
ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

# Implementación de un algoritmo flexible en el robot Mitsubishi MELFA RV-2SDB que permita mejorar la eficiencia en la programación utilizando unidades de medida inercial.

**Pichucho Castellano, Jefferson Paul**

**Sampedro Gómez, Andrés Marcelo**

**Departamento de Energía y Mecánica**

**Carrera de Ingeniería Mecatrónica**

**Trabajo de Titulación, previo a la obtención del Título de Ingeniero Mecatrónico**

**Ing. Mendoza Chipantasi, Darío José**

**26 de octubre del 2020**

**Latacunga**

**DEPARTAMENTO DE CIENCIAS DE LA ENERGÍA Y MECÁNICA**

**CARRERA DE INGENIERÍA MECATRÓNICA**

# CERTIFICACIÓN

Certifico que el trabajo de titulación, **"IMPLEMENTACIÓN DE UN ALGORITMO FLEXIBLE EN EL ROBOT MITSUBISHI MELFA RV-2SDB QUE PERMITA MEJORAR LA EFICIENCIA EN LA PROGRAMACIÓN UTILIZANDO UNIDADES DE MEDIDA INERCIAL PARA EL LABORATORIO DE MECATRÓNICA DE LA UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE SEDE LATACUNGA"**, fue realizado por los señores **Pichucho Castellano, Jefferson Paúl** y **Sampedro Gómez, Andrés Marcelo,** el mismo que ha sido revisado en su totalidad, analizado por la herramienta de verificación de similitud de contenido; por lo tanto cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de las Fuerzas Armadas ESPE. razón por la cual me permito acreditar y autorizar para que lo sustente públicamente.

**Latacunga, 27 de octubre del 2020**

Firmado electrónicamente por:
DARIO JOSE
MENDOZA
CHIPANTASI

.....

**Ing. Mendoza Chipantasi, Darío José**
C.C.: 0603110834

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

# DEPARTAMENTO DE CIENCIAS DE LA ENERGÍA Y MECÁNICA

## CARRERA DE INGENIERÍA MECATRÓNICA

## REPORTE URKUND

URKUND

### Document Information

| | |
|---|---|
| Analyzed document | tesis_final.pdf (D82820091) |
| Submitted | 10/26/2020 9:20:00 PM |
| Submitted by | Mendoza Chipantasi Dario Jose |
| Submitter email | djmendoza@espe.edu.ec |
| Similarity | 4% |
| Analysis address | djmendoza.espe@analysis.urkund.com |

### Sources included in the report

| | | |
|---|---|---|
| SA | **Universidad de las Fuerzas Armadas ESPE / Tesis - Juan Daniel Galarza Panimboza - 14-01-2019.pdf**<br>Document Tesis - Juan Daniel Galarza Panimboza - 14-01-2019.pdf (D46762060)<br>Submitted by: jdgalarza1@espe.edu.ec<br>Receiver: lfescobar.espe@analysis.urkund.com | 3 |
| W | URL: https://repositorio.espe.edu.ec/bitstream/21000/18719/1/T-ESPE-039039.pdf<br>Fetched: 3/14/2020 5:22:17 PM | 4 |
| SA | **Universidad de las Fuerzas Armadas ESPE / TESIS MAESTRIA IMPRESION.docx**<br>Document TESIS MAESTRIA IMPRESION.docx (D42351279)<br>Submitted by: maferm1410@gmail.com<br>Receiver: djmendoza.espe@analysis.urkund.com | 1 |
| W | URL: https://repository.unilibre.edu.co/bitstream/handle/10901/9474/Tesis%20de%20Grado. ...<br>Fetched: 7/23/2020 2:38:13 PM | 4 |
| W | URL: https://repositorio.espe.edu.ec/bitstream/21000/8853/1/T-ESPEL-MEC-0023.pdf<br>Fetched: 11/20/2019 9:35:50 PM | 2 |
| SA | **Universidad de las Fuerzas Armadas ESPE / TESIS COMPLETA.docx**<br>Document TESIS COMPLETA.docx (D15653572)<br>Submitted by: fersitaonate@gmail.com<br>Receiver: hcteran.espe@analysis.urkund.com | 3 |
| SA | **Universidad de las Fuerzas Armadas ESPE / Tesis Almache-Quintana.pdf**<br>Document Tesis Almache-Quintana.pdf (D77763674)<br>Submitted by: dcloza@espe.edu.ec<br>Receiver: dcloza.espe@analysis.urkund.com | 1 |
| W | URL: https://www.scribd.com/document/416436163/2015-Libro-Desarrollo-de-Software-y-Hard ...<br>Fetched: 1/19/2020 8:07:30 PM | 1 |
| SA | **Universidad de las Fuerzas Armadas ESPE / TRABAJO DE TITULACION FINAL1 URKUND .docx**<br>Document TRABAJO DE TITULACION FINAL1 URKUND .docx (D78049516)<br>Submitted by: hrortiz@espe.edu.ec<br>Receiver: hrortiz.espe@analysis.urkund.com | 3 |
| W | URL: https://robotics.ee.uwa.edu.au/courses/robotics/project/festo/MPS_TD_V2.4_EN/Engli ...<br>Fetched: 10/26/2020 9:21:00 PM | 1 |

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

## DEPARTAMENTO DE CIENCIAS DE LA ENERGÍA Y MECÁNICA

## CARRERA DE INGENIERÍA MECATRÓNICA

## RESPONSABILIDAD DE AUTORÍA

Nosotros, **Pichucho Castellano, Jefferson Paúl** con cédula de ciudadanía N°0502879729 **Sampedro Gómez, Andrés Marcelo** con cédula de ciudadanía N°1724904667, declaramos que el contenido, ideas y criterios del trabajo de titulación: **"IMPLEMENTACIÓN DE UN ALGORITMO FLEXIBLE EN EL ROBOT MITSUBISHI MELFA RV-2SDB QUE PERMITA MEJORAR LA EFICIENCIA EN LA PROGRAMACIÓN UTILIZANDO UNIDADES DE MEDIDA INERCIAL PARA EL LABORATORIO DE MECATRÓNICA DE LA UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE SEDE LATACUNGA"**, es de nuestra autoría y responsabilidad, cumpliendo con los requisitos legales, teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de las Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Consecuentemente el contenido de la investigación mencionada es veraz.

**Latacunga, 27 de octubre del 2020**

.....

**Pichucho Castellano, Jefferson Paúl**
C.C.: 0502879729

.....

**Sampedro Gómez, Andrés Marcelo**
C.C.: 1724904667

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

**DEPARTAMENTO DE CIENCIAS DE LA ENERGÍA Y MECÁNICA**

**CARRERA DE INGENIERÍA MECATRÓNICA**

# AUTORIZACIÓN DE PUBLICACIÓN

Nosotros, **Pichucho Castellano, Jefferson Paúl** con cédula de ciudadanía N°0502879729 y **Sampedro Gómez, Andrés Marcelo** con cédula de ciudadanía N°1724904667, autorizamos a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de titulación: **"IMPLEMENTACIÓN DE UN ALGORITMO FLEXIBLE EN EL ROBOT MITSUBISHI MELFA RV-2SDB QUE PERMITA MEJORAR LA EFICIENCIA EN LA PROGRAMACIÓN UTILIZANDO UNIDADES DE MEDIDA INERCIAL PARA EL LABORATORIO DE MECATRÓNICA DE LA UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE SEDE LATACUNGA"**, en el repositorio Institucional, cuyo contenido, ideas y criterios es de nuestra responsabilidad.

**Latacunga, 27 de octubre del 2020**

.....

**Pichucho Castellano, Jefferson Paúl**
C.C.: 0502879729

.....

**Sampedro Gómez, Andrés Marcelo**
C.C.: 1724904667

## DEDICATORIA

Este trabajo lo dedico a mi familia que son la base de mi existir, a mi mami Digna que con su fortaleza ha luchado para darme siempre lo mejor junto a mi papi Rodolfo que es mi apoyo incondicional, mi ñaña Nathaly que con sus consejos ha sido un pilar fundamental en mi vida. A mis abuelitas, ñaños, ñañas, tíos y tías que con sus amor y cariño me otorgan la energía necesaria para continuar con mis metas. A mis amigos Álvaro, Azael y Marcelo, mis primas Kathry, Saya y Samy quienes han estado presente en esta etapa de mi vida y a Nidia quien me ha brindado su amor incondicional y su valiosa compañía.

**Jefferson**

Este proyecto de titulación está dedicado especialmente a mi madre Michael, a mi padre Tulio. Quienes fueron las personas a las que directamente afectó toda la disponibilidad que tuve que dar a mi carrera universitaria, creando en ellos preocupaciones y angustias al no poder ver y abrazar a su hijo durante mucho tiempo, ustedes fueron quienes forjaron mi carácter, me enseñaron valores, humildad y me dieron un hogar digno, pero con necesidades que me hicieron llegar a superarme cada día. A mi hermana Juliana por ser a quien siempre quise dejar el camino correcto a seguir. A mis abuelitos Betty, Elsa y Monfilio por tenerme paciencia y ayudar en mi crianza. A Jenny, Anita, Amparito, Clary, Inés, Pepe, Romel, Edwin, Max y Diego por su apoyo incondicional como mis tías y tíos. A mis primos y primas, por las grandes vivencias que tuve durante mi vida.

**Andrés**

## AGRADECIMIENTOS

En primer lugar, quiero agradecer a Dios por brindarme la inteligencia, sabiduría y conocimiento necesario para culminar esta etapa de mi vida. Mi profundo agradecimiento a la Universidad de las Fuerzas Armadas y a toda la Carrera de Mecatrónica que me brindaron profesores de calidad y calidez permitiéndome crecer día a día como profesional. Agradezco a nuestro tutor Ing. Darío Mendoza que, con su conocimiento y apoyo nos guio durante este proyecto. A mis amigos, compañeros y futuros colegas, en especial a mi mejor amigo y compañero de tesis Marcelo que hemos compartido este largo camino llamado universidad.

**Jefferson**

Agradezco a mis padres quienes nunca dejaron de apoyarme en esta etapa de mi vida, a mis tíos que aportaron con su granito de arena a esta causa, a mis tías por su apoyo, afecto y cariño. A mis abuelos por su cariño y sus consejos. A todos los docentes que formaron parte de mi formación universitaria, cada uno de ellos tienen el crédito de que yo haya llegado a estas instancias. A mi querido Departamento de Mecatrónica y los docentes que lo conforman, por su compañía y refugio. Especialmente al Ing. Vicente Hallo quien en su momento fue el directo de carrera y al Ing. Darío Mendoza quien fue mi docente, luego directo de carrera y posteriormente director de nuestro proyecto de investigación. Agradezco infinitamente por la ayuda que me brindaron a los padres de mi compañero y gran amigo Jefferson, Digna y Rodolfo. Por su puesto a Jefferson quien fue un pilar fundamental en cada una de las dificultades que tuve que atravesar. A la familia Moncada Gómez quienes me apoyaron cuando transcurría un oscuro pasaje de mi carrera universitaria. Tengan presente que su aporte fue determinante y siempre podrán contar conmigo.

**Andrés**

**ÍNDICE DE CONTENIDOS**

## ÍNDICE DE TABLAS

## ÍNDICE DE FIGURAS

## RESUMEN

El presente proyecto tiene como objetivo la implementación de un algoritmo flexible en el Robot MELFA RV-2SDB que permita mejorar la eficiencia en la programación, utilizando unidades de medida inercial. Partiendo del estudio de las aplicaciones de las unidades de medida inercial (IMU) y de las distintas formas de programación del brazo robótico existente. La implementación constará de cuatro etapas principales. La primera etapa hace referencia a la recepción de datos de las IMU, luego empieza el tratamiento de los mismos, lo que consiste en la aplicación de filtros o algoritmos que permitan mapear los movimientos generados por el operador. La segunda etapa es la comunicación de todo el sistema dividiéndose así en dos subetapas: comunicación entre el sensor (IMU) junto con el ordenador de manera inalámbrica y el enlace entre el ordenador junto con el controlador del brazo robótico de forma alámbrica. La tercera etapa consiste en la generación de un archivo ejecutable para el brazo robótico el cual contenga trayectorias otorgadas por el sensor. La cuarta etapa es la presentación de datos en una interfaz gráfica, capaz de seleccionar entre sus distintas formas de operación como teleoperación, capaz de controlar el brazo robótico en tiempo real y programación offline con la finalidad de crear unos conjuntos de trayectorias para su posterior ejecución. Finalmente, para comprobar la eficiencia del algoritmo, se realizarán pruebas de funcionamiento en aplicaciones como: pintura (movimientos complejos).

**PALABRAS CLAVE:**

- **TELEOPERACIÓN**
- **UNIDADES DE MEDIDA INERCIAL**
- **BRAZO ROBÓTICO**

## ABSTRACT

The objective of this project is the implementation of a flexible algorithm in the MELFA RV-2SDB Robot that allows improving the efficiency in programming, using inertial measurement units. Starting from the study of the applications of inertial measurement units (IMU) and the different ways of programming the existing robotic arm. The implementation will consist of four main stages. The first stage refers to the reception of data from the IMUs, then the treatment of the same begins, which consists of the application of filters or algorithms that allow mapping the movements generated by the operator. The second stage is the communication of the entire system, thus being divided into two sub-stages: communication between the sensor (IMU) together with the computer wirelessly and the link between the computer together with the robot arm controller in a wired way. The third stage consists of the generation of an executable file for the robotic arm, which contains trajectories provided by the sensor. The fourth stage is the presentation of data in a graphical interface, capable of selecting between its different forms of operation such as teleoperation, capable of controlling the robotic arm in real time and offline programming in order to create sets of trajectories for later execution. Finally, to check the efficiency of the algorithm, performance tests will be carried out in applications such as: painting (complex movements).



**KEYWORDS:**

- **TELEOPERATION**
- **INERTIAL MEASUREMENT UNITS**
- **ROBOTIC ARM**

# CAPÍTULO I

# GENERALIDADES

## 1.1. Introducción

La aparición de la robótica tiene un lazo con el origen del hombre y su necesidad de idear herramientas para facilitar sus tareas diarias. Desde entonces el ser humano siempre ha buscado simplificar el trabajo pesado o repetitivo, inventando maquinas que cumplan específicos procesos para lograr su cometido. Gracias a los avances tecnológicos, la robótica llegó al nivel de poder imitar movimientos de un brazo humano, con la ayuda de algoritmos que transforman dichos movimientos en coordenadas en el espacio y de sensores que captan los desplazamientos que describe cada articulación.

El presente proyecto tiene la finalidad de mejorar la eficiencia en la programación realizada por el operador del brazo robótico Mitsubishi MELFA RV-2SDB, a través de un algoritmo flexible basado en la lectura de sensores previamente ubicados en las junturas de la extremidad superior del operador, además de tener el acceso a las coordenadas generadas y de crear un archivo con todas las trayectorias descritas para que posteriormente se las pueda dar uso a través de una aplicación o una demostración para usuarios aspirantes.

## 1.2. Antecedentes

La telerobótica pretende la integración de las habilidades del hombre y de la máquina, aumentando la capacidad de sus partes, siendo así, una forma avanzada de teleoperación, donde el mecanismo telecontrolado puede funcionar de manera autónoma, sin la supervisión de su operador

durante cortos periodos de tiempo, incrementando la fiabilidad de las operaciones, facilitar su operación y reducir el tiempo de ejecución, con lo cual, se genera un control compartido en el que se combinan las órdenes del operador con un sistema de control automático. (Gómez de Gabriel et al. 2006)

En el laboratorio de Mecatrónica (ESPE-L) existe un brazo robótico industrial de marca Mitsubishi y modelo MELFA RV-2SDB el cual utiliza la programación por guiado, que consiste en llevar al robot a una determinada posición con la finalidad que registre dicha posición, para su posterior repetición de manera automática. Un modo de satisfacer el problema mecánico de guiar todo el armazón del robot se resuelve con la práctica del guiado pasivo por maniquí. Para este evento se tiene un imitador del robot, mientras que este permanece fuera de línea. El maniquí posee la misma disposición que el robot auténtico, pero es mucho más liviano y por consiguiente sencillo de desplazar. La programación se realiza llevando de la mano a este doble, mientras que el dispositivo de control selecciona y almacena información, con determinada frecuencia, para que finalmente el operador pueda repetir la tarea que realizó el robot. Los robots de pintura fabricados por Nordson se programaron utilizando este procedimiento. (Barrientos et al. 2007)

En 1993 los investigadores Sturman y Zeltzer comprobaron que el rendimiento en los dispositivos de mano completa es superior a la programación por guiado en operaciones de guiado de bajo nivel (en coordenadas cartesianas). Estos dispositivos utilizan una técnica de localizador tridimensional para lograr determinar el lugar en el espacio de la mano y un contiguo de sensores distribuidos para adquirir el lugar de los dedos, los cuales pueden ser mecánicos, a manera de exoesqueleto o potenciómetros de polímero conductivo. Pese a esto, al manejar esta variedad de sensores se encuentran varias desventajas como lo son el complejo armazón mecánico, profunda

instrucción y destreza del operador, además de cantidades excesivas de datos enviados y recibidos. Debido a que, la programación guiada presenta una serie de inconvenientes, como la necesidad de utilizar al propio robot y su entorno para realizar la programación de trayectorias complejas.

En el futuro, la telerrobótica tendrá avances significativos, pues, últimamente ha existido un gran crecimiento en el interés de desarrollar una interfaz entre las interacciones de humanos y robots. Las ondas cerebrales han surgido como una forma común y efectiva de facilitar esta conexión, conocida como interfaz cerebro máquina (BMI), dichas actividades cerebrales se registran a través de varios métodos de neuroimagen invasivos y no invasivos (sensores dentro o fuera de la cabeza) que se traducen en señales de comando para controlar dispositivos protésicos externos como un brazo robótico. (Kilmarx et al. 2017)

### 1.3. Planteamiento del problema

Los métodos de programación de robots son: guiado y textual, los cuales han sido utilizados por su gran precisión y fácil uso en programas sencillos, sin embargo, cuando el operador del robot requiere programar varias trayectorias complejas (distintas a líneas y círculos) con el uso de dispositivos de enseñanza propio del brazo robótico (Teach Pendants), el tiempo necesario para completar esta tarea es excesivo debido a la cantidad de puntos que tendrán que ser almacenados de manera manual.

En los laboratorios de la Universidad (ESPE-L) existen distintos modelos de brazos robóticos como “KUKA” y “Mitsubishi” que utilizan un software privado para su correcto uso y programación, debido a esto y a los costos de adquisición existe solo una copia, lo que impide al estudiante poseer un programa ejecutable en su ordenador capaz de manejar y programar

libremente dichos brazos robóticos. Además, para su manipulación y control se requiere de personal técnico calificado con conocimientos de robótica industrial.

Debido a que el mundo se encuentra en una constante innovación y transformación tecnológica, los brazos robóticos existentes en la Universidad se encuentran descontinuados ya que no poseen un control sensorizado y programación guiada, limitando el aprendizaje de los estudiantes.

Durante la programación de un robot industrial se requiere de conocimientos previos del uso de la aplicación, para ser utilizado, pero los sistemas actuales de la Universidad no tienen la capacidad de aprovechar la experiencia humana adquirida durante la práctica manual de estas aplicaciones.

### 1.4. Descripción resumida del proyecto

El presente proyecto tiene como objetivo la implementación de un algoritmo flexible en el Robot MELFA RV-2SDB de la estación MPS Robot de la marca Festo que permita mejorar la eficiencia en la programación, utilizando unidades de medida inercial para el Laboratorio de Mecatrónica de la Universidad de las Fuerzas Armadas Sede Latacunga. El proyecto parte del estudio y análisis de las aplicaciones de las unidades de medida inercial (IMU) y de las distintas formas de programación del brazo robótico existente. A partir de esta información, se investigará los pasos necesarios para realizar una sinergia entre dichos temas.

La implementación constará de cuatro etapas principales: adquirir y filtrar señales de los sensores (IMU), realizar la comunicación sensor-ordenador-controlador del brazo robótico,

generar un archivo ejecutable compatible con el controlador y por último crear una interfaz gráfica capaz de mostrar distintos datos y modos de operación (teleoperación o programación).

La primera etapa es la adquisición y tratamiento de señales de los sensores que hace referencia a la recepción de datos de las IMU, productos ya existentes en el mercado los cuales pueden ser utilizados por su reducido tamaño, la resolución de datos y facilidad de programación. Luego de adquirir los datos o señales empieza el tratamiento de los mismos, lo que consiste en la aplicación de filtros o algoritmos que permitan mapear el movimiento y desplazamientos generados por el operador.

La segunda etapa es: la comunicación de todo el sistema, se puede dividir en dos subetapas esenciales: la primera subetapa es la comunicación entre el sensor (IMU) y el ordenador de manera inalámbrica, permitiendo el uso de los datos arrojados del sensor en un software para su posterior tratamiento de señales. La segunda subetapa es el enlace entre el ordenador y el controlador del brazo robótico de forma alámbrica debido a que la respuesta de ordenador-controlador debe ser rápida.

La tercera etapa consiste en la generación de un archivo ejecutable para el brazo robótico el cual contenga trayectorias y posiciones otorgadas por el sensor, gracias a que el controlador posee funciones especiales que serán explotadas como la creación de archivos, almacenamiento de posiciones, etc. mediante comandos ejecutados desde el ordenador.

La cuarta etapa: es la presentación de datos en una interfaz gráfica, la cual poseerá distintas funcionalidades siendo capaz de seleccionar entre sus distintas formas de operación como son:

teleoperación capaz de controlar el brazo robótico en tiempo real y programación offline con la finalidad de crear unos conjuntos de trayectorias para su posterior ejecución.

Finalmente, para comprobar la eficacia del algoritmo, se realizarán pruebas de funcionamiento en aplicaciones como: pintura (movimientos complejos) o teleoperación (control en tiempo real del brazo robótico con datos del sensor). Una vez generado los archivos para el controlador del brazo robótico serán ejecutados para determinar si los valores de posición son adecuados para dichas aplicaciones, tomando en consideración el tiempo total utilizado para culminar con satisfacción la tarea designada.

### 1.5. Justificación e importancia

El manual de usuario del controlador CRn-700 series del brazo robótico Mitsubishi MELFA RV-2SDB contiene la descripción detallada de como programar manualmente un movimiento sencillo ocupando trayectorias lineales o circulares paso por paso, en el que surgen dos problemas como la familiarización de la interfaz y la manipulación del robot, una vez superado estos inconvenientes es posible generar trayectorias programadas lo que conlleva un tiempo excesivo para culminar un simple programa. Es por ello que es necesario implementar un algoritmo capaz de generar trayectorias complejas de manera sencilla y fácilmente manipulable.

Los brazos robóticos industriales al ser productos de grandes empresas poseen softwares de código cerrado, impidiendo analizar su código fuente para adaptarlo a distintas formas de trabajo, debido a esto es de suma importancia implementar un algoritmo flexible en código abierto permitiendo la compresión, modificación y uso de los estudiantes de la Universidad de las Fuerzas Armadas.

Hasta el momento, no existen robots industriales que sean tipos sensoriales capaces de responder a sensores externos, debido al poco desarrollo de esta tecnología por lo que resulta relevante analizar soluciones para que un robot industrial sea capaza de receptar movimientos de un sensor colocado en la mano del operador, convertirlos en el lenguaje estructurado que procesa el controlador del brazo robótico para ejecutarlos y repetirlos. Por consecuente el archivo final pudo utilizar la experiencia del operador ya que envió las posiciones que el usuario conoce para que un trabajo sea completado con éxito. El algoritmo que se pretende implementar tiene como función el sustituir el proceso donde el operador tiene que programar manualmente al brazo robótico, enseñándole cada una de las posiciones hacia donde tiene que moverse.

La importancia del presente proyecto radica en poner en funcionamiento este algoritmo a nivel universitario en el país y que el código abierto utilizado pueda ayudar a replicar esta tarea en robots de tipo educativo, dado que aún no se ha logrado analizar una segunda opción al momento de conformar el lenguaje de programación para mover un brazo robótico, y esto sería un aporte necesario para el desarrollo de nuevas aplicaciones en el ámbito de Robótica Industrial.

En sinopsis, este proyecto pretende implementar un algoritmo que registre datos de un artefacto mecatrónico que estructure lenguaje de programación para manejar el brazo robótico Mitsubishi MELFA citado anteriormente, con el objetivo de mejorar la eficiencia en su manipulación y fortalecer conocimientos en la teoría de Robótica Industrial que será necesaria para el desarrollo de la Ingeniería Mecatrónica a nivel nacional.

## 1.6. Objetivos

### 1.6.1. *Objetivo General*

- Implementar un algoritmo flexible en el robot MELFA RV-2SDB que permita mejorar la eficiencia en la programación utilizando unidades de medida inercial.

### 1.6.2. *Objetivos Específicos*

- Recopilar información sobre robótica industrial, telerobótica, sensores que manejen unidades de medida inercial, y lenguaje de programación aplicado a brazos robóticos.
- Adquirir y procesar por computador las señales de medida inercial procedentes del sensor utilizando software libre.
- Establecer un enlace entre los componentes del sistema: sensores, controlador y ordenador manejando distintas formas de comunicación.
- Diseñar e implementar el algoritmo flexible de movimiento en el robot mediante lenguaje estructurado u orientado a objetos.
- Comprobar la eficiencia en la programación al utilizar unidades de medida inercial, mediante pruebas de funcionamiento del algoritmo en el brazo robótico.

## 1.7. Hipótesis

¿Al implementar el algoritmo flexible en el robot MITSUBISHI MELFA RV-2SDB utilizando unidades de medida inercial se mejorará la eficiencia en la programación?

### 1.7.1. *Variable Independiente*

Algoritmo Flexible en el robot MELFA RV-2SDB

### 1.7.2. *Variable Dependiente*

Eficiencia en la programación

# CAPÍTULO II

# FUNDAMENTOS TEÓRICOS E INVESTIGACIÓN

## 2.1. Anatomía y Biomecánica del Brazo Humano

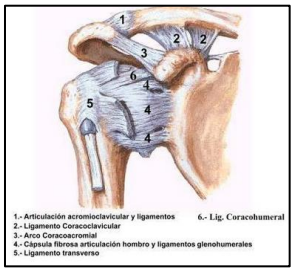
Para el presente trabajo se estudió el comportamiento de las articulaciones hombro, codo y muñeca. Estas articulaciones son de vital importancia ya que proveen información de desplazamientos angulares. (Fisco et al. 2008)

### 2.1.1. *Hombro*

El hombro posee dos huesos primordiales: húmero y omoplato (escápula). Estos huesos están conectados por ligamentos, que son tejido fuerte fibroso, y amalgamados a los músculos por tendones que permiten 5 distintos movimientos que se utilizó en el presente trabajo. (Giraldo, 2009).

**Figura 1**

*Articulaciones acromioclavicular y hombro*

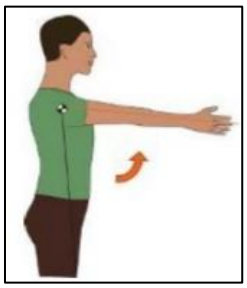



*Nota*. La figura muestra las articulaciones del hombro. (Giraldo, 2009).

**Movimiento de Flexión del Hombro.** Como se puede observar en la Figura 2, este movimiento consiste en flexionar de 0° a 180° los músculos deltoides, coracobraquial y supraespinoso (Kapandji, 2006).

**Figura 2**

*Flexión del hombro*

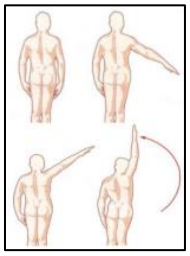


*Nota.* La figura muestra el movimiento de flexión del hombro. (Kapandji, 2006).

**Movimiento de Abducción del Hombro.** Consiste en un movimiento de elevación del brazo donde se coordinan las articulaciones del hombro con los movimientos de la cintura escapular. Llenado a un máximo de 180° como se puede observar en la Figura 3 (Kapandji, 2006).

**Figura 3**

*Abducción del hombro*

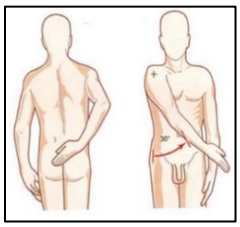


*Nota.* La figura muestra el movimiento de abducción del hombro. (Kapandji, 2006).

**Movimiento de Aducción del Hombro.** Es el movimiento contrario al de Abducción, pero tomando en cuenta que sería imposible una aducción absoluta desde la posición en reposo. Teniendo la postura que se puede observar en la Figura 4 (Kapandji, 2006).

**Figura 4**

*Aducción del hombro*

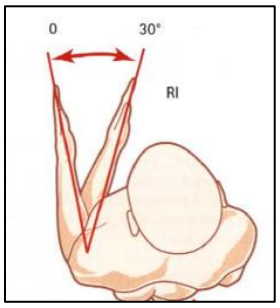


*Nota.* La figura muestra el movimiento de aducción del hombro. (Kapandji, 2006).

**Rotación Interna del Hombro.** Consiste en colocarse en la posición anatómica fisiológica, seguido se realiza un giro de 30° en relación con dicha posición como se puede visualizar en la Figura 5 (Kapandji, 2006).

**Figura 5**

*Rotación interna del hombro*

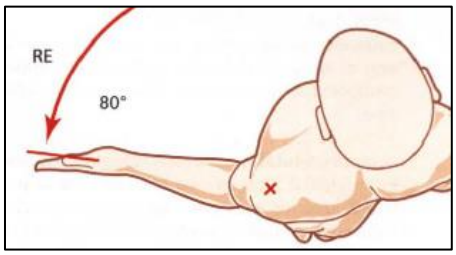



*Nota.* La figura muestra el movimiento de rotación interna del hombro. (Kapandji, 2006).

**Rotación Externa del Hombro.** Consiste en una rotación de amplitud 80°. No es la más usada por el ser humano, pero se la tomo en cuanta por su amplio radio de giro como se puede ver en la Figura 6 (Kapandji, 2006).

**Figura 6**

*Rotación externa del hombro*




*Nota.* La figura muestra el movimiento de rotación externa del hombro. (Kapandji, 2006).

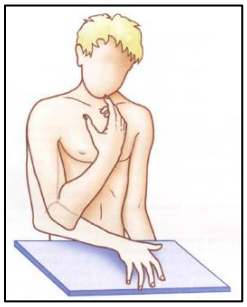
### 2.1.2. *Codo*

Es la articulación que tiene como principal función unir el brazo con el antebrazo. Posee cuatro distintos movimientos que se utilizaron en el presente proyecto (Giraldo, 2009).

**Movimiento de Flexión del Codo.** Es el movimiento clásico que las personas realizan cuando requieren aproximar un alimento hacia la boca. Combina la función del músculo bíceps braquial como se puede observar en la Figura 7 (Kapandji, 2006).

**Figura 7**

*Flexión del codo*

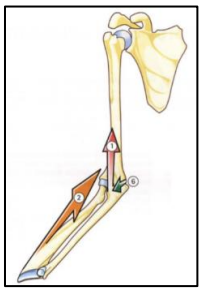


*Nota.* La figura muestra el movimiento de flexión del codo. (Kapandji, 2006).

**Movimiento de Extensión del Codo.** El movimiento de extension del codo se debe a la acción de un solo músculo conocido como triceps branquial especifícamente la acción del músculo ancóneo (Kapandji, 2006). Como se lo puede apreciar en la Figura 8.

**Figura 8**

*Extensión del Codo*

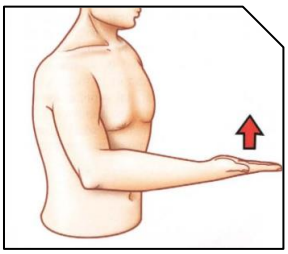


*Nota.* La figura muestra el movimiento de extensión del codo. (Kapandji, 2006)

**Pronación del Codo.** Para lograr esta posición la persona debe estar sentada, de preferencia, luego dirigir la palma de la mano hacia arriba con el pulgar hacia afuera. Como se puede visualizar en la Figura 9.

**Figura 9**

*Pronación del codo*

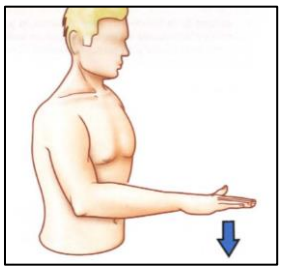


*Nota.* La figura muestra el movimiento de pronación del codo. (Kapandji, 2006)

**Supinación del Codo.** Para lograr esta posición la persona debe estar sentada, de preferencia, luego dirigir la palma de la mano hacia abajo con el pulgar hacia adentro. Como se puede observar en la Figura 10.

**Figura 10**

*Supinación del codo*



*Nota.* La figura muestra el movimiento de supinación del codo. (Kapandji, 2006)

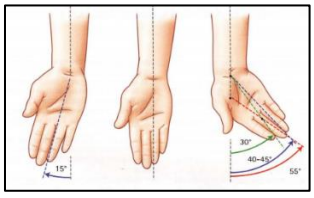
### 2.1.3. *Muñeca*

La muñeca es aquella articulación la cual puede moverse atribuyéndole la habilidad de precisión para recoger cosas, posee dos grados de libertad de forma que se puede orientar hacia cualquier ángulo de dirección en torno a ejes oblicuos, sin embargo, con la rotación del antebrazo añade un tercer grado de libertad, por consiguiente los movimientos naturales de esta articulación quedan definidos por una combinación de estos (Kapandji, 2006).

**Movimiento de extensión de la muñeca.** Consiste en medir la amplitud de movimiento de la inclinación de la muñeca, tal como sigue: la amplitud cubital llega hasta 45° y la radial no sobre pasa los 15°. Estas combinaciones de movimientos pueden ir variando de acuerdo a la referencia de dirección, como se observa en la Figura 11 (Kapandji, 2006).

**Figura 11**

*Extensión de la muñeca*

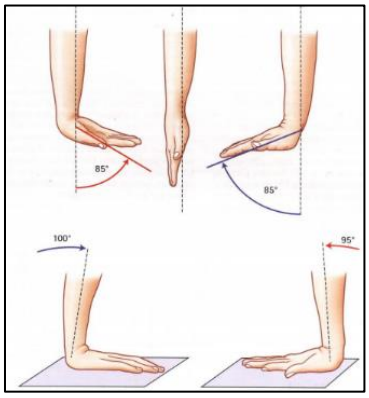



*Nota.* La figura muestra el movimiento de extensión de la muñeca. (Kapandji, 2006)

**Movimiento de flexión de la muñeca.** Consiste en medir los movimientos de flexión de la muñeca, los cuales pueden ser: flexión dorsal y activa llegan hasta 85°, en cuanto para la flexión pasiva y extensión pasiva, ambas llegan a movimientos mayores de 90°; a continuación se muestra la Figura 12 con estos movimientos (Kapandji, 2006).

**Figura 12**

*Flexión de la muñeca*




*Nota.* La figura muestra el movimiento de flexión de la muñeca. (Kapandji, 2006)

## 2.2. Definición de Robot

Es una máquina operada automáticamente que puede suplantar el esfuerzo humano, muchas veces se asemejan a los humanos debido a que desarrollan tareas complejas como caminar, manipular objetos o hablar (Barrientos, et al., 2007).

### 2.2.1. *Robótica*

Es aquellas técnicas de la ingeniería en sus diferentes ramas para diseñar y desarrollar máquinas con movimientos repetitivos hasta robots que pueden imitar tareas humanas o la sustitución de mano de obra de individuos (Sánchez et al., 2007).

### 2.2.2. *Robot Industrial Manipulador*

Es una multiaplicación encargada de manipular de 3 o más ejes, reprogramable, con control automático, predestinado a ser manejado en aplicaciones de sistematización industrial. Incluye al

manipulador que es el sistema mecánico y accionadores, también envuelve al sistema de control que es el software y hardware de control y potencia (Barrientos, et al., 2007).

### 2.3. Brazo Robótico Mitsubishi MELFA RV-2SDB

El brazo robótico de estas características permite efectuar el trabajo productivo de una manera eficiente, debido a su movilidad de ±240° de giro, que permite aprovechar al máximo todo el entorno de trabajo. Así mismo, las herramientas de programación para el desarrollo de pruebas y sistemas para este brazo robótico son muy extensas (Mitsubishi Electric, s. f.).

Por otro lado, la integración de esta tecnología puede realizarse sencillamente con otros elementos de automatización, en vista de que su arquitectura permite acoplarse vía Ethernet, cabe recalcar que la interfaz Ethernet admite conectar una cámara para el desarrollo de captación de imágenes (Mitsubishi Electric, s. f.).

