrite(PULTRES,HIGH);
              delayMicroseconds(2000);
              digitalWrite(PULTRES,LOW);
              delayMicroseconds(2000);}
              }
              if(cont==4){
              for (int i = 0; i <= 20 ; i++){
              digitalWrite(DIRTRES,0);
              digitalWrite(PULTRES,HIGH);
              delayMicroseconds(2000);
              digitalWrite(PULTRES,LOW);
              delayMicroseconds(2000);}
              }


for (int i = 0; i <= 1300 ; i++){
digitalWrite(DIRDOS,0);
digitalWrite(PULDOS,HIGH);
delayMicroseconds(2000);
digitalWrite(PULDOS,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 640 ; i++){
digitalWrite(DIRUNO,1);
digitalWrite(PULUNO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULUNO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 3400 ; i++){
digitalWrite(DIRCINCO,1);
digitalWrite(PULCINCO,HIGH);
```

```
delayMicroseconds(2000);
digitalWrite(PULCINCO,LOW);
delayMicroseconds(2000);}

                if(cont==3){
                for (int i = 0; i <= 20 ; i++){
                digitalWrite(DIRTRES,1);
                digitalWrite(PULTRES,HIGH);
                delayMicroseconds(2000);
                digitalWrite(PULTRES,LOW);
                delayMicroseconds(2000);}
                }
                if(cont==4){
                for (int i = 0; i <= 20 ; i++){
                digitalWrite(DIRTRES,1);
                digitalWrite(PULTRES,HIGH);
                delayMicroseconds(2000);
                digitalWrite(PULTRES,LOW);
                delayMicroseconds(2000);}
                }

 for (int i = 0; i <= 180 ; i++){
digitalWrite(DIRDOS,0);
digitalWrite(PULDOS,HIGH);
delayMicroseconds(2000);
digitalWrite(PULDOS,LOW);
delayMicroseconds(2000);}
delay(5000);
for (int i = 0; i <= 250 ; i++){
digitalWrite(DIRUNO,1);
digitalWrite(PULUNO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULUNO,LOW);
delayMicroseconds(2000);}

digitalWrite(PISTON, HIGH);
delay(5000);

for (int i = 0; i <= 600 ; i++){
digitalWrite(DIRCINCO,0);
digitalWrite(PULCINCO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULCINCO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 1000 ; i++){
digitalWrite(DIRUNO,0);
digitalWrite(PULUNO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULUNO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 5430 ; i++){
digitalWrite(DIRDOS,1);
digitalWrite(PULDOS,HIGH);
delayMicroseconds(2000);
digitalWrite(PULDOS,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 680 ; i++){
```

```
digitalWrite(DIRUNO,1);
digitalWrite(PULUNO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULUNO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 1000 ; i++){
digitalWrite(DIRCINCO,1);
digitalWrite(PULCINCO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULCINCO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 130 ; i++){
digitalWrite(DIRDOS,0);
digitalWrite(PULDOS,HIGH);
delayMicroseconds(2000);
digitalWrite(PULDOS,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 30 ; i++){
digitalWrite(DIRDOS,1);
digitalWrite(PULDOS,HIGH);
delayMicroseconds(2000);
digitalWrite(PULDOS,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 680 ; i++){
digitalWrite(DIRUNO,0);
digitalWrite(PULUNO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULUNO,LOW);
delayMicroseconds(2000);}

while (digitalRead(25) == LOW){
digitalWrite(DIRCUATRO,0);
digitalWrite(PULCUATRO,HIGH);
delayMicroseconds(4000);
digitalWrite(PULCUATRO,LOW);
delayMicroseconds(4000);}

for (int i = 0; i <= 5000 ; i++){
digitalWrite(DIRDOS,0);
digitalWrite(PULDOS,HIGH);
delayMicroseconds(2000);
digitalWrite(PULDOS,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 1400 ; i++){
digitalWrite(DIRTRES,1);
digitalWrite(PULTRES,HIGH);
delayMicroseconds(2000);
digitalWrite(PULTRES,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 900 ; i++){
digitalWrite(DIRUNO,1);
digitalWrite(PULUNO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULUNO,LOW);
```

```
delayMicroseconds(2000);}

for (int i = 0; i <= 1300 ; i++){
digitalWrite(DIRCINCO,0);
digitalWrite(PULCINCO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULCINCO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 350 ; i++){
digitalWrite(DIRDOS,0);
digitalWrite(PULDOS,HIGH);
delayMicroseconds(2000);
digitalWrite(PULDOS,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 230 ; i++){
digitalWrite(DIRUNO,1);
digitalWrite(PULUNO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULUNO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 1600 ; i++){
digitalWrite(DIRCINCO,0);
digitalWrite(PULCINCO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULCINCO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 1200 ; i++){
digitalWrite(DIRUNO,0);
digitalWrite(PULUNO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULUNO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 5520 ; i++){
digitalWrite(DIRDOS,1);
digitalWrite(PULDOS,HIGH);
delayMicroseconds(2000);
digitalWrite(PULDOS,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 400 ; i++){
digitalWrite(DIRUNO,1);
digitalWrite(PULUNO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULUNO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 70 ; i++){
digitalWrite(DIRTRES,1);
digitalWrite(PULTRES,HIGH);
delayMicroseconds(2000);
digitalWrite(PULTRES,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 100 ; i++){
digitalWrite(DIRUNO,1);
```

