ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELECTRÍCA Y ELECTRÓNICA

ENSAMBLAJE Y CONTROL DE UN ROBOT CON ARQUITECTURA ANTROPOMÓRFICA DE 5 GRADOS DE LIBERTAD

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y AUTOMATIZACIÓN

TOMO II

MIGUEL ALEJANDRO PÁEZ LÓPEZ

miguel.paez@epn.edu.ec

DIRECTOR: NELSON GONZALO SOTOMAYOR OROZCO, MSc.

nelson.sotomayor@epn.edu.ec

DMQ, octubre 2022

CERTIFICACIONES

Yo, Miguel Alejandro Páez López declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

MIGUEL PÁEZ

Certifico que el presente trabajo de integración curricular fue desarrollado por Miguel Alejandro Páez López, bajo mi supervisión.

NELSON SOTOMAYOR, MSc.
DIRECTOR

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como los productos resultantes del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

MIGUEL ALEJANDRO PÁEZ LÓPEZ

NELSON GONZALO SOTOMAYOR OROZCO, MSc.

DEDICATORIA

El presente trabajo de titulación va dedicado a mis padres Mariana y Miguel y a mi hermano Israel, a mi abuelito Gonzalo que no pudo estar para verme culminar mi carrera universitaria pero siempre estuvo pendiente y es parte de mi vida. Finalmente, a mi sobrina Astrid Margarita esperando que su vida sea llena de éxitos y sirva de ejemplo para seguir adelante.

AGRADECIMIENTO

A mis padres Mariana y Miguel por apoyarme en las decisiones que he tomado en mi vida, darme la fuerza para no rendirme y enseñarme que siempre que se desea algo hay que luchar para lograrlo, por estar conmigo en los largos días e interminables noches de desvelo.

A mi hermano por ser mi guía y ejemplo para seguir, por siempre estar conmigo y alegrarme con sus locuras, por enseñarme a ver las cosas de una manera distinta y seguir mis propios sueños.

A mi compañera de Trabajo de Integración Curricular y amiga Evelyn, por apoyarme en los momentos más difíciles de la universidad por estar conmigo y ayudarme desde que empezamos la universidad hasta el final, agradecerle por enseñarme la fuerza y carácter para seguir adelante, por darme una nueva perspectiva de la vida y sobre todo estar conmigo a pesar de los días y noches de estrés y cansancio que vivimos.

A mi tutor, MSc, Nelson Sotomayor, por el tiempo y la paciencia que dedico a guiarme para poder completar el presente Trabajo de Integración Curricular.

A mis compañeros Francisco, Dario, Giovanny y Pablo por ser los amigos incondicionales que estuvieron junto a mí dándome ese apoyo para seguir y enseñándome que la vida no siempre es fácil y cada día es una nueva lucha.

ÍNDICE DE CONTENIDO

RESUMEN

En el Tomo II del presente trabajo de integración curricular, se plantea como finalidad el control del robot de 5 grados de libertad que se ensambló en el Tomo I. Para lo cual se realiza el control en lenguaje de bajo nivel y alto nivel, los mismos que serán enlazados por un bloque de memoria compartida. Para el lenguaje de bajo nivel se utiliza un microcontrolador de la plataforma Arduino que se encargue del control de los motores, permitiendo tanto el envío y recepción de información utilizando comunicación serial RS-485, mientras que, en el programa en alto nivel se encuentra la cinemática para el control de los ángulos de movimiento, el algoritmo para el trazado de figuras y la interfaz gráfica del usuario, todo esto en el software Matlab. Finalmente, se dispone de una memoria compartida desarrollada en el software Visual Studio, en esta se tiene el acondicionamiento de los datos, tanto de entrada como de salida y el envío de estos entre Matlab y Arduino.

Al concluir con este proyecto, en el laboratorio de control y sistemas se contará con un prototipo de un brazo articulado en el cual se podrá probar diferentes algoritmos de control para lograr que el brazo realice las tareas que el usuario necesite, siempre y cuando estas se encuentren dentro de las capacidades físicas del robot.

PALABRAS CLAVE: antropomórfico, memoria compartida.

ABSTRACT

In Volume II of this curricular integration work, the purpose is controlling the robot with 5 degrees of freedom that was assembled in Volume I. For which the control is carried out in low-level and high-level language, both will be linked by a shared memory block. For the low-level language, a microcontroller from the Arduino platform is used to control the motors, allowing both the sending and receiving of information using RS-485 serial communication, while the high-level program contains the kinematics for the control of the angles of movement, the algorithm for drawing figures and the graphical user interface, all this in the Matlab software. Finally, there is a shared memory developed in the Visual Studio software, in which the data, input and output, is conditioned, and they are sent between Matlab and Arduino.

At the end of this project, the control and systems laboratory will have a prototype of an articulated arm in which different control algorithms can be tested to ensure that the arm performs the tasks that the user needs, these are within the physical capabilities of the robot.

KEY WORDS: anthropomorphic, shared memory.

1. INTRODUCCIÓN

Una vez que se cuenta con el robot antropomórfico gracias a lo realizado en el Tomo I, se plantea el control de un robot manipulador, capaz de realizar el trazado de figuras predeterminadas por medio del control de sus cinco grados de libertad.

El sistema embebido, Arduino, cuenta con el control de bajo nivel, en el cual se tiene los comandos para el movimiento de los motores y la lectura de la posición de estos, además de su inicialización y configuración de la velocidad de comunicación. Estos datos son enviados y recibidos desde una memoria compartida creada en el software Visual Studio, la cual sirve como enlace entre Simulink de Matlab y Arduino, es decir, entre el lenguaje en alto nivel y el lenguaje en bajo nivel.

Para la comunicación desde Simulink hacia Visual Studio se utiliza un bloque llamado “Generic Share Memory Block” en este se debe especificar la cantidad de datos enviados y recibidos, y el tipo de datos que se está trabajando. Por otro lado, la comunicación con Arduino se lo realiza por medio de su puerto de comunicación serial USB hacia el computador.

En lo referente a la memoria compartida se tiene toda la programación relacionada a la apertura y cierre de comunicación, tiempos de funcionamiento, valores que recibe por parte de la memoria compartida de Simulink, datos recibidos de Arduino, transformación y acondicionamiento de la información, etc. [1]. Esto permite que a futuro el operador o la persona encargada de la manipulación del presente brazo no tenga que preocuparse por todos estos parámetros, permitiéndole centrarse en otros aspectos como controladores, añadir otras funciones o más figuras para el seguimiento de rutas que serían adicionadas por medio del lenguaje en alto nivel.

Por otro lado, dentro del control de alto nivel, Simulink, se tendrá la conexión con el operador, en este punto se creará la interfaz gráfica con todos los parámetros editables para la manipulación y puesta en funcionamiento del brazo robótico.

El producto final será una unión entre el software y hardware el mismo que se encontrará en el laboratorio de Control y Sistemas de la Escuela Politécnica Nacional en el cual se pretende que los estudiantes puedan manipular y trabajar sobre este equipo lo cual permitirá reforzar y mejorar sus conocimientos.

1.1 OBJETIVO GENERAL

Ensamblar y controlar un robot industrial con arquitectura antropomórfica de cinco grados de libertad

1.2 OBJETIVOS ESPECÍFICOS

Diseñar un sistema de control en Matlab-Simulink para realizar figuras predeterminadas y una interfaz computacional usando memoria compartida para enlazarse con el control en bajo nivel.

Implementar el software del controlador en bajo nivel utilizando un sistema embebido el cual será encargado del control de los motores que forman parte del robot antropomórfico.

Realizar pruebas experimentales a través del trazado de figuras predeterminadas sobre un patrón de referencia.

1.3 ALCANCE

Desarrollar una interfaz gráfica en Matlab-Simulink que permita realizar el control del robot antropomórfico de cinco grados de libertad para realizar al menos dos figuras geométricas.

Desarrollar de un bloque de memoria compartida para enlazar el programa en alto nivel con el controlador en bajo nivel.

Implementación del software de un controlador en bajo nivel utilizando la plataforma Arduino la cual se encargará de controlar el movimiento de los motores y la comunicación con la interfaz gráfica en Matlab-Simulink.

Validación del prototipo trazando las figuras predefinidas sobre un patrón establecido, las cuales serán escogidas por el usuario en la interfaz de Matlab.

1.4 MARCO TEÓRICO

Todo lo referente a la parte teórica se encuentra explicado en el Tomo I, es por ello que para una mejor comprensión del presente trabajo, es necesario haberlo revisado y leído antes.

2. METODOLOGÍA

El Tomo II de este trabajo de integración curricular presenta un método explicativo y experimental debido a que toma el ensamblaje realizado y explicado en el Tomo I para llevar a cabo el diseño de software que sea capaz de realizar el trazado de figuras predeterminadas.

En el capítulo 1 se encuentra los objetivos que se deben alcanzar en este Tomo, así como los alcances, no obstante, se debe señalar que la fase teórica se encuentra desarrollada en el Tomo I de este trabajo de integración curricular.

De la misma manera, una parte de la fase metodológica se encuentra en el Tomo I, se trata del ensamblaje mecánico y electrónico del robot antropomórfico, sin embargo, el diseño de software donde se explica el control en bajo y alto nivel se desarrolla en este Tomo.

Una vez realizado el diseño de software se continua con la fase de simulación por lo que esta se encuentra dentro del capítulo 2, es aquí donde se incluye la memoria compartida para enlazar el controlador de bajo nivel con el programa en alto nivel, para que ambos se encarguen del movimiento de los motores y logren el objetivo del trazado de figuras predeterminas.

Por último, en el capítulo 3 se tiene la fase de validación y pruebas de funcionamiento, en este se coloca resultados del funcionamiento obtenido por el desarrollo que se lleva de acuerdo con lo señalado en el presente trabajo.

Antes de iniciar con el diseño de los programas de control es necesario realizar la configuración de los parámetros internos de los motores.

2.1 CONFIGURACIÓN DE LOS PARÁMETROS INTERNOS DE LOS MOTORES

Es necesario realizar la configuración de los parámetros internos de los motores esto debido a que en algunos casos es necesario limitar su ángulo de movimiento para que cumpla con los grados de rotación de cada eslabón como se menciona en el Tomo I. Para el caso de los motores Dynamixel se adquiere el módulo USB2-Dynamixel, con el cual se puede configurar los parámetros por medio de un programa en la PC al enlazar dicho módulo con los motores, mientras que, para los motores Torxis se realiza la configuración por medio de una tarjeta que se encuentra en el interior de dichos motores, de igual manera esta configura los parámetros por medio de un software en la PC.

2.1.1 CONFIGURACIÓN DE MOTORES DYNAMIXEL

Para la configuración de los motores de la marca Dynamixel se adquirió el módulo USB2-Dynamixel, este se encarga de toda la configuración interna de dichos motores, como: voltajes y corrientes permitidas, posiciones máximas y mínimas, velocidades de giro, torques máximos y mínimos, identificador de cada motor, entre otros. En este punto la fijación es en la estructuración de los ángulos de movimiento de los motores, así como la velocidad de giro, esto debido a la forma constructiva del brazo la cual requiere limitantes en los ángulos de giro.

2.1.1.1 Módulo USB2-Dynamixel

El módulo USB2-Dynamixel permite modificar y configurar los valores de los motores que pertenecen a la marca Dynamixel (Figura 2.1) [1]. Este módulo también se lo puede observar en la Figura 2.2, en esta se nota que el módulo consta de un conector serial 232, un conector de USB, un conector de 4 pines y otro de 3 pines, estos dos últimos es en donde se conecta los motores para realizar su respectiva configuración [2].

Figura 2.1. Módulo USB2-Dynamixel [1]

Figura 2.2. Estructura del módulo USB2-Dynamixel [2]

2.1.1.2 Dynamixel WIZARD 2.0

Este software se encuentra disponible en la página oficial de la marca Dynamixel, para la configuración de los motores es necesario su instalación previa.

Como paso inicial se debe conectar el motor al módulo, realizado esto, este módulo se lo conecta a un puerto USB de la PC, una vez conectado y abierto dicho software, entre las opciones que se encuentra en la parte superior izquierda se localiza un icono de lupa "Scan" (Figura 2.3), con esto se escanea todos los puertos de comunicación disponibles hasta encontrar un motor de la marca Dynamixel.

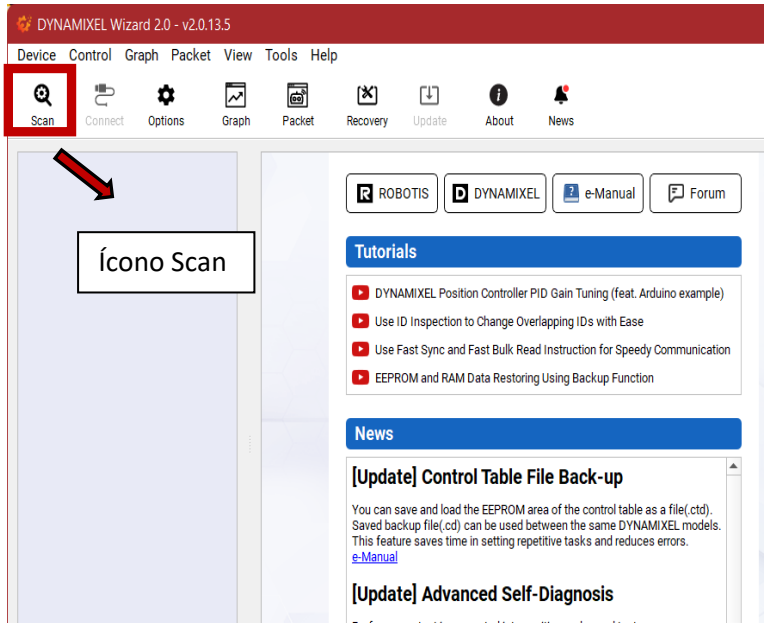


Figura 2.3. Selección de "Scan"

Es posible que este escaneo tarde demasiado tiempo por lo cual, si se desea en el apartado "Options", se puede modificar los parámetros de escaneo, permitiendo así, disminuir los tiempos de búsqueda, como se observa en la Figura 2.4.

Una vez encontrado el dispositivo (Figura 2.5) que se desea modificar se mostrará en el lado izquierdo de la pantalla en el cual se indica el modelo y la versión de este, en la parte central de la pantalla se despliega toda la información que posee este tipo de actuador, como se muestra en la Figura 2.6, finalmente en la parte derecha se permite realizar pruebas del movimiento del actuador con el fin de corroborar los valores que se están configurando, para esto es preciso destacar que se debe energizar el actuador de acuerdo a sus características propias.