Dentro de este marco es importante reconocer las especificaciones del brazo robótico de la serie SDB, de acuerdo con las especificaciones de proveedor Mitsubishi (2010) son las siguientes:

**Tabla 1**

*Especificaciones de proveedor Mitsubishi*

| Ítem | | Unidades | Especificaciones |
|---|---|---|---|
| **Método de detección de posición** | | | mediante encoder |
| **Longitud del brazo** | Brazo superior | mm | 230 |
| | Antebrazo | | 270 |
| **Operando rango** | Cintura | grados | ± 480 |
| | Hombro | | ± 240 |
| | Codo | | 160 |

| Ítem | | Unidades | Especificaciones |
|---|---|---|---|
| | Giro de muñeca | | ± 400 |
| | Paso de muñeca | | ± 240 |
| | Rotación de muñeca | | ± 720 |
| **Velocidad de movimiento** | Cintura | Grados/s | 225 |
| | Hombro | | 150 |
| | Codo | | 275 |
| | Giro de muñeca | | 412 |
| | Paso de muñeca | | 450 |
| | Rotación de muñeca | | 720 |
| **Velocidad máxima resultante** | | mm/s | 4.400 |
| **Carga** | Máxima | kg | 3.0 |
| | índice | | 2.0 |
| **Repetibilidad de pose** | | mm | ± 0.02 |
| **Temperatura ambiente** | | °C | 0 – 40° |
| **Masa** | | kg | 19 |
| **Admisible momento carga** | Giro de muñeca | N.m | 4. 17 |
| | Paso de muñeca | | |
| | Rotación de muñeca | | |
| **Admisible inercia** | Giro de muñeca | $kg.m^2$ | 0. 18( 0. 27) |
| | Paso de muñeca | | 0. 18( 0. 27) |
| | Rotación de muñeca | | 0. 04( 0. 1) |
| **Punto central del brazo delantero con radio alcanzable** | | mm | 504 |
| **Cableado de herramientas** | | | Entrada manual 4 puntos / salida manual 4 puntos |
| **Tuberías neumáticas para herramientas** | | | Lado principal: Φ4 x 4 (de la base a la sección del antebrazo) |

| Ítem | Unidades | Especificaciones |
|---|---|---|
| **Presión de suministro** | MPa | 0.5 ± 10% |
| **Especificación de protección** | | IP30 (para todos los ejes) |
| **Grado de limpieza** | | - |
| **Color de pintura** | | Gris claro (equivalente a Munsell: 0.08GY7.64 / 0.81) |

*Nota.* (Electrical, 2017)

A continuación se muestra las especificaciones estándar del controlador según las especificaciones del proveedor Mitsubishi (2010):

**Tabla 2**

*Especificaciones estándar Mitsubishi*

| Ítem | | Unidades | Especificaciones | Observaciones |
|---|---|---|---|---|
| **Tipo** | | | CR1DA-771 | |
| **Número de eje de control** | | | Simultáneamente 6 (máximo) | |
| **Capacidad de memoria** | Posiciones programadas y No. de pasos | Paso de punto | 13.000<br>26.000 | |
| | Número de programas | | 256 | |
| **Lenguaje del robot** | | | MELFA-BASIC V<br>o<br>MELFA-BASIC IV | |
| **Método de enseñanza** | | | Método de enseñanza de pose, método MDI | |

| Ítem | | Unidades | Especificaciones | Observaciones |
|---|---|---|---|---|
| **Entrada y salida externa** | Entrada y salida | punto | 0/0 | Max. 256/256 por opción |
| | Entrada / salida dedicada | | Asignado con entrada / salida de uso general | |
| | Entrada de parada especial | punto | 1 | |
| | Entrada/salida de apertura/ cierre manual | punto | Entrada 4 puntos / Salida 0 puntos | Se pueden agregar hasta 4 puntos de salida como una opción |
| | Entrada de parada de emergencia | punto | 1 | 2 contactos son compatibles con el contacto b |
| | Entrada de interruptor de puerta | punto | 1 | 2 contactos son compatibles con el contacto b |
| | Habilitación de entrada de dispositivo | punto | 1 | 2 contactos son compatibles con el contacto b |
| | Salida de parada de emergencia | punto | 1 | |
| | Modo de salida | punto | 1 | |
| | Salida de error del robot | punto | 1 | |
| | Adición del eje de sincronización | punto | 1 | |
| **Interface** | RS – 232C | puerto | | Para expansión como la computadora personal, Sensor de visión |
| | Ethernet | puerto | 1: para T / B, 1: para clientes | 10BASE-T/100BASE-Tx |

| Ítem | | Unidades | Especificaciones | Observaciones |
|---|---|---|---|---|
| | USB | puerto | 1 | Ver. 2.0 Solo función de dispositivo |
| | Ranura dedicada a mano | slot | 1 | Dedicado a la interfaz neumática de mano |
| | Interfaz de eje adicional | canal | 1 | SSCNET III |
| | Interfaz de seguimiento | canal | 1 | Para la conexión del cable del codificador |
| | Opción de slot | slot | 1 | Para instalación de interfaz opcional |
| **Fuente de poder** | Rango de voltaje de entrada | V | 1-fase, AC180 a 253 | |
| | Capacidad de potencia | KVA | 0.5 | No incluye corrientes pico |
| | Frecuencia de suministro de energía | Hz | 50/60 | |
| **Dimensiones del contorno** | | mm | 240(A) x 290(D) x 200(H) | Excluyendo protuberancias |
| **Masa** | | kg | Aproximadamente 9 | |
| **Construcción** | | | Tipo de suelo autónomo<br>Tipo abierto (IP20) | IP20 |
| **Humedad ambiental** | | %RH | 45 a 85 | Sin gotas de rocío |
| **Toma de tierra** | | Ω | 100 o menos | Tierra de puesta a tierra, de clase D |
| **Color de pintura** | | | Gris claro | Munsell 0.08GY7.64/0.81 |

*Nota.* (Electrical, 2017)

De este modo conociendo cada una de las especificaciones tanto del brazo robótico como del controlador de este, es posible tener en cuenta los requerimientos necesarios para el desarrollo de la presente aplicación.

### 2.3.1. *Estación del Robot de FESTO*

La estación cuenta con la función principal de transportar piezas a través de la rampa, para posteriormente ubicarlas en el retenedor de montaje. El controlador es de la marca Mitsubishi que funciona en conjunto con el Brazo robótico RV–2SDB, es potente, ligero y pequeño. De la misma manera el robot típico, permite sujetar piezas y montar en diferentes tipos de piezas, la interfaz de conexión permite que el robot esté listo para adoptar los valores nominales de la red para ejes. MPS Estaciones (s. f.)

La estación del robot de acuerdo con el catálogo del proveedor MPS Estaciones (s. f.) cuenta con varios dispositivos tales como:

- El módulo retenedor el cual posee dos características principales de asentamiento superior e inferior, cuyas funciones son de permitir asentar la pieza independientemente de su orientación y la de tener un pasador de bloqueo, respectivamente.
- Robot RV-2SB con teachbox R32TB, el cual posee 6 grados de libertad de alta precisión, añadido la función de tiempos reducidos para libertad de movimientos ampliados, con un controlador, juego de baterías y cable para programar.

**Tabla 3**

*Especificaciones de electroválvula FESTO*

| Especificación | Observación |
|---|---|
| **Tensión aplicada simultáneamente a ambas bobinas** | La válvula mantiene su posición de conmutación. |
| **Si no se recibe corriente en ambas bobinas** | La válvula ocupa su posición central por acción de muelle. |
| **Ventajas** | |
| **Para los CPE10, CPE14 y CPE18:** | |
| **Configuración sencilla de baterías mediante bloques distribuidores de sólido aluminio para entre 2 hasta 10 posiciones de válvulas o mediante bloques distribuidores modulares de robusto material sintético** | Los cables de conexión NEBV para tamaños CPE10 y CPE14 incluyen una reducción de corriente. Por lo tanto, todas las válvulas CPE tienen un tiempo de utilización del 100 %. |

*Nota.* (Bulletin, 2010)

## 2.4. Definición de Términos Relacionados con la Manipulación

Dentro de los términos relacionados con la manipulación comprenden 3 conceptos:

### 2.4.1. *Teleoperación*

Es la unión de tecnologías que perciben la manipulación o dirección a distancia de un dispositivo por un individuo, es decir, es aquella labor que un humano realiza a distancia sobre un dispositivo sea este de operar o gobernar (Barrientos, et al., 2007).

### 2.4.2. *Telemanipulación*

Es la unión de tecnologías que comprenden la acción o dirección a distancia por un humano de un manipulador. Dicho de otra forma, es la acción que realiza individuo al manejar o mandar a distancia un manipulador (Barrientos, et al., 2007).

### 2.4.3. *Telerrobótica*

Es la unión de tecnologías que abarca la monitorización y reprogramación a distancia de un robot por un ser humano. El cual es denominado telerrobot o robot teleoperado (Barrientos, et al., 2007).

## 2.5. Programación de Robots

La programación de los robots es una de las características principales que los hacen populares para ser implementados en las diferentes aplicaciones que se les puede dar, debido a la reprogramación que estos poseen los hacen de fácil adaptabilidad con el entorno de trabajo. La programación no es más que la secuencia de instrucciones que el robot debe seguir con el fin de llevar a cabo una tarea, estas instrucciones están compuestas por variables que a medida que se va ejecutando el programa alojado en la memoria del sistema se van actualizando, a su vez con la integración de entradas y salidas que poseen los robots hace posible la sincronización con el entorno que lo rodea, para esto existen métodos de programación por un lado la de forma guiada y por otro de procedimiento textual (Barrientos, et al., 2007).

### 2.5.1. *Programación por Guiado o Aprendizaje*

Como su nombre lo indica es aquella forma de programar en la que es como "enseñar" al brazo robótico los procedimientos para realizar una determinada tarea, dichos movimientos son almacenados en la unidad de control del robot, de esta manera se puede distinguir tres tipos de programación por guiado; la primera forma de programar es a través del guiado pasivo directo, la cual consiste en guiar manualmente el extremos del robot e ir moviendo en los puntos deseados siguiendo una trayectoria adecuada; la segunda forma es aquella que es guiado pasivo por maniquí, la cual hace alusión a un doble del robot más liviano y más fácil de manipular y de iguales configuraciones que el robot original; la tercera se trata del guiado activo, este método permite programar el robot manipulando sus articulaciones a través de un joystick situado en el panel de programación (Barrientos, et al., 2007).

### 2.5.2. *Programación Textual*

Este tipo de programación es la que permite al robot hacer una tarea determinada mediante el uso de códigos escritos en leguaje formal, en las que se determina los parámetros o acciones que debe llevar a cabo, para esto es importante mencionar que se distinguen tres niveles de programación textual: la primera de estas es a nivel robot, en esta se pueden utilizar varios lenguajes de programación, en cada una de las líneas de programación se define parámetros como la velocidad, posición, precisión, etc.; a nivel de objeto es la segunda de las tres clasificaciones, en estas las sentencias de código son más simple que la anterior debido a que las instrucciones se lo realizan con respecto a los objetos a manejar, que posteriormente un planificador ejecutará las instrucciones consultando en la base de datos; la tercera programación es a nivel de tarea, en esta

se ejecuta una sola sentencia en la que el robot realiza la tarea, en vez de programar como debe hacerlo. (Barrientos, et al., 2007)

### 2.6. Sistema de Captura de Movimiento Inercial

Los sistemas de captura de movimiento consisten en que, a través de sensores se registra los movimientos anatómicos incluidos ángulos y posiciones de las articulaciones del cuerpo humano, para posteriormente reconstruirlo de manera digital; las aplicaciones que se le pueden atribuir a estos sistemas van desde animación digital hasta captura de rendimiento deportivo. De este modo los sistemas se pueden clasificar en, sistemas basados en marcadores y sistemas "sin marcadores", la diferencia entre cada uno es que, el primero si bien la captura es más exacta utiliza un sistema muy invasivo de sensores, mientras que para en el segundo sistema, la captura de movimiento es de forma remota es decir no se utiliza equipamiento adicional de sensores aunque la captura de movimiento es menos precisa; cabe recalcar que estas técnicas para la captura de movimiento también son llamadas Mocap. (Unzueta, 2014)

Dentro de este marco los métodos y sistemas para la captura de movimiento, se clasifican primeramente por sistemas ópticos, los cuales utilizan un sistema de cámaras para que de esta manera las capturas puedan traslaparse unas con otras, de esta manera acercarse más fielmente a los movimientos reales de la persona, así mismo para otras aplicaciones donde se requiera ser más preciso en las capturas de movimientos, se utilizan una serie de marcadores dedicados para cada aplicación que se requiera. (López, 2016)

Por otra parte los sistemas de captura no ópticos, es un tipo de tecnología donde utilizan algoritmos, modelos biomecánicos, y sensores inerciales, de este último los datos generados a

través de la captura de movimiento son grabados y transmitidos inalámbricamente hacia un ordenador, esto supone una gran ventaja debido a que registran en total seis grados de libertad y en tiempo real, en la gran mayoría de estos sistemas los datos obtenidos se los representa en unidades de medida de inercia por sus siglas IMU, el cual debido a su gran versatilidad proporciona combinaciones de datos de variaciones de rotación tales como el magnetómetro, acelerómetro y giroscopio. (López, 2016)

Ahora bien, los sistemas de captura magnéticos debido a su portabilidad pueden ser fácilmente adaptados para la realización de captura de movimientos, sin embargo, la precisión es muy baja y solo son utilizados para ciertas aplicaciones específicas. (Gómez Echeverry et al., 2018)

Finalmente se indica una tabla de comparaciones en las que se representa las ventajas y desventajas al momento de utilizar los diferentes sistemas de captura de movimiento, de acuerdo con Gómez Echeverry et al. (2018):

**Tabla 4**

*Desventajas de sistemas de captura de movimiento*

| Tecnología | Ventajas | Desventajas |
|---|---|---|
| **Sistemas de captación de movimiento ópticos con marcadores** | Alta precisión | El espacio para utilizar es limitado |
| | Se puede estudiar movimientos complejos | No puede ser utilizado para estudios de movimientos deportivos |
| | Ampliamente utilizado en animación 3D | Debido a su robustez necesita de un espacio adecuado |

| Tecnología | Ventajas | Desventajas |
|---|---|---|
| | Útil para tecnologías relacionadas con realidad aumentada y virtual | Se debe preparar el espacio de trabajo |
| | Se puede procesar los datos sin ningún inconveniente | Alto costo del equipo |
| **Sistemas de captación de movimiento ópticos sin marcadores** | Es de bajo costo frente a los demás sistemas | Baja precisión de parámetros angulares |
| | Posee telerehabilitación | No permite realizar estudios de alta complejidad |
| | Fácil manipulación | |
| | Se puede procesar los datos sin ningún inconveniente | |
| | Posee muy buena precisión en parámetros espaciotemporales | |
| **Sistemas IMU y magnéticos** | Son sistemas portables | Tiene baja precisión |
| | Permite hacer estudios deportivos | No permite realizar estudios de alta complejidad |
| | Fácil manipulación | |
| | Permiten hacer estudios fuera de los laboratorios de prueba | |
| | Es de bajo costo frente a los demás sistemas | |

*Nota.* (López, 2016)

### 2.6.1. *Sensores de Medición Inercial*

Estos sensores se clasifican en tres tipos, en primera instancia se tiene el giroscopio, el cual puede ser de tipo mecánico, óptico y de tipo MEMS; el giroscopio de tipo MEMS utilizado para el presente trabajo, es un sensor de bajo costo e igualmente con este dispositivo se puede determinar la orientación y la velocidad angular, no obstante su precisión puede llegar a ser más baja que la de tipo óptico (López, 2016).

Siguiendo con la clasificación de los sensores de medición, se tiene el acelerómetro el cual se clasifica en tres tipos, el de tipo mecánico, de estado sólido, y de tipo MEMS; los acelerómetros de tipo MEMS que se utilizó en el presente trabajo, utilizan un desplazamiento llamado pick-off, el cual obedece a la segunda ley de Newton, en la que intervienen la masa y la aceleración. Además, pueden utilizar vibraciones causadas a un elemento, en el que se mide la diferencia de frecuencias causadas, son pequeños, livianos y consumen bajas potencias cuando entran en operación. (López, 2016)

Por último los magneteorológicos, son aquellos que miden intensidades de campo magnético, de la misma manera que los anteriores sensores, estos se pueden dividir en varios tipos, los cuales se distinguen de tipo mecánico, óptico y tipo MEMS; los de tipo MEMS que se utilizó en el presente trabajo, tienen características de ser pequeños y de fácil manipulación (López, 2016).

## 2.7. Herramientas Matemáticas para la Localización Espacial

Para que el robot pueda realizar tareas de manipulación que se le ordena debe conocer la posición y orientación del elemento a manipular con respecto a la base del robot. Para ello, se debe contar con diversas herramientas matemáticas que le permitan realizar las especificaciones mencionadas.

### 2.7.1. *Representación de la posición*

Para tratar de ubicar un cuerpo rígido en el espacio se debe conocer su posición y orientación. Ambos aspectos deben ser ligados a un sistema de referencia definido, utilizando herramientas matemáticas que faciliten el proceso. Dentro de un plano bidimensional, la posición de un cuerpo rígido necesita de dos grados de libertad y por ello la visión del cuerpo queda

determinada por 2 módulos independientes. En el espacio tridimensional será indispensable emplear 3 componentes (Barrientos, et al., 2007).

### 2.7.2. *Sistema cartesiano de referencia*

Estos sistemas son definidos a través de ejes perpendiculares entre si con un origen fijo, nombrados sistemas cartesianos. En el caso de trabajar en el plano bidimensional, el sistema OXY es determinado por dos vectores coordenados OX y OY perpendiculares y con un punto en común O. En el caso de un plano tridimensional, el sistema OXYZ estará combinado por una terna ortonormal a derechas de vectores unitarios OX, OY y OZ (Barrientos, et al., 2007).

### 2.7.3. *Coordenadas cartesianas*

Para trabajar en un plano con sistema coordenado OXY asociado, un punto será indicado por las unidades $(x, y)$ según los ejes del sistema. Este punto asocia un vector $P(x, y)$, que inicia en el origen O de dicho sistema hasta llegar al punto a. Por tal razón, la posición del extremo del vector p se caracteriza por 2 componentes $(x, y)$ llamadas coordenadas cartesianas y que son proyecciones del vector p sobre los ejes OX y OY (Barrientos, et al., 2007).

### 2.7.4. *Representación de la orientación*

La representación de la orientación en el espacio tridimensional aparece determinada por tres grados de libertad o tres componentes independientes. De tal forma, referir de forma natural la ubicación de un objeto con relación a un sistema de referencia, es frecuente fijar un objeto a un nuevo sistema y posteriormente estudiar la correlación espacial entre los dos sistemas. Es decir,

esta relación aparecerá cedida por la posición y orientación del sistema agrupado al objeto referente al de referencia (Barrientos, et al., 2007).

### 2.7.5. *Matrices de rotación*

Son un método extendido dentro de las descripciones de orientaciones, debido al bienestar que suministra el uso del algebra matricial. Dentro de un sistema de referencia OXY y OUV con un mismo origen O, siendo el sistema OXY el de referencia fija y el OUV el sistema móvil, solidario al objeto (Barrientos, et al., 2007).

### 2.7.6. *Ángulos de Euler*

Estos ángulos figuran los valores de los giros a realizar sobre 3 ejes ortogonales entre sí, de modo que girando continuamente el sistema OXYZ sobre estos ejes octonormales los valores de $\varphi$, $\theta$, $\psi$, da como resultado el Sistema OUVW (Barrientos, et al., 2007).

### 2.7.7. *Matrices de transformación homogénea*

Son aquellas que admiten una representación conjunta, facilitando su uso a través del álgebra matricial (Barrientos, et al., 2007).

### 2.7.8. *Coordenadas y matrices homogéneas*

En principio las coordenadas homogéneas tienen por objetivo representar la orientación y posición de un sólido, de tal forma que un vector de n dimensiones se representa en n+1 dimensiones; aquella dimensión que se le aumenta al vector original, se le conoce como un parámetro de valor de escala. Una vez conocido este concepto, aparece la aplicación de matrices homogéneas, que no es más que una matriz cuadrada 4X4 que sirve para representar un sistema de

coordenadas diferente, a partir de una vector de coordenadas homogéneas (Barrientos, et al., 2007).

### 2.7.9. *Aplicación de las matrices homogéneas*

Dentro de las aplicaciones de las matrices, están en poder representar a través de un sistema de referencia una matriz de traslación y rotación; de la misma forma permite representar un vector r dado en un sistema determinado de coordenadas a otro sistema de referencia; otras de las aplicaciones es que dado un vector r se puede trasladar y rotar otro sistema de referencia (Barrientos, et al., 2007).

De igual manera las dos aplicaciones fundamentales de las matrices homogéneas es la traslación y rotación; por un lado la traslación como su nombre lo indica, traslada un vector de coordenadas, de tal manera que dicho vector dado un sistema de coordenadas determinado, aplicando la traslación tendrá coordenadas $p = p_x i + p_y j + p_z k$, con respecto a un sistema OXYZ; finalmente en la rotación se aplica el mismo principio, pero esta vez rotando en referencia al sistema OXYZ (Barrientos, et al., 2007).

## 2.8. Herramientas Computacionales

### 2.8.1. *Arduino*

Es una plataforma en la que se permite programar a través de código abierto el cual posee licencia FREE, basados tanto en su software como su hardware; a su vez son fáciles de usar debido a la amplia información que existe sobre el mismo, aunado de que es económico, se adapta a

cualquier sistema operativo; su software es extensible y el lenguaje de programación se ejecutan es través de la biblioteca C++ y C AVR. (ARDUINO, 2018)

### 2.8.2. *Matlab*

Es un software matemático el cual debido a su lenguaje de programación permite realizar códigos basados en matemáticas computacionales, a su vez el software hace posible efectuar algoritmos, analizar datos, y crea tanto modelos como aplicaciones, adicionalmente el software cuenta con licencia para estudiantes, lo que hace posible el libe manejo del mismo; así mismo la herramienta de Toolbox son librerías de Matlab que están orientadas al cálculo técnico, las cuales están abiertas al público en general para ser descargadas. (Mathworks, s. f.)

De la misma manera la librería Robotics toolbox Peter Cork, es uno de los varios paquetes que proporciona el Toolbox de Matlab; este en específico es útil para la simulación y estudios de brazos robóticos, que permite analizar funciones tales como generación trayectorias, dinámica y cinemática, a su vez como este toolbox es de libre descarga es intuitivo y enfocado a la enseñanza. (Corke, 2017)

Por otro lado, el Robotics toolbox de ARTE, está orientado al ámbito de la docencia por lo cual es libre descarga para el público, esta herramienta permite desde representación D-H del robot hasta la programación de forma interactiva del mismo. (Hernandez, 2013)

### 2.8.3. *MeshLab*

Es un software de código abierto, es decir debido a su licencia FREE permite su libre descarga. El software posee potentes herramientas para texturizar, inspeccionar, limpiar, renderizar, curar, editar y convertir mallas triangulares 3D. (Corsini et al., 2012)

### 2.8.4. *SolidWorks*

Es un software CAD el cual permite modelar elementos mecánicos en 2D o 3D, al igual que Matlab posee una licencia dedicado a la docencia, el cual permite el libre manejo de este software sin fines comerciales. (SolidWorks, s. f.)

### 2.8.5. *Rhinoceros 3D*

Es una herramienta la cual permite modelar elementos en 3D con gran detalle y precisión, de la misma manera posee la licencia TRIAL que autoriza el libre uso del software para ámbitos educativos. (Rhinoceros, s. f.)

### 2.8.6. *EasyEDA*

Es un conjunto de herramientas EDA gratuitas dedicadas al diseño de circuitos, ya que su licencia FREE permite su libre uso, además es compatible con todos los sistemas operativos (EasyEDA, s. f.).

## 2.9. Cinemática del Robot

Estudia el movimiento del robot con relación a un sistema de referencia sin discurrir las fuerzas que interceden. De tal forma, la cinemática se alarma por la representación analítica del

movimiento espacial del robot como una función del tiempo y específicamente por la relación en el extremo final del robot entre la posición y orientación con valores que despojan sus coordenadas articulares (Barrientos, et al., 2007).

### 2.9.1. *El problema cinemático directo*

El problema directo está basado en fijar cual es la posición y orientación del extremo final del robot, en relación a un sistema de coordenadas referente una vez distinguidos los valores de las articulaciones y los medidas geométricas de los elementos del robot (Barrientos, et al., 2007).

### 2.9.2. *Resolución del problema cinemático directo mediante matrices de transformación homogénea*

La resolución de los problemas cinemáticos directos se enfoca en obtener una matriz de transformación homogénea T que se relacione en la posición y orientación del extremo del robot referido al sistema referencial fijo situado en la base de este. La matriz T funciona mediante coordenadas articulares (Barrientos, et al., 2007).

### 2.9.3. *Algoritmo de Denavit Hartenberg para la obtención del modelo cinemático Directo*

Es un método matricial que permite encontrar la localización que debe tomar cada sistema de coordenadas $\{S_i\}$ unido a cada eslabón i de una cadena articulada, para poder sistematizar la creación de ecuaciones cinemáticas de la cadena completa. Eligiendo el sistema de coordenadas asociados a cada eslabón como la representación propuesta por D-H, es viable pasar

consecutivamente a través de 4 transformaciones básicas que dependen a las particularidades geométricas del eslabón (Barrientos, et al., 2007).

1. Rotación alrededor del eje $Z_{i-1}$ un ángulo θ
2. Traslación a lo largo de $Z_{i-1}$ una distancia $d_i$; vector $d_i$ (0,0, $d_i$ ).
3. Traslación a lo largo de $X_i$ una distancia $a_i$; vector $a_i$ ($a_i$ ,0,0).
4. Rotación alrededor del eje $X_i$ un ángulo $\alpha_i$.

Estas transformaciones describen al sistema móvil, ya que el producto de matrices no es conmutativo, las transformaciones se deben ejecutar en el orden señalado (Barrientos, et al., 2007).

### 2.9.4. *Cinemática Inversa*

El propósito de la cinemática inversa es hallar los valores de coordenadas que se necesite en las articulaciones del robot, para establecer una posición y orientación en el extremo del mismo. En este sentido, al hablar de cinemática inversa quiere decir que es obligatoriamente saber la configuración del robot. En cierta medida la resolución de la cinemática inversa queda simplificada debido a que, la mayoría de los robots posee solo tres grados de libertad, y de la misma manera el extremo del robot se considera a giros sobre ejes. Para ciertos robots con 6 grados de libertad se los puede dividir en análisis de posicionamiento, sin tomar en cuenta el extremo del robot que son dedicados a la orientación, es decir se los desarrolla de manera independiente los primeros grados de libertad y luego continuando con los siguientes, a esta práctica se la llama desacoplo cinemático (Barrientos, et al., 2007).

## 2.10. Control Cinemático

El control cinético instaura las trayectorias que seguirá cada articulación del robot a lo extenso del tiempo para conseguir los objetivos fijados por el usuario, estas trayectorias se eligen considerando las restricciones físicas propias de los accionamientos y a diversos criterios de calidad de recorrido, como suavidad o precisión de la misma (Barrientos, et al., 2007).

### 2.10.1. *Tipos de trayectorias*

El robot para poder realizar tareas específicas debe moverse desde un punto inicial hasta un final. El movimiento que realiza en este transcurso puede realizarlo de maneras infinita por trayectorias espaciales. Por ello, se debe considerar por su sencillez de ejecución o por su utilidad y aplicación a diversas tareas disponer de trayectorias punto a punto o continuas (Barrientos, et al., 2007).

### 2.10.2. *Trayectorias punto a punto*

Dentro de los tipos de trayectorias Barrientos, et al., (2007) afirma que cada articulación va evolucionando desde su posición inicial hasta la final, habitualmente cada actuador trata de transportar a su articulación al punto de destino en el mínimo tiempo.

## 2.11. Aplicaciones de los Robots

En la actualidad los robots se utilizan de manera amplia en la industria dentro de los procesos de manufactura, ya que la definición de robot industrial indica que es multifuncional, no obstante, la practica ha confirmado que su adaptación es óptima en determinados procesos donde es más beneficioso. Por otro lado, la aparición de robots propuestos a aplicaciones no industriales

llamados genéricamente robots de servicio que ayudan a eximir al ser humano de tareas peligrosas o ampliar sus capacidades, los mismos no son elaborados a gran escala. A continuación, se redacta sobre las aplicaciones industriales más frecuentes del robot, ventajas frente a otras alternativas (Barrientos, et al., 2007).

### 2.11.1. *Aplicaciones industriales de los robots*

El establecimiento de un robot industrial en un definitivo proceso requiere un minucioso estudio previo donde se examine ventajas e inconvenientes que conlleva la introducción del robot. Referente al tipo de robot a utilizar, habrá que reflexionar aspectos de diversas índoles como área de acción, velocidad de carga, capacidad de control, coste etc. (Barrientos, et al., 2007)

### 2.11.2. *Aplicación de materiales pintura*

La aplicación de materiales de pintura es el terminado de superficies por recubrimiento de un material ya sea por fines decorativos o de protección. Dentro de los métodos de fabricación es una parte muy crítica (Barrientos, et al., 2007).

El procedimiento es cubrir una superficie con una mezcla de aire y el material elegido, este último pulverizado a través de una pistola, para obtener una homogeneidad en el reparto del material, vigilado visualmente por el operario. Por otra parte, el entorno donde se aplica la pintura es peligroso al tener poco espacio y ser un lugar cerrado, con una atmósfera tóxica, alto nivel de ruido y riesgo de incendio. Estas situaciones han hecho de la pintura y operaciones similares, un proceso de robotización, empleando un robot que elimina los inconvenientes ambientales y permita ganar en la homogeneidad, en la calidad del acabado, ahorro de pintura y productividad (Barrientos, et al., 2007).

Los robots de pintura son precisos para este fin, el método de programación preferido es el de aprendizaje o guiado. Suelen ser robots articulares, ligeros, con 6 o más grados de libertad que proyectan pintura en todos los huecos de la pieza, tiene protecciones especiales para protegerse de las partículas en suspensión adentro de la cabina de pintura y sus diferentes consecuencias (Barrientos, et al., 2007).

**Figura 13**

*Robot de pintura*

*Nota.* La figura muestra un robot de pintura. (Barrientos, et al., 2007).

### 2.12. Requerimientos del Operador

Al ser un tema de eficiencia y eficacia de un proceso, los primeros operadores son las personas que están relacionados con la programación de distintos robots manipuladores industriales como los estudiantes y profesores de la carrera de Mecatrónica, que encuentran distintas necesidades durante el control y programación del brazo robótico.

Los requerimientos planteados por el operador son:

- Una interfaz para interactuar con el brazo robótico.
- Programación sencilla.
- Los dispositivos electrónicos a utilizar deben ser fáciles de implementar y con estructura robusta.
- Debe ser de un bajo costo.

## 2.13. Requerimientos Técnicos

En base a las distintas peticiones realizadas por los operadores se define los siguientes requerimientos:

Interfaz Hombre-Máquina interactivo y sencillo de utilizar.

- El algoritmo debe poseer distintos modos de operación como tele operación y programación.
- Sensores de posiciones angulares fáciles de implementar.
- Sensores con conexión inalámbrica y con autonomía de carga eléctrica.
- Placa de circuito impreso.
- Estructura, materiales y dimensiones para la base de la herramienta de trabajo.
- Conexión sencilla.
- Sistema de bajo costo.

## 2.14. Propuesta e Identificación de Sistemas

En esta sección se propone la utilización de Unidades de Medida Inercial colocadas en la extremidad superior aprovechando los movimientos biomecánicos que proporciona, logrando

captar ángulos de desplazamiento simultáneamente para luego ser procesados y enviados a un ordenador. Por medio de un algoritmo de control basado en transformaciones homogéneas específicas, se ejecutará trayectorias que el operador desee almacenar en el brazo robótico con el fin de realizar una aplicación. Dicha aplicación permitirá comprobar que el algoritmo reduce el tiempo de programación y a la vez tiene un uso en los procesos de fabricación, como es el caso del recubrimiento de un cierto material (Pintura).

Debido a la dificultad del trabajo es necesario dividir en cuatro sistemas que se encuentra estrechamente relacionados entre sí, siendo estos: sistema mecánico, electrónico, neumático y computacional, lo que permitirá satisfacer los requerimientos del operador.

### 2.14.1. *Sistema Mecánico*

El sistema mecánico fue desarrollado para garantizar la sujeción de la herramienta de trabajo al sexto eje del brazo robótico además de ser liviana, de tamaño reducido y fácil montaje.

**Figura 14**

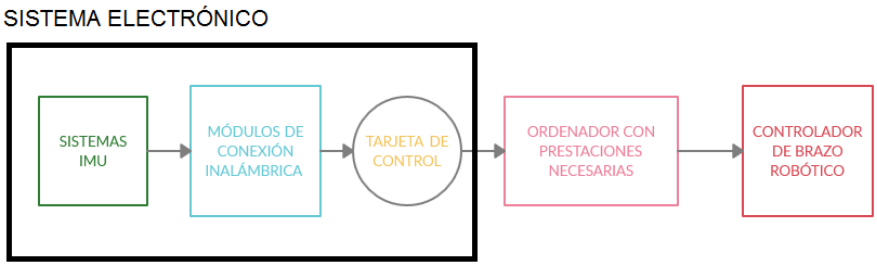
*Sistema Mecánico*

### 2.14.2. *Sistema Electrónico*

El sistema electrónico está constituido por distintos dispositivos como los Sistemas de Medida Inercial, módulos de conexión inalámbrica para la recepción de datos, una tarjeta de control que funciona como servidor de todos los datos adquiridos por los sensores, indicadores visuales para confirmar el estado de conexión de los sensores, el ordenador que posea las prestaciones necesarias para la recepción tratamiento y envió de datos hacia el controlador del brazo robótico.

**Figura 15**

*Sistema Electrónico*




### 2.14.3. *Sistema Computacional*

Debido a que el sistema computacional es el encargado de unir todos los sistemas, se requiere de un ordenador (hardware) con prestaciones que garanticen la interacción de instrucciones (software) capaces de generar archivos ejecutables para los dispositivos electrónicos (sensores, tarjeta de control). Además, se debe realizar cálculos matriciales con datos proporcionados de los sensores por lo que se necesita un programa matemático especializado en resolver problemas matemáticos de localización espacial y cinemática directa e inversa del robot,

comandado por una interfaz programable. Adicionalmente es necesario de un software capaz de proporcionar trayectorias basadas en una superficie con geometría compleja.

**Figura 16**

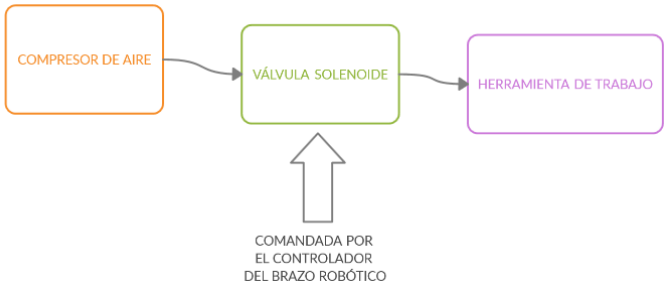
*Sistema Computacional*


### 2.14.4. *Sistema Neumático*

El sistema neumático posee un compresor, una válvula solenoide que se encuentra comandado por el controlador del brazo robótico y una herramienta de trabajo que será una pistola de aire que pulveriza pintura y estará acoplada al sistema mecánico.

**Figura 17**

*Sistema Neumático*



# CAPÍTULO III

# DISEÑO DE SISTEMAS PARA EL ALGORITMO FLEXIBLE DE CONTROL

En este capítulo se explica el proceso de desarrollo de los distintos sistemas que utiliza el algoritmo flexible implementado en el brazo robótico utilizando la metodología de diseño de Hans Gugelot. (Anzora, 2010 ). La metodología parte de la etapa de información e investigación, previamente argumentada en el Capítulo II, donde se recopila información de las instalaciones donde se desarrolló el proyecto, los requerimientos del operador y los fundamentos teóricos. Para la etapa de diseño y selección, intervienen conceptos científicos para la configuración y elaboración de los componentes de cada sistema justificando cada decisión.

## 3.1. Selección de Componentes del Sistema

Una vez identificado todos los sistemas que son necesarios para implementar un algoritmo flexible de programación (sección 2.14), es primordial seleccionar los componentes que cubran cada una de las necesidades planteadas por el operador teniendo en cuenta la disponibilidad de los mismos en el mercado y su sencilla adquisición.

### 3.1.1. *Selección de Componentes del Sistema Mecánico*

El sistema mecánico está compuesto por una base que será el portaherramientas del efector final del brazo robótico.

**Portaherramientas.** Debido a que la herramienta de trabajo debe encontrarse sujeta al sexto eje del brazo robótico y este posee un sistema de sujeción conformado por cuatro agujeros roscados (M5 x 0.8mm), se diseñó una base mediante software CAD (SolidWorks) para su

posterior impresión en 3D. Debido a esto se presentan los siguientes materiales que se tomó en cuenta para su elaboración.

**Tabla 5**

*Características de los materiales para impresión 3D*

| Características | ABS (Acrilonitrilo Butadieno Estireno) | PLA Fibra de carbono Proto-Pasta | PLA (Ácido poli láctico) |
|---|---|---|---|
| **Tecnología de fabricación** | Impresión 3D por adición | Impresión 3D por adición | Impresión 3D por adición |
| **Costo por rollo** | $24 | $32 | $24 |
| **Propiedades físicas** | Temperatura de fusión 215 °C.<br>Es un polímero compuesto por acrilonitrilo, butadieno y estireno (Terpolímero)<br>60 MPa (Flexión)<br>42 MPa (Tracción)<br>2GPa (Módulo Elástico) | Temperatura de fusión 195°C.<br>Elaborado a base de materias primas 85% y un 15% de fibra de carbono.<br>82.5 MPa (Flexión)<br>65 MPa (Tracción)<br>2.9 GPa (Módulo Elástico) | Temperatura de fusión 180°C.<br>Elaborado a base de materias primas naturales y renovables (maíz).<br>100 MPa (Flexión)<br>34 MPa (Tracción)<br>3.5 GPa (Módulo Elástico) |

*Nota.* (Makeitfrom, 2015)

Los materiales para la Impresión 3D del portaherramientas presentan similares características es necesario ponderarlos bajo diferentes criterios para su selección.

**Tabla 6**

*Evaluación de los materiales para impresión 3D*

| **Evaluación** | | **ABS (Acrilonitrilo Butadieno Estireno)** | | **PLA Fibra de carbono Proto-Pasta** | | **PLA (Ácido poliláctico)** | |
|---|---|---|---|---|---|---|---|
| **Criterios de selección** | Peso (%) | Calificación (0-5) | Evaluación ponderada | Calificación (0-5) | Evaluación ponderada | Calificación (0-5) | Evaluación ponderada |
| **Límite elástico** | 35 | 3 | 1.05 | 4 | 1.4 | 5 | 1.75 |
| **Resistencia a la tracción y flexión** | 30 | 4 | 1.2 | 4 | 1.2 | 3 | 0.9 |
| **Costo** | 15 | 3 | 0.45 | 2 | 0.3 | 5 | 0.75 |
| **Accesibilidad al material** | 20 | 5 | 1.2 | 3 | 0.6 | 5 | 1 |
| **Total** | 100 | --- | 3.9 | --- | 3.5 | --- | 4.4 |
| **Ponderación** | | 2 | | 3 | | 1 | |
| **Aprobación** | | No | | No | | Si | |

Analizando la ponderación obtenida en la Tabla 6 se decidió ocupar es el PLA, debido a que es utilizado comúnmente en impresiones 3D cuando no se requiere altos valores de resistencia, como es el caso del portaherramientas, que su mayor carga es soportar el peso de la pistola de gravedad que es aproximadamente 120gr.

### 3.1.2. *Selección de Componentes del Sistema Electrónico*

Para la selección de las partes del sistema eléctrico se consideró en utilizar componentes comerciales adaptables y reprogramables que ocupen software libre cumpliendo varios requerimientos técnicos analizando sus dimensiones y costos.

**Sensores.** Este elemento es el componente principal del sistema electrónico, debido a que capta posiciones angulares de la biomecánica de la extremidad superior y las envía a la tarjeta de control. Se los localizó en tres puntos específicos (muñeca, codo, hombro) para aprovechar los grados de libertad que poseen. Además, permitirá situar dos áreas de trabajo para el robot, la primera localizada en su posición original en la Estación de Trabajo de FESTO y la segunda desplazada hacia el lado contrario de la Estación, para evitar el contacto con los dispositivos y poder realizar la aplicación de pintura.

