Universidad Nacional de Asunción

Facultad de Ingeniería

Robótica I

Proyecto Final de Robótica I: Brazo Robotico Soldador

Profesores:
Dr. Fernando Brunetti
Est. Magno Quintana

Alumnos:
Jose Ortiz
Carlos Gaona
Hans Mersch

Índice

1. Estado del Arte

La robótica es un campo bastamente explorado principalmente en países del primer mundo. El empleo de la misma en los campos de manufactura y fabricación nos permite obtener productos con mayor rapidez y confiabilidad.

Existen diversos tipos de robots, uno de los mas comunes son los brazos robóticos de 5 Grados de Libertad(GDL). Estos brazos poseen una alta versatilidad, pues solo es necesario reemplazar la herramienta en el elemento terminal para darle una nueva aplicación. Algunos aplicaciones de brazos robóticos son la agricultura[5], sonografía[12], soldadura en manufactura[3, 10], entre otros.

La soldadura eléctrica o por resistencia es un proceso termoeléctrico en el que se genera calor, mediante el paso de la corriente eléctrica a través de las piezas, en la zona de unión de las dos partes que se desea unir. Esto se realiza por un tiempo controlado con precisión y bajo una presión también controlada. Los metales en este proceso se fusionan entre sí sin necesidad de aporte, es decir, por aplicación de presión y la corriente eléctrica sobre las áreas que se van a soldar sin aporte de otro material. Gracias a la corriente eléctrica el material de unión se calienta llegando a fusionar. Cuando la corriente eléctrica se apaga este se solidifica[9].

La soldadura eléctrica se caracteriza por ser un proceso sencillo que no requiere de preparación especial. Es fácil adquirir los insumos y posee una versatilidad de los equipos que se usan para desarrollar esta técnica. Sin embargo, encontrar un operario que tenga la habilidad de realizar una soldadura con la técnica adecuada de acuerdo al caso que se presente es desafiante. Los seres humanos pueden cometer errores que pueden resultar costosos en algunas aplicaciones. Contar con un robot que tenga la capacidad de ejecutar la soldadura adecuada a cada caso puede resultar beneficioso para toda empresa que se dedique al rubro de la metalmecánica.

En este trabajo, empleando los materiales disponibles en el Laboratorio de Automatización y Robótica(LAR) de la Facultad de Ingeniería de la Universidad Nacional de Asunción (FIUNA), se busca implementar modulo de exhibición de un robot de 5 GDL que sea capaz de simular un proceso de soldadura eléctrica empleando visión artificial para un área de 40x40[cm].

2. Objetivos

2.1. Objetivo General

- Implementar un robot de 5 GDL que emplea visión artificial para simular un proceso de soldadura eléctrica en un área de 40x40[cm] a través de las instrucciones recibidas en una GUI

2.2. Objetivos Específicos

- Diseñar una GUI sencilla e intuitiva.
- Implementar una visión artificial y el procesamiento de imágenes.
- Diseñar los eslabones requeridos para que el brazo robótico pueda trabajar en un área de 40x40[cm].
- Establecer un protocolo de comunicación entre la computadora y el hardware a utilizar para controlar el brazo.

3. Marco Teórico

Para el diseño de un brazo robótico de 5 GDL que simule un proceso de soldadura es necesario combinar los conocimientos de varias disciplinas. A continuación, se relatara detalladamente las investigaciones realizadas en cada rama incursionada.

3.1. Hardware a Utilizar

3.1.1. Dispositivo de Control

El primer paso es elegir el hardware que vamos a utilizar. La selección del hardware de control influirá en la programación y la selección de los periféricos adecuados. Al analizar las opciones disponibles, tuvimos en cuenta los siguientes dispositivos:

Raspberry Pi: Es un ordenador de bajo coste y formato compacto destinado al desarrollado para hacer accesible la informática a todos los usuarios. Este microprocesador trabaja principalmente con distribuciones de Linux, por lo que es altamente eficiente[7].

Arduino: Arduino es una plataforma de creación de electrónica de código abierto, la cual está basada en hardware y software libre, flexible y fácil de utilizar para los creadores y desarrolladores. Esta plataforma permite crear diferentes tipos de microordenadores de una sola placa a los que la comunidad de creadores puede darles diferentes tipos de uso[6].

Si bien LAR posee en su inventario ambos dispositivos. El arduino presenta ciertas ventajas que lo vuelven mas atractivo para su uso en comparación al Raspberry Pi. La principal ventaja de Arduino es la experiencia que los miembros de este grupo poseen con esta placa de desarrollo.

Los miembros de este grupo ya trabajaron con anterioridad con el Arduino en cátedras como Proyecto 1, como también en proyectos personales, como impresoras 3D. Utilizar el Raspberry implicaría aprender de cero sobre el funcionamiento de este microprocesador para familiarizarse con el mismo. Debido a que el tiempo era limitado, y que el Raspberry del laboratorio ya estaba reservado para otro proyecto. Se opto por el uso del Arduino MEGA para el control del brazo.

3.1.2. Selección de los motores

Para mover los eslabones del robot es necesario emplear motores con el torque adecuado. A continuación, se presentaran los motores disponibles en el laboratorio con una breve descripción de sus características:

Servomotores: Son motores que se controlan con un modulación de ancho de pulso(PWM), por lo que su control es sencillo, pero tiene la limitación de que su torque es limitado. Su costo es accesible.

Motores Paso a Paso: Son motores que se controlan con un driver complementario. Su control es de mayor precisión que el de un servomotor. Su costo es elevado.

Teniendo en cuenta las estructuras de proyectos de años anteriores disponibles en LAR, optamos por una configuración mixta. Un motor paso a paso controlara el movimiento horizontal del brazo, mientras que 4 servomotores controlaran el movimiento vertical y la profundidad a la que se realizara la soldadura.

3.2. Programación

3.2.1. Lenguaje de Programación

El lenguaje de programación se encargara de realizar el procesamiento de imágenes, y mandara las señales correspondientes. Realizando la investigación correspondiente, los lenguajes que nos resultaron atractivos para esta implementación son:

Matlab: Es una plataforma de programación y cálculo numérico utilizada por millones de ingenieros y científicos para analizar datos, desarrollar algoritmos y crear modelos. Cuenta con una comunidad que crea librerías para usos específicos como, por ejemplo, conexión con el Arduino y aplicaciones de robótica.

Python: Es un lenguaje de programación optimizado para el procesamiento de datos y esta enriquecido con librerías capaces de generar curvas de aproximación a partir de puntos, procesamiento de imágenes, visión artificial, entre otros.

Originalmente planeábamos realizar la programación con Matlab y las librerías de la comunidad. Sin embargo, para poder emplear dichas librerías era necesario tener una cuenta de Mathworks con licencia activa, y un Arduino MEGA original. Nosotros contábamos con una licencia educativa y una replica del Arduino MEGA. Luego de una búsqueda intensiva dentro de la comunidad de Matlab no encontramos nada que pueda ayudarnos. Entonces, no fue posible utilizar estas librerías con el hardware que teníamos disponible, por lo que desistimos de utilizar Matlab en esta implementación.

