Universidad Nacional Autónoma de México

Facultad de Ingeniería

Carrera: Ingeniería Mecatrónica

Asignatura: Robótica

Documentación: robot con 5 grados de libertad.

Profesor: Ing. Erik Peña Medina

Integrantes:

- Hernández Chagoyán, Vanessa Mariana
- Lawrence Ramírez David Daniel
- García Zamora Yael Sebastián

Semestre 2024-1

Contenido

Introducción

Este proyecto se centra en dar seguimiento a los progresos de iniciativas previas en el ámbito de la Robótica, específicamente en el contexto de la materia impartida en la Facultad de Ingeniería. La tarea principal consiste en desarrollar un robot con forma de garra, diseñado con el propósito de capturar un objeto en un punto "A" y transportarlo hacia un punto "B". Sin embargo, diversos desafíos, como la magnitud del proyecto, la limitación de tiempo del semestre y la elección de una plataforma para la creación de una interfaz gráfica, han convertido este proyecto en un reto significativo para cada equipo participante.

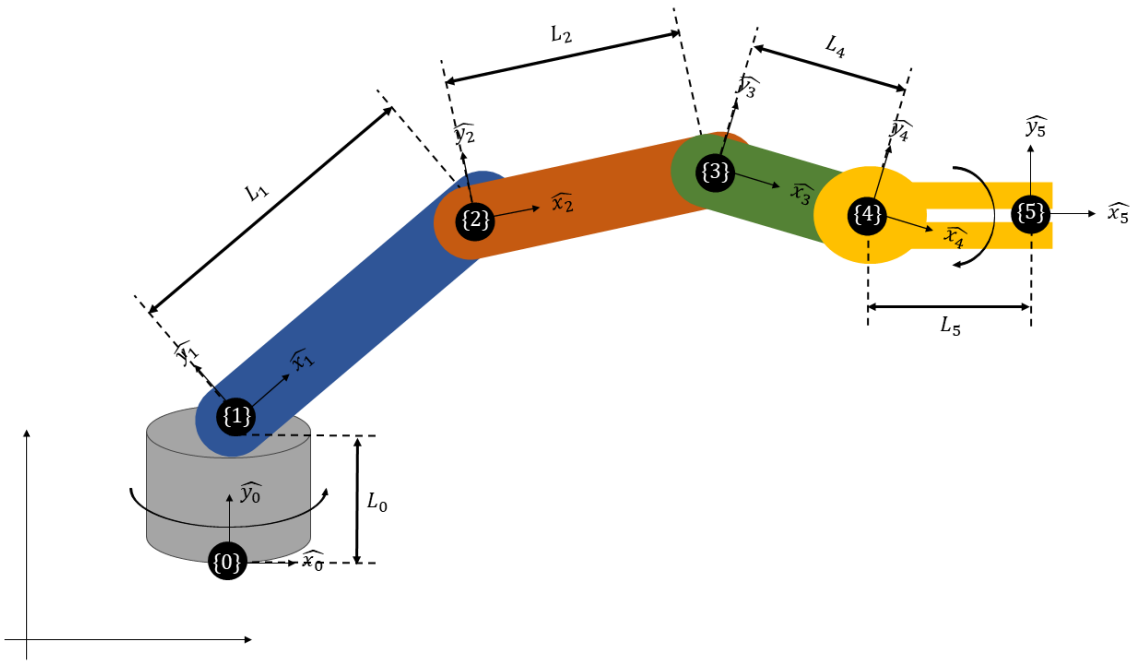
Este informe se enfocará en dar continuidad a la implementación de una interfaz gráfica destinada a controlar un servomotor. El objetivo es regular la dinámica del servo aprovechando la librería "Simulink Support Package for Arduino Hardware". Además, se abordará el desarrollo de un ensamblaje del robot simulado en Fusión como parte integral de este proceso.

Objetivos

- Planteamiento de un modelo cinemático directo e inverso de la postura del robot.
- Generación del modelo virtual del robot.
- Diseño e implementación de una interfaz de control de los motores mediante una interfaz gráfica.
- Implementación de una interfaz gráfica de programación del robot para adoptar posturas para la realización de tareas.

Análisis cinemático

El análisis cinemático parte del siguiente diagrama de un robot 5R:



Cinemática directa:

Lo primero que realizamos es un modelo de posición en base a la tabla de grados de libertad

i	ai	di	αi	θi
1	0	L1	π/2	θ1
2	L2	0	0	θ2
3	L3	0	0	θ3
4	0	0	π/2	θ4
5	0	L4 + L5	0	θ5

En base a la tabla podemos realizar las transformadas homogéneas asociadas a cada sistema de referencia

Transformada homogénea del primer eslabón

H1 =

$$\begin{pmatrix} \cos(\theta_1) & 0 & \sin(\theta_1) & 0 \\ \sin(\theta_1) & 0 & -\cos(\theta_1) & 0 \\ 0 & 1 & 0 & L_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Transformada homogénea del segundo eslabón

H2 =

$$\begin{pmatrix} \cos(\theta_2) & -\sin(\theta_2) & 0 & L_2\cos(\theta_2) \\ \sin(\theta_2) & \cos(\theta_2) & 0 & L_2\sin(\theta_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Transformada homogénea del tercer eslabón

H3 =

$$\begin{pmatrix} \cos(\theta_3) & -\sin(\theta_3) & 0 & L_3\cos(\theta_3) \\ \sin(\theta_3) & \cos(\theta_3) & 0 & L_3\sin(\theta_3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Transformada homogénea del cuarto eslabón

H4 =

$$\begin{pmatrix} \cos(\theta_4) & 0 & \sin(\theta_4) & 0 \\ \sin(\theta_4) & 0 & -\cos(\theta_4) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Transformada homogénea del quinto eslabón

H5 =

$$\begin{pmatrix} \cos(\theta_3) & -\sin(\theta_3) & 0 & 0 \\ \sin(\theta_3) & \cos(\theta_3) & 0 & 0 \\ 0 & 0 & 1 & L_4 + L_5 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Multiplicación de todas las transformadas

H0_5 =

$$\begin{pmatrix} \sin(\theta_1)\sin(\theta_3) + \sigma_1\cos(\theta_1)\cos(\theta_3) & \cos(\theta_3)\sin(\theta_1) - \sigma_1\cos(\theta_1)\sin(\theta_3) & \sigma_4\cos(\theta_1) & \cos(\theta_1)\,\sigma_3 \\ \sigma_1\cos(\theta_3)\sin(\theta_1) - \cos(\theta_1)\sin(\theta_3) & -\cos(\theta_1)\cos(\theta_3) - \sigma_1\sin(\theta_1)\sin(\theta_3) & \sigma_4\sin(\theta_1) & \sin(\theta_1)\,\sigma_3 \\ \frac{\sin(\sigma_2)}{2} + \frac{\sin(\theta_2+\theta_4)}{2} & \frac{\cos(\sigma_2)}{2} - \frac{\cos(\theta_2+\theta_4)}{2} & -\sigma_1 & L_1 + L_3\sin(\theta_2+\theta_3) + L_2\sin(\theta_2) - L_4\,\sigma_1 - L_5\,\sigma_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