Los sensores IMU considerados como alternativas para el presente proyecto se pueden observar en la Tabla 7.

**Tabla 7**

*Sensores IMU considerados y sus características*

| Características | smSFM1 - 10 DOF BLE IMU Module | Tactigon ONE V1.0 |
|---|---|---|
| **Conectividad** | -BLE<br>-USB 2.0 | BLE (Bluetooth Low Energy)<br>-Micro USB |
| **Baterías** | Polímero de litio recargable | Li-Ion recargable |
| **Memoria para código** | 256Kb flash | 512Kb flash |

| Características | smSFM1 - 10 DOF BLE IMU Module | Tactigon ONE V1.0 |
|---|---|---|
| **IMU** | -Acelerómetro 3 ejes<br>-Giroscopio 3 ejes<br>-Magnetómetro 3 ejes | -Acelerómetro 3 ejes<br>-Giroscopio 3 ejes<br>-Magnetómetro 3 ejes |
| **Lenguaje de programación** | Python<br>Qt<br>C++ | Arduino |
| **Dimensión** | 36.8 x 24.0x5mm | 46.9x15.2x5.5 mm |
| **Precio sin envío** | $90 | $80 |

*Nota.* (Olmedo, 2019)

Acorde a las características presentadas, se pudo evidenciar que el sensor Tactigon ONE V1.0 y el SMSFM1 poseen las prestaciones necesarias para cumplir con los requerimientos técnicos como es el caso de conexión inalámbrica, batería recargable y lecturas de posiciones angulares. Para la selección del sensor se tomó en cuenta los siguientes criterios:

**Tabla 8**

*Evaluación de los sensores IMU*

| Evaluación | | smSFM1 - 10 DOF BLE IMU Module | | Tactigon ONE V1.0 | |
|---|---|---|---|---|---|
| **Criterios de selección** | Peso (%) | Calificación (0-5) | Evaluación ponderada | Calificación (0-5) | Evaluación ponderada |
| **Conectividad** | 20 | 4 | 0.8 | 4 | 0.8 |
| **Autonomía** | 20 | 4 | 0.8 | 4 | 0.8 |
| **Costo** | 10 | 2 | 0.2 | 4 | 0.4 |

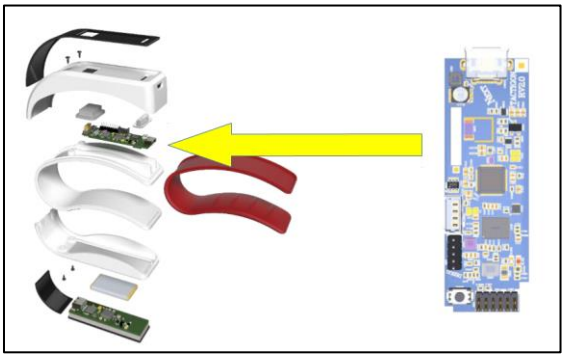
| Evaluación | | smSFM1 - 10 DOF BLE IMU Module | | Tactigon ONE V1.0 | |
|---|---|---|---|---|---|
| **Criterios de selección** | Peso (%) | Calificación (0-5) | Evaluación ponderada | Calificación (0-5) | Evaluación ponderada |
| **Cantidades de señales** | 30 | 5 | 1.5 | 5 | 1.5 |
| **Recursos para la programación** | 20 | 3 | 0.6 | 4 | 0.8 |
| **Total** | 100 | --- | 3.9 | --- | 4.3 |
| **Ponderación** | | 2 | | 1 | |
| **Aprobación** | | No | | Si | |

La calificación de los dos sensores mencionados anteriormente es muy similar, pero se eligió el sensor Tactigon ONE V1.0 debido a su lenguaje de programación en Arduino. Criterio muy importante, ya que Tactigon desarrolló una librería especial en Arduino para facilitar la toma y envío de datos hacia distintos dispositivos con la misma conectividad.

Incluso la empresa Tactigon desarrolló el producto T-Skin, viendo la necesidad de capturar los movimientos angulares producidos por la muñeca. Creando así un dispositivo ergonómico que se coloca en la mano y además posee cuatro botones para poder utilizarlos como señales digitales para controlar diferentes tareas.

**Figura 18**

*Tactigon SKIN y The Tactigon ONE V1.0*



*Nota.* La figura muestra el microcontrolador que posee el sensor T-SKIN. (Tactigon, 2020)

**Tarjeta de control.** Este elemento es el encargado de recibir las señales enviadas a través de la conexión inalámbrica por parte de los sensores, mediante la ejecución de un algoritmo envía los datos procesados al ordenador. Se ha considerado las tarjetas: Arduino MEGA y Arduino DUE.

**Tabla 9**

*Tarjetas de control consideradas y sus características*

| Características | Arduino DUE | Arduino MEGA |
|---|---|---|
| **Voltaje de operación** | 3.3VDC | 5VDC |
| **Controlador** | ARM Cortex M3<br>32 bits | Atmega 2560<br>8bits |
| **Pines digitales** | 54(12PWM) | 54(15PWM) |
| **Frecuencia del reloj** | 84Mhz | 16Mhz |
| **Entradas análogas** | 12 | 16 |
| **Tipo de conexión** | Puerto serie | Puerto serie |
| **Puertos Serie hardware** | 4 | 4 |
| **Precio** | $35 | $15 |

*Nota.* (arduino.cl, 2019)

En la Tabla 9 Se muestran las características técnicas de las posibles tarjetas de control que se pueden emplear. El Arduino DUE otorga las prestaciones necesarias con una frecuencia de reloj mucho más alta que el promedio, incluso un controlador con una resolución de 32 bits. Arduino Mega posee 4 puertos seriales y un controlador de gama moderada y a un precio accesible. Para seleccionar la tarjeta de control se tomaron en cuenta criterios de selección y se los ponderó.

**Tabla 10**

*Evaluación de las tarjetas de control*

| **Evaluación** | | **Arduino DUE** | | **Arduino MEGA** | |
|---|---|---|---|---|---|
| **Criterios de selección** | Peso (%) | Calificación (0-5) | Evaluación ponderada | Calificación (0-5) | Evaluación ponderada |
| **Procesamiento** | 60 | 4 | 2.4 | 3 | 1.8 |
| **Configuración** | 25 | 4 | 1 | 4 | 1 |
| **Costo** | 15 | 4 | 0.6 | 5 | 0.75 |
| **Total** | 100 | --- | 4 | --- | 3.55 |
| **Ponderación** | | 1 | | 2 | |
| **Aprobación** | | Si | | No | |

Con base a la ponderación obtenida se decidió por la tarjeta Arduino DUE, ya que posee una excelente calificación en cada uno de los criterios, es apta para la aplicación y su costo de adquisición no es elevado.

**Bluetooth.** Para cumplir uno de los requerimientos del operador, la conexión inalámbrica se realizó mediante módulos bluetooth que permiten la recepción de los datos enviados por parte del sensor. Los módulos Bluetooth que se consideraron fueron:

**Tabla 11**

*Características de los módulos Bluetooth*

| Características | Bluefruit LE UART Friend | HM-10 | ESP32 |
|---|---|---|---|
| **Voltaje de operación** | 5VDC | 5VDC | 5VDC |
| **Tipos de comandos** | AT | AT | AT |
| **Corriente de operación** | 7 mA | 8.5mA | 120mA |
| **Frecuencia del reloj** | 16Mhz | - | 200Mhz |
| **Velocidad de transmisión** | 9600 baudios (defecto) | 9600 baudios (defecto) | 115200 baudios (defecto) |
| **Dimensiones** | 21x32x5 mm | 13x28x2.2 mm | 55x25x5 mm |
| **Precio** | $18 | $12 | $25 |

*Nota.* (arduino.cl, 2019)

Para decidir el Bluetooth a utilizar, se consideraron criterios de selección y se los ponderaron como se puede observar.

**Tabla 12**

*Evaluación de los módulos Bluetooth*

| Evaluación | | Bluefruit LE UART Friend | | HM-10 | | ESP32 | |
|---|---|---|---|---|---|---|---|
| **Criterios de selección** | Peso (%) | Calificación (0-5) | Evaluación ponderada | Calificación (0-5) | Evaluación ponderada | Calificación (0-5) | Evaluación ponderada |
| **Tamaño** | 35 | 4 | 1.4 | 5 | 1.75 | 3 | 1.05 |
| **Transmisión de datos** | 30 | 4 | 1.2 | 4 | 1.2 | 5 | 1.5 |
| **Costo** | 15 | 4 | 0.6 | 4 | 0.6 | 3 | 0.45 |

| Evaluación | | Bluefruit LE UART Friend | | HM-10 | | ESP32 | |
|---|---|---|---|---|---|---|---|
| **Criterios de selección** | Peso (%) | Calificación (0-5) | Evaluación ponderada | Calificación (0-5) | Evaluación ponderada | Calificación (0-5) | Evaluación ponderada |
| **Consumo de corriente** | 20 | 4 | 0.8 | 4 | 0.8 | 2 | 0.4 |
| **Total** | 100 | --- | 4 | --- | 4.35 | --- | 3.4 |
| **Ponderación** | | 2 | | 1 | | 3 | |
| **Aprobación** | | Si | | Si | | No | |

Debido a que el módulo bluetooth ESP32 posee sobredimensionamiento en tamaño y un consumo corriente de operación alta que podría ocasionar sobrecarga en la tarjeta de control, su calificación fue baja. Mientras que el módulo Bluefruit y HM-10 cumplen con los requerimientos técnicos, en cuanto al consumo de corriente, a dimensiones físicas y precio de venta. Por lo que se decidió a estos dos últimos para la implementación.

### 3.1.3. *Selección de Componentes del Sistema Computacional*

El sistema computacional está conformado por rutinas programadas en Arduino que recopilan datos de las IMU o sensores, los cuales son enviados a MATLAB como variables del algoritmo de control que utiliza ARTE (A Robotic Toolbox for Education) y robotics toolbox para realizar escalamiento, procesamiento y simulación de trayectorias de puntos para luego ser enviados al controlador del brazo robótico mediante comandos en formato ASCII.

Los programas utilizados son Arduino, Matlab y Rhinoceros:

Arduino para la programación de sensores y tarjeta de control, debido a que manejan librerías desarrolladas en C++.

MATLAB genera el resultado de la cinemática directa e inversa del brazo robótico para su posterior simulación, al ser un software con aplicaciones de interfaces permite enlazar todos los sistemas descritos anteriormente mediante un HMI que interactúe con el operador. Rhinoceros permite encontrar puntos y trayectorias de una superficie de trabajo para su posterior procesamiento en Matlab.

### 3.1.4. *Selección de Componentes del Sistema Neumático*

**Compresor.** El compresor es el encargado de suministrar una presión constante de aire para todo el sistema neumático.

**Tabla 13**

*Características de los compresores*

| Características | Mini compresor PORTEN | Compresor BEST BD-101A |
|---|---|---|
| **Voltaje de operación** | 12VDC | 110VAC |
| **Presión máxima** | 2 bares | 8 bares |
| **Caudal** | 15 L /min | 130L/min |
| **Potencia** | 12W | 550W |
| **Dimensiones** | 25.2x15.2x13.2 cm | 45x45x62 cm |
| **Precio** | $30 | $250 |

*Nota.* (Porten, 2020)

En primera instancia se eligió el mini compresor porten por sus dimensión y precio, pero debido a la potencia del motor la presión decae al suministrar aire a diferentes conductos. Por otra parte, el compresor BEST BD-101A suministra presión suficiente para el sistema neumático y la herramienta de trabajo sin tener pérdida de aire por cierto tiempo.

**Válvula solenoide.** La electroválvula es el dispositivo capaz de controlar el paso del aire mediante pulsos eléctricos comandados por el brazo robótico. Debido a que es un sistema implementado en la estación de trabajo de FESTO del laboratorio de Mecatrónica, se optó por utilizar la válvula solenoide existente.

**Tabla 14**

*Características de la válvula solenoide*

| Características | Válvula Solenoide CPE-M1BH-5/3G-QS4-B |
|---|---|
| **Voltaje de operación** | 24VDC |
| **Presión máxima** | 8 bares |
| **Presión mínima** | 3 bares |
| **Válvulas y posiciones** | 5 vías y 3 posiciones con el centro cerrado |
| **Dimensiones** | 34x34x34 cm |
| **Caudal nominal** | 180 L/min |

*Nota.* (Electrical, 2017)

**Herramienta de trabajo.** Este elemento es el efector final del brazo robótico que pulveriza pintura o barniz en una superficie utilizando la presión del aire, para realizar tareas de detalle o pintar pequeñas piezas de forma tridimensional con superficies complicadas.

**Tabla 15**

*Características de las pistolas para pintar*

| Características | Pistola de Gravedad (PORTEN PAE-1215) | Pistola de Succión (PORTEN PPE-4515) | Pistola por presión (KRIPXE 950-PL) |
|---|---|---|---|
| **Tipo de operación** | Manual | Manual | Manual |
| **Presión de trabajo** | 2 bares | 0.26 bares | 3.5 bares |
| **Consumo de aire** | 15 L/min | 150L/min | 240L/min |
| **Boquilla de paso** | 0.5mm | 1.5mm | 2.5mm |
| **Aplicación** | Procesos de acabado finos, maquetas, piezas pequeñas. | Proyectos de repinte tipo hobby, superficies metálicas, madera, etc. | Para productos lisos y salpicados en hebras. |
| **Precio** | $45 | $60 | $350 |

*Nota.* (Aguilar, 2019)

Acorde a las características presentadas se pudo evidenciar que la pistola de gravedad (PORTEN PAE 12-15) es superior con respecto a sus competidoras, en cuanto a precio y aplicación, a pesar de que la boquilla de paso es de 0.5mm. Por otra parte, la pistola de succión (PORTEN PPE-4515) también sería una buena elección, pero su modo de uso limitaría la tarea del presente trabajo debido a que no se puede usar en configuraciones donde la pistola apunte hacia el suelo. En cuanto a la pistola por presión (KRIPXE 950-PL) posee las prestaciones adecuadas pero su alto precio la vuelve inaccesible.

Siendo así la pistola de gravedad (PORTEN PAE 12-15) la más económica, accesible y con las características necesarias para realizar la aplicación que se requiere en este trabajo.

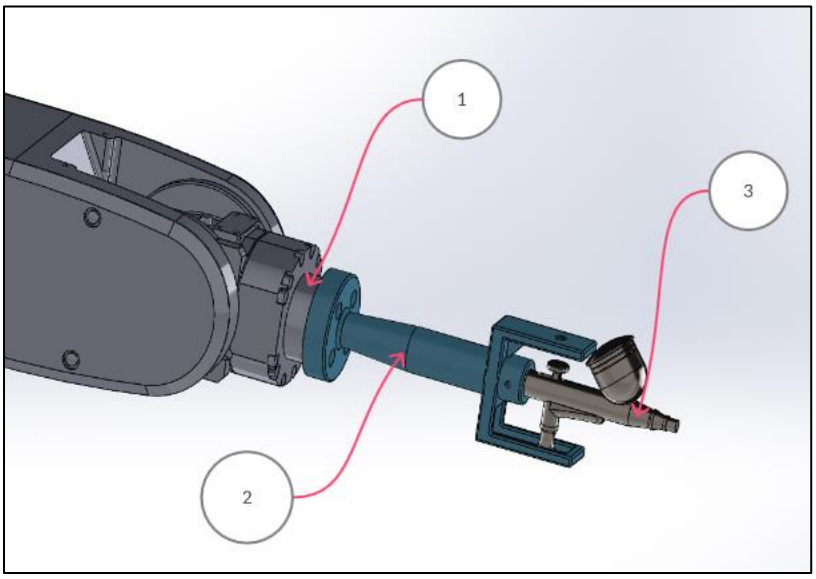
## 3.2. Diseño e Implementación del Sistema Mecánico

Una vez identificado el sistema mecánico se realizó el diseño CAD partiendo de las dimensiones de la pistola de gravedad seleccionada en la sección 3.1.4 y el material de fabricación en la sección 3.1.1.

Los componentes involucrados del sistema mecánico se muestran en la Figura 19.

**Figura 19**

*Componentes del sistema mecánico*




*Nota.* 1. Sexto eje del brazo robótico que permite una rotación de la herramienta, 2. Portaherramientas encargado de la sujeción de la pistola de gravedad, 3. Pistola de gravedad PORTEN PAE-1215.

### 3.2.1. *Análisis CAE del Portaherramientas*

Un elemento mecánico está bien diseñado si no falla por tracción, flexión, torsión o deformación excesiva (Mott, 2009). En el caso de la estructura del portaherramientas la falla por compresión no se produce, debido a que no tiene una fuerza en la punta de esta que la comprima. La falla por tracción puede darse con el movimiento del brazo robótico, junto con la presión del aire que sale por la pistola de gravedad, tratando de estirar al portaherramientas. La falla por flexión se puede dar ya que el portaherramientas actúa como una viga en voladizo con una carga distribuida, siendo esta el peso de la pistola de gravedad. La falla por torción no pueda darse, al no presentar ninguna fuerza tratando de retorcer al portaherramientas. Por último, el portaherramientas al estar sometido a cargas presentó deformación, siendo esta mínima.

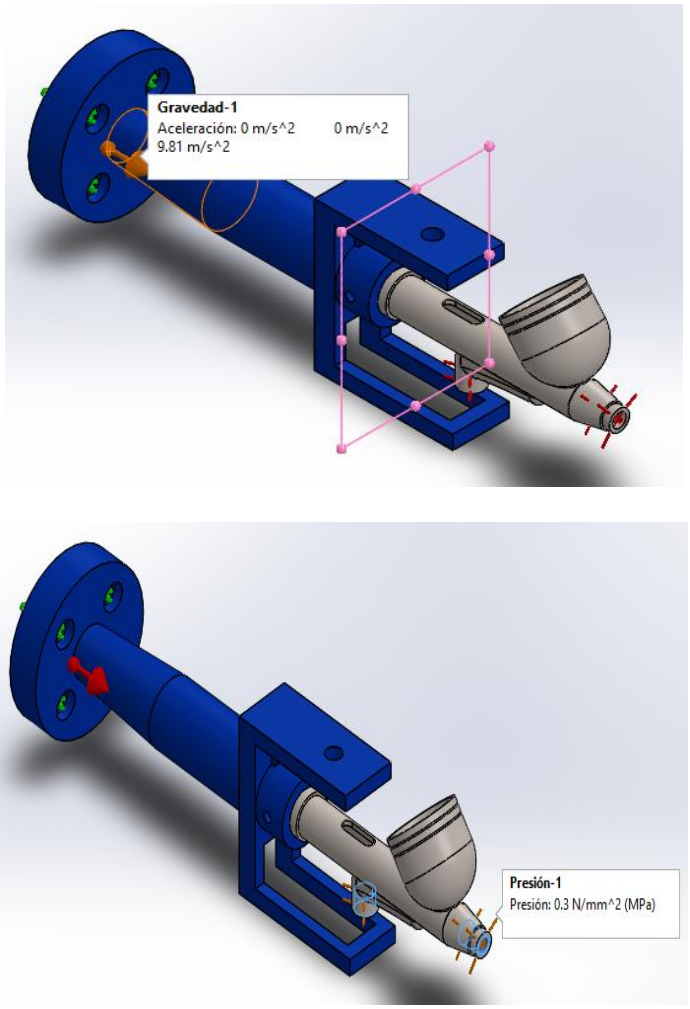
Por este motivo se realizó un análisis CAE en SolidWorks para determinar los valores de esfuerzos, desplazamientos y deformación máxima que posee el elemento cuando es sometido a fuerzas externas.

Las fuerzas externas son aquellas que no forman parte de la estructura, como es el caso del peso de la pistola de gravedad fabricada en acero inoxidable y la presión que ejerce el circuito neumático en la entrada y salida de la misma. El peso de la pistola de gravedad es alrededor de 1.2N considerando una gravedad 9.81 $\frac{m}{s^2}$ y la presión que suministra el compresor es de 0.3 MPa.

En la Figura 20 se muestra las condiciones de frontera que representan las fuerzas, gravedad y presiones en la estructura.

**Figura 20**

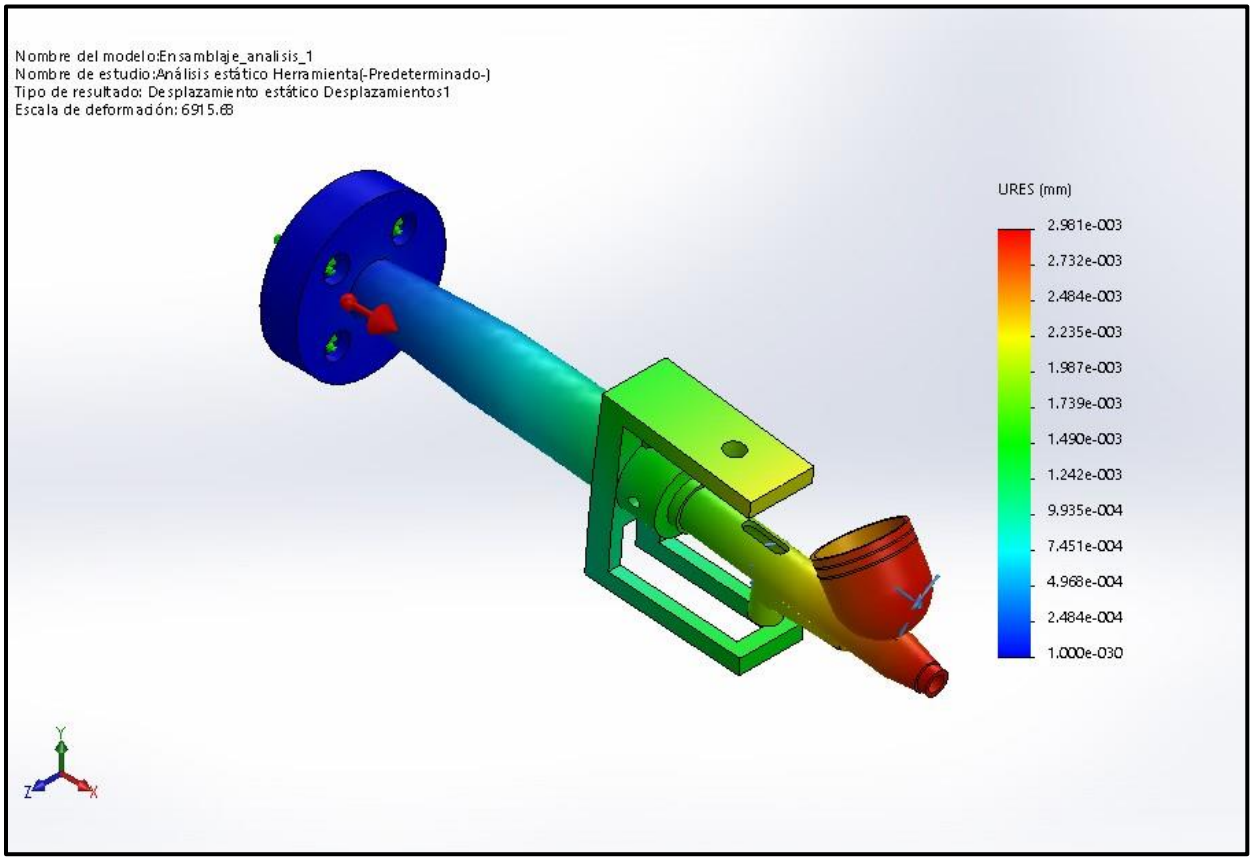
*Condiciones de frontera en el Portaherramientas*




La Figura 21 se muestra en una vista isométrica superior, donde se puede observar que la mayor deformación se presenta en la punta de la pistola de gravedad con un valor de 0.02981mm, y dicho valor se puede considerar insignificante por lo que no afecta al portaherramientas.

**Figura 21**

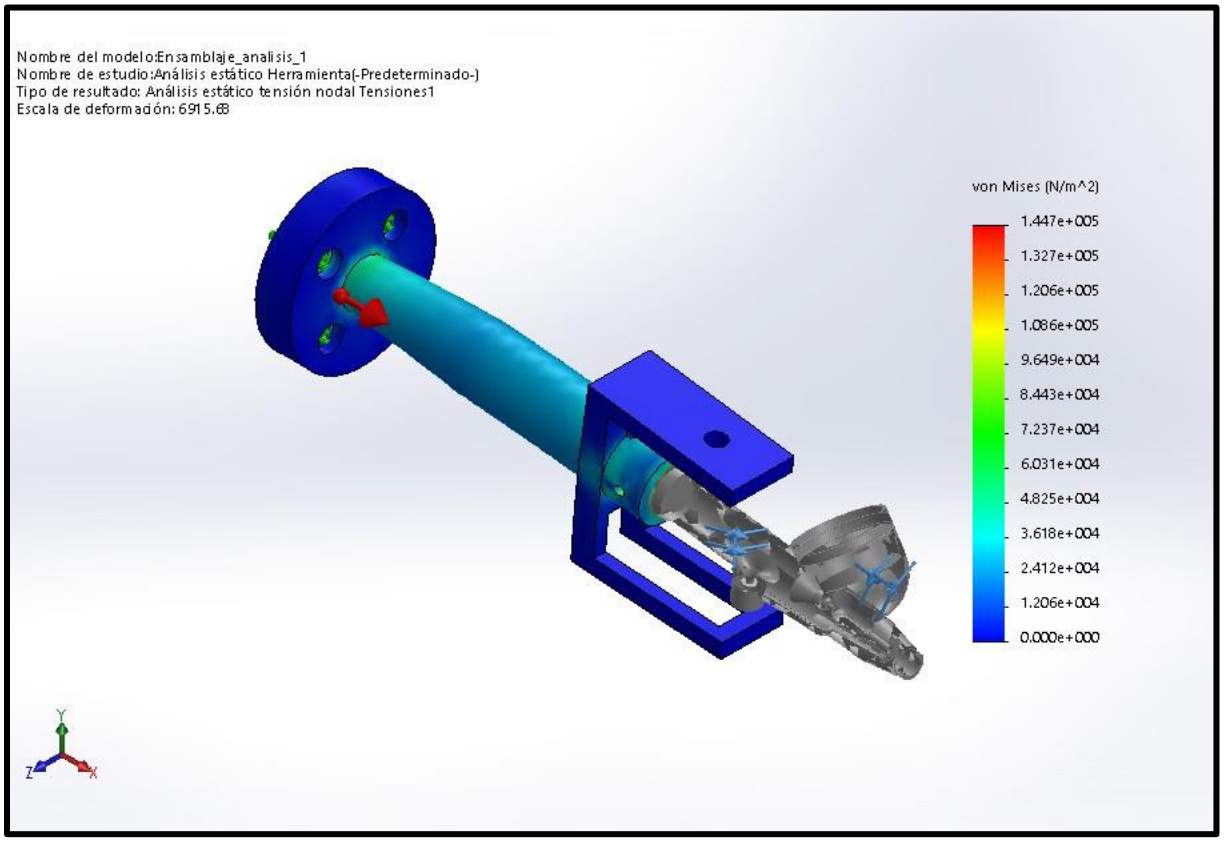
*Deformación en el Portaherramientas*




En la Figura 22 se muestra el esfuerzo que produce la carga y presión suministrada por el compresor el mismo que tiene un valor máximo de 0.1447 MPa, debido a que el PLA tiene un límite elástico de 60MPa y el acero inoxidable 170MPa, por todo esto el portaherramientas no fallará por tracción, flexión o deformación.

**Figura 22**

*Esfuerzos en el Portaherramientas*

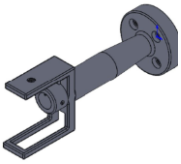



### 3.2.2. *Implementación del Sistema Mecánico*

Para implementar el sistema mecánico, se utilizó la tecnología de impresión 3D para elaborar el portaherramientas utilizando el material seleccionado en la sección 3.1.1 bajo los siguientes parámetros ilustrados en la Tabla 16 y colocados en el software libre Ultimaker Cura.

**Tabla 16**

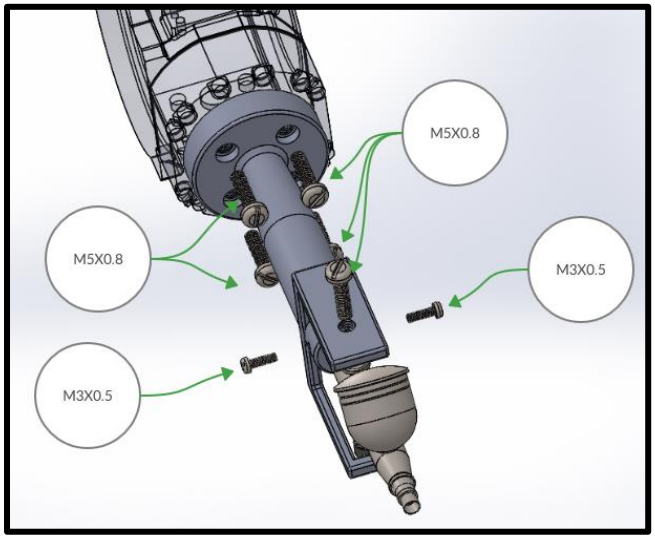
*Parámetros para la impresión 3D*

| Material | Elemento | Altura de capa [mm] | Temperatura del extrusor [°C] | Velocidad de impresión [mm/s] |
|---|---|---|---|---|
| **PLA** |  | 0.2 | 200 | 40 |

Se colocaron cuatro tornillos M5x0.8 para unir el portaherramientas al sexto eje, un tornillo M5x0.8 para presionar el accionamiento de la pistola de gravedad y por último 2 tornillos M3x0.5 para fijar la herramienta de trabajo a este.

**Figura 23**

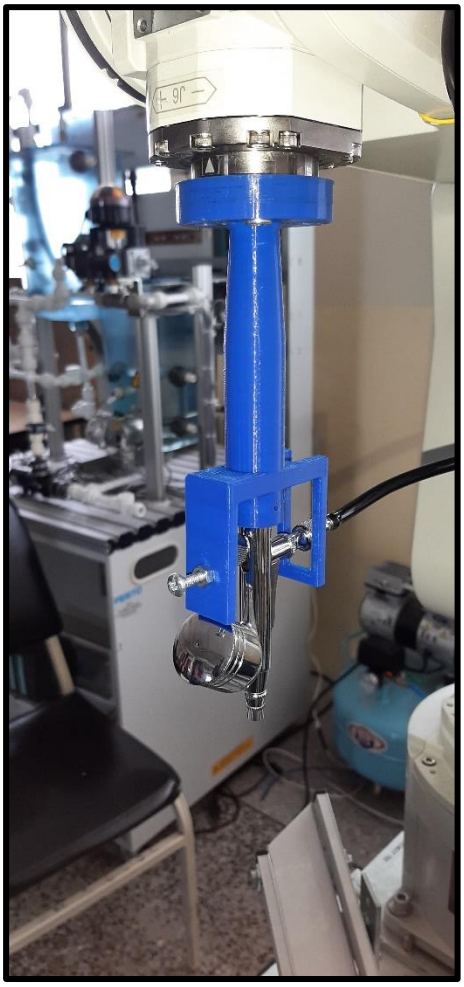
*Configuración del sistema mecánico*



El resultado final de la implementación se lo pueda apreciar en la Figura 24.

**Figura 24**

*Implementación del sistema mecánico*



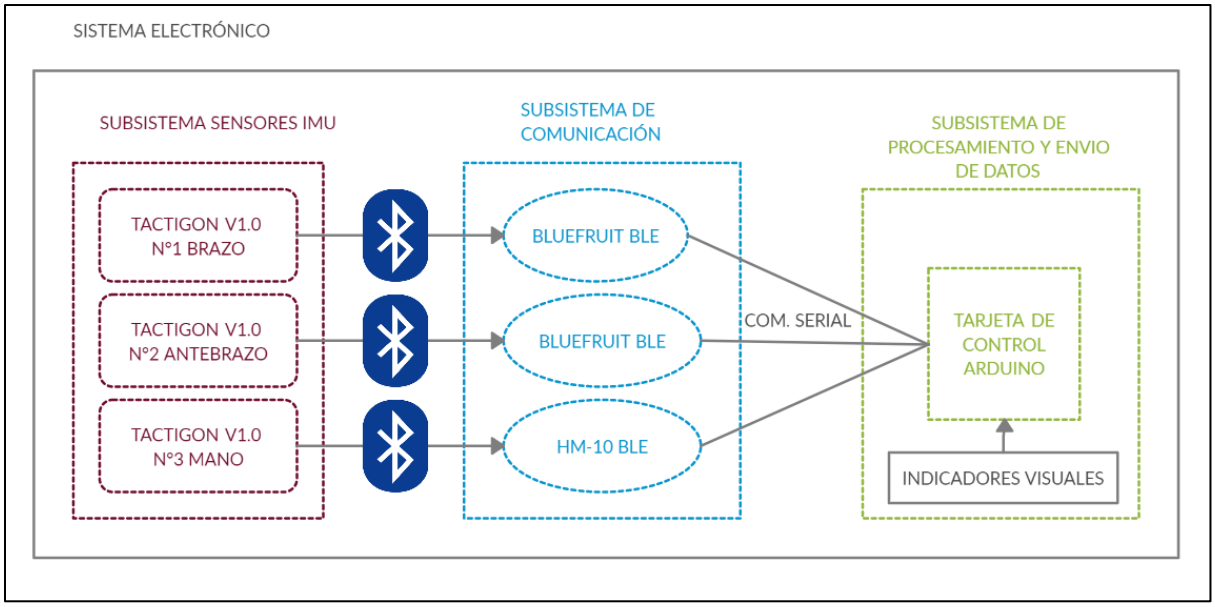
### 3.3. Diseño e Implementación del Sistema Electrónico

Para interpretar el diseño electrónico se elaboró un cuadro general de los distintos componentes seleccionados en la sección 3.1.2. La relación que poseen, se muestra en la Figura 25.

El sistema electrónico se divide en tres subsistemas: sensores IMU, comunicación, procesamiento y envío de datos lo que facilita interpretar cada parte del proceso.

**Figura 25**

*Esquema del sistema electrónico*



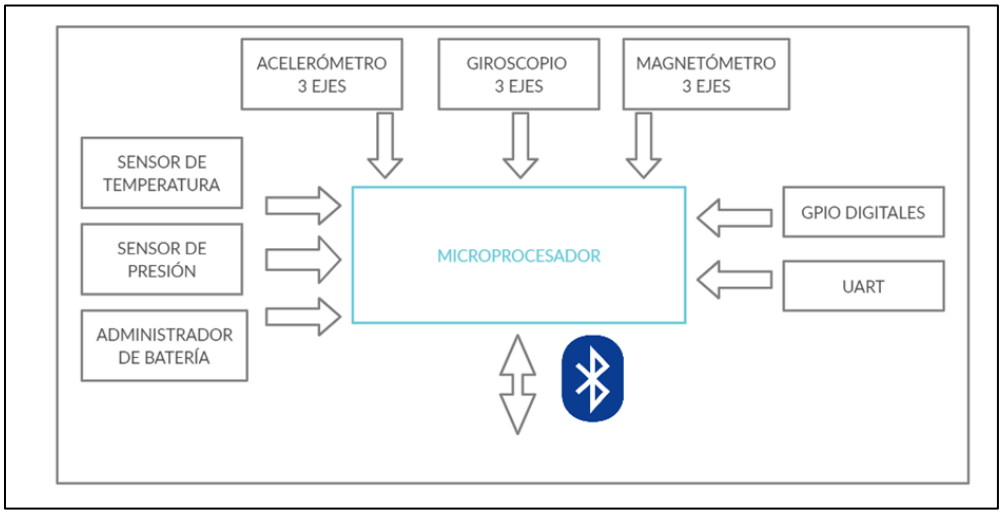

### 3.3.1. *Subsistema Sensores IMU*

El primer subsistema está compuesto por tres Tactigon V1.0 que poseen IMU de 9 Grados de libertad (Acelerómetro, Giroscopio y Magnetómetro) permitiendo el cálculo de desplazamientos angulares, este subsistema se encuentra conectado al subsistema de comunicación mediante BLE (Bluetooth de Bajo Consumo) comunicándose de manera inalámbrica. Cada Tactigon V1.0 posee una batería de litio recargable para un trabajo continuo de hasta 8 horas.

En la Figura 26 se muestra la arquitectura del hardware del sensor Tactigon V1.0.

**Figura 26**

*Arquitectura del sensor Tactigon ONE V1.0*




Por otra parte, en la Figura 27 se representa el algoritmo que utiliza TactigonV1.0 para procesar los datos otorgados por las IMU (utilizando el filtro de Kalman) y así obtener las coordenadas angulares deseadas. Gracias a la arquitectura y al algoritmo de procesamiento de datos propios de Tactigon V1.0 es posible programarlos en Arduino de una manera sencilla.

**Figura 27**

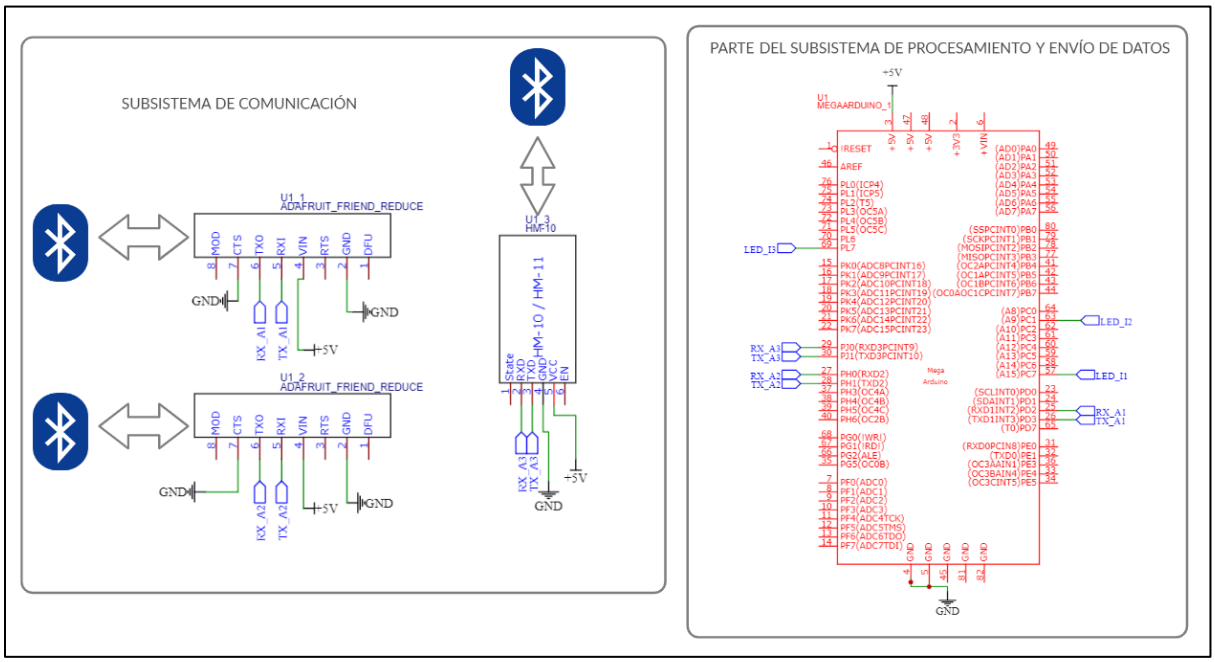
*Algoritmo de procesamiento Tactigon ONE V1.0*

### 3.3.2. *Subsistema de Comunicación*

Este subsistema está conformado por tres módulos BLE que su única tarea es captar los datos enviados del subsistema de sensores y enviar al último subsistema mediante comunicación serial como se indica en la Figura 28.