Afortunadamente, Python y sus librerías son mas flexibles con el uso de hardware no oficial. Otra ventaja que tuvimos al realizar la implementación con Python es que dichas librerías son maduras y ya cuentan con secciones de errores y problemas frecuentes, por lo que encontrar soluciones a los problemas que se iban presentando fue mucho mas sencillo.

3.2.2. Control de los Motores

Como previamente fue mencionado en este proyecto se trabajara con una combinación de servomotores y un motor paso a paso. Estos motores serán controlados con el Arduino, que a su vez sera controlado por comunicación serial a través del puerto USB de la computadora. Para realizar este control es necesario utilizar el protocolo de comunicación adecuado.

Tras un sondeo en internet, encontramos al protocolo Firmata. El protocolo Firmata es un protocolo genérico para comunicarse con microcontroladores desde el software en una computadora host. El objetivo central de este protocolo es hacer del microcontrolador una extensión del entorno de programación en la computadora(host) de una manera que se sienta natural en ese entorno de programación.

La finalidad del protocolo Firmata es ser un protocolo abierto y flexible para que cualquier entorno de programación pueda admitirlo, y simple de implementar tanto en el microcontrolador como en la computadora host para garantizar una amplia gama de implementaciones[11].

Afortunadamente, Python cuenta con varias librerías que trabajan con este protocolo. Algunas de estas librerías son:

PyMata: es un cliente Python de alto rendimiento, multiproceso y sin bloqueo para el protocolo Firmata que admite el protocolo StandardFirmata completo.

PyFirmata: pyFirmata es una interfaz de Python para el protocolo Firmata. Es totalmente compatible con Firmata 2.1 y tiene algunas funcionalidades de la versión 2.2. Se ejecuta en Python 2.7, 3.3 y 3.4.

Inicialmente planteamos realizar la programación utilizado PyFirmata, sin embargo, esta librería presenta la limitación de que solo puede realizar la comunicación con servomotores. Por fortuna, la librería PyMata, pese a ser una librería mas antigua y con menos soporte que el PyFirmata, posee la capacidad de controlar tanto servomotores como motores paso a paso. Por ende, optamos por utilizar la librería PyMata para esta implementación.

3.2.3. Visión Artificial

Para la implementación de la visión artificial es necesario trabajar con una librería especializada en esa rama. Luego de realizar la investigación correspondiente, consideramos utilizar una de las siguientes librerías:

OpenCV: Es una biblioteca de software de aprendizaje automático y visión artificial de código abierto. La biblioteca tiene más de 2500 algoritmos optimizados, que incluyen un conjunto completo de algoritmos de aprendizaje automático y visión por computadora clásicos y de última generación.

Matlab: Cuenta con el *Computer Vision Toolbox*, el cual tiene múltiples funciones, aplicaciones y algoritmos para ayudar con tareas relacionadas con la visión por computadora, como detección y seguimiento de objetos en fotogramas de vídeo, reconocimiento de objetos, calibración de cámaras, realización de visión estéreo y procesamiento de cargas puntuales 3D

SimpleCV: Es un framework de código abierto utilizado para crear aplicaciones de visión artificial. Con este framework se puede acceder a varias bibliotecas de visión por computadora de alta potencia, como OpenCV, sin tener que aprender primero sobre profundidades de bits, formatos de archivo, espacios de color, administración de búfer, valores propios o almacenamiento de matriz versus mapa de bits.

Originalmente, utilizar exclusivamente Matlab era muy tentadora. Sin embargo, debido al problema de incompatibilidad mencionado en la **sección 3.2.1**, tuvimos que descartar esta librería.

Si bien SimpleCV cuenta con OpenCV dentro de sus librerías de visión artificial, posee la desventaja de que sus casos de uso son limitados y presentaría dificultades al momento de implementarla para el procesamiento de imágenes en este proyecto.

Por esta razón y el hecho de que la cátedra recomienda la ultima opción restante, optamos por el uso de OpenCV para esta implementación practica.

3.2.4. Diseño de la GUI

Para el diseño de la interfaz gráfica de usuario, o GUI; se tuvo en cuenta las siguientes librerías:

TKinter: El paquete TKinter ("Tk interfase") es la interfaz estándar de Python para el kit de herramientas GUI Tcl/Tk. Tanto Tk como TKinter están disponibles en la mayoría de las plataformas Unix, incluido macOS, así como en los sistemas Windows.

Pygubu: es una herramienta RAD[1] que permite el desarrollo rápido y fácil de interfaces de usuario para el módulo TKinter de Python. Las interfaces de usuario diseñadas se guardan como archivos XML y, al usar el generador Pygubu, las aplicaciones pueden cargarlas dinámicamente según sea necesario.

El uso de Pygubu era bastante tentador, sin embargo, esta herramienta presentaba problemas de incompatibilidad con la librería de visión artificial OpenCV. Por lo tanto, para esta implementación, nos limitamos a utilizar exclusivamente la librería TKinter.

3.3. Diseño Asistido por Computadora

Existe una diversa cantidad de software que nos permite realizar el diseño asistido por computadora(CAD). El uso de un software de tipo CAD facilita significativamente el diseño de piezas en comparación a métodos convencionales[8]. Para esta implementación se considero utilizar uno de los siguientes programas:

SolidWorks: es un programa paramétrico orientado al ensamblaje que ofrece funciones avanzadas para profesionales. Sus complejas interfaces personalizables están dirigidas a usuarios experimentados.

Fusion 360: es un sistema de piezas de múltiples componentes que combina herramientas de modelado paramétrico, directo y de malla. También está completamente basado en la nube, lo que promueve la colaboración para equipos remotos.

Si bien ambos programas tienen funcionalidades similares, Fusion 360 posee ciertas ventajas sobre SolidWorks, por ejemplo:

- La posibilidad de adquirir una cuenta educativa gratuita.
- Trabaja con la nube, por lo que el acceso a los archivos es rápido y accesible.
- SolidWorks lanza una nueva versión anualmente, y cada versión trabaja con un nuevo tipo de archivo, lo cual produce problemas de incompatibilidad. Fusion 360 es una versión única, por lo que no tiene este problema.
- Sus herramientas permiten realizar un diseño con mayor rapidez y versatilidad.

Por estas ventajas, optamos por utilizar Fusion 360 para el diseño de todos los eslabones para el brazo robotico.