where

$$\sigma_1 = \cos(\theta_2 + \theta_3 + \theta_4)$$

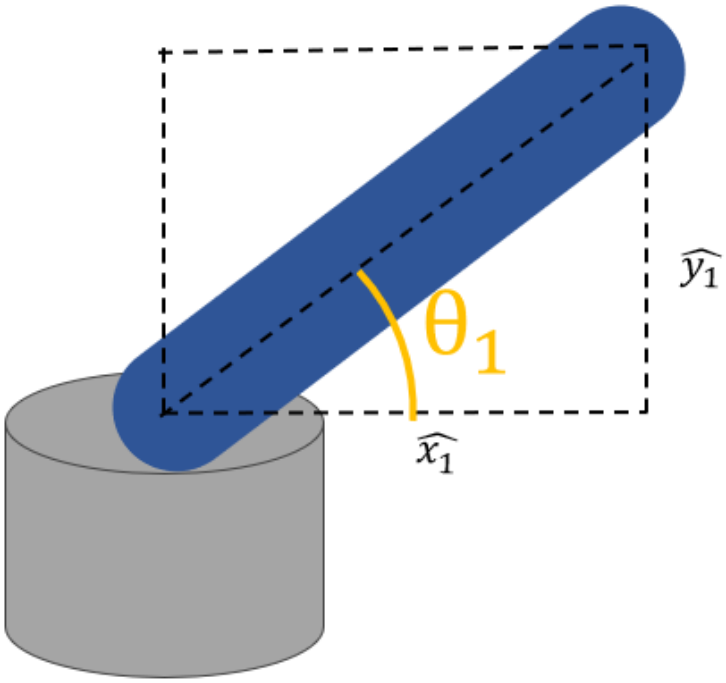
$$\sigma_2 = \theta_2 + 2\,\theta_3 + \theta_4$$

$$\sigma_3 = L_3\cos(\theta_2 + \theta_3) + L_2\cos(\theta_2) + L_4\,\sigma_4 + L_5\,\sigma_4$$

$$\sigma_4 = \sin(\theta_2 + \theta_3 + \theta_4)$$

Más información

1

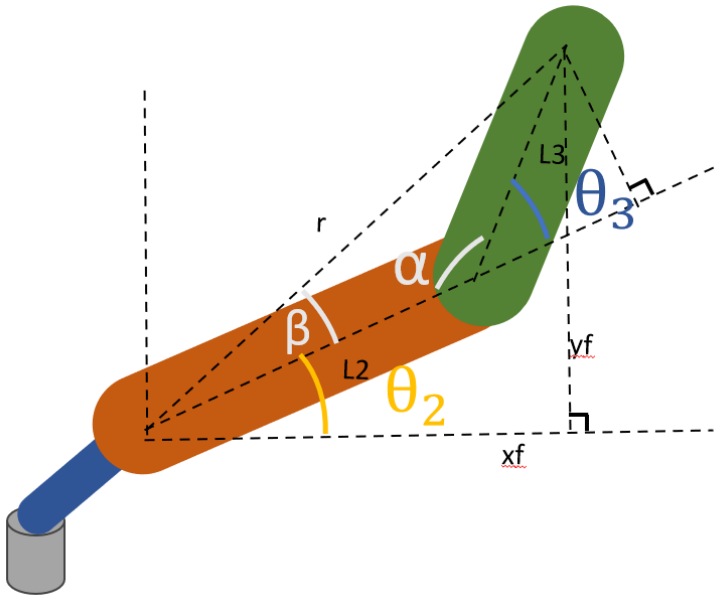


$$\theta 1 = \tan^{-1}(\frac{y1}{x1})$$

2

y

3



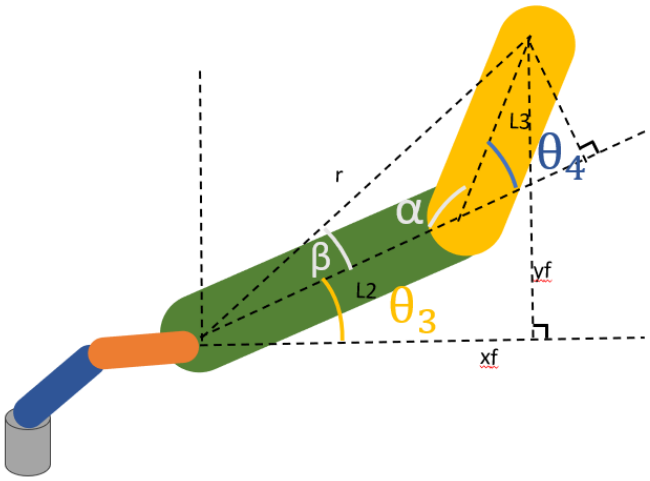
$$\theta 2 = \tan^{-1}\frac{yf}{xf} - \beta$$

$$\beta = \tan^{-1}\frac{L3 * \sin\theta 3}{(L2 + L3) * \cos\theta 3}$$

$$\theta 3 = \tan^{-1}\frac{\sqrt{1 + D^2}}{D}$$

$$D = \frac{L2^2 + L3^2 - r^2 - (zf - h)^2}{2 * L2 * L3}$$

4

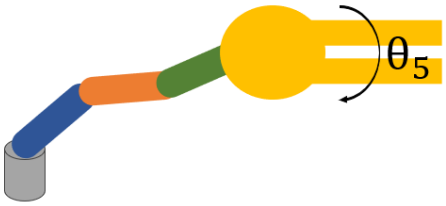


$$\theta 4 = \tan^{-1}\frac{\sqrt{1+DR^2}}{DR}$$

$$D = \frac{L3^2 + L4^2 - {r_R}^2 - (zf-h)^2}{2 * L3 * L4}$$

$$r_R = \sqrt{L3^2 + L4^2}$$

5



$$\theta 5 = \tan^{-1}\frac{z_f}{y_f}$$

Más información

Simulink y Arduino

Conexión Simulink con Arduino

Para poder establecer la conexión de Arduino con el programa en Simulink, se debe de contar con la librería de Arduino en Matlab, la cual se consigue en el entorno de Simulink, en la pestaña de *Apps,* entrando al apartado de *Get Add-ons* y buscando "Simulink Support Package for Arduino Hardware".

Figura 2. 1. Librería para la conexión de Simulink con Arduino.

Para configurar la tarjeta de Arduino que se empleará, hay que entrar a la pestaña de *Simulation*. Para la versión de Matlab R2023a, se debe de entrar a la sección *Prepare* y seleccionar la opción *Model Settings.* En esta ventana, se seleccionó la tarjeta Arduino UNO, pues para este proyecto se empleó dicho microcontrolador.

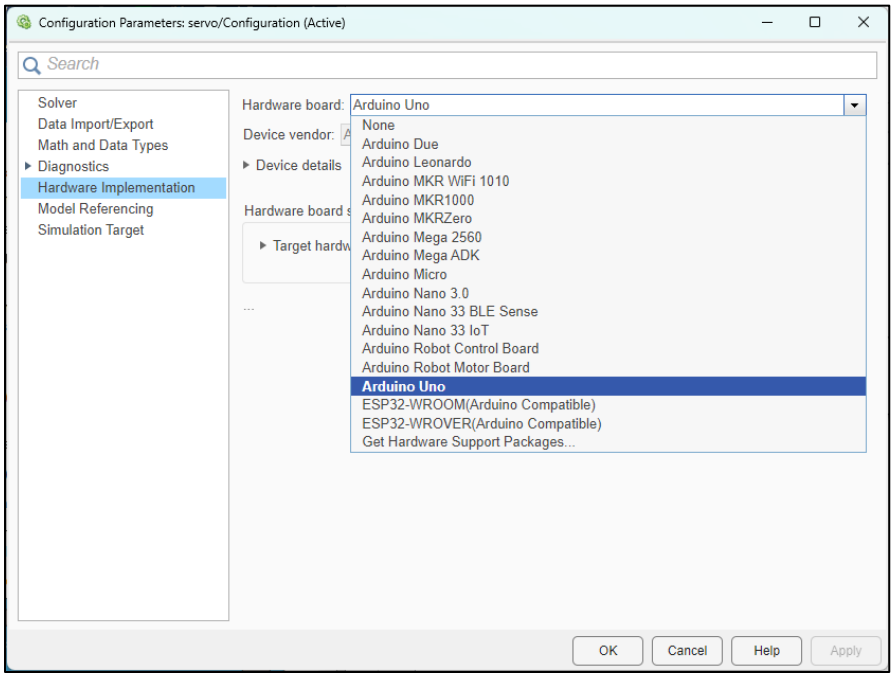


Figura 2. 2. Configuración de la tarjeta de Arduino en Simulink.