```
digitalWrite(PULUNO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULUNO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 3200 ; i++){
digitalWrite(DIRCINCO,1);
digitalWrite(PULCINCO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULCINCO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 200 ; i++){
digitalWrite(DIRDOS,0);
digitalWrite(PULDOS,HIGH);
delayMicroseconds(2000);
digitalWrite(PULDOS,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 340 ; i++){
digitalWrite(DIRTRES,1);
digitalWrite(PULTRES,HIGH);
delayMicroseconds(2000);
digitalWrite(PULTRES,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 170 ; i++){
digitalWrite(DIRUNO,0);
digitalWrite(PULUNO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULUNO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 800 ; i++){
digitalWrite(DIRDOS,1);
digitalWrite(PULDOS,HIGH);
delayMicroseconds(2000);
digitalWrite(PULDOS,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 140 ; i++){
digitalWrite(DIRUNO,1);
digitalWrite(PULUNO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULUNO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 140 ; i++){
digitalWrite(DIRUNO,0);
digitalWrite(PULUNO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULUNO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 800 ; i++){
digitalWrite(DIRDOS,0);
digitalWrite(PULDOS,HIGH);
delayMicroseconds(2000);
digitalWrite(PULDOS,LOW);
delayMicroseconds(2000);}
```

```
for (int i = 0; i <= 170 ; i++){
digitalWrite(DIRUNO,1);
digitalWrite(PULUNO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULUNO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 250 ; i++){
digitalWrite(DIRUNO,1);
digitalWrite(PULUNO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULUNO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 120 ; i++){
digitalWrite(DIRDOS,1);
digitalWrite(PULDOS,HIGH);
delayMicroseconds(2000);
digitalWrite(PULDOS,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 1600 ; i++){
digitalWrite(DIRCINCO,0);
digitalWrite(PULCINCO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULCINCO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 750 ; i++){
digitalWrite(DIRUNO,0);
digitalWrite(PULUNO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULUNO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 2800 ; i++){
digitalWrite(DIRDOS,0);
digitalWrite(PULDOS,HIGH);
delayMicroseconds(2000);
digitalWrite(PULDOS,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 100 ; i++){
digitalWrite(DIRUNO,1);
digitalWrite(PULUNO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULUNO,LOW);
delayMicroseconds(2000);}

for (int i = 0; i <= 1600 ; i++){
digitalWrite(DIRCINCO,1);
digitalWrite(PULCINCO,HIGH);
delayMicroseconds(2000);
digitalWrite(PULCINCO,LOW);
delayMicroseconds(2000);}

digitalWrite(PISTON, LOW);

    if(cont==4){
```

```
            cont=5;}
        if(cont==3){
            cont=4;}
        if(cont==2){
            cont=3;}
        if(cont==1){
            cont=2;}
        if(cont==0){
            cont=1;}
    }
   }
  }
  }
  }
  }
 }

inputString="";
stringComplete = false;
}
}
 void serialEvent()
 {
  while (Serial.available())
  {
    char inChar = (char)Serial.read();
    if(inChar == '\n')
    {
      stringComplete = true;
    }
    else
    {
      inputString += inChar;
    }
  }
 }
```

ANEXO B: PROGRAMACIÓN PARA DETERMINAR EL NÚMERO MÁXIMO DE PASOS DEL MOTOR 1

```
int PULUNO=14;
int DIRUNO=15;
int cont = 0;

void setup() {
  pinMode (PULUNO, OUTPUT);
  pinMode (DIRUNO, OUTPUT);
  pinMode (8, INPUT);
  pinMode (10, INPUT);
  Serial.begin(9600);
}

void loop() {

 if(digitalRead(8) == HIGH){
       Serial.println(cont);
      while (digitalRead(10) == LOW){
       digitalWrite(DIRUNO,0);
       digitalWrite(PULUNO,HIGH);
       delayMicroseconds(2000);
       digitalWrite(PULUNO,LOW);
       delayMicroseconds(2000);
      cont++;

       }
       Serial.println(cont);
      }
}
```

ANEXO C: PROGRAMACIÓN PARA DETERMINAR EL NÚMERO MÁXIMO DE PASOS DEL MOTOR 2

```
int PULDOS=16;
int DIRDOS=17;
int cont = 0;

void setup() {

  pinMode (PULDOS, OUTPUT);
  pinMode (DIRDOS, OUTPUT);
  pinMode (8, INPUT);
  pinMode (11, INPUT);
  Serial.begin(9600);
}

void loop() {


 if(digitalRead(8) == HIGH){
       Serial.println(cont);
      while (digitalRead(11) == LOW){
       digitalWrite(DIRDOS,0);
       digitalWrite(PULDOS,HIGH);
       delayMicroseconds(2000);
       digitalWrite(PULDOS,LOW);
       delayMicroseconds(2000);
      cont++;

       }
       Serial.println(cont);
      }
}
```

ANEXO D: PROGRAMACIÓN PARA DETERMINAR EL NÚMERO MÁXIMO DE PASOS DEL MOTOR 3

```
int PULTRES=18;
int DIRTRES=19;
int cont = 0;

void setup() {

  pinMode (PULTRES, OUTPUT);
  pinMode (DIRTRES, OUTPUT);
  pinMode (8, INPUT);
  pinMode (12, INPUT);
  Serial.begin(9600);
}

void loop() {

 if(digitalRead(8) == HIGH){
       Serial.println(cont);
      while (digitalRead(12) == LOW){
       digitalWrite(DIRTRES,0);
       digitalWrite(PULTRES,HIGH);
       delayMicroseconds(2000);
       digitalWrite(PULTRES,LOW);
       delayMicroseconds(2000);
      cont++;

       }
       Serial.println(cont);
      }
}
```

ANEXO E: PROGRAMACIÓN PARA DETERMINAR EL NÚMERO MÁXIMO DE PASOS DEL MOTOR 5

```
int PULCINCO=22;
int DIRCINCO=23;
int cont = 0;

void setup() {
  pinMode (PULCINCO, OUTPUT);
  pinMode (DIRCINCO, OUTPUT);
  pinMode (PISTON, OUTPUT);
  pinMode (8, INPUT);
  pinMode (9, INPUT);
  Serial.begin(9600);
}

void loop() {
 if(digitalRead(8) == HIGH){
       Serial.println(cont);
      while (digitalRead(9) == LOW){
       digitalWrite(DIRCINCO,0);
       digitalWrite(PULCINCO,HIGH);
       delayMicroseconds(2000);
       digitalWrite(PULCINCO,LOW);
       delayMicroseconds(2000);
      cont++;