4. Diagrama de Bloques

4.1. Diagrama General

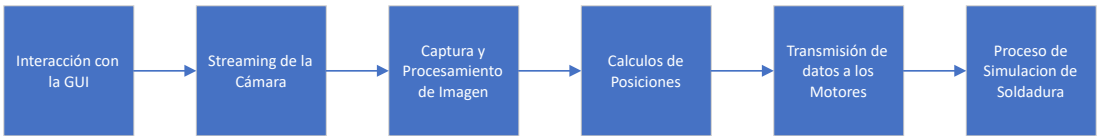


Figura 1: Diagrama de bloques del funcionamiento general del proyecto

4.2. Interacción con la GUI

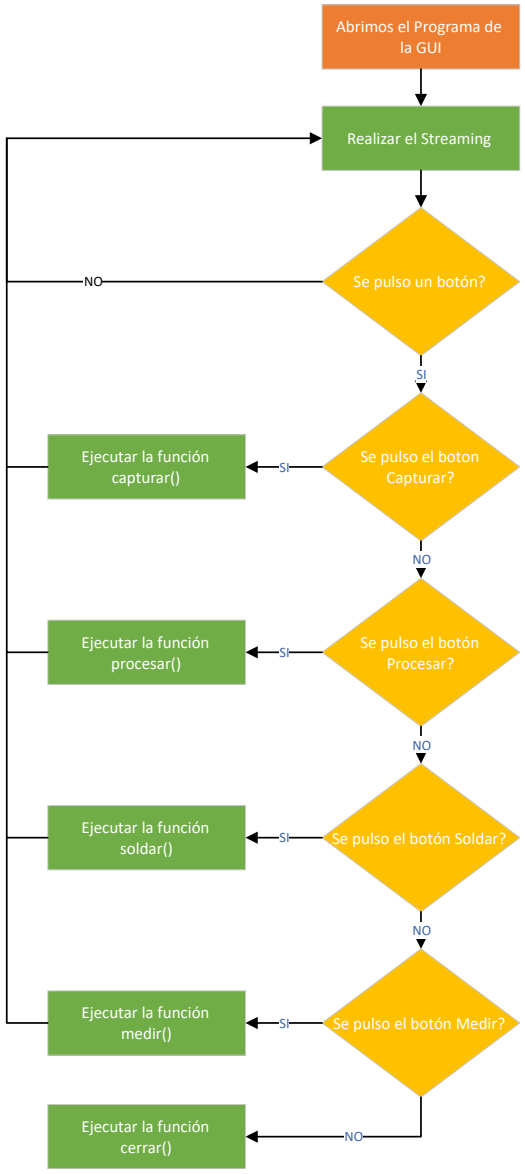


Figura 2: Diagrama de bloques de la interacción con la GUI

4.3. Streaming de la Cámara

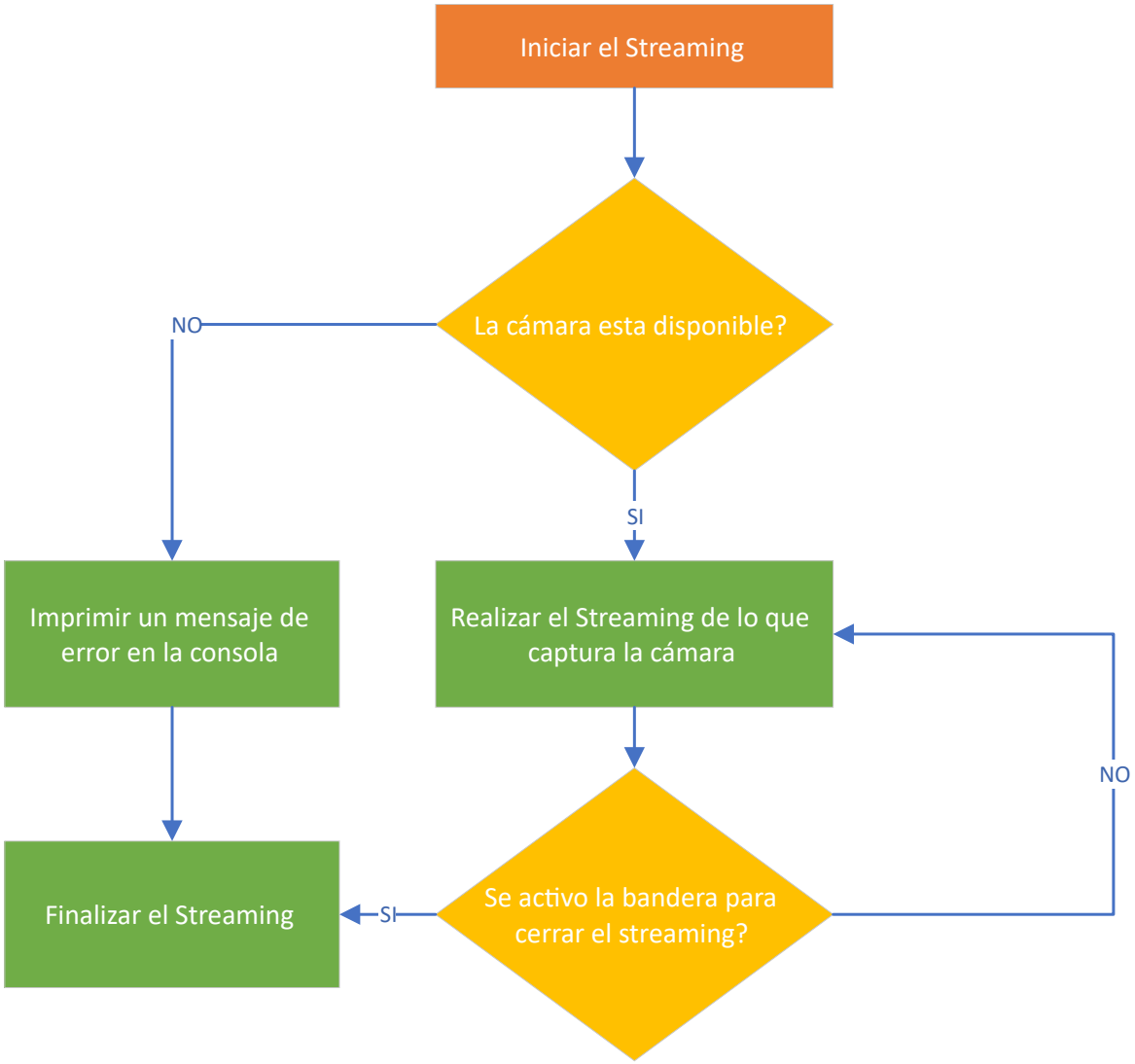


Figura 3: Diagrama de bloques del streaming de la cámara

4.4. Funcionamiento de la Captura de Imágenes

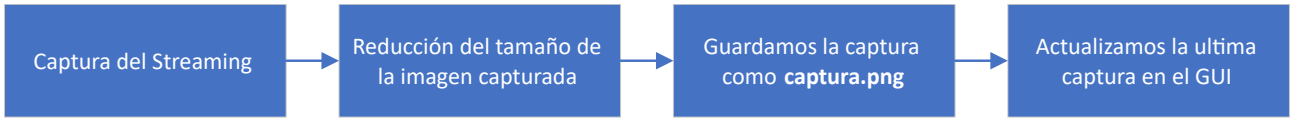


Figura 4: Diagrama de bloques de la captura de imágenes

4.5. Funcionamiento del Procesamiento de Imágenes

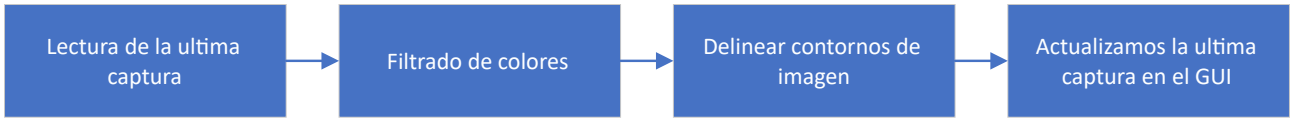


Figura 5: Diagrama de bloques del procesamiento de imágenes

4.6. Función de Carga de Cuerpos y Generación de Movimiento

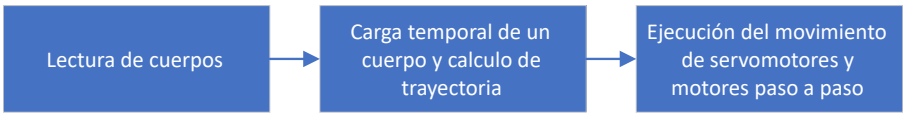


Figura 6: Diagrama de bloques de la carga de cuerpos y generacion de movimiento

5. Diseño del Proyecto

Teniendo en cuenta el diagrama de bloques de la **figura 1**, se realizara la explicación paso a paso de lo que se realiza en cada etapa.