La forma más sencilla para manipular un servomotor mediante el entorno de Simulink, se deben seguir los siguientes pasos:

1. Hacer la conexión física del servomotor con el Arduino:

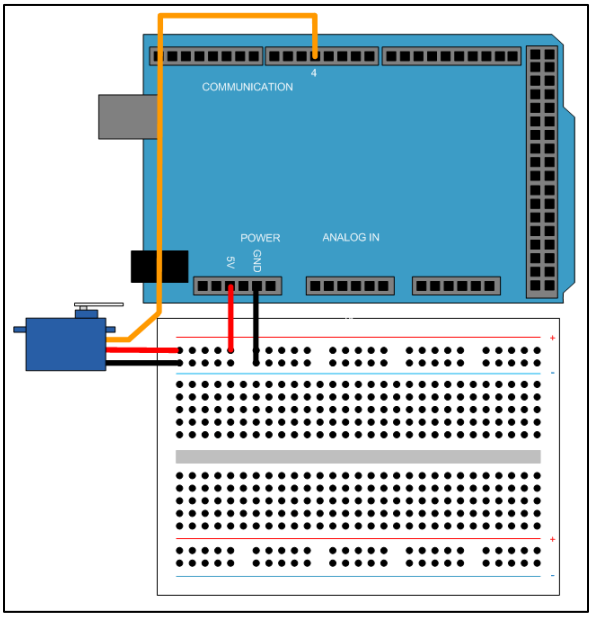


Figura 2. 3. Conexión física del servomotor en una tarjeta de Arduino [1].

2. Hacer la conexión del servomotor en Simulink. Para realizar esto, se debe de asignar una señal de entrada.

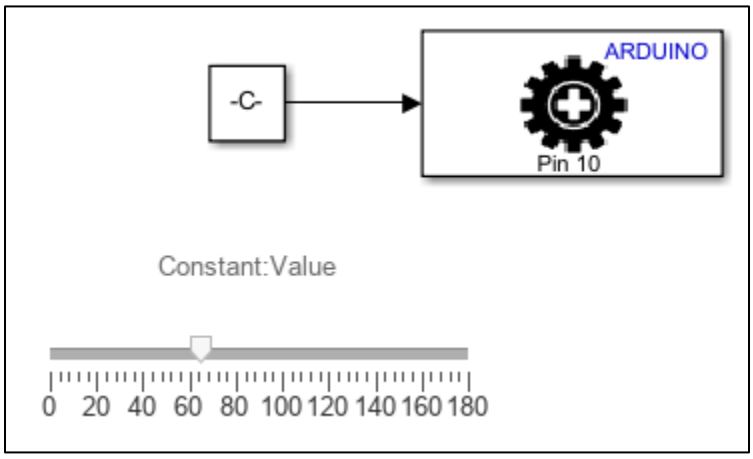


Figura 2. 4. Conexión del Servomotor en Simulink.

Para esta conexión, se configuró el servomotor en el pin digital 10 del Arduino Uno. Se configura el bloque del servomotor "*Standard Servo Write*" de la siguiente forma, para asignarle el pin correspondiente del microcontrolador:

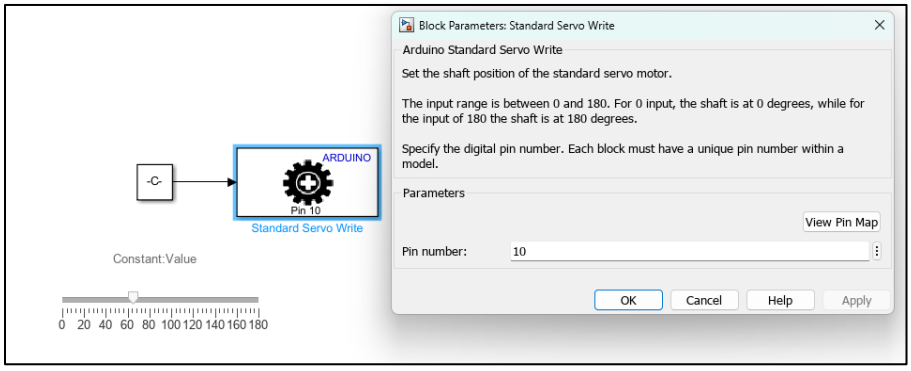


Figura 2. 5. Configuración del bloque "Standard Servo Write".

Para probar la conexión de Arduino Uno con Simulink, la señal de entrada es un *Slider,* es decir, una señal variable controlada por el usuario, y varía de 0 a π radianes.

Primitiva en Simulink

En este proyecto nos referimos con “primitiva” a la simulación que se hace en el entorno de Simulink que, en lugar de tener las proporciones exactas del proyecto físico real, emplea como base, eslabones y pinzas, figuras geométricas simples, como prismas circulares y rectangulares.

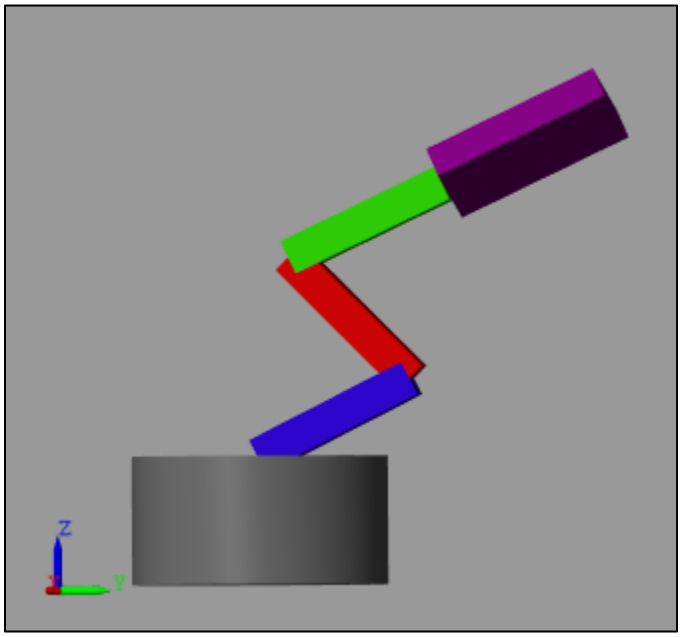
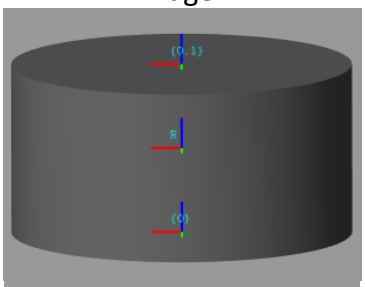
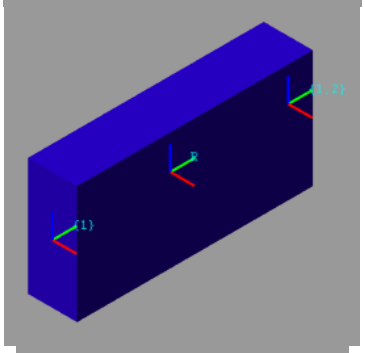
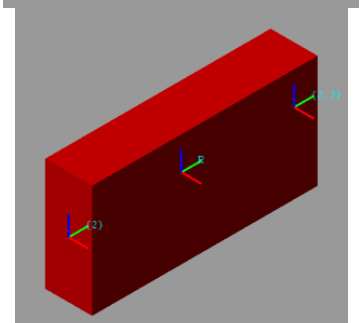
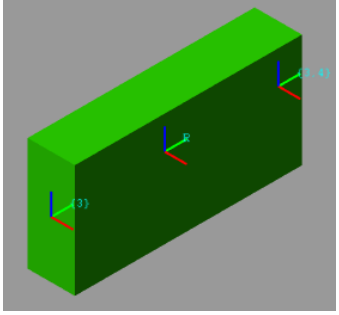


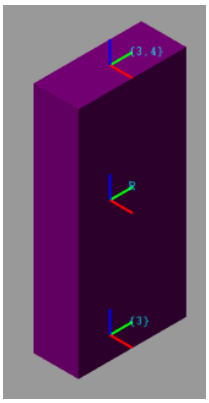
Figura 2. 6. Simulación de la primitiva en Simulink del robot 5GDL.