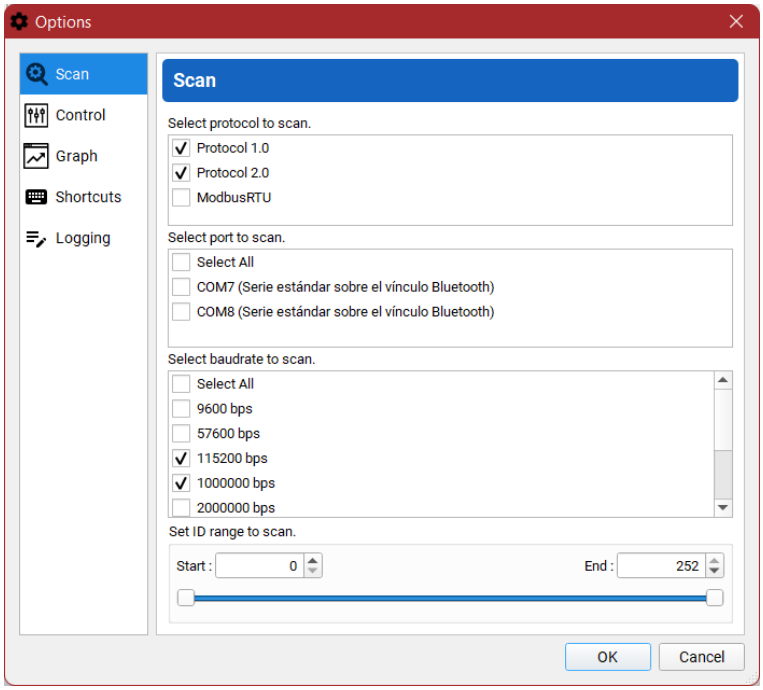


Figura 2.4. Opciones de “Scan”

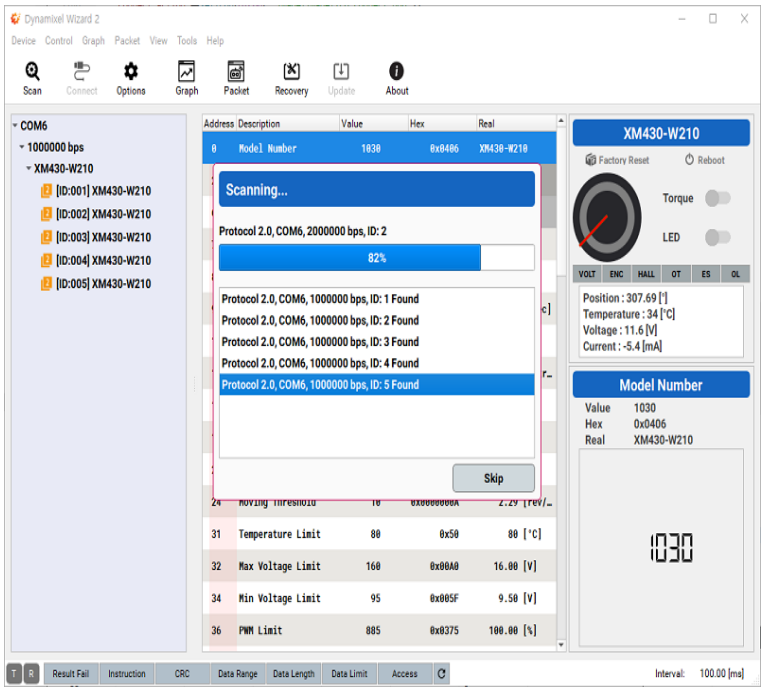


Figura 2.5. Escaneo de actuador

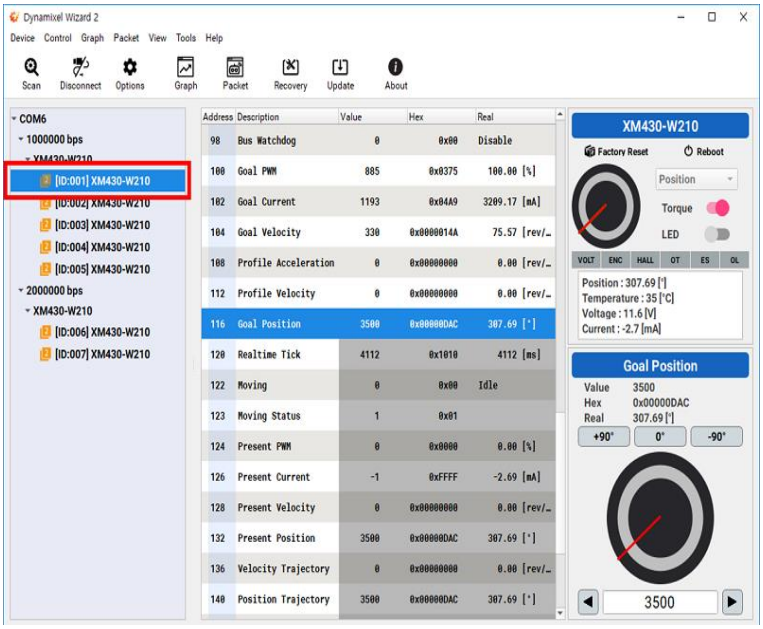


Figura 2.6. Pantalla principal del software Dynamixel Wizard

Para comprender de mejor manera, en la Figura 2.7 se muestra la estructura de la pantalla principal del software Dynamizel Wizard, en esta figura se muestra el actuador detectado, su velocidad, asimismo se indica la tabla de control que es donde se puede modificar los valores que ya se encuentran determinados en el actuador.

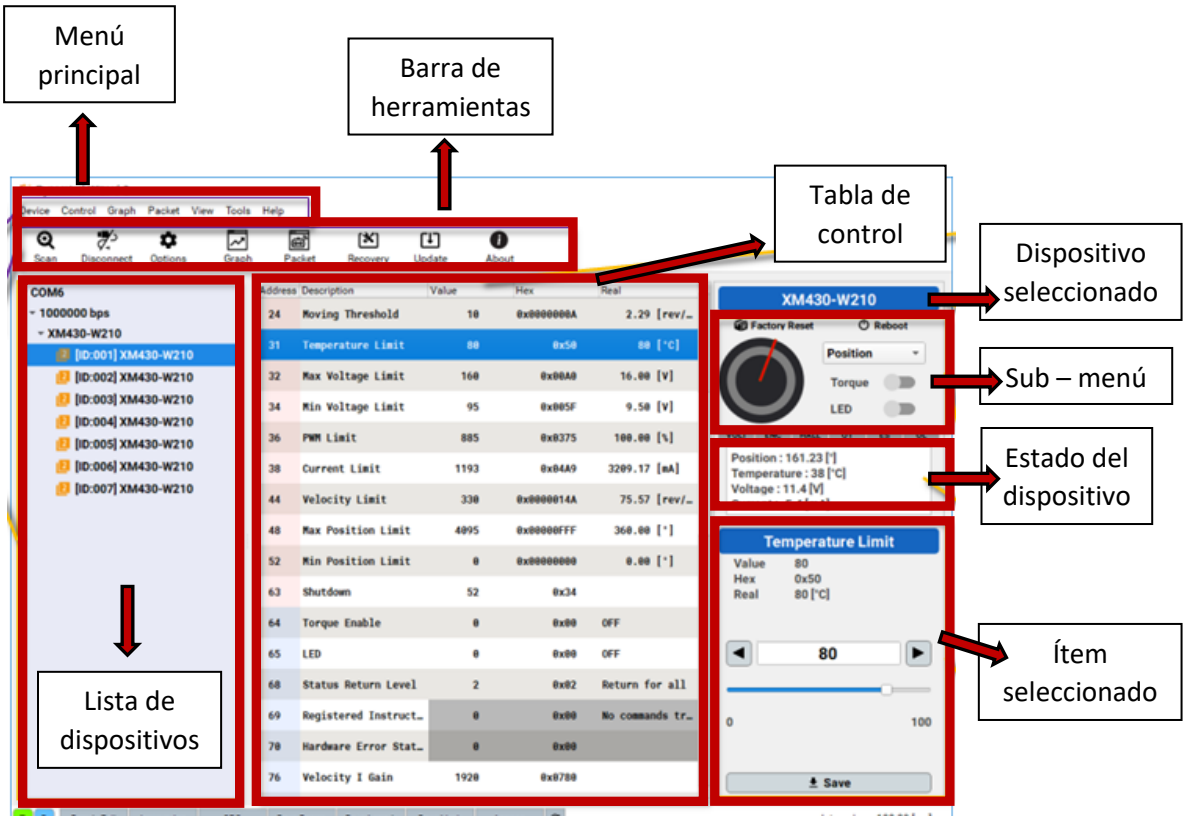


Figura 2.7. Estructura del Software Wizard [1]

2.1.1.3 Configuración de motor Dynamixel MX-28

Como se observó en el Tomo I, este motor es el encargado del movimiento de la pinza, por lo que se trata del último eslabón, el movimiento que este realiza es desde 0° hasta 360°, por lo que se modificó con el módulo previamente explicado, además cuenta con un identificador número 2, los parámetros de este motor se pueden visualizar en la Figura 2.8.

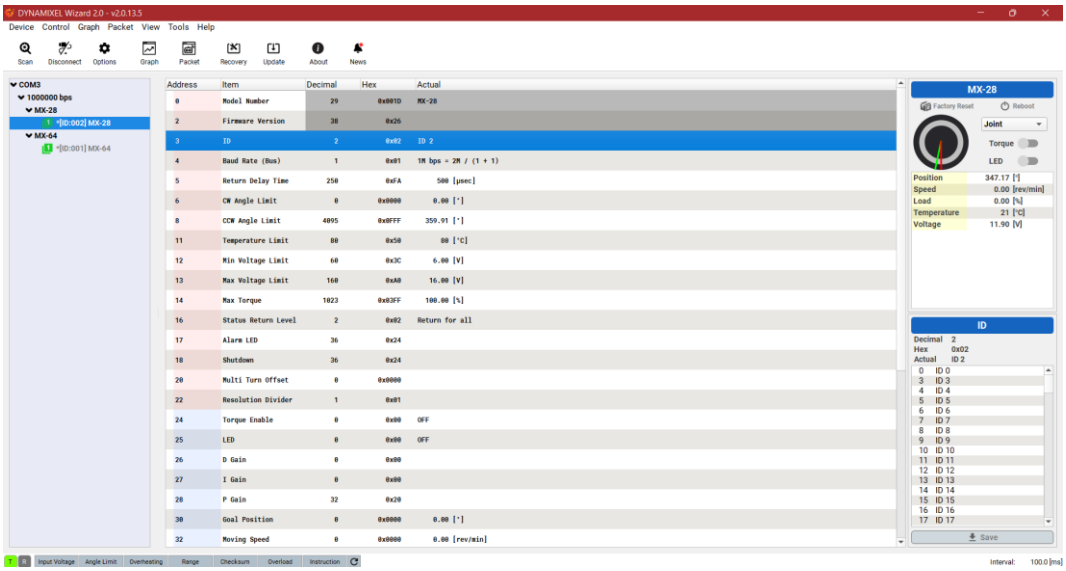


Figura 2.8. Parámetros Dynamixel MX-28

2.1.1.4 Configuración de motor Dynamixel MX-64

De igual manera que el Dynamixel MX-28, este motor se restringe los ángulos de movimiento, dado que se encuentra en la muñeca, se lo configuró para realizar un movimiento de 180°, adicional este cuenta con el identificador número 1, sus parámetros se pueden observar en la Figura 2.9.

2.1.1.5 Configuración de motor Dynamixel Pro

Este motor al tener mayor torque que los dos motores previamente mencionados posee una velocidad mayor, por lo que este parámetro se disminuyó a 8 rev/min, adicionalmente, se restringió su movimiento dado que se encuentra en el antebrazo, por lo que varía desde 0° hasta 150°, asimismo cuenta con el identificador número 15 (Figura 2.10).

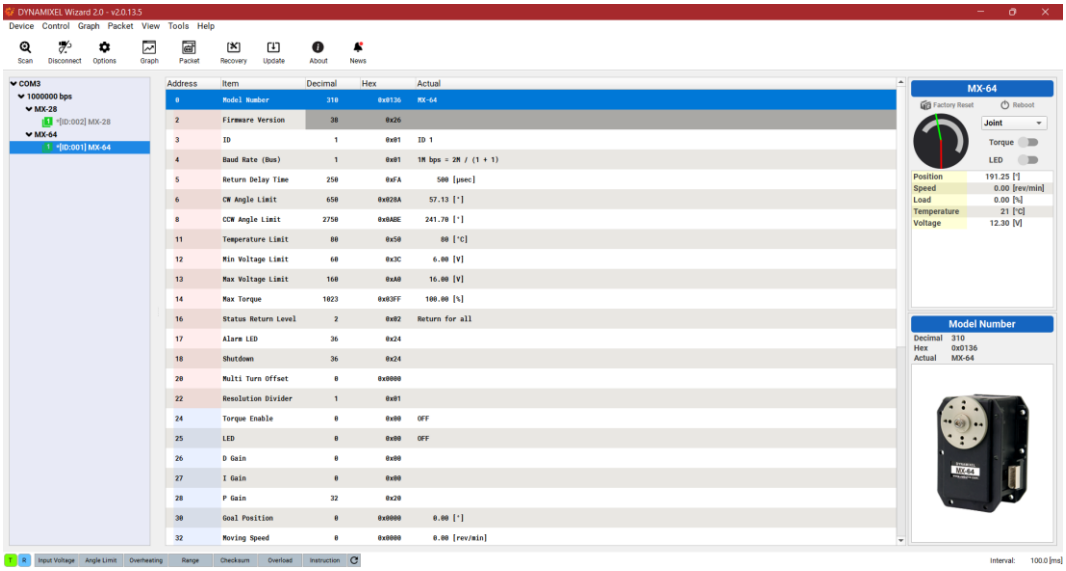


Figura 2.9. Parámetros Dynamixel MX-64

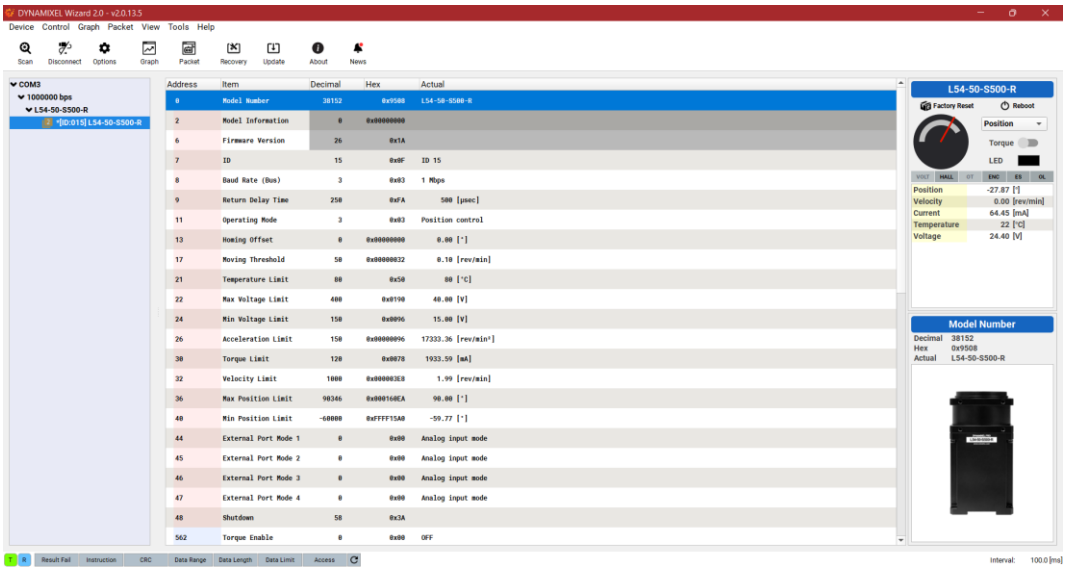


Figura 2.10. Parámetros Dynamixel PRO

2.1.2 CONFIGURACIÓN DE MOTORES TORXIS

Para la configuración de los motores Torxis no se adquirió ningún módulo, debido a que dentro de estos se encuentra la tarjeta Jrk G1 21v3, la cual se encarga de realizar el control de este tipo de motores. Cabe mencionar que está tarjeta se encuentra actualmente descontinuada y el mismo fabricante de los motores, Gearwurx, recomienda la utilización

de la tarjeta de control Jrk G2 21v3. Para la configuración de este tipo de tarjetas se tiene el software “Jrk Configuration Utility”

2.1.2.1 Pololu Jrk G1 21V3 [3]

La tarjeta de control Jrk G1 21v3, Figura 2.11, es un controlador de motores USB con realimentación, es capaz de soportar cuatro modos de comunicación: USB, nivel lógico serial, voltaje analógico y radio control. La realimentación se puede utilizar para trabajos en lazo cerrado de velocidad y posición. La corriente de salida soportada es aproximadamente 3 A, y se recomienda trabajar en un rango de voltaje entre 8 y 28 V. En este caso el fabricante de esta tarjeta, Pololu, también recomienda la utilización de la tarjeta Jrk G2 21v3.

Figura 2.11. Tarjeta de control Jrk G1 21v3 [3]

2.1.2.2 Jrk Configuration Utility [4]

Este software se encuentra disponible en la página oficial de la marca Pololu, para la configuración de los motores es necesario su instalación previa.

