

GRADO EN INGENIERÍA EN TECNOLOGÍA
INDUSTRIAL
TRABAJO FIN DE GRADO

***HERRAMIENTA MATLAB PARA
CONTROL DE UN MICROROBOT***

Alumno/Alumna: Arizaga, Echebarria, Julen

Director/Directora: <Zubizarreta, Pico, Asier>

Curso: 2017-2018

Fecha: <18, 07, 2018>

ÍNDICE

| | |
|--|----|
| TRABAJO FIN DE GRADO | 1 |
| 1. Resumen | 5 |
| 1.1. Castellano | 5 |
| 1.2. Euskera | 5 |
| 1.3. Inglés | 5 |
| 2. Lista de tablas e ilustraciones | 6 |
| 2.1. Lista de tablas | 6 |
| 2.2. Lista de ilustraciones..... | 6 |
| 3. Introducción..... | 8 |
| 4. Contexto..... | 9 |
| 4.1. Origen de la robótica | 9 |
| 4.2. Historia de los autómatas | 9 |
| 4.3. Historia de la robótica..... | 11 |
| 4.4 Brazo robótico | 12 |
| 4.6. Programa EduBot..... | 14 |
| 4.5. Brazo robótico Braccio | 15 |
| 5. Objetivos y Alcance | 16 |
| 5.1. Objetivos | 16 |
| 5.2. Alcance..... | 17 |
| 6. Beneficios del proyecto..... | 19 |
| 6.1. Beneficios económicos..... | 20 |
| 6.2. Beneficios del producto final..... | 21 |
| 6.3. Beneficios de seguridad | 21 |
| 7. Análisis de alternativas | 23 |
| 7.1. Placa Arduino Uno | 23 |
| 7.2. Placa Raspberry Pi 1 | 24 |
| 7.3. Placa hecha a medida | 25 |
| 7.4 Elección de la solución..... | 26 |
| 8. Metodología..... | 27 |
| 8.1. Visión global de la aplicación: panel principal | 27 |
| 8.1.1. Funciones en el panel principal..... | 28 |

| | |
|--|----|
| 8.2. Panel Cinemática Directa..... | 29 |
| 8.2.1. Cálculo del Problema cinemático directo..... | 32 |
| 8.2.2. Funciones en el panel de Cinemática Directa | 34 |
| 8.3. Panel Cinemática Inversa | 35 |
| 8.3.1. Cálculo del Problema cinemático inverso..... | 39 |
| 8.3.2. Funciones en el panel de Cinemática Inversa..... | 40 |
| 8.4. Panel de Espacio de Trabajo..... | 41 |
| 8.4.1 Funciones en el panel de Espacio de Trabajo..... | 43 |
| 8.5. Panel de Matriz de Denavit-Hartenberg..... | 43 |
| 8.5.1 Funciones en el panel de Matriz DH..... | 44 |
| 9. Descripción de tareas..... | 45 |
| 9.1. Seguimiento del proyecto con el tutor | 45 |
| 9.2. Adquisición de nociones básicas de Matlab y Arduino | 45 |
| 9.3. Entendimiento del programa EduBot | 46 |
| 9.4. Estudio preliminar del robot..... | 46 |
| 9.5. Programación | 46 |
| 9.5.1. Programación en Arduino en la placa base | 46 |
| 9.5.2. Programación en Matlab | 47 |
| 9.6. Montaje del robot..... | 47 |
| 9.7. Validación del programa..... | 47 |
| 9.8. Redacción del documento | 47 |
| 10. Diagrama de Gantt..... | 48 |
| 11. Presupuesto..... | 49 |
| 11.1. Horas de trabajo..... | 49 |
| 11.2. Material necesario..... | 49 |
| 11.3. Amortizaciones..... | 49 |
| 11.4. Otros gastos | 50 |
| 11.5. Costes indirectos..... | 50 |
| 11.6. Presupuesto total | 51 |
| 12. Conclusión..... | 52 |
| 13. Bibliografía | 53 |
| ANEXO I: Funciones del programa..... | 54 |

| | |
|--|----|
| ANEXO II: Manual de instrucciones..... | 59 |
| Inicio..... | 59 |
| Cinemática directa | 59 |
| Cinemática Inversa | 61 |
| Espacio de trabajo | 61 |
| Matriz de DH..... | 62 |

1. Resumen

1.1. Castellano

El objetivo del presente trabajo final de grado es controlar el brazo robótico Braccio mediante el programa informático Matlab. El brazo robótico Braccio de Tinkerkit tiene 5 grados de libertad más una pinza, lo cual acaban siendo 6 servos (pequeños motores). Para lograr el objetivo se tendrá que establecer un puente mediante Matlab y los servos del brazo robótico, los cuales podrán ser controlados por una interfaz gráfica que se creará. Para la creación de esta interfaz gráfica además de sus subyacentes programas, se necesitarán las longitudes de las partes del brazo y los límites a los que puedan girar los servos. La interfaz gráfica tendrá tres funciones: control mediante cinemática inversa, control mediante cinemática directa y espacio de trabajo

1.2. Euskera

Hurrengo gradu amaierako lanaren helburua Braccio beso robotikoa kontrolatzea da Matlab programa informatikoarekin. Tinkerkit-en Braccio beso robotikoak 5 askatasun gradu eta pintza bat dauzka, beraz hauek mugitzeko 6 servo (motore txikiak) beharko dira. Lan honen helburua gauzatzeko Matlab eta servoen arteko konexio bat egin beharko da, eta hauek Matlaben programatu beharko den interfaz grafiko batetik kontrolatu ahalko dira. Interfaz grafiko hau eta honek behar dituen programak egiteko, Braccio beso robotikoaren atalen luzeerak eta servoen biratzeko limiteak beharko dira. Interfazak hurrengo hiru funtzioak edukiko ditu: zinematika zuzenaren arabeko kontrola, alderantzizko zinematikaren araberako kontrola eta lan eremua.

1.3. Inglés

The objective of the present final degree work is to control the robotic arm Braccio from the informatic program Matlab. The Braccio robotic arm from Tinkerkit has 5 degrees of freedom plus a gripper, with a total of 6 servos (small motors). In order to reach the objective a connection between Matlab and the servos of the robotic arm has been developed, so that it is possible to control the servos via a Graphic User Interface (GUI). For the creation of this GUI and its subjacent programs, the lengths of the arm's parts and the limits at which the servos can turn will be needed. The GUI will have three functions: control by forward kinematics, control by inverse kinematics and workspace.

2. Lista de tablas e ilustraciones

2.1. Lista de tablas

| | |
|---|----|
| Tabla 1: Grados de Libertad de Braccio..... | 15 |
| Tabla 2. Tabla de calificación de la placa Arduino Uno | 23 |
| Tabla 3. Tabla de calificación de la placa Raspberry Pi 1 | 25 |
| Tabla 4. Tabla de calificación de la placa hecha a medida | 26 |
| Tabla 5. Tabla del diagrama de Gantt | 48 |
| Tabla 7. Presupuesto de las horas de trabajo..... | 49 |
| Tabla 8. Presupuesto del material necesario..... | 49 |
| Tabla 9. Presupuesto de las amortizaciones..... | 50 |
| Tabla 10. Presupuesto de otros gastos..... | 50 |
| Tabla 11. Presupuesto de los costes indirectos..... | 50 |

2.2. Lista de ilustraciones

| | |
|---|----|
| Ilustración 1: Robot Braccio armado | 8 |
| Ilustración 2: Robot Braccio desmontado..... | 8 |
| Ilustración 3: Robot de Leonardo con funcionamiento interno | 10 |
| Ilustración 4: Autómata "El escritor" con mecanismo interno | 11 |
| Ilustración 5. Tipos de robots según morfología | 13 |
| Ilustración 6: Interfaz gráfica de EduBot | 14 |
| Ilustración 7: Robot Braccio con ejes..... | 15 |
| Ilustración 8. Programas utilizados..... | 18 |
| Ilustración 9. Programas en la carpeta "2d" | 18 |
| Ilustración 10: Cadena de montaje robotizada..... | 19 |
| Ilustración 11: Ventas de robots en el mundo [World Economic Forum] | 20 |
| Ilustración 12. Proceso de pintura automatizado | 22 |
| Ilustración 13. Proceso de pintura manual | 22 |
| Ilustración 14. Placa Arduino Uno | 24 |
| Ilustración 15. Placa Raspberry Pi 1 | 25 |
| Ilustración 16. Ventana inicial de la interfaz gráfica..... | 27 |
| Ilustración 17. Panel de cinemática directa | 30 |

| | |
|---|----|
| Ilustración 18. Segundo panel de cinemática directa | 31 |
| Ilustración 19. Datos de entrada y salida de la cinemática directa e inversa | 32 |
| Ilustración 20. Imagen de la notación de Denavit-Hartenberg..... | 33 |
| Ilustración 21. Matriz de Denavit-Hartenberg..... | 33 |
| Ilustración 22. Matriz DH del robot Braccio | 34 |
| Ilustración 23. Panel de cinemática inversa | 35 |
| Ilustración 24. Codo abajo (izquierda) y codo arriba (derecha)..... | 37 |
| Ilustración 25. Segundo panel de cinemática inversa..... | 38 |
| Ilustración 26. Mensaje error cinemática inversa..... | 39 |
| Ilustración 27. Panel Espacio de Trabajo | 41 |
| Ilustración 28. Espacio de trabajo de Braccio en 2D si se bloquea la articulación 1 | 42 |
| Ilustración 29. Panel de Matriz DH | 43 |
| Ilustración 30. Diagrama de Gantt | 48 |
| Ilustración 31. Programas utilizados | 54 |
| Ilustración 32. Ventana para iniciar el programa..... | 59 |
| Ilustración 33. Panel de cinemática directa: modificar coordenadas | 60 |
| Ilustración 34. Panel de cinemática directa: ver coordenadas cartesianas..... | 60 |
| Ilustración 35. Panel de cinemática inversa: introducir datos | 61 |

3. Introducción

Los robots están adquiriendo cada vez más importancia en cualquier entorno, desde el industrial hasta el doméstico. Es por esto que cada vez se invierte más tiempo y dinero en su estudio, para así poder desarrollar robots más rápidos, más seguros y en general, más eficaces. Ahora se explicará más en detalle el caso de los brazos robóticos.

Un brazo robótico es un tipo de brazo mecánico, normalmente programable, con funciones parecidas a las de un brazo humano. Estos tipos de robots han adquirido mucha presencia en la industria en los últimos años, sobre todo en la cadena de montaje, debido a su programabilidad, velocidad, precisión y versatilidad, ya que un único brazo robótico puede realizar infinidad de movimientos y funciones, las cuales recuerdan a las posibilidades de un brazo humano. El uso de dichos brazos en la industria no hace más que aumentar: en abril de 2017 la industria mundial contaba con 1,63 millones de robots según la IFR (International Federation of Robotics), y se espera que alcance los 2,6 millones en 2019.

En las siguientes imágenes podemos ver el brazo robótico Braccio de Tinkerkit que se va a utilizar durante este proyecto.



Ilustración 1: Robot Braccio armado



Ilustración 2: Robot Braccio desmontado

El objetivo de este trabajo es adentrarse en el mundo de la robótica y su programación. Para ello se utilizará un brazo robótico pequeño y desmontable, pero aun así de

morfología similar a la de los robots que se utilizan hoy en día en la industria. Se programará este brazo robótico y se desarrollará una aplicación en Matlab para poder controlarlo de manera fácil y sencilla.

A lo largo de este trabajo se va a utilizar el programa informático EduBot, el cual es un programa desarrollado por alumnos y profesores de la UPV para controlar diferentes tipos de robots. Para nuestro propósito se van a utilizar los programas y la interfaz gráfica, modificándolos para así adaptarlos a nuestro brazo robótico y las funciones que queremos que cumpla.

4. Contexto

4.1. Origen de la robótica

La robótica se define como la técnica que aplica la informática al diseño y empleo de aparatos que, en sustitución de personas, realizan operaciones o trabajos, por lo general en instalaciones industriales (RAE).

El término “Robot” fue acuñado por el escritor checo Karel Čapek en 1923, en su obra dramática *Rossum's Universal Robots* a partir de la palabra checa *robot*, que significa servidumbre o trabajo forzado. Aunque el término “Robot” sea bastante nuevo, en la prehistoria ya se hablaba sobre los autómatas, los que se definen como máquinas que imitan la figura y los movimientos de un ser animado.

4.2. Historia de los autómatas

Históricamente los primeros autómatas se remontan en la prehistoria donde las estatuas de algunos de sus dioses o reyes despedían fuego de sus ojos. Esta finalidad religiosa continuó hasta la Grecia clásica donde existían estatuas con movimiento gracias a las energías hidráulicas, conocimientos que quedaron plasmados en el primer libro que trató la figura de los robots automática, escrita por Herón de Alejandría (10 d.C – 70 d.C). Más adelante en el año 805 los hermanos Banu Musa escribieron el “Libro de Mecanismos Ingeniosos” en el que son descritos un centenar de mecanismos y autómatas, y cómo emplearlos.

Siglos más tarde, Leonardo da Vinci diseñó al menos dos autómatas, uno de los cuales se considera uno de los primeros con forma completamente humana, y aunque no fue construido en la antigüedad, sí lo ha sido en la actualidad según los dibujos originales, y podía mover los brazos, girar la cabeza y sentarse.



Ilustración 3: Robot de Leonardo con funcionamiento interno

La época de esplendor del autómata y el automatismo llegó en el siglo XVIII donde, junto con los avances tecnológicos y científicos, se desarrollaron los mejores y más perfectos autómatas de la historia.

Uno de los mejores y más conocido creadores de autómatas de la historia fue Pierre Jaquet-Droz, suizo nacido en 1721, quien creó los tres autómatas más complejos y famosos del siglo XVIII: La pianista, El dibujante y El escritor. De estas tres obras maestras, la más compleja fue El escritor, autómata compuesto por más de 6000 piezas que podía escribir utilizando la pluma gracias a una rueda donde se seleccionaban los caracteres uno a uno, pudiendo escribir así pequeños textos de unas cuarenta palabras de longitud. Este autómata además de escribir textos, realizaba movimientos propios de un ser humano como mojar la tinta y escurrir el sobrante para no manchar el papel, levantar la pluma como si estuviera pensando y seguir con la mirada el papel y la pluma mientras escribe.



Ilustración 4: Autómata "El escritor" con mecanismo interno

Más adelante, a finales del siglo XIX y comienzos del siglo XX se siguieron elaborando autómatas, pero menos elaborados, y estuvieron más guiados al mundo del espectáculo. Finalmente con el estallido de la Primera Guerra Mundial la industria de los autómatas desaparece y dará paso a los modernos robots más adelante.