El robot de este proyecto cuenta con 5 eslabones: 1 base circular, 3 eslabones y una pinza (para propósitos de esta simulación, la pinza se simulará como un cuarto eslabón). Las dimensiones empleadas son:

Tabla 1. Elementos de Simulink

Elemento	Imagen	Dimensiones
Base		$d = 0.09[m]$ $h = 0.09[m]$
Eslabón 1		$x = 0.025[m]$ $y = 0.12[m]$ $z = 0.06[m]$
Eslabón 2		$x = 0.025[m]$ $y = 0.12[m]$ $z = 0.06[m]$
Eslabón 3		$x = 0.025[m]$ $y = 0.12[m]$ $z = 0.06[m]$

Pinzas



$$x = 0.025[m]$$
$$y = 0.06[m]$$
$$z = 0.13[m]$$

Estos eslabones se interconectaron en el entorno de Simulink de la siguiente forma:

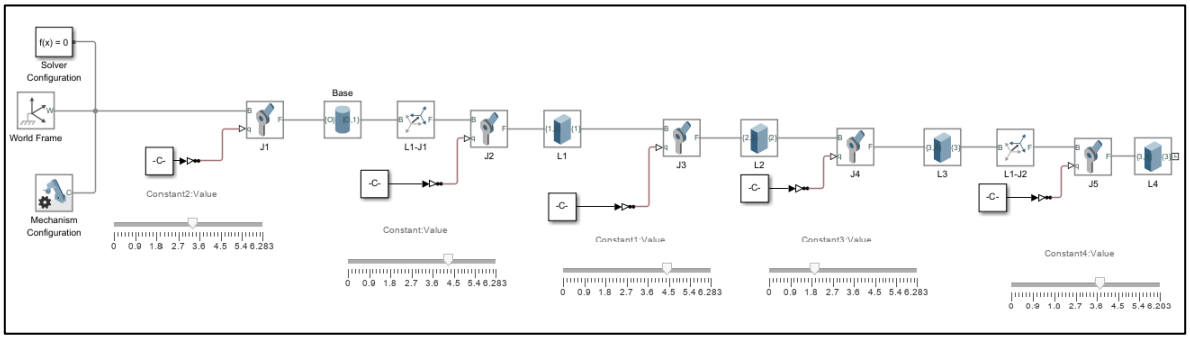


Figura 2. 7. Diagrama de bloques en Simulink

Cada entrada de las juntas es una variable, la cuál es controlada mediante un Slider. Los límites de este van desde 0 a 2π radianes (para que puedan dar un giro completo).

Simulación en Simulink

Para ejecutar la simulación de la "primitiva", primero se ejecuta la simulación de un eslabón, el cuál será manipulado mediante un bloque generador de señales. Esta señal estará definida de acuerdo con las necesidades del proyecto. Para lograr dicho objetivo, se necesita del bloque "*Signal Editor*".

Para utilizar el bloque anterior existen diferentes alternativas. En este proyecto, se realizó la señal deseada en Excel con la siguiente estructura:

TIME	DATA
0	0
5	1.5707963...
10	3.1415926...
15	1.5707963...
20	0

Figura 2. 8. Tabla de datos para la creación de una señal.

Este archivo se importa en el bloque "*Signal Editor*" dándole doble click y acceder al botón "*Signal Editor user interface*". Posteriormente se selecciona la opción "*import*" localizada en la pestaña "*signal editor*" y, finalmente, se procede a buscar el archivo de Excel correspondiente a la señal.

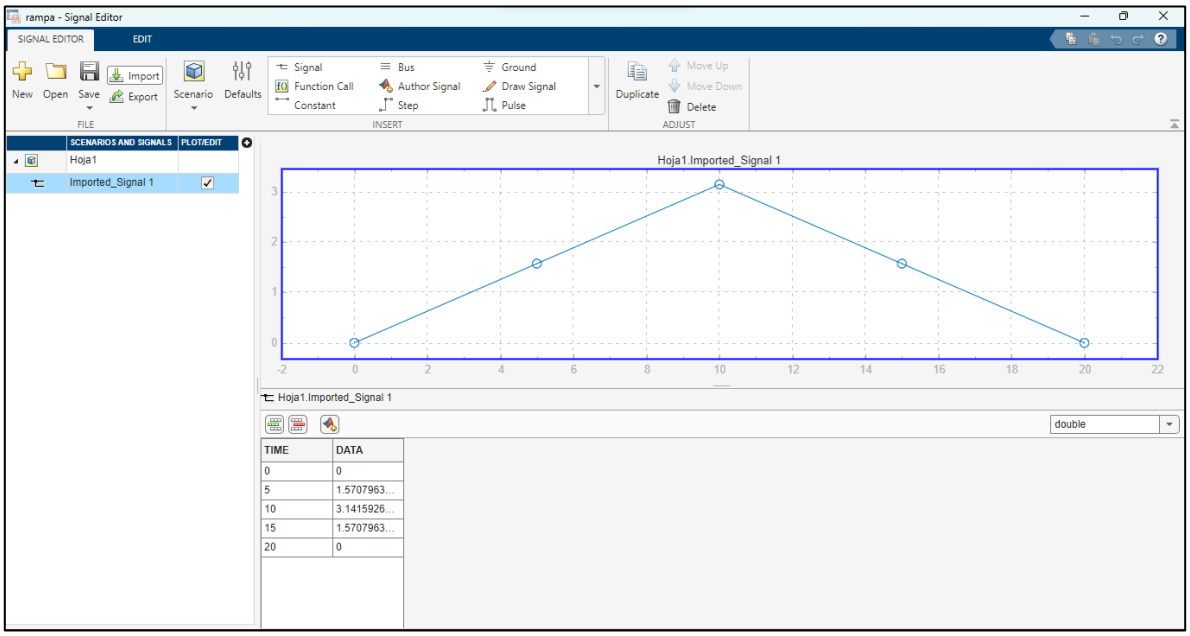


Figura 2. 9. Señales en el bloque Signal Editor.

Para poder hacer que un eslabón tenga el comportamiento de la señal generada, se requiere la siguiente estructura en el entorno de Simulink.