Como primer paso se debe conectar el módulo a la PC, por medio de un cable USB a mini-B, una vez realizado esto y abierto el software mencionado la pantalla se mostrará como se puede observar en la Figura 2.12. Además, en la tarjeta Jrk se debe visualizar un led rojo y un led verde encendidos. Se recomienda que si no se va a realizar pruebas de movimiento en los motores su alimentación de potencia este apagada o desconectada. Dentro del software en la parte superior se tiene información general de la tarjeta y el motor como: su número de serie, versión de Firmware, posición objetivo, realimentación escalada, contador de periodo PID, periodo PID excedido, motor detenido y errores, un poco más debajo de esta información se tiene 5 pestañas: Input, Feedback, PID, Motor, Errors.

En la pestaña “Input”, se puede configurar el modo de entrada, su interfaz serial y comunicación, el identificador del dispositivo, escalar la entrada y modificar manualmente la posición objetivo. Es necesario notar que algunos de estos parámetros únicamente son editables dependiendo del modo de entrada que se está utilizando.

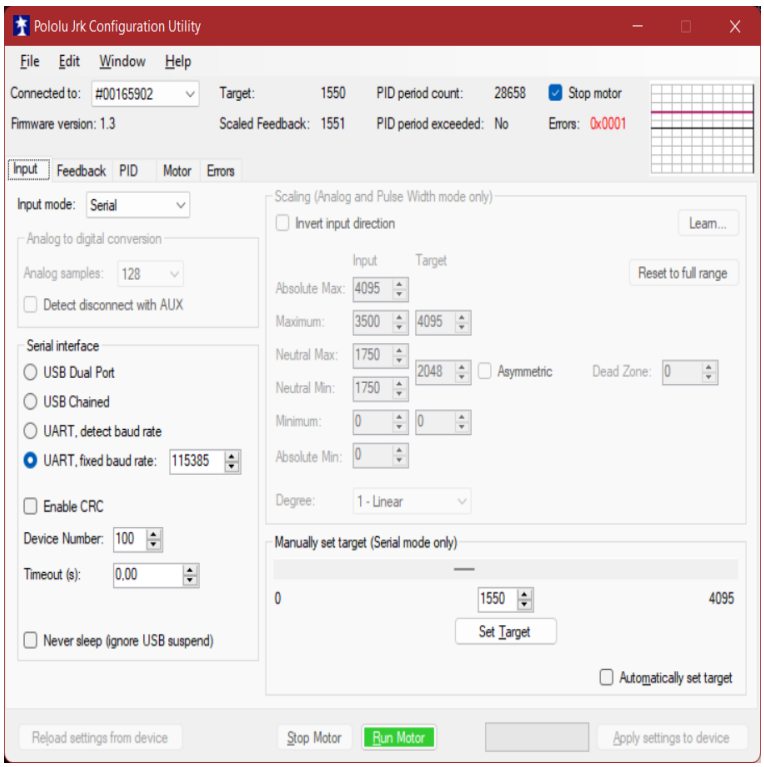


Figura 2.12. Pantalla inicial software Jrk Configuration Utility

En la pestaña “Feedback” (Figura 2.13.), se debe seleccionar el modo con el cual se va a realimentar la información ya sea voltaje analógico, frecuencia o ninguno. Una vez seleccionado esto se procede con el escalamiento de la realimentación esto para determinar como el valor realimentado en crudo se transforma en valores de realimentación deseados.

Continuando se tiene la pestaña “PID” (Figura 2.14.), cómo es de suponer en esta sección se puede configurar los parámetros: proporcional (P), integral (I) y derivativo (D), además de otros valores como son: periodo PID, limite integral, zona muerta de realimentación.

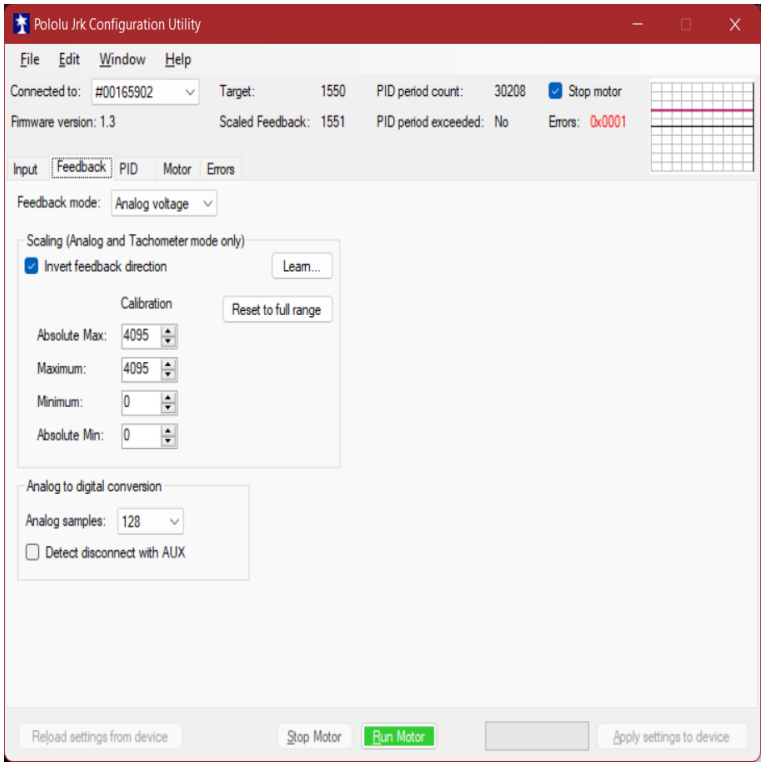


Figura 2.13. Pestaña “Feedback”

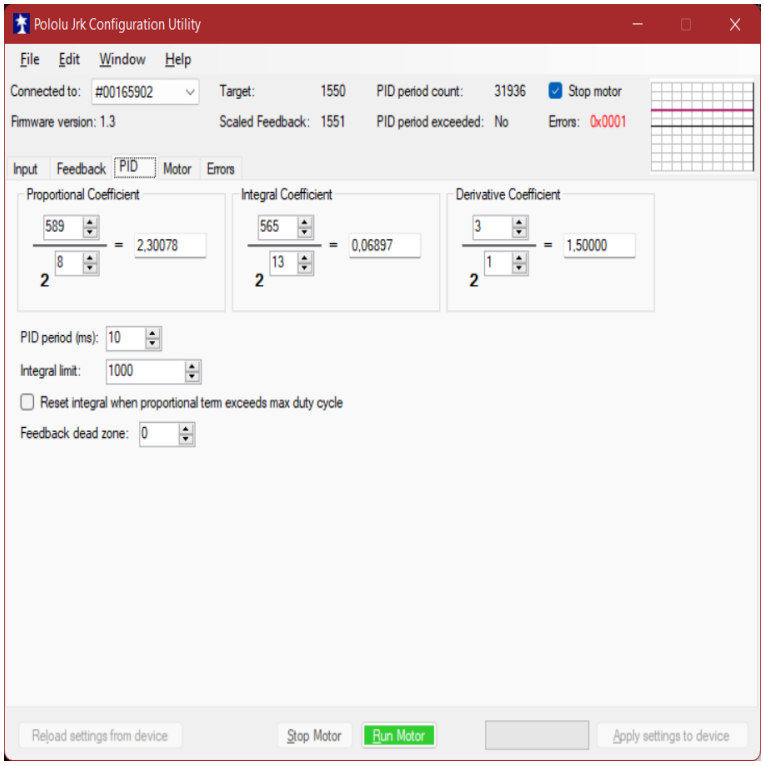


Figura 2.14. Pestaña “PID”

En la pestaña “Motor” (Figura 2.15.), se puede configurar parámetros internos del motor como son: máximo ciclo de trabajo, máxima aceleración, duración de frenado, máxima corriente, calibración de corriente, estos parámetros se pueden configurar de manera simétrica o asimétrica, sabiendo que el modo asimétrico permite configurar independientemente estos valores para giro horario y giro antihorario. También, permite cambiar y detectar la dirección de giro del motor.

Por último, se tiene la pestaña “Errors” (Figura 2.16.), en esta se puede visualizar los diferentes errores que dispone la tarjeta de control, así como habilitarlos, deshabilitarlos ó habilitarlos y permanezcan activados hasta que se solucione el error.

Como se puede observar en ninguna de las pestañas mencionadas permite delimitar la posición máxima y mínima del motor, por ello para limitar el giro de los motores Torxis de acuerdo con lo explicado en el Tomo I, se implementará restricciones por software que serán explicadas posteriormente.

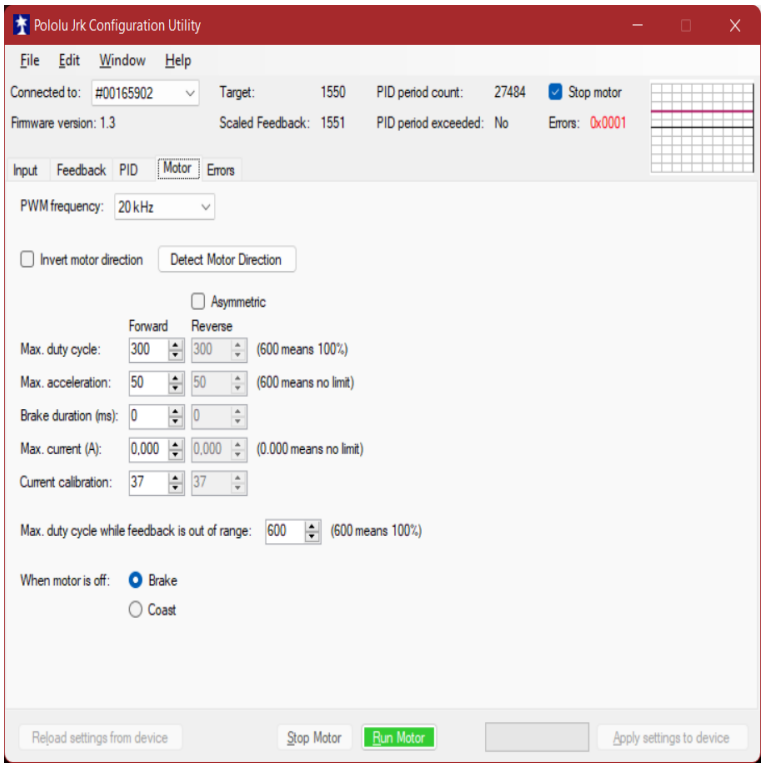


Figura 2.15. Pestaña “Motor”

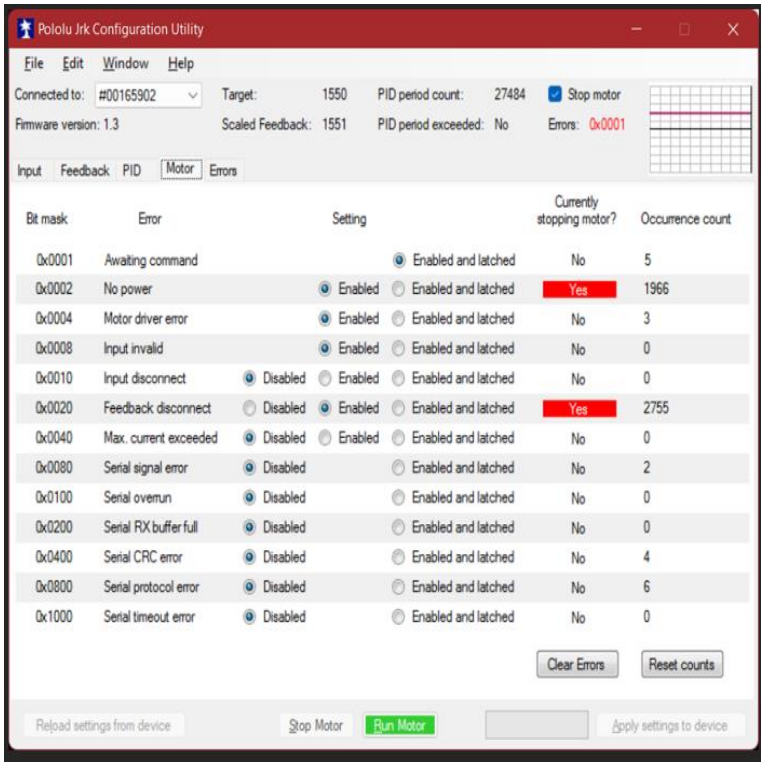


Figura 2.16. Pestaña “Errors”

2.2 DISEÑO DE SOFTWARE

La etapa de software es la unión entre diversos softwares, estos son: MATLAB, Visual Studio y Arduino, cada uno de ellos cumplen con una tarea específica para conformar el sistema completo, el mismo que se trata del programa en alto nivel, el controlador en bajo nivel y el bloque de memoria compartida. Cada uno de estos se explica en esta sección.

En la Figura 2.17 se indica un esquema general del funcionamiento del software para el trazado de figuras, en este se incluye desde los pasos iniciales hasta la finalización del trazado de figuras, se inicia desde la compilación y carga del programa en el sistema Arduino, este paso no es necesario hacerlo todo el tiempo, pero si recomendable, debido a la inicialización de los motores. La siguiente etapa corresponde al software Visual Studio, este es el enlace entre Matlab y Arduino, es decir el trazado de figuras y la acción que realiza los motores respectivamente, una vez que se ejecute el programa de Visual Studio solicita la elección del puerto de comunicación en el cual se encuentra conectada la placa Arduino, concluida esta acción se procede a inicializar la simulación del software Simulink de Matlab, una vez funcionando se regresa al software Visual Studio en donde se debe inicializar la comunicación.

Figura 2.17. Esquema general

El software que se encarga del comando de motores es Arduino, este se ocupa del controlador en bajo nivel, su objetivo es el recibir la posición deseada de los motores desde el software Visual Studio y enviar los comandos respectivos a los motores vía comunicación serial para completar esta tarea. Su diagrama de flujo se puede observar en la Figura 2.18.

Para el envío y recepción de información desde y hacia los motores se utiliza librerías predeterminadas, pues estas se encargan de comandar dichos motores, estas se utilizan tanto para los motores Dynamixel como para los Torxis. El propósito de la programación de este algoritmo es recibir la información desde la memoria compartida, esta información se recibe en dos registros de 8 bits, un registro alto y uno bajo, luego se realiza el respectivo desplazamiento para obtener un solo registro que contiene la información necesaria para realizar el movimiento del motor. Posteriormente, se realiza la lectura de la posición de cada motor, la recepción de este valor se almacena en un registro que posteriormente se dividirá en registro alto y bajo para enviar la información a la memoria compartida.

Cabe mencionar que para la comunicación con los motores se utiliza tres puertos de comunicación, el primero se encarga de los motores de la marca Dynamixel, el segundo de los motores de la marca Torxis y el tercero realiza la comunicación con la PC, también se debe notar, que en los motores se realiza a velocidades de comunicación diferentes debido a que sus tarjetas de control no permitían modificar dichos valores.

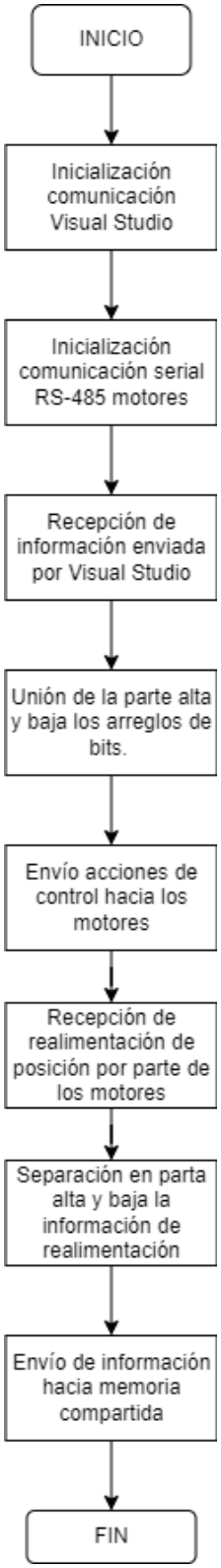


Figura 2.18 Diagrama de flujo Arduino

2.2.1 PROGRAMACIÓN DE MEMORIA COMPARTIDA CON VISUAL STUDIO

El software que se encarga de la memoria compartida es el Visual Studio, este programa ayuda con el enlace entre la comunicación de Matlab y Arduino, por lo cual una de sus primeras acciones a realizar es la apertura de comunicación con el bloque de memoria compartida ubicado en Simulink de Matlab y enlazarlo con el sistema embebido, asimismo, se encarga de realizar las acciones de recepción y tratamiento de datos, acondicionamiento y transformación de variables. Su diagrama de flujo se puede observar en la Figura 2.19.