4.3. Historia de la robótica

Hoy en día se utiliza la palabra "robot" ampliamente para describir cualquier tipo de autómata, pero tal como se define la robótica hoy en día, ésta debe aplicar la informática, que a su vez depende de la electrónica digital. Por lo tanto, no se puede hablar de robótica antes del año 1947, fecha en la que John Bardeen, Walter Houser Brattain y William Bradford Shockley inventaron el transistor y abrieron las puertas a la electrónica digital, aunque realmente no fue hasta después cuando se introdujo la electrónica digital en los autómatas, pues el primer microcontrolador data de 1970.

Desde 1970 hasta el día de hoy la robótica ha tenido un crecimiento exponencial, estando hoy en día presente en la mayoría de procesos industriales. En 1971 ya se desarrolló el "Stanford Arm", un pequeño brazo de robot de accionamiento eléctrico, y en 1978 y 1979 ya se desarrollaron los robots PUMA (Programmable Universal Machine

for Assambly) y SCARA (Selective Compliance Arm for Robotic Assambly) respectivamente, los cuales fueron introducidos en la cadena de montaje para automatizarla. En los subsiguientes años se hicieron mejores robots tanto para la industria como para el hogar.

4.4 Brazo robótico

Un brazo robótico es un tipo de brazo mecánico, normalmente programable, con funciones parecidas a las de un brazo humano; éste puede ser la suma total del mecanismo o puede ser parte de un robot más complejo. Las partes de estos manipuladores o brazos son interconectadas a través de articulaciones que permiten tanto un movimiento rotacional (tales como los de un robot articulado), como un movimiento traslacional o desplazamiento lineal.

Este tipo de robots son los más utilizados en la industria actualmente debido a su programabilidad, velocidad, precisión y versatilidad. Los principales parámetros que caracterizan a estos robots son los siguientes:

- **Número de grados de libertad.** Es el número total de grados de libertad de un robot, dado por la suma de G.D.L. de las articulaciones que lo componen.
- **Espacio de accesibilidad o espacio (volumen) de trabajo.** Es el conjunto de puntos del espacio accesibles al punto terminal (PT), que depende de la configuración geométrica del manipulador. Un punto del espacio se dice totalmente accesible si el PT puede situarse en él en todas las orientaciones que permita la constitución del manipulador y se dice parcialmente accesible si es accesible por el PT pero no en todas las orientaciones posibles.
- **Capacidad de posicionamiento del punto terminal.** Se concreta en tres magnitudes fundamentales: resolución espacial, precisión y repetibilidad, que miden el grado de exactitud en la realización de los movimientos de un manipulador al realizar una tarea programada.
- **Capacidad de carga.** Es el peso que puede transportar el elemento terminal del manipulador.
- **Velocidad.** Es la máxima velocidad que alcanzan el PT y las articulaciones.

Dependiendo del proceso que se quiera llevar a cabo, se le exigirá al robot diferentes parámetros, y dependiendo de ellos se obtendrá uno u otro robot. Los brazos robóticos aparte de diferentes parámetros también pueden ser clasificados en diferentes tipos, dependiendo de su morfología:

- **Cartesiano:** Es un robot en el que el brazo tiene tres articulaciones prismáticas, cuyos ejes son coincidentes con los ejes cartesianos.
- **Cilíndrico:** Es un robot cuyos ejes forman un sistema de coordenadas cilíndricas, con una articulación rotacional sobre una base y articulaciones lineales para el movimiento en altura y en radio.
- **Polar:** Es un robot cuyos ejes forman un sistema polar de coordenadas. Cuenta con dos articulaciones rotacionales y una lineal.
- **Esférico (o de brazo articulado):** Es un robot cuyos ejes forman un sistema polar de coordenadas. Cuenta con tres articulaciones rotacionales.
- **Mixto:** Posee varios tipos de articulaciones, combinaciones de las anteriores. Es destacable la configuración **SCARA** (*Selective Compliance Assembly Robot Arm*)
- **Paralelo:** Posee brazos con articulaciones prismáticas o rotacionales concurrentes.

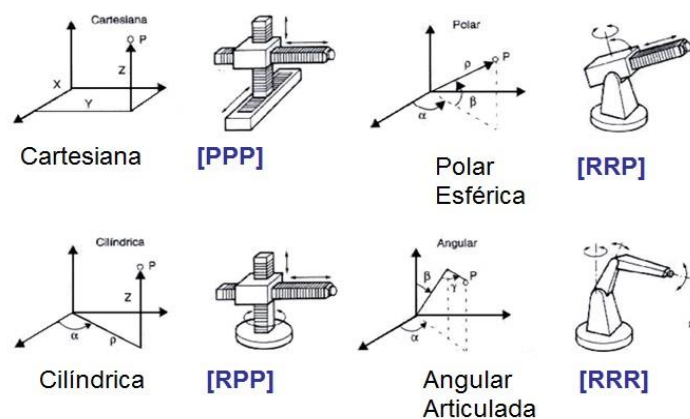


Ilustración 5. Tipos de robots según morfología

4.6. Programa EduBot

EduBot es un programa informático educacional hecho para la simulación del control de diferentes robots por los alumnos y el profesorado de la UPV en el entorno de Matlab. Ya que el objetivo de este trabajo es el de controlar un brazo robótico mediante Matlab, se ha echado mano de los programas y la interfaz gráfica de Edubot, modificándolas para que se ajusten a nuestro robot y las opciones de control.

EduBot es capaz de simular el control de los robots Planar 2R, SCARA, STANFORD y PUMA560 y realizar diferentes acciones como podemos ver en la siguiente imagen, la cual es la interfaz gráfica de EduBot para el control del robot Planar 2R.

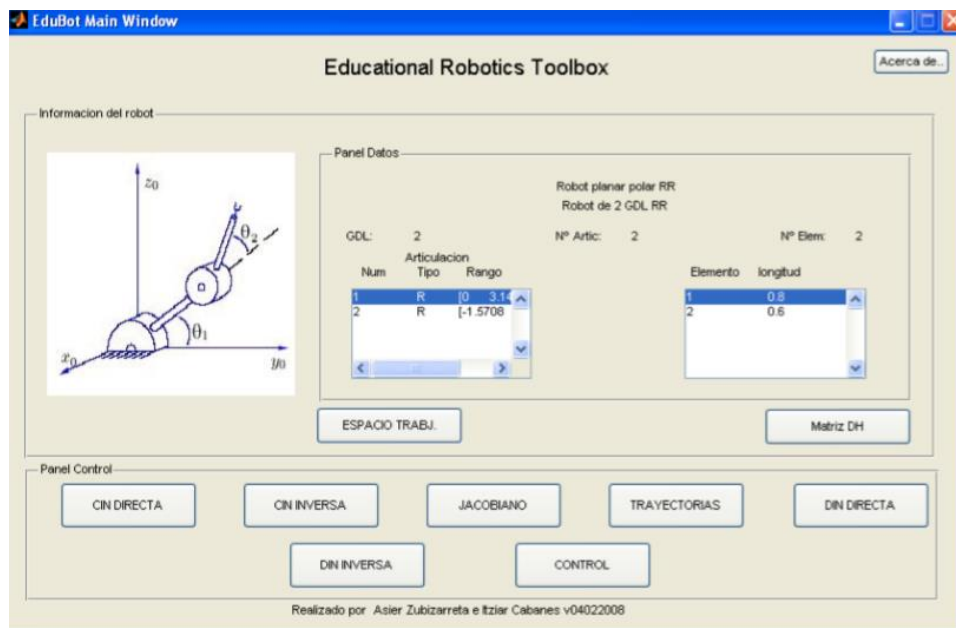


Ilustración 6: Interfaz gráfica de EduBot

Como se puede ver en la imagen EduBot puede realizar diversas simulaciones, pero para el caso del robot Braccio solamente se va a utilizar la cinemática inversa, cinemática directa y el espacio de trabajo.

4.5. Brazo robótico Braccio

En nuestro caso disponemos de un brazo robótico de 5 G.D.L. más pinza, con poca capacidad de carga y velocidad relativamente baja. Además de esto, por la disposición y la cantidad de sus grados de libertad (todos rotacionales) se considera un brazo robótico angular o antropomórfico.

Los 5 grados de libertad están definidos en la siguiente tabla:

| Etiqueta ejes | Nombre ejes |
|---------------|---------------------------------|
| Eje J1 | Eje de rotación de la base |
| Eje J2 | Eje de rotación del hombro |
| Eje J3 | Eje de rotación del codo |
| Eje J4 | Eje de inclinación de la muñeca |
| Eje J5 | Eje de rotación de la muñeca |

Tabla 1: Grados de Libertad de Braccio

En la siguiente figura se muestra el robot Braccio junto con sus ejes

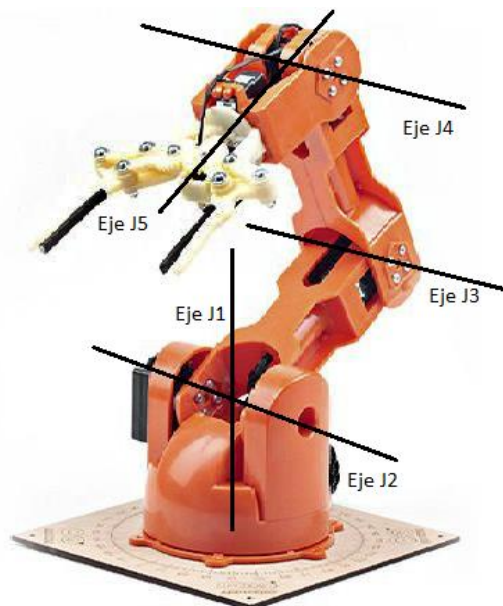


Ilustración 7: Robot Braccio con ejes

5. Objetivos y Alcance

5.1. Objetivos

El objetivo de este proyecto es el control del brazo robótico Braccio mediante Matlab. Para ello se tendrá que desarrollar una GUI (Graphical User Interface) con todos los programas necesarios para desarrollar la cinemática directa e inversa.

Por lo tanto, la aplicación tendrá que cumplir con las siguientes especificaciones:

- Layout de la interfaz gráfica: El layout de la GUI deberá ser sencilla e intuitiva, de manera que el usuario pueda controlar el robot sin tener conocimientos en informática o robótica
- Cinemática directa e inversa: En la GUI habrá un botón para calcular la cinemática directa y otro para calcular la cinemática inversa. Debido a esto tendrá que haber un cúmulo de subprogramas que desarrollen esta función cuando sea necesario. Calcular la cinemática directa es sencillo, ya que el usuario dará los grados a los que se deban rotar los servos del robot. Calcular la cinemática inversa es algo más complicado, ya que el usuario dará la posición y orientación en la que quiere que se sitúe la pinza (coordenadas XYZ), y se tendrá que calcular los grados de rotación de los servos necesarios para llegar a la posición y orientación deseadas.
- Espacio de trabajo: En la GUI habrá un botón que le mostrará al usuario los lugares a los que le será posible llegar con la pinza. Se le mostrará al usuario una imagen en 2D con coordenadas YZ, ya que en realidad el espacio de trabajo será la imagen en 2D rotada 180° frente al eje Z.
- Enviar órdenes: Esta es una función que deberá cumplir el programa. Para poder controlar el robot, este tendrá que estar conectado al puerto USB del ordenador mediante un dispositivo adecuado, cuya elección se realizará posteriormente.

5.2. Alcance

Para este trabajo que debe cumplir con las especificaciones dichas previamente, se deberán desarrollar ciertos programas, y se utilizarán otros ya desarrollados. Los programas ya desarrollados se cogerán del programa informático EduBot.

Los programas ya desarrollados y las funciones que lo hacen posible serán los siguientes:

- Cálculo de cinemática directa
 - Calcular_Cinemática_Directa
 - Cargar_robot
 - Demo_Window_CIn_Direct
 - Fkine
 - Globales

- Cálculo de cinemática inversa
 - Calcular_Cinemática_Inversa
 - Cargar_robot
 - Ikine
 - Demo_Window_CIn_Invers
 - Globales
 - Tr2eul
 - Eul2tr
 - Transl
 - Rotx, roty, rotz
 - Tr2diff, tr2rot
 - Jacob0, jacobn

- Cálculo de espacio de trabajo
 - Demo_Window_EspacTrabajo
 - Espacio_Trabajo_2D
 - Grupo de funciones bajo carpeta "2d"

Los programas que deberán ser desarrollados serán los siguientes:

- Programa que contenga los datos del robot Braccio
 - Braccio
 - BraccioCreator

- Programa que haga el enlace entre Matlab y arduino, mandando los grados de rotación de cada servo
 - Braccio_enviar_coordenadas

- Layout de la GUI, así como asociación de cada botón con su correspondiente función
 - ControlBraccio1
- Programa para que el robot se mueva una vez reciba los grados de rotación de cada servo, este programa deberá ser hecho en código Arduino, debido a que la placa base es de Arduino

En la siguiente imagen se muestran los programas utilizados para realizar el control del brazo robótico Braccio.

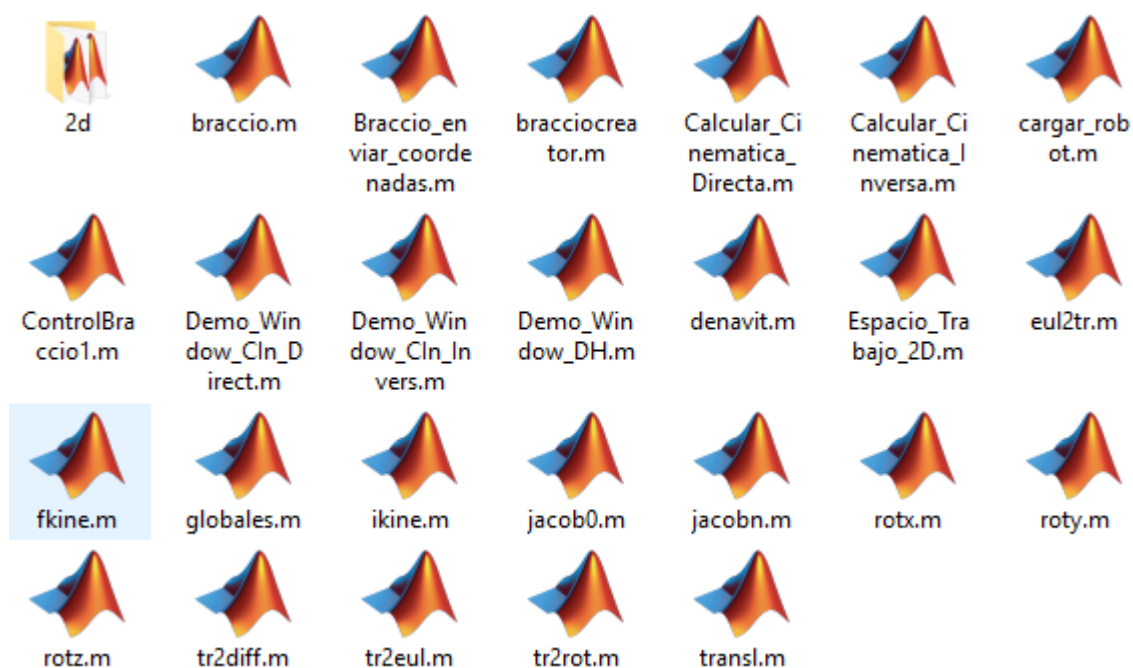


Ilustración 8. Programas utilizados



Ilustración 9. Programas en la carpeta "2d"

6. Beneficios del proyecto

Tal y como se ha indicado previamente, los robots utilizados en la industria están aumentando exponencialmente. La implementación de estos robots en la industria tiene numerosos beneficios, tanto económica como socialmente, además de contribuir al avance tecnológico. Ya que el fin de este proyecto es familiarizarse con el control de brazos robóticos, se van a analizar los beneficios que trae su implementación en la industria, con la consecuente automatización.