       }
       Serial.println(cont);
      }
}
```

ANEXO F: BASE DE DATOS PARA LA ELABORACIÓN DE GRAFICAS DE CONSUMO GENERAL

SEG.	CH1			CH2		
	V	A	W	V	A	W
1	24	0.25	6	24	1	24
2	24	1.08	25.92	24	1	24
3	24	1.11	26.64	24	1	24
4	24	1.46	35.04	24	1	24
5	24	1.46	35.04	24	1	24
6	24	1.46	35.04	24	1	24
7	24	1.47	35.28	24	1	24
8	24	1.47	35.28	24	1	24
9	24	1.47	35.28	24	1	24
10	24	1.47	35.28	24	1	24
11	24	1.54	36.96	24	1	24
12	24	1.6	38.4	24	1	24
13	24	1.6	38.4	24	1	24
14	24	1.6	38.4	24	1	24
15	24	1.6	38.4	24	1	24
16	24	1.52	36.48	24	1	24
17	24	1.53	36.72	24	1	24
18	24	1.52	36.48	24	1	24
19	24	1.52	36.48	24	1	24
20	24	1.52	36.48	24	1	24
21	24	1.52	36.48	24	1	24
22	24	1.52	36.48	24	1	24
23	24	1.52	36.48	24	1	24
24	24	1.53	36.72	24	1	24
25	24	1.53	36.72	24	1	24
26	24	1.53	36.72	24	1	24
27	24	1.53	36.72	24	1	24
28	24	1.53	36.72	24	1	24
29	24	1.52	36.48	24	1	24
30	24	1.53	36.72	24	1	24
31	24	1.52	36.48	24	1	24
32	24	1.52	36.48	24	1	24
33	24	1.5	36	24	1	24
34	24	1.48	35.52	24	1.82	43.68
35	24	1.48	35.52	24	1.75	42
36	24	1.48	35.52	24	1.88	45.12
37	24	1.48	35.52	24	1.76	42.24
38	24	1.48	35.52	24	0.98	23.52
39	24	1.48	35.52	24	0.99	23.76
40	24	1.48	35.52	24	0.98	23.52
41	24	1.48	35.52	24	0.99	23.76
42	24	1.48	35.52	24	0.97	23.28

43	24	1.46	35.04	24	0.97	23.28
44	24	1.49	35.76	24	0.98	23.52
45	24	1.49	35.76	24	0.98	23.52
46	24	2.57	61.68	24	1.05	25.2
47	24	2.57	61.68	24	1.05	25.2
48	24	2.57	61.68	24	1.05	25.2
49	24	2.57	61.68	24	1.05	25.2
50	24	2.57	61.68	24	1.05	25.2
51	24	1.48	35.52	24	0.98	23.52
52	24	1.48	35.52	24	0.97	23.28
53	24	1.48	35.52	24	0.98	23.52
54	24	1.48	35.52	24	0.99	23.76
55	24	1.48	35.52	24	0.98	23.52
56	24	1.48	35.52	24	0.99	23.76
57	24	1.48	35.52	24	0.98	23.52
58	24	1.48	35.52	24	0.99	23.76
59	24	1.53	36.72	24	1	24
60	24	1.53	36.72	24	0.99	23.76
61	24	1.53	36.72	24	0.98	23.52
62	24	1.53	36.72	24	0.99	23.76
63	24	1.53	36.72	24	0.98	23.52
64	24	1.54	36.96	24	0.99	23.76
65	24	1.53	36.72	24	0.99	23.76
66	24	1.54	36.96	24	0.99	23.76
67	24	1.54	36.96	24	0.99	23.76
68	24	1.54	36.96	24	0.99	23.76
69	24	1.53	36.72	24	0.99	23.76
70	24	1.53	36.72	24	0.99	23.76
71	24	1.53	36.72	24	0.98	23.52
72	24	1.53	36.72	24	0.98	23.52
73	24	1.54	36.96	24	0.98	23.52
74	24	1.54	36.96	24	0.98	23.52
75	24	1.54	36.96	24	0.98	23.52
76	24	1.54	36.96	24	0.98	23.52
77	24	1.55	37.2	24	0.97	23.28
78	24	2.59	62.16	24	0.97	23.28
79	24	2.59	62.16	24	0.97	23.28
80	24	2.59	62.16	24	0.97	23.28
81	24	2.59	62.16	24	0.97	23.28
82	24	1.51	36.24	24	0.95	22.8
83	24	1.51	36.24	24	0.96	23.04
84	24	1.51	36.24	24	0.97	23.28
85	24	1.51	36.24	24	0.97	23.28
86	24	1.51	36.24	24	0.97	23.28
87	24	1.51	36.24	24	0.98	23.52
88	24	1.51	36.24	24	0.97	23.28

89	24	1.51	36.24	24	0.97	23.28
90	24	1.51	36.24	24	0.98	23.52
91	24	1.51	36.24	24	0.99	23.76
92	24	1.51	36.24	24	0.97	23.28
93	24	1.51	36.24	24	0.98	23.52
94	24	1.51	36.24	24	0.99	23.76
95	24	1.51	36.24	24	1.01	24.24
96	24	1.52	36.48	24	1.01	24.24
97	24	1.52	36.48	24	1.01	24.24
98	24	1.52	36.48	24	1.01	24.24
99	24	1.52	36.48	24	1.01	24.24
100	24	1.52	36.48	24	1.01	24.24
101	24	2.61	62.64	24	1.01	24.24
102	24	1.51	36.24	24	1.3	31.2
103	24	1.51	36.24	24	1.3	31.2
104	24	1.51	36.24	24	1.3	31.2
105	24	1.51	36.24	24	1.3	31.2
106	24	1.51	36.24	24	1.27	30.48
107	24	1.51	36.24	24	1.25	30
108	24	1.52	36.48	24	1.29	30.96
109	24	2.62	62.88	24	1.29	30.96
110	24	2.63	63.12	24	1.29	30.96
111	24	2.63	63.12	24	1.29	30.96
112	24	2.64	63.36	24	1.29	30.96
113	24	1.89	45.36	24	1.29	30.96
114	24	1.56	37.44	24	1.28	30.72
115	24	1.56	37.44	24	1.28	30.72
116	24	1.56	37.44	24	1.28	30.72
117	24	1.56	37.44	24	1.28	30.72
118	24	1.56	37.44	24	1.28	30.72