**Figura 28**

*Subsistema de comunicación*



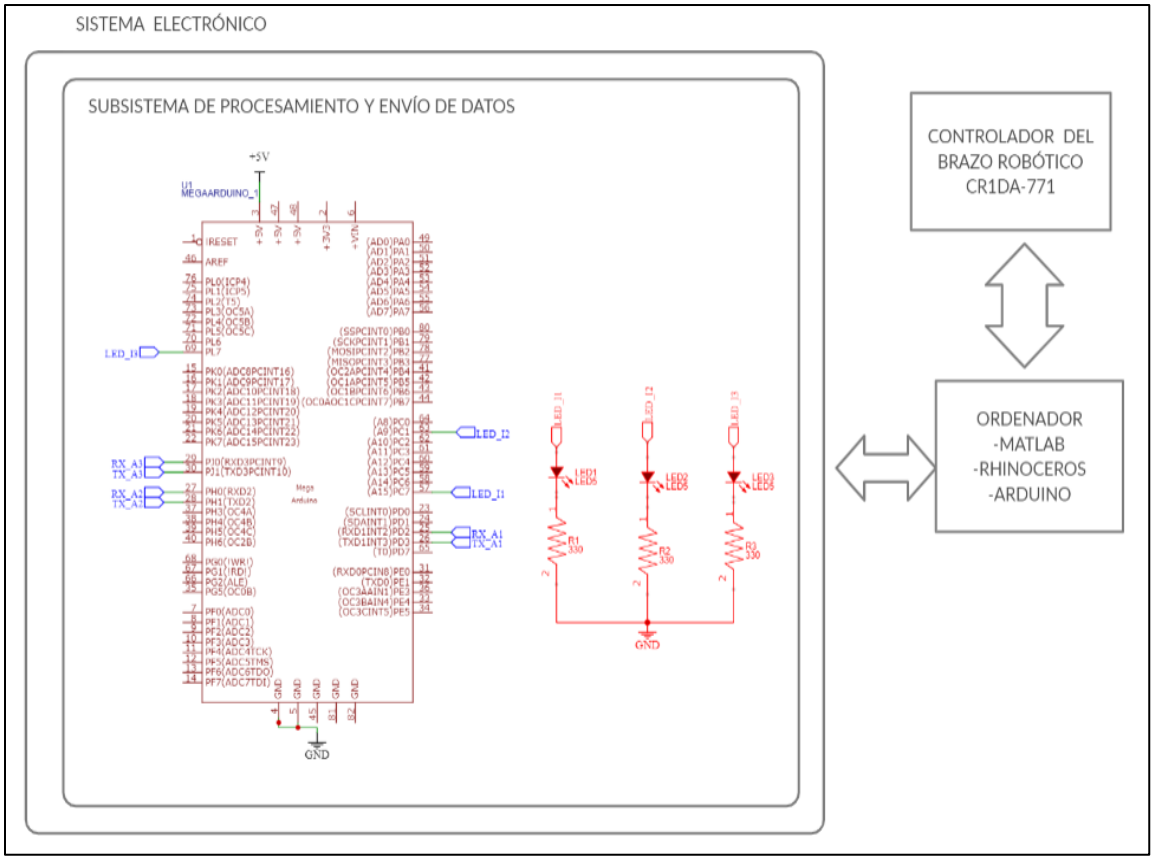

### 3.3.3. *Subsistema de procesamiento y envío de datos*

Este subsistema se encuentra conectado directamente con el ordenador mediante comunicación serial (transmitido por cable USB), enviando continuamente datos recolectados del subsistema de comunicación.

Además, posee indicadores visuales que muestran el estado de conexión de los tres Tactigon V1.0 como se indica en la Figura 29.

**Figura 29**

*Subsistema de procesamiento y envío de datos*




### 3.3.4. *Diseño de la PCB*

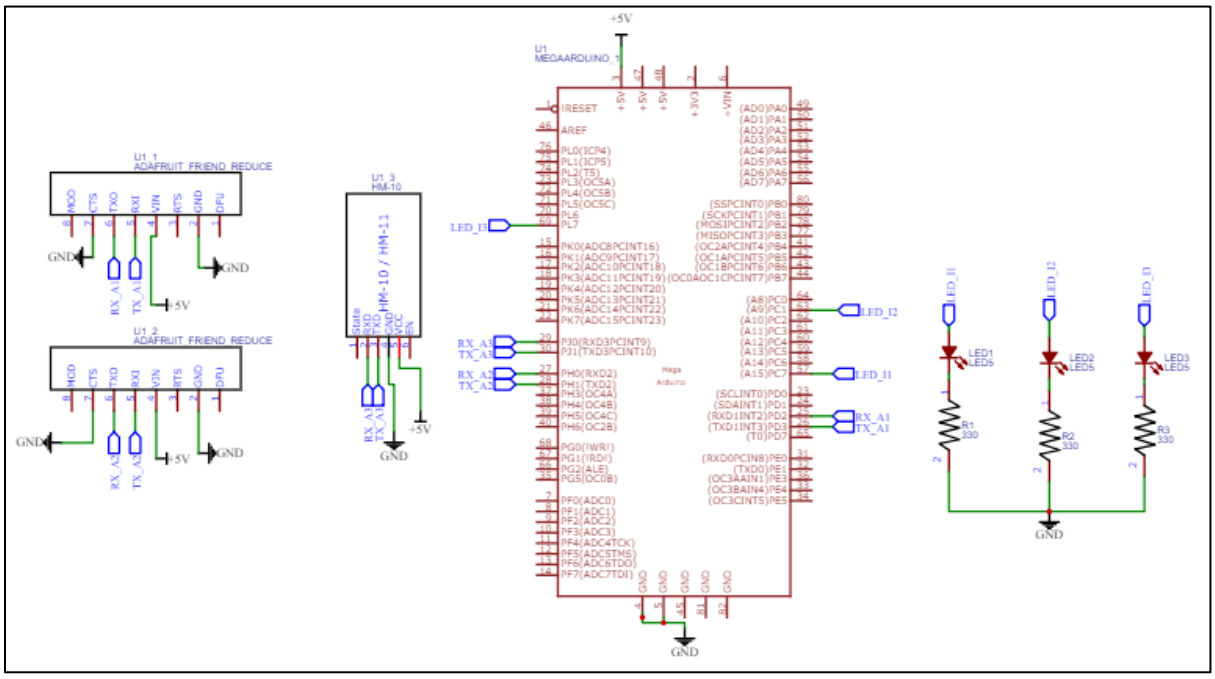
Una vez diseñada el sistema electrónico, se requiere realizar una placa de circuito impreso (PCB) en donde se puedan montar todos los componentes electrónicos para garantizar que su integridad sea compacta y pequeña de acuerdo con la sección 3.1.2, para ello se utilizó el software

EasyEDA que permite la creación de PCB de una manera intuitiva, sencilla y gratuita. (Mitzner, 2009)

En la Figura 30 se muestran las conexiones de los dispositivos electrónicos, para su posterior trazado de pistas, generadas en EasyEDA.

**Figura 30**

*Diseño de PCB*




Se realizó una placa de circuitos modular (Shield) que se coloca encima de una tarjeta de control, para dar una funcionalidad extra y así cumplir con los requerimientos del operador en cuanto a tamaño. (Mitzner, 2009)

Una PCB requiere el cálculo del ancho de pista que tiene como variables el espesor del cobre y la sección transversal que se encuentra en función del amperaje máximo de consumo y la variación de temperatura con respecto al ambiente.

**Cálculo de la sección transversal.** Primero, se calcula el consumo de corriente total de los dispositivos electrónicos como se observa en la Tabla 17.

**Tabla 17**

*Parámetros para el cálculo de la sección transversal*

| Dispositivos electrónicos | Cantidad | Corriente de consumo |
|---|---|---|
| **Led** | 3 | 20mA |
| **Módulo Bluefruit** | 2 | 7mA |
| **HM-10** | 1 | 8.5mA |
| **Total** | | 82.5mA≈100mA |

Segundo, se calcula la variación de temperatura con respecto al ambiente se tiene que, la temperatura máxima de trabajo del circuito es de 25 °C al estar en condiciones ambientales (20°C-25°C).

$\Delta T$ $(Variación\ de\ Temperatura)$

$T_{máx}$ $(Temperatura\ máxima\ de\ trabajo) = 25\ °C$

$T_o$ $(Temperatura\ ambiente) = 20\ °C$

$$\Delta T = T_{máx} - T_o$$

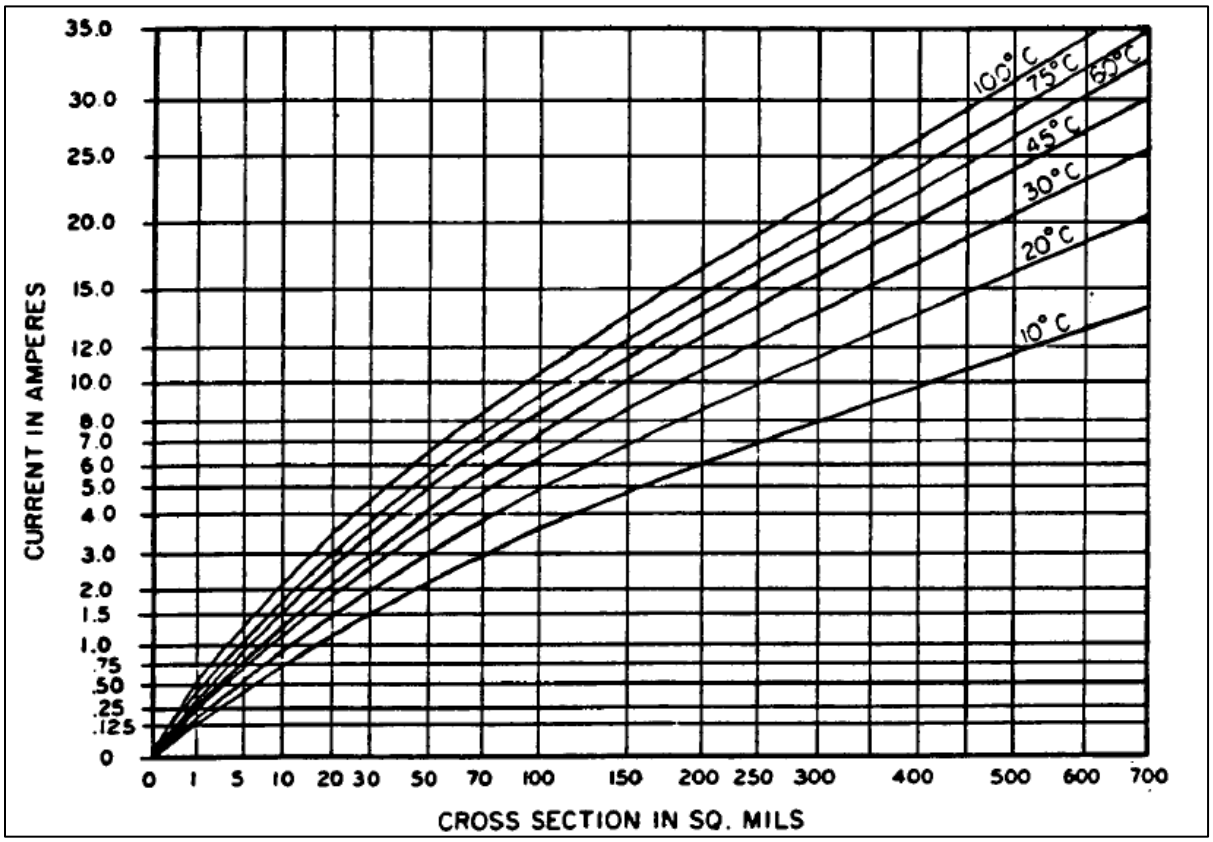
$$\Delta T = 25°C - 20°C$$

$$\Delta T = 5°C$$

Tercero, en base a la Figura 31 de la norma IPC-2221 que representa la Corriente de consumo total versus la variación de temperatura se obtiene que:

**Figura 31**

*Corriente de consumo total versus la variación de temperatura*

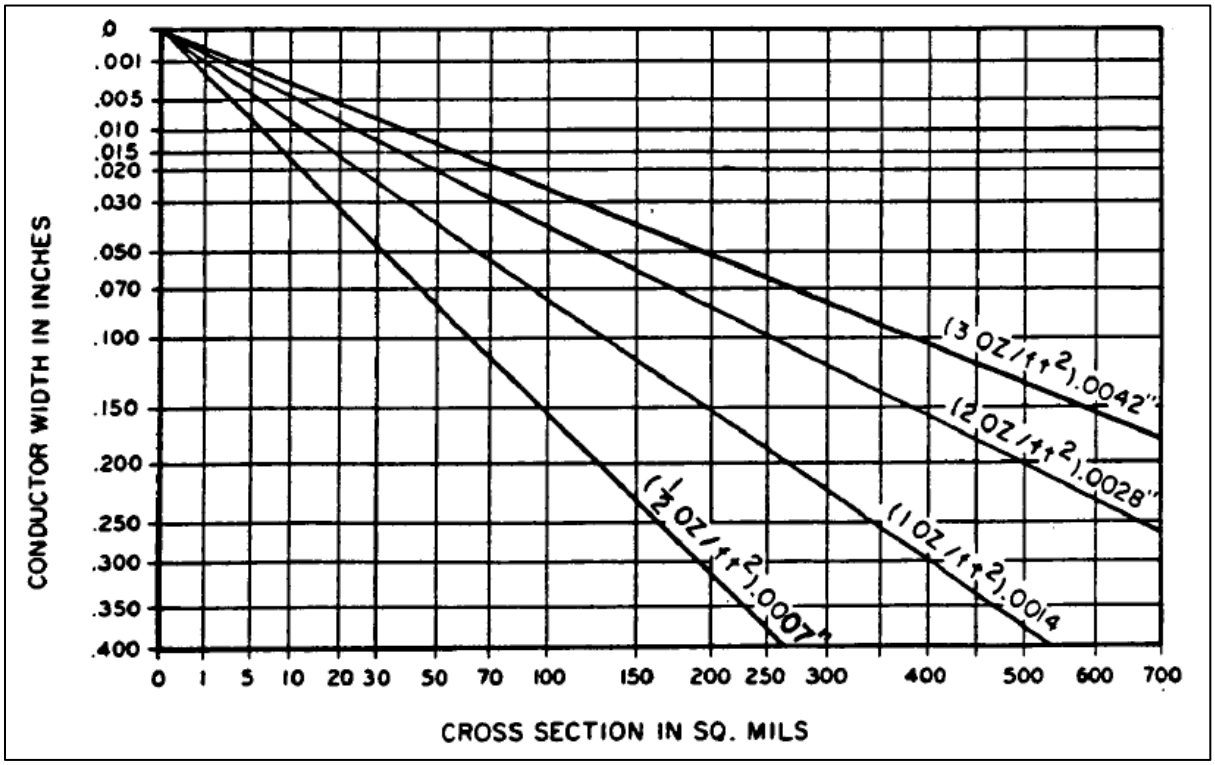



*Nota.* La figura muestra la curva corriente vs. sección transversal. (Institute, 1999).

La sección transversal es de $1.0 - 5.0\ mils^2 (0.000645\ mm^2 - 0.0029\ mm^2)$ con lo que se procede a la Figura 32 para calcular el ancho de pista.

**Figura 32**

*Ancho del conductor vs. Ancho de pista*




*Nota.* La figura muestra el espesor del conductor vs. sección transversal. (Institute, 1999).

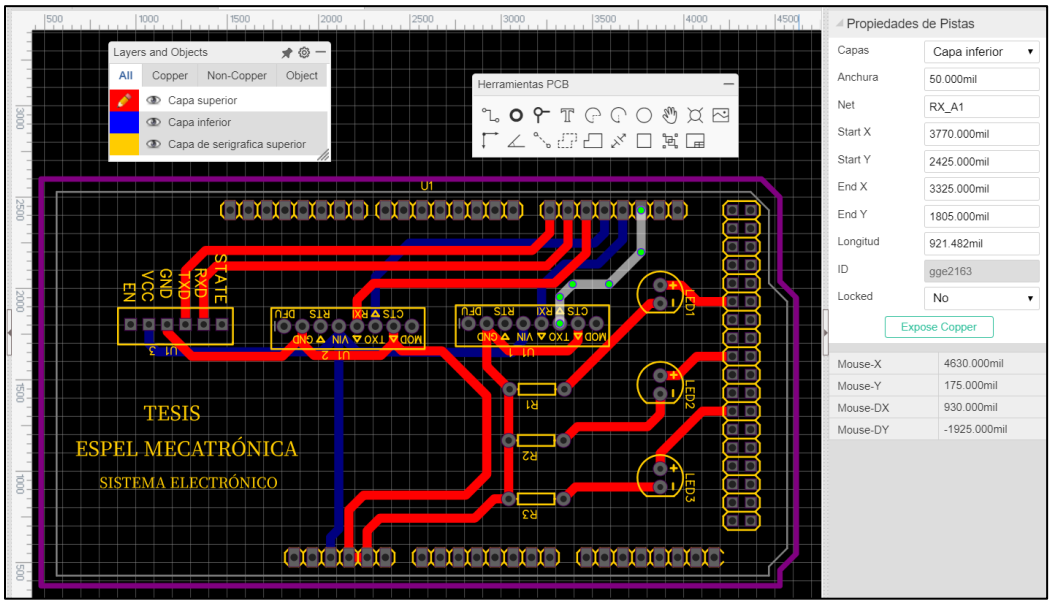
Finalmente, tomando el valor del espesor del cobre de $1\,\frac{oz}{ft^2}$ $(305\,\frac{g}{m^2})$, siendo este el valor comercial del mercado local, se obtiene el ancho de pista de la PCB con un valor mínimo de $0.005\ in$ $(0.127mm\ o\ 5\ mil)$.

Basados en que el ancho de pista no puede ser menor a 5 mil calculado mediante el procedimiento detallado previamente, para la manufactura de la placa se utilizo el valor de 50 mil debido a que es un estándar en las máquinas de prototipado para PCB.

Introduciendo los valores en el EasyEDA se obtiene la configuracion que se puede observar en la Figura 33.

**Figura 33**

*PCB en EasyEDA*




### 3.3.5. *Implementación del sistema electrónico*

Los dispositivos Tactigon V1.0 se colocan en tres lugares de la extremidad superior como se observa en la Figura 34.

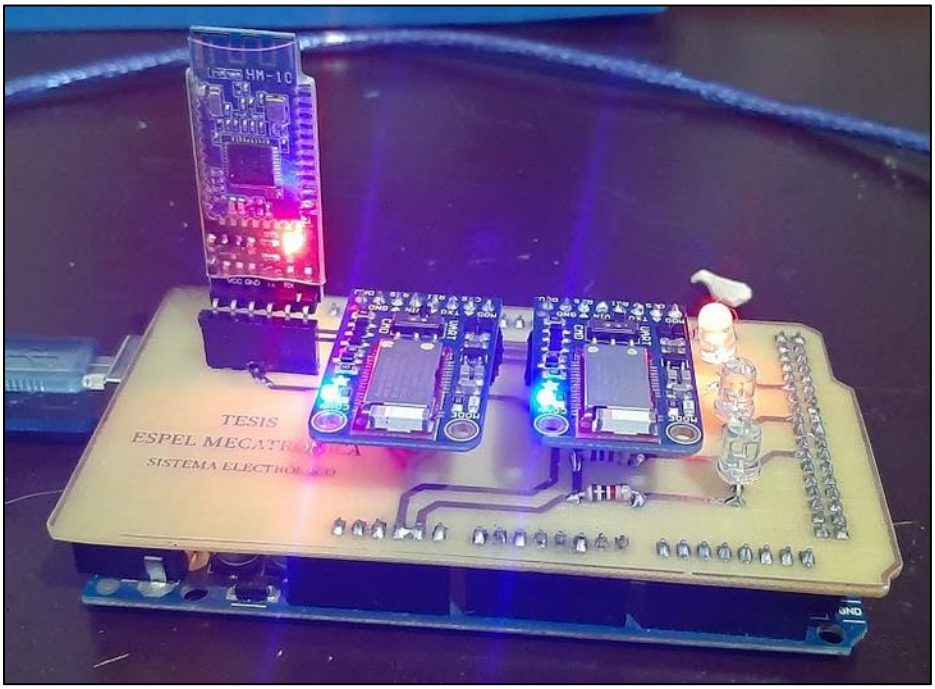
**Figura 34**

*Implementación del sistema electrónico*

Una vez fabricada la placa (shield) se procede a equiparla con los dispositivos electrónicos del subsistema de comunicación y del subsistema de procesamiento y envío de datos, mediante soldadura blanda utilizando estaño y cautín.

**Figura 35**

*PCB montada en el Arduino*

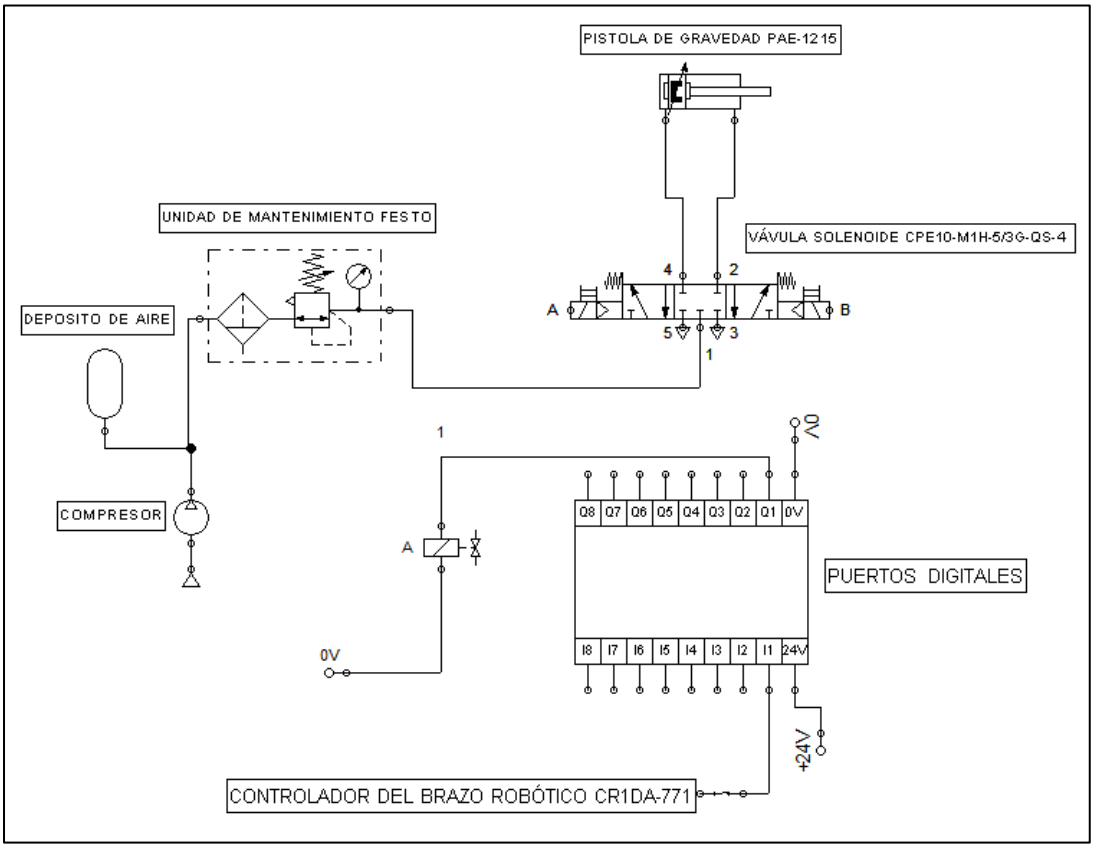



### 3.4. Esquema del Sistema Neumático

La Estación de Robot (Calidad Industrial) de FESTO que posee el laboratorio de Mecatrónica tiene una gama extensa de dispositivos que son utilizados en tareas como el montaje de piezas. Para trasladar piezas emplea un sistema neumático acoplado a lo largo de la estación hasta el efector final del brazo robótico (pinza de sujeción). El presente proyecto ocupó el sistema con la configuración representada en la Figura 36.

**Figura 36**

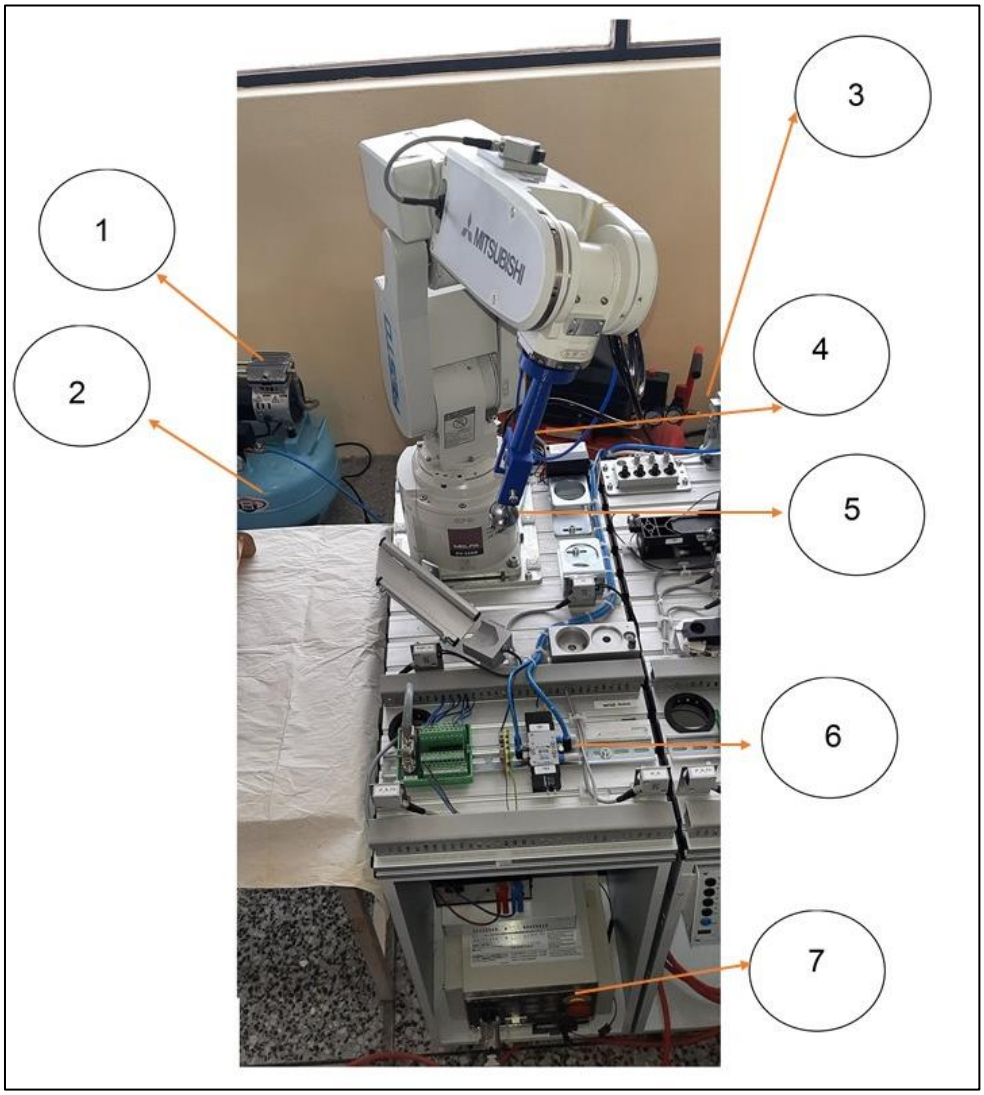
*Configuración del sistema neumático*




La configuración permitió que el controlador del brazo robótico CR1DA-771, al receptar las ordenes emitidas por el Sistema Electrónico y Computacional, accione la válvula solenoide de la Estación de Trabajo proporcionando la alimentación de aire a la pistola de gravedad como se puede visualizar en la Figura 37.

**Figura 37**

*Sistema neumático*




*Nota.* Los componentes involucrados del sistema neumático: 1. Compresor BEST BD-101, 2. Depósito de aire, 3. Unidad de mantenimiento FESTO, 4. Portaherramientas, 5. Pistola de gravedad PAE-1215, 6. Válvula solenoide CPE10-M1BH-5/3G-QS4-B, 7. Controlador CR1DA-771.

# CAPÍTULO IV

# INTEGRACIÓN DE SISTEMAS E IMPLEMENTACIÓN DEL ALGORITMO DE CONTROL

En el presente capítulo se muestran los pasos para el diseño del algoritmo de control, partiendo de la implementación de los sistemas electrónicos, neumáticos, mecánicos y computacionales para finalmente realizar la programación que enlace los sensores, tarjeta de control, ordenador y controlador del brazo robótico. Los mismos que permitieron realizar una aplicación a nivel industrial, como lo es el acabado de superficies por recubrimiento (Pintura).

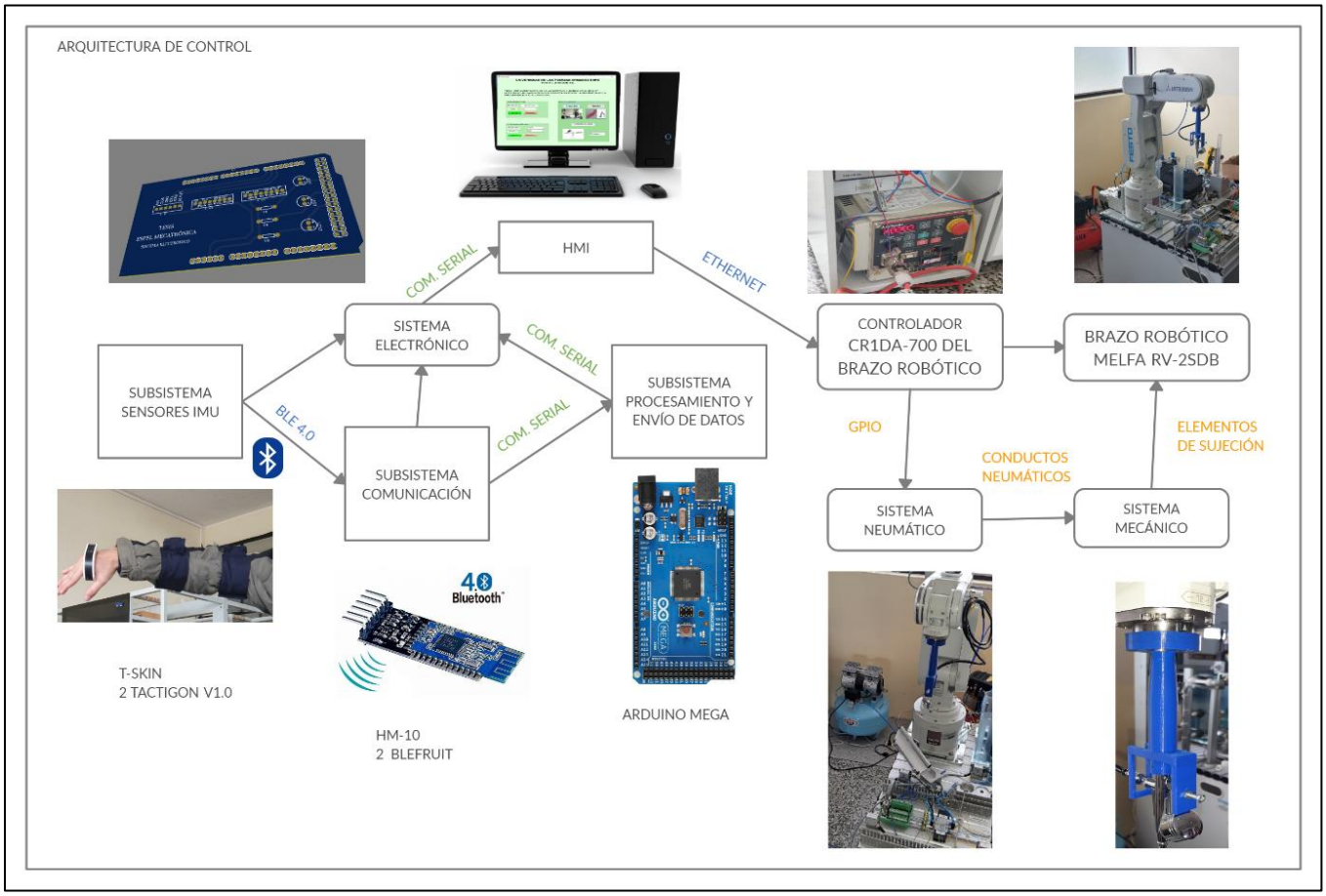
## 4.1. Arquitectura de Control

Para poder ilustrar la conexión entre los diferentes sistemas se requiere establecer una organización jerárquica con su respectiva comunicación , para ello se establece la interfaz hombre máquina (HMI) como el elemento superior que recibe datos del sistema electrónico a través de la comunicación serial procesándolos con la ayuda de MATLAB, para su posterior envío al controlador CR1DA-711 mediante protocolo ethernet (RJ-45) permitiendo controlar la posición del brazo robótico y el accionamiento adecuado del sistema neumático entradas y salidas digitales propias del controlador.

En la Figura 38 se muestra la arquitectura de control que se implementó en el presente proyecto.

**Figura 38**

*Arquitectura de control*




## 4.2. Parámetros Denavit Hartenberg

### 4.2.1. *DH del Brazo Robótico*

Tomando en cuenta la sección 2.3 se introducen los valores de las dimensiones de los ejes permitiendo generar los parámetros DH colocados en la Tabla 18.

**Tabla 18**

*DH del Brazo Robótico*

| $\boldsymbol{\theta}$ | $\boldsymbol{q(1)}$ | $\boldsymbol{q(2) - \frac{\pi}{2}}$ | $\boldsymbol{q(3)}$ | $\boldsymbol{q(4)}$ | $\boldsymbol{q(5)}$ | $\boldsymbol{q(6)}$ |
|---|---|---|---|---|---|---|
| **D** | 0.295 | 0 | 0 | 0.27 | 0 | 0.07 |
| **a** | 0 | 0.23 | 0.05 | 0 | 0 | 0 |
| $\propto$ | -pi/2 | 0 | -pi/2 | pi/2 | -pi/2 | 0 |

### 4.2.2. *DH del Brazo Humano*

Se consideró las dimensiones corporales de la investigación de (Hamlet y Díaz, 2007) encontrando que la longitud del brazo y antebrazo son de 34 cm y 24 cm respectivamente, además se toma en cuenta la biomecánica del hombro y codo los cuales poseen 2 GDL cada uno, permitiendo generar los parámetros DH expuestos en la Tabla 19.

**Tabla 19**

*DH del Brazo humano*

| | | | | |
|---|---|---|---|---|
| **Link (1)** | **0** | **0** | **0** | **-pi/2** |
| **Link (2)** | 0 | 0 | 0.34 | -pi/2 |
| **Link (3)** | 0 | 0 | 0 | Pi/2 |
| **Link (4)** | 0 | 0 | 0.33 | Pi/2 |

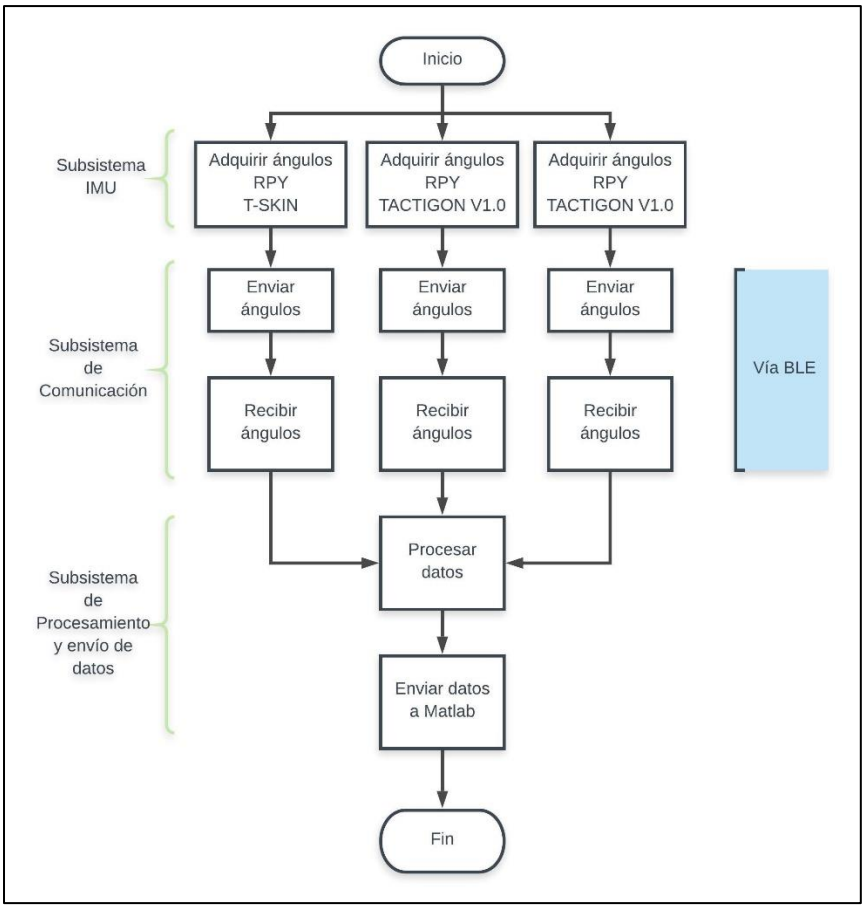
## 4.3. Algoritmo de Control del Sistema Electrónico

En esta sección se muestra el algoritmo implementado en el sistema electrónico y la programación que poseen los subsistemas detallados en la sección 3.3, utilizando lenguajes de programación en Arduino y Matlab.