En la siguiente imagen podemos apreciar una cadena de montaje de coches totalmente automatizada por brazos robóticos.



Ilustración 10: Cadena de montaje robotizada

Así pues, estos son los beneficios que traen los brazos robóticos, así como su control e implementación en la industria:

6.1. Beneficios económicos

El aspecto económico es posiblemente el principal motivo por el cual ha aumentado tanto la robótica y la automatización en los últimos años. La implementación de la robótica en la industria ha hecho que no sean necesarios tantos empleados en la cadena de montaje, disminuyendo así el empleo precario y aumentando el empleo cualificado, pues es necesaria gente capaz de controlar estos robots.

En la siguiente imagen podemos apreciar las ventas de robots industriales en el mundo, y su aumento en los últimos años:

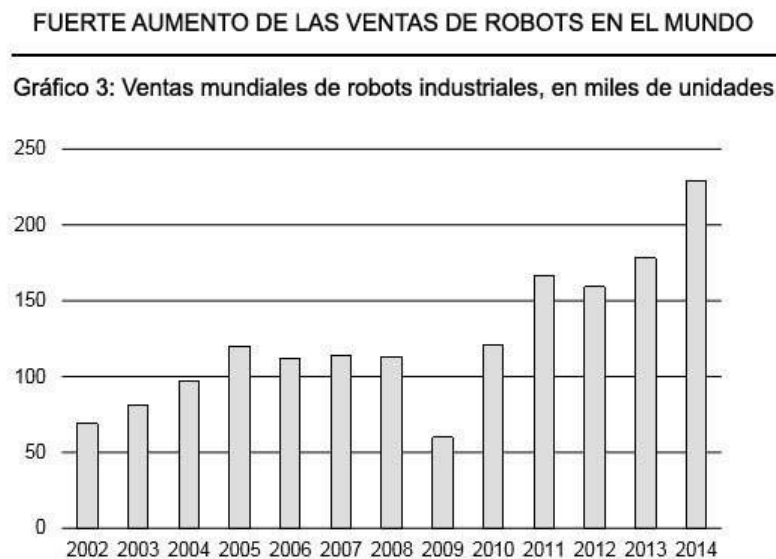


Ilustración 11: Ventas de robots en el mundo [World Economic Forum]

En cuanto a la empresa, los beneficios económicos se crean debido a las siguientes características que aporta la robótica:

- Aumento de producción y eficiencia debido a una disminución de los tiempos de producción y una mayor productividad, así como un mayor aprovechamiento de los materiales
- La reparación necesaria por los robots es mínima.
- Capacidad de adaptarse a los cambios de demanda, produciendo más o menos dependiendo del mercado actual.
- Ahorro en los costes de formación y entrenamiento de los operarios

6.2. Beneficios del producto final

En cuanto al producto final, se pueden apreciar notables mejoras respecto a un producto hecho manualmente sin la ayuda de robots industriales. Los robots industriales pueden ser mucho más precisos y rápidos que un ser humano, y pueden repetir los mismos movimientos una y otra vez con ninguna, o muy pequeña, desviación (debido a la precisión original del robot y su control). Esto se traduce en los siguientes beneficios para el producto final:

- Mejora en la calidad del producto final, así como su acabado. Sin duda uno de los aspectos más importantes de la automatización industrial
- Debido al aumento de precisión, se pueden hacer productos que antes eran imposibles de hacer debido a la técnica existente
- Vida útil del producto prolongada

6.3. Beneficios de seguridad

Como se ha comentado, la robotización y automatización de los procesos industriales trae consigo la pérdida de empleo precario, ya que muchos de estos empleos solían ser peligrosos debido a que el operario debía de estar cerca de máquinas muy calientes, o rotando a grandes velocidades, lo cual, aunque se tomaran medidas de seguridad, seguía siendo peligroso. Con la llegada de la robotización, estos trabajadores han sido sustituidos por máquinas, evitando así riesgos innecesarios.

Un ejemplo de esto es el proceso de pintura, el cual desprende vapores tóxicos. Este proceso es muy utilizado en todo tipo de productos en toda la industria, y antes el operario debía de estar dotado de un sistema de protección tipo máscara para no respirar los vapores, lo que seguía siendo peligroso. Hoy en día se utilizan células de pintura automatizadas, con las cuales se pueden controlar las condiciones ambientales de la célula e implementar extractores de aire con filtros para depurar este aire.



Ilustración 12. Proceso de pintura automatizado



Ilustración 13. Proceso de pintura manual

7. Análisis de alternativas

El objetivo de este proyecto es el control del brazo Braccio mediante Matlab. Por tanto, el lenguaje de programación de alto nivel se encuentra ya definido en las especificaciones. Nótese que aunque hay una gran variedad de programas informáticos en el mercado, y muchos de ellos podrían ser utilizados para controlar este robot, pero la mayoría de ellos no cuentan con la potencia que tiene Matlab para el desarrollo de operaciones matemáticas (las cuales tienen que ser llevadas a cabo para calcular el ángulo de rotación de los servos).

Sin embargo, el robot Braccio no puede ser conectado directamente a Matlab, y se requiere una tarjeta intermedia para hacer de puente entre el robot y el PC con Matlab. A continuación se evaluarán tres alternativas.

7.1. Placa Arduino Uno

Arduino es una compañía internacional que diseña y manufactura placas base para construir dispositivos digitales que puedan interactuar con el mundo real, y además tiene su propio lenguaje basado en C++.

Ya que la placa base que utiliza el brazo robótico Braccio es de Arduino, el entorno de programación facilitado por Arduino tiene la ventaja de ser el mismo que debemos utilizar para programar la placa base, por lo tanto no tendríamos que hacer ningún enlace entre dos lenguajes o programas informáticos diferentes. Otra de las ventajas que tiene es que Arduino es que el entorno de desarrollo es gratuito, y además existe una comunidad que comparte ideas y proyectos desarrollados en este lenguaje. Además de esto es remarcable el precio, pues la placa más básica (Arduino Uno) cuesta 20€ en la página oficial.

La principal desventaja de utilizar Arduino es que Matlab cuenta con muchos más subprogramas y funciones, así como Simulink, una herramienta que es de mucha utilidad para el control.

En la siguiente tabla se valoran de 1 a 5 los aspectos más importantes del programa (siendo 1 muy malo y 5 muy bueno), para así poder compararlo con los demás:

| Coste | Facilidad de uso | Potencial |
|--------------|-------------------------|------------------|
| 5 | 4 | 3 |

Tabla 2. Tabla de calificación de la placa Arduino Uno



Ilustración 14. Placa Arduino Uno

7.2. Placa Raspberry Pi 1

La placa Raspberry Pi es la placa más básica hecha por la fundación Raspberry, la cual, en diferencia a Arduino, manufactura miniordenadores (Arduino manufactura microcontroladores).

En el caso de las placas manufacturadas por la fundación Raspberry, el usuario puede elegir el lenguaje en el que quiera escribir el código, lo cual es una gran ventaja si el usuario sabe programar en algún lenguaje en concreto, o tiene funciones ya hechas en algún lenguaje. Para ello, el usuario tendrá que instalar el sistema operativo del miniordenador así como el lenguaje en el que quiera programar.

En torno a las placas Raspberry Pi también existe una comunidad que ayuda al usuario con los problemas que pueda tener, así como ofrecerle diferentes retos y proyectos. Aunque la comunidad en torno a Raspberry Pi sea amplia, cabe destacar que no es tan grande como la de Arduino.

En el caso de las placas Raspberry Pi, ya que son miniordenadores y no microcontroladores, son más completos y pueden realizar tareas más complejas, pero eso conlleva un aumento de precio respecto a Arduino. La placa más básica es la Raspberry Pi 1 y cuesta 30€.

Ya que en nuestro caso el código que se ejecutará en la placa base será corto y sencillo, realmente no necesitaremos muchas de las características que nos ofrece esta placa, pero aun así, es una alternativa viable.

En la siguiente tabla se valoran de 1 a 5 los aspectos más importantes del programa (siendo 1 muy malo y 5 muy bueno), para así poder compararlo con los demás:

| Coste | Facilidad de uso | Potencial |
|--------------|-------------------------|------------------|
| 3 | 3 | 4 |

Tabla 3. Tabla de calificación de la placa Raspberry Pi 1



Ilustración 15. Placa Raspberry Pi 1

7.3. Placa hecha a medida

Otra posibilidad es la de hacer una placa o encargar una placa hecha a medida.

Para hacer una placa a medida se tendrían que tener nociones avanzadas de electrónica, y habría que diseñar la placa de acuerdo con los requerimientos y permita el enlace entre el robot Braccio y el programa informático Matlab..

El principal beneficio de esta placa hecha a medida sería su facilidad de uso, pues podríamos encargarla de manera que no tuviéramos que programar nada. En cuanto a lo demás, serían todo desventajas. Al ser una placa tan simple, su potencial para futuros trabajos y modificaciones sería nulo, y aun siendo simple, su precio sería mucho mayor que el de la Raspberry Pi 1 o el de Arduino Uno.

En la siguiente tabla se valoran de 1 a 5 los aspectos más importantes del programa (siendo 1 muy malo y 5 muy bueno), para así poder compararlo con los demás:

| Coste | Facilidad de uso | Potencial |
|--------------|-------------------------|------------------|
| 1 | 5 | 1 |

Tabla 4. Tabla de calificación de la placa hecha a medida

7.4 Elección de la solución

Para elegir la placa que se va a utilizar para el control del brazo robótico Braccio se tendrán en cuenta los tres parámetros de las tablas. Para comparar una placa con otra en su totalidad, se hará la suma de los puntos obtenidos en cada apartado, y así se podrán comparar las placas fácilmente.

| | Coste | Facilidad de uso | Potencial | Total |
|----------------------|--------------|-------------------------|------------------|--------------|
| Placa Arduino Uno | 5 | 4 | 3 | 12 |
| Placa Raspberry Pi 1 | 3 | 3 | 4 | 10 |
| Placa hecha a medida | 1 | 5 | 1 | 7 |

Como se puede observar, la placa Arduino es la placa que más puntos ha obtenido debido a su precio y simplicidad. Por lo tanto, la placa que se utilizará será la Arduino Uno

En resumen, dado que lo que nuestra placa base deberá de hacer es muy simple, se optará por la placa más simple y barata de todas: Arduino Uno.

8. Metodología

En este apartado se explicará el funcionamiento de la Interfaz gráfica de usuario (GUI) creada, y se detallarán los programas utilizados tanto al abrir interfaces como al clicar en botones, así como su función.

8.1. Visión global de la aplicación: panel principal

De acuerdo con el objetivo de este proyecto, se ha requerido el desarrollo de una interfaz gráfica con diferentes botones para acceder a diferentes funciones que se tenían que realizar. Para el desarrollo de la interfaz gráfica que ha cogido como modelo la interfaz de EduBot, y se ha modificado para el caso del robot Braccio, quitando así funcionalidades que no eran necesarias.

Cuando se abre la interfaz nos encontramos con diferentes botones para la realización de los diferentes modos de control. Al clicar en estos botones se abrirán otras ventanas para así poder controlar el robot de la manera que se requiera.

Para un entendimiento más claro de lo que se va a explicar a continuación se muestra la ventana inicial de la interfaz gráfica.

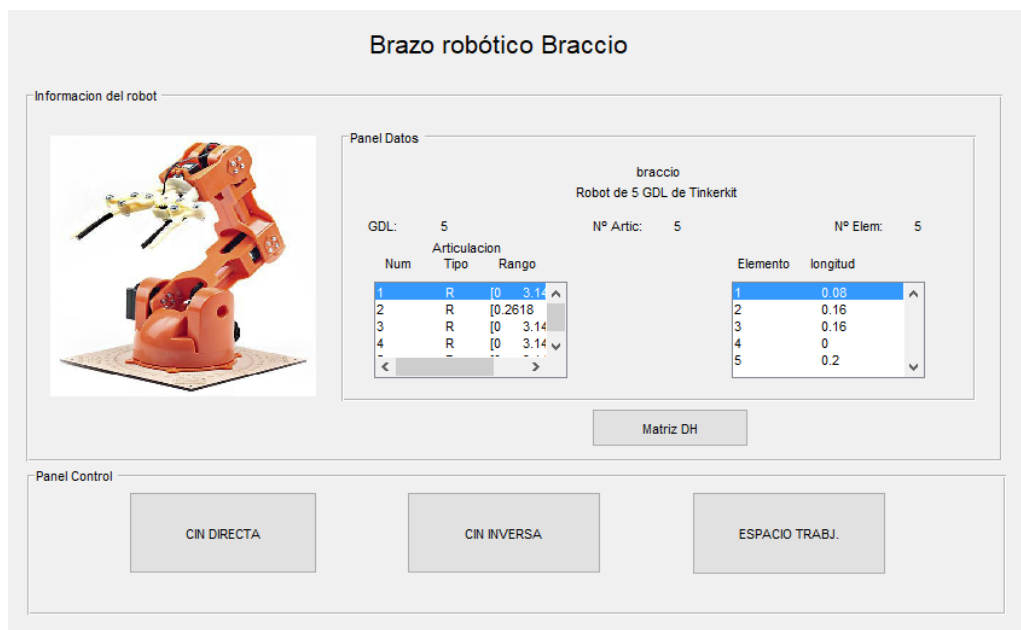


Ilustración 16. Ventana inicial de la interfaz gráfica

Podemos observar que la ventana inicial consta de 4 botones, además de una imagen, dos listas y algo de texto.

La imagen mostrada es una imagen del robot Braccio que se va a controlar, y las dos listas muestran lo siguiente:

- Lista 1: muestra los grados de libertad del brazo robótico Braccio, así como su tipo (todos rotacionales en este caso) y el rango de cada uno de ellos (en radianes).
- Lista 2: muestra el número de elementos además de la longitud de cada uno de ellos.