119	24	1.56	37.44	24	1.29	30.96
120	24	1.56	37.44	24	1.28	30.72
121	24	1.56	37.44	24	1.28	30.72
122	24	1.57	37.68	24	1.28	30.72
123	24	1.57	37.68	24	1.28	30.72
124	24	1.57	37.68	24	1.28	30.72
125	24	1.57	37.68	24	1.28	30.72
126	24	1.57	37.68	24	1.28	30.72
127	24	1.57	37.68	24	1.28	30.72
128	24	1.57	37.68	24	1.28	30.72
129	24	1.57	37.68	24	1.28	30.72
130	24	1.57	37.68	24	1.27	30.48
131	24	1.57	37.68	24	1.27	30.48
132	24	1.57	37.68	24	1.27	30.48
133	24	1.56	37.44	24	1.27	30.48
134	24	2.62	62.88	24	1.27	30.48

135	24	2.62	62.88	24	1.27	30.48
136	24	2.61	62.64	24	1.27	30.48
137	24	2.62	62.88	24	1.27	30.48
138	24	2.61	62.64	24	1.23	29.52
139	24	1.53	36.72	24	1.23	29.52
140	24	1.53	36.72	24	1.23	29.52
141	24	1.53	36.72	24	1.23	29.52
142	24	1.53	36.72	24	1.32	31.68
143	24	2.64	63.36	24	1.32	31.68
144	24	2.64	63.36	24	1.32	31.68
145	24	1.53	36.72	24	1.69	40.56
146	24	1.53	36.72	24	2.11	50.64
147	24	1.53	36.72	24	2.12	50.88
148	24	1.53	36.72	24	2.13	51.12
149	24	1.53	36.72	24	2.1	50.4
150	24	1.55	37.2	24	1.34	32.16
151	24	1.58	37.92	24	1.33	31.92
152	24	1.58	37.92	24	1.33	31.92
153	24	1.58	37.92	24	1.33	31.92
154	24	1.58	37.92	24	1.34	32.16
155	24	1.58	37.92	24	1.33	31.92
156	24	1.58	37.92	24	1.33	31.92
157	24	1.58	37.92	24	1.33	31.92
158	24	1.58	37.92	24	1.33	31.92
159	24	1.58	37.92	24	1.33	31.92
160	24	1.58	37.92	24	1.33	31.92
161	24	1.58	37.92	24	1.33	31.92
162	24	1.58	37.92	24	1.33	31.92
163	24	1.58	37.92	24	1.34	32.16
164	24	1.58	37.92	24	1.34	32.16
165	24	1.58	37.92	24	1.34	32.16
166	24	1.58	37.92	24	1.34	32.16
167	24	1.58	37.92	24	1.34	32.16
168	24	1.58	37.92	24	1.34	32.16
169	24	1.58	37.92	24	1.34	32.16
170	24	1.58	37.92	24	1.34	32.16
171	24	1.58	37.92	24	1.34	32.16
172	24	1.58	37.92	24	1.34	32.16
173	24	1.58	37.92	24	1.34	32.16
174	24	1.58	37.92	24	1.34	32.16
175	24	1.56	37.44	24	1.33	31.92
176	24	2.63	63.12	24	1.33	31.92
177	24	2.63	63.12	24	1.33	31.92
178	24	2.63	63.12	24	1.33	31.92
179	24	1.94	46.56	24	1.33	31.92
180	24	1.53	36.72	24	1.24	29.76

181	24	1.53	36.72	24	1.26	30.24
182	24	1.53	36.72	24	1.23	29.52
183	24	1.53	36.72	24	1.26	30.24
184	24	1.53	36.72	24	1.24	29.76
185	24	1.59	38.16	24	1.26	30.24
186	24	2.62	62.88	24	1.28	30.72
187	24	1.53	36.72	24	1.26	30.24
188	24	1.53	36.72	24	1.26	30.24
189	24	1.52	36.48	24	1.26	30.24
190	24	1.52	36.48	24	1.24	29.76
191	24	1.52	36.48	24	1.24	29.76
192	24	1.52	36.48	24	1.24	29.76
193	24	1.52	36.48	24	1.28	30.72
194	24	2.63	63.12	24	1.29	30.96
195	24	2.63	63.12	24	1.29	30.96
196	24	2.63	63.12	24	1.29	30.96
197	24	2.63	63.12	24	1.29	30.96
198	24	2.12	50.88	24	1.29	30.96
199	24	1.59	38.16	24	1.28	30.72
200	24	1.59	38.16	24	1.28	30.72
201	24	1.59	38.16	24	1.28	30.72
202	24	1.59	38.16	24	1.28	30.72
203	24	1.59	38.16	24	1.28	30.72
204	24	1.59	38.16	24	1.28	30.72
205	24	1.59	38.16	24	1.28	30.72
206	24	1.59	38.16	24	1.28	30.72
207	24	1.59	38.16	24	1.28	30.72
208	24	1.59	38.16	24	1.28	30.72
209	24	1.59	38.16	24	1.28	30.72
210	24	1.59	38.16	24	1.28	30.72
211	24	1.59	38.16	24	1.28	30.72
212	24	1.59	38.16	24	1.29	30.96
213	24	1.59	38.16	24	1.29	30.96
214	24	1.59	38.16	24	1.29	30.96
215	24	1.59	38.16	24	1.29	30.96
216	24	1.59	38.16	24	1.29	30.96
217	24	1.59	38.16	24	1.29	30.96
218	24	1.59	38.16	24	1.29	30.96
219	24	1.59	38.16	24	1.29	30.96
220	24	2.63	63.12	24	1.29	30.96
221	24	1.51	36.24	24	1.26	30.24
222	24	1.51	36.24	24	1.24	29.76
223	24	1.51	36.24	24	1.26	30.24
224	24	1.51	36.24	24	1.24	29.76
225	24	1.51	36.24	24	1.25	30
226	24	1.52	36.48	24	1.27	30.48

227	24	1.52	36.48	24	1.26	30.24
228	24	1.52	36.48	24	1.25	30
229	24	1.52	36.48	24	1.27	30.48