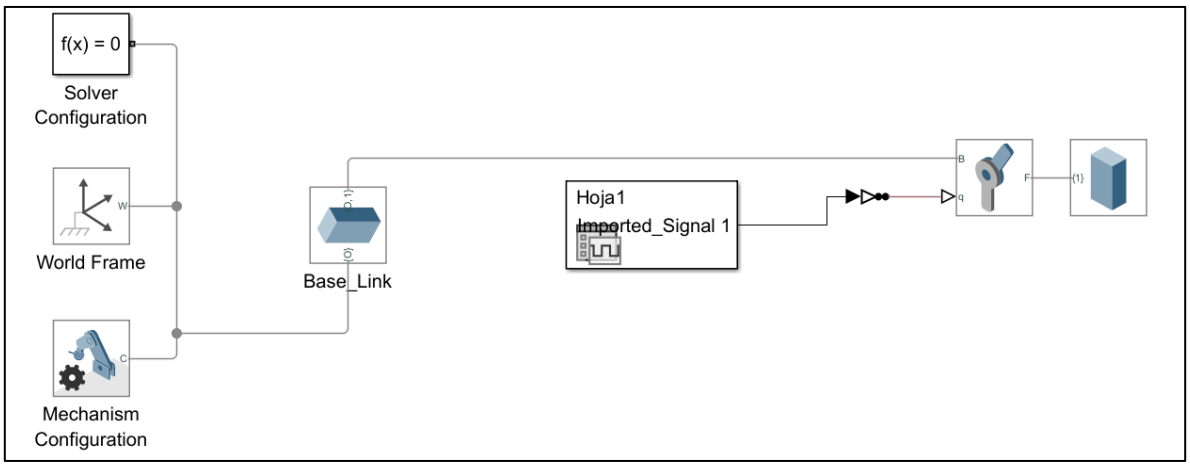


Figura 2. 10. Configuración de un eslabón para que simule una señal específica.

Por otro lado, si se quiere colocar una señal similar a la anterior pero para controlar un servomotor, no se puede implementar el bloque "*Signal Editor*", sino que se debe de emplear el bloque "*Desired Shaft Angle*" de la siguiente forma:

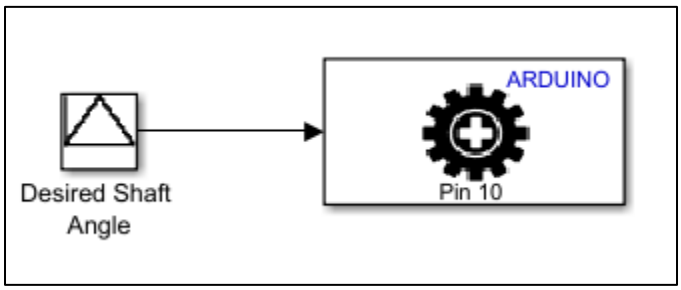


Figura 2. 11. Configuración en Simulink para controlar un Servomotor con un Arduino UNO.

Este bloque se configura de la siguiente forma:

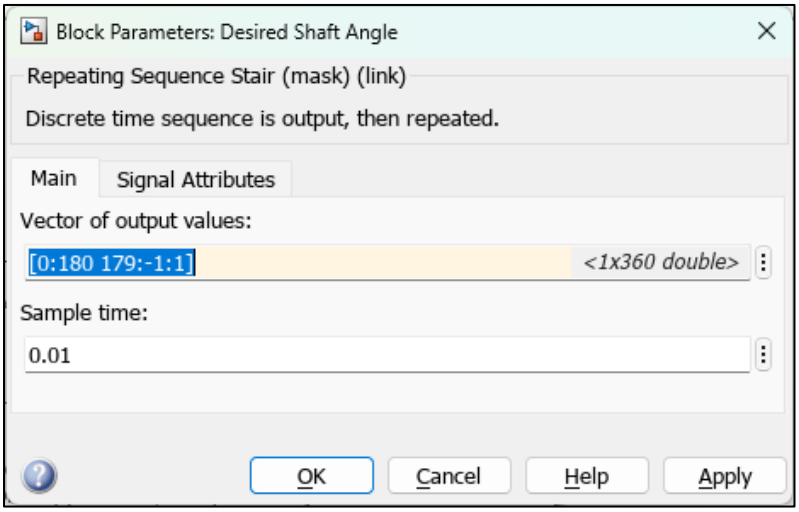


Figura 2. 12. Configuración del bloque "Desired Shaft Angle" para controlar un servomotor.

Nota importante

Cabe resaltar que **NO** se puede ejecutar la simulación de un eslabón al mismo tiempo en que se controla un servomotor en el entorno de Simulink. Es decir, no se recomienda hacer lo siguiente:

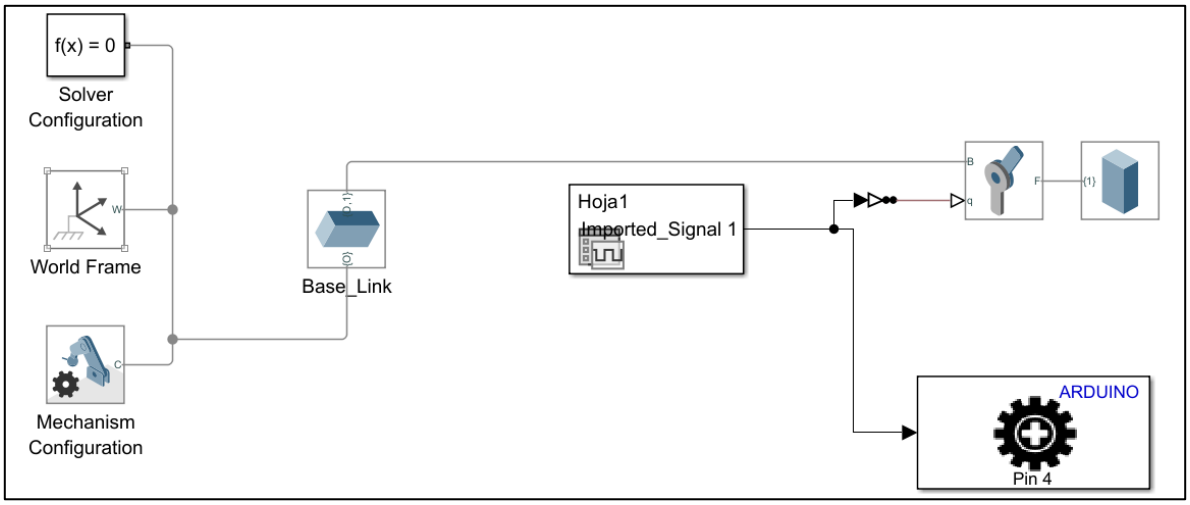


Figura 2. 13. Intento de simulación de un eslabón y control de un servomotor de forma simultánea.

Debido a que, si se sube el código al microcontrolador Arduino UNO, el entorno de Simulink "bloquea" la opción de simular el eslabón y viceversa. Además, las señales generadas por el "*Signal Editor*" no son compatibles con la entrada del bloque "*Standard Servo Write*", por lo que se generará un error.

Sin embargo, si se desea diseñar una señal en particular para que siga el servomotor, se debe de emplear el bloque "*Repeating Sequence Stair*". Este funciona de la siguiente forma:

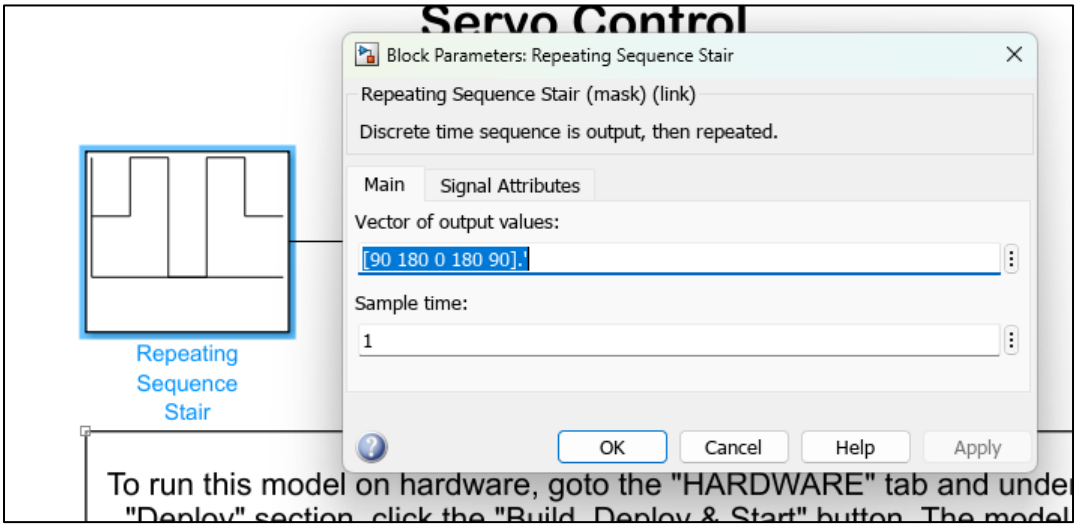


Figura 2. 14. Bloque Repeating Sequence Stair de Simulink.