Los botones mostrados en la imagen abren diferentes ventanas para la realización de diferentes tareas. Aquí explicaremos brevemente cuáles son esas tareas:

- Cin. Directa: Realizar la cinemática directa y enviar los ángulos requeridos por los servos a la placa.
- Cin. Inversa: Realizar la cinemática inversa y enviar los ángulos requeridos por los servos a la placa.
- Espacio Trabajo: Muestra el espacio de trabajo en 2D de cualquier brazo robótico entre 1 y 4 GDL.
- Matriz DH: Muestra la matriz de Denavit Hartenberg del brazo robótico Braccio.

Cada una de estas funcionalidades será detallada a continuación.

8.1.1. Funciones en el panel principal

Este panel principal tiene la simple función de dejar elegir al usuario la función que quiere realizar, por lo tanto simplemente tiene que cargar la información del robot y pasársela a la siguiente ventana que se abra.

Al inicializarse el programa se ejecuta la función `ControlBraccio1_OpeningFcn`. Esta función busca el archivo con extensión `.mat` que contiene los datos del robot Braccio, y mediante el programa `cargar_robot` los almacena en una variable global, para así poder utilizarla en los siguientes paneles. Además de cargar estos datos para poder hacer los cálculos matemáticos, esta función también hace la conexión entre el ordenador y el robot Braccio mediante el puerto USB que se esté utilizando, para que más tarde se le puedan pasar a este las coordenadas articulares de los servos.

Para establecer la conexión deberá haber un cable que esté conectado a la placa base y al ordenador, además deberá haber un alimentador conectado a la placa base también para que cuando se le envíe la orden a la misma el robot se mueva.

Cuando el usuario clics sobre cualquiera de los botones, el programa vuelve a cargar los datos del robot y se los pasa al siguiente panel mediante una función.

A continuación se muestra cada botón con la función correspondiente que hace aparecer el siguiente panel:

- **Cin. Directa:** "Demo_Window_CIn_Direct".
- **Cin. Inversa:** "Demo_Window_CIn_Invers".
- **Espacio Trabajo:** "Espacio_Trabajo_2D".
- **Matriz DH:** "Demo_Window_DH".

8.2. Panel Cinemática Directa

El panel de cinemática directa es el panel que se abre al clicar sobre el botón de "CIN DIRECTA" en el panel principal, y su función es desarrollar la cinemática directa, es decir, le daremos los valores que queramos a las articulaciones y obtendremos la posición (en coordenadas cartesianas) en la que se encontrará nuestro punto final (en este caso, la pinza) para esos valores.

El panel de cinemática directa se muestra en la siguiente imagen.

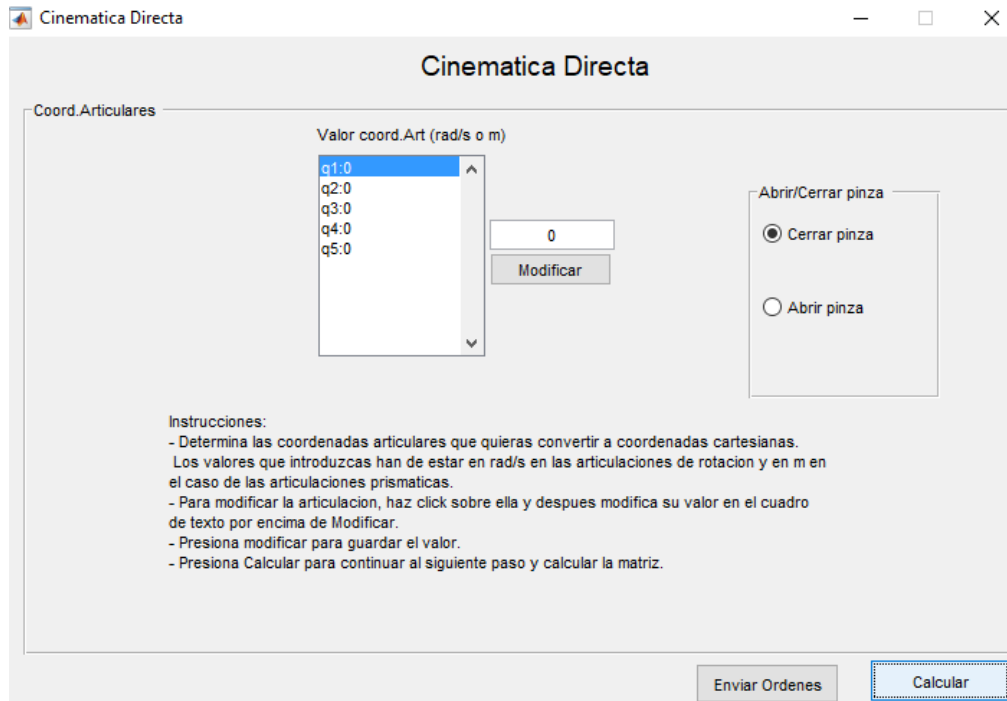


Ilustración 17. Panel de cinemática directa

Como se puede ver en la imagen, el panel es bastante simple e intuitivo. Ya que en nuestro caso disponemos de 5 servos, estos se muestran en una lista en el centro del panel (q1 equivale al primer servo, q2 al segundo etc.), con valores iniciales de 0rad/s.

Para modificar los valores articulares de los servos, tal como está escrito en el apartado de instrucciones en el panel, simplemente tendremos que clicar en el servo que queramos modificar, escribir el nuevo valor en el recuadro que se encuentra a la derecha y clicar sobre “modificar”.

Además de modificar los valores articulares, también se puede elegir entre cerrar o abrir la pinza gracias al recuadro de la derecha, lo cual no tendrá efecto sobre las coordenadas cartesianas finales.

Una vez establecidas las coordenadas articulares de nuestros servos, se deberá presionar el botón “calcular” que nos llevará al siguiente panel, el cual nos mostrará las coordenadas cartesianas de cualquiera de nuestros servos.

Se muestra a continuación dicho panel.

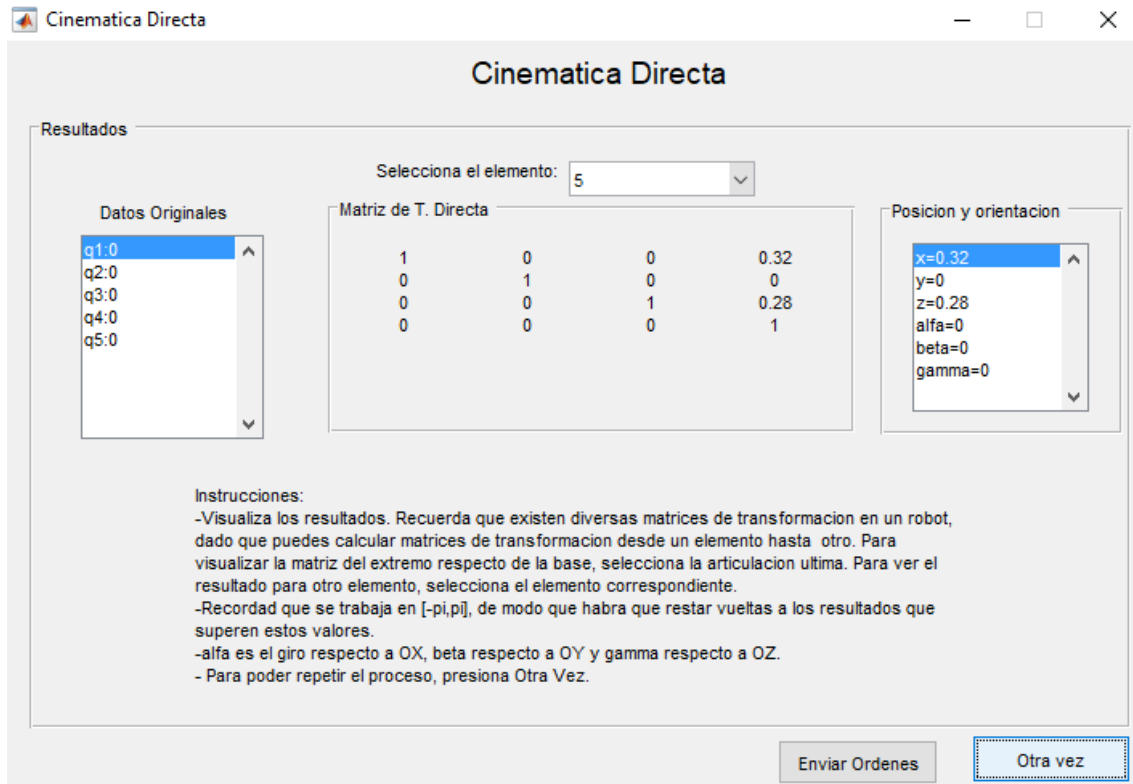


Ilustración 18. Segundo panel de cinemática directa

Como se puede ver en la imagen, este panel simplemente muestra la posición y orientación del elemento seleccionado en el recuadro de arriba, así como su matriz de transformación para las coordenadas articulares que se han establecido previamente. Cabe destacar que en este panel podemos cambiar el elemento del cual queremos saber la posición y orientación, pero está configurado para que por defecto aparezcan las coordenadas cartesianas del último elemento, pues normalmente es el que se desea ubicar.

Además de esto, en el panel también se pueden ver las instrucciones a seguir una vez vistas las coordenadas cartesianas, así como unas advertencias sobre los datos y resultados.

Una vez establecidas las coordenadas articulares, clicando sobre “calcular” podremos ver las coordenadas cartesianas en las que se encontraría la pinza del brazo robótico Braccio y, finalmente, para poder controlar el robot y mover sus articulaciones de acuerdo a lo que hemos establecido en el programa, simplemente tendremos que darle al botón “Enviar Órdenes”.

Cabe destacar que al clicar sobre el botón “Enviar Órdenes” se envían las coordenadas articulares establecidas por el usuario, pero es posible que estas coordenadas estén

fuera del rango del robot Braccio. Si esto ocurriese, saldría un texto en Matlab, avisándonos de que ha habido un error, y se establece la coordenada posible más próxima. Por ejemplo, la base solo puede tener valores articulares entre 0 y 180°, pero si le damos un valor de -90° se movería por defecto a 0° y aparecería un texto en Matlab avisando de ello.

8.2.1. Cálculo del Problema cinemático directo

En esta sección se detalla la teoría y desarrollo del problema cinemático directo del robot. El problema cinemático directo trata de conseguir la posición y orientación del elemento final de un robot respecto al origen de coordenadas cartesianas, que normalmente se haya en la base o primera articulación de dicho robot, teniendo como dato de entrada las coordenadas articulares que se le van a imponer a las articulaciones del robot.

En la siguiente imagen podemos ver los datos de entrada y los datos que son objetivo en el caso de la cinemática directa y la cinemática inversa.

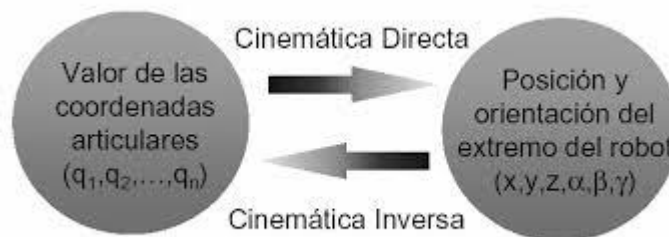


Ilustración 19. Datos de entrada y salida de la cinemática directa e inversa

Para resolver el problema de cinemática directo se utilizan las matrices de transformación, en las que se utiliza la notación de Denavit-Hartenberg, la cual expresa la posición y orientación de la articulación i respecto a la articulación $i-1$ mediante los siguientes datos:

1. θ_i : Ángulo de rotación alrededor del eje z_{i-1} desde x_{i-1} a x_i .
2. d_i : Distancia de traslación a lo largo de z_{i-1} .
3. a_i : Distancia de traslación a lo largo de x_i .

4. α_i : Ángulo de rotación alrededor del eje x_i desde z_{i-1} a z_i .

En la siguiente imagen se pueden visualizar tres articulaciones, junto a sus ejes y sus parámetros de Denavit-Hartenberg.

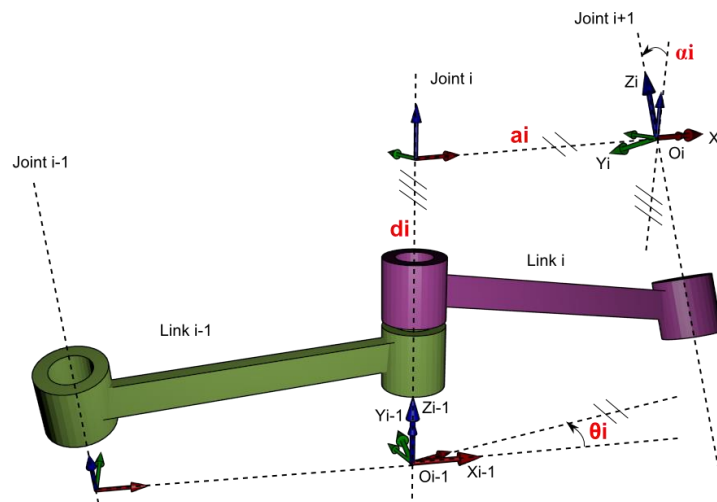


Ilustración 20. Imagen de la notación de Denavit-Hartenberg

Una vez se tengan los valores de Denavit-Hartenberg de una articulación a otra, se deberá de rellenar la matriz con estos valores, siendo la matriz la siguiente:

$${}^{i-1}A_i = \begin{pmatrix} \cos \theta_i & -\cos \alpha_i \cdot \sin \theta_i & \sin \alpha_i \cdot \sin \theta_i & a_i \cdot \cos \theta_i \\ \sin \theta_i & \cos \alpha_i \cdot \cos \theta_i & -\sin \alpha_i \cdot \cos \theta_i & a_i \cdot \sin \theta_i \\ 0 & \sin \alpha_i & \cos \alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Ilustración 21. Matriz de Denavit-Hartenberg

Para obtener la posición y orientación del elemento final del robot primero se tendrá que obtener la matriz de transformación de dicho elemento respecto al origen de coordenadas cartesianas. Esto se logra postmultiplicando las matrices, como podemos ver a continuación:

$${}^0T_5 = {}^0T_1 * {}^1T_2 * {}^2T_3 * {}^3T_4 * {}^4T_5$$

Finalmente, la matriz 0T_5 contendrá la posición y orientación del elemento final 5 respecto al origen de coordenadas cartesianas que se encontrará en 0. Los parámetros de Denavit-Hartenberg del brazo robótico Braccio se detallan en la siguiente imagen.

| alpha | A | Thetha | D |
|---------|------|--------|------|
| 1.5708 | 0 | 0 | 0.08 |
| 0 | 0.16 | 0 | 0 |
| 0 | 0.16 | 0 | 0 |
| -1.5708 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0.2 |

Ilustración 22. Matriz DH del robot Braccio

8.2.2. Funciones en el panel de Cinemática Directa

Este panel tiene la función de desarrollar la cinemática directa, es decir, de obtener la posición y orientación en la que se ubicará la pinza del robot teniendo como datos de entrada las coordenadas articulares.