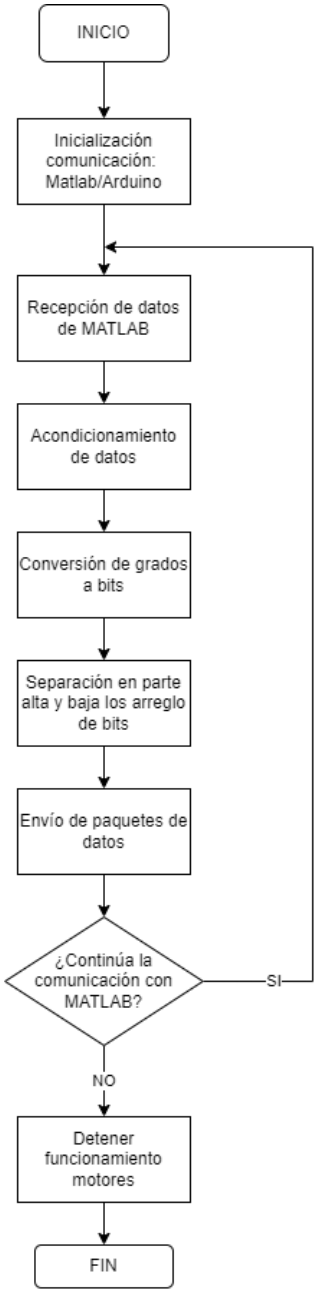


Figura 2.19 Diagrama de flujo Visual Studio

Debido a que el tiempo de procesamiento de información en Visual Studio es menor que en Matlab, se realiza la conversión de ángulos a bits y viceversa dentro de la programación del software Visual Studio, también se tiene el respectivo acondicionamiento de cada uno de los actuadores, del mismo modo se implementa una restricción de movimiento para los motores Torxis, esto ya que el software encargado de la configuración de la tarjeta de control no disponía de esta característica a diferencia del software utilizado en la configuración de los parámetros de los motores de la marca Dynamixel. La restricción de movimiento se la realiza con la ayuda de un lazo condicional “IF” limitando el movimiento de 0 a 180° para el motor Torxis de la base y de 0 a 90° para los dos motores Torxis de elevación.

2.2.2 ALGORITMO PRINCIPAL IMPLEMENTADO EN MATLAB

Como se mencionó previamente en la programación de Matlab se tiene el algoritmo para el trazado de figuras, este genera los puntos en los ejes X, Y, Z, que representan cada figura lo cual posteriormente con la ayuda de la cinemática inversa transforma estos puntos en los grados que se debe mover cada articulación para posicionarse en las coordenadas deseadas, en la Figura 2.21 se muestra el diagrama de flujo.

2.2.3 DIAGRAMA DE BLOQUES EN SIMULINK

En el diagrama de bloques de Simulink se tiene dos partes importantes, la primera se trata del bloque que posee las líneas de código para el del trazado de figuras y cinemática inversa, en este elemento se tiene como entradas la señal de la figura deseada, la señal de inicio de trazado y la señal de reloj, como es de esperar, sus salidas son los ángulos correspondientes a cada articulación (en rad/s), y una señal de control que indica cuando se terminó el trazado de la figura, este bloque se lo puede apreciar en la Figura 2.20.

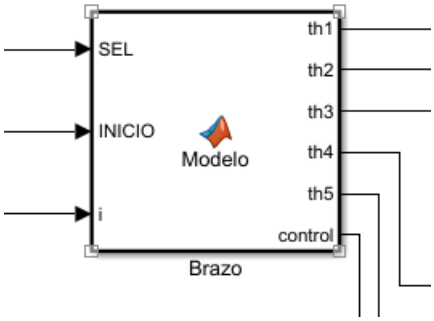


Figura 2.20 Bloque de programación

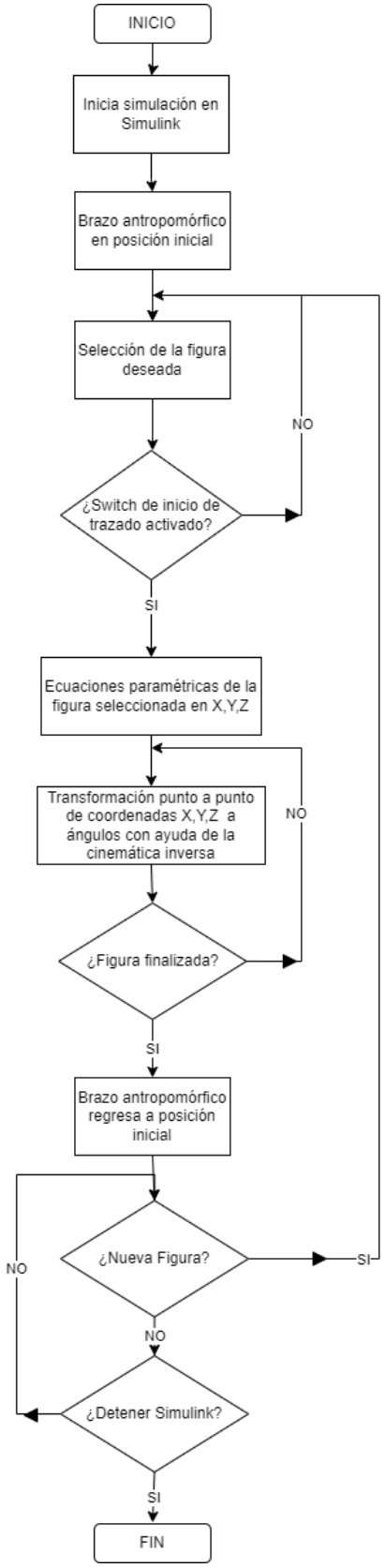


Figura 2.21 Diagrama de flujo Matlab

La segunda parte esencial es el bloque de memoria compartida, el cual, como se mencionó anteriormente, sirve como enlace para la transmisión y recepción de datos, en este bloque se debe configurar el número de elementos de entrada, el número de elementos de salida y el tipo de datos de cada uno de ellos, el bloque de memoria compartida se puede observar en la Figura 2.22.

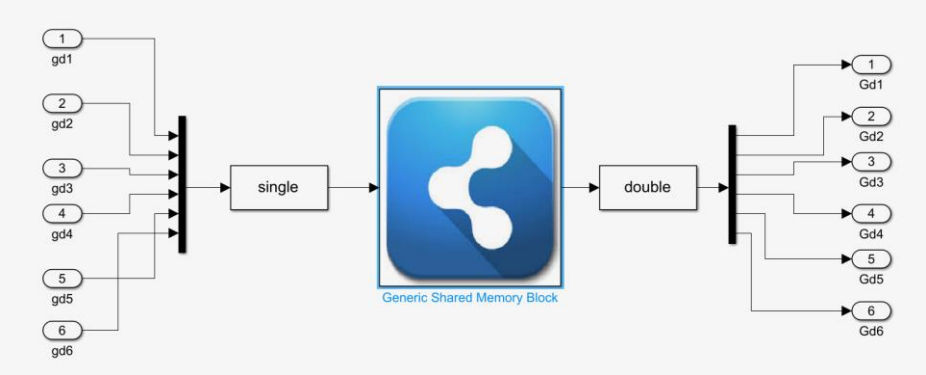


Figura 2.22 Bloque de memoria compartida

Adicional a los elementos previamente mencionados se tiene bloques que sirven para mostrar datos, acondicionar valores, etc. El diagrama completo de Simulink se lo puede apreciar en la Figura 2.23.

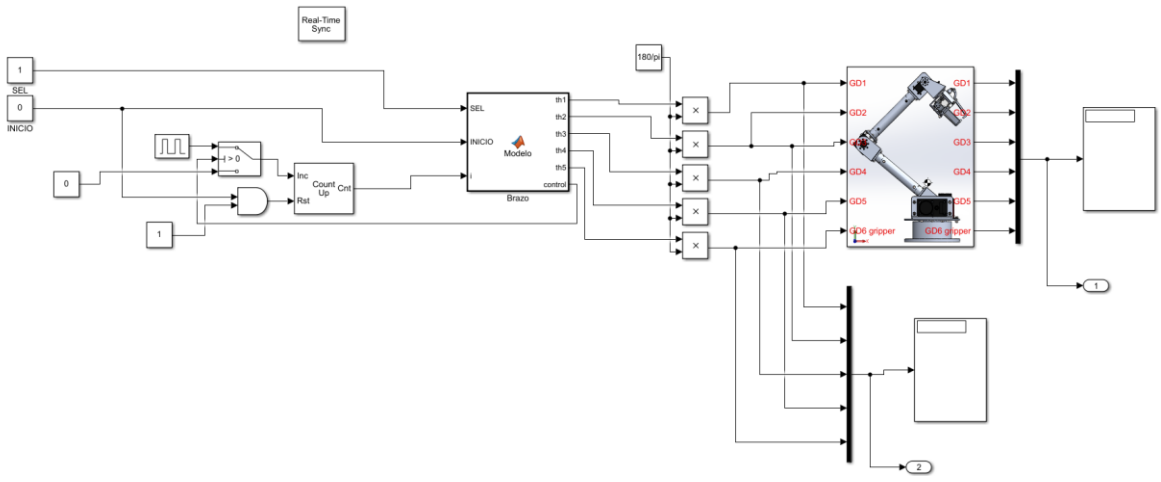


Figura 2.23 Diagrama de bloques

2.2.4 INTERFAZ GRÁFICA DE MATLAB

Se diseñó la interfaz gráfica en el propio Simulink la cual se encarga de la unión entre el operador y la parte digital, con esto se pretende facilitar el uso del brazo antropomórfico. Esta interfaz comanda el funcionamiento del Simulink de Matlab y se muestra en la Figura 2.24. Las partes que lo conforman son:

1) Menú para la selección de las figuras disponibles
2) Switch para inicia el trazado
3) Display en el cual se indica el valor teórico de los ángulos para los 5 grados de libertad
4) Display en el cual se indica el valor realimentado de los ángulos de los motores para los 5 grados de libertad
5) Bloque que contiene en su interior todo el diagrama de bloques Simulink

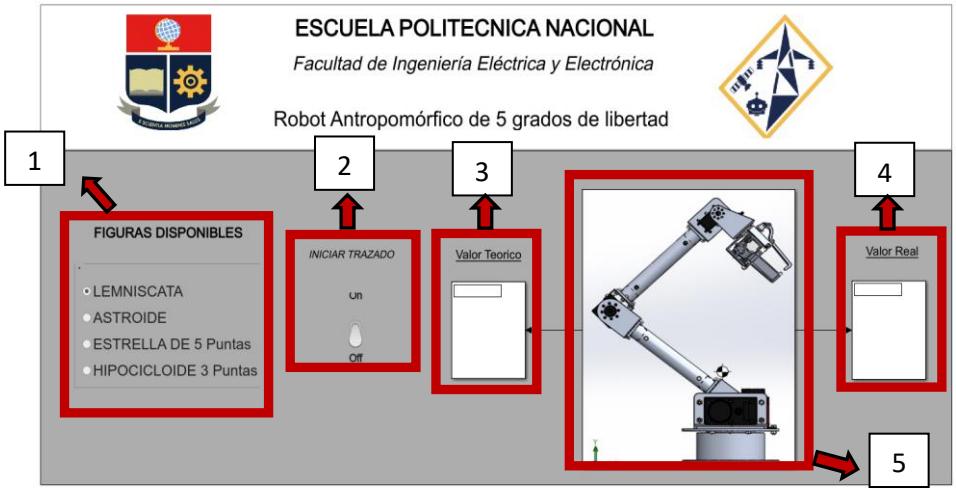


Figura 2.24 Diseño de la interfaz gráfica

2.2.5 ALGORITMO PARA EL TRAZADO DE FIGURAS

En esta sección se explica las ecuaciones para el trazado de figuras, dado que las figuras son generadas en dos dimensiones, se tomará en cuenta los ejes *xy* para obtener las ecuaciones paramétricas, esto quiere decir que el eje *z* se mantendrá constante.

Las figuras que representarán el seguimiento de ruta del efector final se obtienen mediante las siguientes ecuaciones paramétricas:

2.2.5.1 Lemniscata [5]

Las ecuaciones paramétricas de la lemniscata se encuentran en la Ec (1.1) y en la Ec (1.2)

$$x(t) = [r_x * \cos(t)] + x_c \qquad \textit{Ec (1.1)}$$

$$y(t) = [r_y * \text{sen}(2t)] + y_C \qquad \textit{Ec (1.2)}$$

Donde:

$r_x = apertura\ en\ el\ eje\ x$

$r_y = apertura\ en\ el\ eje\ y$

- Figura de Matlab

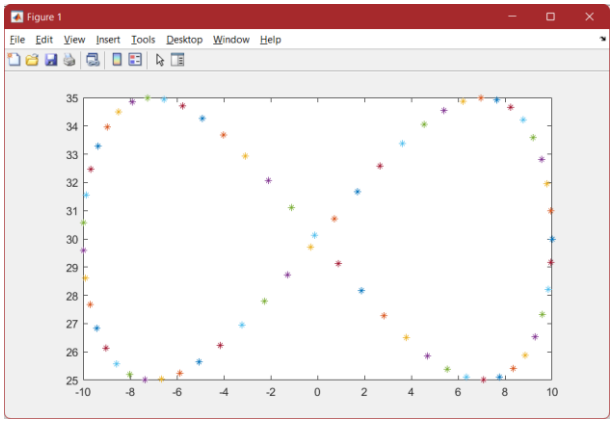


Figura 2.25 Trazado de lemniscata

- Figura Simulación con brazo

La simulación con brazo es una ayuda para saber cómo se va a mover el brazo físico al momento de realizar el trazado de las figuras, esto se lo realiza con ayuda del software Matlab y la librería “rvctools”, cabe mencionar que para esta simulación se utilizó la cinemática inversa que es implementada en el brazo físico, en otras palabras, la trayectoria que siga el brazo durante la simulación será la misma que realice el brazo de manera física.

Para realizar la simulación, como es de esperar, el primer paso es cargar la librería “rvctools” en Matlab, esta ayuda a ejecutar los comandos que crean la simulación del brazo antropomórfico con sus cinco grados de libertad, para más detalle dirigirse al Anexo II.

Una vez ejecutada la librería se abre el programa “RobotManipuladorSimulacion”, dentro de este hay que centrarse en las líneas de código que se debe modificar para graficar cualquier figura y la forma en la que se debe realizar. Para esto es necesario conocer las ecuaciones paramétricas de la figura que se desea trazar en función de X, Y y Z y sus límites. En la Figura 2.26 se muestra las ecuaciones paramétricas implementadas en la programación de Matlab correspondientes de la Lemniscata.

```
%Límites y número de puntos
t=0:0.05:2*pi;
%Ingrese la figura en sus variables X,Y,Z
 X=cos(t);
 Y=sin(2*t)+30;
 Z=10;
```

Figura 2.26 Ecuaciones para Lemniscata

En la Figura 2.26 se puede ver que los límites son de 0 a 2π en pasos de 0.05, y la gráfica por cómo se expresa variará en los ejes X y Y manteniendo Z constante. Posteriormente se dispone de un lazo “for” aquí es muy importante tomar en cuenta la cantidad de pasos que se dispuso en los límites, si se tiene más pasos se debe incrementar la dimensión de este lazo, esto se conoce por medio de la división del límite máximo, en este caso 2π, entre el valor de cada paso, 0.05, lo cual da 126, este será el límite máximo del lazo “for”, dentro de este lazo se tiene la cinemática del brazo, al igual que los comandos para graficar la trayectoria que sigue.