En el vector de valores se escriben las posiciones angulares (en grados, no en radianes) que deberá seguir el servomotor. En la sección "sample time", se define el paso (en segundos) en los que cambiará de posición. Esta secuencia se repite hasta que se termine el tiempo de muestreo.

Fusion: Planos de las piezas/Modelo del robot 5GDL

Se utilizó el software de Fusion 360, una herramienta de modelado 3D, para crear modelos detallados de cada componente del brazo robótico. Fusion 360 permitió una aproximación integral al diseño, ya que proporciona herramientas muy utiles para la creación de formas complejas, simulación de movimiento y análisis estructural.

A continuación, se presentan algunas capturas de pantalla que destacan diferentes aspectos del proceso de modelado:

- En primer lugar, iniciamos el proceso creando los bocetos correspondientes a cada componente, tales como el servomotor, el eslabón y la pinza. Posteriormente, utilizando la herramienta de extrusión, creamos los cuerpos en cuestión. Utilizando los siguientes planos:

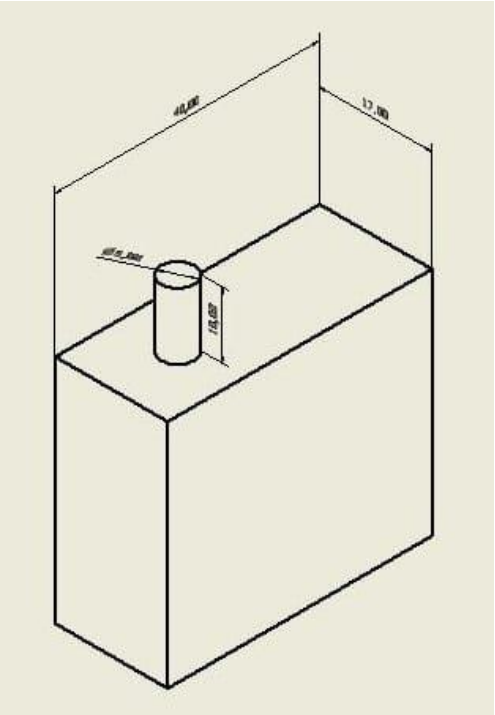


Figura – Plano del servomotor

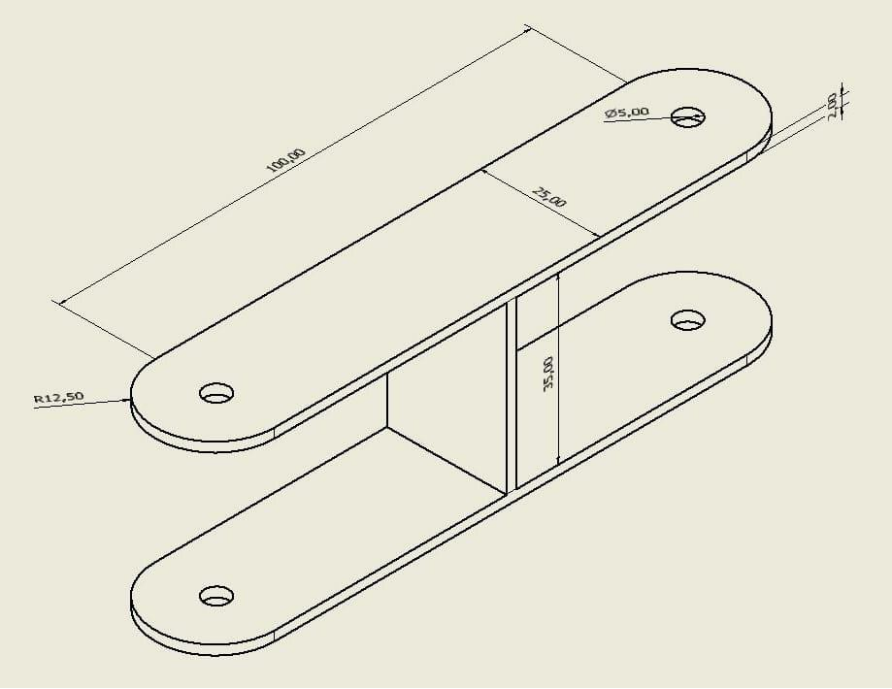


Figura – Plano del eslabón

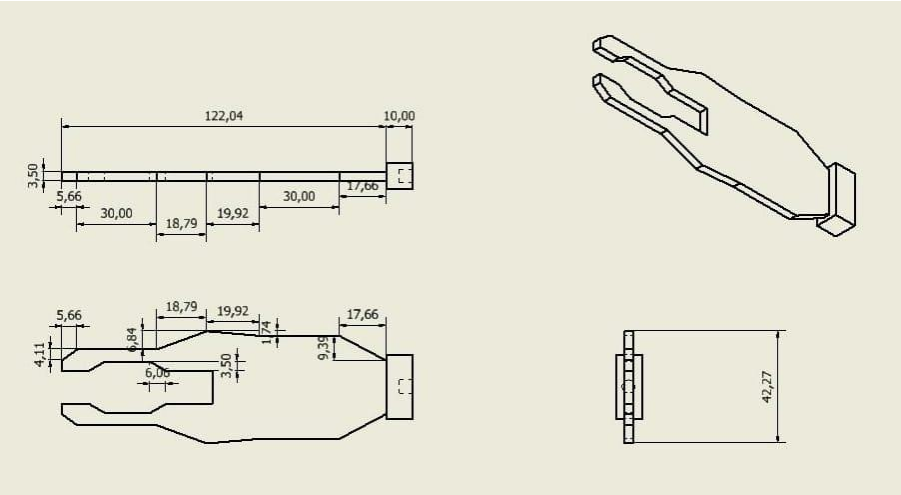


Figura – Plano de la pinza

- Convertir a componentes

Un punto importante a considerar es que se debemos crear componentes a partir de los cuerpos que modelamos.

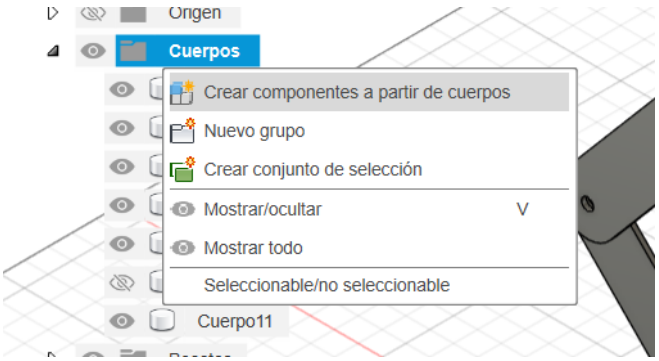


Figura – Crear componentes a partir de cuerpos

- Ensamble

Para colocar los componentes en relación unos con otros mediante un ensamble y definir el movimiento relativo entre ellos, utilizamos la función de unión.

Figura – Ensamblar

Seleccionamos los ejes de los primeros servomotores y definimos el movmimiento de la unión como ‘Revolución’.