Como se ha dicho anteriormente, este panel se abre clicando en el botón “CIN DIRECTA” del panel principal, mediante el cual se ejecuta la función “Demo_Window_CIn_Direct”, que abre el nuevo panel de cinemática directa, además de pasarle los datos del robot Braccio.

Una vez dentro de este panel, el usuario especificará las coordenadas articulares del robot y presionará el botón “calcular”, mediante el cual se ejecutará la función “Calcular_Cinemática_Directa” a la que se le pasan los datos del robot y las coordenadas articulares. Esta función chequeará que los datos sean correctos, y a continuación creará las matrices mediante la función “denavit”, a la cual se le pasarán los datos necesarios (d, a, θ , α) para construir la matriz de acuerdo con la notación de Denavit-Hartenberg.

Una vez se hayan construido las matrices, se irán postmultiplicando para finalmente obtener la matriz final 0T_5 que exprese la posición y orientación del elemento final respecto al origen.

Finalmente, se obtendrán los datos de esta matriz y se mostrarán en formato de lista como puede verse en la ilustración 15.

A parte de la función principal de resolver la cinemática directa, este panel también tiene la función de enviar los datos de las coordenadas articulares al robot Braccio, para que así este se pueda mover para que el elemento final se ubique en la posición calculada. Esta función llamada “Braccio_enviar_coordenadas” se ejecuta al clicar

en el botón “Enviar órdenes”, recogiendo las coordenadas articulares de las articulaciones y el estado de la pinza y mandándolos vía puerto USB a la placa base.

8.3. Panel Cinemática Inversa

El panel de cinemática directa es el panel que se abre al clicar sobre el botón de “CIN INVERSA” en el panel principal, y su función es desarrollar la cinemática inversa, es decir, le escribiremos las coordenadas cartesianas en las que queremos que se encuentre nuestra pinza y nos devolverá las coordenadas articulares, que luego utilizaremos para enviárselas a la placa base, y por consiguiente para que la pinza del brazo robótico se mueva a la posición que se le ha establecido.

En la siguiente imagen podemos ver el panel de la cinemática inversa.

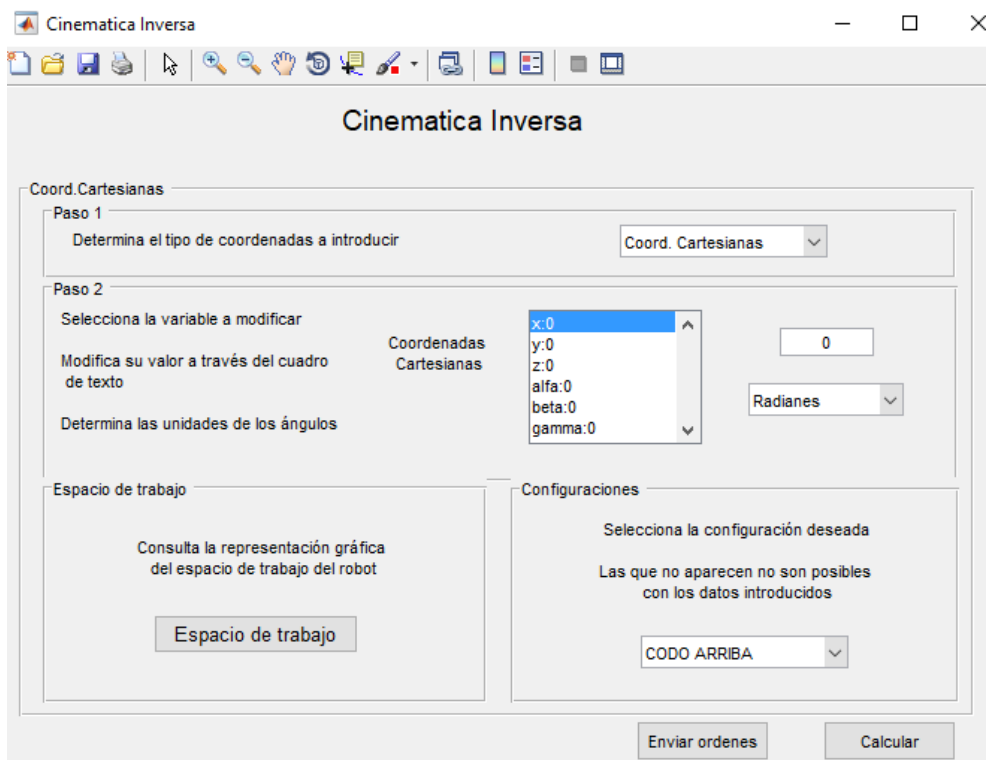


Ilustración 23. Panel de cinemática inversa

Como se puede ver en la imagen, la cinemática inversa consta dos pasos: determinación de coordenadas a introducir y la introducción de dichas coordenadas.

Primero se tiene que elegir el tipo de coordenadas. Se puede elegir entre “Coordenadas cartesianas” y “Coordenadas articulares”.

Sea cual sea el tipo de coordenadas elegido, el procedimiento es el mismo: tal como se indica en el paso 2, se tendrá que seleccionar la variable a modificar y cambiar su valor mediante el cuadro de texto que se encuentra a la derecha, teniendo en cuenta que se puede elegir la unidad de los ángulos entre radianes o grados, pero que las longitudes siempre estarán en metros.

Una vez establecidas las coordenadas se debe clicar en el botón “calcular”, mediante el cual se abrirá una nueva ventana que mostrará las coordenadas articulares necesarias para que la pinza del robot llegue a las coordenadas cartesianas que se han indicado.

Cabe destacar que en el panel existen dos partes que no se han mencionado todavía: Espacio de trabajo y configuraciones. Estas dos partes no están operativas a día de terminar este proyecto, pero ya que estaban en la interfaz heredada de EduBot, se han dejado con intención de seguir mejorando este proyecto en un futuro. Así, la funcionalidad de estas dos partes sería la siguiente:

- Espacio de Trabajo: clicando en este botón se mostraría en 3D el espacio de trabajo del brazo robótico Braccio, es decir, los lugares a los que puede llegar con la pinza
- Configuraciones: muchas veces existe más de una combinación de coordenadas articulares posibles para que la pinza se sitúe en las coordenadas cartesianas establecidas. Por ello, en configuraciones debería aparecer las posibilidades “Codo arriba” y “Codo abajo”, para así poder elegir, cuando sea posible, si queremos que el codo de nuestro brazo robótico esté arriba o abajo. A continuación se ilustran las dos posibilidades mediante imágenes.

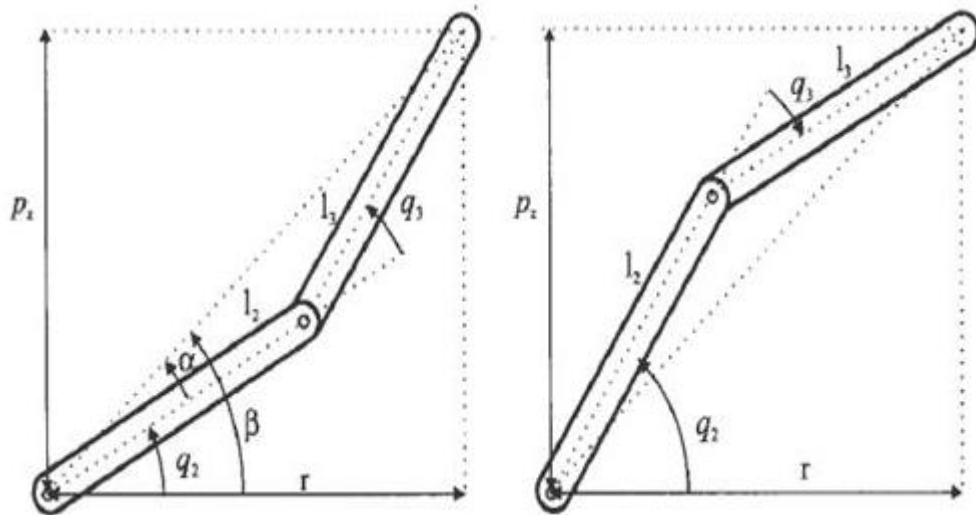


Ilustración 24. Codo abajo (izquierda) y codo arriba (derecha)

Ya que a día de hoy estas funciones todavía no están activas, es indiferente seleccionar “Codo arriba” o “Codo abajo” en la sección de configuraciones, y al clicar en el botón de “Espacio de trabajo” no sucede nada.

Prosiguiendo con lo anterior a este inciso sobre estas dos funciones del panel, una vez se hayan introducido las coordenadas cartesianas en el formato indicado, se debe clicar en el botón “calcular”, lo cual nos llevará a la ventana que se muestra a continuación.

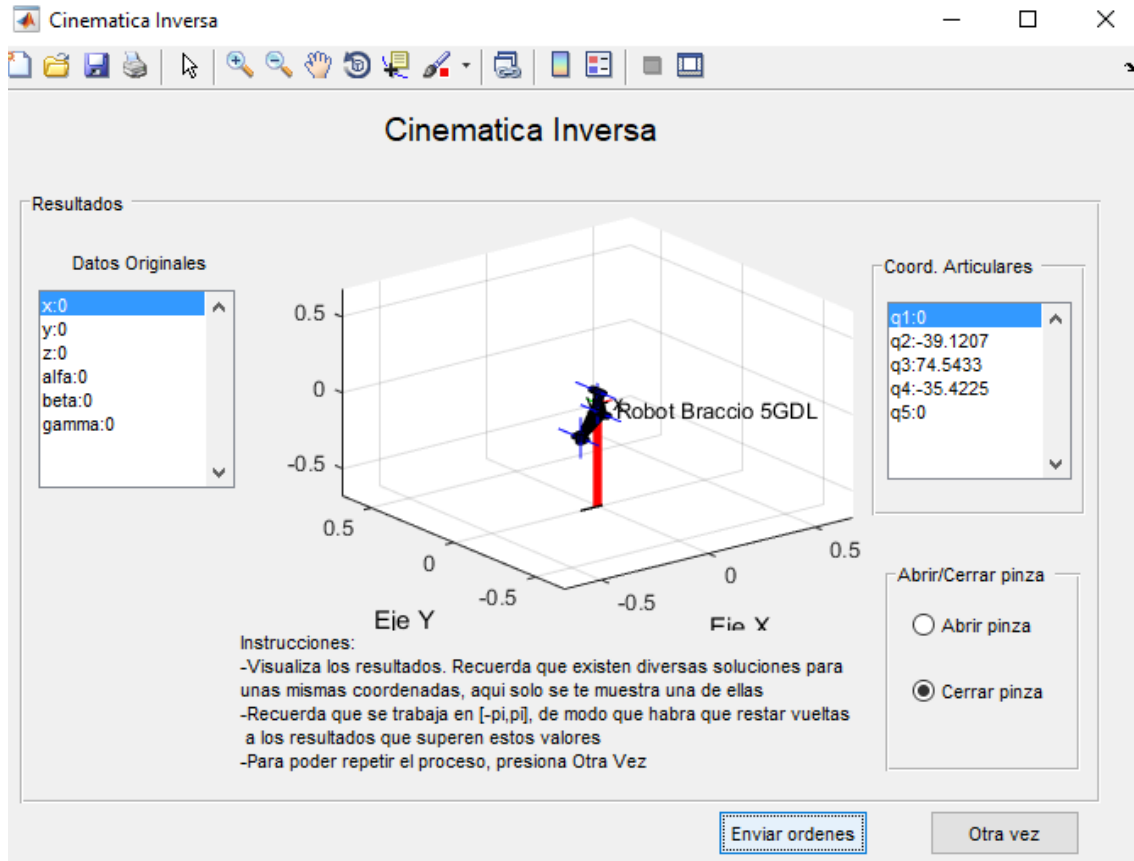


Ilustración 25. Segundo panel de cinemática inversa

Como se puede ver en la imagen, en este segundo panel se muestran diversas listas, imágenes y botones.

A la izquierda se muestran los datos originales escritos por el usuario en la anterior ventana en forma de lista, y a la derecha se muestran las coordenadas articulares de cada uno de los servos en forma de lista también, las cuales han tenido que ser calculadas por el programa.

En el centro del panel se puede ver una representación 3D del robot Braccio en la posición estipulada por las coordenadas articulares. Cabe destacar que esta representación está hecha simplificando los diferentes elementos del robot mediante líneas. Además de esto, en la parte inferior también se pueden apreciar unas instrucciones a seguir por el usuario, parecidas a las que se hayan en el segundo panel de la cinemática directa.

En este panel también, al igual que en la cinemática directa, hay un recuadro para poder especificar si la pinza debe estar abierta o cerrada. Si el usuario ve algún tipo de problema con las coordenadas articulares, o simplemente quiere cambiar las coordenadas cartesianas originales, simplemente deberá de presionar el botón “otra vez” para que se abra de nuevo el primer panel de la cinemática directa. Una vez el usuario esté de acuerdo con los resultados y quiera plasmarlos en el brazo robótico Braccio, que deberá estar conectado al ordenador, simplemente deberá de presionar el botón “Enviar órdenes”.

Se tiene que recalcar que el programa que calcula la cinemática inversa no tiene en cuenta las limitaciones reales de nuestro robot, por lo tanto es posible que las coordenadas articulares calculadas no sean posibles para el robot Braccio, en cuyo caso aparecerá una ventana con el siguiente mensaje de error:

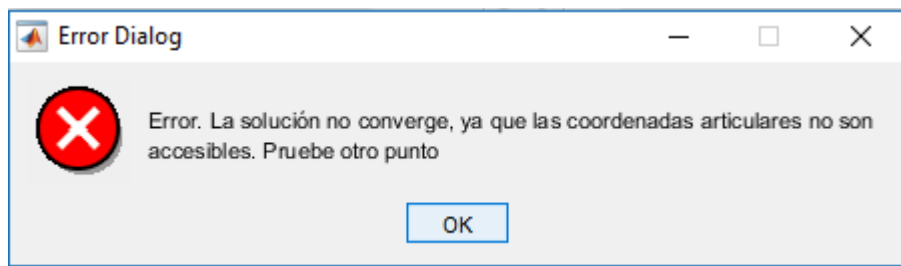


Ilustración 26. Mensaje error cinemática inversa

8.3.1. Cálculo del Problema cinemático inverso

El problema cinemático inverso trata de obtener las coordenadas articulares de una cadena de articulaciones que debe tener un robot para que el elemento final llegue a la posición y orientación especificadas. Este problema es más complejo que el de la cinemática directa, ya que consiste en la resolución de una serie de ecuaciones que normalmente no tienen una única solución, mientras que en la cinemática directa simplemente se deben postmultiplicar matrices.

Dependiendo de la cadena de articulaciones la resolución de la cinemática inversa puede requerir simplemente la resolución de una serie de ecuaciones explícitas, pero normalmente las ecuaciones suelen ser implícitas y se necesita de un método iterativo para poder resolver el problema. También hay que tener en cuenta que el punto en el que se quiera situar el elemento final tiene que estar dentro del espacio de trabajo, porque si no es así será imposible que el robot llegue a tal punto y la cinemática inversa no tendrá solución.