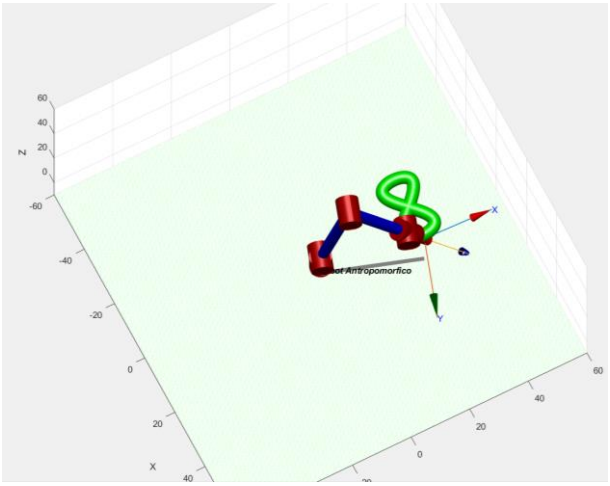


Figura 2.27. Simulación con brazo

Finalmente, resta correr el programa y si todos los valores fueron configurados correctamente se abrirá una ventana con el movimiento del brazo punto a punto, en la Figura 2.27 se muestra la figura completada.

Este mismo procedimiento se debe realizar para cada figura que se desee simular, cabe mencionar que es recomendable ejecutar una simulación de las figuras antes de proceder a su implementación por medio del brazo físico.

2.2.5.2 Hipocicloide de 3 puntas [6]

En la Ec (1.3) y Ec (1.4) se tiene las ecuaciones paramétricas de la hipocicloide de 3 puntas.

$$x(t) = S[2\cos(t) + \cos(2t)] + x_c \qquad \textit{Ec (1.3)}$$

$$y(t) = S[2\sin(t) - \cos(2t)] + y_c \qquad \textit{Ec (1.4)}$$

Donde

S: $escalamiento\ para\ el\ tamaño\ de\ la\ figura$

- Figura de Matlab

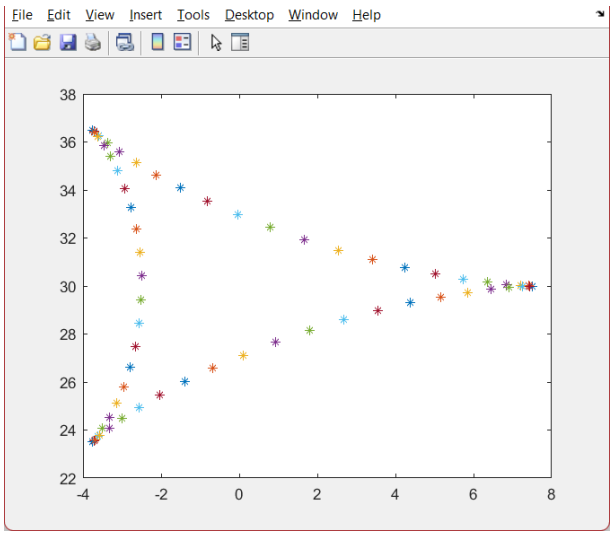


Figura 2.28 Trazado con plot

- Figura Simulación con brazo

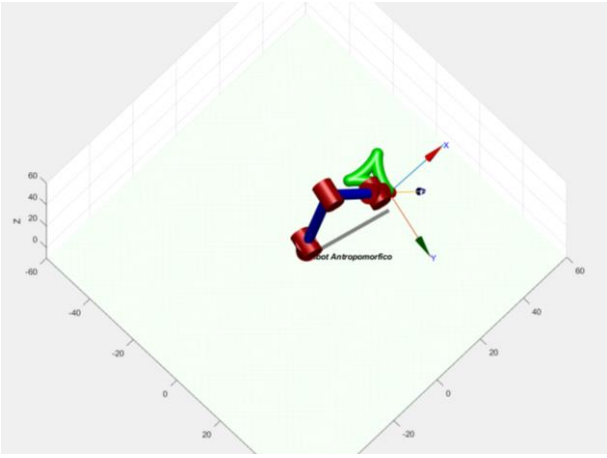


Figura 2.29. Simulación con brazo

2.2.5.3 Hipocicloide de 4 puntas o Astroide [6]

La hipocicloide de 4 puntas también se lo conoce como astroide, sus ecuaciones paramétricas se tienen en la Ec (1.5) y en la Ec (1.6)

$$x(t) = S[3\cos(t) + \cos(3t)] + x_c \qquad \textit{Ec (1.5)}$$

$$y(t) = S[3\sin(t) - \cos(3t)] + y_c \qquad \textit{Ec (1.6)}$$

- Figura de Matlab

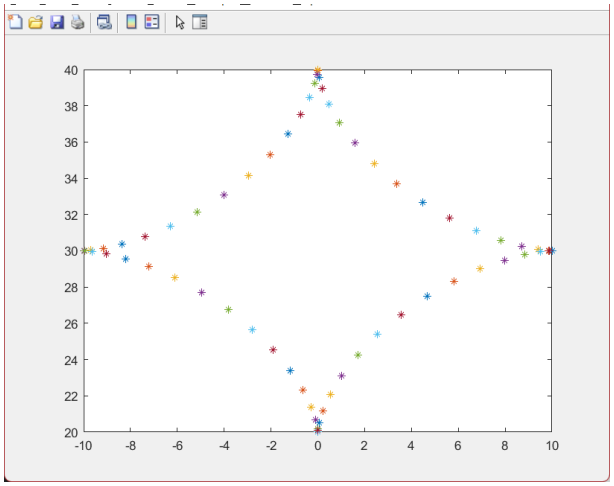


Figura 2.30 Trazado con plot

- Figura Simulación con brazo

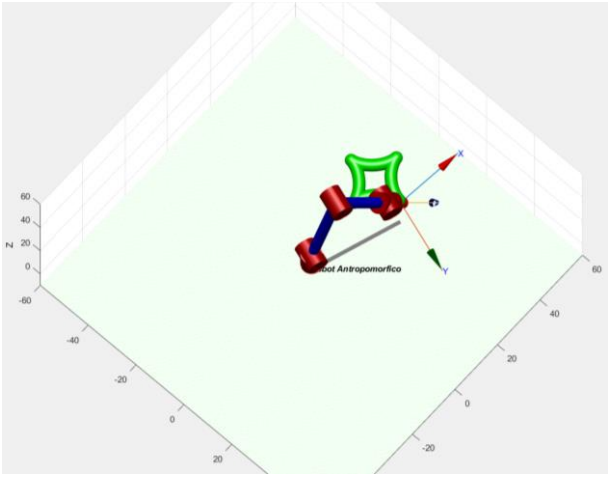


Figura 2.31. Simulación con brazo

2.2.5.4 Estrella de 5 puntas [6]

La estrella de 5 puntas posee las ecuaciones paramétricas que se muestran en la Ec (1.7) y en la Ec (1.8)

$$x(t) = S[4\cos(t) + \cos(4t)] + x_c \qquad \textit{Ec (1.7)}$$

$$y(t) = S[4\sin(t) - \cos(4t)] + y_c \qquad \textit{Ec (1.8)}$$

- Figura de Matlab

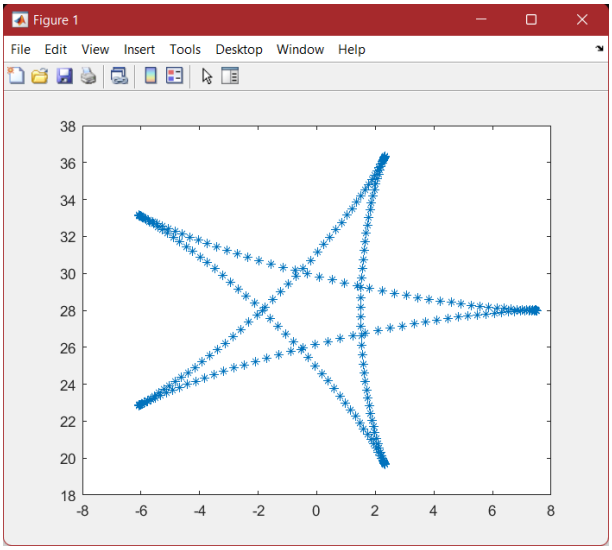


Figura 2.32 Trazado con plot

- Figura Simulación con brazo

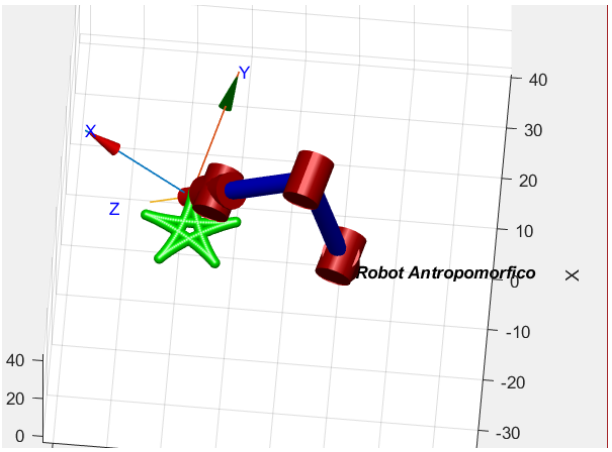


Figura 2.33. Simulación con brazo

Una vez acondicionado todos los motores, conocido los diferentes softwares que se van a utilizar y su funcionamiento, y realizadas las simulaciones, el siguiente capítulo constará de las pruebas y resultados obtenidos al implementar el trazado de figuras en el robot antropomórfico de 5 grados de libertad.

3. RESULTADOS CONCLUSIONES RECOMENDACIONES

Para validar el funcionamiento del presente trabajo de integración curricular se realiza pruebas de funcionamiento, de manera individual probando el movimiento de todos los motores en sus posiciones límites y de manera grupal realizando el trazado de las figuras propuestas.

3.1 PRUEBAS

En esta sección se procede a realiza pruebas de cada uno de los motores y de manera conjunta, con el fin de realizar el trazado de las figuras propuestas.

3.1.1 PRUEBAS INDIVIDUALES

Se procede a realizar las pruebas para todos motores de manera individual con ayuda del software Arduino, esto tiene como objetivo verificar que la configuración de los parámetros realizada ya sea con ayuda del módulo USB2-Dynamixel para el caso de los motores Dynamixel o con ayuda del software “Jrk Configuration Utility” para los motores Torxis es la correcta.

- Motor Dynamixel MX-28

En este caso se trata del último grado de libertad teniendo en cuenta que en este se colocará el elemento final, en este caso es un marcador el cual se encarga de realizar el trazado de las figuras, este motor no posee restricciones físicas por lo tanto se puede mover en todo su rango.

- Motor Dynamixel MX-64

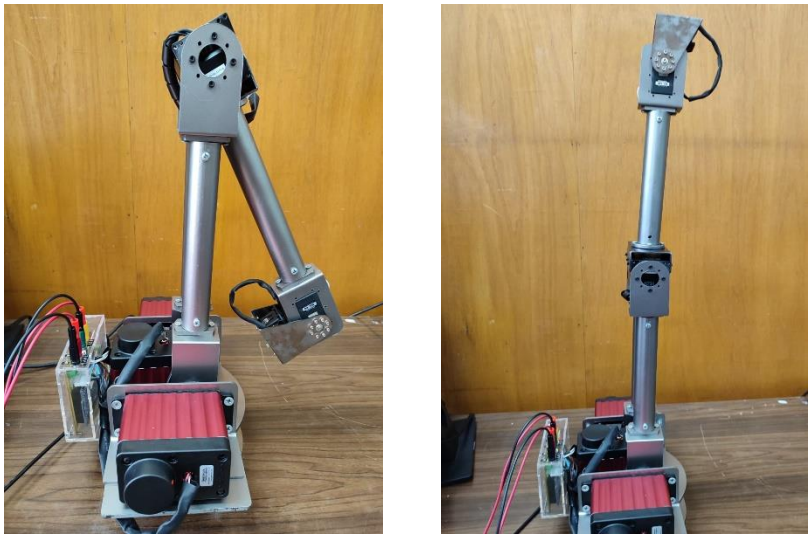
Si se recuerda el Tomo I, el motor MX-64 se ubica en la muñeca en el cuarto grado de libertad y debido a sus limitaciones físicas sus movimientos se limitan a 180°, en la Figura 3.1 se puede observar las dos posiciones límites que posee esta articulación.

- Motor Dynamixel PRO

En este caso se prueba el motor de la tercera articulación la cual posee límites físicos en su parte inferior, mientras que se limita su posición superior debido a que tener un rango más amplio no ayuda al propósito de este proyecto, sus posiciones límites son de 0° a 150° esto se observa en la Figura 3.2.

a) b)

Figura 3.1. Límites de posición de motor Dynamixel MX-64, a) límite inferior, b) límite superior

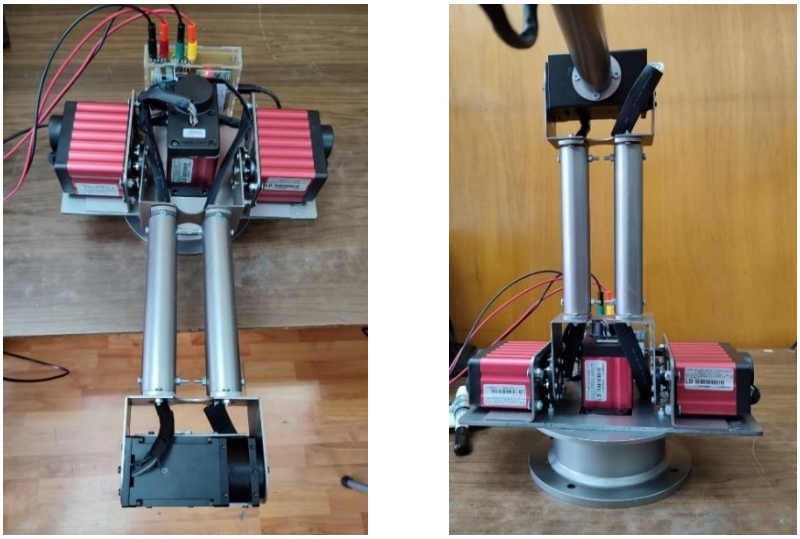


a) b)

Figura 3.2. Límites de posición de motor Dynamixel PRO, a) límite inferior, b) límite superior

- Motor Torxis, elevación

En este caso se menciona los dos motores Torxis ubicados en los laterales para elevación y descenso del brazo completo, en este caso los límites son físicos ya que la forma constructiva del brazo no permite un movimiento más amplio, siendo su rango de 0° a 90°.

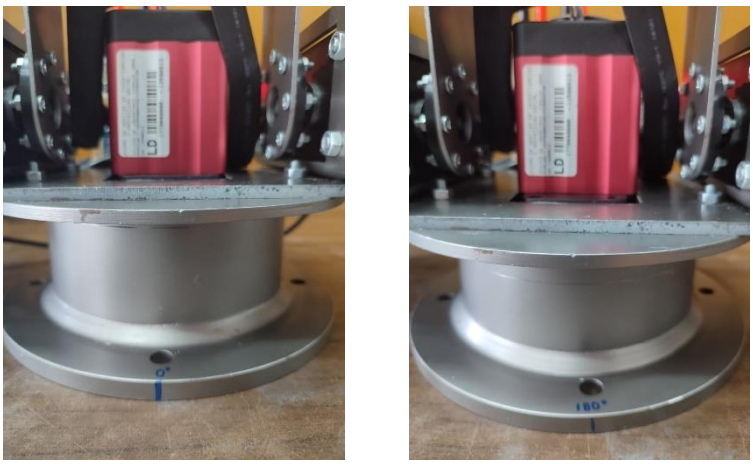


a) b)

Figura 3.3. Límites de posición de los motores Torxis de elevación, a) límite inferior, b) límite superior

- Motor Torxis, giro de la base

Finalmente, para la primera articulación se tiene una rotación de 0° a 180°, físicamente si puede moverse los 360°, sin embargo, se limitó este rango para el propósito de este proyecto, en la Figura 3.4 se puede observar el posicionamiento de la base y por lo tanto de todo el brazo. Para representar su posición se realizó marcas en la parte estática de la base.



a) b)

Figura 3.4. Límites de posición del motor Torxis de la base, a) posición en 0°, b) posición en 180°

3.1.2 PRUEBAS GRUPALES

En esta sección ya se realiza la validación de las figuras trazadas, en cada caso se compara la figura realizada por el brazo en contraste con un patrón de la misma figura impresa, de esta manera se puede observar en que puntos difiere del patrón.