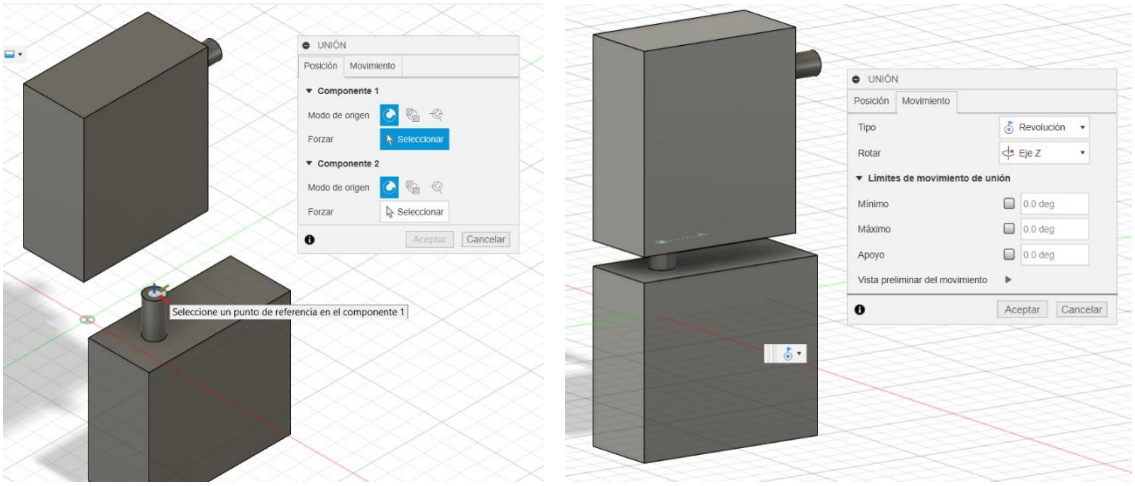


Figura – Ensamble servomotores

Continuando con los eslabones, seleccionamos ambos:

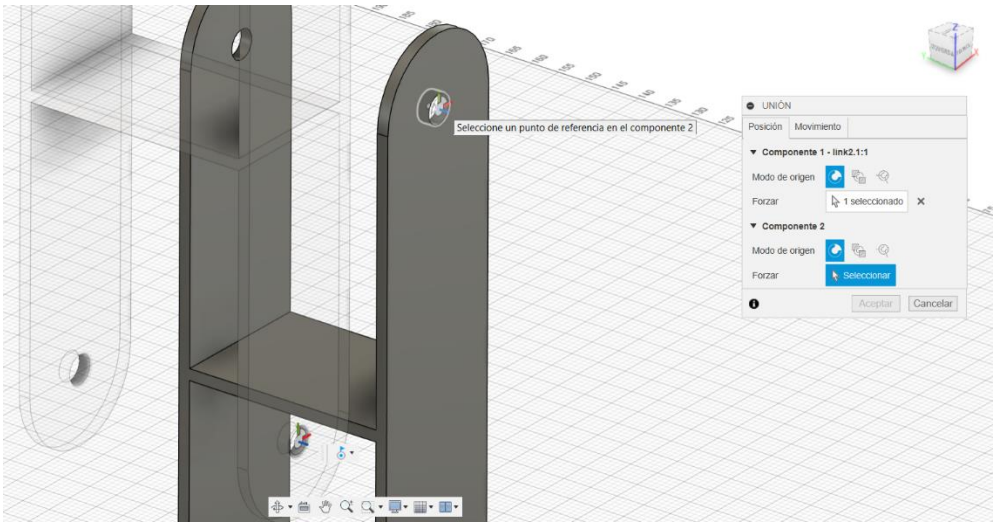


Figura – Ensamble eslabones

Seleccionamos el movimiento de la unión como ‘Revolución’ y definimos los limites a -115° y 115°.

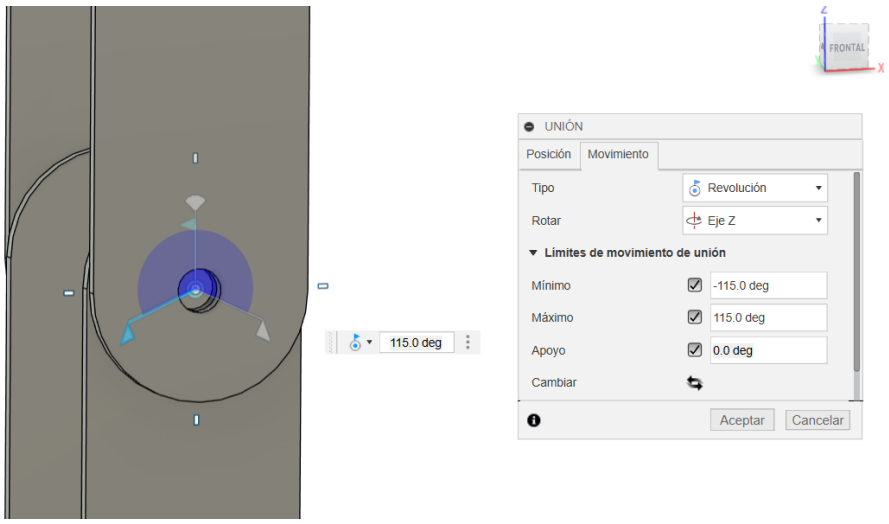


Figura - Relación de movimiento

Finalmente, definimos la unión de la pinza.

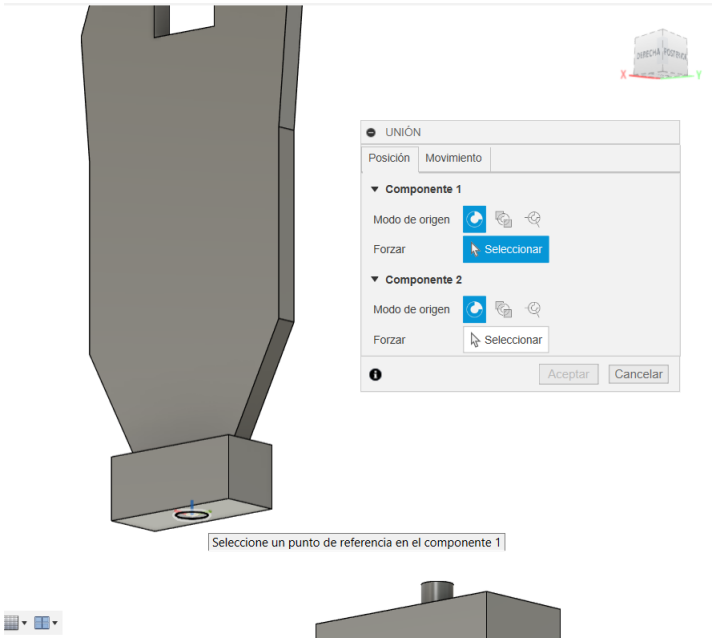


Figura - Unión pinza

Como resultado, obtenemos el ensamble completo del robot.

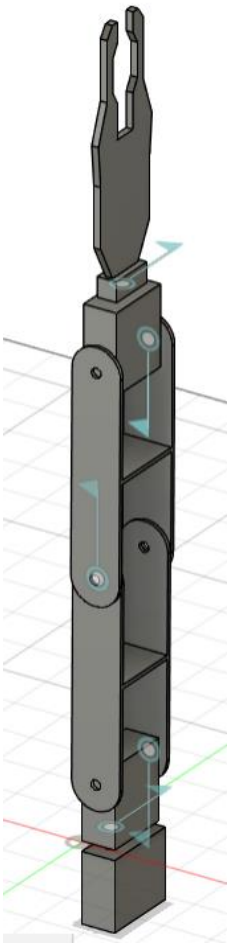


Figura – Ensamble robot

- Instalación Bullet URDF

A continuación, procedemos a instalar el plug in Fusion2Pybullet para poder convertir y exportar el archivo en formato URDF.