Para resolver la cinemática inversa del robot se utiliza el programa "Ikin". A este programa se le tienen que pasar los datos del robot, la posición y orientación del elemento final y una máscara en caso de que sea necesario (será necesario para robots con menos de 6 grados de libertad, por lo que la necesitaremos para el robot Braccio). Este programa itera mediante el método de Newton-Raphson para así obtener una solución al problema de la cinemática inversa y las coordenadas articulares necesarias para que el elemento final se sitúe donde se ha especificado. Para la iteración se necesitará un punto de partida que puede ser especificado previamente, y en caso de que no sea especificado se tomará el valor nulo por defecto.

8.3.2. Funciones en el panel de Cinemática Inversa

Este panel aparece cuando el usuario clicca sobre el botón "CIN INVERSA" en el panel principal, mediante la función "Demo_Window_CIn_Invers" a la que se le transmiten los datos del robot Braccio.

Al inicializarse el panel se ejecuta la función "Demo_Window_CIn_Invers_OpeningFcn" que chequea que los datos del robot Braccio sean correctos, y si no lo fueran daría un mensaje de error.

Este problema de la cinemática inversa se resuelve cuando el usuario pulsa el botón "calcular" mediante la función "Calcular_Cinematica_Inversa_2". Esta función recoge los datos del robot, las coordenadas cartesianas establecidas y la configuración (codo arriba / codo abajo que actualmente no se utiliza), crea una máscara si fuera necesaria y ejecuta la subfunción "ikine", la cual ha sido detallada en el anterior apartado.

Cabe destacar que si la subfunción "ikine" no encuentra solución al problema, o bien las coordenadas articulares no están dentro de los límites establecidos, aparecerá un mensaje de error informando al usuario del tipo de problema.

Una vez se ha resuelto el problema cinemático inverso y se tienen las coordenadas articulares, el usuario puede hacer que el robot se mueva presionando sobre el botón "Enviar órdenes" que ejecuta la función "Braccio_enviar_coordenadas", la cual, como se ha explicado en el apartado de cinemática directa, envía estas coordenadas a la placa base del robot para que este se mueva.

8.4. Panel de Espacio de Trabajo

El panel de Espacio de Trabajo permite al usuario visualizar el espacio de trabajo de cualquier brazo robótico que tenga entre 1 y 4 articulaciones y trabaje en 2 dimensiones. Para ello el usuario simplemente tendrá que introducir los datos de cada una de las articulaciones

En la siguiente imagen podemos ver la ventana que surge al clicar sobre “ESPACIO TRABJ.” en el panel principal.

Generador de Espacio de Trabajo en 2D

Num. Articulaciones

2

Articulacion 1

Inicio Final Longitud

Articulacion 2

Articulacion (grados) Longitud Elemento

Inicio Final Longitud

Articulacion 3

Articulacion (grados) Longitud Elemento

Inicio Final Longitud

Articulacion 4

Articulacion (grados) Longitud Elemento

Inicio Final Longitud

Dibujar Espacio de Trabajo

Ilustración 27. Panel Espacio de Trabajo

Como se puede apreciar en la imagen, en la parte superior de este panel hay un recuadro que permite al usuario elegir el número de articulaciones que tiene su brazo robótico. Una vez elegidos el número de articulaciones (1 mínimo y 4 máximo), se podrá

especificar la longitud de cada una de las articulaciones, así como el mínimo y máximo de sus coordenadas articulares en grados.

Una vez se han rellenado todos los campos, el usuario deberá de clicar sobre el botón “Dibujar Espacio de Trabajo” para que el programa genere una figura en 2D del espacio de trabajo del brazo robótico.

Se puede observar que el brazo robótico Braccio no cumple con las condiciones impuestas, pues trabaja en 3D y tiene 5 articulaciones, pero aun así, se puede obtener su espacio de trabajo en 2D si se bloquea la base. Si bloqueamos la base, la primera coordenada articular del robot Braccio, nos encontraremos con un robot que tiene 4 articulaciones y que trabaja en 2D, por lo tanto podemos obtener su espacio de trabajo. Si bien es verdad que este espacio de trabajo es solamente el de una coordenada articular de la base, para obtener el espacio de trabajo en 3D simplemente se tendría que hacer girar el plano que conforma este espacio de trabajo 180° frente al eje Z, consiguiendo así un volumen, el que sería el espacio de trabajo real del brazo robótico Braccio. De todas formas, se cree más visual el espacio de trabajo en 2D, el cual podemos observar en la siguiente imagen.

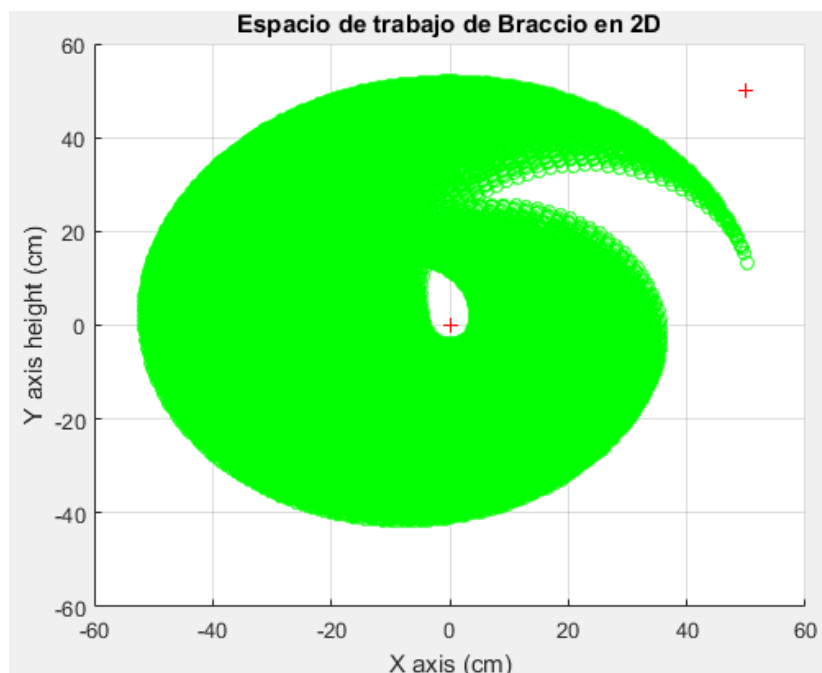


Ilustración 28. Espacio de trabajo de Braccio en 2D si se bloquea la articulación 1

8.4.1 Funciones en el panel de Espacio de Trabajo

Al clicar sobre el botón “ESPACIO TRABJ.” en el panel principal se ejecuta la función “Demo_Window_EspacTrabajo” que abre el panel de Espacio de Trabajo y le pasa a este los datos del robot Braccio.

Para obtener el espacio de trabajo de acuerdo a los datos introducidos se deberá clicar sobre el botón “Dibujar Espacio de Trabajo”, que ejecuta la función “Espacio_Trabajo_2D”. Esta función recoge todos los datos introducidos en el panel y se los envía a la subfunción “seleccion2D”, una función iterativa que marca con puntos verdes todos los puntos a los que el robot puede acceder.

8.5. Panel de Matriz de Denavit-Hartenberg

Este panel surge al clicar sobre el botón “Matriz DH” en el panel principal, y simplemente nos muestra la matriz de Denavit-Hartenberg para el brazo robótico Braccio.

En la siguiente imagen se puede ver tanto la ventana que surge.

| alpha | A | Thetha | D | R/P |
|---------|------|--------|------|-----|
| 1.5708 | 0 | 0 | 0.08 | R |
| 0 | 0.16 | 0 | 0 | R |
| 0 | 0.16 | 0 | 0 | R |
| -1.5708 | 0 | 0 | 0 | R |
| 0 | 0 | 0 | 0.2 | R |

Ilustración 29. Panel de Matriz DH

Como se puede ver, en este panel podemos visualizar en la mitad del panel los valores de la matriz de Denavit-Hartenberg así como el tipo de articulación, que en este caso será rotacional para todos los casos. El único botón que se encuentra en este panel es el de “Cerrar ventana”, que tal como su nombre indica simplemente cierra la ventana.

El objetivo de este panel es simple: indicar al usuario familiarizado con la metodología de Denavit-Hartenberg cuáles son los valores para el robot Braccio.

8.5.1 Funciones en el panel de Matriz DH

El panel de Matriz DH se abre mediante la función “`Demo_Window_DH`”, la cual se ejecuta cuando el usuario clica sobre el botón “Matriz DH” en el panel principal, pasándole los datos del robot Braccio.

Este panel simplemente coge de los datos del robot la matriz de Denavit-Hartenberg y se la muestra al usuario, por lo que no tiene ninguna otra función adicional.

9. Descripción de tareas

En este apartado se especificarán de forma cronológica las tareas que se han tenido que realizar para acabar este proyecto, así como el tiempo y material necesarios para ello. Para la realización del proyecto se contará con 21.5 semanas y se estimará una media de trabajo de 2 horas diarias, lo que harán 215 horas de trabajo total teniendo en cuenta cinco días de trabajo laborable por semana.

9.1. Seguimiento del proyecto con el tutor

El proyecto comienza el día en el que se habla con el tutor para la adjudicación del mismo. En la reunión se acuerda con el tutor el objetivo del proyecto, así como el material y programas que van a ser necesarios y que se podrán utilizar para alcanzar durante el proyecto. Además de este primer día, se mantiene el contacto con el tutor durante todo el proyecto, para poder consultar dudas y problemas que surgen a lo largo de este. En general las reuniones con el tutor son cortas y concisas, por lo que a lo largo del curso se estimará su duración en 1 día, es decir, 0.2 semanas.

9.2. Adquisición de nociones básicas de Matlab y Arduino

Para la realización del proyecto se han tenido que adquirir nociones sobre programación en lenguaje Arduino y Matlab.

- Arduino: Se tendrá que aprender qué es y cómo funciona tanto la placa base de Arduino así como el lenguaje de Arduino en sí. Para ello se ha utilizado un libro sobre Arduino y la comunidad de Arduino online. Se estima el tiempo necesario en 1 semana.
- Matlab: Se han tenido que refrescar conceptos sobre Matlab, así como aprender unos nuevos y aprender cómo se hace una Interfaz Gráfica de Usuario (GUI). Para ello se han utilizado unos manuales, la comunidad de Matlab online y foros. Se estima el tiempo necesario para esto en 2 semanas.

9.3. Entendimiento del programa EduBot

Para controlar el brazo robótico Braccio se debe de hacer una interfaz gráfica así como programas que hagan la parte matemática necesaria. Para ello se ha tomado como referencia el programa EduBot, que cuenta con programas e interfaz gráfica para el control de otros brazos robóticos. Por ello ha sido necesario el entendimiento del funcionamiento de la interfaz gráfica de EduBot y también muchos de sus programas. Además de esto, ha sido una referencia clara para ver cómo se tendría que programar y que tipo de funciones se tendrían que utilizar. Se estima el tiempo necesario para entender completamente este programa en 4 semanas.

9.4. Estudio preliminar del robot

Ya que el objetivo de este proyecto era el de controlar el brazo robótico Braccio, se tenía que resolver la cinemática directa e inversa. Por lo tanto se han tenido que refrescar conceptos y estudiar a cerca de esto. Para ello se ha estudiado cómo hacer la matriz de Denavit-Hartenberg del robot Braccio y cómo si se podría hacer la cinemática inversa de manera no iterativa. Se estima el tiempo necesario para este estudio en 1 semana

9.5. Programación

Para poder controlar el robot se ha tenido que programar una interfaz gráfica con sus funciones, así como la programación básica en Arduino para la placa base.

9.5.1. Programación en Arduino en la placa base

En la placa base se ha tenido que hacer un programa que recoja los datos que le son mandados mediante USB, chequeé que son correctos y se los transfiera al brazo robótico para así poder moverlo. Aunque el programa final es bastante sencillo, se han tenido problemas a la hora de transmitir datos mediante USB. Se estima el tiempo necesario para la realización de este programa en 1,5 semanas

9.5.2. Programación en Matlab

Para hacer la GUI en Matlab así como las funciones necesarias se ha tomado como ejemplo el programa EduBot y se le han modificado tanto la GUI como las funciones que tenía, además de crear unas nuevas. Debido a que muchas de las funciones matemáticamente complejas que se utilizan para controlar el robot se han cogido del programa EduBot, la realización de esta programación no ha sido tan compleja. Se estima el tiempo necesario para la realización y modificación de todos estos programas en 6 semanas.

9.6. Montaje del robot

Ya que el robot Braccio venía por piezas y desmontado, se ha tenido que montar de acuerdo a las instrucciones que traía. Al ser un robot pequeño y didáctico el montaje ha sido sencillo, estimando su duración en 1 día (0.2 semanas).

9.7. Validación del programa

Una vez se haya hecho la parte de programación necesaria para el control del brazo robótico Braccio y se tenga dicho robot montado y conectado al ordenador, se ha probado el programa para ver si el robot responde de acuerdo a lo establecido. Como era de esperar ha habido algún que otro problema debido al montaje o a la conexión entre Arduino y Matlab, pero nada grave. Se estima la duración de la validación así como de la corrección de errores en 3 días (0.6 semanas)

9.8. Redacción del documento

Finalmente, una vez se haya terminado el programa necesario para controlar el brazo robótico Braccio se debe de redactar el informe del proyecto. La redacción de este se ha dejado para último momento, para así poder concentrarse en la parte de programación hasta acabarla. Para la redacción del documento se ha tenido que buscar información adicional para así realizar el contexto y el alcance del proyecto. Se estima la duración de la redacción en 5 semanas.

10. Diagrama de Gantt

Mediante el diagrama de Gantt se puede ver las tareas necesarias así como su duración y predecesoras de manera cronológica.