5.1. Interacción con la GUI

La GUI fue diseñada utilizando la librería TKinter de Python. El objetivo principal al realizar el diseño de la misma fue la simplicidad y su facilidad de uso, pues esta debe mostrar en pantalla el Streaming, la ultima captura de pantalla realizada, entre otros. Para este proyecto se diseñaron dos GUIs, a continuación, se explicara con mas detalles las características de cada una

5.1.1. GUI Original

La GUI original sigue el esquema lógico presentado en la **figura 2**. Por defecto, la GUI realiza el streaming de lo que capture la cámara. Este procedimiento solo sera interrumpido si y solo si uno de los botones de la GUI es pulsado. Cada botón ejecuta una función diferente. A continuación, se realizara una descripción mas detallada de lo que hace cada función.

capturar(): Realiza una captura de pantalla, la guarda en un archivo y luego actualiza la ultima captura realizada en la GUI.

procesar(): Esta función se encarga de realizar el procesamiento de la imagen capturada para obtener así el mapa de bits que se utilizara en el proceso de soldadura.

soldar(): En esta función se recibe la imagen procesada, crea una carga temporal de cuerpos independientes detectados en procesar(), genera la cinemática inversa de cada elemento del cuerpo, y envía por puerto serial al Arduino el conjunto de instrucciones para accionar los servomotores y el paso a paso.

medir(): Realiza la medición necesaria para obtener la proporción de píxeles/cm.

cerrar(): Activa las banderas necesarias para cerrar el programa y reiniciar el kernel.

Esta GUI es bastante completa, sin embargo, ocupa gran parte de la pantalla, como se puede apreciar en la **figura 7**, y el uso continuo de la cámara consume recursos importantes cuando se realiza el calculo de posiciones. Por lo tanto, se busco diseñar una GUI mas sencilla que consuma menos recursos de la computadora.

5.1.2. GUI Modificada

Tras realizar las modificaciones necesarias a la GUI original, se obtuvo la GUI modificada de la **figura 8**. Esta GUI cuenta con un modo standby, en la cual no se realiza el streaming, y un modo en operación, en el cual se activa el streaming.

Para iniciar el modo en operación, es necesario pulsar el botón iniciar, el cual activa la cámara. Dentro del streaming se puede apreciar un rectángulo verde, el cual seria para ubicar el área de trabajo. Al realizar la captura del streaming, se recorta el área exterior a dicho rectángulo, y se reajusta la imagen para conservar las dimensiones originales de la imagen (512x384 píxeles), por lo que ya no era necesario añadir el botón medir.

Los botones *Captura,Procesar,Soldar* y *Cerrar* tienen las mismas funciones que las de la GUI original. La ultima imagen que se muestra en la interfaz gráfica es el mapa de bits de lo que se soldara. La reducción de las imágenes mostradas en la GUI nos permitió que la misma se adapte mejor a la pantalla de otras computadoras. Si bien esta nueva GUI consume menos recursos y se adapta mejor a otros dispositivos, posee una desventaja significativa que nos impidió utilizarla para la presentación final.

Durante el desarrollo de este proyecto, el programador encargado de la visión artificial y de la GUI utilizaba Linux como sistema operativo, mientras que el programador encargado del movimiento del robot utilizaba Windows. Como en todo momento se trabajo con Python, el uso de diferentes sistemas operativos no presento problemas significativos a lo largo del proyecto. Sin embargo, para la implementación de esta ultima GUI, se utilizo librerías exclusivas de Linux, y el control del brazo solo funcionaba con Windows.

Como el objetivo principal del proyecto es implementar un robot que se mueva, optamos por sacrificar la estética y consumo de menos recursos de la nueva GUI y volvimos a trabajar con la GUI original, que era funcional.

5.2. Streaming de la Cámara

El streaming se realiza conforme se muestra en la **figura 5**. Primeramente analiza si la cámara esta o no disponible. Si la cámara no esta disponible, se imprime un mensaje en la consola. Caso contrario, se realiza el streaming de lo que se captura con la cámara. El streaming termina cuando se activa la bandera para terminar el streaming. Si la bandera no es activada, el streaming continua.

5.3. Captura y Procesamiento de la Imagen

Como se menciono previamente en la **sección 5.1**, las funciones encargadas de realizar la captura de la imagen y el procesamiento de la imagen son capturar() y procesar() respectivamente.

5.4. Calculo de Posiciones

Una vez procesada la imagen se procede a implementar el cálculo de posiciones, el procesamiento arroja una tupla de cuerpos, cada cuerpo contiene un array de vectores de posiciones de píxeles de contorno de la linea a soldar. Cómo el plano de trabajo es conocido (vertical a una distancia definida por diseño), cada vector se puede primeramente escalar al área de trabajo de 40x40[cm] y proyectar sobre éste.

Primeramente, se procede a describir modelo cinemático directo Algoritmo de Denavit Hartenber(DH), se define que el área de trabajo se dispondrá en un plano paralelo al YZ del sistema de la base este plano se encontrará a una distancia de 26 cm de la base del robot, por lo que se tiene que la coordenada en X referido al sistema de base es conocido. Entonces podemos proyectar el array de vectores obtenidos anteriormente en este plano de trabajo, por lo que las coordenadas (x,y,z) son conocidas.

Luego de crear manualmente el modelo cinemático del robot, programamos una nueva clase Soldador() empleando la librería de roboticstoolbox[2] que, entre su muchas ventajas, permite calcular la cinemática inversa del robot por un método iterativo empleando el comando iKine() inverse-Kinematic, también ésta librería cuenta con un soporte de matplotlib[4], por lo que nos permite graficar en un espacio 3D un modelo de "fideos"del robot, como se puede apreciar en la **figura 15** lo que ayuda a la visualización del proceso y la obtención de resultados.

Una vez que creamos el modelo, se calcula la cinemática inversa para cada conjunto (x,y,z), esto genera 5 variables articulares (q1,q2,q3,q4,q5)(ver **figura 16**), en el programa cada variable articular es un vector donde los elementos i-ésimos están ordenados y todos poseen la misma longitud, las variables articulares calculadas se expresan en radianes, por lo que es necesario pasarlas a grados sexagesimales, además de que los flotantes deben ser redondeadas a sus enteros mas cercanos, esto ocasiona que no se tenga control sobre la cantidad de puntos repetidos, por lo que se pierde precisión en este proceso, se intenta obtener la mayor cantidad de puntos durante el procesamiento para compensar esto.