230	24	1.52	36.48	24	1.24	29.76
231	24	1.52	36.48	24	1.24	29.76
232	24	1.52	36.48	24	1.26	30.24
233	24	1.52	36.48	24	1.24	29.76
234	24	1.57	37.68	24	1.29	30.96
235	24	1.57	37.68	24	1.29	30.96
236	24	2.62	62.88	24	1.29	30.96
237	24	1.57	37.68	24	1.29	30.96
238	24	1.57	37.68	24	1.29	30.96
239	24	1.57	37.68	24	1.29	30.96
240	24	2.63	63.12	24	1.29	30.96
241	24	1.56	37.44	24	1.29	30.96
242	24	1.56	37.44	24	1.29	30.96
243	24	1.56	37.44	24	1.29	30.96
244	24	2.63	63.12	24	1.29	30.96
245	24	2.63	63.12	24	1.29	30.96
246	24	1.51	36.24	24	1.26	30.24
247	24	1.51	36.24	24	1.24	29.76
248	24	1.51	36.24	24	1.26	30.24
249	24	1.51	36.24	24	1.23	29.52
250	24	1.52	36.48	24	1.25	30
251	24	1.52	36.48	24	1.25	30
252	24	1.52	36.48	24	1.27	30.48
253	24	2.62	62.88	24	1.29	30.96
254	24	2.63	63.12	24	1.29	30.96
255	24	2.64	63.36	24	1.29	30.96
256	24	1.57	37.68	24	1.29	30.96
257	24	1.57	37.68	24	1.29	30.96
258	24	1.57	37.68	24	1.29	30.96
259	24	1.57	37.68	24	1.29	30.96
260	24	1.57	37.68	24	1.29	30.96
261	24	1.57	37.68	24	1.29	30.96
262	24	1.57	37.68	24	1.29	30.96
263	24	1.57	37.68	24	1.29	30.96
264	24	1.57	37.68	24	1.29	30.96
265	24	1.57	37.68	24	1.29	30.96
266	24	1.57	37.68	24	1.29	30.96
267	24	1.51	36.24	24	1.25	30
268	24	1.52	36.48	24	1.26	30.24
269	24	1.52	36.48	24	1.26	30.24
270	24	1.52	36.48	24	1.24	29.76
271	24	1.52	36.48	24	1.25	30
272	24	1.52	36.48	24	1.3	31.2

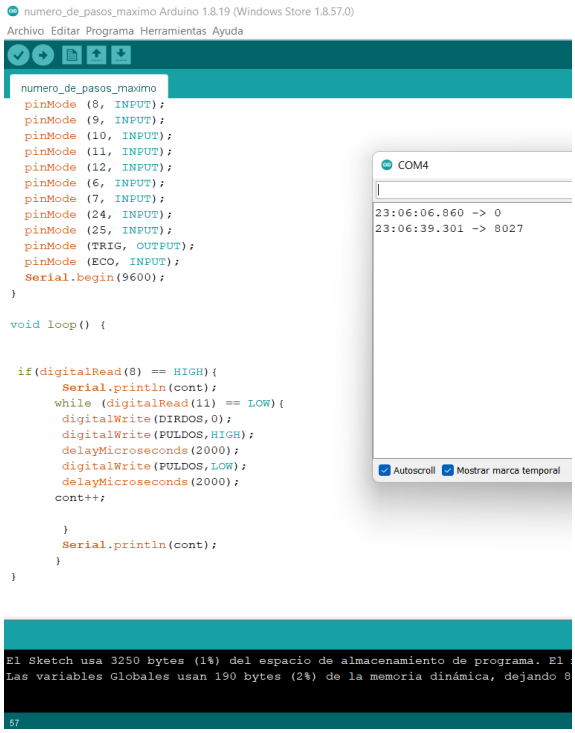
273	24	1.52	36.48	24	1.3	31.2
274	24	1.52	36.48	24	1.3	31.2
275	24	1.52	36.48	24	1.31	31.44

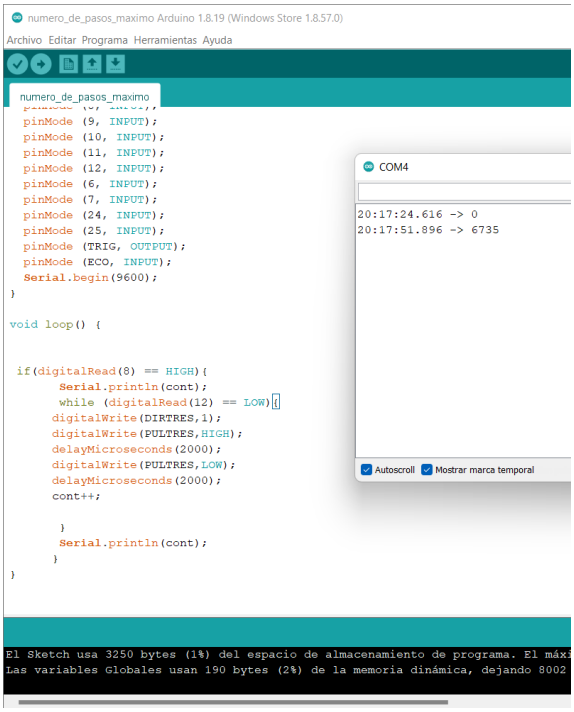
ANEXO G: PASOS REQUERIDOS PARA EL DESPLAZAMIENTO DEL ROBOT CARTESIANO DURANTE EL ENSAMBLE

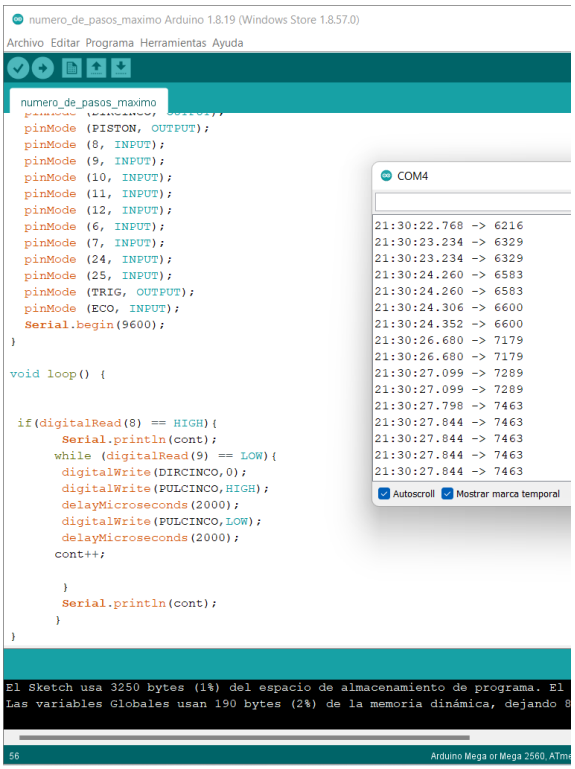
Dirección	Actuadores	Caja 1	Caja 2	Caja 3	Caja 4	Caja 5
1	Motor 2	1300	1300	1300	1300	1300
0	Motor 3	4180	4180	4180	4180	4180
0	Motor 2	400	400	400	400	400
1	Motor 5	2000	2000	2000	2000	2000
1	Motor 1	900	1350	1800	2250	2700
1	Motor 2	200	200	200	200	200
0	Motor 5	2000	2000	2000	2000	2000
0	Motor 1	20	20	20	20	20
1	Motor 2	1500	1500	1500	1500	1500
0	Motor 1		450	900	1350	1800
1	Motor 3	2060	2060	2060	2060	2060
0	Motor 3				20	20
0	Motor 2	1300	1300	1300	1300	1300
1	Motor 1	640	640	640	640	640
1	Motor 5	3400	3400	3400	3400	3400
1	Motor 3				20	20
0	Motor 2	180	180	180	180	180
1	Motor 1	250	250	250	250	250
0	Motor 5	600	600	600	600	600
0	Motor 1	1000	1000	1000	1000	1000
1	Motor 2	5430	5430	5430	5430	5430
1	Motor 1	680	680	680	680	680
1	Motor 5	1000	1000	1000	1000	1000
0	Motor 2	130	130	130	130	130
1	Motor 2	30	30	30	30	30
0	Motor 1	680	680	680	680	680
0	Motor 2	5000	5000	5000	5000	5000
1	Motor 3	1400	1400	1400	1400	1400
1	Motor 1	900	900	900	900	900
0	Motor 5	1300	1300	1300	1300	1300
0	Motor 2	350	350	350	350	350
1	Motor 1	230	230	230	230	230
0	Motor 5	1600	1600	1600	1600	1600
0	Motor 1	1000	1000	1000	1000	1000
1	Motor 2	5520	5520	5520	5520	5520
1	Motor 1	200	200	200	200	200
1	Motor 3	70	70	70	70	70
1	Motor 1	100	100	100	100	100
1	Motor 5	3000	3000	3000	3000	3000
0	Motor 2	200	200	200	200	200
1	Motor 3	340	340	340	340	340
0	Motor 1	170	170	170	170	170
1	Motor 2	800	800	800	800	800

1	Motor 1	140	140	140	140	140
0	Motor 1	140	140	140	140	140
0	Motor 2	800	800	800	800	800
1	Motor 1	170	170	170	170	170
1	Motor 1	250	250	250	250	250
1	Motor 2	120	120	120	120	120
0	Motor 5	1600	1600	1600	1600	1600
0	Motor 1	650	650	650	650	650
0	Motor 2	2800	2800	2800	2800	2800
1	Motor 5	1600	1600	1600	1600	1600

ANEXO H: CÓDIGO EN C++ PARA CONTABILIZAR EL NÚMERO DE PASOS PARA CADA EJE DEL ROBOT CARTESIANO







ANEXO I: CÓDIGO EN C# DE LA INTERFAZ GRÁFICA REALIZADA EN VISUAL STUDIO 2022

