UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN

DESARROLLO DE DOS SIMULADORES VIRTUALES PARA APOYO A LA CLASE DE ROBÓTICA EN LÍNEA EN LOS TEMAS DE CINEMÁTICA DIRECTA E INVERSA DE ROBOTS INDUSTRIALES.

T E S I S

PARA OBTENER EL TÍTULO DE:

INGENIERO MECÁNICO

PRESENTA

ALEXIS OMAR CEJUDO TIRADO

ASESOR:

DR. PATRICIO MARTÍNEZ ZAMUDIO

Ciudad Nezahualcóyotl, Estado de México, 2021

RESUMEN

En este trabajo de tesis se presenta el desarrollo de dos simuladores virtuales que tienen como propósito la innovación y el mejoramiento del proceso enseñanza-aprendizaje que beneficien a los alumnos a comprender los temas de cinemática directa e inversa de una manera interactiva, creativa, con nuevas formas de pensar, para motivar el interés y la imaginación de los estudiantes, innovando y mejorando el proceso enseñanza-aprendizaje. Se desarrolla el material didáctico con simuladores, así como la puesta en marcha en las clases en línea y de forma presencial, se realizan pruebas en clase, donde se observan los resultados con los alumnos de la asignatura de robótica del Dr. Patricio Martínez del semestre 2021-1.

Este trabajo se encuentra dividido por cinco capítulos, y al final la conclusión a la que se llegó con el desarrollo del presente trabajo.

El Capítulo 1 se presenta la información sobre la robótica y como está teniendo un importante impacto a nivel mundial para la industria 4.0, además se muestra el uso de los simuladores en área de la educación para fortalecer el conocimiento de los alumnos en las asignaturas.

El Capítulo 2 se muestra el uso de la herramienta de Matlab y puede ser utilizada para desarrollar el análisis cinemático de los robots seriales industriales, así como también, se muestra el desarrollo de la cinemática directa e inversa del robot IRB-140 de la marca ABB.

En el Capítulo 3 se presenta el desarrollo que se llevó a cabo para realizar la implementación de los dos simuladores, se muestra la parte visual y el código de programación de ambas aplicaciones.

En el Capítulo 4 se realiza la comparación de los resultados obtenidos con la aplicación contra los de la teoría para observar que ambas aplicaciones funcionen correctamente.

Y por último en el capítulo 5 se muestran los resultados de aplicar la herramienta desarrollada en los alumnos de Robótica utilizando los dos simuladores desarrollados para las clases en línea, impartidas en la Facultad de Ingeniería y la Facultad de Estudios Superiores Aragón.

Agradecimientos

"Todo lo que más te cuesta, es lo más disfrutas" estas son unas de las demasiadas enseñanzas que mis padres me dieron. Nada de esto sería posible sin su apoyo, sin su esfuerzo. Por demostrarme que los hechos valen más que mil palabras. Gracias por todo.

A mis padres Diana Tirado y al Ing. Antonio Vijosa

A Dios por brindarme de su amor y su bondad para permitirme realizar este trabajo

Por siempre acompañarme, además de sacarme una sonrisa cada que lo necesitaba.

A mis hermanas Evelyn y Jaquelin Vijosa

A todos los integrantes de una gran Familia que me abrieron un pequeño lugar en sus corazones, dándome la oportunidad de ser parte de ellos.

A todos los Vijosa

Por demostrarme que, aunque la familia tenga sus adversidades siempre tiene que estar unida, además de brindarme de su apoyo, palabras de aliento y risas.

A todos los Tirado

A una hermosa persona que me dio todo su cariño, amor y grandes consejos de vida.

A mi abuela Rosa Carrillo.

A dos grandes personas que marcaron mi vida para siempre y ahora están apoyándome desde el cielo. Un gran ingeniero que me enseño, que no importa tener poco, lograrás tus éxitos siempre y cuando quieras salir adelante. Y una hermosa mujer que fue como una segunda madre para mí.

A mi abuela María de las Nieves

Y

Al Ing. Antonio Vijosa

Gracias por todo.

Por brindarme de su apoyo y de todo su conocimiento en el área de la ingeniería para poder realizar este trabajo.

A mis tíos la Ing. Imelda Vijosa

Y

Al Ing. Ubaldo Flora

A dos grandes personas por siempre darme un lugar en su casa y por sus sabios consejos.

A mis tíos, Mireya Tirado

Y

A Ismael Duran

Por brindarme su apoyo, tiempo y sobre todo confianza para realizar este trabajo.

A mi asesor el Dr. Patricio Martínez, Zamudio

A todas aquellas personas que no mencione pero que siempre creyeron en mí, que me dieron una palabra de aliento para seguir adelante y a no rendirme. Quiero decirles gracias.

A los alumnos del grupo 1803 de la carrera ingeniería mecánica de la FES Aragón y los alumnos de ingeniería mecatrónica de Ciudad universitaria, gracias por prestarme su valioso tiempo para hacer las pruebas necesarias para esta tesis.

Agradezco en lo que corresponde a la DGAPA, por el apoyo brindado para la realización de este trabajo, a través del proyecto UNAM-DGAPA-PAPIME-PE110120: "Desarrollo de material didáctico y banco de pruebas para el fortalecimiento de la enseñanza en Instrumentación y Robótica introduciendo CPS."

INTRODUCCIÓN

La llegada del virus SARS-CoV-2 a México, afectó drásticamente en varios sectores del país, como fue el turismo, la economía y la educación.

La educación en la UNAM se vio afectada totalmente, pues más de 350,000 alumnos inscritos se quedaron sin clases presenciales para el ciclo escolar 2021-1. Lo cual se optó por impartir el semestre en línea para no dejar a los alumnos sin clases. [64]

El problema radica en que algunas materias requieren de hacer experimentos o prácticas en laboratorios para comprender mejor los conceptos y que los alumnos desarrollen habilidades que fortalecen el aprendizaje. Una de esas asignaturas es la Robótica que se imparte en la Facultad de Estudios Superiores Aragón y en la Facultad de Ingeniería en Ciudad Universitaria. Esta asignatura es obligatoria para los alumnos de la FES Aragón del módulo de biomecánica de séptimo semestre y optativa para otros módulos, y para los alumnos de la Facultad de Ingeniería que la cursan a partir del noveno semestre.[26] [28] Es una materia de difícil comprensión y más si no se cuenta con el material o herramientas necesarias.

Qué los alumnos comprendan las bases de la robótica es importante, debido al avance tecnológico. Cada vez millones de personas pierden sus empleos y son reemplazadas por robots que pueden realizar tareas eficientes, rápidas y repetitivas. Con la llegada de la industria 4.0 alrededor de 9 millones de empleos serán sustituidos por la automatización, pero también generarán otros empleos.[56]

Es importante que las futuras generaciones tengan las bases necesarias para entender a la industria 4.0; La robótica tendrá un impacto importante en esta área por lo que se debe de enseñar con bases sólidas y bien fundamentadas, ya sea de forma presencial o en línea.

Osorio Villa, et. Presentan el trabajo titulado “El uso de simuladores educativos para el desarrollo de competencias en la formación universitaria de pregrado”, en el artículo se menciona que: “*El uso de simuladores educativos proporciona*

herramientas de apoyo a la formación y aprendizaje del alumno ya sea en programas presenciales, a distancia y/o virtuales" [47]. En el artículo se concluye que el uso de los simuladores ayuda a la formación escolar de los alumnos.

JUSTIFICACIÓN

Los simuladores virtuales permiten acercarse a la realidad ya sea en entornos difíciles de acceder o en simular situaciones de la vida real. El usar simuladores virtuales para dar a entender mejor un tema ha sido demostrado en varios estudios dando como resultado una mejor compresión del tema y mayor interés a las clases. Varias personas que toman clases en línea se frustran porque no se cuenta con las herramientas necesarias, así como con la preparación suficiente.[20] [47]

Con el desarrollo de material didáctico se espera que los alumnos cuenten con las bases teóricas de los temas de robótica, debido a que es uno de los pilares que sobre sale en la industria 4.0. Dentro de las ventajas que ofrecen los simuladores es el costo de adquisición y mantenimiento dado que conseguir un Robot industrial para la enseñanza suele ser costoso, por lo que se puede ser sustituido utilizando simuladores. así como se pueden observar varios aspectos de la robótica como el movimiento, la posición y orientación del efector final. [11] [63]

Utilizar simuladores con el propósito de la innovación y el mejoramiento del proceso enseñanza-aprendizaje, refuerza el conocimiento a la materia de robótica, que será de gran ayuda para los estudiantes de la FES Aragón y la Facultad de Ingeniería que la cursan en línea o de forma presencial.

HIPÓTESIS

La hipótesis en este trabajo de tesis se centra en validar sí:

¿El uso de simuladores virtuales innovará, mejorará el proceso enseñanza-aprendizaje, así como fortalecerá el conocimiento de la asignatura de robótica en línea, en los temas de cinemática directa e inversa, en los alumnos de la FES Aragón y la Facultad de Ingeniería?.

OBJETIVO

Desarrollar material didáctico para el fortalecimiento de la enseñanza en Robótica aplicando tecnologías emergentes con potencial innovador que permitan una enseñanza creativa.

OBJETIVOS PARTICULARES

- Desarrollar los modelos matemáticos para validar la cinemática directa e inversa de un robot IRB-140.
- Desarrollar material didáctico basado en software para la simulación de la cinemática directa e inversa.
- Desarrollar dos aplicaciones virtuales con la herramienta GUI y las librerías Robotic Toolbox de Peter Corke y ARTE Robotic de MATLAB.
- Presentar a los alumnos la aplicación con el fin de obtener una retroalimentación de las aplicaciones virtuales de la cinemática directa e inversa en Matlab.

OBJETIVOS ESPECÍFICOS

- Utilizar los programas desarrollados en este trabajo de tesis para explicar la cinemática directa e inversa a los alumnos que toman la asignatura de robótica en el semestre 2021-1.
- Realizar una encuesta para ver si los alumnos aprendieron de una mejor manera o no.

Índice

1. CAPÍTULO 1: LA ROBÓTICA EN EL MUNDO Y EL USO DE SIMULADORES VIRTUALES PARA LA ENSEÑANZA

1.1. INDUSTRIA 4.0 Y LA ROBÓTICA

La industria 4.0 se centra en la interconectividad, la automatización, el aprendizaje automático y los datos en tiempo real. Busca cambiar la forma de producir y de vender un producto, utilizando y combinando los sistemas autónomos (robots, controladores, etc.) con el mundo de los datos (nube, inteligencia artificial, etc.) para desarrollar la llamada “fabrica inteligente”. [15]

Los pilares tecnológicos de la industria 4.0 que se destacan son:

1. Sistemas de integración.
2. Máquinas y sistemas autónomos (robots).
3. Internet de las cosas (IoT).
4. Manufactura aditiva.
5. Big data y análisis de grandes datos.
6. Computación en la nube.
7. Simulación de entornos virtuales.
8. Inteligencia Artificial.
9. Ciberseguridad.
10. Realidad aumentada.