Lo descargamos directamente de GitHub a través del siguiente enlace:

https://github.com/yanshil/Fusion2PyBullet/blob/master/README.md

Para instalarlo en Fusion 360, en la pestaña de 'Tools', seleccionamos add-ins y en el signo '+' lo seleccionamos directo de nuestra carpeta de descargas:

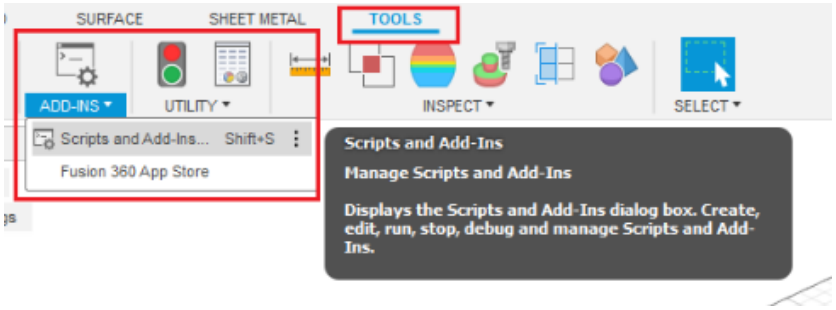


Figura – Instalación Bullet

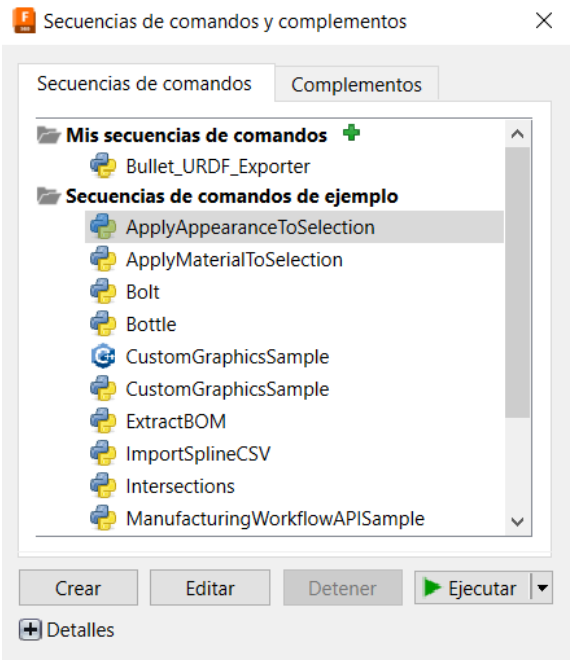


Figura - Complementos

- Exportación

Finalmente, para exportar el archivo en la pestaña de 'Tools', seleccionamos add-ins, Bullet_URDF_Exporter y por último, damos click en ´Ejecutar´.

- Probar ensamble

Para verificar que las uniones de nuestro ensamble sean correctas, entramos al siguiente enlace y arrastramos la carpeta que exportamos en formato URDF.

https://gkjohnson.github.io/urdf-loaders/javascript/example/bundle/index.html

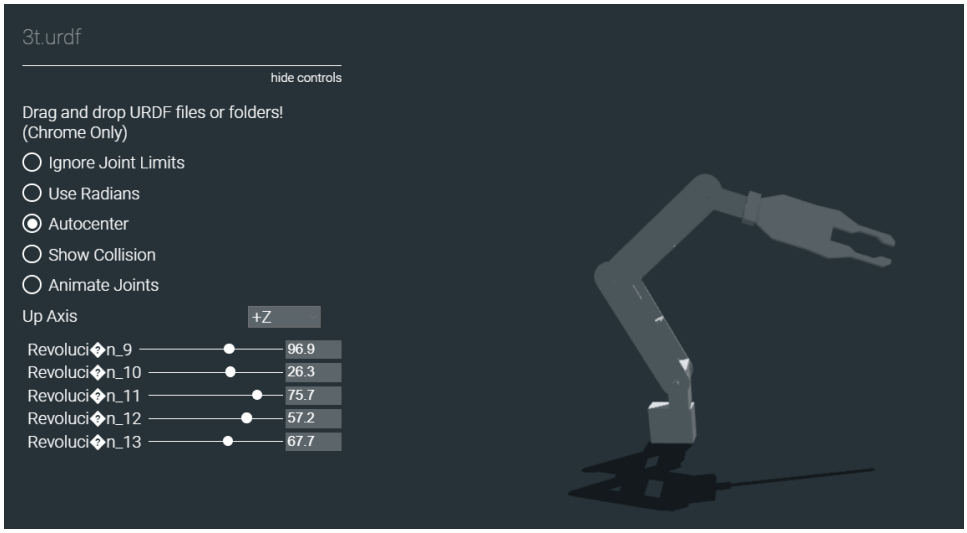


Figura – URDF Viewer

A lo largo del proceso de modelado, se aprovecharon las capacidades paramétricas de Fusion 360 para establecer relaciones entre los componentes, facilitando así la realización de cambios y ajustes sin comprometer la integridad del diseño general. Además, al ser una plataforma basada en la nube, facilitó la colaboración en tiempo real, permitiendo que el equipo acceda y comente sobre el modelo en cualquier fase del proyecto.

Las herramientas de simulación de Fusion 360 fueron fundamentales para evaluar la viabilidad del diseño, permitiendo la visualización de movimientos y la detección de interferencias.

Conclusión

En el proceso de desarrollo de un brazo robótico utilizando Arduino, Matlab - Simulink y Fusion, se partió de la premisa de controlar los servomotores de un robot de 5R. Inicialmente, se retomaron simulaciones en SolidWorks, empleadas por el equipo anteriores como interfaz gráfica. Sin embargo, debido a restricciones de licencias en la Facultad, se optó por rehacer las piezas en Fusion y cambiar el controlador de los servomotores de un microcontrolador *Raspberry Pi Trading Ltd Single Board Computer RPI4B* a Arduino UNO.

Las conclusiones se dividen en dos aspectos cruciales: la interfaz gráfica en Fusion y los controladores con Arduino.

En cuanto a estos últimos, se eligió utilizar Simulink incorporado a Matlab, aprovechando la licencia disponible en la Facultad. Se llevó a cabo una simulación de un robot de 5R "primitivo" para controlar los ángulos de cada eslabón. Al obtener las señales adecuadas para cada eslabón, se buscó transmitirlas a un servomotor controlado por Arduino. Se propuso la utilidad de la librería "Simulink Support Package for Arduino hardware", pero se encontró incompatibilidad entre los datos de la señal generada con el bloque "Signal Editor" y el bloque "Estándar Servo Write". A pesar de este inconveniente, se logró controlar la secuencia del servomotor empleando la librería, basándonos en el ejemplo del bloque "Repeating Sequence Stair", donde se define un vector que representa la magnitud del ángulo en radianes, generando un movimiento continuo del servomotor.

En cuanto a la interfaz gráfica en Fusion, se procedió a diseñar cada pieza del robot de 5R, agregando restricciones para visualizar el movimiento del robot y obtener una comprensión más clara de su funcionalidad. Este proceso detallado en Fusion permitió una representación visual precisa de cada componente del brazo robótico en desarrollo.

Fuentes bibliográficas

[1] Simulink Support Package for Arduino hardware. (2023, 13 septiembre). Simulink Support Package for Arduino Hardware - File Exchange - MATLAB CentralFile Exchange - MATLAB Central. https://la.mathworks.com/matlabcentral/fileexchange/40312-simulink-support-package-for-arduino-hardware

[2] P. Erik, Apuntes de: Robótica, Facultad de Ingeniería, UNAM, 2023.

[3] B. Antonio, Fundamentos de robótica 2a ed. Interamericana de España: McGraw, 2007.