5.5. Transmisión de Datos a los Motores

Luego del proceso del calculo de los vectores de variables articulares, se procede a enviar los datos por el puerto serial. Como se expuso anteriormente se emplea el protocolo de comunicación Pymata, que permite una tasa de transferencia de 57600 bps. En el Arduino Mega, se instala una versión de FirmataExpress, que lee cada valor en el puerto serial así como la asignación del pin para servo o pines para el motor paso a paso, además se emplearon retardos de tiempos entre la ejecución de cada comando de escritura en el puerto serial, esto para evitar errores de lectura, o un cuello de botella el Arduino.

5.6. Proceso de Simulación de Soldadura

A partir del calculo de posiciones, se procede a generar la gráfica en el área de trabajo, ésta gráfica simula el proceso de Soldadura que realiza el robot, cabe destacar el el control de trayectoria es punto a punto, ya que para implementar un sistema realimentado se deberían agregar algún tipo sensor como encoders incrementales o absolutos a las articulaciones. El control de las articulaciones es mono articular de forma secuencial, cada articulación ejecuta su movimiento en un lapso de tiempo determinado por la velocidad del servomotor, para futuras versiones se planea cambiar esto por una ejecución coordinada empleando drivers o desarrollando una placa de control. El robot cuenta con una posición de inicio y final, el robot se desplaza en forma totalmente vertical hasta el punto de inicio de un cuerpo, una vez completada la soldadura de dicho contorno especificado, el robot se pone en modo espera en posición vertical y se desplaza hasta el punto de inicio del siguiente contorno detectado si lo hubiese en el área de trabajo

5.7. Diseño Digital y Físico del Brazo

En lo que respecta al diseño del brazo, una vez que se decidió armar el brazo con perfiles de Aluminio que se encontraban en el laboratorio de Automatización y Robótica, lo cual ayudo a reducir un poco los gastos del proyecto así como reducir peso del brazo. Se procedió a diseñar acoples de diferentes tamaños para poder alargar los eslabones, como se puede apreciar en la **figura 9**, y de esta forma poder cumplir con las especificaciones de área de trabajo del proyecto. De esta forma surgieron varios prototipos de brazos de diferentes medidas los cuales se pueden apreciar en las **figuras 10 y 11**.

Para el montaje del brazo, primero se procedió a imprimir los acoples para armar los eslabones del brazo, luego el brazo fue montado al carruaje de la base que actúa como articulación prismática, este carruaje fue reutilizado de un proyecto anterior así como la estructura que soporta el área de trabajo del brazo robótico (ver **figura12**). En la **figura13** se observa como quedo el brazo montado es su espacio e trabajo.

En cuanto al elemento terminal del brazo robótico(herramienta), optamos utilizar un pincel de pizarra(ver **figura14**), el cual simula ser el electrodo del soldador, dejando la marca en el papel cuando pasa por el perfil dibujado para la soldadura. El detalle que tiene el diseño es que cuenta con una holgura que lleva un resorte en su interior (ver **figura 14b**), la finalidad de esta holgura es que la punta del pincel no se separe de la hoja de papel cuando el brazo realiza el proceso de soldadura.

5.8. Presupuesto del Proyecto

Para la elaboración de este proyecto fue necesario invertir en los siguientes artículos:

Descripción	Precio en USD	Conversión a Gs.	TOTAL
Primer pedido de soportes de servomotores	26.4	184.800	184.800
Flete del primer pedido de soportes de servomotores		39.773	39.773
Segundo pedido de soportes de servomotores	14.97	104.790	104.790
Flete del segundo pedido de soportes de servomotores		21.000	21.000
Pintura negra en Spray		56.200	56.200
Hojas Tamaño A2		9.000	9.000
TOTAL (Gs.)			**415.563**

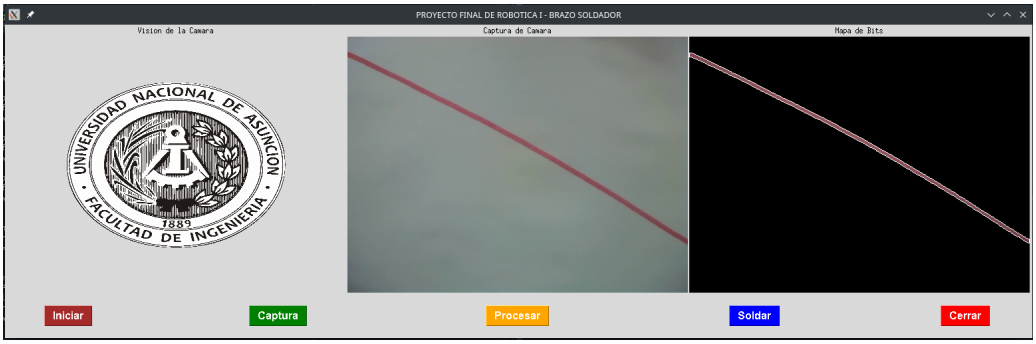
Cabe destacar que contamos con el apoyo de LAR, el cual nos proporciono los servomotores y el motor paso a paso, también nos cedieron el filamento necesario para realizar las impresiones necesarias para armar los eslabones. Sin el apoyo del laboratorio el costo de este proyecto hubiese aumentado significativamente.

6. Conclusión

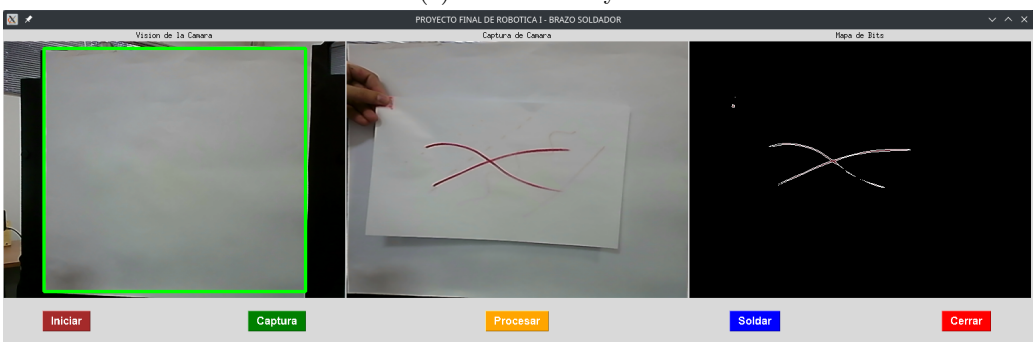
En este proyecto final de la asignatura Robótica 1 pudimos aplicar conocimientos adquiridos tanto en la cátedra como en otras de la carrera ingeniería mecatrónica. se presentaron varias complicaciones en el transcurso, las cuales las fuimos solucionando de acuerdo iban apareciendo, si bien llegando a la etapa final se presentaron problemas que no se pudieron solucionar como el problema de las holguras mecánicas que presentaban los servomotores, debido al desgaste de los mismos y también la precisión de los servomotores que manejan una resolución de grados enteros y no grados decimales. Una solución para este problema seria realizar un control directo del PWM controlando así el ancho de pulso en milisegundos, lo que nos permitiría un control mas preciso. Una solución propuesta para futuras versiones puede implementar Pymata 4 el cual permite el control PWM, así como la escritura y lectura de Pines digitales y analógicos, también presenta una mejor tasa de transferencia en baudios (Bps). Finalmente podemos decir que se lograron todos los objetivos del proyecto, dejando espacio para versiones posteriores mejoradas teniendo en cuenta lo anteriormente citado.