Todas juegan un papel importante en los avances tecnológicos y por lo menos una ya la conoces, pero la industria 4.0 busca juntar estas tecnologías para revolucionar la industria que conocemos actualmente.[11]

En la actualidad hay más de un millón de robots operando en el mundo, ha sido tanto el avance tecnológico que estos sistemas no solamente se utilizan en las industrias, si no también, se ocupan en los hospitales, en el hogar, en la desactivación de bombas, operaciones de rescate y búsqueda, en el transporte de materiales, etc. [9]

Los manipuladores se han utilizado en la industria automotriz desde hace más de 50 años, la desventaja de estos robots es que necesitan de celdas de protección porque no pueden colaborar cerca de los humanos, no son capaces de presenciar a un ser humano, y trabajar cerca de uno es sumamente peligroso.[7] [32]

La industria 4.0 sustituirá al humano en tareas peligrosas y repetitivas, pero además en esta nueva era de la robotización, la unión entre estos sistemas autónomos con las tecnologías cómo el Big Data, inteligencia artificial, sensores y controladores inteligentes, permitió crear una nueva generación de robots que son capaces no sólo de desarrollar tareas repetitivas sino también de poder tomar decisiones propias y pueden trabajar cerca de un humano sin necesidad de usar celdas de seguridad. [11]

Carbajal Rojas (2017) dice "*La cuarta revolución industrial, así como ocurrió con la tercera, la segunda y la primera, impactarán a los sistemas productivos, a los modos de producción y a la educación superior en ingeniería."* [14] por lo que los alumnos deben de mejorar sus habilidades, perfeccionar el conocimiento en estas áreas, para estar preparados para el futuro.

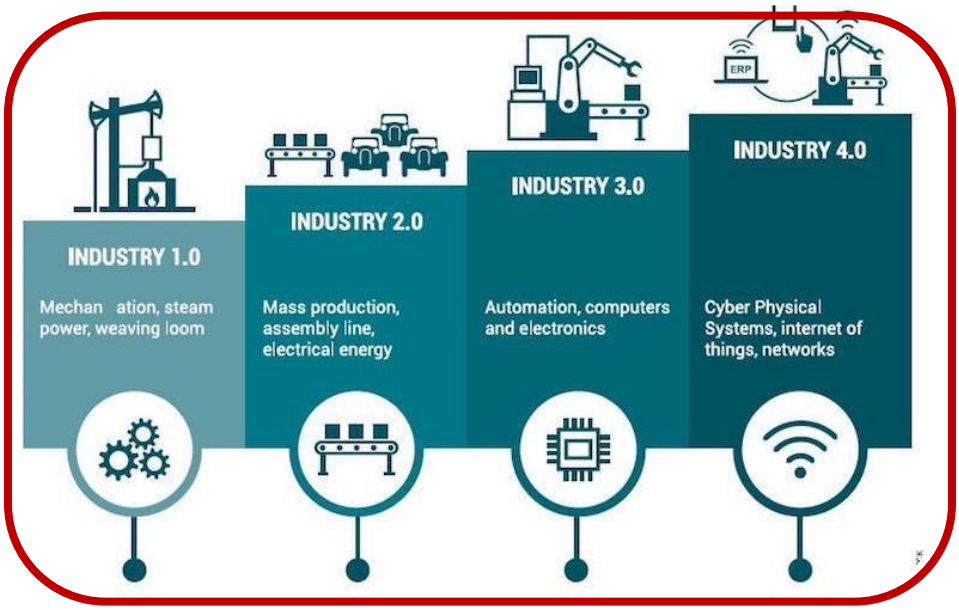


Ilustración 1. Evolución de las Industria

1.2. MÉXICO Y LA INDUSTRIA 4.0

Actualmente México aspira a desarrollar la cuarta revolución industrial, hay potencial en el país, pero la falta de un gobierno interesado para implementar esta tecnología, ha hecho que las industrias como la automotriz, la aeronáutica o la tecnológica, que han logrado implementarla sean de origen extranjero y no nacional. [49] Gracias estas compañías en el 2018, México adquirió alrededor de 5,500 robots industriales posicionándolo, en la novena posición sobrepasando a España con 5,300 manipuladores en la industria. Las grandes potencias en tecnología se encuentran en las primeras posiciones, China es uno de los países con más robots superando a Estados Unidos por más del triple. [52] [66]

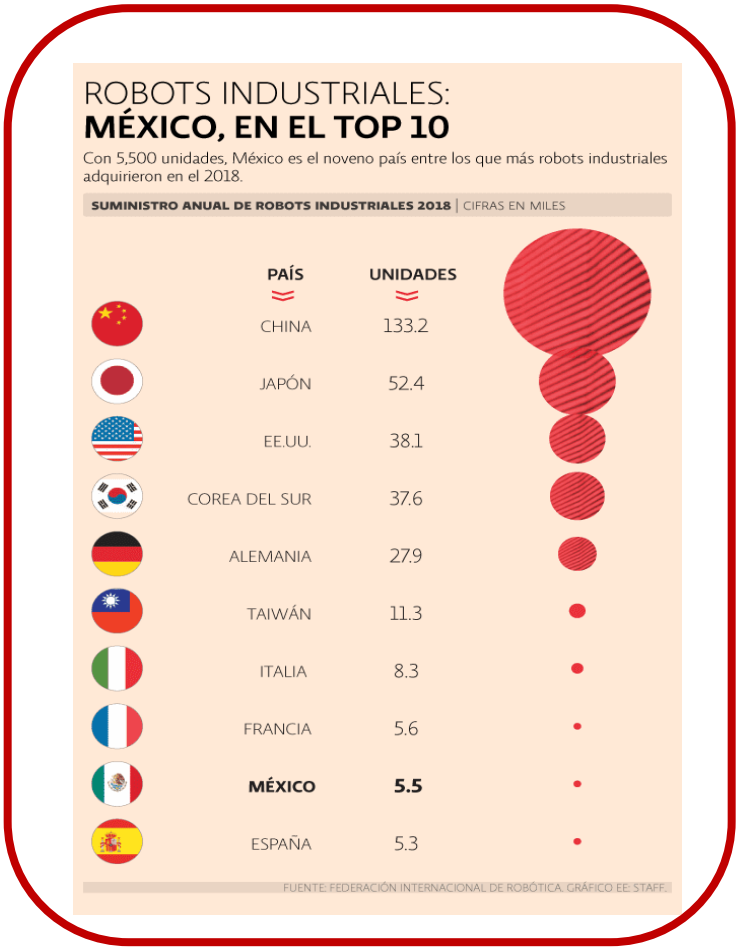


Ilustración 2. Federación Internacional de Robótica (2018), Países con más robots industriales en el mundo, Recuperado en https://www.eleconomista.com.mx/tecnologia/

México se encuentra en un dilema, pagar por la mano de obra como lo ha venido haciendo, o empezar a innovar. Si sigue por el camino de utilizar personas en vez de invertir en la automatización, perdería más probabilidades de mejorar su nivel competitivo internacional. Podrían tomar el ejemplo de los líderes digitales que capacitan a las personas que se puedan ver afectadas por la automatización para que no pierdan sus empleos. Todo lleva un proceso, se requiere de inversión y de mucha paciencia, pero es mejor estar preparados para esta nueva revolución industrial que vino para quedarse.[49]

1.3. EFECTOS DE LA INDUSTRIA 4.0

Con la llegada de la cuarta revolución industrial, muchas personas alrededor del mundo se verán afectadas, se estima que para el 2025 la cifra de empleos que serán reemplazados por robots será alrededor de 140 millones. El proceso robótico de automatización (RPA por sus siglas en inglés) busca simular las acciones de una persona en un proceso industrial.[65] Además de los desempleos, otros sectores están siendo afectados, tal es el caso de la educación y la necesidad de que los alumnos aprendan a manejar este tipo de tecnología. Algunos inconvenientes que destacan en la industria 4.0 son:

- Muchas industrias pueden quedar desactualizadas porque no se están adaptando a la nueva tecnología.
- El personal necesario para esta tecnología ahora tendrá que estar mejor preparado, y su remuneración será mayor.

Es necesario que los alumnos estén preparados para esta clase de tecnología pues será un cambio radical en la industria. Las empresas empezarán a implementar estas tecnologías y reemplazarán a las personas en procesos repetitivos y pesados. Por lo tanto, es necesario aprender más sobre este tema.[5] [61]

1.4. LOS COBOTS

Con la llegada de la tercera revolución industrial en 1969, se implementaron los modelos de automatización, las computadoras y la electrónica para los procesos industriales, se sustituyeron humanos por robots en tareas pesas y repetitivas. Con el paso del tiempo estos robots fueron perfeccionándose, llegando a ser mucho más rápidos, precisos y hasta realizar tareas las 24 horas del día los 7 días a la semana. Pero estos manipuladores son extremadamente peligrosos para que trabajen cerca de humanos, se necesita de celdas de protección para delimitar el área de trabajo del robot y así evitar que las personas se acerquen tanto y puedan sufrir un accidente, pues estos robots no tienen la capacidad de detectar a un individuo cerca y, además, trabajan a velocidades altas.[11]

Con la llegada de la industria 4.0 los robots ahora tienen la capacidad de trabajar a altas velocidades y al mismo tiempo pueden trabajar cerca de una persona o colaborando sin necesidad de tener celdas de seguridad. A esta clase de robots se les conoce como, robots colaborativos o Cobots.

Como ejemplo de robots colaborativos se expone la experiencia vivida el día jueves 23 de junio del 2020, donde el Ing. Jesús Delgado Mata, supervisor de ventas de la empresa GUREGO (Datos del ingeniero en el **Apéndice 1**), presento un Cobot de la marca DOOSAN en la empresa AMD Maquinaria en Querétaro (**Ilustración 3**). Este autómata tiene la capacidad de programar su propia zona de seguridad sin tener vallas alrededor, es sensible a la colisión, esto quiere decir que al sentir un pequeño contacto en sus motores ya sea con un objeto o persona el robot se detiene automáticamente. Dependiendo la herramienta final que se le coloque, puede ser utilizado para realizar distintas tareas como “pick and place”, inspección visual, montaje, armado, etc. Estos robots son sencillos de programar pues sólo haciendo un clic en el botón central que se ubican en el último eslabón (**Ilustración 4**) se crea un “target point” o “punto objetivo” con el fin de que siga una trayectoria para realizar la tarea deseada (Más datos sobre el robot ver **Apéndice 2**).