La Figura 39 muestra el flujograma detallado del algoritmo de programación del sistema electrónico.

**Figura 39**

*Flujograma de programación del sistema electrónico*



### 4.3.1. *Programación del Subsistema IMU*

En la Figura 40 se muestran las librerías utilizadas para el sensor Tactigon v1.0 programado en Arduino.

**Figura 40**

*Librerías en Arduino para el sensor Tactigon V1.0*

```
#include <tactigon_led.h>
#include <tactigon_IMU.h>
#include <tactigon_BLE.h>
#include <tactigon_IO.h>
```

Gracias a las librerías cargadas y desarrolladas por la empresa “The Tactigon”, es posible adquirir los ángulos Roll, Pitch y Yaw de una manera sencilla como se indica en la Figura 41 para su posterior envío hacia el subsistema de comunicación.

**Figura 41**

*Adquisición de los ángulos RPY en Arduino*

```
    qData = qMeter.getQs();
    roll=round(radToDeg(qData.roll));
    yaw=round(radToDeg(qData.yaw));
    pitch=round(radToDeg(qData.pitch));
```

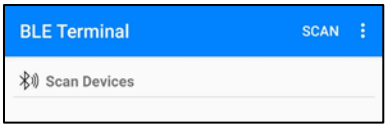
### 4.3.2. *Programación del Subsistema de Comunicación*

Para establecer una conexión satisfactoria entre los sensores y el sistema de comunicación se debe conocer las MAC y UUID de los distintos módulos BLE (HM-10 y Bluefruit).

Para ello se utilizó la app “BLE terminal” para encontrar dichas características como se indica en la Figura 42.

**Figura 42**

*BLE terminal app*




Una vez que se encontró dichas características es necesario utilizar el código mostrado en la Figura 43 para establecer la conexión y envío de ángulos hacia el módulo BLE seleccionado.

**Figura 43**

*Código para la conexión y envió de ángulos hacia el módulo BLE*

```
#define TARGET_MAC {0xEB, 0x1B, 0xD2, 0xD7, 0x6B, 0x77};
#define TARGET_CHAR "6e400002-b5a3-f393-e0a9-e50e24dcca9e
 bleManager.InitRole(TACTIGON_BLE_CENTRAL);
 targetUUID.set(TARGET_CHAR);
 bleManager.setTarget(targetMAC, targetUUID);
 bleManager.writeToPeripheral((unsigned char *)bleBuff, strlen(bleBuff));
```

Características del módulo Adafruit Bluefruit #2

Envío de ángulos

### 4.3.3. *Programación del Subsistema de Procesamiento y Envío de Datos*

Una vez que los datos se encuentran en los módulos BLE son transmitidos a la tarjeta de control mediante comunicación serial (RX, TX), por lo que se abrieron los tres puertos seriales existentes y de esta forma adquirir los datos para su posterior envío como se indica en la Figura 44.

**Figura 44**

*Adquisición de los datos para su posterior envío a la tarjeta de control*

```
Serial1.begin(9600);
Serial2.begin(9600);
Serial3.begin(9600);

while (Serial1.available())

//sscanf permite receptar los datos de los módulos ble
match1 = sscanf(buff1, "u%dr%dy%d", &num1, &a11, &a21);

Serial.print(roll1);
Serial.print(",");
Serial.print(yaw1);
Serial.print(",");
Serial.print(roll2);
Serial.print(",");
Serial.print(yaw2);
Serial.print(",");
Serial.print(roll3);
Serial.print(",");
Serial.print(yaw3);
Serial.print(",");
Serial.print(pitch3);
Serial.print(",");
Serial.println(envio);
```

Inicializar puertos seriales

Bucle de un puerto serial activo

Recepción de ángulos

Envío de datos

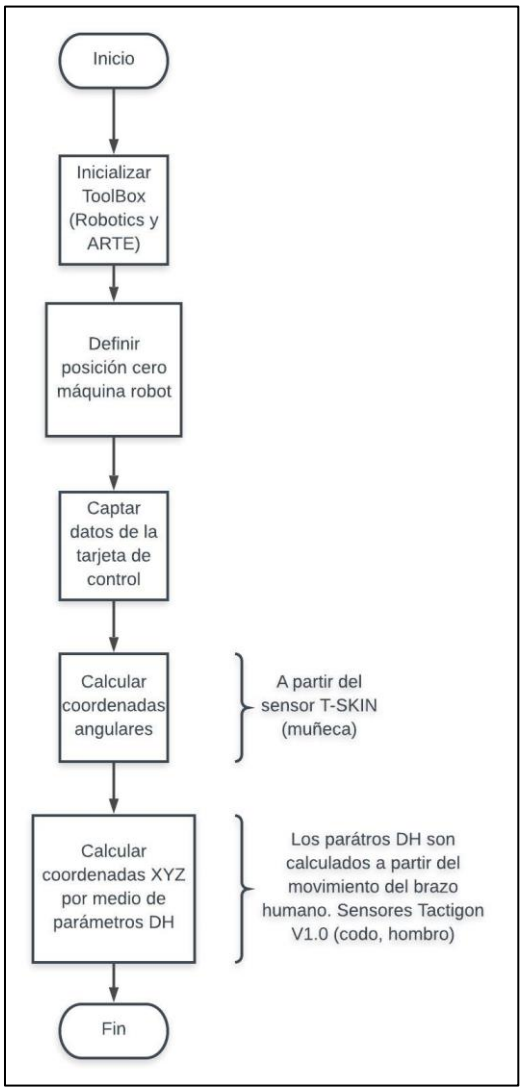
*Nota.* Los valores de roll 1 y yaw 1 representan los ángulos originales del sensor Tactigon V1.0 N°1 mientras que roll2 y yaw2 son propios del Tactigon V1.0 N°2. Por último, roll3, yaw3, pitch3 y la variable “envío” son originales del T-Skin.

## 4.4. Algoritmo para la Transformación de Ángulos RPY a Posiciones XYZ

En esta sección se presenta la explicación sobre la transformación de ángulos RPY a posiciones XYZ, de esta forma el controlador brazo robótico podrá interpretar los movimientos emitidos por el brazo humano.

**Figura 45**

*Flujograma para la Transformación de Ángulos RPY a Posiciones XYZ*



### 4.4.1. *Programación para Inicializar los Toolboxes*

En la Figura 46 se muestran los comandos para inicializar los Toolbox Robotics y ARTE.

**Figura 46**

*Comandos para iniciar los Toolboxes*

```
%Inicializar el ToolBox
startup_rvc;
init_lib;
```

### 4.4.2. *Programación para Definir Posiciones Cero del Brazo Robótico*

Debido a que la transformación de posiciones angulares RPY a posiciones XYZ debe poseer una ubicación conocida, se establece un punto constante o "posición cero" que es implementado mediante el código de la Figura 47 que sirve para definir un origen del sistema referencia para el posterior cálculo de posiciones.

**Figura 47**

*Definición de áreas de trabajo*

```
%ÁREA DE TRABAJO1
pos_cero_maquina.x1 = 300;
pos_cero_maquina.y1 = 0;
pos_cero_maquina.z1 = 310;
pos_cero_maquina.a1 = 180;
pos_cero_maquina.b1 = 0;
pos_cero_maquina.c1 = 180;
%ÁREA DE TRABAJO2
pos_cero_maquina.x2 = 0;
pos_cero_maquina.y2 = -250;
pos_cero_maquina.z2 = 320;
pos_cero_maquina.a2 = 180;
pos_cero_maquina.b2 = 0;
pos_cero_maquina.c2 = 90;
```

### 4.4.3. *Recibir Datos de la Tarjeta de Control*

En la Figura 48 se muestra el código para la recepción de datos, los cuales son enviados desde la tarjeta de control mediante comunicación serial.

**Figura 48**

*Código para la recepción de datos de la tarjeta de control*

```
s = serial(com_serial,'BaudRate',baudios_puerto,'DataBits',8);
%Abrir el puerto serial
timeout = 1;
set(s,'Timeout',timeout);
fopen(s);

 valor=fscanf(s,'%d,%d,%d,%d,%d,%d,%d,%d');
```

Configuración y apertura de la comunicación serial

Recepción de datos

La variable “valor” posee todos los datos del algoritmo por lo que se procede a realizar un cambio de variables para un mejor entendimiento durante el algoritmo como se presenta en la Figura 49.

**Figura 49**

*Código para cambiar la variable de los datos obtenidos*

```
%Cambio de variable de los datos obtenidos
%en la comunicación serial
angulo1_roll = valor(1)+calibrar.cal_roll1;
angulo1_yaw = valor(2)+calibrar.cal_yaw1;
angulo2_roll = valor(3)+calibrar.cal_roll2;
angulo2_yaw = valor(4)+calibrar.cal_yaw2;
angulo3_roll  = valor(5)+calibrar.cal_roll3;
angulo3_yaw = valor(6)+calibrar.cal_yaw3;
angulo3_pitch = valor(7)+calibrar.cal_pitch3;
simu_real = valor(8);
```

### 4.4.4. *Programación para el Cálculo de Coordenadas Rectangulares XYZ*

Una vez obtenido los datos en distintas variables, se procede a utilizar angulo1_roll, angulo1_yaw propias del sensor N°1 (hombro) y angulo2_roll, angulo2_yaw del sensor N°2 (codo) como datos de entrada para las funciones de Robotics Toolbox, permitiendo el cálculo de las coordenadas XYZ mediante los parámetros DH (Denavit Hartenberg) propios del brazo humano como se indica en la Figura 50.

**Figura 50**

*Código para introducir los parámetros DH*

```
%link (revolute , d ,a, alpha)

L(1) = Link ([0 0 0 -pi/2])          Parámetros DH del
L(2) = Link ([0 0 0.34 -pi/2 ])      brazo humano
L(3) = Link ([0 0 0 pi/2 ])
L(4) = Link ([0 0 0.33 pi/2 ])
%0.33 = 0.23(longitud del antebrazo) + 0.1(longitud de la
%muñeca hacia el T-skin

robo = SerialLink (L , 'name' , 'Brazo_Jeff')  Carga de los parámetros DH

                                     Vector qf1 conformado por 2
qf1= [yaw1 roll_1 yaw2 roll_2];      sensores (hombro y codo)
T = robo.fkine(qf1);                 Cálculo de coordenadas XYZ
coor_trabajo_X = T.t(1);
coor_trabajo_Y = T.t(2);
coor_trabajo_Z = T.t(3);
%Conversion amm
diferencia_X = pos_x*1000-coor_trabajo_X*1000;
diferencia_Y = pos_y*1000-coor_trabajo_Y*1000;
diferencia_Z = pos_z*1000-coor_trabajo_Z*1000;
posicionX_real =diferencia_X+pos_cero_maquina.x;
posicionY_real =diferencia_Y+pos_cero_maquina.y;  Coordenadas XYZ finales
posicionZ_real =diferencia_Z+pos_cero_maquina.z;
```

### 4.4.5. *Programación para el Cálculo Coordenadas Angulares ABC*

Luego de obtener las coordenadas XYZ se procede al cálculo de las coordenadas angulares ABC utilizando angulo3_roll, angulo3_yaw, angulo3_pitch propias del sensor N°3 (muñeca) con la entrada del comando *"tr2rpy"* como se indica en la Figura 51.

**Figura 51**

*Código para el cálculo de coordenadas angulares ABC*

```
roll_pitch_yaw =
tr2rpy(roty(deg2rad(angulo3_roll))*rotx(deg2rad(pos_cero_maquina.a+
angulo3_pitch))*rotz(deg2rad(pos_cero_maquina.a+angulo3_pitch)),'deg');
disp(roll_pitch_yaw);
posicionA_real = roll_pitch_yaw(1);
posicionB_real = roll_pitch_yaw(2)+pos_cero_maquina.b;
posicionC_real = roll_pitch_yaw(3);
```

## 4.5. Programación Offline con Rhinoceros

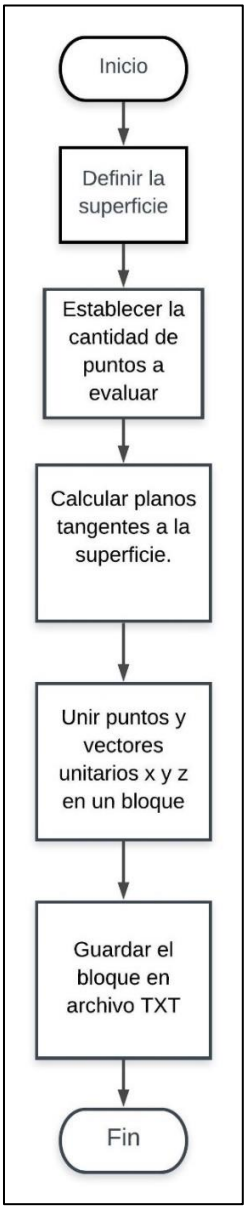
El software Rhinoceros es una herramienta para el modelado en 3D que utiliza el complemento Grasshopper para obtención de geometría 3D (puntos, vectores y planos) mediante lenguaje de programación visual.

### 4.5.1. *Algoritmo para Obtener Puntos y Vectores Unitarios de una Superficie Compleja*

Para la obtención de los puntos y vectores unitarios se utilizó el flujograma que se puede observar en la Figura 52.

**Figura 52**

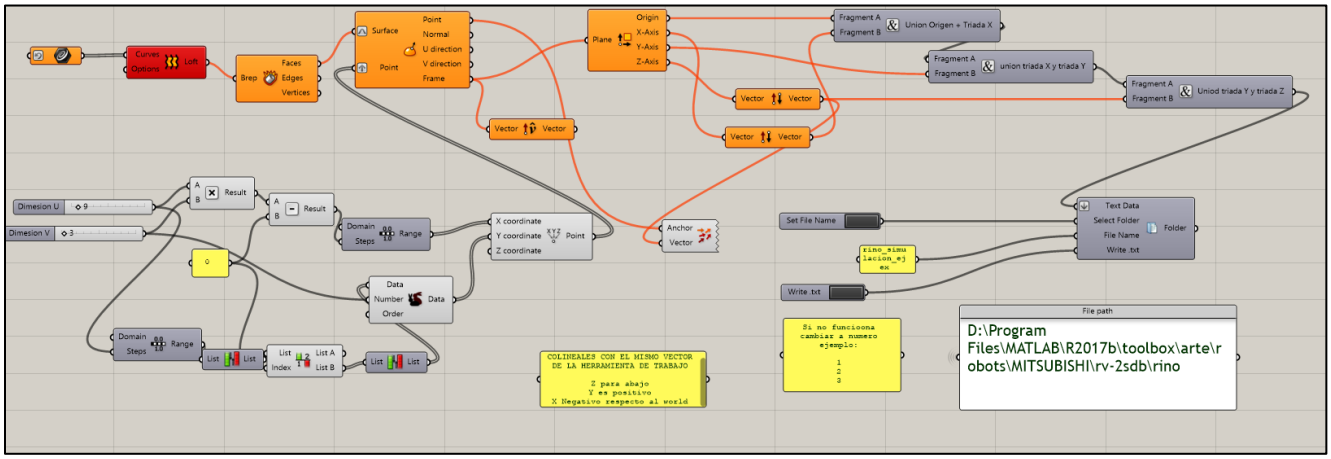
*Flujograma para la obtención de puntos y vectores unitarios*




Para realizar la captura de puntos y vectores unitarios x, y, z en un archivo con extensión TXT se requiere aplicar un algoritmo programado en Grasshopper como se visualiza en la Figura 53.

**Figura 53**

*Programación gráfica en Grasshopper*

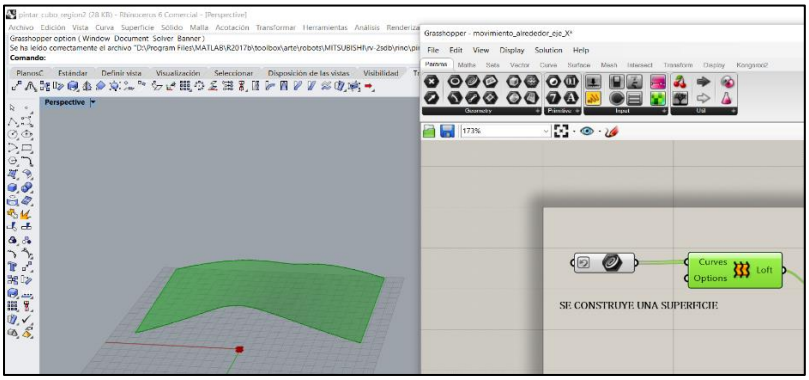



Es posible dividir este algoritmo en 5 etapas, debido a su complejidad serán detalladas a continuación:

**Definir la Superficie.** Se define la superficie utilizando Rhinoceros y dos bloques de programación en Grasshopper como se muestra en la Figura 54.

**Figura 54**

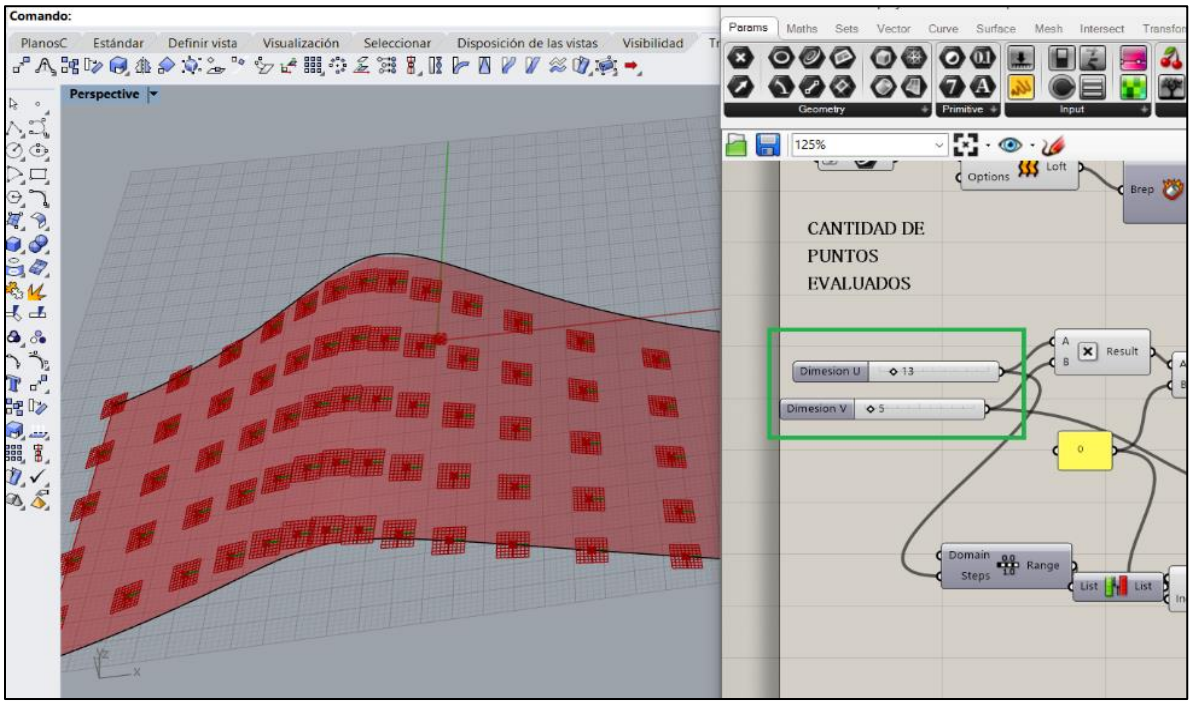
*Definir la superficie en Rhinoceros*



**Establecer la Cantidad de Puntos a Evaluar.** Una vez obtenida la superficie de trabajo, se requiere dividirla en un cierto número de filas (slider Dimensión V) y de columnas (Slider Dimensión U) lo que permite aumentar o disminuir los puntos a evaluar en la superficie como muestra los bloques de programación de la Figura 55.

**Figura 55**

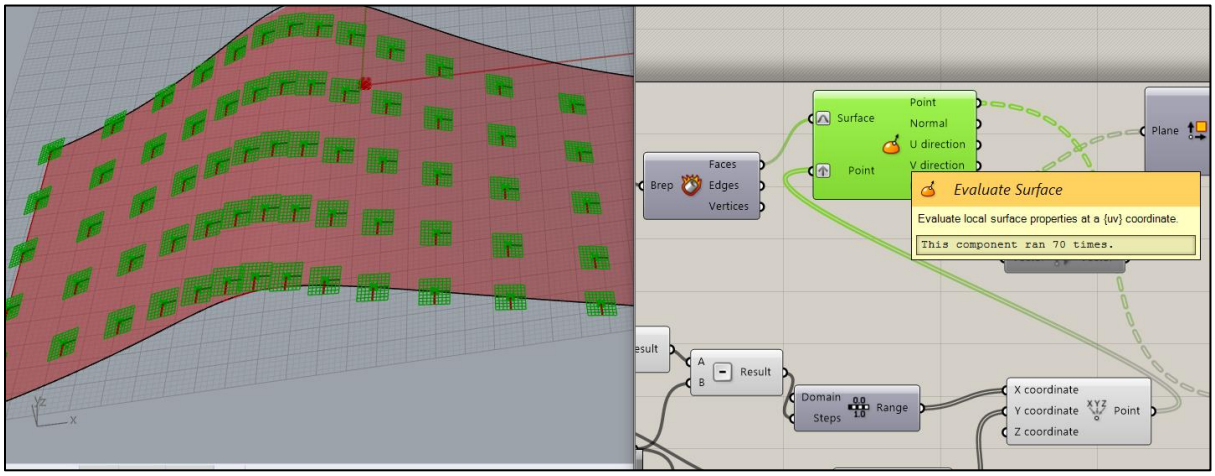
*Cantidad de puntos a evaluar en la superficie*




**Calcular Planos Tangentes a la Superficie**. Al tener los puntos alrededor de la superficie es necesario generar un plano tangente a la superficie, para ello se utiliza un bloque de programación gráfica llamado “Evaluate Surface” como se obversa en la Figura 56.

**Figura 56**

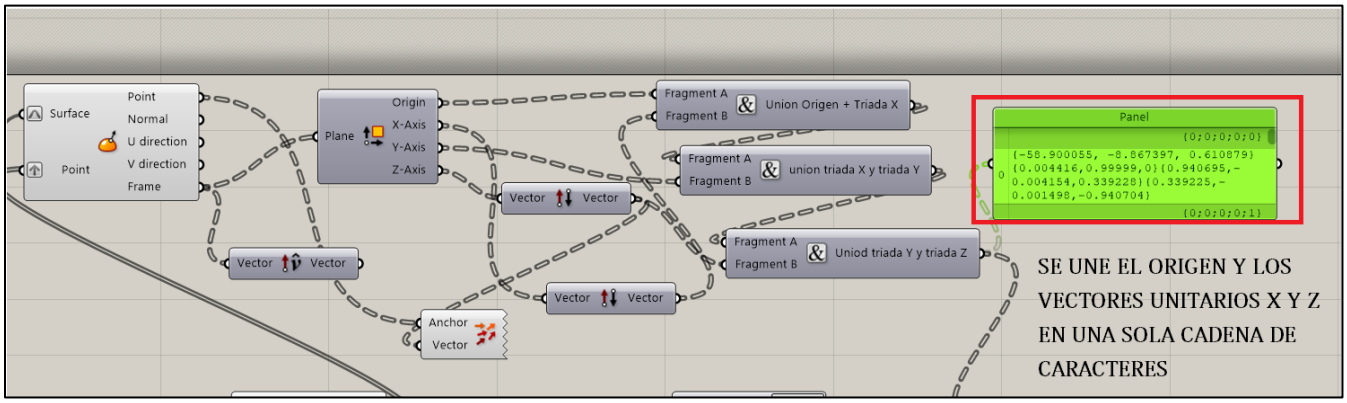
*Planos tangentes a la superficie*




**Unir Puntos y Vectores Unitarios X Y Z en un Bloque**. En la Figura 57 se muestra la unión de los vectores unitarios con el origen de los planos generados por los puntos tangentes a la superficie en una sola cadena de caracteres para su posterior guardado.

**Figura 57**

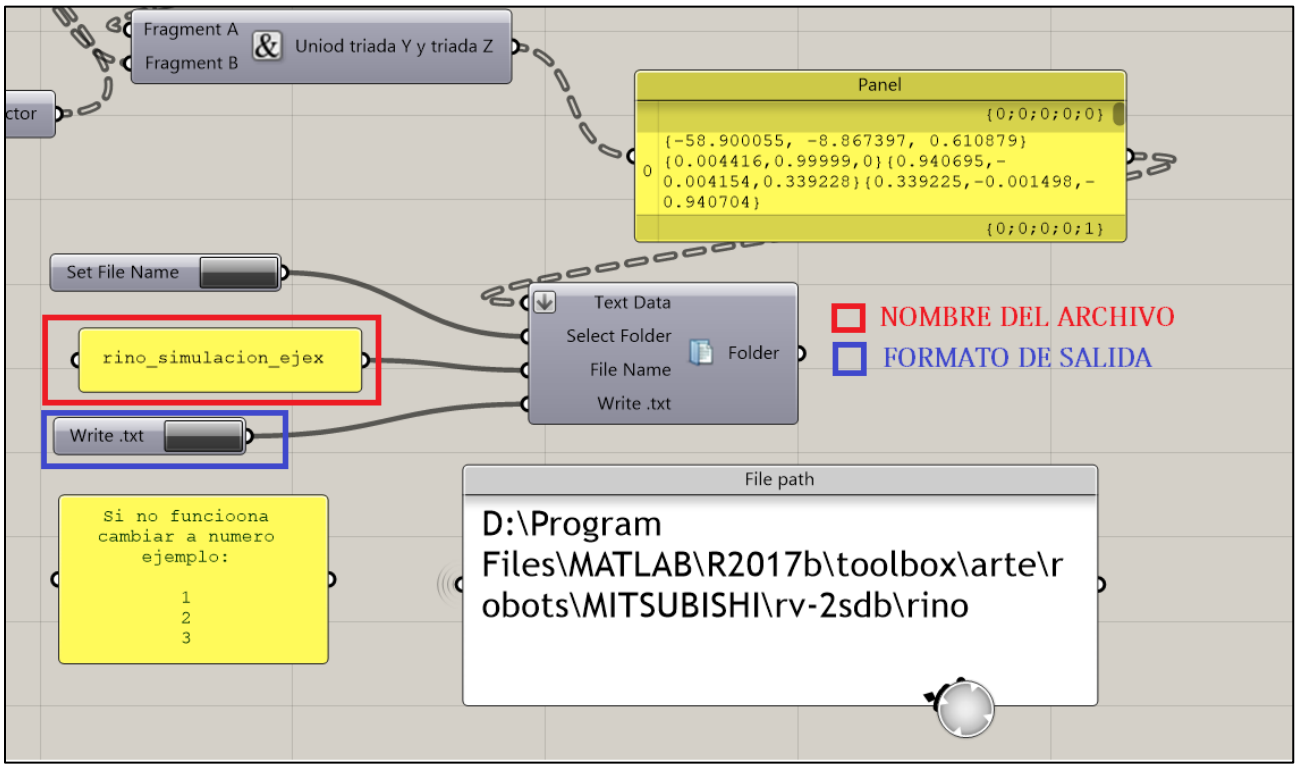
*Unión de los vectores unitarios*



**Guardar el Bloque en Archivo TXT.** Una vez obtenidos los puntos y vectores unitarios en una sola lista de caracteres, es posible copiarlos en un archivo con extensión TXT como se detalla en la Figura 58, para su posterior simulación.

**Figura 58**

*Guardar el Bloque en Archivo TXT*

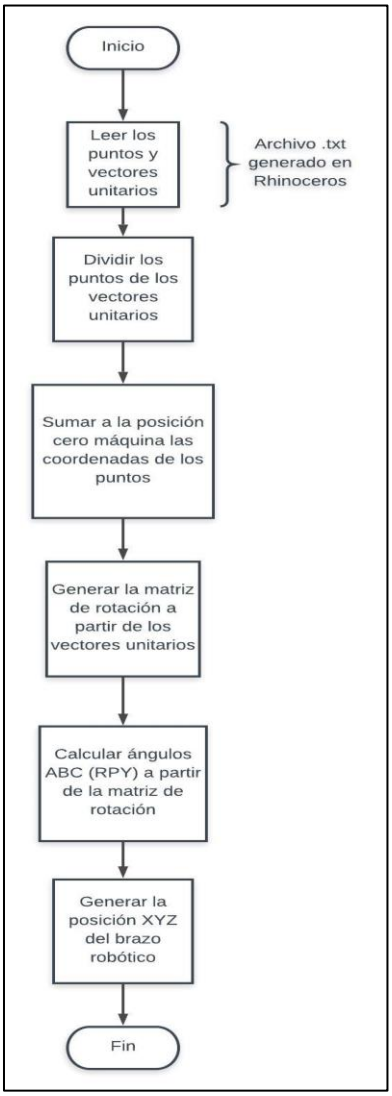



### 4.5.2. *Algoritmo para la Transformación de Puntos y Vectores Unitarios Obtenidos en RHINOCEROS a Posiciones XYZ*

El algoritmo a seguir para la transformación de puntos y vectores unitarios obtenidos en Rhinoceros a posiciones XYZ se puede apreciar en la Figura 59.

**Figura 59**

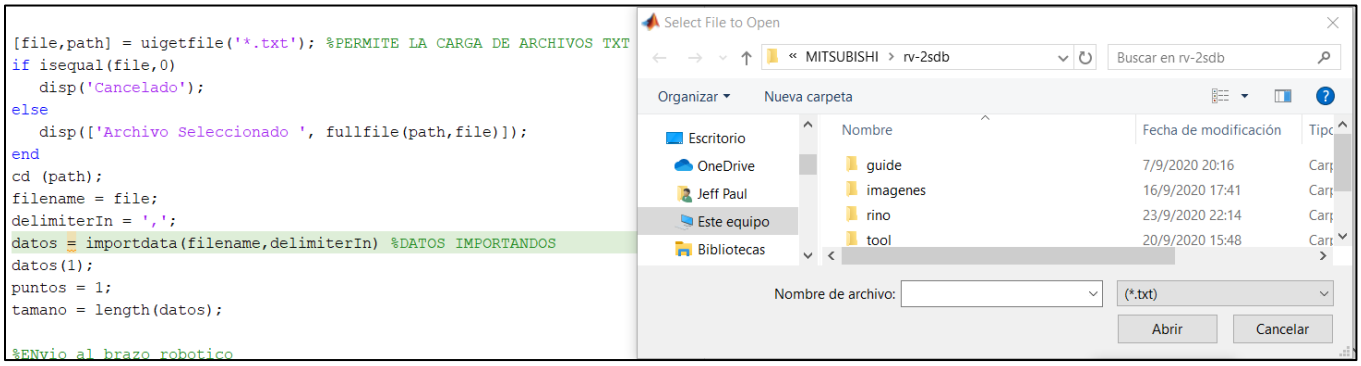
*Flujograma para la transformación de puntos y vectores unitarios a posiciones XYZ*




**Programación para la Transformación de Puntos y Vectores Obtenidos en Rhinoceros en Posiciones XYZ.** En primera instancia se utiliza el comando *uigetfile('*.txt')* que permite a MATLAB seleccionar y adquirir los datos de un archivo con extensión TXT para su posterior utilización como se indica en la Figura 60.

**Figura 60**

*Programación para la Transformación de Puntos y Vectores a Posiciones XYZ*




Una vez cargados los datos, se los divide en puntos y vectores unitarios, calculando primero las coordenadas XYZ mediante la adición entre la posición CERO ROBOT y las coordenadas de los orígenes de los puntos, mediante la programación mostrada en la Figura 61.

**Figura 61**

*Cálculo de coordenadas XYZ*

```
origen(puntos,:) = origen(puntos,:) +[pos_cero_maquina.x pos_cero_maquina.y pos_cero_maquina.z];
```

En segunda instancia, los vectores unitarios son cargados en una matriz de rotación como se muestra en la Ecuación 1 con la programación detallada en la Figura 62 permitiendo encontrar las coordenadas angulares ABC.

$$A = \begin{bmatrix} u_x & v_x & w_x \\ u_y & v_y & w_y \\ u_z & v_z & w_z \end{bmatrix} \qquad \textbf{Ec. 1}$$

**Figura 62**

*Obtención de coordenadas angulares ABC*

```
matriz_rot = [vector_normalx(puntos,:);vector_normaly(puntos,:);vector_normalz(puntos,:)];
abc= tr2rpy(matriz_rot,'deg');
```

En le Figura 63 detalla la unión entre las coordenadas XYZ y coordenadas ABC formando una posición del brazo robótico para su posterior envío al controlador.

**Figura 63**

*Unión entre las coordenadas XYZ y coordenadas ABC*

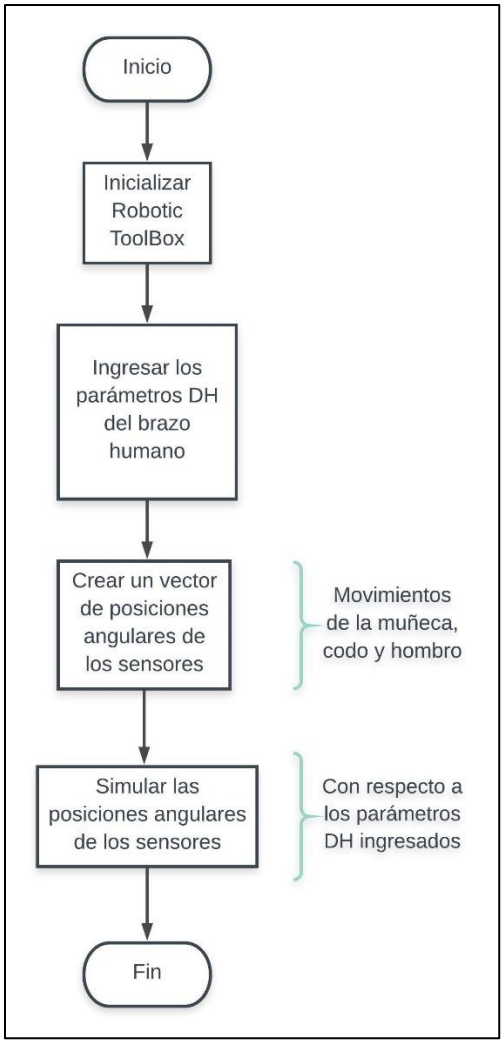

## 4.6. Simulaciones

En esta sección se describió los pasos para desarrollar la simulación de los movimientos del brazo humano hacia el brazo robótico por medio de los Toolbox Robotics y ARTE.

### 4.6.1. *Robotics Toolbox como Simulador del Movimiento con Respecto al Brazo Humano*

**Figura 64**

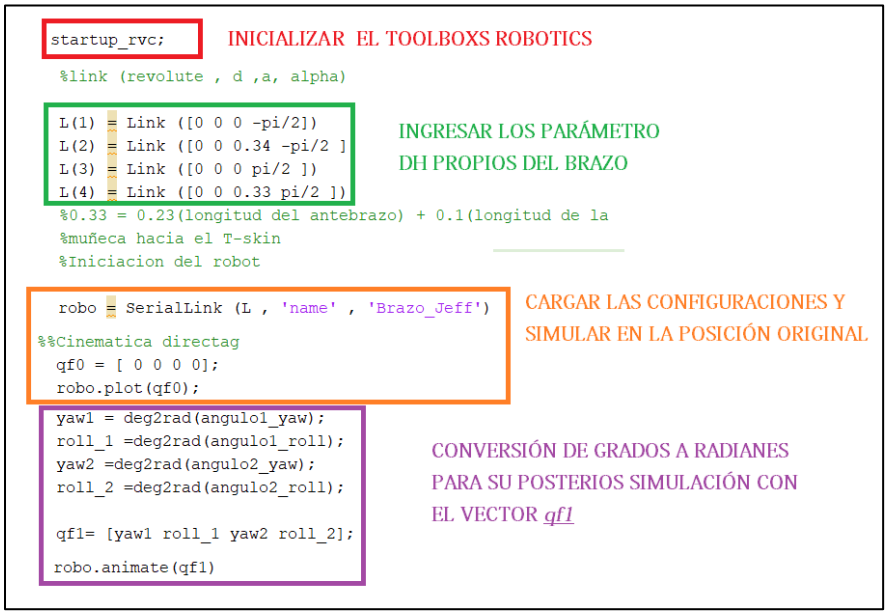
*Flujograma de la configuración del Toolbox Robotics*




**Programación para Simular el Movimiento del Brazo Humano Utilizando Robotics Toolbox**. En la Figura 65 se detalla la programación utilizada para realizar una simulación visual de los movimientos del brazo humano.

**Figura 65**

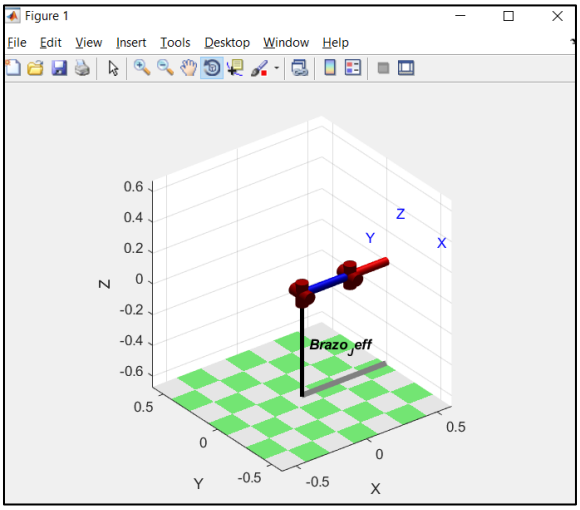
*Programación para simular el movimiento del brazo humano usando Robotics Toolbox*




Obteniendo como resultado como se muestra en la Figura 66.