7. Anexos

Figura 7: GUI original

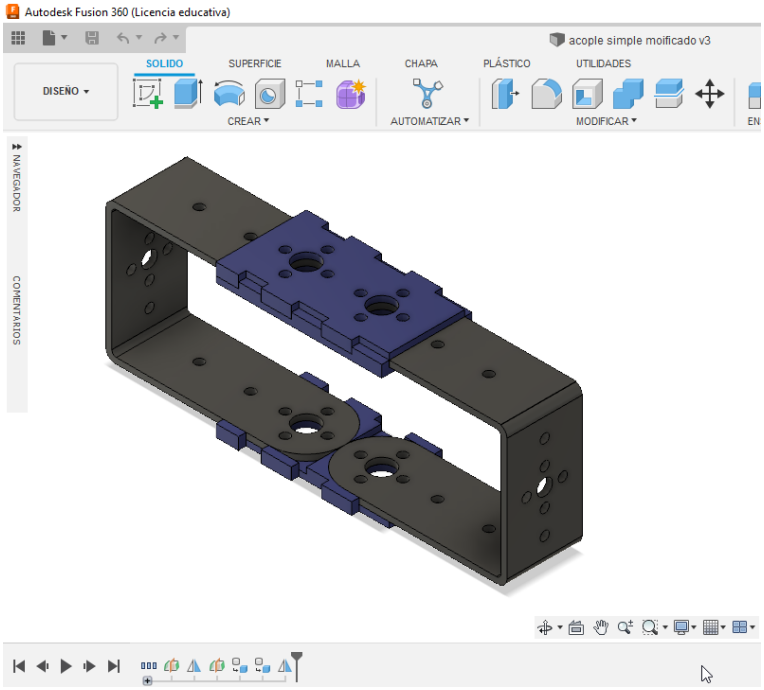


(a) Modo Standby

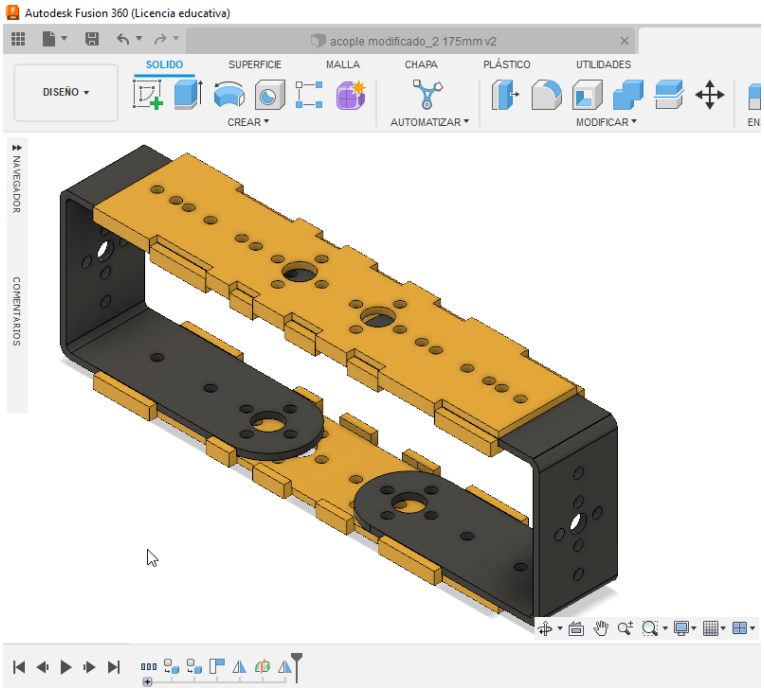


(b) Modo en Operación

Figura 8: GUI Modificada

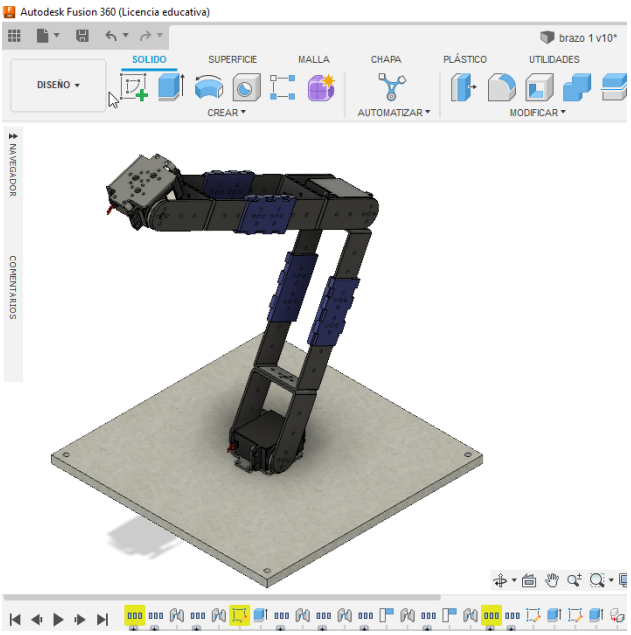


(a) Acople Simple

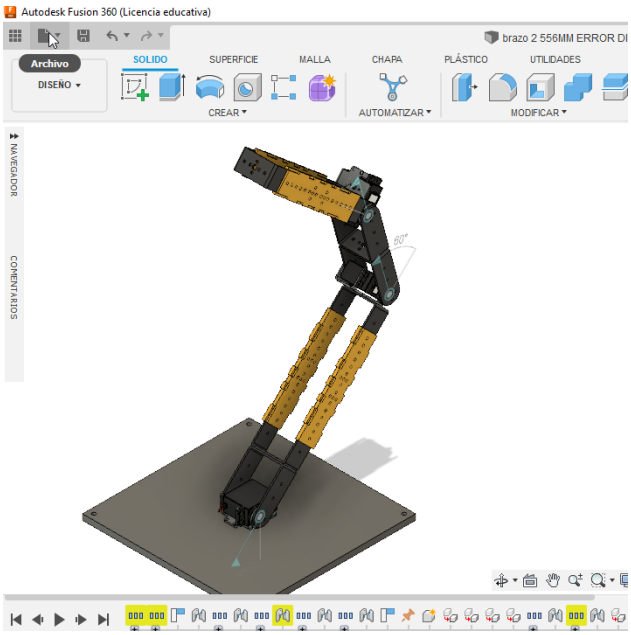


(b) Acople Múltiple

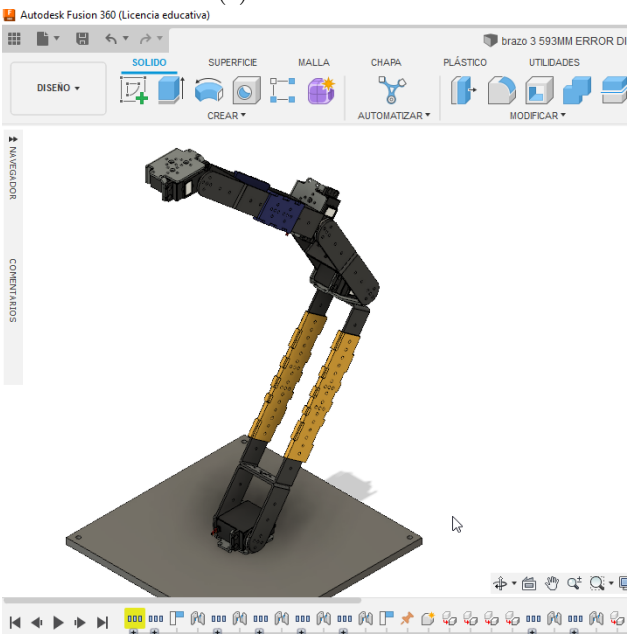
Figura 9: Diseños de Acoples para eslabones del brazo robótico