Ilustración 3. COBOT de la marca Doosan, presentado en AMD Maquinaria Querétaro

Ilustración 4. Eslabón final del robot y sus botones

1.5 ROBOTS DE SERVICIO

Esta clase de robots se utilizan alrededor del mundo para realizar otro tipo de tareas. Estos autómatas son más utilizados para el entretenimiento, protección, limpieza, transporte, inspecciones, vigilancia, mantenimiento, etc. Los robots de servicio surgieron por la necesidad de facilitar algunas tareas de la vida cotidiana, haciéndolas automáticas y rápidas.[23] Esta necesidad ha generado que aparezcan cada vez más robots de servicio, por ejemplo:

Bleeper Sport, es un robot diseñado para exploración acuática, consta de tres hélices propulsoras, luces halógenas, es tele-operado por un "joystick" y cuenta con una cámara que emite imagen a color. Llega a una profundidad máxima de 50 metros y pesa tan solo 10 Kg. De acuerdo con Jesse Ausubel científico de Rockefeller, menciona que esta clase de tecnologías van a ofrecer una gran oportunidad para poder explorar el océano, ya que hasta el momento solo se ha explorado entre el 5-10%.[24]

Ilustración 5. Bleeper Sport (2008), Robot para exploración marina, Recuperado en: http://www.fondear.org/

Hay otras funciones para los robots de servicio, el cual no tiene que ser del todo científica, si no también, para servicio a las personas. Tal es el caso del Hotel Henn Ha que se encuentra en Japón y es el primer hotel donde es atendido por más de 230 robots. [1]

Ilustración 6. *Hotel Henn-Na (2018), Robots para servicios de hotelería. Recuperado en: https://bunkerdb.com/*

Con los robots de servicio se busca reemplazar a las personas en tareas sumamente peligrosas, por ejemplo, en la detección y desactivación de bombas. Tal es el caso del robot CUTLASS, un robot que sirve para la desactivación de equipos explosivos, fabricado por el corporativo Northrop Grumman.[8]

Ilustración 7. *CUTLASS (2008), Robot para desactivación de bombas, Recuperado en: https://www.army-technology.com/*

Existe una gran variedad de robots de servicio para efectuar diferentes tareas. En el **Apéndice 3** se muestra un mapa mental sobre la clasificación de robots de servicio y de los robots industriales.

1.6 APLICACIONES DE LOS MANIPULADORES EN LA INDUSTRIA 4.0.

Los robots se empezaron a utilizar desde la tercera revolución industrial en 1969, ahora con la llegada de la actual “Industrial 4.0” y de los cobots, las empresas están apuntando a esta clase de herramientas porque se han convertido en un instrumento viable y rentable para reducir costos de producción, bajar el índice de riesgos, entre otros factores más. La gran ventaja de estos robots colaborativos es la seguridad de poder laborar junto con una persona, y mientras el robot se encarga de trabajos pesados, la persona se encarga de supervisar que todo este correcto. [46]

Los robots industriales cada vez son más utilizados en distintos sectores industriales, realizando tareas de almacenamiento, soldadura, atornillado, acomodado, armado, o para trabajos que son repetitivos.[60] La empresa Universal Robots, fabricantes de Robots Colaborativos (Cobots) muestra los departamentos donde se utilizan sus robots hoy en día

Alimentación y Agricultura

Ilustración 8. *Industria Cascina, Italia, Robots de UR utilizados para el empaquetado de huevos, Recuperado de https://www.universal-robots.com*

Mobiliario y equipamiento

Ilustración 9. *Industria Etalex, Canadá, Cobots de UR utilizados para producción, Recuperado de https://www.universal-robots.com*

Electrónica y tecnología

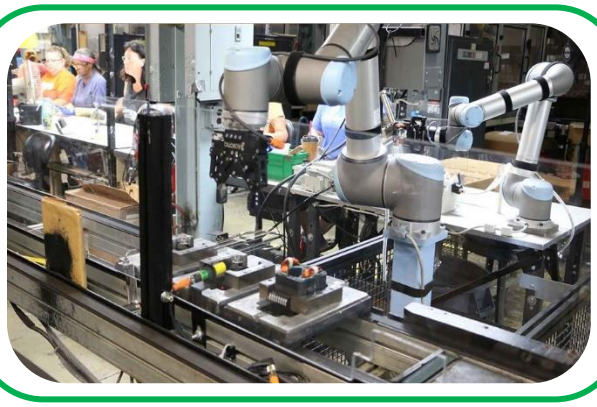


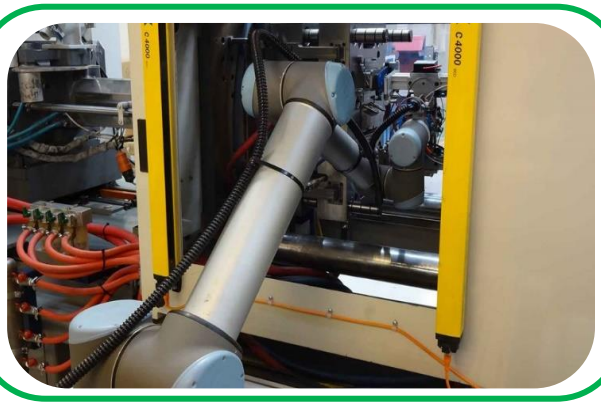
Ilustración 10. *Industria Scott Fetzer Electrical Group, Estados Unidos, Robots para la industria electrónica y tecnológica, Recuperado de https://www.universal-robots.com*

Metal y Mecanizado

Ilustración 11. *Industria Ferdinand Wagner, Alemania, Robots utilizados para mecanizado, Recuperado de https://www.universal-robots.com*

Automoción y subcontratistas

Ilustración 12. *Industria Automotriz, Robots utilizados para armado de autos, Recuperado de https://www.universal-robots.com*



Plásticos y polímeros

***Ilustración 13**. Industria Talbot Techologies, Nueva Zelanda, Robots utilizados para la producción de plásticos, Recuperado de https://www.universal-robots.com*

Farmacéutica y química

***Ilustración 14.** Industria Opticon, Dinamarca, Robots utilizados para manipulación de audífonos, Recuperado de https://www.universal-robots.com*

Científica e investigación

***Industria 15.** Industria Aurolab, Nueva Zelanda, Robots utilizados para la investigación, Recuperado de https://www.universal-robots.com*

1.7 MORFOLOGÍA DEL ROBOT INDUSTRIAL

El término morfología del robot especifica cómo está constituido un manipulador industrial, que elementos lo conforman y como es su estructura. Ya que hay varios manipuladores industriales que realizan diferentes tareas, todos cuentan con elementos semejantes, los cuales son:

- Estructura mecánica.
- Actuadores.
- Transmisores.
- Sensores.
- Controlador. [62]

1.7.1 ESTRUCTURA MECÁNICA

Mecánicamente, un robot está formado por eslabones unidos por articulaciones que permiten un movimiento relativo entre cada dos eslabones consecutivos. La composición física de la mayoría de los manipuladores industriales guarda cierta semejanza con la fisiología de un brazo humano, puesto que cuenta con un "cuerpo", "brazo", "codo" y una "muñeca", este tipo de anatomía le proporciona al manipulador un amplio campo de trabajo.[51] [62]

1.7.2 ACTUADORES

Los Actuadores son dispositivos mecánicos que se encargan de generar fuerzas suficientes para darle movimiento a un manipulador. Se clasifican como rotativos y lineales. Y pueden ser accionados por energía neumática, eléctrica e hidráulica.[37]

En base al reporte de prácticas en el año 2016 de la Universidad Don Bosco en Guatemala por el departamento de ingeniería para la asignatura fundamentos de Robótica y con los "datasheets" de los actuadores utilizados actualmente en la industria, se realizó la **Tabla 1.1** dónde se observa las desventajas y ventajas de los diferentes tipos de actuadores. [6] [29] [35] [62]

ACTUADORES			
TIPO DE ACTUADOR	**Neumático**	**Hidráulico**	**Eléctrico**
FUNCIONAN CON	Aire a presión	Aceite	Corriente eléctrica
MODELOS	Cilindros Neumáticos Motor de paletas Motor de pistón	Cilindros Hidráulicos Motor de paletas Motor de pistones axiales	Cilindros Eléctricos Motor de paletas Motor de pistones axiales
VENTAJAS	Baratos Rápidos Sencillos Robustos	Rápidos Alta relación potencia-peso Auto-lubricantes Alta capacidad de carga	Precisos Fácil control Sencilla instalación Silenciosos
DESVENTAJAS	Dificultad de control continuo Instalación especial Ruidoso	Difícil mantenimiento Instalación especial Frecuentes fugas Caros	Potencia limitada
IMAGEN			

TABLA 1.1 *Actuadores*

1.7.3 TRANSMISIÓN

Entre el actuador y el punto o eje de salida es necesario transmitir el movimiento, ya sea aumentando o reduciendo la velocidad del actuador. Dependiendo la tarea que se va a programar al manipulador, es necesario hacer un estudio para elegir entre tantos elementos de transmisión de movimiento el indicado, para evitar fallas en el equipo. [22]

1.7.4 SENSORES

Uno de los sistemas importantes que no debe faltar hoy en día para la industria 4.0, es el sistema sensorial, que se encarga básicamente de que el robot tenga conocimiento de su entorno de trabajó, su posición y la velocidad en sus articulaciones. [57]

1.7.5 CONTROLADOR

Para poder comunicar al manipulador con el software de programación, es necesario de un controlador. También los controladores tienen la ventaja ser ocupados con un simulador, esto sirve para poder observar el comportamiento del robot antes de usar el programa en un manipulador real y así evitar pérdidas de material o fallas al mecanismo. [53]

1.8 SIMULADORES

De acuerdo con la Real Academia Española un simulador es *"…Aparato que reproduce el comportamiento de un sistema en determinadas condiciones, aplicado generalmente para el entrenamiento de quienes deben manejar dicho sistema*". [50]
Un simulador nos acerca a la realidad, nos crea entornos en los cuales puede que sea muy difícil de acceder, simula un dispositivo en determinadas condiciones y da resultados de su comportamiento, con esto se ahorra tiempo a la hora de fabricar un sistema, debido a que se reduce el número de errores en el proceso de fabricación del prototipo. También sirve para capacitar al personal de trabajo antes de entrar en contacto con el dispositivo real, así se evitan accidentes o incluso dañar el equipo. [42]

En algunas escuelas se utilizan para la enseñanza a los alumnos, acercándolos a un entorno similar al de la vida real. Siemens una empresa de automatización, está teniendo acuerdos con la Secretaría de educación Pública en Querétaro para donar equipo de simulación, para preparar a los alumnos a la industria 4.0. [55]