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Linq.Expressions;
using System.Text;
using System.Threading.Tasks;
using System.Timers;
using System.Windows.Forms;

namespace INTERFAZ
{
   public partial class INICIO : Form
   {
      private int conteo;

      public string INICIO8 { get; private set; }
      public string BINICIO { get; private set; }

      public INICIO()
      {
         InitializeComponent();
         conteo = 0;
      }

      private void button1_Click(object sender, EventArgs e)
      {
         try
         {
            serialPort1.WriteLine("BINICIO");
            int y = pictureBox1.Location.Y;
            int x = pictureBox4.Location.Y;
            int a = pictureBox11.Location.Y;
            if (y >= 211)
            {
               if (x >= 211)
               {
                  if (a >= 244)
                  {
                     if (conteo <= 4)
                     {
                        timer19.Enabled = true;
                     }
                  }
               }
            }
         }

         catch(Exception error)
         {
            MessageBox.Show(error.Message);
         }


      }
```

```
private void buttonSALIR_Click(object sender, EventArgs e)
{
   this.Close();
}

private void INICIO_Load(object sender, EventArgs e)
{
   btninicio.Enabled = true;
   serialPort1.PortName = "COM4";
   serialPort1.Open();
}

private void timer1_Tick(object sender, EventArgs e)
{
   int y = pictureBox1.Location.Y;
   int x = pictureBox1.Location.X;
   y = y + 1;
   if (y >= 211)
   {
      timer1.Enabled = false;
   }
   Point punto = new Point(x, y);
   pictureBox1.Location = punto;

}
private void timer2_Tick(object sender, EventArgs e)
{
   int y = pictureBox4.Location.Y;
   int x = pictureBox4.Location.X;
   y = y + 1;
   if (y >= 211)
   {
      timer2.Enabled = false;
   }
   Point punto = new Point(x, y);
   pictureBox4.Location = punto;
}

private void timer3_Tick(object sender, EventArgs e)
{
   int y = pictureBox5.Location.Y;
   int x = pictureBox5.Location.X;
   x = x - 1;
   if (x <= 508)
   {
      timer3.Enabled = false;
      timer20.Enabled = true;
   }
   Point punto = new Point(x, y);
   pictureBox5.Location = punto;
}

private void timer4_Tick(object sender, EventArgs e)
{
   int y = pictureBox11.Location.Y;
   int x = pictureBox11.Location.X;
   y = y + 1;
   if (y >= 310)
   {
```

```
        y = 311;
    }
    if (y >= 311)
    {
        x = x + 1;
    }
    if (x >= 528)
    {
        timer4.Enabled = false;
        timer5.Enabled = true;
    }
    Point punto = new Point(x, y);
    pictureBox11.Location = punto;
}

private void timer5_Tick(object sender, EventArgs e)
{
    int y = pictureBox11.Location.Y;
    int x = pictureBox11.Location.X;
    y = y - 1;
    if (y <= 211)
    {
        x = 1262;
        timer5.Enabled = false;
        timer7.Enabled = true;
        timer6.Enabled = true;
        timer21.Enabled = true;
    }
    Point punto = new Point(x, y);
    pictureBox11.Location = punto;
}

private void timer6_Tick(object sender, EventArgs e)
{
    int y = pictureBox1.Location.Y;
    int x = pictureBox1.Location.X;
    if (y >= 211)
    {
        y = 112;
        timer6.Enabled = false;
    }
    Point punto = new Point(x, y);
    pictureBox1.Location = punto;
}

private void timer7_Tick(object sender, EventArgs e)
{
    int y = pictureBox3.Location.Y;
    int x = pictureBox3.Location.X;
    if (x >= 1262)
    {
        x = 528;
        y = 211;
        timer7.Enabled = false;
        timer21.Enabled = true;
    }
    Point punto = new Point(x, y);