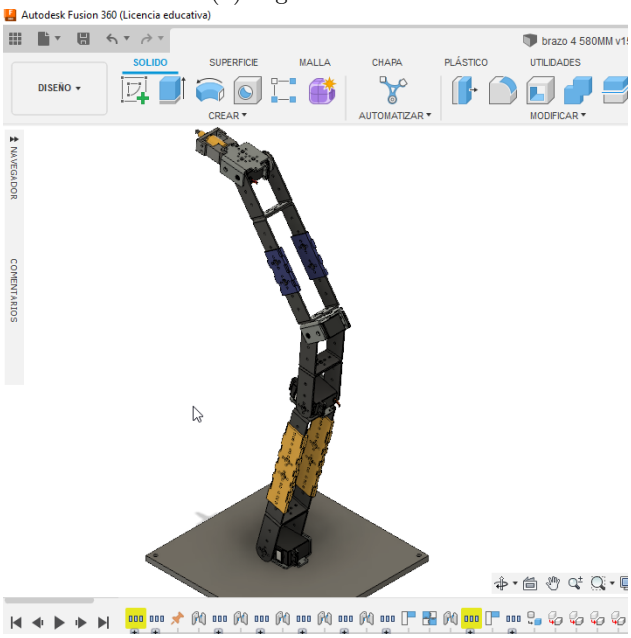
(a) Primer Diseño



(b) Segundo Diseño

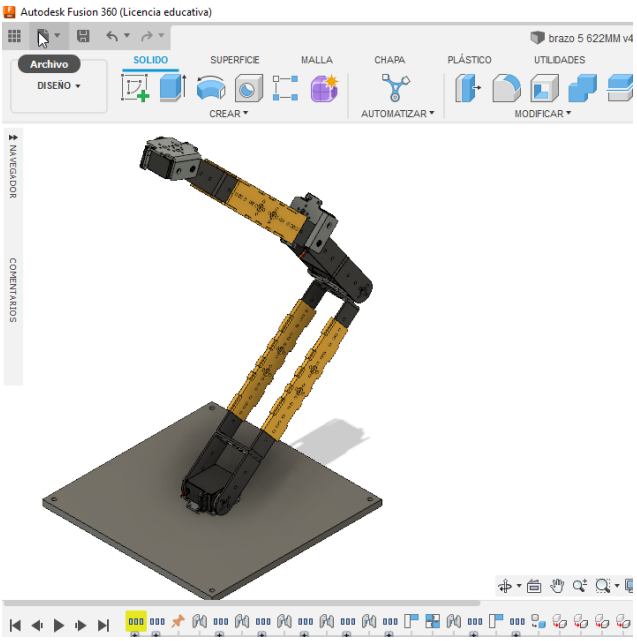


(c) Tercer Diseño

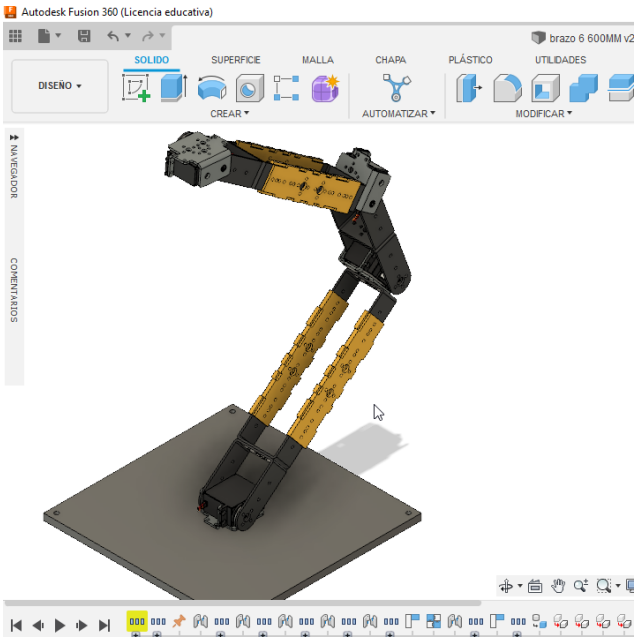


(d) Cuarto Diseño

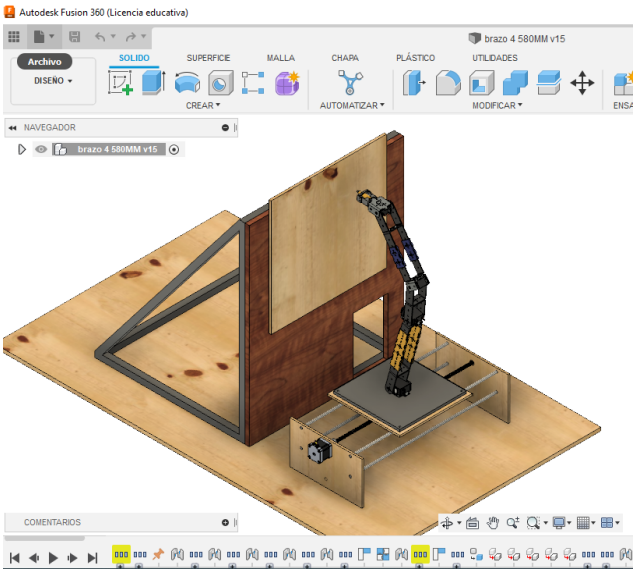
Figura 10: Primeros diseños para el brazo robótico