Los simuladores son utilizados en distintas áreas. Por ejemplo, en el transporte, las escuelas de manejo los utilizan para enseñar a las personas a conducir sin tener la necesidad de un carro, la ventaja de usar simuladores para las clases de manejo son varias, evita accidentes con otros automovilistas, daños al automóvil, también

se pueden generar varios escenarios como ciudades transitadas, carreteras, montañas, etc., preparando al usuario a conducir en cualquier entorno. [58]

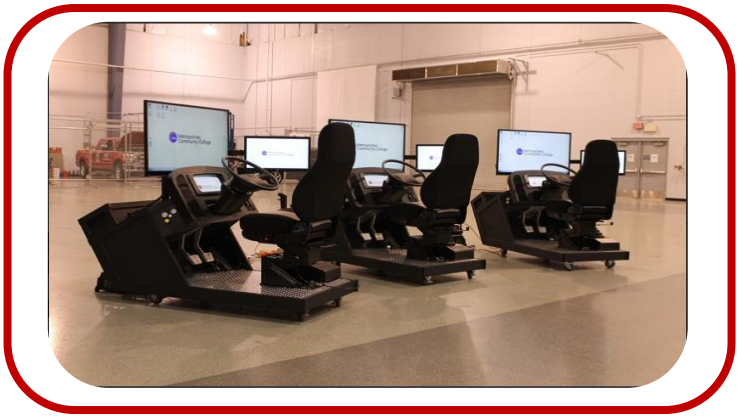


Ilustración 16. *CloudLabs, Hardware para simulación de transporte, Recuperado en: http://www.etechsimulation.com/*

Los militares utilizan simuladores para preparar a los soldados en situaciones que pueden llegar a ocurrir o para mejorar sus habilidades, con ayuda de estas herramientas se puede simular guerras, convoyes, práctica de tiros, tripulación de vehículos militares, etc. [59]

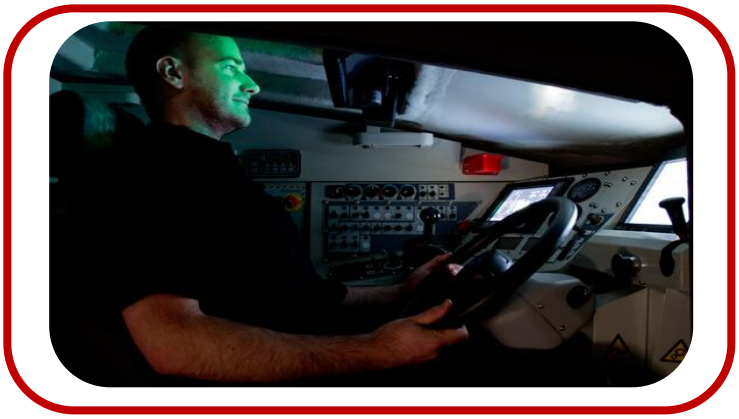


Ilustración 17. *THOROUGHTEC SIMULATION, Hardware para simulación militar, Recuperado en: http://www.thoroughtec.com/es/cyberwar-simuladores-militares/*

En la medicina se usan simuladores para mejorar la práctica de los residentes, el Dr. Gerhard Heinze Martin, jefe de la subdivisión Especializaciones Médicas Posgrado, destaco que la incorporación de estas tecnologías es sumamente importante para la facultad de medicina en la UNAM, los médicos tendrán un

entrenamiento mucho más rígido y tendrán la habilidad de operar de una forma adecuada para la garantizar la seguridad de los pacientes.[36]

Ilustración 18. *Centro de Enseñanza por Simulación de Posgrado UNAM, Hardware para simulación médica, Recuperado en: http://gaceta.facmed.unam.mx/index.php/2019/05/22/nuevos-simuladores-para-entrenamiento-de-medicos-residentes/*

En la aviación se utilizan para capacitar a los pilotos para sus primeros vuelos, evitando que cometan errores o trágicos accidentes a la hora de pilotear un avión real. [33]

Ilustración 19. *ASTA(2012), Hardware para simulación de aviación, Recuperado en: https://theaviationist.com/2012/11/06/asta/*

Para la Robótica, se pueden encontrar varios simuladores en el mercado, algunos son de libre uso y otros tienen un costo por licencia. Uno de ellos es RobotStudio de la Empresa ABB. Este software de simulación permite realizar la programación de un robot sin necesidad de estar conectado al robot o de tenerlo físicamente,

además se puede programar desde la comodidad de una oficina. Esta herramienta permite realizar tareas de capacitación, programación, sin necesidad de interrumpir la producción. Cuenta con realidad aumentada para visualizar al robot mediante hologramas y esto permite observar las fallas del equipo sin necesidad de tenerlo cerca.[4]

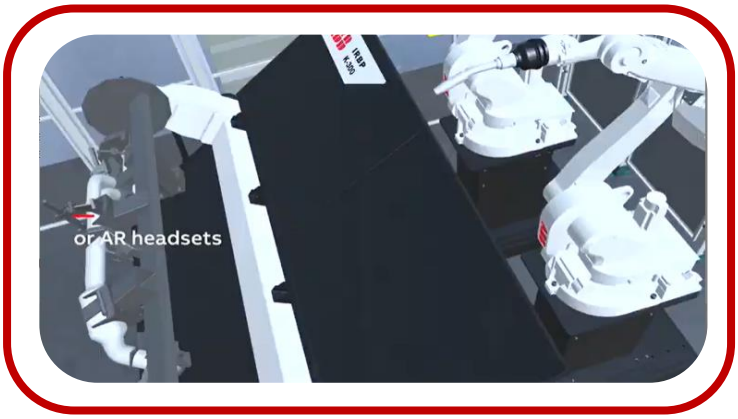


Ilustración 20. *ABB, Simulación de Robots por hologramas, Recuperado en: https://new.abb.com/products/robotics/es/robotstudio*

1.8.1 USO DE SIMULADORES EN LA EDUCACIÓN

"*El uso de simuladores educativos proporciona herramientas de apoyo a la formación y aprendizaje del alumno ya sea en programas presenciales, a distancia y/o virtuales*". [47] Durante la pandemia por el coronavirus (COVID-19), varias escuelas se vieron afectadas de manera significativa, pues los profesores no contaban con herramientas necesarias para impartir en línea su materia. El uso de simuladores virtuales no solamente apoya las clases de manera presencial, si no también, para clases virtuales.

Aunque se tengan pocos estudios sobre la efectividad del uso de simuladores en el ámbito educativo, los pocos descubrimientos que se tiene sobre este tema han dado resultados muy satisfactorios donde se ha observado que su uso potencializa el aprendizaje de los alumnos además de aumentar su motivación hacia la materia.[47]

Un estudio realizado por la escuela Iberoamericana, sobre de como el uso de simuladores influye en el conocimiento de los alumnos. Se obtuvieron resultados satisfactorios. En una escala donde 1 era muy negativo, hasta 6 era muy positivo

se obtuvieron resultados notables en los puntos de: “Originalidad”, “Favorece el proceso de aprendizaje” y “Adecuación al desempeño profesional” donde se obtuvieron 5.15, 5 y 5.04 puntos respectivamente. [13]

La investigación desarrollada por departamento de Investigación e Innovación en Tecnología y Educación del Tecnológico de Monterrey se determinó que el empleo de material de simulación cambio el ambiente enseñanza-aprendizaje en:

- Facilidad de implementación.
- Modelación de situaciones reales.
- Función motivadora.
- Apoyo didáctico al docente. [17]

La manera de enseñar en las escuelas tiene que actualizarse, los jóvenes aprenden de diferentes maneras, a las que se utilizaba décadas anteriores. “… *El hecho de que las generaciones actuales han nacido de la mano del ciberespacio, del aprendizaje autónomo, de los juegos de vídeo y demás, permite reconocer que el sistema educativo y las estrategias de aprendizaje de los estudiantes de ingeniería, deben partir del conocimiento de la forma como aprenden los jóvenes en la actualidad*”.[16] Se tienen que buscar una manera de enseñar donde no se vea a la tecnología como un enemigo sino como un aliado.

1.9 COMPARACIÓN DE COSTOS ENTRE UTILIZAR UN ROBOT INDUSTRIAL CONTRA UN SIMULADOR PARA LA ENSEÑANZA.

Adquirir un robot industrial para que los alumnos aprendan a programarlo, ejecutar una trayectoria, o hacer que el robot efectué una tarea, no es nada barato. En la Tabla 1.2 y 1.3 se observa una cotización realizada con Miguel Bórquez asesor de ventas de la empresa ABB (Datos del vendedor en el Apéndice 4), de un robot industrial IRB-120 con una celda “FlexTrainer” que se utiliza para enseñar y capacitar al personal.

La celda de entrenamiento FlexTrainer cuenta con 3 versiones para aplicaciones específicas:

1. Soldadura

2. Manipulación de materiales

3. Sistema guiado por visión robótica.[1]

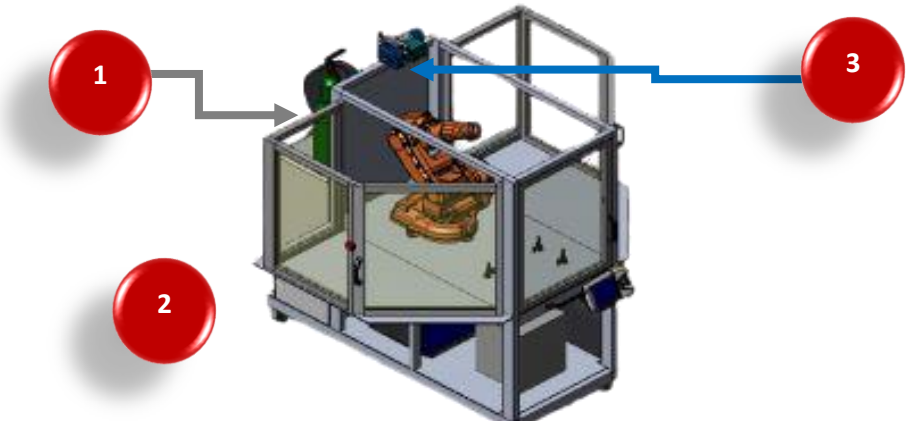


Ilustración 21. *Celda FlexTrainer de la marca ABB*

ROBOT CON CELDA PARA UNIVERSIDADES O CAPACITACIÓN		
INCLUYE	**COSTO USD $**	**COSTO MXN $**
Robot Industrial IRB-120 **Celda FlexTrainer con software RobotStudio para 100 computadoras.**	$50,000	$1 USD = $ 21.33 MXN $ 50, 000 USD = $ 1,066,500 MXN
Total	**$50,000 USD**	**$ 1,066,500 MXN**

TABLA 1.2 *Costos de Robot IRB-140 con celda FlexTrainer*

Robot IRB-120 y licencia RobotStudio para uso Individual		
INCLUYE	**COSTO USD $**	**COSTO MXN $**
Robot Industrial IRB-120.	$13,500.00	$1 USD = $ 21.33 MXN $15,250 USD = $ 325,282 MXN
Licencia RobotStudio.	$1,750.00	
Total	**$15,250.00 USD**	**$ 325,282 MXN**

TABLA 1.3 *Costos de Robot IRB-140 sin celda FlexTrainer*

Por otro lado, Matlab es un software utilizado por ingenieros para analizar y diseñar sistemas, este instrumento es tan poderoso que permite desarrollar simulaciones de sistemas mecatrónicos, software de control, y más. [38] En la página oficial del Software de Matlab, se señalan los costos de las licencias para hacer uso de la herramienta, en la **Tabla 1.4** se presenta una cotización sobre una licencia estudiantil con las librerías necesarias para estudiar robótica. [16]

La UNAM cuenta con licencia estudiantil para el uso de estar herramienta, los alumnos la pueden obtener ingresando con su número de cuenta en el portal Software UNAM.