Para realizar el trazado de figuras se debe seguir el siguiente procedimiento:

1) Energizar el sistema, las fuentes de alimentación de 12 y 24V se encuentran debidamente identificadas en la carcasa (Figura 3.5), adicional a ello, se debe tener en cuenta el amperaje requerido para cada una de las fuentes tal como se detalla en el Tomo I del presente Trabajo de Integración Curricular.

Figura 3.5. Borneras de alimentación

2) Se debe conectar la salida de comunicación del módulo Arduino a la PC, el puerto de salida se encuentra en el lado izquierdo de la misma carcasa (Figura 3.6).

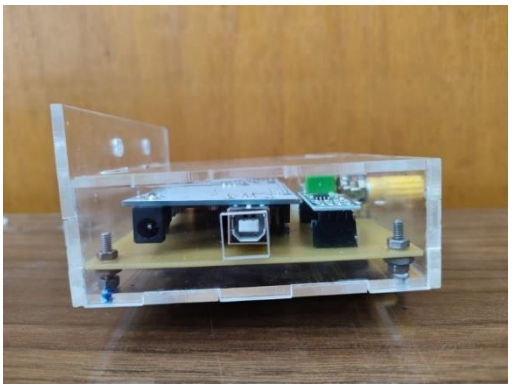


Figura 3.6. Puerto de Comunicación

3) Ejecutar la memoria compartida creada en el software Visual Studio 2012, dentro de ella se solicita ingresar el puerto de comunicación al cual se encuentra conectado el Arduino Mega, (Figura 3.7). Una vez ingresado se presiona “Enter”.

Una vez finalizado estos pasos se muestra un mensaje que indica que Matlab-Simulink debe estar en ejecución antes de poder continuar, esto se puede observar en la Figura 3.7, es por lo que en este punto el HMI se debe ejecutar.

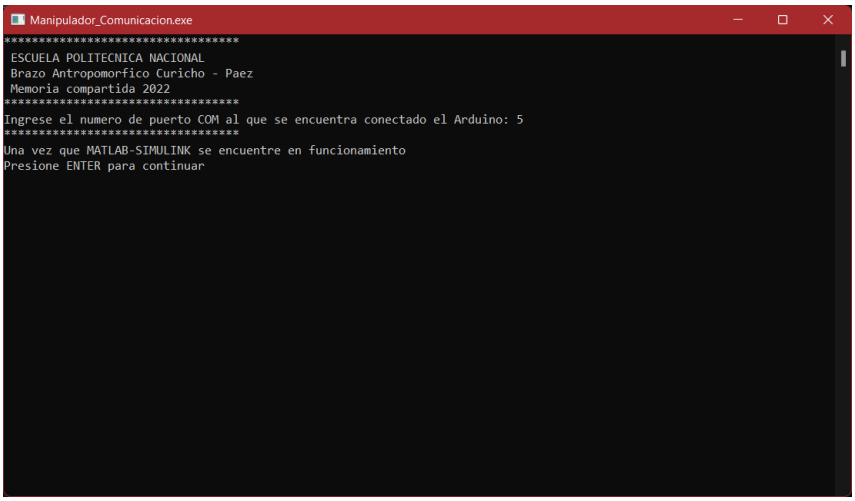


Figura 3.7. Instrucción de ejecución de Simulink

Para saber que el programa de Simulink se encuentra listo en la parte inferior el tiempo de simulación debe empezar a contar, además, en el display correspondiente a “Valores Teóricos” se muestra los ángulos de posición inicial del brazo, como se aprecia en la Figura 3.8.

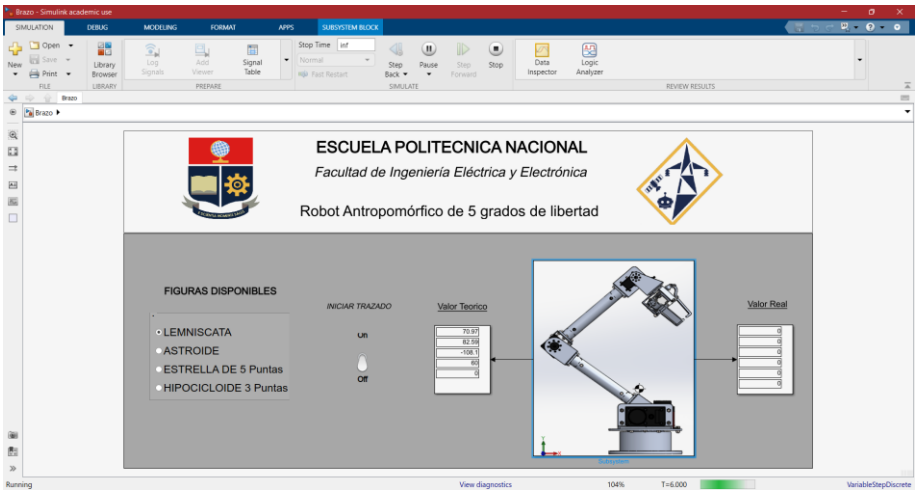


Figura 3.8. Simulación HMI

4) Finalmente, para terminar de enlazar la comunicación entre Arduino Mega y la PC es necesario regresar al ejecutable de la memoria compartida en Visual Studio y presionar la tecla “Enter”. Si los pasos previos fueron correctos la pantalla del ejecutable se visualizará como la Figura 3.9, en la cual se deben mostrar los ángulos recibidos y enviados desde y hacia Matlab.

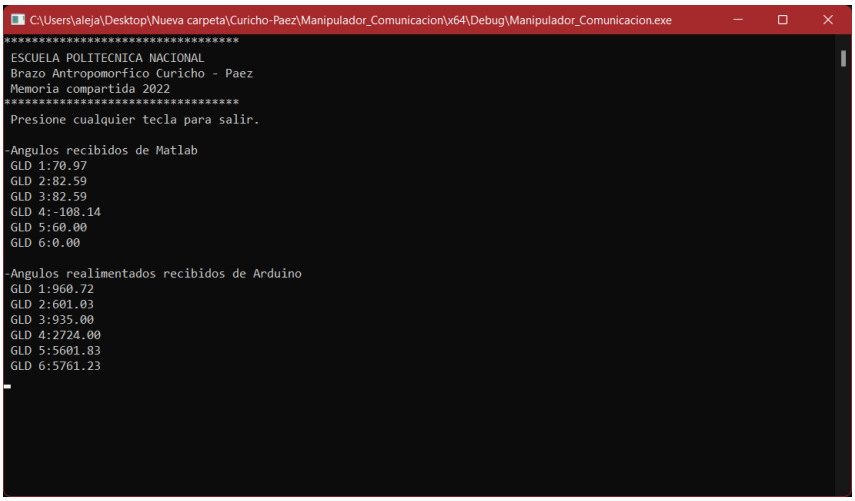


Figura 3.9 Pantalla del ejecutable con la comunicación enlazada con éxito

Para más detalle sobre la puesta en marcha del sistema se recomienda revisar el Anexo I.

Para realizar el trazado nada más resta elegir la figura deseada del menú de figuras disponibles y por medio del switch iniciar el trazado. La primera prueba realizada se lo hizo con la figura lemniscata. El primer trazado de prueba se puede observar en la Figura 3.10, la misma que se dibuja sobre una pizarra.

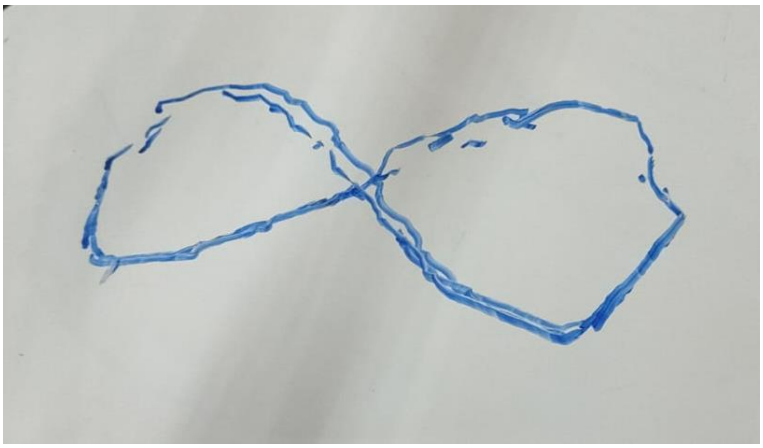


Figura 3.10. Prueba 1 lemniscata

En la Figura 3.10 se puede notar que el trazado no posee ni exactitud ni precisión, tomando en cuenta que las simulaciones fueron realizadas con las mismas ecuaciones y cinemática, es claro que la falla se encuentra en la parte física, específicamente en los motores. Es por ello que se procede a verificar el posicionamiento de cada uno de los motores, en lo referente a los motores de la marca Dynamixel, por medio de su módulo USB2Dynamixel, se pudo notar que su posicionamiento posee gran precisión y exactitud incluso cuando su movimiento se realiza en décimas de grados, por otro lado, los motores de la marca Torxis tenían una variación en su realimentación lo cual generaba inexactitud en su movimiento, por ello se realiza la sintonización del PID de los tres motores, una vez sintonizados estos parámetros se realiza nuevamente el trazado de la misma figura, esta se puede observar en la Figura 3.11.

Figura 3.11. Prueba 2 lemniscata

Si se compara la Figura 3.10 con la Figura 3.11 se puede observar que su repetibilidad mejora considerablemente además de que el trazado se asemeja más a la figura deseada. Cabe mencionar que aún existen fallas en el trazado debido a los motores Torxis, a pesar de que el PID de estos ha sido sintonizado adecuadamente.

Una vez obtenido resultados adecuados con respecto al trazado de figuras se procede a realizar las pruebas con el resto de figuras, para ello se diseñó una plataforma donde se coloca el patrón de referencia en una hoja cuadriculada y sobre esta se dispone de un vidrio, para que el trazado se realice sobre este.

3.1.2.1 Lemniscata

Para la figura de Lemniscata se implementa la Ec (1.1) y la Ec (1.2) en la cinemática inversa de Simulink, para obtener como resultado la figura realizada por el brazo robótico esto se puede notar en la Figura 3.12, en la cual se tiene dicha figura realizada por el brazo en comparación con la realizada por el comando “Plot” de Matlab.

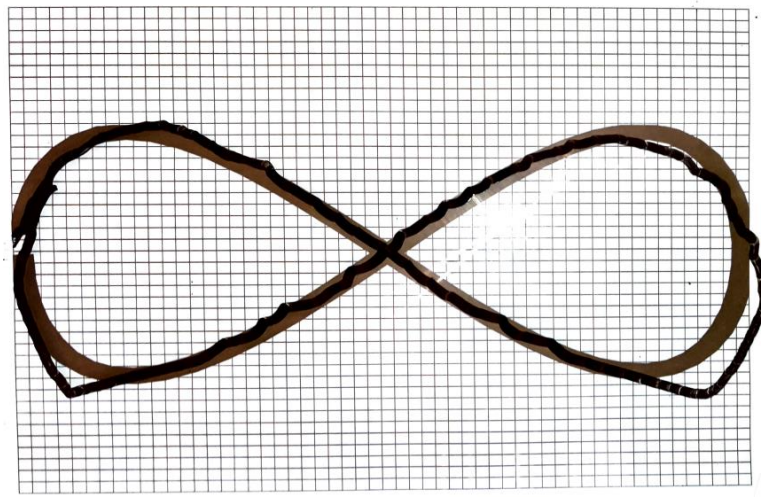


Figura 3.12. Prueba de trazado lemniscata

El tiempo de simulación para esta figura es de 105 segundos, en el trazado de esta figura se puede observar que en la parte inferior, derecha e izquierda, se tiene un desvío en comparación con la referencia esto se debe a un movimiento involuntario realizado por los motores Torxis de la base y elevación, en estos puntos se ha notado que estos motores tienen un inconveniente en seguir la referencia, es decir, las órdenes enviadas por Simulink, ya que como se puede notar en las simulaciones realizadas en la Figura 2.23, la cinemática inversa es correcta por lo que en simulación el brazo robótico sigue la trayecto deseada.

Si se compara la Figura 3.11 con la Figura 3.12, a simple vista se observa que el trazado es diferente, esto se debe a que esta última figura se encuentra sobre el patrón, pues, ambas figuras son las mismas.

3.1.2.2 Astroide o hipocicloide de 4 puntas

Para el trazado del astroide se utiliza la Ec (1.5) y Ec (1.6), al incluirlas para el trazado físico con el brazo robótico da como resultado la Figura 3.13.

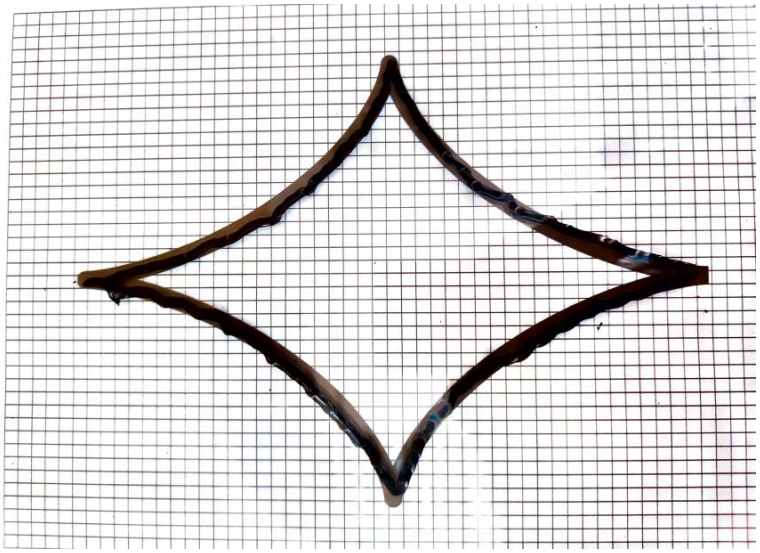


Figura 3.13. Prueba de trazado astroide

Al igual que la Figura 3.12 su tiempo de trazado es de 105 segundos, en este caso se puede observar un mejor resultado ya que tanto la figura de patrón como la realizada por el brazo son muy similares.

3.1.2.3 Hipocicloide de 3 puntas

Para esta figura se utiliza la Ec (1.3) y la Ec (1.4), los resultados obtenidos por parte del brazo robótico se muestran en la Figura 3.14.

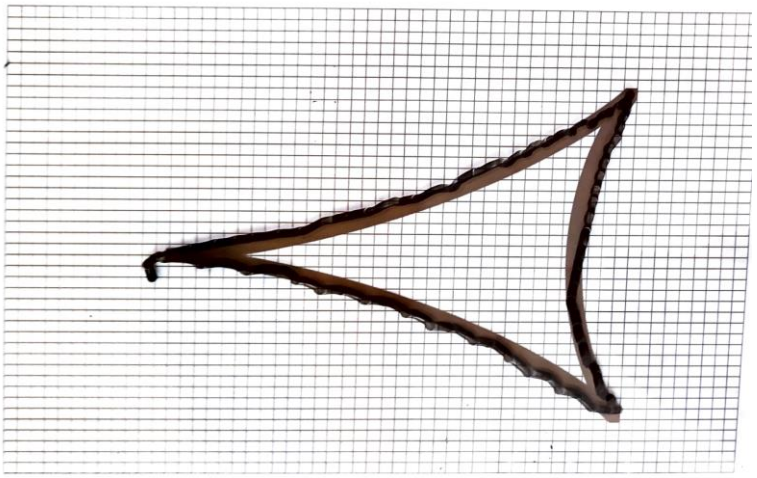


Figura 3.14. Prueba de trazado hipocicloide de 3 puntas

Su tiempo de trazado es de 105 segundos, se puede observar que el trazado de la figura realizada por el brazo en comparación con su patrón es similar, se tiene un pequeño desvío en el lado izquierdo debido a que en este punto es donde se inicia el trazado de la figura de la hipocicloide de 3 puntas.