(a) Quinto Diseño



(b) Sexto Diseño



(c) Diseño del brazo seleccionado

Figura 11: Diseños para el brazo robótico

Figura 12: Partes Reutilizadas

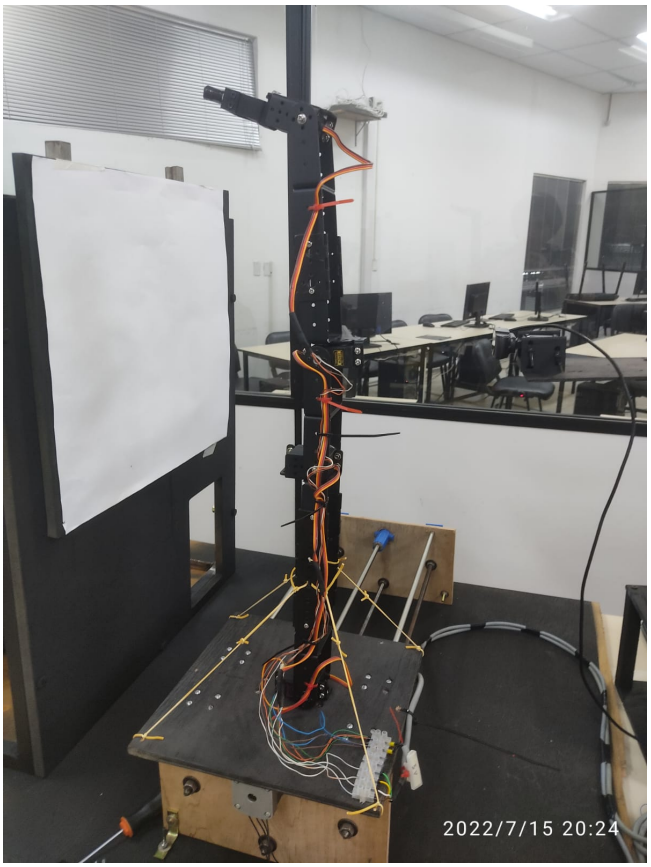
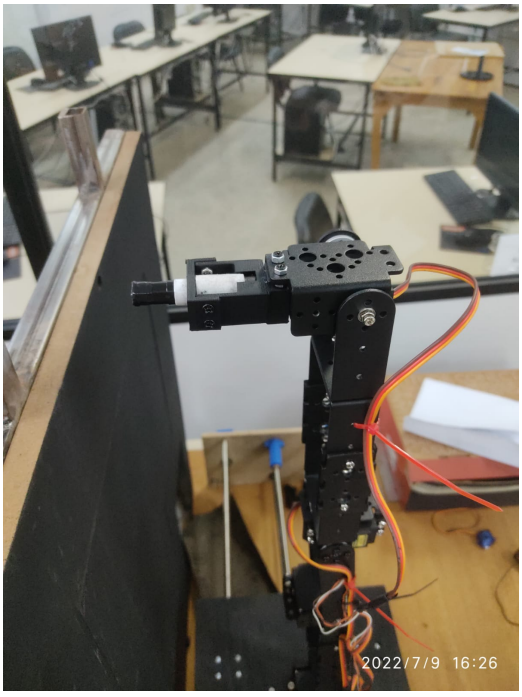


Figura 13: Brazo montado en espacio de trabajo

(a) Soporte de la herramienta del brazo

(b) Se observa el resorte en el interior de soporte



(c) soporte de herramienta montado en el brazo

Figura 14: Implementación del soporte para la herramienta del brazo robótico

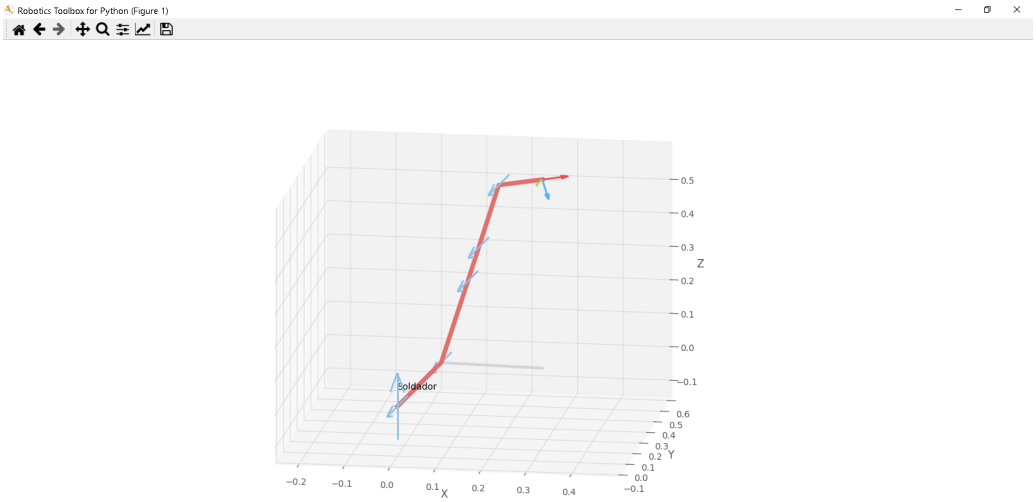


Figura 15: Representacion del robot con la libreria roboticstoolbox

θ_j	d_j	a_j	α_j	q^-	q^+
q1	0	0	90.0°	0.0°	0.0°
0.0°	q2	0	0.0°	0.0	1.2
q3 + 90°	0	0.255	0.0°	0.0°	90.0°
q4 - 90°	0	0.105	0.0°	0.0°	166.0°
q5 - 90°	0	0.195	0.0°	5.7°	90.0°
q6 + 90°	0	0.1	90.0°	-90.0°	90.0°

Figura 16: Cinematica con la libreria roboticstoolbox

Bibliografía

[1] Paul Beynon-Davies y col. «Rapid application development (RAD): an empirical review». En: *European Journal of Information Systems* 8.3 (1999), págs. 211-223.

[2] Peter Corke y Jesse Haviland. «Not your grandmother's toolbox–the Robotics Toolbox reinvented for Python». En: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, págs. 11357-11363.

[3] Immanuel Edinbarough, Roberto Balderas y Subhash Bose. «A vision and robot based on-line inspection monitoring system for electronic manufacturing». En: *Computers in Industry* 56.8-9 (2005), págs. 986-996.

[4] J. D. Hunter. «Matplotlib: A 2D graphics environment». En: *Computing in Science & Engineering* 9.3 (2007), págs. 90-95. DOI: 10.1109/MCSE.2007.55.

[5] Rajesh Kannan Megalingam y col. «Robotic arm design, development and control for agriculture applications». En: *2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS)*. IEEE. 2017, págs. 1-7.

[6] *Qué es Arduino, cómo funciona y qué puedes hacer con uno*. https://www.xataka.com/basics/que-arduino-como-funciona-que-puedes-hacer-uno.

[7] *Raspberry Pi: Qué es, para qué sirve y qué podemos hacer*. https://www.profesionalreview.com/2021/07/18/que-es-raspberry-pi/.

[8] MMM Sarcar, K Mallikarjuna Rao y K Lalit Narayan. *Computer aided design and manufacturing*. PHI Learning Pvt. Ltd., 2008.

[9] *Soldadura Eléctrica - ¿Cuándo usarla? ¿Cómo hacerla?* https://www.cursodesoldadura.org/soldadura-electrica/.

[10] I Staretu. «Robotized application of assembly and soldering–case study». En: *IOP Conference Series: Materials Science and Engineering*. Vol. 1009. 1. IOP Publishing. 2021, pág. 012056.

[11] Hans-Christoph Steiner. «Firmata: Towards Making Microcontrollers Act Like Extensions of the Computer.» En: *NIME*. 2009, págs. 125-130.

[12] Daniel R Swerdlow y col. «Robotic arm–assisted sonography: Review of technical developments and potential clinical applications». En: *American Journal of Roentgenology* 208.4 (2017), págs. 733-738.