LICENCIA ESTUDIANTIL PARA EL SOFTWARE MATLAB		
Incluye	**Costo USD $**	**Costo MXN $**
MATLAB **Simulink** **Control System Toolbox** **Curve Fitting Toolbox** **DSP System Toolbox** **Image Processing Toolbox Instrument Control Toolbox** **Optimization Toolbox** **Parallel Computing Toolbox Signal Processing Toolbox** **Statistics and Machine Learning Toolbox** **Symbolic Math Toolbox**	$99.00	$1 USD = $ 21.33 MXN $ 99 USD = $ 2,111.67 MXN
Librerías para aprender robótica y simular robots ARTE (A Robotic Toolbox for Education)	$0	$1 USD = $ 21.33 MXN $ 10 USD = $ 213.3MXN
Robotics System Toolbox.	$10.00	
TOTAL	**$15,250.00 USD**	**$2,324.97 MXN**

TABLA 1.4 *Costo de una Licencia Estudiantil de Matlab para uso individual*

2. CAPÍTULO 2 USO DE MATLAB EN ROBÓTICA Y LA CINEMÁTICA DIRECTA E INVERSA DE ROBOT IRB-140

2.1 MATLAB.

Matlab es un Software con un lenguaje de programación basado en matrices, dado que el tipo de dato básico que resuelve es una matriz. [31] Esta herramienta es muy utilizada por ingenieros y científicos, tiene un lenguaje muy potente para realizar cálculos matemáticos complejos, permite realizar gráficas para facilitar la visualización de datos y también realizar aplicaciones interactivas con interfaces de usuario personalizadas. Además, cuenta con “Toolboxes” como “ARTE Robotics”, “Robotics Toolbox de Peter Corke”, complementarios para aplicaciones de investigación científica e ingeniería.

Matlab es muy utilizado para aplicaciones como:

- Análisis de datos
- Deep Learnig
- Robótica
- Visión Artificial
- Procesamiento de señales
- Fianzas cualitativas
- Sistemas de control. [38] [40]

2.1.1 GUI (INTERFACES GRÁFICAS DE USUARIO)

Es una herramienta de Matlab que permite crear interfaces graficas de usuario para desarrollar aplicaciones personalizadas. Las personas que no cuentan con el conocimiento del lenguaje de programación de Matlab puedan hacer uso de estas aplicaciones de una manera muy fácil, ahorrando el tiempo de aprender un lenguaje nuevo y de estar escribiendo a cada rato los comandos para ejecutar una función. [39]. La ventaja de las GUI es que se pueden crear ejecutables para que otras personas que no cuenten con Matlab puedan hacer uso de las aplicaciones. [41]

2.2 MATLAB PARA LA ROBÓTICA

"Los investigadores y los ingenieros de robótica utilizan MATLAB y Simulink para diseñar y ajustar los algoritmos, modelizar sistemas del mundo real y generar código automáticamente, todo desde un mismo entorno de software". [44] El software permite desarrollar varias funciones para el estudio, el diseño y el control no solo de robots manipuladores, sino también de robots móviles.

Existen tres toolboxes que son de gran ayuda para el estudio de la robótica con Matlab, los cuales son:

- Robotics System Toolbox.
- Robotics Toolbox por Peter Corke.
- ARTE: A Robotics Toolbox for Education.

2.2.1 ROBOTICS SYSTEM TOOLBOX

Este toolbox adicional a Matlab permite desarrollar varias funciones para el estudio, comprensión y visualización de robots manipuladores y móviles de una manera interactiva, cuenta con una amplia variedad de comandos para realizar el diseño, el control y el análisis cinemático y dinámico en diferentes entornos.

Algunas de las acciones que se pueden desarrollar con esta herramienta son:

- Crear modelos de robots propios o cargar modelos de robots comerciales con los que cuenta la librería (**Véase el Apéndice 5**), para poder desarrollar el estudio de la cinemática y dinámica.
- Simula el seguimiento de trayectorias para los robots manipuladores.
- Sincroniza modelos de Simulink con simuladores 3D para generar versiones virtuales de entornos de trabajo.
- Genera el código para la comprobación de colisiones y realiza pruebas rápidas con el hardware en físico. [45]

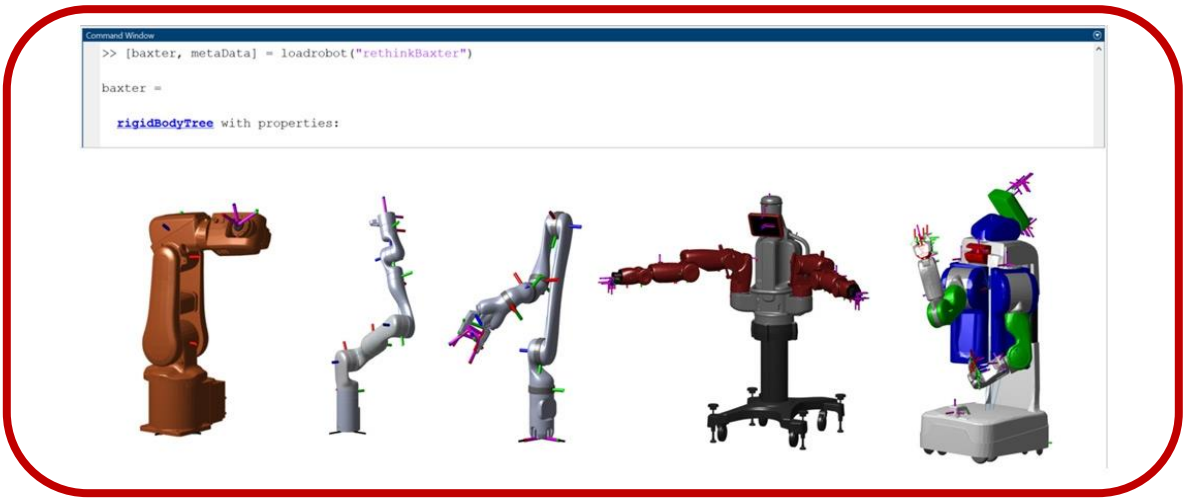


Ilustración 22. *Interfaz de Matlab con Robotics System Toolbox, Recuperado en: https://es.mathworks.com/products/robotics.html*

2.2.2 ROBOTIC TOOLBOX POR PETER CORKE

Esta librería fue desarrollada por Peter Corke, un profesor distinguido de la Universidad de Tecnología de Queensland del Centro QUT de Robótica, miembro de la Academia Australiana de Tecnología e Ingeniería y miembro de la Federación Internacional de Robótica (IFRR). [19]

Este toolbox proporciona ejemplos de robots manipuladores conocidos de la marca Kinova, Universal Robots y además contiene manipuladores clásicos como el robot Stanford que tiene en su estructura una junta prismática. También permite desarrollar funciones para robots móviles, como desarrollo del algoritmo de planificación de ruta, localización y construcción de mapa.[18]

2.2.3 ARTE (A ROBOTICS TOOLBOX FOR EDUCATION)

Por otro lado, se encuentra librería ARTE (Una herramienta de Robótica para la Educación por sus siglas en inglés A Robotics Toolbox for Education). Desarrollado por Arturo Gil y estudiantes de robótica en el año 2012 en la Universidad Miguel Hernández (UMH) en España. Esta herramienta, a comparación de las anteriores, solamente está enfocada a robots manipuladores.

Las principales características de esta herramienta son:

- Muestra de manera visual los sistemas de referencia en el manipulador para obtener los parámetros de Denavit-Hartenberg.
- Se pueden describir y observar la posición, la velocidad y aceleración de las coordenadas de la articulación cuando se realiza un movimiento.
- Se pueden hacer simulaciones de programación con los distintos manipuladores.
- Se pueden incluir robots diseñados por uno mismo.
- Cuenta con cinemática directa e inversa de robots paralelos. [63]

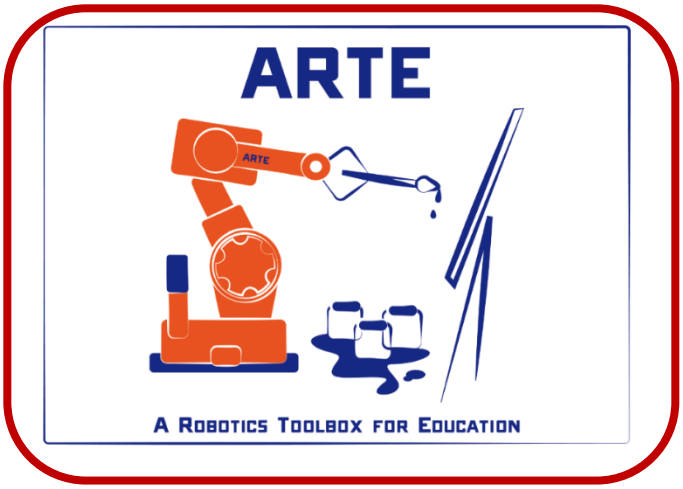


Ilustración 23. *Logo del toolbox para Matlab, Recuperado en: https://arvc.umh.es/arte/index_en.html#list*

2.3 ROBOT IRB-140

Para el estudio de la cinemática directa e inversa y el desarrollo de las simulaciones en los programas desarrollados, se utilizará como ejemplo el Robot IRB-140 de la marca ABB. Es un robot para uso industrial que realizar distintas tareas dependiendo el efector final que se le ajuste. Puede realizar trabajos como paletizado, llegando a cargar hasta 5 kg, es un robot compacto pues llega a medir en su posición "HOME" 810mm de altura (**Ilustración 24**), a comparación del robot IRB-2400 que también es un robot multiusos, pero para tareas más pesadas, pues llega a cargar hasta 20 Kg, este robot mide 1564mm de altura en su posición "HOME" (**Ilustración 25**). [2] [3]

Para conocer más robots de la marca ABB véase el **Apéndice 6**.