3.1.2.4 Estrella de 5 puntas

Con ayuda de la Ec (1.7) y la Ec (1.8) se realiza el trazado de la estrella de 5 puntas, esto da como resultado la Figura 3.15 realizada por el brazo.

Su tiempo de simulación es de alrededor de 210 segundos, esto se debe a que posee el doble de puntos para su trazado en comparación con las figuras previas, la falla más grande que se puede notar es en la parte inferior izquierda, en la cual se nota que existe un desfase con la figura de patrón, en esta sección el brazo empieza a tener problemas para llegar a los ángulos propuestos generando esas ondulaciones o gradas.

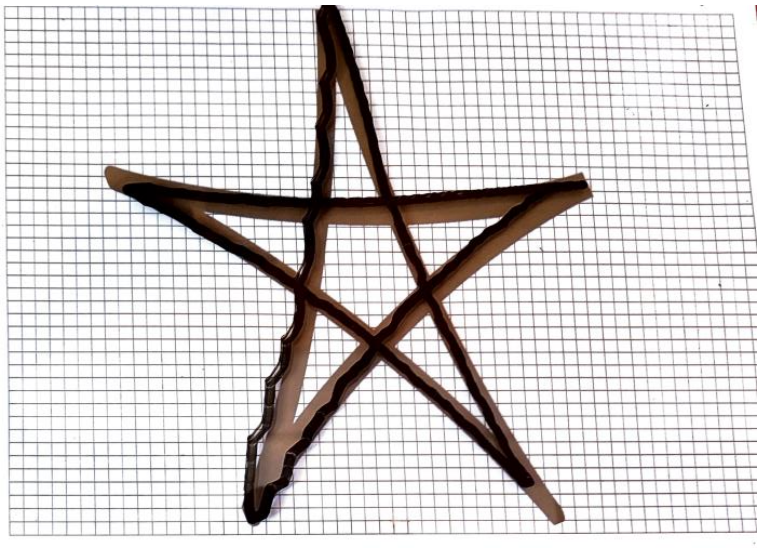


Figura 3.15. Prueba de trazado estrella de 5 puntas

Para el trazado de todas las figuras se utilizó un marcador color negro mientras que el patrón de referencia es de color café, de esta manera se puede notar el contraste entre las figuras, adicionalmente, se debe notar que en todas las figuras existen ondas o gradas en algunos casos es más notorio y en otros es casi imperceptible.

3.2 CONCLUSIONES

- Con el trazado de figuras se debe notar que se ha cumplido con los objetivos planteados del Tomo II de este proyecto de integración curricular, el trazado se lo realiza tanto por simulación como de manera física gracias a la implementación del brazo antropomórfico, sin embargo, cabe mencionar que en lo referente a simulación las figuras son totalmente precisas y exactas, por otro lado, en el trazado físico de las figuras se tiene un cierto porcentaje de error.
- El error que se produce durante el trazado de figuras de manera física puede ser causa de los motores Torxis ubicados en la base, es correcto notar que estos motores poseen un gran torque lo cual le permite ejercer mucha fuerza esto le permite mover y manipular toda la estructura del brazo, en cambio, su exactitud al ubicarse en determinada posición si posee un error de consideración, además que el software encargado de configurar su tarjeta de control y la tarjeta de control en sí, son considerados obsoletos por los mismos fabricantes los cuales a su vez recomiendan su cambio.
- La configuración de los parámetros internos no se tenía previsto, sin embargo, para el caso de los motores Dynamixel MX-28, MX-64 se limitó su movimiento y en el caso del PRO, a más de limitar su movimiento, también se disminuyó su velocidad, esto se lo realizó de una manera relativamente sencilla debido a que el software encargado de esto fue muy amigable y completo.
- En la configuración de los motores Torxis fue un reto lograr mejorar su posicionamiento ya que incluso se realizó modificación del PID y la realimentación, esto se lo hizo de manera heurística, es decir, prueba y error, a pesar de ello, si se logró una mejora en el trazado de figuras, es posible que con un análisis más a fondo y netamente dedicado a la configuración de los parámetros de los motores Torxis se puedan lograr mejores resultados.
- Una vez entendido el funcionamiento de la cinemática inversa de un robot de cinco grados de libertad y gracias a su implementación en líneas de código, la implementación de cualquier figura se vuelve sencilla si se conoce sus puntos en los ejes X, Y y Z.
- La configuración de los PID internos de los motores Torxis no era un objetivo de este trabajo de integración curricular, sin embargo, de no ser modificados estos

parámetros las figuras obtenidas no se asemejaban en nada a las figuras obtenidas por simulación.

3.3 RECOMENDACIONES

- Siempre se debe encender la alimentación de 12 y 24 V antes de conectar la placa Arduino a la PC, de lo contrario la inicialización de los motores no será la correcta y en el caso del motor Dynamixel PRO se quedará estático.
- Verificar que en el área de trabajo del brazo no se encuentre objetos o personas, el sistema no posee sensores externos que detectan la presencia de elementos extraños y por lo tanto no detendrá su movimiento hasta que se le indique por medio de la PC o se apague su alimentación.
- Si se desea realizar el trazado de una nueva figura, es preferible realizar su simulación previa con esto se puede identificar sus puntos de partida, el tamaño que tendrá y que dimensiones tendrá en la implementación física.
- Se recomienda realizar un análisis más a fondo de los motores Torxis, ya sea configurando sus parámetros internos o actualizando su tarjeta de control para mejorar las figuras obtenidas.

4. REFERENCIAS BIBLIOGRÁFICAS

[1] «DataSheet Dynamixel Driver,» Robotics e-Manual, 2018. [En línea]. Available: https://emanual.robotis.com/docs/en/software/dynamixel/dynamixel_sdk/device_setup/. [Último acceso: 01 08 2022].

[2] A. Aceves López, «Instalar USB2Dynamixel,» Robotics CO., LTD., 13 10 2011. [En línea]. Available: https://homepage.cem.itesm.mx/aaceves/Bogobots/seminario/Bioloid.pdf. [Último acceso: 01 08 2022].

[3] Support Pololu, «Pololu Robotics & Electronics,» Pololu, 2009. [En línea]. Available: https://www.pololu.com/product/1392. [Último acceso: 12 08 2022].

[4] Support Pololu, «Pololu Robotics & Electronics,» Pololu, 2014. [En línea]. Available: https://www.pololu.com/docs/0J38/3.a. [Último acceso: 12 08 2022].

[5] J. Rivera Berrío, «Curva de lemniscata,» de *Curvas paramétricas*, Medellín, 2014, p. 20.

[6] Gaussianos, «Astroide, cardioide y demás -oides,» 2011. [En línea]. Available: https://www.gaussianos.com/astroide-cardioide-y-demas-oides/. [Último acceso: 09 08 2022].

5. ANEXOS

Anexo I: Manual de Usuario

Anexo II: Programa de Simulación

ANEXO I

MANUAL DEL USUARIO

I.1 Introducción

El brazo antropomórfico de 5 grados de libertad al cual pertenece el presente manual de usuario está diseñado para realizar el trazado de figuras por medio del control de sus articulaciones utilizando cinemática inversa, es decir, se utilizan las ecuaciones paramétricas de las diferentes figuras predeterminadas para transformar esas coordenadas de X, Y y Z en ángulos que sirven para comandar el movimiento de los motores que conforman el equipo.

I.2 Objetivo

Este manual de usuario está destinado para guiar al operador del brazo antropomórfico desde la conexión de su alimentación, pasando por el funcionamiento de los programas que son utilizados, hasta el trazado de figuras, con ello se pretende evitar cualquier daño tanto para el equipo, así como para el operador, de tal manera que el usuario pueda aprovechar al máximo las ventajas que posee el brazo antropomórfico de 5 grados de libertad.

I.3 Características técnicas del robot

Toda su estructura fue creada en acero de transmisión, en posición vertical alcanza una altura máxima de 83.5 cm, y su peso aproximado de 20 Kg, consta de seis motores: tres motores Torxis i00600 colocados en la base, un motor Dynamixel PRO ubicado en el codo, un motor Dynamixel MX-64 en la muñeca y un motor Dynamixel MX-28 para el giro de la pinza.

Figura I.1. Brazo antropomórfico de 5 grados de libertad.

- **Características eléctricas**
  - Alimentación: 12V-3A y 24V-2A
  - Comunicación: Serial con cable USB a Arduino mega

I.4 Requisitos

- **Software**
  - Matlab-Simulink con versión R2020b o superior
  - Arduino con versión 1.8.15 o superior
  - Visual Studio con versión 2012 o superior

- **Hardware**
  - Fuente de alimentación 12 V y 3 A
  - Fuente de alimentación 24 V y 2 A
  - Cable USB para Arduino mega
  - Cables de conexión banana-lagarto

I.5 Funcionamiento

1) Conexión.

El orden de conexión de los elementos es importante ya que de esto dependerá que todos los motores se muevan correctamente, lo primero es conectar las fuentes de alimentación, cada bornera se encuentra debidamente identificada para 12 o 24 V y la polaridad se determina por “A” bornera positiva y “B” bornera negativa Figura I.2. Una vez que se verifico que los voltajes y polaridad son los correctos se procede a encender las fuentes de alimentación.

Figura I.2. Borneras de alimentación

A continuación, se conecta la conexión del módulo Arduino a la PC, el puerto de salida se encuentra en el lado izquierdo de la carcasa, Figura I.3.

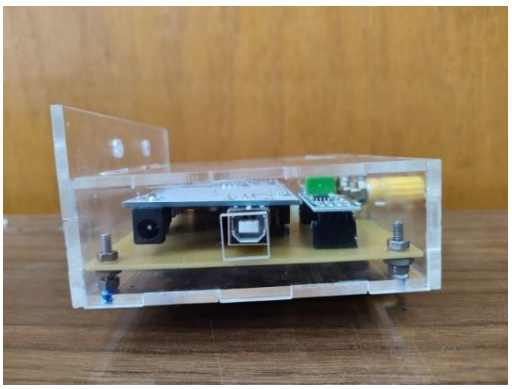


Figura I.3. Puerto de Comunicación

2) Memoria compartida

Uno de los archivos que se encuentra en la carpeta del proyecto es “Manipulador_Comunicacion.exe” se trata de un ejecutable creado en Visual Studio 2012, al abrirlo se presentará la pantalla mostrada en a Figura I.4. En ella se pide colocar el

número del puerto de comunicación al cual se encuentra conectado el Arduino Mega, si no se conoce, se puede consultar por medio del “Administrador de dispositivos” de Windows en el apartado “Puertos”. Una vez ingresado el número se presiona “Enter”.

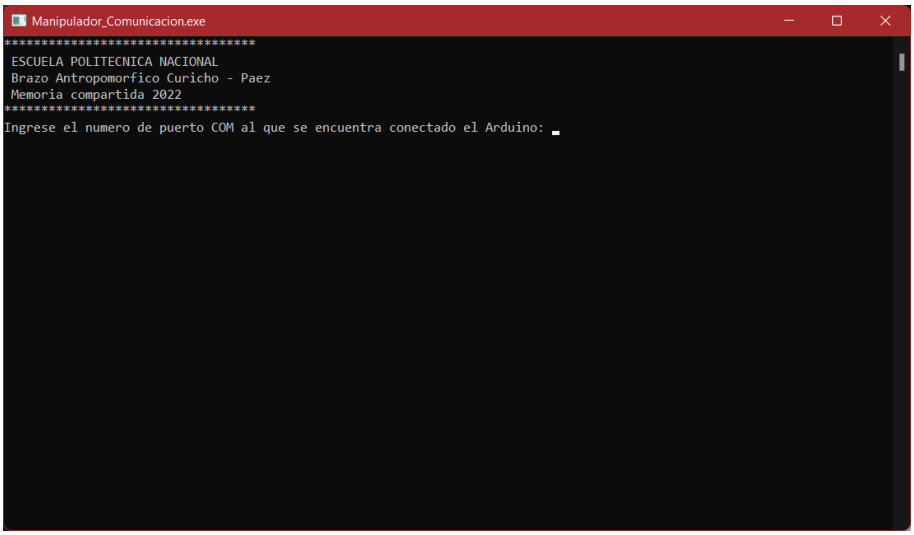


Figura I.4. Pantalla inicial

A continuación, desplegará un mensaje que indica que Matlab-Simulink debe estar en funcionamiento para continuar, como se ve en la Figura I.5. Antes de continuar con el programa “Manipulador_Comunicación.exe” es necesario abrir Simulink.

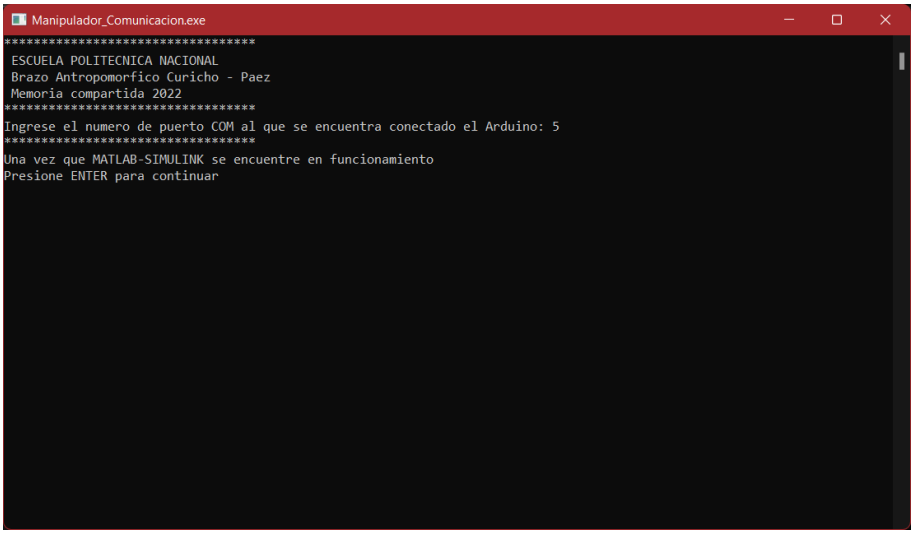


Figura I.5. Instrucción número dos

3) Matlab-Simulink

El siguiente programa que se utilizará es Simulink, de igual manera en la carpeta del proyecto se encuentra el archivo llamado “Brazo” el cual al abrirlo se muestra la interfaz de usuario, Figura I.6.

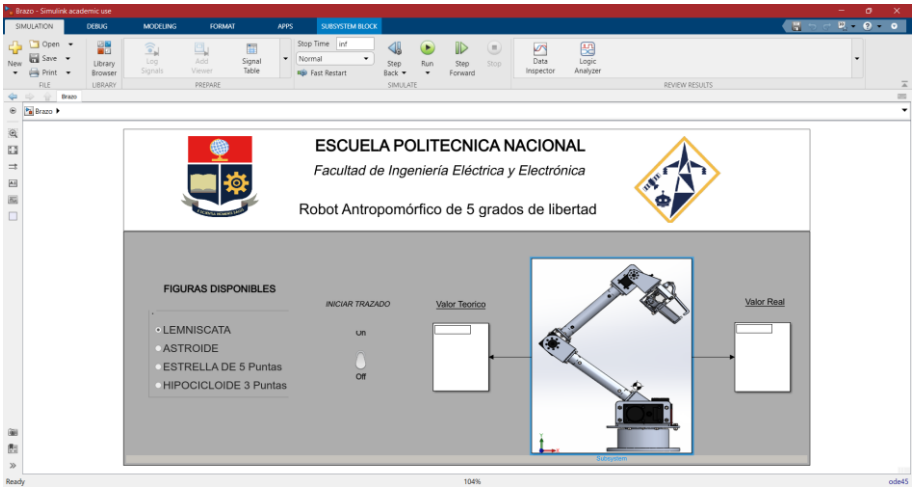


Figura I.6. Interfaz gráfica de usuario

Únicamente resta correr la simulación por medio del pulsador verde que se encuentra en la parte superior, dependiendo de las características del computador el iniciar la simulación puede tardar unos instantes. Se puede dar cuenta que el programa está listo cuando en la parte inferior el tiempo de simulación empieza a contar, además en el display correspondiente a los “Valores Teóricos” se muestra los ángulos de la posición iniciar del brazo. Figura I.7.