    pictureBox3.Location = punto;
}
```

```
private void timer8_Tick(object sender, EventArgs e)
{
   int y = pictureBox3.Location.Y;
   int x = pictureBox3.Location.X;
   y = y + 1;
   if (y >= 484)
   {
      timer8.Enabled = false;
      timer22.Enabled = true;
   }
   Point punto = new Point(x, y);
   pictureBox3.Location = punto;
}

private void timer9_Tick(object sender, EventArgs e)
{
   int y = pictureBox3.Location.Y;
   int x = pictureBox3.Location.X;
   x = x + 1;
   if (x >= 631)
   {
      timer9.Enabled = false;

   }
   Point punto = new Point(x, y);
   pictureBox3.Location = punto;
}

private void timer10_Tick(object sender, EventArgs e)
{
   int y = pictureBox5.Location.Y;
   int x = pictureBox5.Location.X;
   x = x + 1;
   if (x >= 610)
   {
      timer10.Enabled = false;
      timer23.Enabled = true;

   }
   Point punto = new Point(x, y);
   pictureBox5.Location = punto;
}

private void timer11_Tick(object sender, EventArgs e)
{
   int y = pictureBox4.Location.Y;
   int x = pictureBox4.Location.X;
   y = y + 1;
   if (y >= 484)
   {
      y = 112;
      timer11.Enabled = false;
      timer12.Enabled = true;
      timer13.Enabled = true;

   }
   Point punto = new Point(x, y);
   pictureBox4.Location = punto;
}
```

```
private void timer12_Tick(object sender, EventArgs e)
{
   int y = pictureBox3.Location.Y;
   int x = pictureBox3.Location.X;
   if (x >= 631)
   {
      x = 1262;
      y = 71;
      timer12.Enabled = false;

   }
   Point punto = new Point(x, y);
   pictureBox3.Location = punto;
}

private void timer13_Tick(object sender, EventArgs e)
{
   int y = pictureBox7.Location.Y;
   int x = pictureBox7.Location.X;
   if (x >= 1262)
   {
      y = 484;
      x = 631;
      timer13.Enabled = false;
      timer24.Enabled = true;
   }
   Point punto = new Point(x, y);
   pictureBox7.Location = punto;
}

private void timer14_Tick(object sender, EventArgs e)
{
   int y = pictureBox7.Location.Y;
   int x = pictureBox7.Location.X;
   y = y - 1;
   if (y <= 334)
   {
      timer14.Enabled = false;
      timer17.Enabled = true;
      timer16.Enabled = true;
   }
   Point punto = new Point(x, y);
   pictureBox7.Location = punto;
}

private void timer15_Tick(object sender, EventArgs e)
{
   int y = pictureBox7.Location.Y;
   int x = pictureBox7.Location.X;
   x = x + 1;
   if (x >= 967)
   {
      x = 1262;
      y = 149;
      timer15.Enabled = false;
   }
   Point punto = new Point(x, y);
   pictureBox7.Location = punto;
}
```

```
private void button6_Click(object sender, EventArgs e)
{
   timer1.Enabled = true;
   int y = pictureBox11.Location.Y;
   int x = pictureBox11.Location.X;
   if (x >= 1000)
   {
      x = 416;
      y = 244;
   }
   Point punto = new Point(x, y);
   pictureBox11.Location = punto;
}

private void button7_Click(object sender, EventArgs e)
{
   timer2.Enabled = true;
}

private void button8_Click(object sender, EventArgs e)
{
   int y = pictureBox11.Location.Y;
   int x = pictureBox11.Location.X;
   if (x >= 1000)
   {
      x = 416;
      y = 244;
   }
   Point punto = new Point(x, y);
   pictureBox11.Location = punto;
}

private void timer16_Tick(object sender, EventArgs e)
{
   conteo++;
   label2.Text = conteo.ToString();
   timer16.Enabled = false;
}

private void btnreset_Click(object sender, EventArgs e)
{
   if (conteo >= 0)
   {
      conteo = 0;
      label2.Text = conteo.ToString();
   }
   if (conteo >= 0)
   {
      int y = pictureBox1.Location.Y;
      int x = pictureBox1.Location.X;
      if (x >= 0)
      {
         x = 530;
         y = 112;