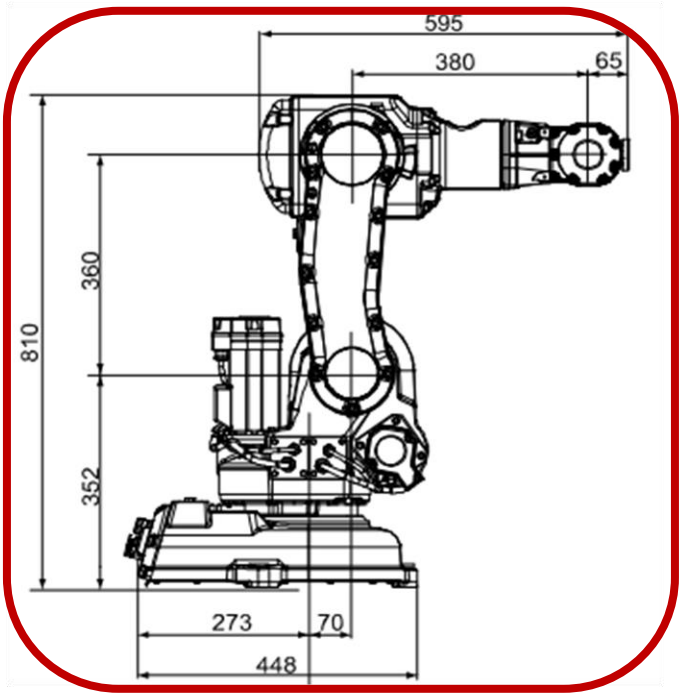


Ilustración 24. *ABB (2019), Medidas del Robot IRB 140, Recuperado en: https://new.abb.com/products/robotics/es/robots-industriales/irb-140*

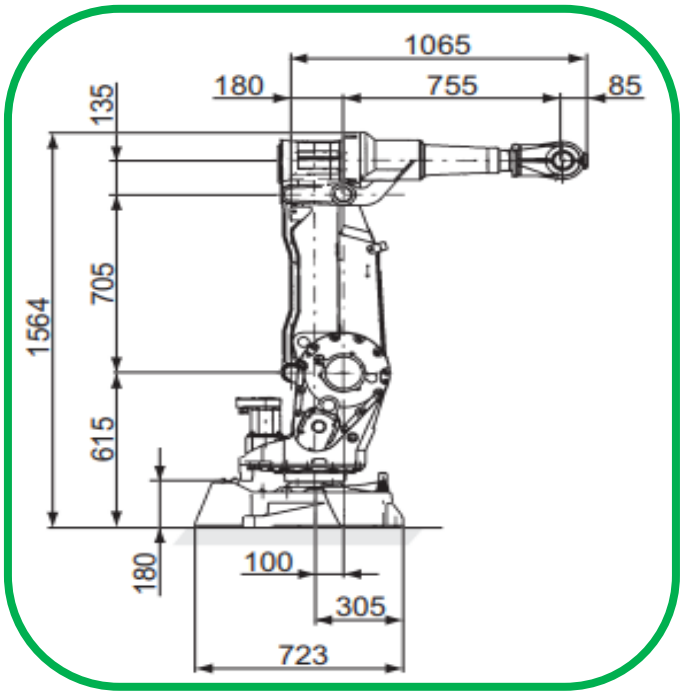


Ilustración 25. *ABB (2019), Medidas del Robot IRB 140, Recuperado en: https://new.abb.com/products/robotics/es/robots-industriales/irb-2400*

2.3.1 MORFOLOGÍA DEL ROBOT IRB-140

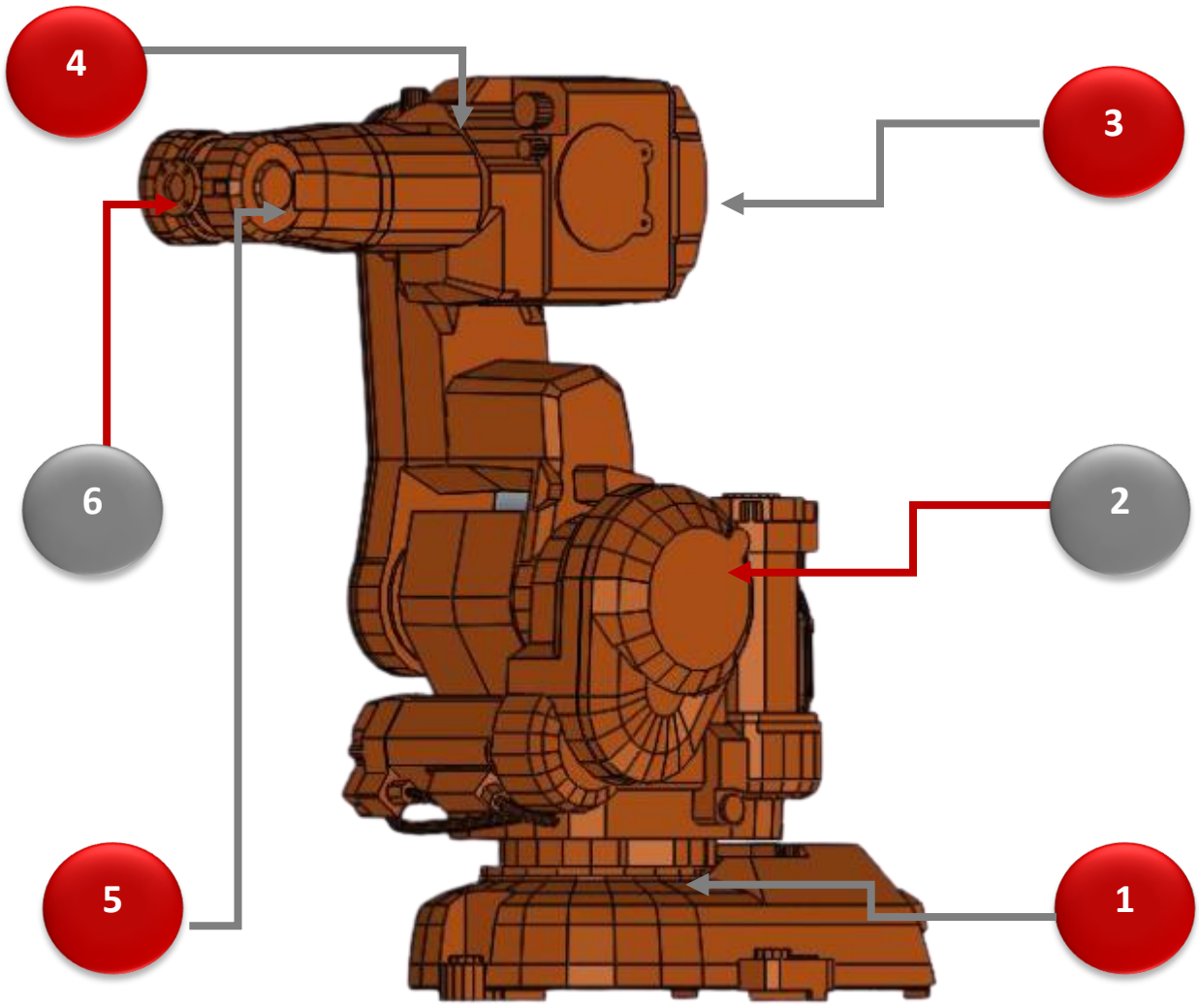
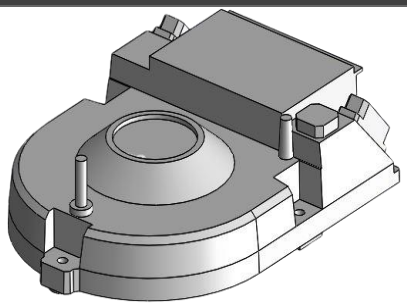
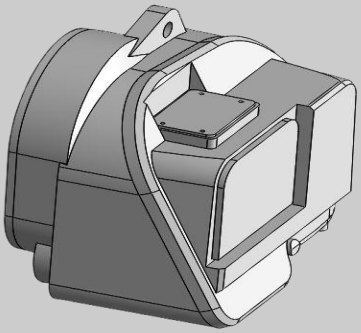
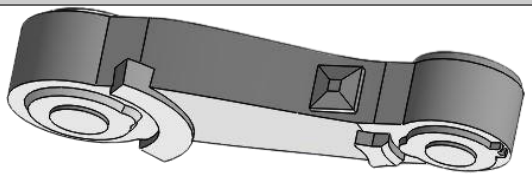
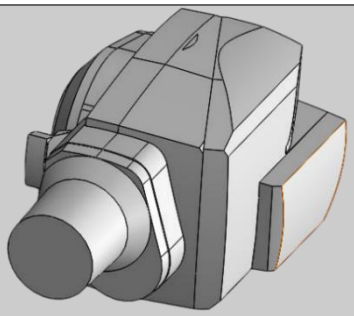
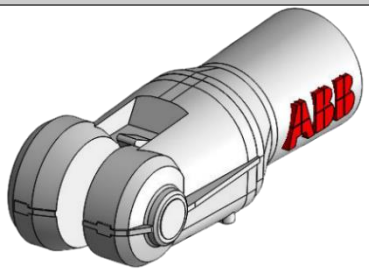


Ilustración 26. *Localización de las articulaciones*

JUNTAS	
NO.DE JUNTA	**MOVIMIENTO**
1	Rotación
2	Oscilación
3	Oscilación
4	Rotación
5	Oscilación
6	Rotación

Tabla 1.5 *Articulaciones del robot, Recuperado en: https://new.abb.com/products/robotics/es/robots-industriales/irb-140/datos*

La estructura del robot se conforma por los eslabones que se presentan en la **Tabla 1.6**.

ESLABONES		
#	NOMBRE	IMAGEN
1	BASE	
2	ESLABÓN 1	
3	ESLABÓN 2	
4	ESLABÓN 3	
5	ESLABÓN 4	

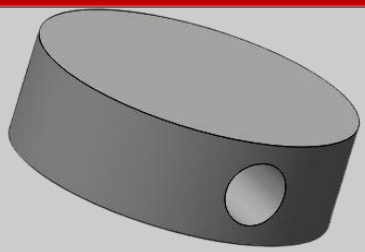
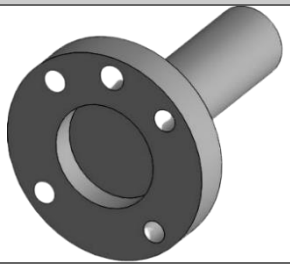
ESLABONES		
6	ESLABÓN 5	
7	EFECTOR FINAL	

Tabla 1.6 Eslabones que conforman la estructura del robot IRB-140
https://new.abb.com/products/robotics/es/robots-industriales/irb-140/datos

Su alcance máximo con el brazo estirado horizontalmente, instalado en el piso es de 0.810 metros, tiene un peso de 98kg excluyendo los cables del controlador, puede ser montado de distintas maneras, ya sea en el piso, en una pared o suspendido en un techo, dependiendo del tipo de montaje que tenga, cambiara su campo de trabajo y el valor de las cargas soportadas.