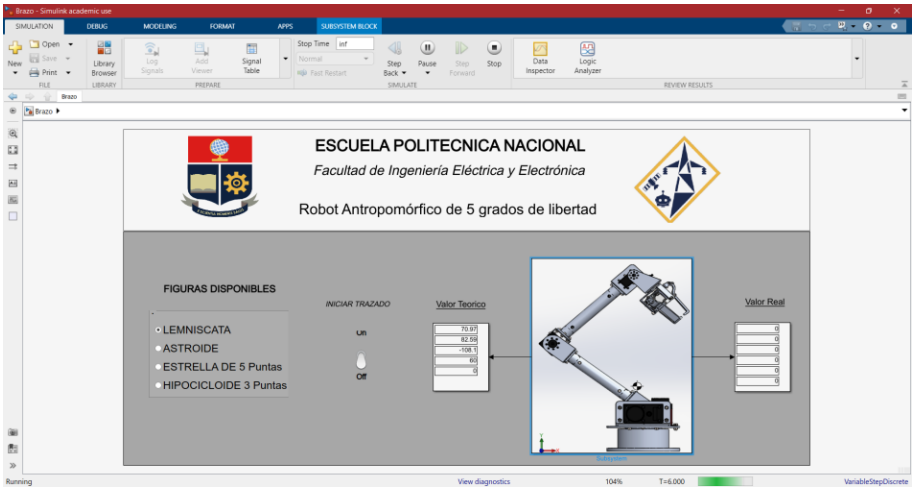


Figura I.7. Simulación interfaz gráfica de usuario

Para terminar de enlazar la comunicación entre el Arduino Mega y la PC se regresa al ejecutable “Manipulador_Comunicación.exe” y se presiona la tecla “Enter”. Si los pasos previos fueron realizados de la manera correcta la pantalla del ejecutable se verá como la Figura I.8.

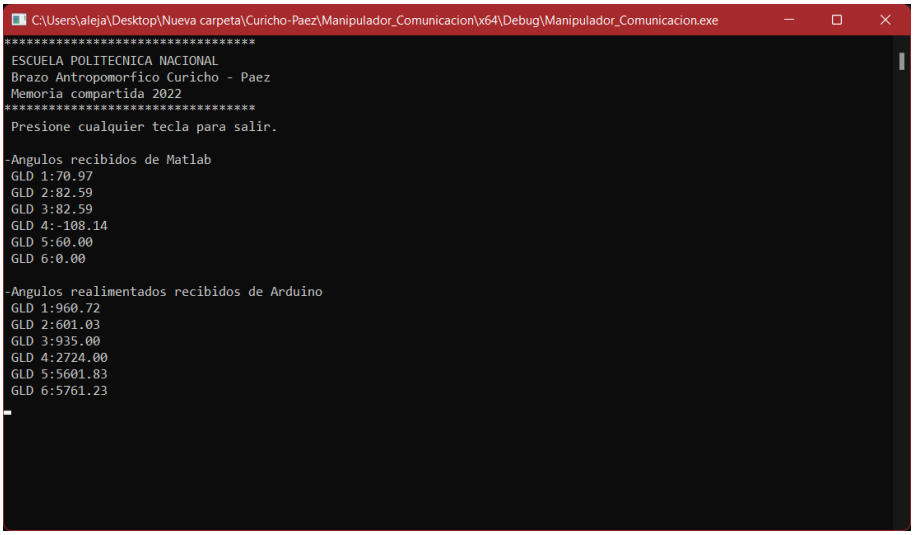


Figura I.8 Pantalla del ejecutable con la comunicación enlazada con éxito.

Cabe mencionar que una vez enlazada la comunicación el brazo se moverá a su posición inicial, por lo cual de aquí en adelante se debe tener especial cuidado de no entrar en el espacio de trabajo del brazo.

4) Trazado de figuras

Primero se debe seleccionar la figura deseada en el menú de “Figuras Disponibles”, a continuación, se cambia el switch de “Iniciar Trazado” a ON para que inicie con el trazado de figuras. El brazo iniciará su movimiento. Los ángulos enviados desde Matlab y recibidos del Arduino se pueden visualizar tanto en Simulink como en el ejecutable “Manipulador_Comunicación.exe”.

5) Detener el movimiento

Sin importar la ubicación en la cual se encuentre el brazo o si se encuentra trazando figuras, al cambiar el switch de “Iniciar Trazado” a OFF el brazo regresará a su posición inicial.

6) Cerrar los programas

Para cerrar todos los programas se lo puede realizar en sentido inverso a como fueron abiertos, es decir, primero debe detener la simulación en Simulink, luego, en el ejecutable “Manipulador_Comunicación.exe” únicamente resta presionar cualquier tecla para terminar su ejecución y se termina apagando las fuentes de alimentación.

I.6 Precauciones

El brazo antropomórfico posee una estructura sumamente sólida, y los motores, en especial de la base, tiene un torque elevado por lo cual el equipo puede causar daño al operador y cualquier usuario que se encuentre en el área de trabajo. En el caso de producirse algún incidente se puede recurrir a apagar directamente las fuentes de alimentación, en especial la fuente de 12 V que controla los motores Torxis que se encuentran en la base.

I.7 Errores que pueden producirse

- Si los motores no se mueven:

Este error puede producirse por dos razones, la primera se debe a que el puerto de comunicación al cual se encuentra conectado el módulo de Arduino no es el correcto al ingresarlo en el numeral 2 de la sección I.5 se debe cerrar los programas e iniciar nuevamente para cambiar el puerto de comunicación. Otro motivo por el cual sucede esto es debido a que el cable de comunicación que se conecta al puerto del Arduino no se encuentra bien colocado, asegúrese de que ingreso totalmente.

- El motor Dynamixel PRO del codo no se mueve:

Esto sucede cuando se conecta el módulo Arduino a la computadora antes que encender las fuentes de alimentación.

I.8 Recomendaciones

- Antes de la puesta en marcha verifique que no se encuentre ningún objeto en el espacio de trabajo del brazo.
- Para el trazado de figuras asegúrese de que la pinza se encuentre sujetando con fuerza el elemento final con el cual se va a dibujar.

ANEXO II

PROGRAMA DE SIMULACIÓN

II.1 Introducción

Se proporciona al usuario y operador del brazo antropomórfico un programa realizado en Matlab para que pueda probar y verificar nuevas figuras que posteriormente puedan ser implementadas de manera física por brazo ya que la cinemática utilizada para la simulación es la misma utilizada para el brazo, es decir, los mismos movimientos realizados en simulación se realizarán en físico.

II.2 Objetivos

El propósito del programa de simulación es ayudar al usuario a probar nuevas figuras de una manera segura, probar el posicionamiento de las figuras, desplazarlas, escalarlas, entre otros.

II.3 Funcionamiento

Dentro de la carpeta del presente proyecto se encontrará una subcarpeta llamada “Simulación”, dentro de ella a su vez tendrá una carpeta llamada “rvctools” y un archivo de Matlab llamado “RobotManipuladorSimulacion”. El primero no es más que la librería para que el segundo archivo pueda ejecutarse.

- Librería: rvctools

Al abrir esta carpeta se mostrará los elementos que se pueden observar en la Figura II.1, de lo cual nos centraremos en el archivo “startup_rvc” el cual es un archivo de Matlab.

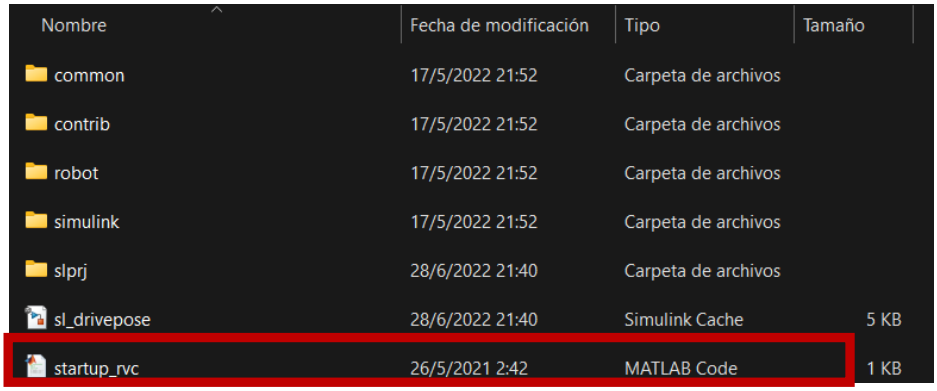


Figura II.1 Carpeta rvctools

Una vez abierto se mostrarán líneas de código pertenecientes a este programa, que para nuestros fines no son relevantes, lo que si es necesario notar es que se indica que en versiones de Matlab inferiores a 7.0 el programa no podrá ejecutarse. Al ejecutar el programa en el "Command Window" se mostrará lo indicado en la Figura II.2, indicando que la librería ya se encuentra disponible.

Figura II.2 Ejecución de startup_rvc

En la última línea de código de la Figura II.2 se informa que al digitar el comando "rtbdemo" se podrá explorar todas las herramientas disponibles en la librería. Al ejecutar el comando se mostrará una ventana como la Figura II.3. Todas las opciones mostradas en esa ventana no corresponden al alcance del presente documento, sin embargo, se recomienda su revisión por parte del lector.

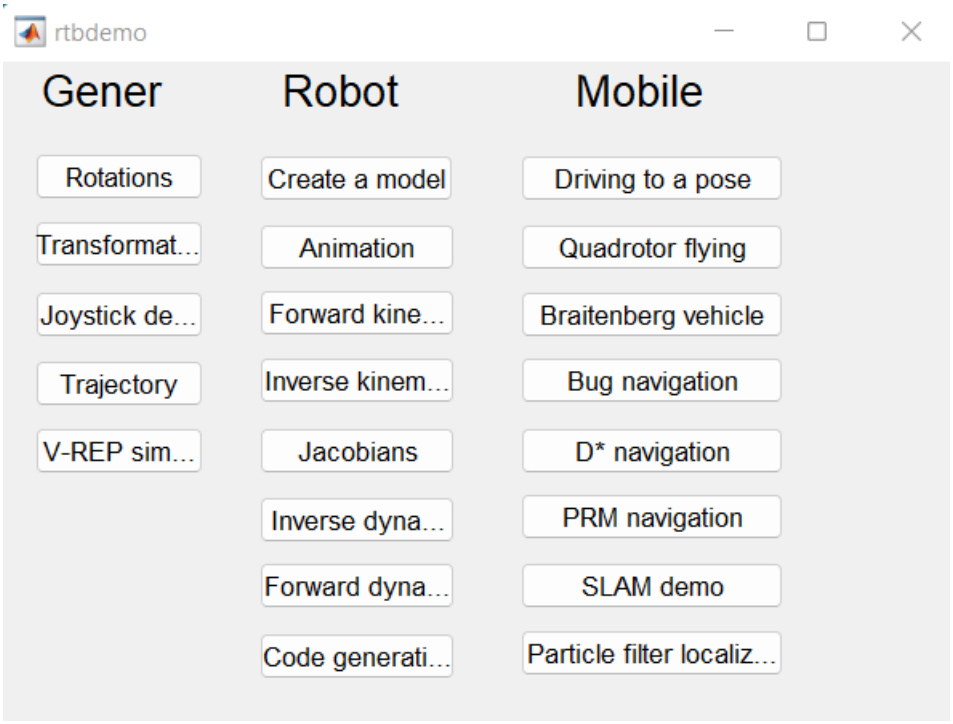


Figura II.3 Ejecución de comando rtbdemo

- Programa

Una vez ejecutado el programa "startup_rvc" se debe abrir el programa "RobotManipuladorSimulacion", dentro de este hay que centrarse en las líneas de código que se debe modificar para graficar cualquier figura y la forma en la que se debe realizar. Lo primero es conocer las ecuaciones paramétricas de la figura deseada en función de X, Y y Z y sus límites. Para el presente documento se mostrará como ejemplo la simulación de un círculo por lo cual sus ecuaciones y límites se muestran en la Figura II.4.

```
%Límites y número de puntos
t=0:0.05:2*pi;
%Ingrese la figura en sus variables X,Y,Z
 X=10*sin(t)+25;
 Y=10*cos(t)+25;
 Z=10;
```

Figura II.4 Ejemplo de ecuaciones para una circunferencia

En la Figura II.4 se puede ver que los límites son de 0 a 2π en pasos de 0.05, y la gráfica por como se expresa variará en los ejes X y Y manteniendo Z constante. Posteriormente se dispone de un lazo “for” aquí es muy importante tomar en cuenta la cantidad de pasos que se dispuso en los límites, si se tiene más pasos se debe incrementar la dimensión de este lazo, esto se conoce por medio de la división del límite máximo, en este caso 2π, entre el valor de cada paso, 0.05, lo cual da 126, este será el límite máximo del lazo “for”, dentro de este lazo se tiene la cinemática del brazo, al igual que los comandos para graficar la trayectoria que sigue.

Finalmente, resta correr el programa y si todos los valores fueron configurados correctamente se abrirá una ventana con el movimiento del brazo punto a punto, en la Figura II.5 se muestra la figura completada.

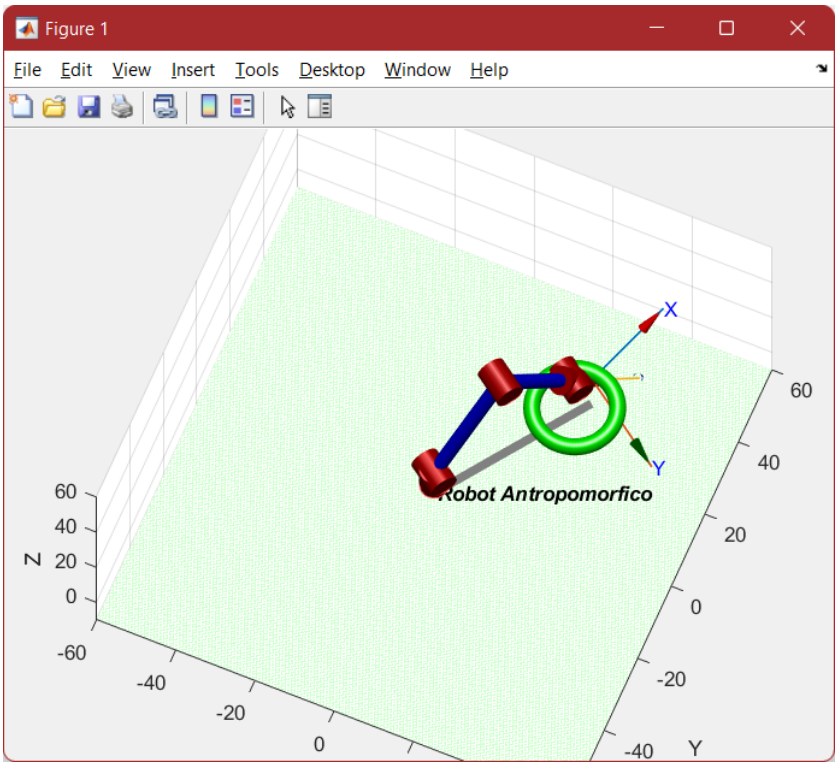


Figura II.5 Figura de ejemplo realizada.