**Figura 66**

*Simulación por medio de Robotics Toolbox*

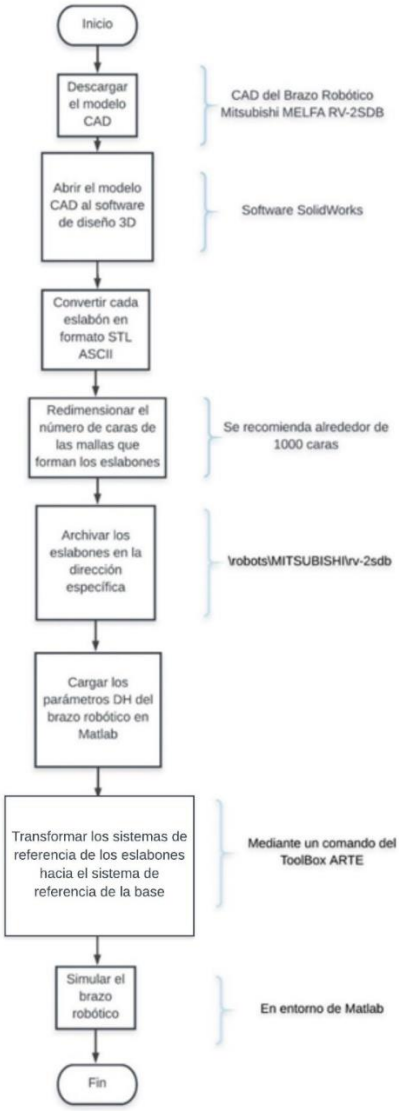


### 4.6.2. *Algoritmo para Utilizar ARTE como Simulador del Brazo Robótico*

En la Figura 67 se puede observar el flujograma del algoritmo para utilizar ARTE como simulador del brazo robótico.

**Figura 67**

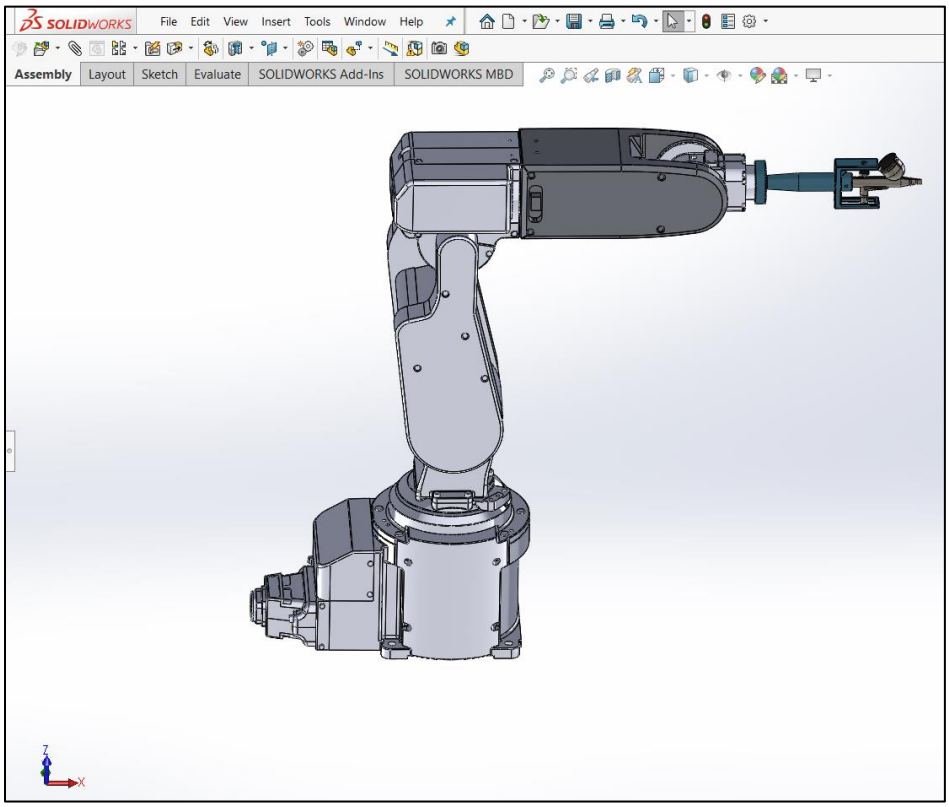
*Flujograma del algoritmo para utilizar ARTE como simulador del brazo robótico*



**Abrir el Modelo CAD.** Se utilizó el software SolidWorks para realizar el ensamble del brazo robótico con todos sus eslabones incluyendo la herramienta de trabajo como se observa en la Figura 68.

**Figura 68**

*Importación del modelo CAD*

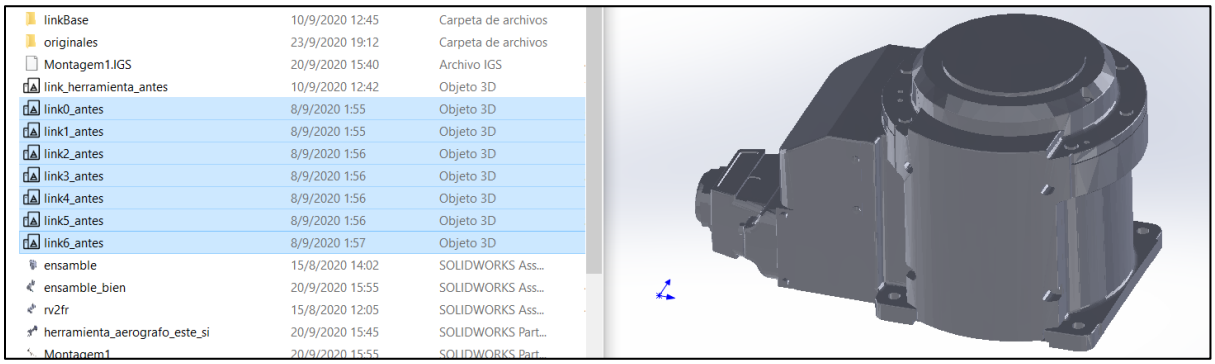



*Nota.* Las piezas del robot MELFA RV-2SDB fueron descargadas del sitio web (Electrical, 2017). Mitsubishi (Global Factory Automation) provee de manera abierta al público las piezas de sus Robots en formato compatible con SolidWorks para su utilización científica.

**Convertir los Eslabones del Brazo Robótico a Formato STL.** Se utilizó SolidWorks para convertir los archivos de cada eslabón al formato STL ASCII. Se debe guardar cada eslabón partiendo de la base, hasta el eslabón del sexto eje con los nombres link0_base.stl, link1_base.stl, link2_base.stl y así sucesivamente.

**Figura 69**

*Eslabones del brazo robótico en formato STL*

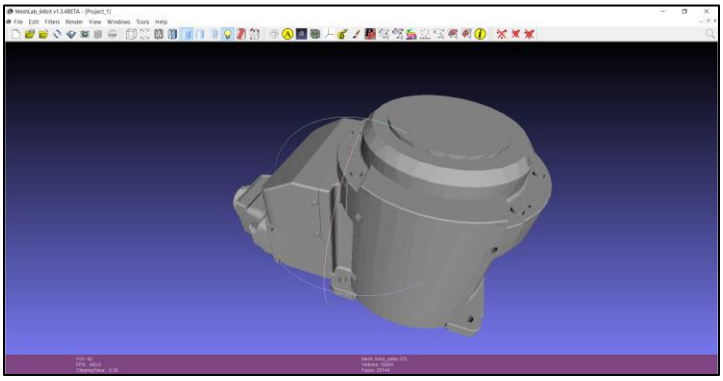



**Redimensionar el Número de Caras de las Mallas que Forman los Eslabones.** Debido a que algunos de los eslabones poseen mallas muy densas, Matlab tomará mucho tiempo en lograr graficarlas, por lo que es mejor reducir la cantidad de parches que forman las mallas de los eslabones. Se utilizó el programa MeshLab para reducir dicha cantidad.

Primero se crea un nuevo proyecto (pestaña File), luego se selecciona la opción *Import Mesh* para introducir el primer eslabón en formato STL ASCII como se observa en la Figura 70.

**Figura 70**

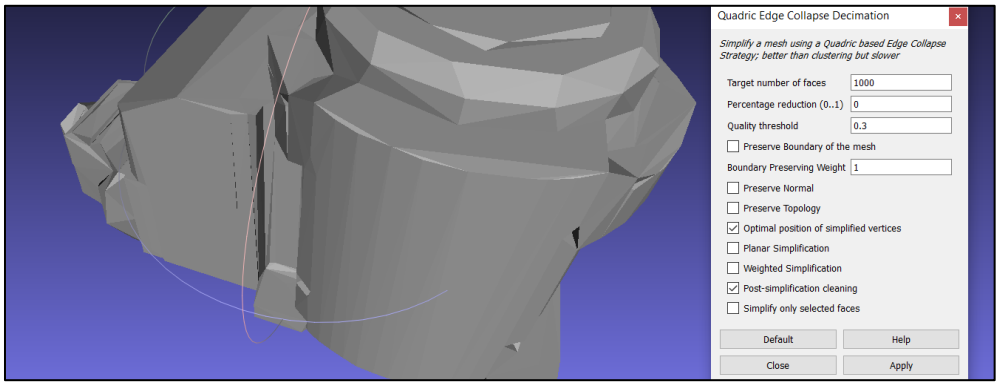
*Piezas del robot en el programa MeshLab*



Una vez cargado el primer eslabón se cambiará el número de parches que forman su malla, escogiendo la opción del menú *Filters*. En la pestaña *Filters* existe el comando *Remashing, Simplification and Reconstruction*, que permite elegir la opción *Quadric Edge Collapse Decimation* para cambiar el número de parches que forman su malla. Se digita un valor de 1000 en *Target number of faces* como se puede ver en la Figura 71.

**Figura 71**

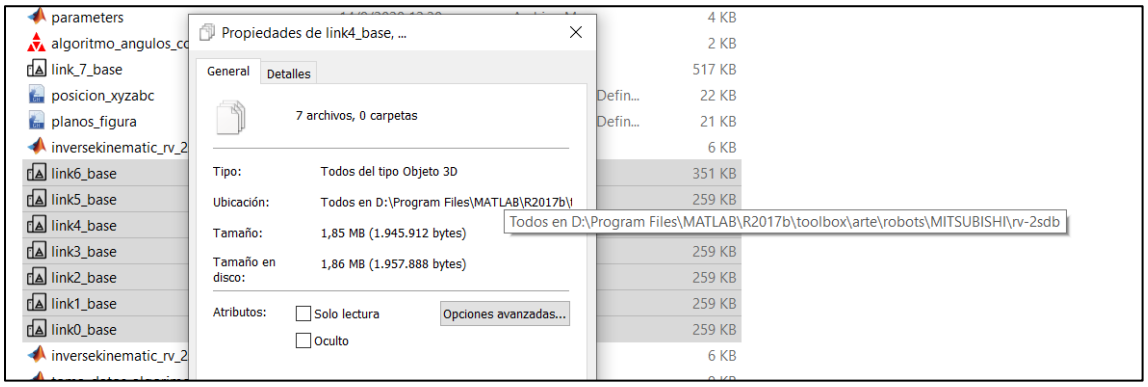
*Target number of faces*



**Archivar los Eslabones en la Dirección Específica.** Para archivar los eslabones tratados, se dirige a la pestaña *File* en la opción *Export Mesh* (desactivar el parámetro *Binary encoding)*, para que el archivo tenga un formato de texto. Luego se repite el proceso para los demás eslabones, como se puede visualizar en la Figura 72.

**Figura 72**

*Eslabones archivados en la Dirección Específica*




**Cargar los Parámetros DH del Brazo Robótico en MATLAB.** Para cargar los parámetros DH del brazo robótico mencionados en la sección 4.2.1, se abre el archivo *parameters.m* para editarlo y colocar los parámetros correctos en base a las dimensiones del modelo (RV-2SDB). Como se puede apreciar en la Figura 73.

**Figura 73**

*Código para cargar los Parámetros DH del Brazo Robótico*

```
%kinematic data
%robot.DH.theta= '[q(1) q(2)-pi/2 q(3)+pi/2 q(4) q(5) q(6)]';
robot.DH.theta= '[q(1) q(2)-(pi/2) q(3) q(4) q(5) q(6)]';
robot.DH.d='[0.295 0 0 0.270 0 0.07]';
robot.DH.a='[0 0.23 0.05 0 0 0]';
robot.DH.alpha= '[-pi/2 0 -pi/2 pi/2 -pi/2 0]';


% robot.DH.theta= '[q(1) q(2)-pi/2 q(3) q(4) q(5) q(6)]';
% robot.DH.d='[0.400 0 0 0.285 0 0.085 ]';
% robot.DH.a='[0 0.340 0.05 0 0 0]';
% robot.DH.alpha= '[-pi/2 0 -pi/2 pi/2 -pi/2 0]';
%number of degrees of freedom
robot.DOF = 6;

%rotational: R, translational: T
robot.kind=['R' 'R' 'R' 'R' 'R' 'R'];

%Jacobian matrix
robot.J=[];
```

**Transformar los sistemas de referencia de los eslabones al sistema de referencia de la base.** Una vez obtenidos los eslabones con extensión STL y el archivo parameters.m se utiliza el comando *transdorm_to_own* para crear nuevos archivos con el nombre *link0.stl, link1.stl hasta el link6.stl* que poseen un nuevo origen dependiendo de los parámetros DH ingresados.

**Figura 74**

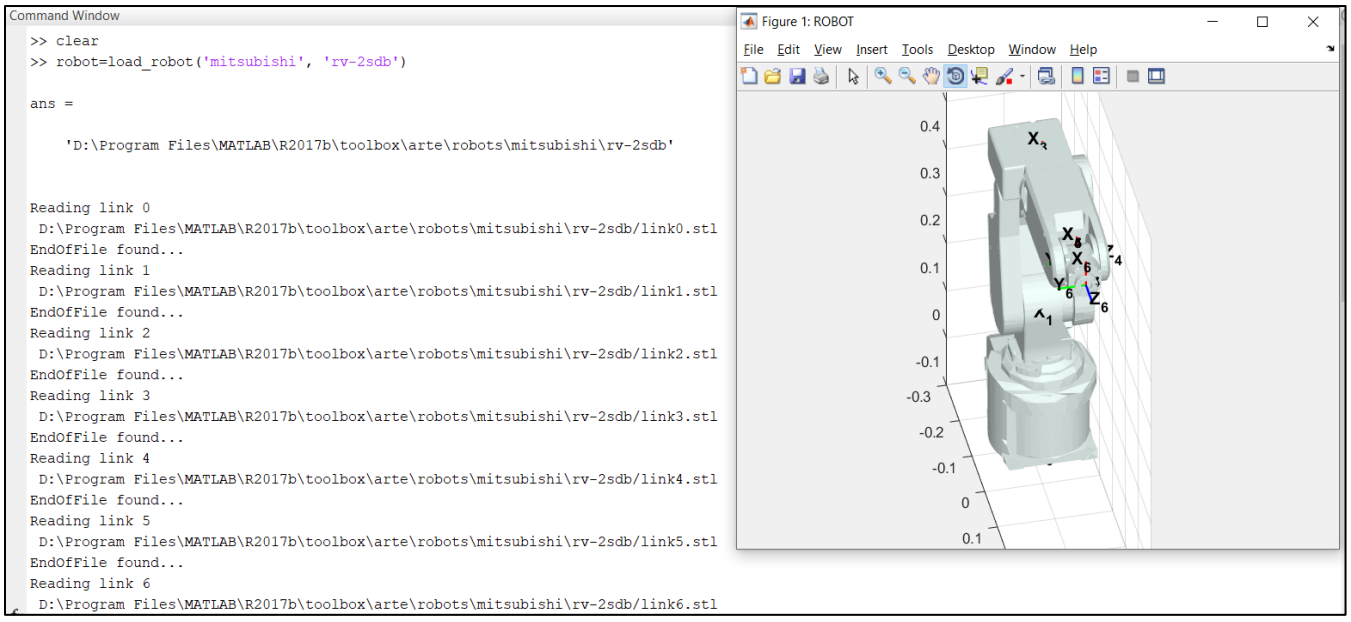
*Código para transformar los sistemas de referencia de los eslabones al de la base*

```
transform_to_own('MITSUBISHI','rv-2sdb\tool',1000)
```

**Simular el brazo robótico.** El comando que permite cargar los archivos STL es *load_robot('mitsubishi','rv-2sdb')* como se muestra en la Figura 75.

**Figura 75**

*Simular el brazo robótico mediante Toolbox ARTE*



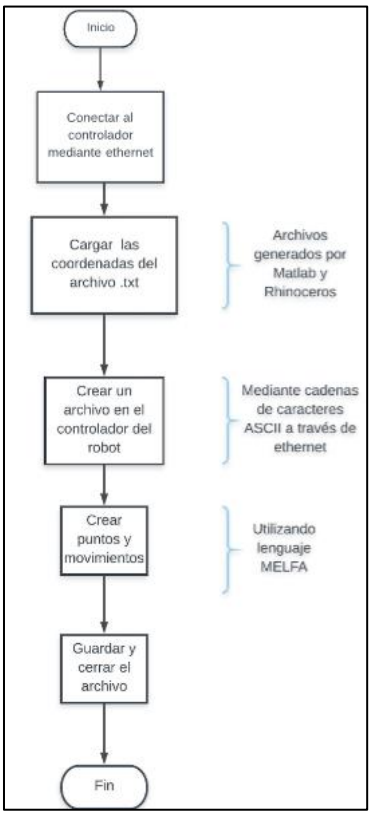

Para realizar algún movimiento adicional se abre una interfaz con el comando *teach*.

## 4.7. Algoritmo para el Envío de Posiciones hacia el Controlador Mitsubishi CR1DA-711

En la Figura 76 se puede observar el flujograma con los pasos para enviar las posiciones hacia el controlador Mitsubishi CR1DA-711.

**Figura 76**

*Flujograma para el envío de posiciones hacia el controlador CR1DA-711*




**Programación del algoritmo para el envío de posiciones al controlador del brazo robótico.** En primera instancia se procede a conectar el controlador a través de ethernet como se indica en la Figura 77 definiendo la dirección IP 192.168.0.20 y el puerto 10001.

**Figura 77**

*Código para ingresar la dirección IP*

```
t = tcpip(string(ip_direccion),puerto_ip);
%Abrir el puerto serial
fopen(t);
```

En la Tabla 20 se visualiza una lista de los comandos más importantes en ASCII.

**Tabla 20**

*Comandos para el control externo del controlador CR1DA-771*

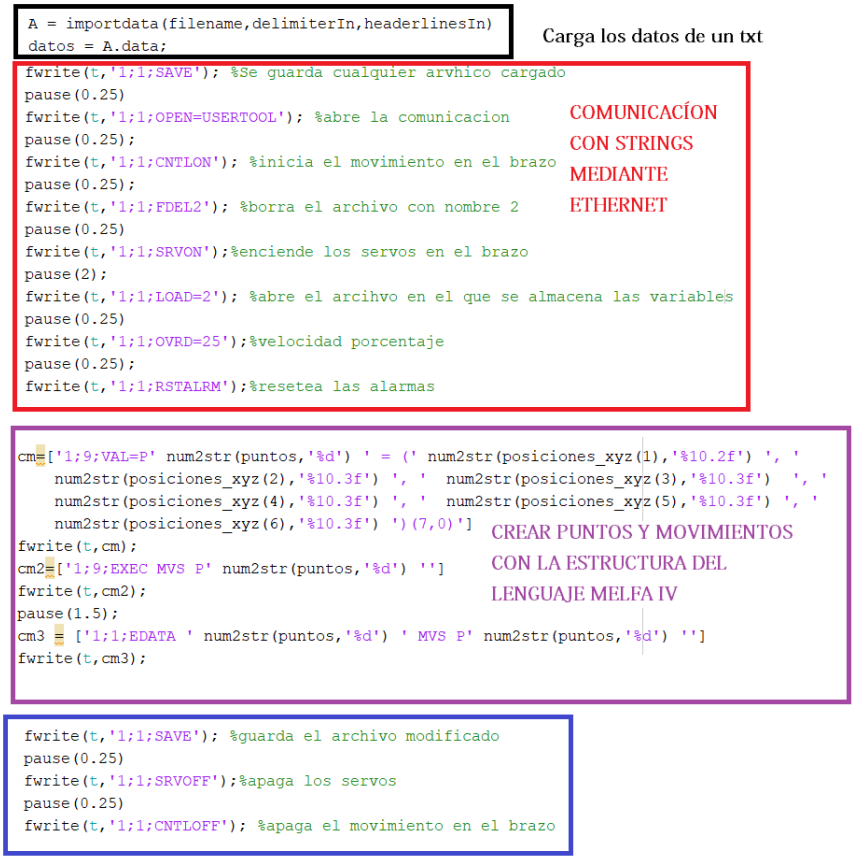
| Función | Comando | Ejemplo | Retorno |
|---|---|---|---|
| **Abre la comunicación hacia el controlador** | OPEN | 1;1;OPEN=USERTOOL | QoK3F;3F;7,0;3,5,A,1E,32,46,64;MB4;PRM;RV-4A;CRn-5xx;MELFA;03-11-19;Ver.J4;ENG; COPYRIGHT(C)1999-2003 MITSUBISHI ELECTRIC CORPORATION ALL RIGHTS RESERVED;1;1;8; |
| **Carga un archivo para editarlo** | LOAD=<Nombre del programa> | 1;1;LOAD=100 | QoK |
| **Crea una variable pueden ser enteros, posiciones xyz o por juntas.** | VAL=<Nombre de la variable>=<Valor> | 1;9;VAL=M1=3 | Qok |
| **Permite la ejecución de comandos MELFA-BASIC IV** | EXEC<Instrucción en el lenguaje MELFA-BASIC IV o > | 1;1;EDATA10 MOV P1 | Qok |

| Función | Comando | Ejemplo | Retorno |
|---|---|---|---|
| **Enciende los servos** | SRV<ON/OFF> | 1;1;SRVON | Qok |
| **Muestra la posición del efector final en distintos tipos.** | <Tipo>POS<información><br>**Tipo:**<br>J: Juntas<br>P: XYZ<br>X:3-axis<br>R: Cilíndricas<br>**Información:**<br>1-8: Un solo eje<br>F: Todos los ejes | 1;1;PPOSF | QoKX;290.62;Y;-0.09;Z;11.26;A;-179.94;B;-0.26;C;179.93;L1;0.00;;7,0;100;0.00;00000000 |

En la Figura 78 se detalla la programación utilizada para el envío de posiciones XYZ calculadas en MATLAB hacia el controlador que pueden ser ejecutadas en cualquier momento.

**Figura 78**

*Código de programación utilizado para el envío de posiciones XYZ al controlador*




## 4.8. Algoritmo que Utiliza el HMI para el Control y Programación del Brazo Robótico Mitsubishi

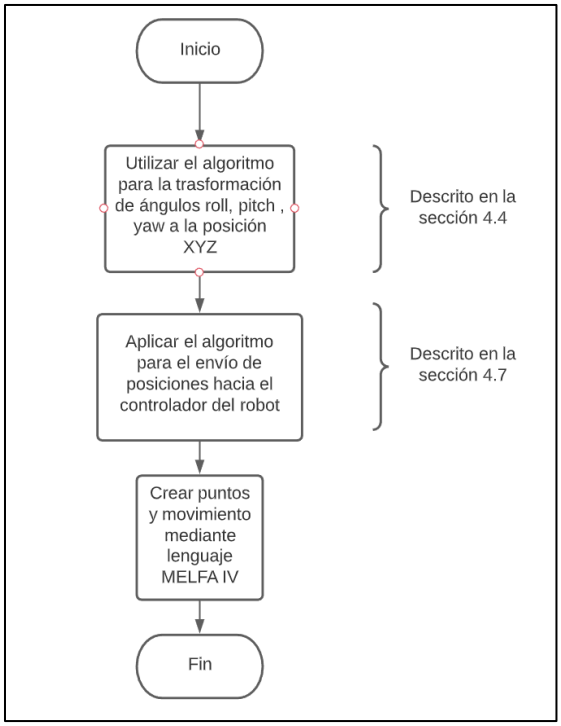
Debido a que el HMI posee varias formas de controlar y programar el brazo robótico se lo divide en: control por tiempo real, programación mediante movimientos y programación offline con Rhinoceros.

### 4.8.1. *Algoritmo para el Tiempo Real*

En la Figura 79 se detalla el algoritmo que permite controlar al brazo robótico en sincronía con los movimientos del brazo humano, estos movimientos son captados por los sensores Tactigon.

**Figura 79**

*Flujograma para el tiempo real*



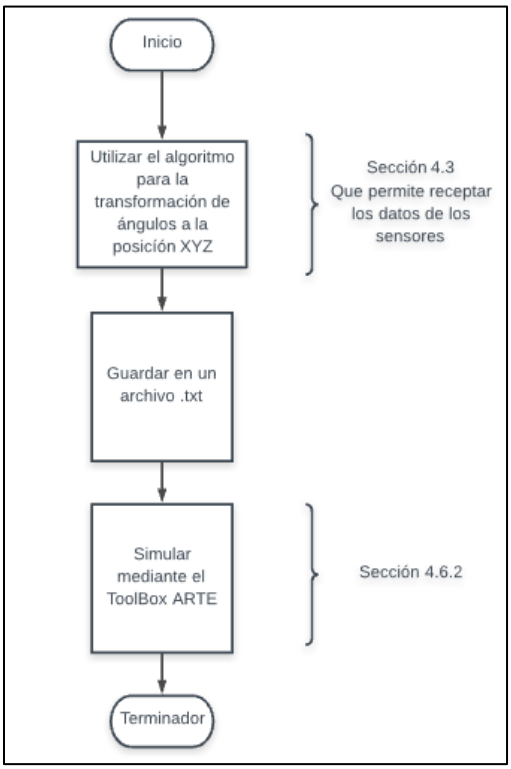

En primera instancia se utiliza el algoritmo para la transformación de ángulos RPY a posiciones XYZ descrita en la sección 4.4 para obtener dichas posiciones y luego ser cargadas mediante programación MELFA, la cual es una etapa del algoritmo que envía posiciones al controlador.

### 4.8.2. *Algoritmo para la Programación Mediante Movimientos Biomecánicos*

Esta programación consiste en capturar todos los movimientos generados por el brazo humano y guardarlos en un archivo TXT para su posterior simulación y envío de datos al controlador permitiendo su ejecución en cualquier momento.

**Figura 80**

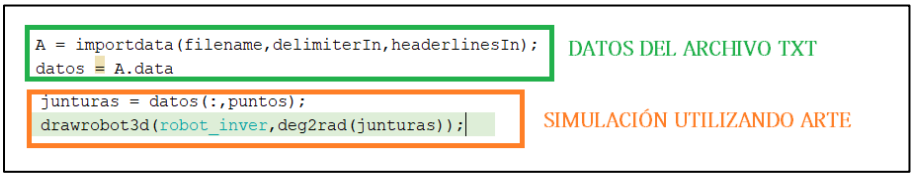
*Flujograma para la programación mediante movimientos*




Una vez guardadas las posiciones XYZ en un archivo TXT se procede a utilizar ARTE como simulador, utilizando la programación de la Figura 81, el cual permite mostrar el brazo robótico evaluado en las posiciones cargadas.

**Figura 81**

*Código para mostrar el brazo robótico evaluado en posiciones*

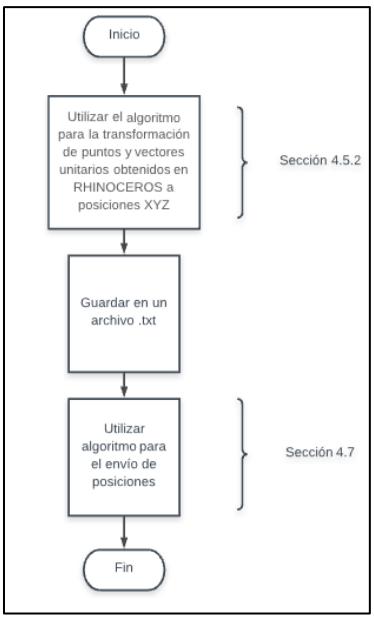



### 4.8.3. *Algoritmo para la Programación Offline con Rhinoceros*

La programación offline consiste en generar puntos y orientaciones con el software Rhinoceros los cuales son guardados en un archivo TXT para ser simulados y enviados al controlador. En la Figura 82 se describe el proceso para programar las posiciones del brazo robótico mediante Rhinoceros.

**Figura 82**

*Flujograma para la programación offline con Rhinoceros*



## 4.9. Implementación de la interfaz para la Comunicación del Algoritmo Flexible

Se realiza una interfaz en MATLAB que contiene 4 ventanas principales detalladas de la siguiente manera:

**Ventana de Comunicación o Principal.** En la Figura 83 se muestra la comunicación entre la tarjeta de control y MATLAB mediante comunicación serial, en este caso es el COM6 y la velocidad de transmisión (baudios) es de 9600. Además, se muestra la comunicación entre MATLAB y el controlador mediante ethernet utilizando la IP 192.168.0.20 y el puerto 10001 por defecto.

**Ventana de tiempo real.** En la Figura 84 se observa la ventana de tiempo real. Para iniciar se debe presionar el botón inicio para comenzar con el algoritmo. El sensor Tactigon V1.0 es el encargado de iniciar o culminar la toma de datos (el promedio de toma de datos es de dos posiciones por segundo), es posible orientar el área de trabajo seleccionando los distintos botones. Para finalizar la simulación en tiempo real es necesario presionar en el botón finalizar.

**Ventana de algoritmo con movimientos.** En la Figura 85 se visualiza la ventana de tiempo real. Se inicia la toma de posiciones con respecto al movimiento del brazo humano, luego se procede a seleccionar el área de trabajo y por último se debe pulsar el botón finalizar para continuar con el guardado y filtrado de las posiciones.

**Ventana de simulación.** En la Figura 86 se divide en dos tipos de simulaciones: algoritmo de movimiento y Offline Rhino los cuales permiten simular cada uno de los tipos de programación. Se debe tomar en cuenta el área de trabajo, debido a que si no se posicionan bien los orígenes de posiciones pueden producirse errores en el envío de posiciones al controlador.