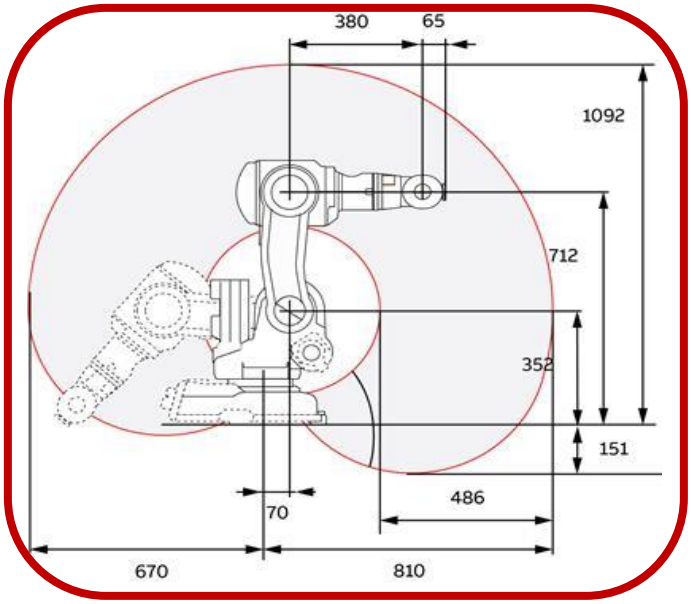


Ilustración 27. *IRB 140 (2019), Área de trabajo instalado en el piso, Recuperado en: https://new.abb.com/products/robotics/es/robots-industriales/irb-140/datos*

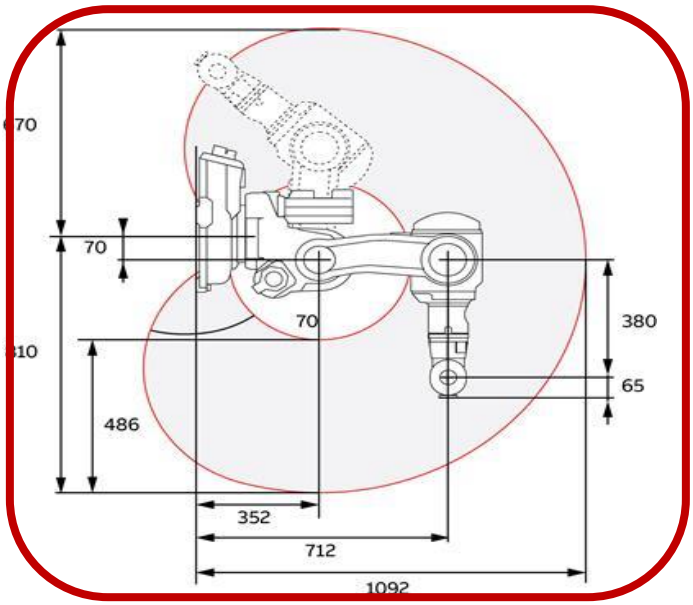


Ilustración 28. *IRB 140 (2019), Área de trabajo instalado en una pared, Recuperado en: https://new.abb.com/products/robotics/es/robots-industriales/irb-140/datos*

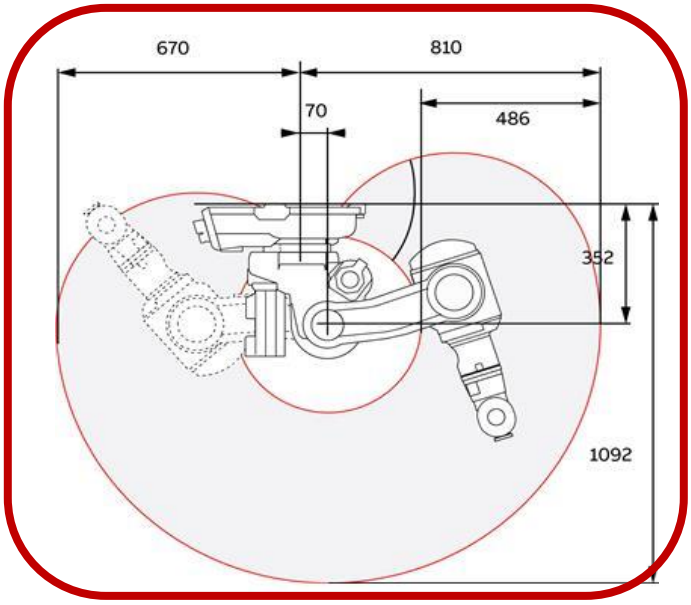


Ilustración 29. *IRB 140 (2019), Área de trabajo instalado en el techo, Recuperado en: https://new.abb.com/products/robotics/es/robots-industriales/irb-140/datos*

Las **tablas 1.7, 1.8, 1.9** muestran los valores de las fuerzas soportadas en operación y las cargas máximas del robot. Es muy importante observar el valor de las cargas máximas que el desarrollador coloca en el manual, pues protege al robot y a sus actuadores a que no sean dañados.

FUERZAS SOPORTADAS MONTADO EN EL PISO		
Fuerza	**Carga de Resistencia en Funcionamiento**	**Carga Máxima**
Fuerza en el plano XY	± 1020 N	± 2000 N
Fuerza en el eje Z	-1000 ± 620 N	- 1000 ± 1250 N
Torque en el plano XY	± 700 Nm	± 1500 Nm
Torque en el eje Z	± 250 Nm	± 470 Nm

Tabla 1.7 *Fuerzas soportadas instalado en el piso, Recuperado en: ABB (2019), Product specification IRB 140, pág. 17.*

FUERZAS SOPORTADAS MONTADO EN UNA PARED		
Fuerza	**Carga de Resistencia en Funcionamiento**	**Carga Máxima**
Fuerza en el plano XY	± 1750 N	± 2800 N
Fuerza en el eje Z	± 850 N	± 1600 N
Torque en el plano XY	± 1020 Nm	± 1710 Nm
Torque en el eje Z	± 250 Nm	± 485 Nm

Tabla 1.8 *Fuerzas soportadas instalado en la pared, Recuperado en: ABB (2019), Product specification IRB 140, pág. 18.*

FUERZAS SOPORTADAS SUSPENDIDO		
Fuerza	**Carga de Resistencia en Funcionamiento**	**Carga Máxima**
Fuerza en el plano XY	± 1020 N	± 2000 N
Fuerza en el eje Z	+1000 ± 620 N	+1000 ± 1250 N
Torque en el plano XY	± 700 Nm	± 1500 Nm
Torque en el eje Z	± 250 Nm	± 470 Nm

Tabla 1.9 *Fuerzas soportadas suspendido, Recuperado en: ABB (2019), Product specification IRB 140, pág. 18.*

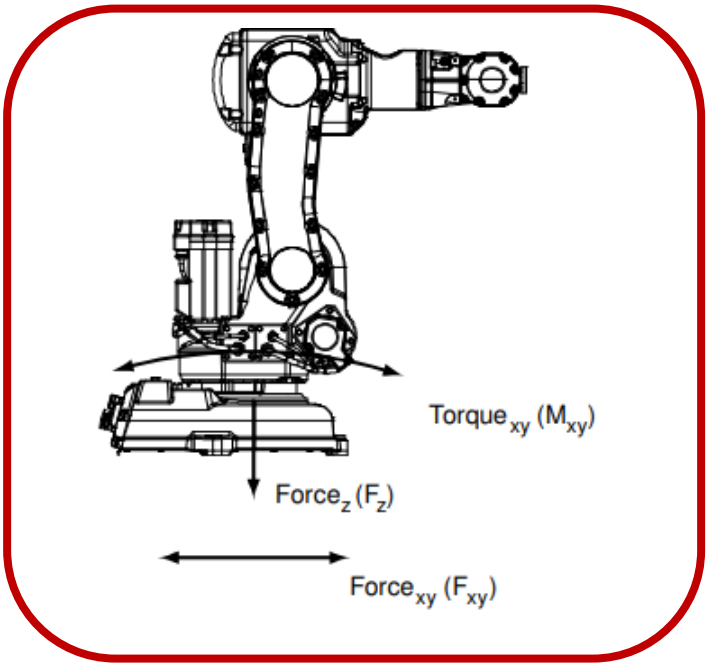


Ilustración 30. *IRB 140 (2019), Representación gráfica torques y fuerzas en XY, Recuperado en: ABB (2019), Product specification IRB 140, pág. 18.*

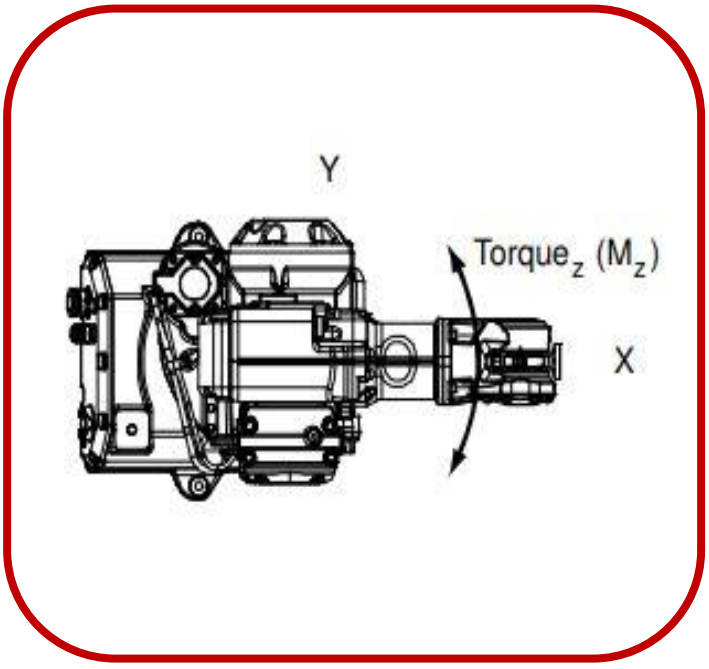


Ilustración 31. *IRB 140 (2019), Representación gráfica Torque en Z, Recuperado en: ABB (2019), Product specification IRB 140, pág. 18.*