      }
      Point punto = new Point(x, y);
      pictureBox1.Location = punto;
   }
   if (conteo >= 0)
   {
```

```
    int y = pictureBox3.Location.Y;
    int x = pictureBox3.Location.X;
    if (x >= 0)
    {
        x = 1262;
        y = 71;
    }
    Point punto = new Point(x, y);
    pictureBox3.Location = punto;
}
if (conteo >= 0)
{
    int y = pictureBox4.Location.Y;
    int x = pictureBox4.Location.X;
    if (x >= 0)
    {
        x = 634;
        y = 112;
    }
    Point punto = new Point(x, y);
    pictureBox4.Location = punto;
}
if (conteo >= 0)
{
    int y = pictureBox5.Location.Y;
    int x = pictureBox5.Location.X;
    if (x >= 0)
    {
        x = 610;
        y = 480;
    }
    Point punto = new Point(x, y);
    pictureBox5.Location = punto;
}
if (conteo >= 0)
{
    int y = pictureBox11.Location.Y;
    int x = pictureBox11.Location.X;
    if (x >= 0)
    {
        x = 1262;
        y = 225;
    }
    Point punto = new Point(x, y);
    pictureBox11.Location = punto;
}
if (conteo >= 0)
{
    int y = pictureBox7.Location.Y;
    int x = pictureBox7.Location.X;
    if (x >= 0)
    {
        x = 1262;
        y = 149;
    }
    Point punto = new Point(x, y);
    pictureBox7.Location = punto;
}
if (conteo >= 0)
{
```

```
        timer1.Enabled = false;
        timer2.Enabled = false;
        timer3.Enabled = false;
        timer4.Enabled = false;
        timer5.Enabled = false;
        timer6.Enabled = false;
        timer7.Enabled = false;
        timer8.Enabled = false;
        timer9.Enabled = false;
        timer10.Enabled = false;
        timer11.Enabled = false;
        timer12.Enabled = false;
        timer13.Enabled = false;
        timer14.Enabled = false;
        timer15.Enabled = false;
        timer16.Enabled = false;
        timer17.Enabled = false;
        timer18.Enabled = false;

    }
}

private void INICIO_FormClosing(object sender, FormClosingEventArgs e)
{
    if (serialPort1.IsOpen)
    {
        try
        {
            serialPort1.Close();
        }
        catch (Exception error)
        {
            MessageBox.Show(error.Message);
        }
    }
}

private void timer19_Tick(object sender, EventArgs e)
{
    timer3.Enabled = true;
    timer19.Enabled = false;
}

private void timer20_Tick(object sender, EventArgs e)
{
    timer4.Enabled = true;
    timer20.Enabled = false;
}

private void timer21_Tick(object sender, EventArgs e)
{
    timer8.Enabled = true;
    timer21.Enabled = false;
}

private void timer22_Tick(object sender, EventArgs e)
{
    timer9.Enabled = true;
    timer10.Enabled = true;
    timer22.Enabled = false;
```

```
        }

        private void timer23_Tick(object sender, EventArgs e)
        {
            timer23.Enabled = false;
            timer11.Enabled = true;
        }

        private void timer24_Tick(object sender, EventArgs e)
        {
            timer14.Enabled = true;
            timer24.Enabled = false;
        }

        private void timer17_Tick(object sender, EventArgs e)
        {
            timer15.Enabled = true;
            timer17.Enabled = false;
        }

    }
}
```

UNIDAD DE PROCESOS TÉCNICOS Y ANÁLISIS BIBLIOGRÁFICO Y DOCUMENTAL

REVISIÓN DE NORMAS TÉCNICAS, RESUMEN Y BIBLIOGRAFÍA

Fecha de entrega: 24 / 11 / 2023

INFORMACIÓN DEL AUTOR/A (S)
Nombres – Apellidos: HUGO JHONNATAN BONILLA MENA
INFORMACIÓN INSTITUCIONAL
Facultad: MECÁNICA
Carrera: INGENIERÍA INDUSTRIAL
Título a optar: INGENIERO INDUSTRIAL
f. Analista de Biblioteca responsable: Ing. CPA. Jhonatan Rodrigo Parreño Uquillas**. MBA.**

1834-DBRA-UPT-2023