**Figura 83**

*Ventana de Comunicación o Principal*

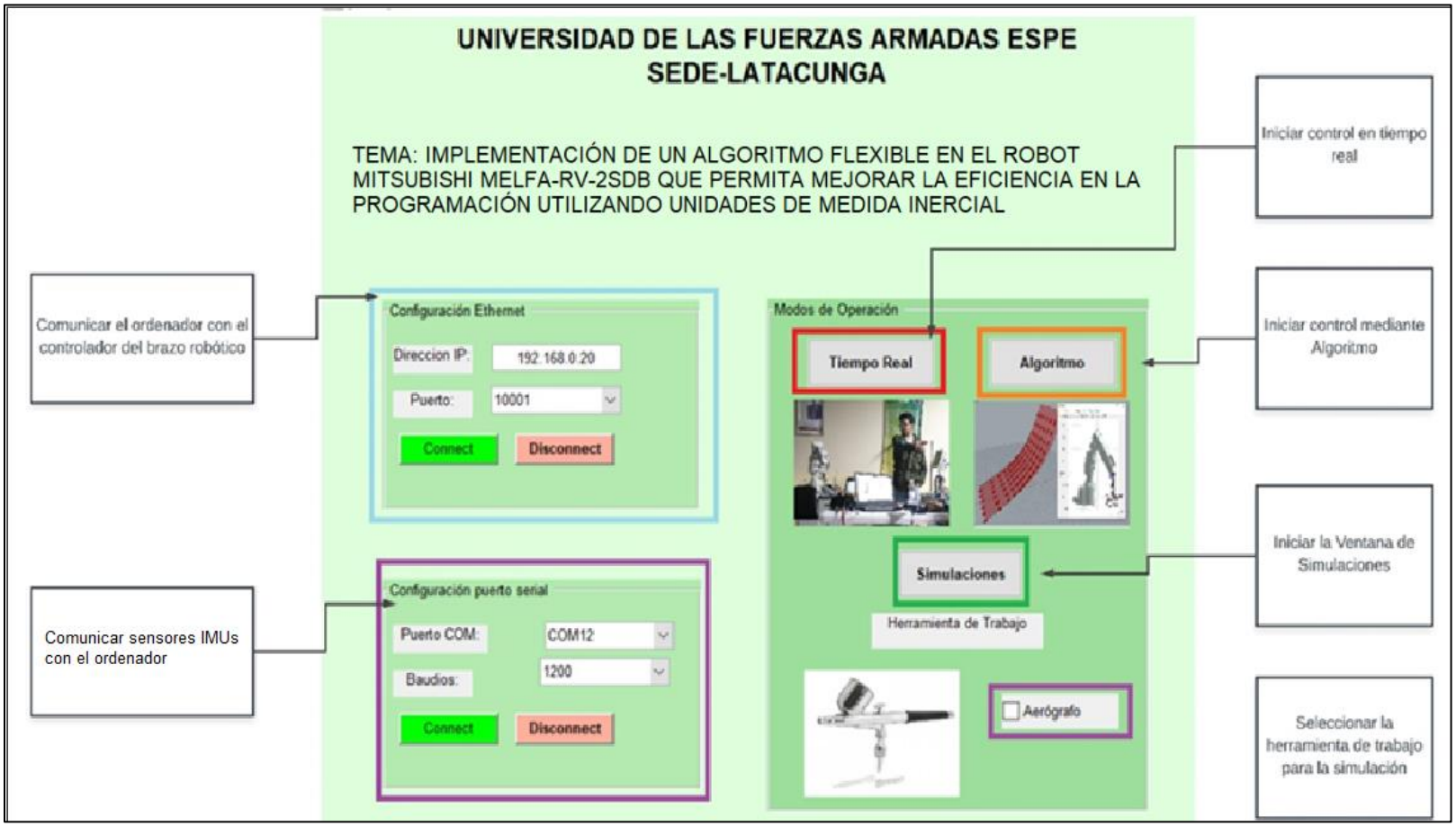


**Figura 84**

*Ventana de tiempo real*

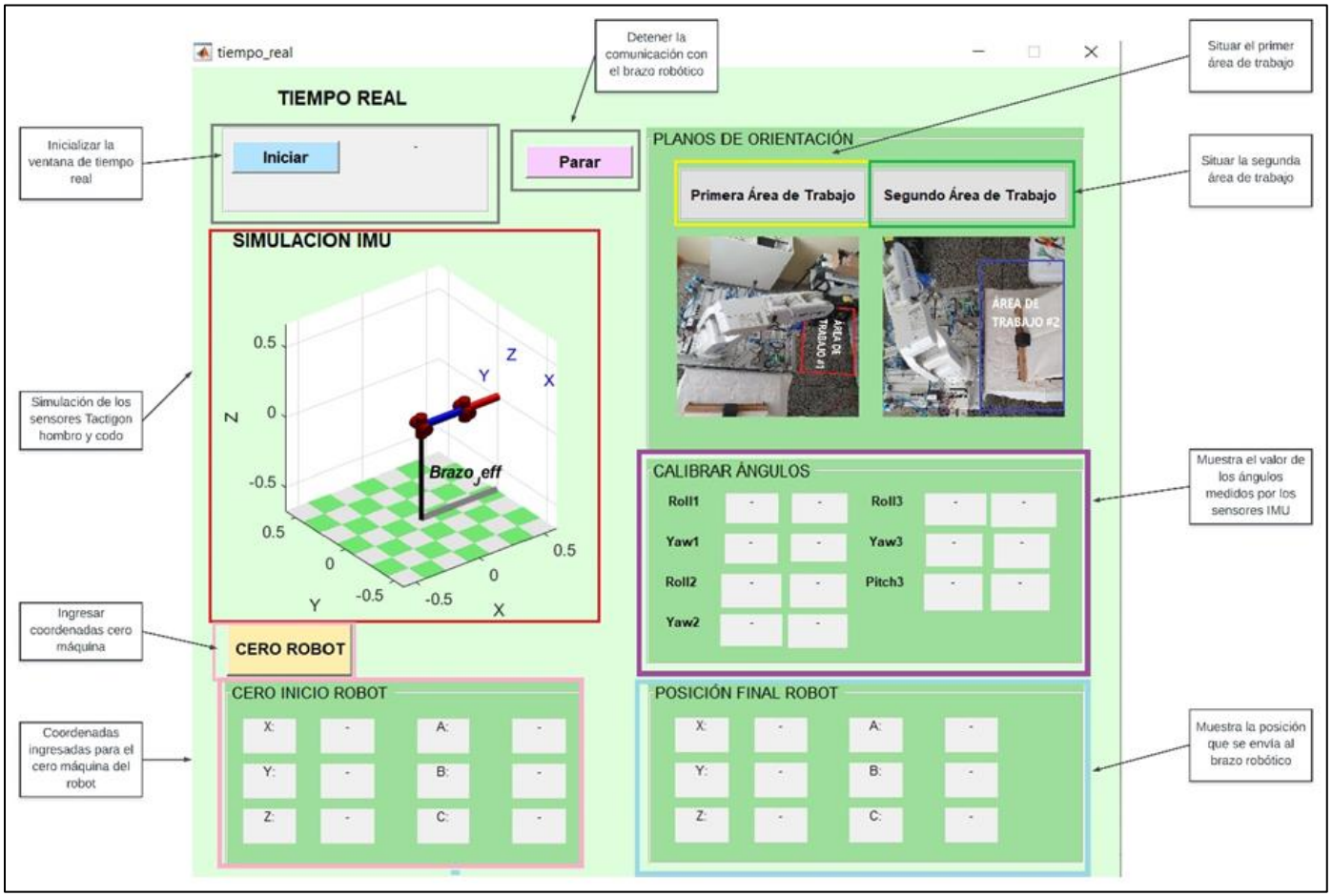


**Figura 85**

*Ventana de algoritmo con movimientos*

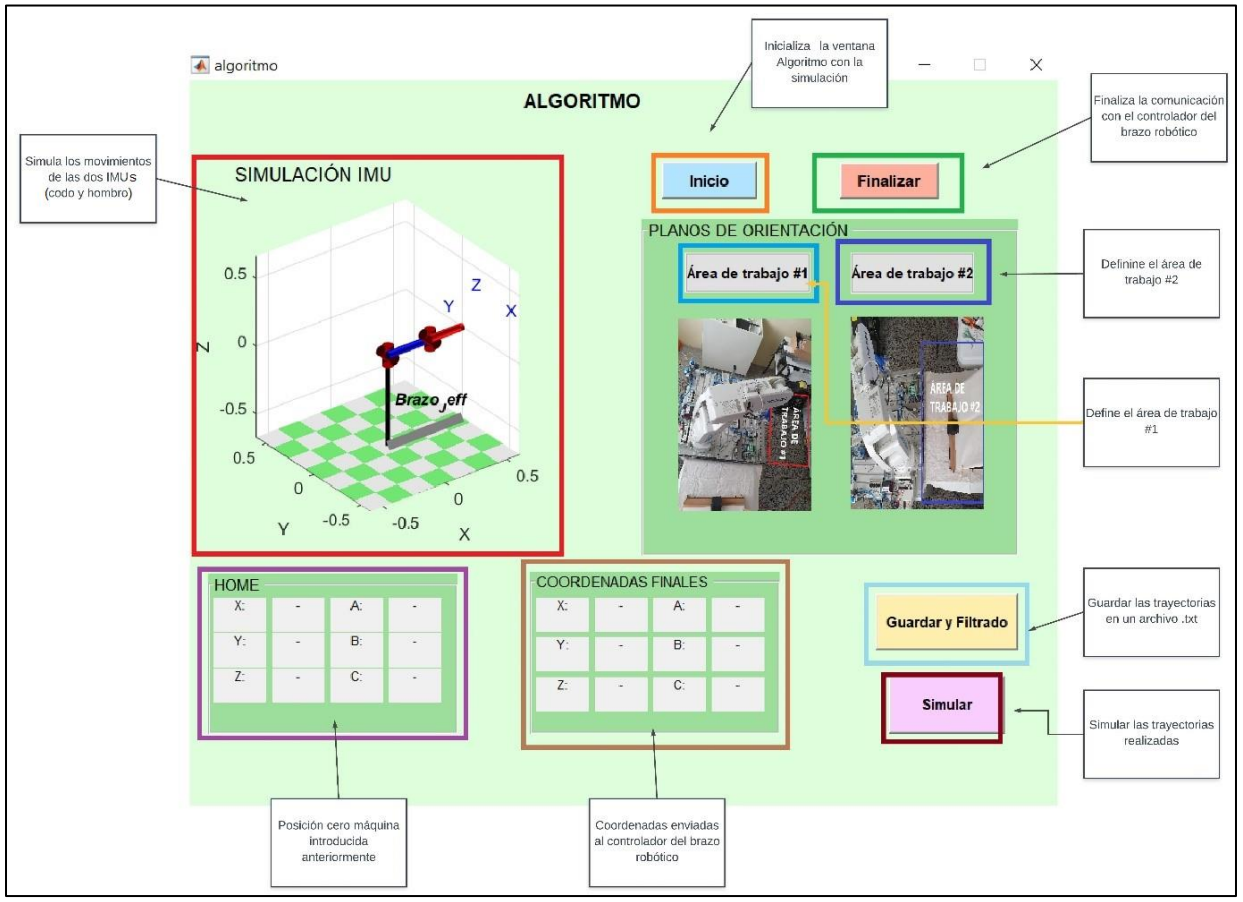
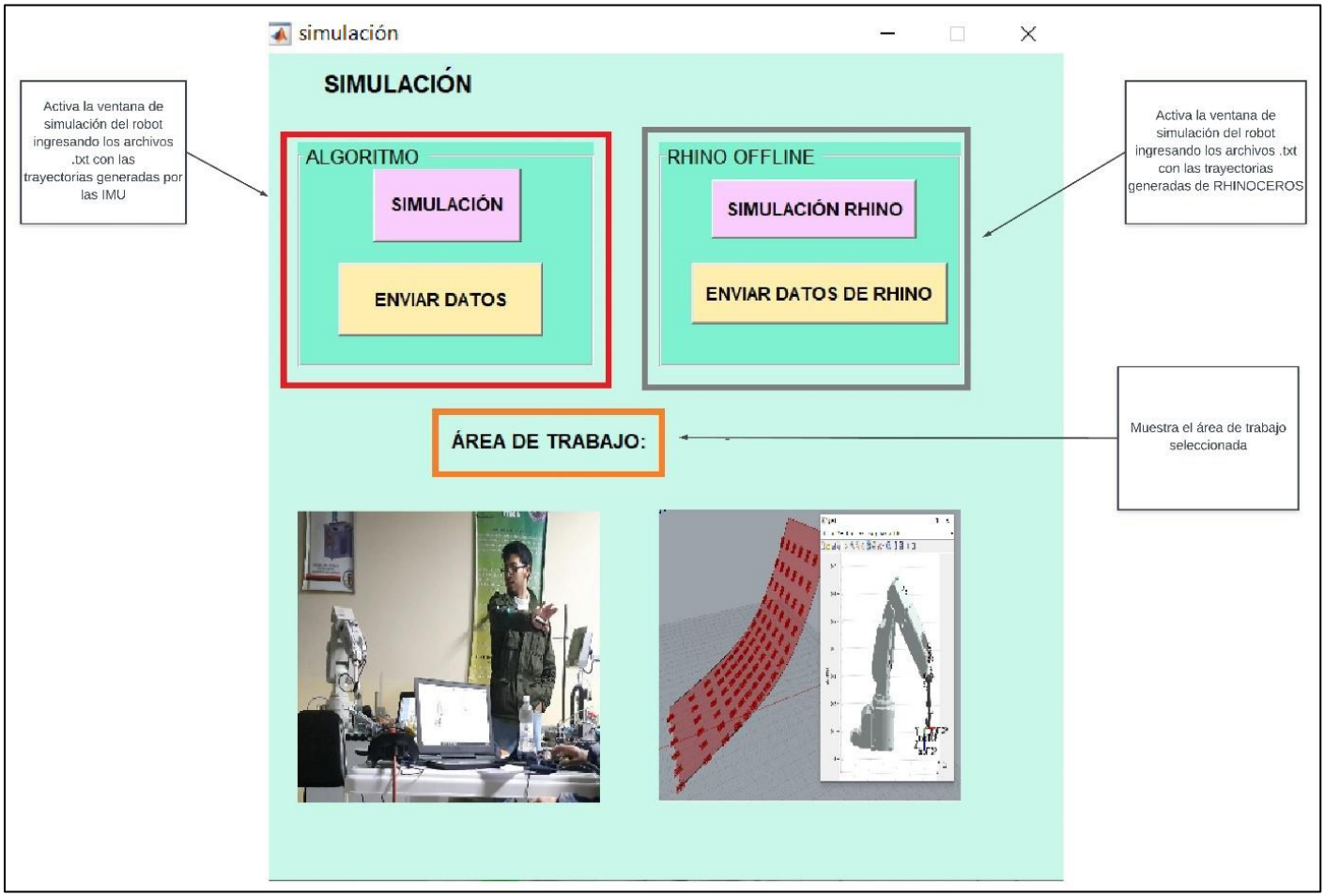


**Figura 86**

*Ventana de simulación*



# CAPÍTULO V

# PRUEBAS DE FUNCIONAMIENTO Y ANÁLISIS DE RESULTADOS

En el presente capítulo se evidencian las pruebas de funcionamiento del algoritmo flexible, además se realiza el análisis de los resultados obtenidos, las pruebas fueron enfocadas en distintos ámbitos para determinar el comportamiento del sistema, tales como: movimientos del hombro, codo y muñeca, tiempos de programación y seguimiento de trayectorias. Además, se implementó el algoritmo flexible para la aplicación de pintura. Dichas pruebas ayudaron a validar la hipótesis planteada.

## 5.1. Pruebas de Biomecánica

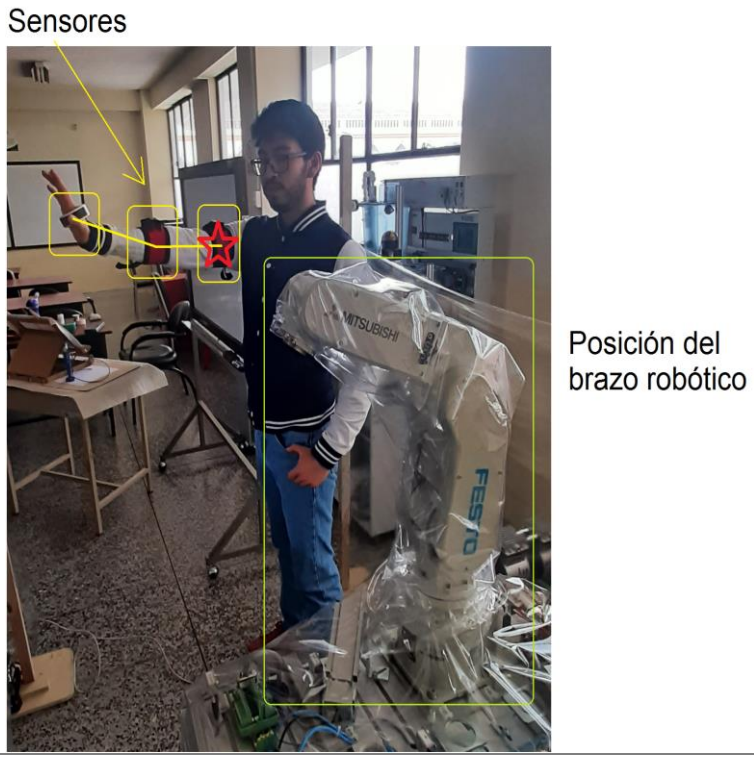
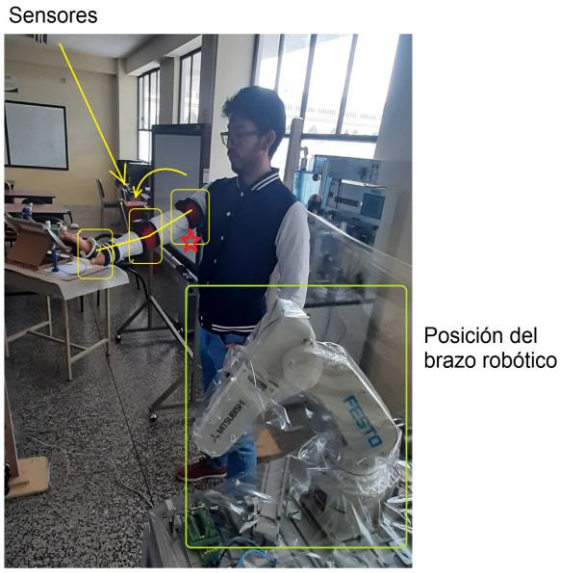
Se realizaron movimientos basados en la biomecánica del brazo humano (sección 2.1) para determinar la respuesta del algoritmo flexible implementado.

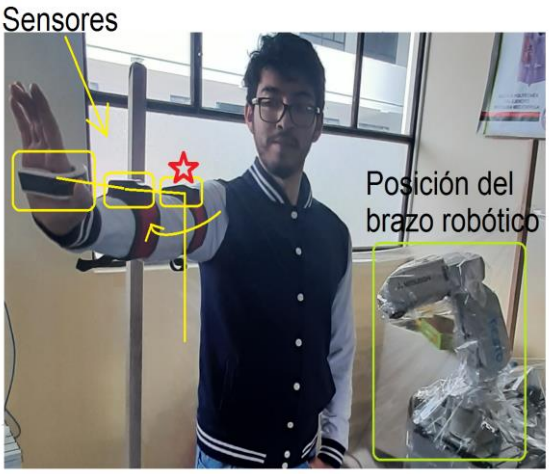
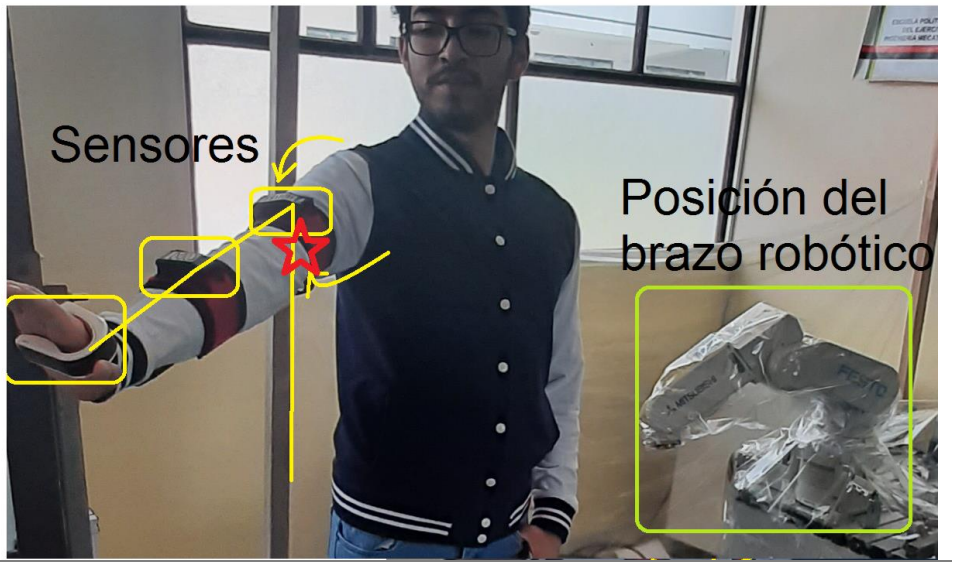
### 5.1.1. *Hombro*

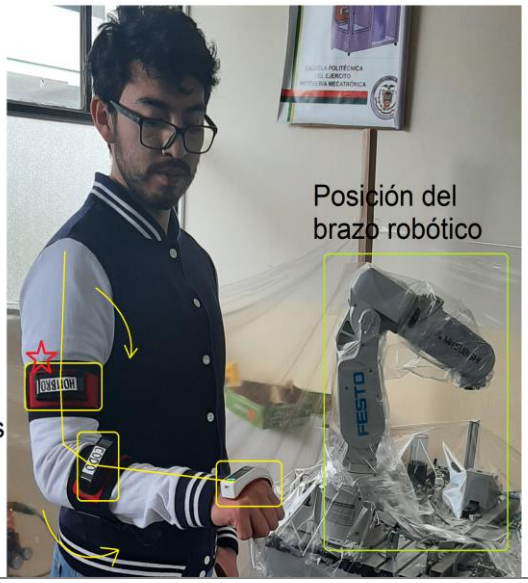
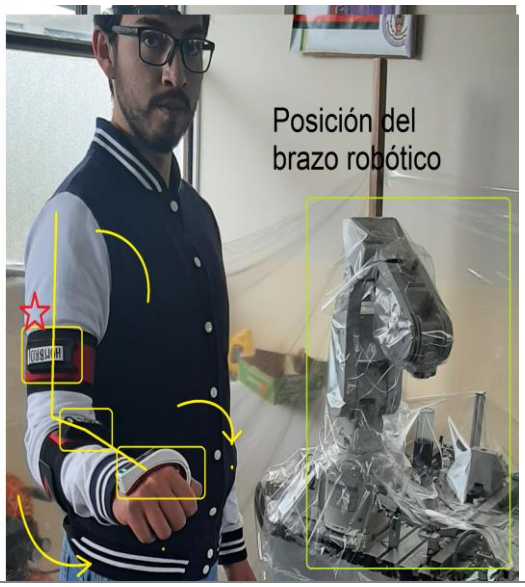
Para el movimiento del hombro, se requieren los valores de las variables, angulo1_roll y angulo1_yaw mencionadas en la sección 4.4.4, del sensor Tactigon V1.0 ubicado en el hombro. La posición final de cada movimiento se puede observar en la Tabla 21.

**Tabla 21**

*Biomecánica del hombro*

| Movimiento | Posición |
|---|---|
| **Flexión** |  |
| **Extensión** |  |

| **Movimiento** | Posición |
|---|---|
| **Abducción** |   |
| **Aducción** |   |

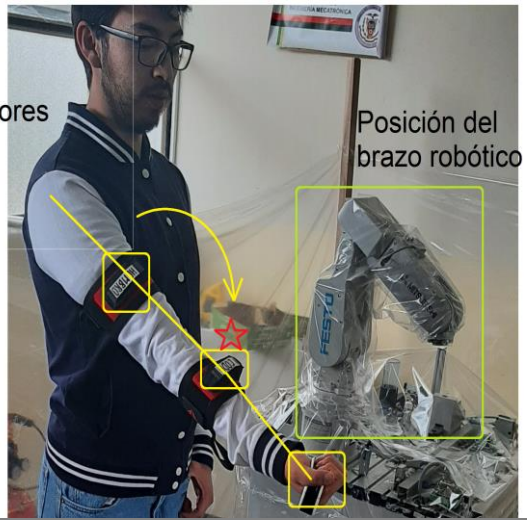
| **Movimiento** | Posición |
|---|---|
| **Rotación medial** |  |
| **Rotación lateral** |  |

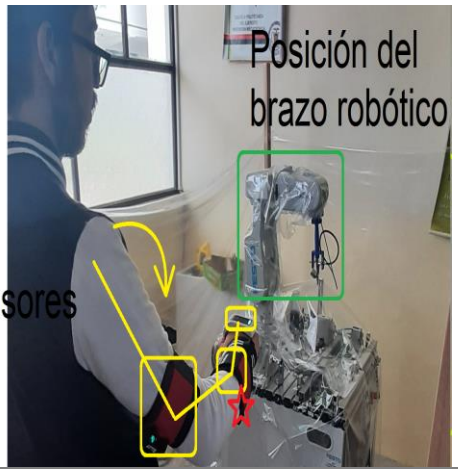
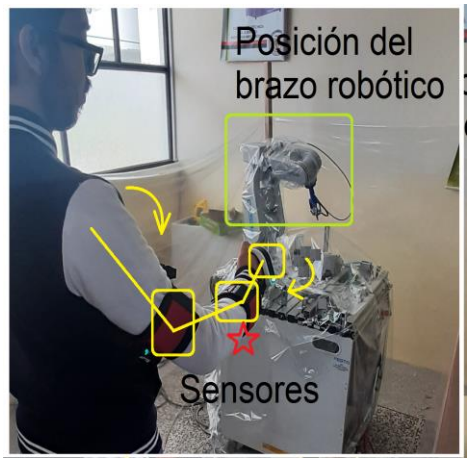
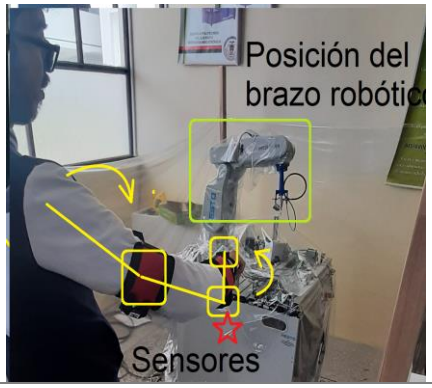
### 5.1.2. *Codo*

Para el movimiento del codo, se requieren los valores de las variables angulo2_roll, angulo2_yaw mencionadas en la sección 4.4.4, del sensor Tactigon V1.0 ubicado en el codo. La posición final de cada movimiento se puede observar en la Tabla 22.

**Tabla 22**

*Biomecánica del codo*

| **Movimiento** | **Posición** |
|---|---|
| **Flexión** |  |
| **Extensión** |  |

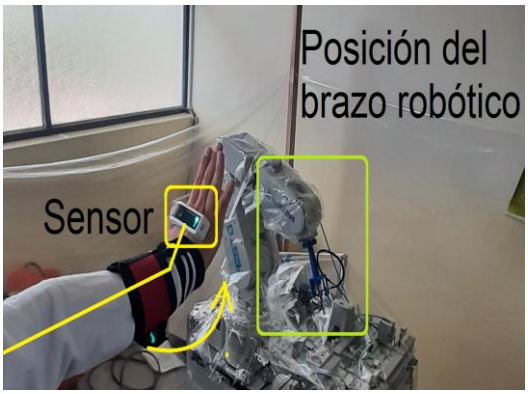
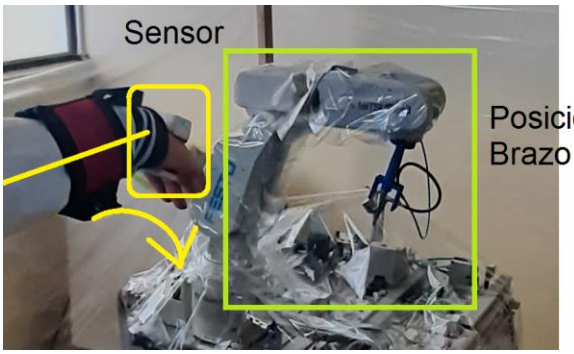
| **Movimiento** | Posición |
| --- | --- |
| **Pronación** |  |
| **Superación** |  <br>  |

### 5.1.3. *Muñeca*

Para el movimiento de la muñeca, se requieren los valores de las variables angulo3_roll, angulo3_yaw y angulo3_pitch mencionadas en la sección 4.4.5, del sensor Tactigon V1.0 (T-Skin) ubicado en la muñeca. La posición final de cada movimiento se puede observar en la Tabla 23.

**Tabla 23**

*Biomecánica de la muñeca*

| **Movimiento** | **Posición** |
|---|---|
| **Extensión** |  |
| **Flexión** |  |

## 5.2. Interpretación de Movimientos Humanos en Coordenadas XYZ

Gracias a la programación que se realizó en la sección 4.4.4 para obtener las coordenadas rectangulares XYZ del brazo robótico, a partir de los ángulos Roll, Pitch y Yaw que proporcionan los sensores Tactigon V1.0 ubicados en el hombro y codo, se logró la interpretación de los movimientos en los ejes $X$, $Y$ y $Z$ como se puede visualizar en la Tabla 24. A su vez, la interpretación de los movimientos alrededor del eje $X$ y $Y$ por parte del sensor Tactigon V1.0 ubicado en la muñeca como se puede observar en la Tabla 24.

**Tabla 24**

*Interpretación de movimientos lineales humanos*

| Movimiento Lineal | Posición |
|---|---|
| **Eje X** |  |

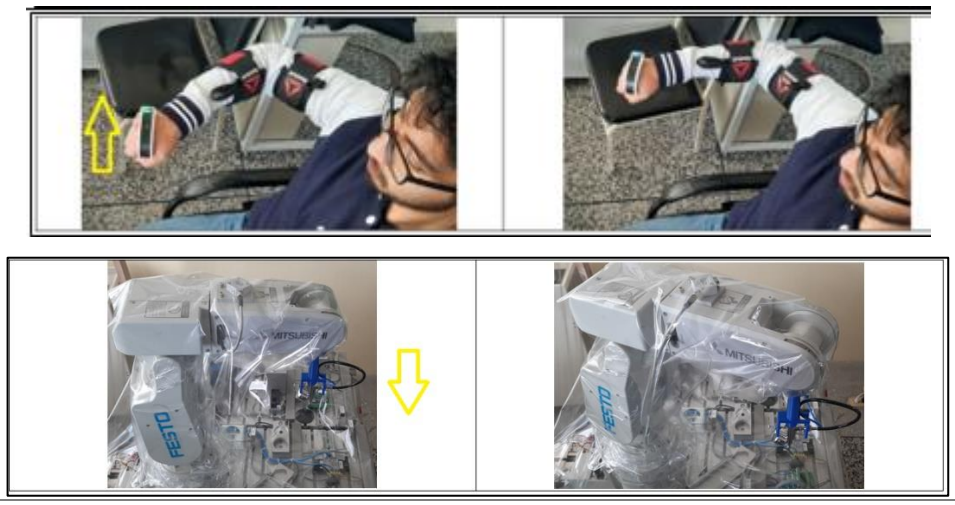
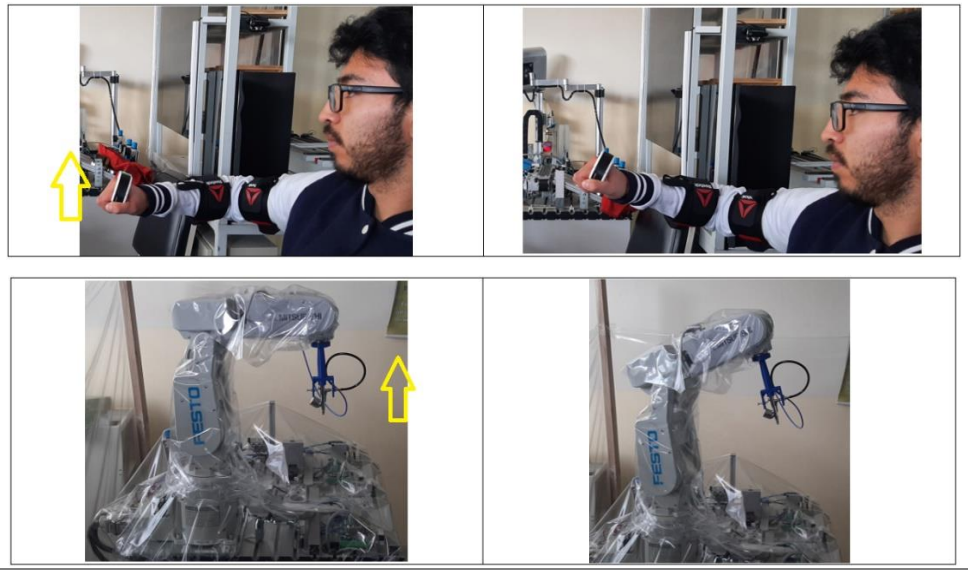
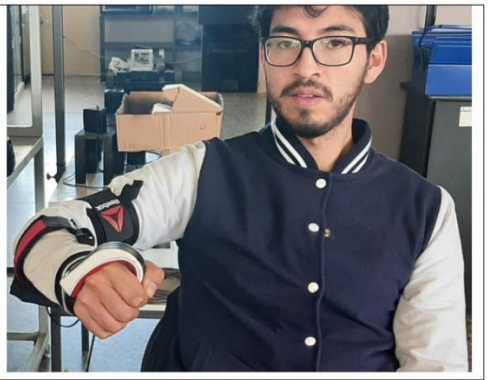
| Movimiento Lineal | Posición |
|---|---|
| **Eje Y** |  |
| **Eje Z** |  |

**Tabla 25**

*Interpretación de movimientos rotacionales humanos*

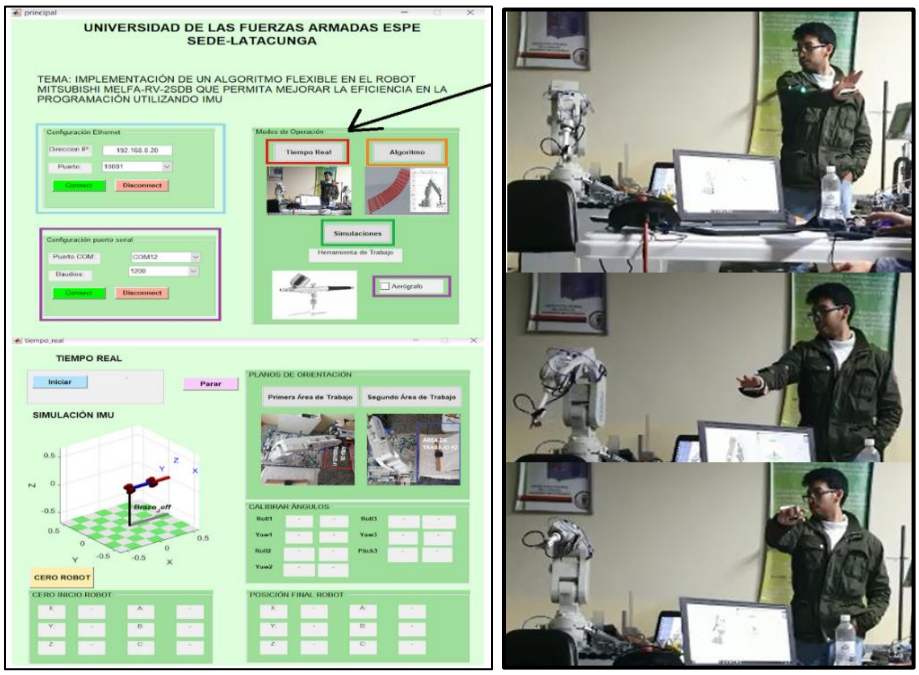
| Movimiento Rotacional | Posición | |
|---|---|---|
| **Alrededor del eje X** |  |  |
| |  |  |
| **Alrededor del eje Y** |  |  |
| |  |  |

## 5.3. Pruebas Modo Tiempo Real

Se realizaron las pruebas al algoritmo planteado en la sección 4.8.1 que permite controlar el brazo robótico de tal forma que, el brazo robótico repita los movimientos del brazo humano interactuando activamente con el operador, como se puede observar en la Figura 87.

**Figura 87**

*Modo Tiempo Real*

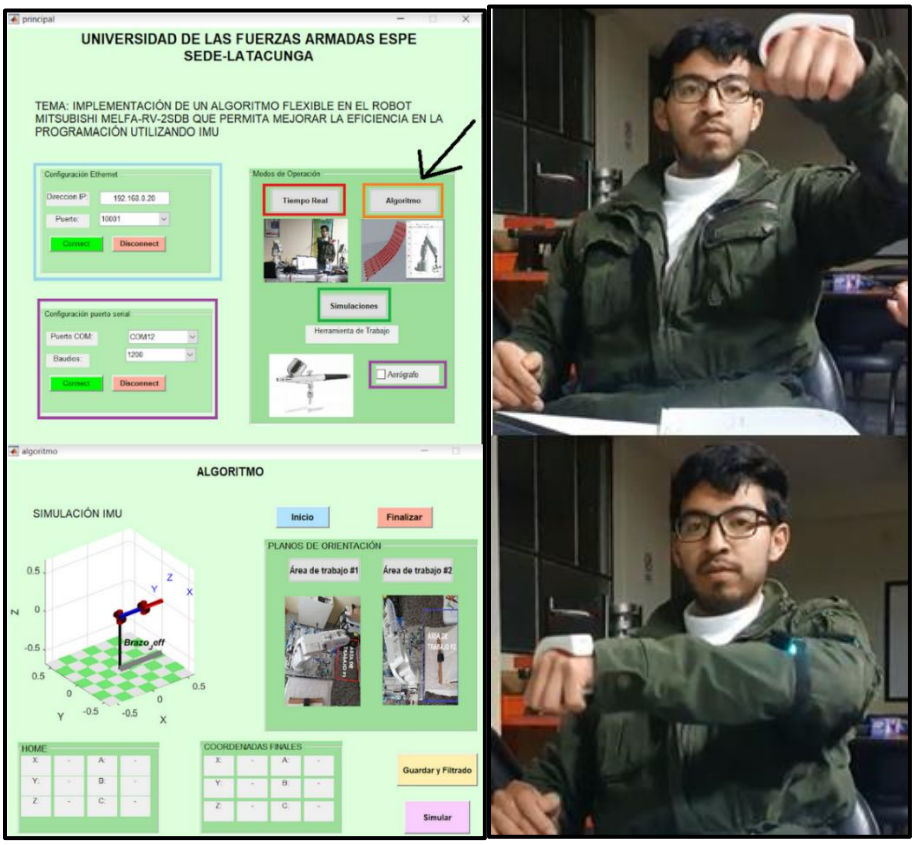



## 5.4. Pruebas Modo Algoritmo Flexible

Se realizaron las pruebas al algoritmo planteado en la sección 4.8.2 que permite capturar los movimientos generados por el brazo humano y guardarlos en un archivo TXT para su posterior simulación y envío de datos al controlador, como se puede observar en la Figura 88.

**Figura 88**

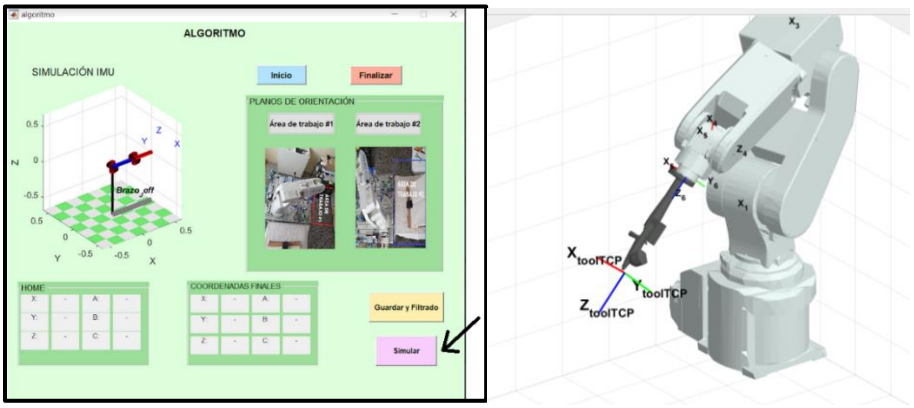
*Modo Algoritmo Flexible*




Luego de generar los movientes el operador se puede visualizar las trayectorias generadas en la simulación del brazo robótico, como se puede apreciar en la Figura 89.

**Figura 89**

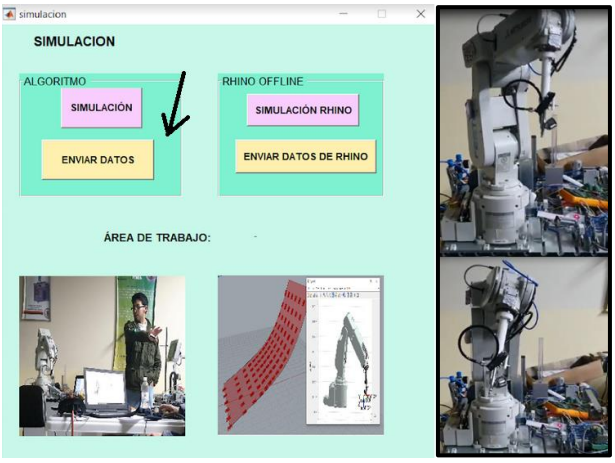
*Simulación de los movimientos realizador por el operador*




Para finalizar con el modo algoritmo flexible, se envían las coordenadas de las trayectorias generadas al controlador del brazo robótico, adicionalmente el operador puede correr el programa utilizando el controlador en Modo Automático, ya que el algoritmo guarda los movimientos realizados en un programa dentro de este.

**Figura 90**

*Envió de datos al controlador del brazo robótico*



## 5.5. Áreas de Trabajo

Para orientar el robot hacia el área en que el operador desea trabajar se configuró dos áreas de trabajo, que se pueden seleccionar en la interfaz, como se puede observar en la Figura 91. La Primera Área de Trabajo para que el brazo interactúe con los componentes de la estación, y la Segunda Área de Trabajo para que el brazo interactúe con una mesa anexa a la estación.

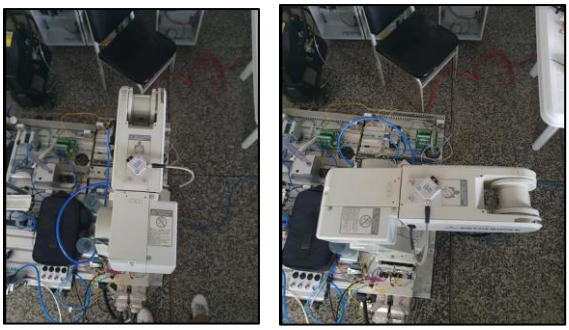
**Figura 91**

*Áreas de trabajo*


Teniendo así, la orientación deseada como se puede ver en la Figura 92.

**Figura 92**

*Áreas de trabajo en el brazo robótico*



## 5.6. Pruebas de Programación con el Algoritmo Flexible usando la Aplicación de Pintura

Para medir la eficiencia en la programación del brazo robótico se midió el valor del tiempo (minutos) en que le toma a un operador realizar diferentes trayectorias, programando mediante el Teach Pendant (Figura 93) y usando la configuración del algoritmo flexible. Tomando en cuenta, un factor adicional (Calificación entre 0 “No similar” y 1 “Similar”) para el algoritmo flexible que permita valorar el resultado de la trayectoria generada usando pintura para marcarla (Figura 94). Se realizaron trayectorias básicas y complejas para evidenciar en totalidad el alcance del algoritmo flexible.