En las siguientes dos imágenes se va a mostrar una tabla del diagrama de Gantt y el diagrama de Gantt mismo.

| Nombre actividad | Fecha inicio | Duración en días | Fecha fin |
|---|--------------|------------------|------------|
| 1.- Seguimiento del proyecto con el tutor | 19/02/2018 | 150 | 19/07/2018 |
| 2.1.- Nociones básicas: Arduino | 20/02/2018 | 7 | 27/02/2018 |
| 2.2.- Nociones básicas: Matlab | 27/02/2018 | 14 | 13/03/2018 |
| 3.- Entendimiento del programa EduBot | 13/03/2018 | 28 | 10/04/2018 |
| 4.- Estudio preliminar del robot | 10/04/2018 | 7 | 17/04/2018 |
| 5.1.- Programación en Arduino | 17/04/2018 | 10,5 | 27/04/2018 |
| 5.2.- Programación en Matlab | 27/04/2018 | 42 | 08/06/2018 |
| 6.- Montaje del robot | 08/06/2018 | 1,4 | 09/06/2018 |
| 7.- Validación del programa | 09/06/2018 | 4,2 | 14/06/2018 |
| 8.- Redacción del documento | 14/06/2018 | 35 | 19/07/2018 |

Tabla 5. Tabla del diagrama de Gantt

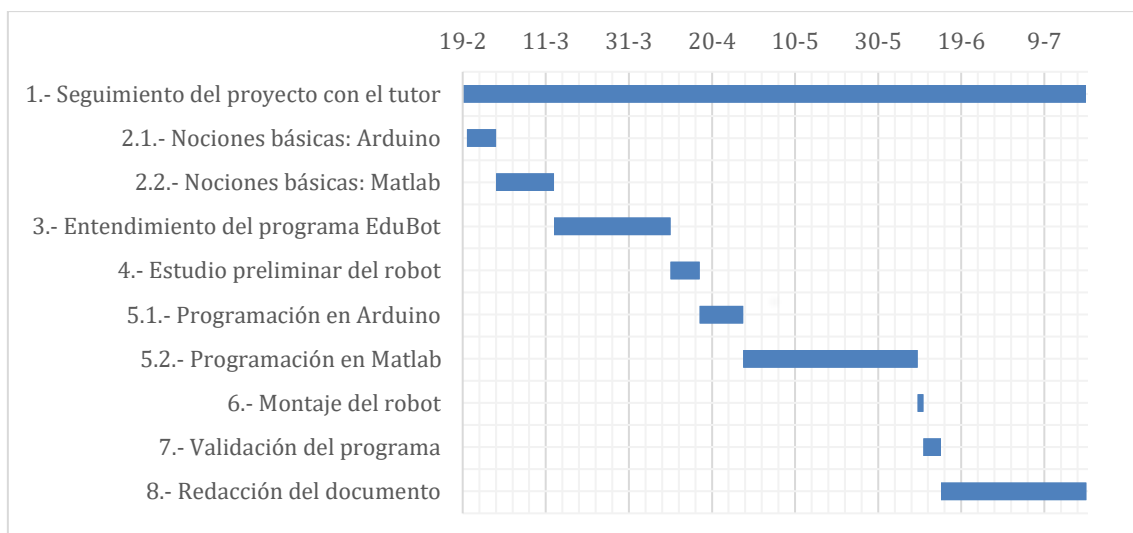


Ilustración 30. Diagrama de Gantt

11. Presupuesto

A continuación se va a realizar el presupuesto del proyecto. Al tratarse de un proyecto de control y programación, no se utilizará maquinaria o material a parte del robot Braccio. Por lo tanto, el gasto económico estará compuesto básicamente por el uso del ordenador, la licencia del programa Matlab y sobre todo horas de ingeniería.

A continuación se especifican los diferentes conceptos en los que se ha gastado, así como su importe.

11.1. Horas de trabajo

El trabajo estará realizado por un ingeniero, por lo que se le asignará un coste de 30€/hora. Teniendo en cuenta que el total de horas realizadas por el ingeniero se han estimado en 215, el presupuesto será el mostrado en la siguiente tabla:

| Horas internas | Horas totales | Coste horario | Coste total |
|-----------------------|----------------------|----------------------|--------------------|
| Ingeniero | 215h | 30€ | 6450€ |
| TOTAL | | | 6450€ |

Tabla 6. Presupuesto de las horas de trabajo

11.2. Material necesario

Para la realización del trabajo se tendrá que adquirir el brazo robótico Braccio así como una placa base Arduino uno. El coste será el siguiente:

| Material | Coste unitario | Unidades | Coste total |
|-----------------|-----------------------|-----------------|--------------------|
| Braccio | 200€ | 1 | 200€ |
| Arduino Uno | 20€ | 1 | 20€ |
| TOTAL | | | 220€ |

Tabla 7. Presupuesto del material necesario

11.3. Amortizaciones

Al realizar el proyecto se han utilizado dos ordenadores, uno propio del alumno y otro de la universidad. Además de esto se ha utilizado el programa Matlab, y su licencia también tiene un coste, por lo que habrá que incluirlo en las amortizaciones. Hay que tener en cuenta que durante mayoría del proyecto se han utilizado los ordenadores

(descontando el tiempo de estudio con libros), pero no durante todo este tiempo se ha utilizado el programa Matlab.

En la siguiente tabla se añade el coste derivado de las amortizaciones:

| Amortizaciones | Coste unitario | Uso | Vida útil | Coste total |
|-------------------------|-----------------------|------------|------------------|--------------------|
| Ordenador propio | 800€ | 150h | 3000h | 40€ |
| Ordenador universitario | 1000€ | 50h | 5000h | 10€ |
| Licencia Matlab | 3000€ | 130h | 4000h | 97.5€ |
| TOTAL | | | | 147.5€ |

Tabla 8. Presupuesto de las amortizaciones

11.4. Otros gastos

En este apartado se tiene en cuenta todo el material utilizado durante la realización de este proyecto, como por ejemplo las fotocopias, bolígrafos etc.

En la siguiente tabla se puede ver el coste total del material de oficina estimado.

| Gastos | Coste total |
|---------------------|--------------------|
| Material de oficina | 30€ |
| TOTAL | 30€ |

Tabla 9. Presupuesto de otros gastos

11.5. Costes indirectos

En los costes indirectos se tiene en cuenta los gastos derivados de la realización del proyecto, como son la luz, electricidad, calefacción, limpieza etc. Como la mayoría de este proyecto se ha realizado desde casa y no en la sala de ordenadores de la universidad, para calcular estos gastos se supondrá un 5% del total de los demás gastos.

En la siguiente tabla se puede visualizar el coste total indirecto estimado.

| Costes indirectos | Suma de gastos | Porcentaje costes indirectos | Coste total |
|--------------------------|-----------------------|-------------------------------------|--------------------|
| Costes | 6847.5€ | 5% | 342.4€ |
| TOTAL | | | 342.4€ |

Tabla 10. Presupuesto de los costes indirectos

11.6. Presupuesto total

A continuación se adjunta una tabla con todos los conceptos en los que se ha gastado, así como su coste y el porcentaje sobre el total.

| Concepto | Coste total | Porcentaje |
|--------------------|--------------------|-------------------|
| Horas de trabajo | 6450€ | 89.71% |
| Material necesario | 220€ | 3.06% |
| Amortizaciones | 147.5€ | 2.05% |
| Otros gastos | 30€ | 0.42% |
| Costes indirectos | 342.4€ | 4.76% |
| TOTAL | 7189.9€ | 100% |

El presupuesto total del proyecto asciende a 7189.9€.

12. Conclusión

La realización de este proyecto le ha permitido al alumno familiarizarse tanto con el control de brazos robóticos como el aprender a programar en el entorno que ofrece Matlab invirtiendo poco en el material que se debe utilizar.

Al ser el brazo robótico Braccio un robot pequeño y didáctico se ha utilizado una placa base Arduino para controlarlo, y por lo tanto se ha aprendido también sobre los diferentes tipos de placas existentes, así como el funcionamiento de la placa Arduino Uno y la programación necesarias en el entorno Arduino para lograr el correcto funcionamiento del brazo robótico Braccio.

Para la realización del control en el entorno Matlab se ha dispuesto del programa EduBot, el cual se ha tenido que comprender, aprendiendo así a entender programas grandes y complejos de terceros, así como a utilizar la comunidad de la que disponen Matlab y Arduino para la realización de programas.

Para el control del brazo robótico se ha tenido que ahondar en los conceptos de cinemática directa e inversa, para que así el programa informático pueda resolver estos problemas sin que el usuario tenga noción de cómo se resuelven. Estos conceptos son esenciales para cualquier tipo de brazo robótico (sobretudo el de cinemática inversa), y el ahondar en estos conceptos es muy importante si en el futuro se quiere trabajar en la industria en la parte de la automatización, pues muchos de los procesos industriales que se automatizan en la actualidad se hacen mediante brazos robóticos.

Una última parte ha sido la de aprender a redactar documentos amplios comentando qué se ha hecho durante el proyecto de manera que alguien que no esté familiarizado con el tema pueda entenderlo.

En general se concluye que este proyecto ha sido muy completo, ya que además de aprender a programar y resolver los problemas en un entorno virtual, se ha podido conectar este entorno virtual con la realidad mediante una conexión entre Matlab y el robot Braccio. Esto ha servido para ver que más allá de código e interfaces en Matlab, este código puede ser transformado en el movimiento real de un robot, pudiendo hacer que un usuario no experimentado controle el brazo robótico Braccio.

13. Bibliografía

1. Paolo Aliverti (2016). Manual de Arduino.
2. Gonzalo Fernández de Córdoba Martos. Creación de Interfaces Gráficas de Usuario (GUI) con Matlab. Septiembre de 2007.
3. Itziar Cabanes Axpe. Una herramienta didáctica para robots industriales: EDUBOT. Departamento de ingeniería de Sistemas y Automática, Escuela de Ingeniería de Bilbao.
4. Diego Orlando Barragán Guerrero. Manual de Interfaz Gráfica de Usuario en Matlab. Mayo de 2008.
5. Ana Rosa Carrera Amuriza y Margarita Martínez Nebreda. Introducción al Matlab. Departamento de Matemática Aplicada, Escuela de Ingeniería de Bilbao.
6. Hernández, A. (2011). Cinemática de mecanismos. Madrid: Síntesis.
7. Es.mathworks.com. (2018). MATLAB Central - MathWorks España. Accesible en: <https://es.mathworks.com/matlabcentral/>
8. Forum.arduino.cc. (2018). Arduino Forum – Index. Accesible en: <https://forum.arduino.cc/>.
9. Es.wikipedia.org. (2018). Brazo robótico. Accesible en: https://es.wikipedia.org/wiki/Brazo_rob%C3%B3tico.
10. Platea.pntic.mec.es. (2018). 5.4 Robots industriales. Accesible en: http://platea.pntic.mec.es/vgonzale/cyr_0708/archivos/_15/Tema_5.4.htm
11. Raspberry Pi. (2018). Download Raspbian for Raspberry Pi. Accesible en: <https://www.raspberrypi.org/downloads/raspbian/>.

ANEXO I: Funciones del programa

En este anexo se mencionarán y explicarán las funciones que se utilizan para el control del brazo robótico Braccio en el entorno de Matlab. Las funciones utilizadas se pueden ver en la ilustración 8, y algunas de ellas ya han sido mencionadas o explicadas en el apartado de metodología. Para un mejor entendimiento se ilustra de nuevo las funciones utilizadas y a continuación se procederá a explicar su funcionamiento.

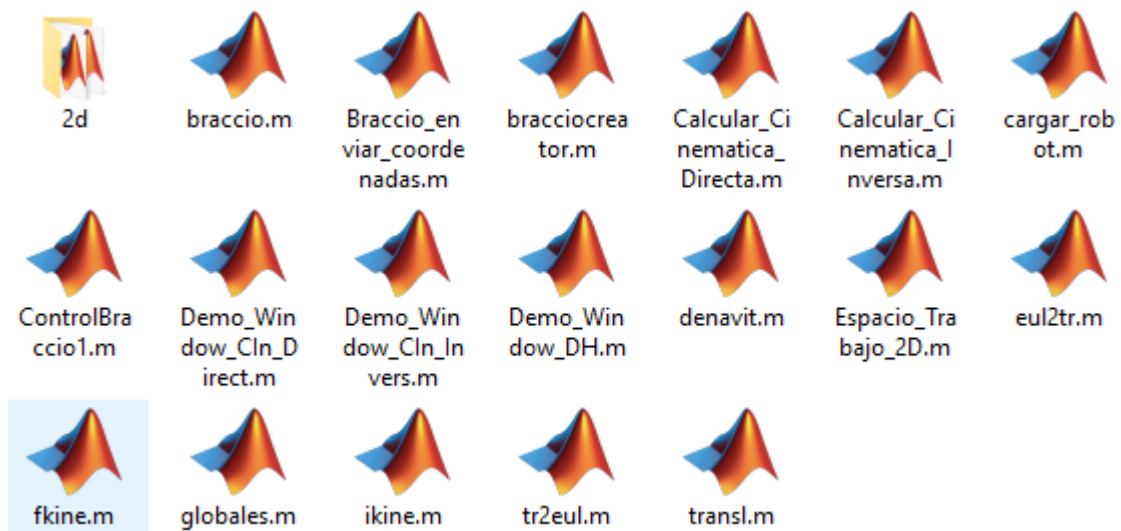


Ilustración 31. Programas utilizados

- Bracciocreator:

Este programa es el responsable de crear el archivo "robot_braccio.mat", que tendrá todos los datos del robot Braccio. Para ello utiliza la función "Braccio" y al objeto robot creado por esta le añade más datos como el autor, el nombre, el rango de las articulaciones, la imagen del robot Braccio etc. Una vez creado el archivo con todos los datos, el programa lo guarda y muestra al usuario el texto "BRACCIO CREADO".

- Braccio:

Este programa es utilizado por el programa "Bracciocreator" para crear un objeto de tipo "robot" con los datos físicos del robot. Estos datos son la tabla de Denavit-Hartenberg,

la masa de cada elemento y su centro de masas entre otros. La masa y el centro de masas no son necesarios para la cinemática, pero sí lo son para la dinámica, y por lo tanto se añaden por si en un futuro se quisiera resolver este problema.

- Braccio_enviar_coordenadas

Este programa se encarga de enviar las coordenadas articulares al robot. Para ello recibe 6 números que serán las coordenadas articulares que mandará a la placa Arduino vía USB. La placa Arduino será finalmente la encargada de mover el brazo robótico Braccio, y mandará de vuelta por el cable USB las coordenadas articulares finales (estas cambiarán respecto a las que se han enviado si están fuera del rango de las articulaciones). Finalmente el programa leerá estas coordenadas y se las mostrará al usuario.

- ControlBraccio1

Este programa es el encargado de hacer aparecer la primera Interfaz Gráfica de Usuario (GUI), así como de enlazar los botones en esta con funciones. Al ejecutar este programa, se busca el archivo con los datos del robot Braccio y se carga mediante el programa "Cargar_robot". Además de esto este programa abre el puerto USB por el que se comunicará con la placa Arduino.

- Demo_Window_CIn_Direct

Este programa se ejecuta al clicar sobre el botón "CIN DIRECTA" en la interfaz creada por el programa "ControlBraccio1". El programa creará la interfaz gráfica encargada de resolver la cinemática directa gracias a las diversas funciones que se encuentran enlazadas a los botones que hay en esta interfaz.

- Demo_Window_CIn_Invers

Este programa se ejecuta al clicar sobre el botón "CIN INVERSA" en la interfaz creada por el programa "ControlBraccio1". El programa creará la interfaz gráfica encargada de resolver la cinemática inversa gracias a las diversas funciones que se encuentran enlazadas a los botones que hay en esta interfaz.