**Figura 93**

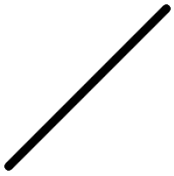
*Programación mediante Teach Pendant*

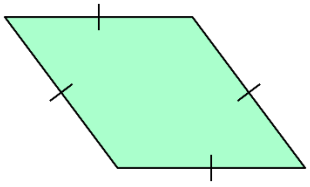
**Figura 94**

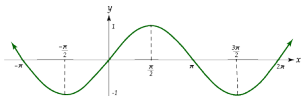
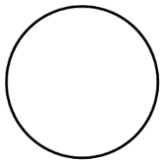
*Trayectorias generadas usando pintura*

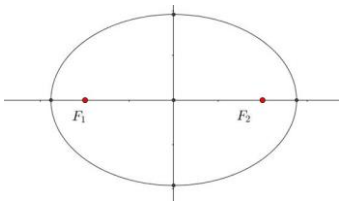
**Tabla 26**

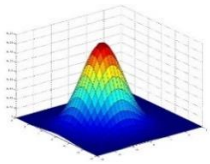
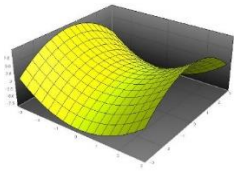
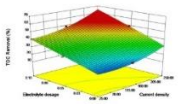
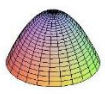
*Tiempo transcurrido para realizar las trayectorias*

| Trayectorias/Tiempo (minutos) | | Teach Pendant (minutos) | Algoritmo Flexible (minutos) | Calificación Algoritmo [0-1] |
|---|---|---|---|---|
| **Simples** | | | | |
|  | Ensayo 1 | 4 | 1.2 | 0.8 |
| | Ensayo 2 | 4 | 1.2 | 0.8 |
| | Ensayo 3 | 3 | 1 | 0.8 |
| Línea | Ensayo 4 | 3 | 0.8 | 0.9 |
| | Ensayo 5 | 3 | 0.8 | 0.9 |
| | Ensayo 6 | 4 | 1.2 | 0.8 |
| | Ensayo 7 | 4 | 1.2 | 0.8 |
| | Ensayo 8 | 3 | 1 | 0.8 |

| **Trayectorias/Tiempo (minutos)** | | | Teach Pendant (minutos) | Algoritmo Flexible (minutos) | Calificación Algoritmo [0-1] |
|---|---|---|---|---|---|
| | | Ensayo 9 | 3 | 0.8 | 0.9 |
| | | Ensayo 10 | 3 | 0.8 | 0.6 |
|  | Zigzag | Ensayo 1 | 10 | 1.5 | 0.6 |
| | | Ensayo 2 | 8 | 1.5 | 0.7 |
| | | Ensayo 3 | 7 | 1.5 | 0.7 |
| | | Ensayo 4 | 7 | 1.5 | 0.8 |
| | | Ensayo 5 | 7 | 1.5 | 0.8 |
| | | Ensayo 6 | 10 | 1.5 | 0.6 |
| | | Ensayo 7 | 8 | 1.5 | 0.7 |
| | | Ensayo 8 | 7 | 1.5 | 0.7 |
| | | Ensayo 9 | 7 | 1.5 | 0.8 |
| | | Ensayo 10 | 7 | 1.5 | 0.8 |
|  | Rombo | Ensayo 1 | 6 | 1.2 | 0.6 |
| | | Ensayo 2 | 6 | 1.1 | 0.6 |
| | | Ensayo 3 | 5.5 | 1 | 0.6 |
| | | Ensayo 4 | 5.5 | 1 | 0.7 |
| | | Ensayo 5 | 5 | 1 | 0.8 |
| | | Ensayo 6 | 6 | 1.2 | 0.6 |
| | | Ensayo 7 | 6 | 1.1 | 0.6 |
| | | Ensayo 8 | 5.5 | 1 | 0.6 |
| | | Ensayo 9 | 5.5 | 1 | 0.7 |
| | | Ensayo 10 | 5 | 1 | 0.8 |
|  | Número “7” | Ensayo 1 | 4.5 | 1.1 | 0.7 |
| | | Ensayo 2 | 4.5 | 1.1 | 0.7 |
| | | Ensayo 3 | 4 | 1 | 0.8 |
| | | Ensayo 4 | 4 | 1 | 0.8 |
| | | Ensayo 5 | 3.5 | 1 | 0.8 |

| Trayectorias/Tiempo (minutos) | | | Teach Pendant (minutos) | Algoritmo Flexible (minutos) | Calificación Algoritmo [0-1] |
|---|---|---|---|---|---|
| | | Ensayo 6 | 4.5 | 1.1 | 0.7 |
| | | Ensayo 7 | 4.5 | 1.1 | 0.7 |
| | | Ensayo 8 | 4 | 1 | 0.8 |
| | | Ensayo 9 | 4 | 1 | 0.8 |
| | | Ensayo 10 | 3.5 | 1 | 0.8 |
| **Circulares** | | | | | |
|  | Función “Seno” | Ensayo 1 | 7 | 1.5 | 0.6 |
| | | Ensayo 2 | 7 | 1.5 | 0.6 |
| | | Ensayo 3 | 6.8 | 1 | 0.7 |
| | | Ensayo 4 | 6.5 | 1 | 0.8 |
| | | Ensayo 5 | 6.5 | 1 | 0.8 |
| | | Ensayo 6 | 7 | 1.5 | 0.6 |
| | | Ensayo 7 | 7 | 1.5 | 0.6 |
| | | Ensayo 8 | 6.8 | 1 | 0.7 |
| | | Ensayo 9 | 6.5 | 1 | 0.8 |
| | | Ensayo 10 | 6.5 | 1 | 0.8 |
|  | Círculo | Ensayo 1 | 5 | 1 | 0.5 |
| | | Ensayo 2 | 5 | 1 | 0.5 |
| | | Ensayo 3 | 4.5 | 1 | 0.7 |
| | | Ensayo 4 | 4.5 | 1 | 0.7 |
| | | Ensayo 5 | 4.5 | 1 | 0.7 |
| | | Ensayo 6 | 5 | 1 | 0.5 |
| | | Ensayo 7 | 5 | 1 | 0.5 |
| | | Ensayo 8 | 4.5 | 1 | 0.7 |
| | | Ensayo 9 | 4.5 | 1 | 0.7 |
| | | Ensayo 10 | 4.5 | 1 | 0.7 |

| Trayectorias/Tiempo (minutos) | | | Teach Pendant (minutos) | Algoritmo Flexible (minutos) | Calificación Algoritmo [0-1] |
|---|---|---|---|---|---|
|  | Elipse | Ensayo 1 | 12 | 1 | 0.6 |
| | | Ensayo 2 | 11 | 1 | 0.7 |
| | | Ensayo 3 | 10 | 1 | 0.7 |
| | | Ensayo 4 | 10 | 1 | 0.6 |
| | | Ensayo 5 | 10 | 1 | 0.7 |
| | | Ensayo 6 | 12 | 1 | 0.6 |
| | | Ensayo 7 | 11 | 1 | 0.7 |
| | | Ensayo 8 | 10 | 1 | 0.7 |
| | | Ensayo 9 | 10 | 1 | 0.6 |
| | | Ensayo 10 | 10 | 1 | 0.7 |
|  | Número “3” | Ensayo 1 | 10 | 1 | 0.6 |
| | | Ensayo 2 | 9 | 1 | 0.7 |
| | | Ensayo 3 | 9 | 1 | 0.7 |
| | | Ensayo 4 | 8 | 1 | 0.8 |
| | | Ensayo 5 | 8 | 1 | 0.8 |
| | | Ensayo 6 | 10 | 1 | 0.6 |
| | | Ensayo 7 | 9 | 1 | 0.7 |
| | | Ensayo 8 | 9 | 1 | 0.7 |
| | | Ensayo 9 | 8 | 1 | 0.8 |
| | | Ensayo 10 | 8 | 1 | 0.8 |
|  | Signo de Interrogación | Ensayo 1 | 10 | 1.1 | 0.4 |
| | | Ensayo 2 | 10 | 1.1 | 0.5 |
| | | Ensayo 3 | 9 | 1.1 | 0.7 |
| | | Ensayo 4 | 9 | 1.1 | 0.7 |
| | | Ensayo 5 | 9 | 1.1 | 0.7 |
| | | Ensayo 6 | 10 | 1.1 | 0.4 |
| | | Ensayo 7 | 10 | 1.1 | 0.5 |

| Trayectorias/Tiempo (minutos) | | Teach Pendant (minutos) | Algoritmo Flexible (minutos) | Calificación Algoritmo [0-1] |
|---|---|---|---|---|
| | Ensayo 8 | 9 | 1.1 | 0.7 |
| | Ensayo 9 | 9 | 1.1 | 0.7 |
| | Ensayo 10 | 9 | 1.1 | 0.7 |
| **Complejas** | | | | |
|  | Campana de Gauss | - | 2 | 0.7 |
|  | Espiral | - | 3 | 0.8 |
|  | Paraboloide hiperbólico | - | 2.5 | 0.9 |
|  | Plano inclinado | - | 2.5 | 0.9 |
|  | Paraboloide invertido | - | 3 | 0.8 |

Debido a la dificultad de replicar las trayectorias complejas mediante la programación por Teach Pendant ya que poseen un alto número de puntos y comandos, se asume que todos los ensayos realizados para estas trayectorias son eficientes.

### 5.6.1. *Validación de la hipótesis*

Se calculó la eficiencia en cada una de las trayectorias generadas comparando el tiempo de programación mediante Teach Pendant (la programación que normalmente se utiliza para el brazo robótico) y la programación mediante el algoritmo flexible, tomando en cuenta el valor de la calificación de la trayectoria generada. Si el valor de la eficiencia calculada sobrepasa el umbral de eficiencia del 100%, se considera como un resultado positivo y se lo toma para llenar la Tabla 36, que se utilizó para una prueba de Chi-Cuadrado con el fin de validar la hipótesis.

$$Eficiencia = \frac{Tiempo\ Normal\ (Teach\ Pendant)}{\frac{Tiempo\ de\ trayectoria\ generada\ (Algoritmo\ Flexible)}{Calificación}} x100\%$$

$$Umbral_{eficiencia} = 100\%$$

Se tiene que: $Si\ Eficiencia\ > Umbral_{eficiencia}\ y\ \ Calificación > 0.6$

Solo si cumple ambas condiciones entonces el Resultado es Positivo ✓ , caso contrario Resultado Negativo ✗.

**Trayectorias Simples**

**Tabla 27**

*Líneas*

| Ensayo | Eficiencia Calculada % | $Eficiencia\ Calculada > Umbral_{eficiencia}$ | $Calificación \geq 0.6$ | Resultado |
|---|---|---|---|---|
| **1** | 266.67 | SI | SI | ✓ |
| **2** | 266.67 | SI | SI | ✓ |
| **3** | 240 | SI | SI | ✓ |
| **4** | 337 | SI | SI | ✓ |
| **5** | 337 | SI | SI | ✓ |
| **6** | 266.67 | SI | SI | ✓ |
| **7** | 266.67 | SI | SI | ✓ |
| **8** | 240 | SI | SI | ✓ |
| **9** | 337 | SI | SI | ✓ |
| **10** | 337 | SI | NO | ✗ |
| | | | Total | 9 |

**Tabla 28**

*Zigzag*

| Ensayo | Eficiencia Calculada % | $Eficiencia\ Calculada > Umbral_{eficiencia}$ | $Calificación > 0.6$ | Resultado |
|---|---|---|---|---|
| **1** | 400 | SI | NO | ✗ |
| **2** | 327 | SI | SI | ✓ |
| **3** | 327 | SI | SI | ✓ |
| **4** | 373 | SI | SI | ✓ |
| **5** | 373 | SI | SI | ✓ |
| **6** | 400 | SI | NO | ✗ |
| **7** | 327 | SI | SI | ✓ |
| **8** | 327 | SI | SI | ✓ |
| **9** | 373 | SI | SI | ✓ |
| **10** | 373 | SI | SI | ✓ |
| | | | Total | 8 |

**Tabla 29**

*Rombo*

| Ensayo | Eficiencia Calculada % | $Eficiencia\ Calculada > Umbral_{eficiencia}$ | $Calificación > 0.6$ | Resultado |
|---|---|---|---|---|
| **1** | 300 | SI | NO | ✗ |
| **2** | 327 | SI | NO | ✗ |
| **4** | 384 | SI | SI | ✓ |
| **4** | 384 | SI | SI | ✓ |
| **5** | 400 | SI | SI | ✓ |
| **4** | 384 | SI | SI | ✓ |
| **5** | 400 | SI | SI | ✓ |
| **4** | 384 | SI | SI | ✓ |
| **5** | 400 | SI | SI | ✓ |
| **4** | 384 | SI | SI | ✓ |
| | | | Total | 8 |

**Tabla 30**

*Número "7"*

| Ensayo | Eficiencia Calculada % | $Eficiencia\ Calculada > Umbral_{eficiencia}$ | $Calificación > 0.6$ | Resultado |
|---|---|---|---|---|
| **1** | 287 | SI | SI | ✓ |
| **2** | 287 | SI | SI | ✓ |
| **3** | 400 | SI | SI | ✓ |
| **4** | 400 | SI | SI | ✓ |
| **5** | 280 | SI | SI | ✓ |
| **6** | 287 | SI | SI | ✓ |
| **7** | 287 | SI | SI | ✓ |
| **8** | 400 | SI | SI | ✓ |
| **9** | 400 | SI | SI | ✓ |
| **10** | 280 | SI | SI | ✓ |
| | | | Total | 10 |

**Trayectorias Circulares**

**Tabla 31**

*Función "Seno"*

| Ensayo | Eficiencia Calculada % | $Eficiencia\ Calculada > Umbral_{eficiencia}$ | $Calificación > 0.6$ | Resultado |
|---|---|---|---|---|
| **1** | 280 | SI | NO | ✗ |
| **2** | 300 | SI | NO | ✗ |
| **3** | 475 | SI | SI | ✓ |
| **4** | 520 | SI | SI | ✓ |
| **5** | 520 | SI | SI | ✓ |
| **6** | 280 | SI | NO | ✗ |
| **7** | 300 | SI | NO | ✗ |
| **8** | 475 | SI | SI | ✓ |
| **9** | 520 | SI | SI | ✓ |
| **10** | 520 | SI | SI | ✓ |
| | | | Total | 6 |

**Tabla 32**

*Círculo*

| Ensayo | Eficiencia Calculada % | $Eficiencia\ Calculada > Umbral_{eficiencia}$ | $Calificación > 0.6$ | Resultado |
|---|---|---|---|---|
| **1** | 250 | SI | NO | ✗ |
| **2** | 250 | SI | NO | ✗ |
| **3** | 315 | SI | SI | ✓ |
| **4** | 315 | SI | SI | ✓ |
| **5** | 315 | SI | SI | ✓ |
| **6** | 250 | SI | NO | ✗ |
| **7** | 250 | SI | NO | ✗ |
| **8** | 315 | SI | SI | ✓ |
| **9** | 315 | SI | SI | ✓ |
| **10** | 315 | SI | SI | ✓ |
| | | | Total | 6 |

**Tabla 33**

*Elipse*

| Ensayo | Eficiencia Calculada % | $Eficiencia\ Calculada > Umbral_{eficiencia}$ | $Calificación > 0.6$ | Resultado |
|---|---|---|---|---|
| **1** | 718 | SI | NO | ✗ |
| **2** | 769 | SI | SI | ✓ |
| **3** | 700 | SI | SI | ✓ |
| **4** | 598 | SI | NO | ✗ |
| **5** | 700 | SI | SI | ✓ |
| **6** | 718 | SI | NO | ✗ |
| **7** | 769 | SI | SI | ✓ |
| **8** | 700 | SI | SI | ✓ |
| **9** | 598 | SI | NO | ✗ |
| **10** | 700 | SI | SI | ✓ |
| | | | Total | 6 |

**Tabla 34**

*Número "3"*

| Ensayo | Eficiencia Calculada % | $Eficiencia\ Calculada > Umbral_{eficiencia}$ | $Calificación > 0.6$ | Resultado |
|---|---|---|---|---|
| **1** | 700 | SI | NO | ✗ |
| **2** | 700 | SI | SI | ✓ |
| **3** | 680 | SI | SI | ✓ |
| **4** | 640 | SI | SI | ✓ |
| **5** | 640 | SI | SI | ✓ |
| **6** | 700 | SI | NO | ✗ |
| **7** | 700 | SI | SI | ✓ |
| **8** | 680 | SI | SI | ✓ |
| **9** | 640 | SI | SI | ✓ |
| **10** | 640 | SI | SI | ✓ |
| | | | Total | 8 |

**Tabla 35**

*Signo de Interrogación*

| Ensayo | Eficiencia Calculada % | $Eficiencia\ Calculada > Umbral_{eficiencia}$ | $Calificación > 0.6$ | Resultado |
|---|---|---|---|---|
| **1** | 363 | SI | NO | ✗ |
| **2** | 454 | SI | NO | ✗ |
| **3** | 520 | SI | SI | ✓ |
| **4** | 543 | SI | SI | ✓ |
| **5** | 543 | SI | SI | ✓ |
| **6** | 363 | SI | NO | ✗ |
| **7** | 454 | SI | NO | ✗ |
| **8** | 520 | SI | SI | ✓ |
| **9** | 543 | SI | SI | ✓ |
| **10** | 543 | SI | SI | ✓ |
| | | | Total | 6 |

**Tabla 36**

*Resultados obtenidos de las pruebas*

| Resultado/trayectoria | Simples | | Circulares | | Total |
|---|---|---|---|---|---|
|  | Línea | 9 | Función “Seno” | 8 | **13** |
| | Zigzag | 8 | Círculo | 6 | **11** |
| | Rombo | 8 | Elipse | 6 | **11** |
| | Número “7” | 10 | Número “3” | 8 | **14** |
| | **Total** | **35** | Signo de Interrogación | 6 | **3** |
| | | | **Total** | **34** | **69** |
|  | Línea | 1 | Función “Seno” | 2 | **2** |
| | Zigzag | 2 | Círculo | 4 | **4** |
| | Rombo | 2 | Elipse | 4 | **4** |
| | Número “7” | 0 | Número “3” | 2 | **1** |
| | **Total** | **5** | Signo de Interrogación | 4 | **2** |
| | | | **Total** | **16** | **21** |
| **Total** | **40** | | **50** | | **90** |

Hipótesis Nula (Ho): El algoritmo flexible no mejora la eficiencia en la programación.

Hipótesis alternativa (Hi): El algoritmo flexible mejora la eficiencia en la programación.

$$ft\ (frecuencia\ teórica) \rightarrow \frac{(total\ de\ la\ fila\ i)x(total\ de\ la\ columna\ j)}{número\ total\ de\ datos}$$

$i \rightarrow 1-2 \qquad j \rightarrow 1-2$

$$35 \rightarrow \frac{40 * 69}{90} = 30.67 \qquad 34 \rightarrow \frac{50 * 69}{90} = 38.33$$

$$5 \rightarrow \frac{40 * 21}{90} = 9.33 \qquad 16 \rightarrow \frac{50 * 21}{90} = 11.67$$

Chi-Cuadrado Calculado ($x^2_{calc}$): $\sum \frac{(f-ft)^2}{ft}$

$f \rightarrow frecuencia\ observada\ (sumatoria\ de\ los\ datos\ en\ la\ tabla)$

$ft \rightarrow frecuencia\ teórica$

($x^2_{calc}$): $$\sum \frac{(f-ft)^2}{ft} = \frac{(35-30.67)^2}{30.67} + \frac{(34-38.33)^2}{38.33} + \frac{(5-9.33)^2}{9.33} + \frac{(16-11.67)^2}{11.67}$$

$$= 0.61 + 0.49 + 2 + 1.61$$

$$x^2_{calc} = 4.71$$

Se trabajó con un nivel de significancia del $\propto = 5\%$. Se calcula entonces la proporcionalidad $p$ con la siguiente fórmula:

$$p = 1 - nivel\ de\ significancia = 0.95$$

Cálculo del Grado de libertad ($v$):

$$v = (n°\ de\ filas - 1) * (n°\ de\ columnas - 1)$$

$$n°\ de\ filas = 2\ (Resultado\ positivo/negativo)$$

$$n°\ de\ columnas = 2\ (Trayectorias\ Simples/Compleja)$$

$$v = (2 - 1) * (2 - 1) = 1$$

Teniendo 1 grado de libertad y para un valor de proporcionalidad del 0.95 se obtiene el calor del Chi Cuadrado en su tabla de distribución:

$$x^2_{tabla} = 3.841$$

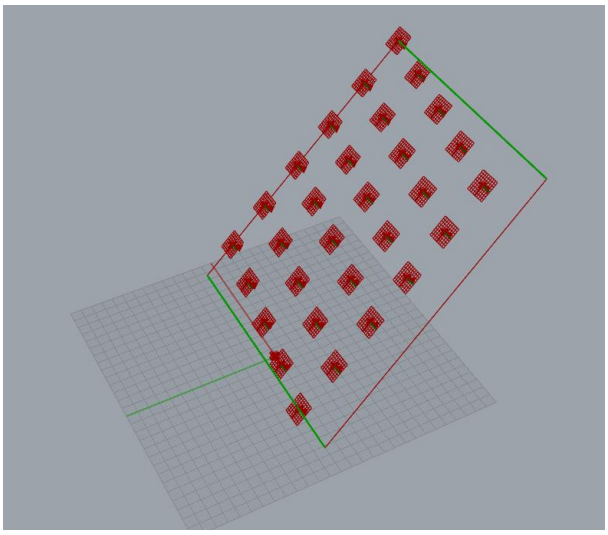
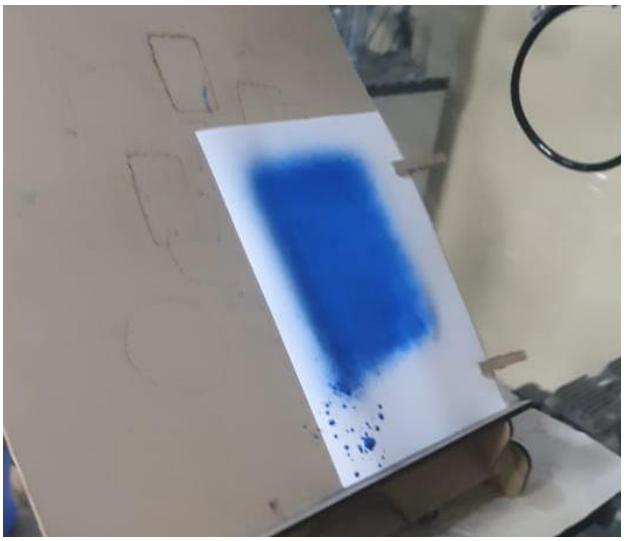
Como el Chi-Cuadrado calculado es mayor que el Chi-cuadrado obtenido en la tabla, entonces se procede a descartar la Hipótesis nula. Por tanto, se procede a validar la hipótesis alternativa que sostiene que el algoritmo flexible mejora la eficiencia en la programación.

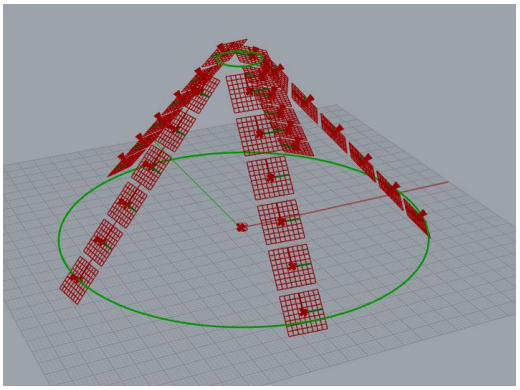
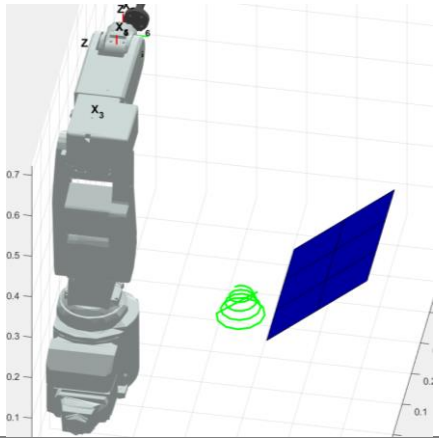
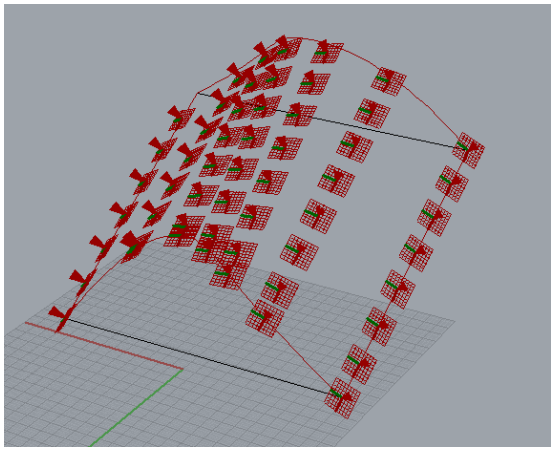
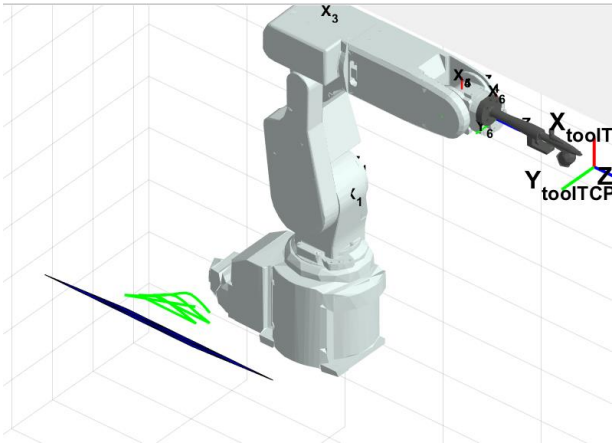
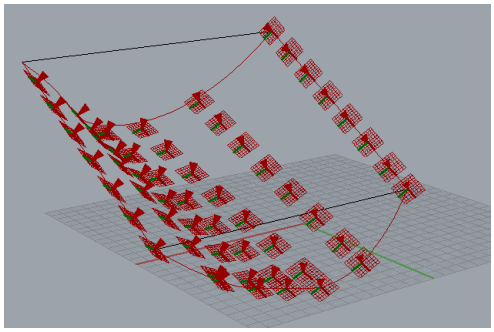
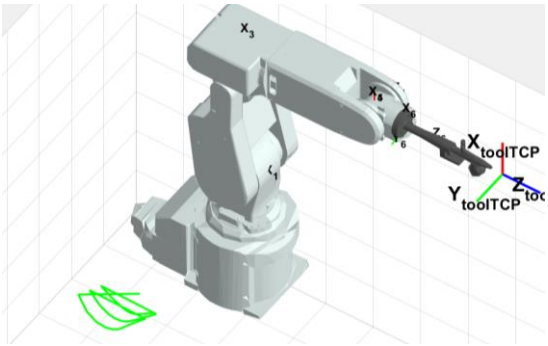
## 5.7. Pruebas del Algoritmo Flexible Mediante Programación Offline con Rhinoceros

Gracias al algoritmo de la sección 4.8.3 que explica el uso de Rhinoceros como un método de programación offline es posible realizar trayectorias sencillas y complicadas con una gran precisión, se toma en cuenta algunos ensayos de las trayectorias complejas como ejemplos para la programación offline como se muestra en la Tabla 37.

**Tabla 37**

*Pruebas Offline con Rhinoceros*

| Programación | Resultado |
|---|---|
| **Plano inclinado** | |
|  |  |

| **Programación** | Resultado |
|---|---|
| **Torbellino** | |
|  |  |
| **Campana de Gauss** | |
|  |  |
| **Vórtice** | |
|  |  |

# CAPÍTULO VI

# CONCLUSIONES Y RECOMENDACIONES

## 6.1. Conclusiones

- Para el diseño del algoritmo flexible se partió de la recopilación de conceptos que intervienen dentro de la robótica industrial como la cinemática directa e inversa que son datos muy importantes junto a los parámetros Denavit-Hartenberg los cuales permiten el cálculo de posiciones del efector final en cualquier momento, además se realizó una búsqueda de los distintos sensores que manejan unidades de medida inercial encontrando que existen varios modelos como mecánicos, ópticos y tipo MEMS, se determinó que el modelo más eficiente para la aplicación es el MEMS debido a su tamaño, posee la capacidad procesar datos, comunicar y actuar sobre su entorno. Por último, se reúne información acerca de los lenguajes de programación aplicados en el brazo robótico encontrando que poseen una categoría enfocada en la forma de programación y se la puede dividir en programación textual y offline.
- Al momento de adquirir y procesar las señales de las unidades de medida inercial se utilizó un microprocesador integrado con bluetooth, batería, giroscopio, acelerómetro y sensor magnético, estos últimos tres datos son utilizados como datos de entrada para un algoritmo propio del microprocesador permitiendo obtención de los movimientos angulares como el roll, pitch y yaw de la superficie en la que se encuentra sujeta en este caso una extremidad del tren superior.

- Para establecer un enlace entre los componentes del sistema, se diseñó una shield bluetooth capaz de conectar todo el sistema electrónico permitiendo enviar los datos de los sensores hacia el ordenador mediante comunicación serial, estos datos son procesados, transformados y enviados al controlador del brazo robótico través de comunicación ethernet. Una de la funciones del algoritmo flexible es el de convertir los datos de los sensores a un lenguaje de programación del brazo robótico además permite cargar posiciones y movimientos al controlador en modo control externo gracias a la gran variedad de comandos como *"1;1;SRVON"* que permite encender los servomotores, *"1;9;EXECMOV P1"* mueve el efector final hacia la posición *P1* y *"1;9;VAL=P2=(x,y,z,a,b,c)(7,0)"* permite almacenar los valores *x,y,z,a,b,c* en la variable posición *P2*.
- Al momento de diseñar el algoritmo flexible se utilizó un lenguaje estructurado debido a la facilidad que se tiene para añadir funcionalidades permitiendo realizar acciones más complejas. El algoritmo flexible está compuesto por pequeños programas que realizan tareas distintas como es el caso del algoritmo que permite transformar los ángulos de los sensores RPY a posiciones XYZ, el algoritmo para enviar las posiciones hacia el controlador o el algoritmo para la programación offline cada uno de estos algoritmos posee un proceso detallado y se encuentran enlazados mediante el HMI implementado en el ordenador.
- Se implementó tres modos de trabajo. El primero es el modo tiempo real, el cual permite al operador comprender la relación que existe entre los movimientos del brazo humano y las posiciones enviadas al brazo robótico en tiempo real, este modo no se guarda las

posiciones dentro del controlador. El segundo es el modo algoritmo flexible que programa al brazo robótico con los movimientos angulares del brazo humano, permite cambiar el nombre del archivo a guardar en el controlador. Por último, el modo offline que utiliza Rhinoceros como software intermediario para calcular las posiciones y movimientos complejos que se requiera realizar.

- Se realizaron pruebas con el algoritmo flexible por medio de trayectorias simples, circulares y complejas. Para validar cada uno de los ensayos que se realizó, se impuso un umbral de eficiencia, con el fin de medir si el resultado posee una alta eficiencia en tiempo de programación y además la trayectoria generada posee similitud con la maniobra que ejecutó el operador. Las trayectorias simples no representaron problema alguno, teniendo 35 resultados positivos de 40 ensayos; las trayectorias circulares tuvieron 34 resultados positivos de 50 ensayos. El algoritmo flexible permite además programar en cuestión de minutos (programación offline) trayectorias complejas utilizando Software como Rhinoceros, lo que resulta imposible de realizar mediante la programación del brazo robótico normal (Teach Pendant), estos ensayos fueron excluidos del análisis para la validación de la hipótesis planteada. Gracias a los resultados obtenidos se logró relacionar las trayectorias generadas mediante el algoritmo flexible y la eficiencia en la programación, mediante la prueba de Chi-Cuadrado ($x^2_{calc} = 4.71 > x^2_{tabla} = 3.84$) en la que se procedió a tomar la hipótesis alternativa, la cual sostiene que el algoritmo flexible mejora la eficiencia en la programación con un nivel de significancia del 5%.
- Durante el movimiento del robot en las aplicaciones se presentaron ciertas singularidades. Las singularidades son aquellos puntos en los que el robot no puede alcanzar las posiciones

y rotaciones calculadas por el algoritmo flexible y se presentan debido a que los ángulos $J_1, J_2, J_3, J_4, J_5\ y\ J_6$ (resultado de las soluciones de la cinemática inversa) no se encuentran dentro del rango de operación del brazo robótico. Las posiciones y rotaciones calculadas se encuentran estrechamente relacionadas con los movimientos de los sensores que son acoplados al brazo humano y son calculadas de la siguiente manera: las posiciones $x, y, z$ dependen de los sensores ubicados en el antebrazo y brazo, mientras que las rotaciones $A, B, C$ dependen únicamente del sensor ubicado en la mano. Se identificó que el sensor ubicado en la mano es el más propenso al envío de singularidades ya que se puede acceder a cualquier punto $x, y, z$, pero no con todas las rotaciones $A, B, C$, un claro ejemplo sucede cuando el brazo robótico se encuentra con todas sus junturas formando un ángulo de 0 grados, solo puede acceder al punto más alto $x = 0, y = 0\ y\ z = máx$ cuando $A = 0°\ y\ B = 0°$ (el valor del ángulo C es indiferente), si se añadiese algún desplazamiento angular extra en $A\ o\ B$ ya no se encontraría una solución para $x = 0, y = 0\ y\ z = máx$ lo que caería en una singularidad.

### 6.2. Recomendaciones

- Una vez finalizado con el diseño del sistema electrónico del proyecto se encontró que, para mejorar la interacción entre el operador y los sensores, se debería implementar un módulo bluetooth en la shield para obtener una conexión inalámbrica y no depender de la comunicación serial.
- Para evitar la desconexión entre los sensores y la shield es necesario que el operador se encuentra a distancia máxima de 2 metros entre los dispositivos.

- Para que el operador obtenga buenos resultados en la aplicación de pintura se requiere utilizar en primera instancia el modo tiempo real para conocer los distintos desplazamientos del brazo robótico.

## REFERENCIAS BIBLIOGRÁFICAS

Anzora, L. I. (2010 ). *Diseño de Identidad Visual para COSABILA, de CORDES (Asociación para la cooperación y el desarrollo comunal de El Salvador).* Antiguo Cuscatlán: Universidad Dr. José Matías Delgado .

ARDUINO. (2018). *What is Arduino?* https://www.arduino.cc/en/Guide/Introduction

arduino.cl. (2019). *Arduino Due*. Recuperado el 8 de Enero de 2020, de https://arduino.cl/producto/arduino-due/

Barrientos, A., Peñín, L. F., Balaguer, C., & Aracil, R. (2007). *Fundamentos de Robótica.* Madrid: McGraw-Hill.

Barrientos, A., Peñin, L. F., Balayer, C., & Aracil, R. (2007). *Fundamentos de Robótica* (2da ed.). McGraw-Hill. https://eltrasteroloco.files.wordpress.com/2017/03/267380685-fundamentos-de-robotica.pdf

Corke, P. (2017). *Robotics Toolbox*. Peter Corke. https://petercorke.com/toolboxes/robotics-toolbox/

Corsini, M., Cignoni, P., & Scopigno, R. (2012). Efficient and Flexible Sampling with Blue Noise Properties of Triangular Meshes. *IEEE Transactions on Visualization and Computer Graphics*, *18*(6), 914-924. https://doi.org/10.1109/TVCG.2012.34

EasyEDA. (s. f.). *Software de diseño de circuitos—EasyEDA*. Recuperado 1 de octubre de 2020, de https://easyeda.com/news/Circuit_Design_Software-LdNmkigec

FESTO. (s. f.). *Válvula universal CPE*. Recuperado 1 de octubre de 2020, de https://www.festo.com/cat/es-ar_ar/products

Giraldo, O. C. (3 de septiembre de 2009). *Tienda de Fisioterapia.* Obtenido de http://www.efisioterapia.net/tienda

Gómez de Gabriel, J. M., Ollero Baturone, A., & García Cerezo, A. J. (2006). *Teleoperación y Telerrobótica.* Madrid: Pearson Educación S.A.

Gómez Echeverry, L. L., Jaramillo Henao, A. M., Ruiz Molina, M. A., Velásquez Restrepo, S. M., Páramo Velásquez, C. A., Silva Bolívar, G. J., Gómez Echeverry, L. L., Jaramillo Henao, A. M., Ruiz Molina, M. A., Velásquez Restrepo, S. M., Páramo Velásquez, C. A., & Silva Bolívar, G. J. (2018). Human motion capture and analysis systems: A systematic review. *Prospectiva*, *16*(2), 24-34. https://doi.org/10.15665/rp.v16i2.1587

Grupo SIRP. (1 de Octubre de 2010). *Tipos de movimiento y grados de libertad.* Recuperado el 2 de Enero de 2020, de https://es.slideshare.net/EducaredColombia/tipos-de-movimiento-y-grados-de-libertad

Hamlet Betancourt, L., & Díaz Sánchez, M. E. (2007). Análisis longitudinal de las dimensiones corporales en adolescentes de la Escuela Nacional de Ballet de Cuba. *APUNTS*, 127-137.

Hernandez. (2013). *ROBÓTICA (1770) » ARTE (A Robotics Toolbox for Education)*. https://umh1770.edu.umh.es/2013/07/23/arte-a-robotics-toolbox-for-education/

Impresoras3D. (12 de Enero de 2017). *ABS y PLA*. Recuperado el 2 de Diciembre de 2019, de https://www.impresoras3d.com/abs-y-pla-diferencias-ventajas-y-desventajas/

Kapandji, A. I. (2006). *Fisiologia Articular Tomo I: Vol. 6ta ed*. Medica Panamericana Sa de. https://www.academia.edu/35112135/Kapandji_Fisiologia_Articular_Tomo_I

Kilmarx, J., Abiri, R., Borhani, S., Jiang, Y., & Xiaopeng, Z. (3 de Abril de 2017). *Springer Link.* Obtenido de Sequence-based manipulation of robotic arm control in brain machine interface: 2018

López. (2016). *Desarrollo de un dispositivo para la medicion y el analisis cinematico de movimiento motriz* [Instituto Politécnico Nacional]. https://tesis.ipn.mx/bitstream/handle/123456789/20771/Desarrollo%20de%20un%20dispositivo%20para%20la%20medicion%20y%20el%20analisis%20cinematico%281%29.pdf?sequence=1&isAllowed=y

Makeitfrom. (12 de Marzo de 2015). *Materiales de impresión 3D PLA.* Recuperado el 3 de Septiembre de 2019, de http://hxx.es/2015/03/12/materiales-de-impresion-3d-i-pla-acido-polilactico/

Martínez, F. (2015). *Sistemas de Coordenadas*. Recuperado el 8 de Enero de 2020, de http://ri.uaemex.mx/bitstream/handle/20.500.11799/63801/secme-?sequence=1

Mathworks. (s. f.). *What is MATLAB?* Recuperado 1 de octubre de 2020, de https://www.mathworks.com/discovery/what-is-matlab.html

Mitsubishi. (2010). *Mitsubishi Industrial Robot SD Series RV-2SD/2SDB Standard Specifications Manual*. https://robotics.ee.uwa.edu.au/courses/robotics/project/festo/MPS_TD_V2.4_EN/English/06_Robot/RV-2SDB/Mitsubishi%20manuals/bfp-a8790b.pdf

Mitsubishi Electric. (s. f.). *Mitsubishi Electric Factory Automation—Spain*. Recuperado 30 de septiembre de 2020, de https://es3a.mitsubishielectric.com/fa/es/service/download

MPS Estaciones. (s. f.). *Estación de Robot Calidad industrial*.

Mitzner, K. (2009). *Complete PCB design using OrCAD Capture and PCB editor.* Elsevier.

Mott, R. (2009). *Resistencia de materiales.* México: Pearson Education. Recuperado el 12 de 1 de 2020

Olmedo, L. (12 de Marzo de 2019). *Tipos de sensores en fotografía y sus características*. Recuperado el 6 de Enero de 2020, de https://blog.foto24.com/tipos-de-sensores-en-fotografia/

Porten. (Agosto de 2020). *Porten Tools*. Obtenido de https://www.portentools.com/aerografos.php

Rhinoceros. (s. f.). *Rhino 6 para Windows y Mac*. Recuperado 1 de octubre de 2020, de https://www.rhino3d.com/

Sánchez, M., Jiménez, S., Rodríguez, M., Bayarri, S., Monllau, F., Palou, R., & Villavicencio, M. (2007). Historia de la robótica: De Arquitas de Tarento al Robot da Vinci. (Parte II). *Actas Urológicas españolas,* 185-196.

SolidWorks. (s. f.). *Software de diseño CAD 3D | SOLIDWORKS*. Recuperado 1 de octubre de 2020, de https://www.solidworks.com/es/home-page-2021

Sturman, D. J., & Zeltzer, D. (1993). *Utility of Whole-Hand Input.* In proc SPIE .

Tactigon, T. (2019 de Septiembre de 2020). *The Tactigon Skin and The Tactigon ONE*. Obtenido de https://www.thetactigon.com/products/

WordPress. (2019). *WordPress Codex*. Recuperado el 13 de Julio de 2019, de https://codex.wordpress.org

Unzueta. (2014). *La captura de movimiento de bajo coste aplicada al rendimiento El caso REPLAY y los deportes tradicionales*. Vicomtech-IK4.

# ANEXOS