- Espacio_trabajo_2D

Este programa se ejecuta al clicar sobre el botón "ESPACIO TRABJ." en la interfaz creada por el programa "ControlBraccio1". El programa creará una interfaz gráfica que permitirá al usuario definir el número de elementos (entre 1 y 4) y su longitud de un robot, para así poder calcular su espacio de trabajo en 2D.

- Demo_Window_DH

Este programa se ejecuta al clicar sobre el botón “Matriz DH” en la interfaz creada por el programa “ControlBraccio1”. Este programa crea una interfaz gráfica sencilla en la que se muestra la matriz de Denavit-Hartenberg del robot Braccio.

- Calcular_Cinematica_Directa

Este programa se utiliza dentro del programa “Demo_Window_CIn_Direct”, y es utilizado para calcular la cinemática directa del robot. Para ello se le envían la estructura del robot, las coordenadas articulares en forma de vector y su localización, y el programa devuelve la matriz de transformación del elemento fijo respecto al último elemento, que en el caso del robot Braccio será la pinza.

- Calcular_Cinematica_Inversa

Gracias a este programa es posible calcular la cinemática inversa del robot, por lo cual se encuentra dentro del programa “Demo_Window_CIn_Invers”. En este programa los datos de entrada son la estructura del robot, un vector con las coordenadas cartesianas del elemento final, otro vector con las coordenadas articulares iniciales y el tipo de conversión. Mediante los subprogramas “ikine”, “Fkine”, “transl” y “eul2tr” que se encuentran explicados más adelante, este programa devolverá un vector con las coordenadas articulares y la matriz de transformación obtenida a partir de las coordenadas articulares iniciales. Además de esto, el programa también devuelve un mensaje de error en caso de que la cinemática inversa no haya podido ser calculada.

- Cargar_robot

Este programa, tal como indica su nombre, tiene como finalidad cargar los datos del robot, y para ello tiene como dato de entrada el nombre del archivo que contiene los datos del robot Braccio.

Cuando este programa se ejecuta por primera vez abre el fichero que contiene los datos del robot Braccio, los guarda en la variable global “MiRobot”, y lee cada apartado para asegurarse de que este relleno, de no ser así saltará un mensaje diciendo que ha habido un error en la lectura y se fijará ese valor en 0 o se dejará en vacío.

Al inicializarse este programa se lee la variable “MiRobot”, por lo que si este programa ya ha sido ejecutado y la variable está llena, este programa no hará nada más que asegurarse de ello.

- Denavit

Este programa es utilizado por el programa "Calcular_Cinematica_Directa", su función es la de recoger los parámetros de Denavit-Hartenberg y devolver la matriz de transformación de acuerdo con estos parámetros.

- Eul2tr

Este programa, utilizado por "Calcular_Cinematica_Inversa", tiene como datos de entrada los ángulos de Euler y da como resultado la matriz de transformación homogénea derivada de estos ángulos.

- Rotx, roty y rotz

Estos programas son utilizados por el programa padre "euls2tr", y su función es la de recoger el ángulo de Euler y devolver la matriz homogénea que resulta cuando theta gira estos grados respecto al eje x, y o z.

- Tr2eul

Este programa tiene exactamente la función inversa del programa "eu2tr", es decir, tiene como dato de entrada una matriz de transformación y da como dato de salida los ángulos de Euler derivados de esta matriz de transformación. Este programa es utilizado por "Calcular_Cinematica_Directa".

- Fkine

Este programa es utilizado para resolver el problema de cinemática directa, y es utilizada por los programas "Calcular_Cinematica_Inversa" e "ikine". Para realizar su función debe tener como datos de entrada la estructura del robot y las coordenadas articulares de los servos, de esta manera dará como resultado las coordenadas cartesianas del elemento final.

- lkine

"lkine" es el programa que se encarga de interpolar para obtener las coordenadas articulares partiendo de las coordenadas cartesianas del elemento final, y por ello se utiliza en el programa "Calcular_Cinematica_Inversa".

Como datos de entrada debe tener la estructura del robot y las coordenadas cartesianas del elemento final, y hay otros dos datos que son opcionales: la posición desde la que

se empezará a interpolar (0 por defecto) y la máscara (necesaria para robots con menos de 6gdl).

Finalmente, mediante un proceso iterativo de interpolación (Newton-Raphson), este programa logra las coordenadas articulares necesarias para que el elemento final se sitúe en las coordenadas cartesianas estipuladas. Cabe destacar que al ser un proceso iterativo, puede que no se encuentre la solución con el margen de error necesario en las iteraciones permitidas, en este caso o en el caso de que no sea posible llegar a las coordenadas cartesianas, saltará un mensaje de error.

- Tr2diff, jacob0, jacobn, tr2rot,

Estos subprogramas son utilizados por "ikine" para poder llevar a cabo su función. No se comentará más sobre ellos debido a que son utilizados mayormente para no sobrecargar el programa principal, con muchas líneas de código.

- Globales

Este subprograma simplemente declara la variable "MiRobot" como global y es utilizado en cada función que tenga que utilizar la estructura del robot. Aunque realmente se podría prescindir de esta función ya que no implica un cambio significativo ni en la legibilidad ni en la cantidad de código, se ha dejado ya que era utilizado anteriormente en EduBot.

- Transl

Este programa tiene como datos de entrada las distancias en las coordenadas X, Y, Z y con ello crea la matriz de translación homogénea.

- Carpeta "2d"

Esta carpeta contiene los siguientes programas: "ejecuta2D", "plotea2D", "primero2D", "rota2D", "seleccion2D". Estos programas son los encargados de realizar una imagen en 2D del espacio de trabajo del robot que introduzca el usuario. Por la complejidad matemática que tienen estos programas y ya que el objetivo principal de este trabajo es el control del brazo robótico y no tanto el saber cuál es su espacio de trabajo, no se comentarán más en detalle estos programas.

ANEXO II: Manual de instrucciones

En este anexo se va a tratar de guiar al usuario a través de Matlab y las interfaces gráficas para poder sí controlar el robot Braccio de manera sencilla.

Inicio

Al iniciar Matlab, el usuario tendrá que ejecutar el programa “ControlBraccio1” para así poder acceder a la interfaz principal. Para hacer esto deberá ir a la carpeta en la que se encuentre este programa y ejecutarlo desde la ventana de comandos o bien clicar sobre el botón “run”, como podemos ver en la siguiente imagen.

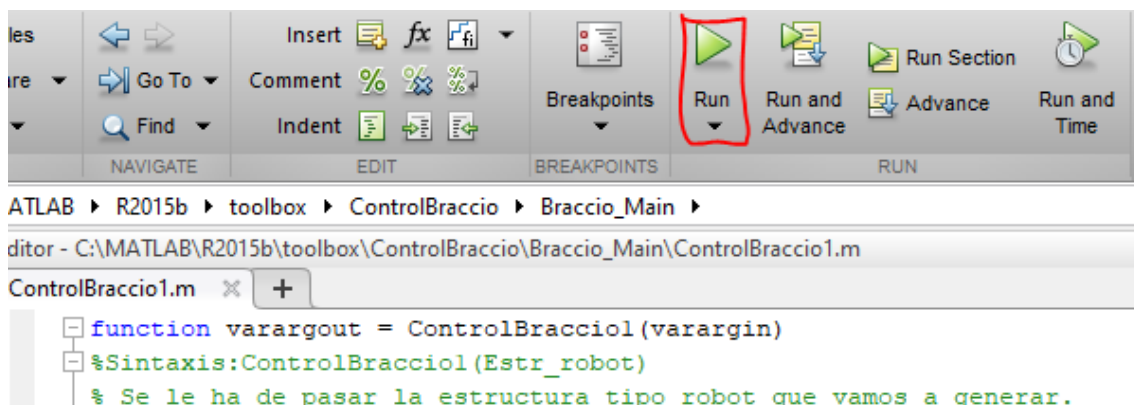


Ilustración 32. Ventana para iniciar el programa

Una vez se haya clicado sobre este botón, aparecerá en pantalla la interfaz gráfica de la ilustración 16. En esta interfaz gráfica el usuario deberá de elegir cual es la función que desea realizar: cinemática directa, cinemática inversa, visualizar el espacio de trabajo o ver la matriz de Denavit-Hartenberg. Para ello tiene diferentes botones en esta interfaz, cada uno de los cuales abre una interfaz para su propósito. Estas interfaces y sus utilizaciones se explicarán a continuación.

Cinemática directa

En este panel que podemos ver en la ilustración 17, el usuario deberá de especificar las coordenadas articulares que quiere que tome el robot Braccio, junto con el estado de la

pinza (abierta/cerrada). Para ello deberá de clicar sobre la articulación que quiera cambiar, escribir las nuevas coordenadas en radianes y darle al botón de modificar, mientras que para especificar el estado de la pinza simplemente se deberá de clicar sobre el estado que se quiera lograr. En la siguiente imagen se muestra esta parte de la interfaz.

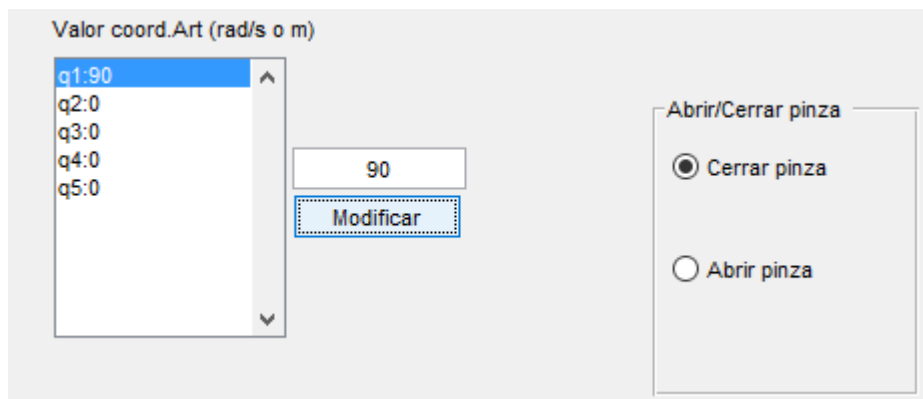


Ilustración 33. Panel de cinemática directa: modificar coordenadas

Cuando el usuario haya establecido las coordenadas articulares que debe tomar el robot, deberá clicar sobre el botón de calcular, con lo que se le abrirá un nuevo panel en el que podrá ver las la posición y orientación que tendría cada elemento si el robot tuviera esas coordenadas articulares. En la siguiente imagen se muestra una parte de la segunda interfaz.

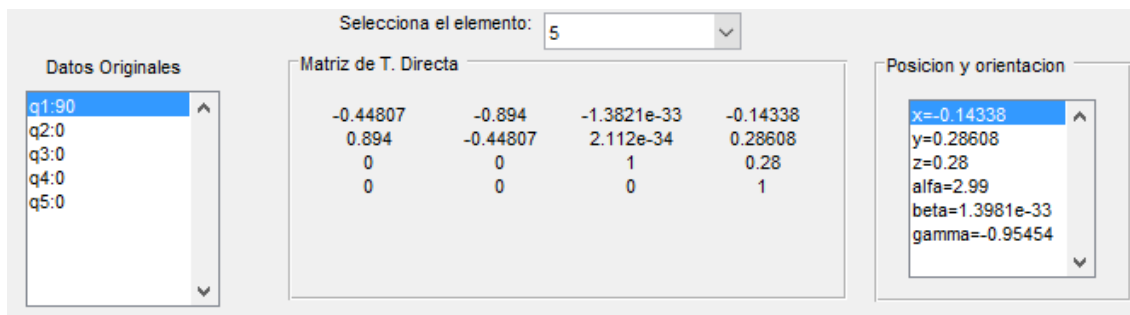


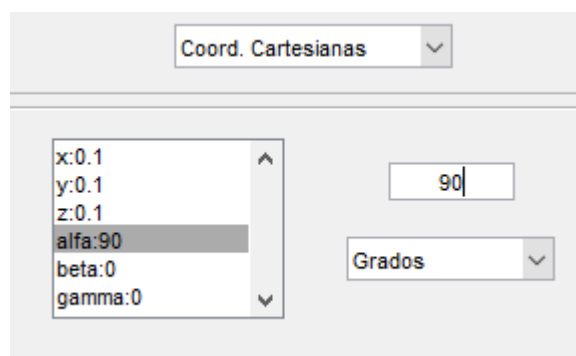
Ilustración 34. Panel de cinemática directa: ver coordenadas cartesianas

Como se puede observar en esta imagen, el usuario puede ver los datos originales y la matriz de transformación directa del elemento que aparece en el recuadro superior (el cual se puede cambiar) además de su posición y orientación.

Finalmente, el usuario podrá mover el robot a las coordenadas que ha establecido simplemente clicando sobre el botón de enviar órdenes.

Cinemática Inversa

El panel de cinemática inversa se puede ver en la ilustración 23. En este panel el usuario deberá de elegir en un primer paso las coordenadas que quiere introducir: articulares o cartesianas. Después deberá de introducir estas coordenadas en el segundo paso, eligiendo las unidades (grados o radianes) en el recuadro que se encuentra a la derecha. Para una mayor visualización se añade una imagen de esta parte del panel.



The image shows a software interface for inverse kinematics. At the top, there is a dropdown menu labeled "Coord. Cartesianas". Below this, on the left, is a list of input fields for joint coordinates: "x:0.1", "y:0.1", "z:0.1", "alfa:90", "beta:0", and "gamma:0". The "alfa:90" field is currently selected. To the right of the list, there is a text input field containing the number "90" and a dropdown menu labeled "Grados".

Ilustración 35. Panel de cinemática inversa: introducir datos

Una vez el usuario haya introducido los datos requeridos, deberá clicar sobre "Calcular", así se irá a una segunda interfaz en la que aparecerán los datos originales, las coordenadas articulares necesarias para conseguir estos datos originales y la opción de abrir/cerrar la pinza. Si el usuario está de acuerdo con las coordenadas articulares conseguidas, deberá darle al botón "Enviar órdenes" para poder mover el robot de acuerdo con estas. De otra manera, el usuario también puede clicar sobre "Otra vez" para volver a la primera pantalla y modificar los datos originales.

Espacio de trabajo

Este panel (ilustración 27) posibilita al usuario obtener el espacio de trabajo de cualquier robot de 1 a 4 grados de libertad en 2 dimensiones. El usuario deberá de establecer el

número de articulaciones que tiene el robot, su longitud y los límites que tienen y a continuación clicar sobre el botón (Dibujar Espacio de Trabajo), con lo que se abrirá una nueva ventana mostrando al usuario el espacio de trabajo de su robot. Hay que recalcar que cuantas más articulaciones tenga el robot, más tardará el programa en mostrar la imagen final.

Matriz de DH

Este panel (ilustración 29) tiene la simple función de mostrar al usuario la matriz de Denavit-Hartenberg, por lo que el usuario simplemente podrá echarle un vistazo y cerrar la ventana.