2.4 CINEMÁTICA DE LOS MANIPULADORES

"La cinemática es la ciencia que estudia el movimiento sin considerar las fuerzas que lo ocasionan. Donde se estudiala posición, la velocidad, la aceleración y todas las derivadas de alto orden de las variables de posición". [21]

Para que un robot realice una tarea o trabajo en la industria, es necesario establecer la posición y la orientación del efector final. Pero existen 2 problemas fundamentales para resolver en la cinemática del robot, los cuales son:

- **Cinemática directa:** Se conocen los valores de las articulaciones, el problema radica en encontrar la posición y orientación del efector final.
- **Cinemática inversa:** Se conoce la posición y orientación del efector final, el problema es encontrar los ángulos que deben tomar las articulaciones para llegar a dicha posición. [54]

Para hallar la posición de un robot en un espacio tridimensional se necesitan de 6 parámetros, de las cuales 3 representan la posición cartesiana **(x,y,z)**, y las otras 3 representan la orientación de la herramienta de trajo o Efector final dada por los ángulos de Euler **(β,α,ϒ).** [51]

2.4.1 CINEMÁTICA DIRECTA DE LOS MANIPULADORES

Un robot se puede considerar como una cadena cinemática formado por eslabones, unidos por articulaciones unos a otros, lo cual se puede establecer un sistema de referencia en la base del robot fijo para hallar la posición de cada uno de los eslabones con respecto a dicho sistema. El problema cinemático directo en pocas palabras es encontrar la matriz de transformación homogénea (**T**) que relacione la posición y orientación del efector final con el sistema de referencia fijo. Existen varios métodos para hallar el modelo cinemático, algunos de los más utilizados son:

- Método de transformación homogénea.
- Método Denavit-Hartenberg. [10]

2.4.1.1 MÉTODO DENAVIT-HARTENBERG Y USO DE MATLAB CON ROBOT IRB-140

Denavit y Hartenberg propusieron un método matricial para deducir las ecuaciones de la cinemática directa de robots manipuladores. El método consiste en escoger adecuadamente los sistemas de coordenadas para pasar del sistema de un eslabón a otro por 4 transformaciones básicas de rotación y translación.

a otro por 4 transformaciones básicas de rotación y translación.

Las 4 transformaciones que se utilizan para pasar de un sistema $\{Si-1\}$ a un sistema $\{Si\}$ son:

1. Una rotación por el eje $Zi-1$ (θi).
2. Una translación a lo largo de eje $Zi-1$(di)
3. Una translación a lo largo del eje Xi (ai)
4. Rotación por el eje Xi (αi).

Como el producto de matrices no es conmutativo (El orden de la multiplicación altera el resultado), las transformaciones se deben hacer en orden sucesivo del menor al mayor. La matriz del sistema **{Si-1}** con el sistema **{Si}** está dada por:

$$^{i-1}A_i = T(z, \theta_i)\ T(0, 0, d_i)\ T(a_i, 0, 0)\ T(x, \alpha_i) \ldots (1)$$

Donde:

$$T(z, (\theta i) = \begin{bmatrix} C\theta i & -S\theta i & 0 & 0 \\ S\theta i & C\theta i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (2)$$

$$T(0, 0, di) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & di \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (3)$$

$$T(ai, 0, 0) = \begin{bmatrix} 0 & 0 & 0 & ai \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (4)$$

$$T(x, \alpha i) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\alpha i & -S\alpha & 0 \\ 0 & S\alpha & C\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (5)$$

Sustituyendo ecuación **(2)**, **(3)**, **(4)**, **(5)** en ecuación **(1)** se tiene:

$$^{i-1}\mathbf{A_i} = \begin{bmatrix} C\theta i & -S\theta i & 0 & 0 \\ S\theta i & C\theta i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & di \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & ai \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\alpha i & -S\alpha & 0 \\ 0 & S\alpha & C\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (6)$$

Resolviendo la multiplicación de las matrices se obtuvo:

$$ {}^{i-1}\mathbf{A}_i = \begin{bmatrix} C\theta i & -C\alpha i S\theta i & S\alpha i S\theta i & aiC\theta i \\ S\theta i & C\alpha i C\theta i & -S\alpha i C\theta i & aiS\theta i \\ 0 & S\alpha i & C\alpha i & di \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (7) $$

La matriz ${}^{i-1}\mathbf{A}_i$ relaciona los sistemas $\{S_{i-1}\}$ y $\{S_i\}$. Los parámetros de Denavit-Hartenberg deben ser elegidos al orden de las multiplicaciones, ver Ecuación (7), esto es conocido como el algoritmo de Denavit-Hartenberg, el cual se utiliza en este trabajo de tesis para describir la cinemática directa del robot IRB-140 como ejemplo.[10]

Dada la configuración y descripción de las articulaciones con las que cuenta el robot (**RRRRRR**) se describe el siguiente diagrama de flujo.

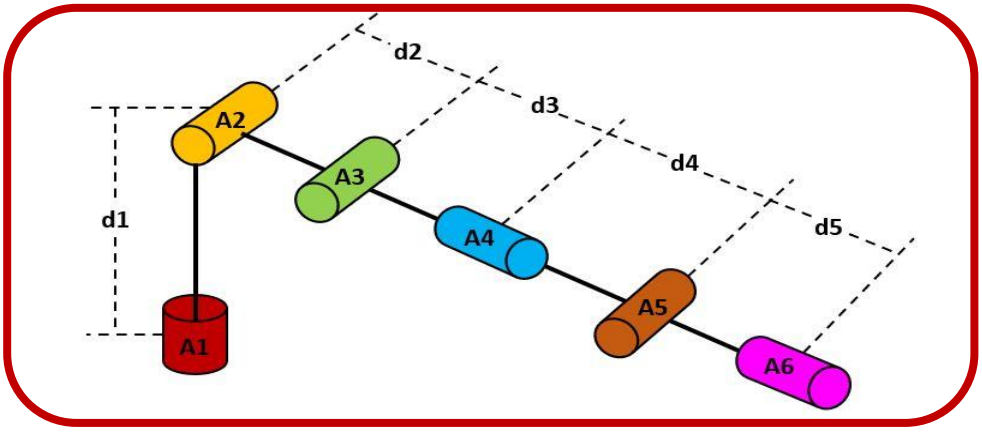


Ilustración 32. *Diagrama de las articulaciones del robot IRB-140*

Algoritmo de Denavit-Hartenberg para el robot IRB-140:

D-H 1. Enumerar los eslabones comenzando con **1** (primer eslabón móvil de la cadena) y acabando con $\mathbf{n}$ (último eslabón móvil). Se numerará como como eslabón **0** a la base fija del robot

D-H 2. Numerar las articulaciones comenzando por 1 (la correspondiente al primer grado de libertad) y acabando en n.

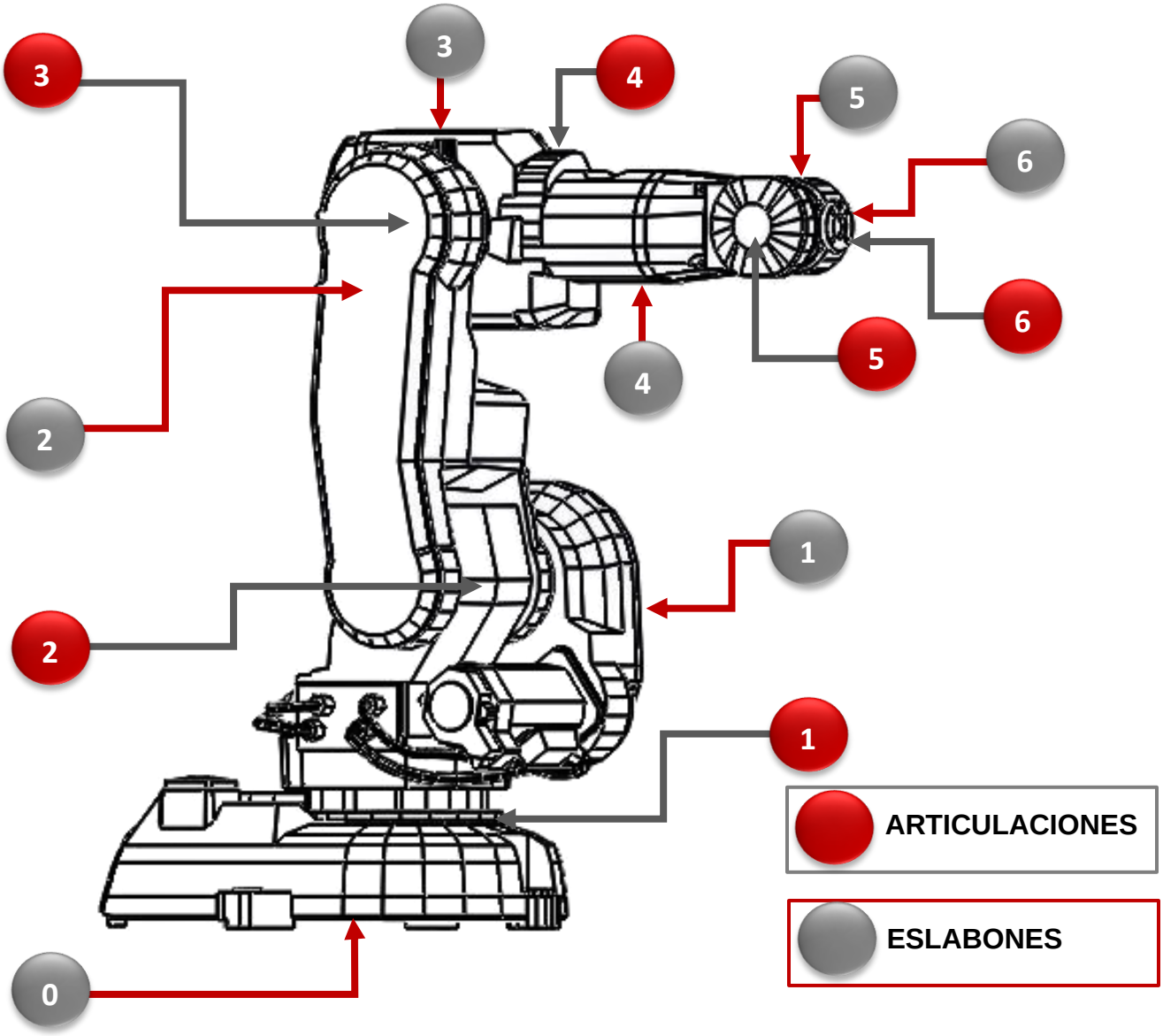


Ilustración 33. *Pasos D-H 1 y D-H 2*

D-H 3. Localizar el eje de cada articulación. Si esta es rotativa, el eje será su propio eje de giro. Si es prismática, será el eje a lo largo del cual se produce el desplazamiento.

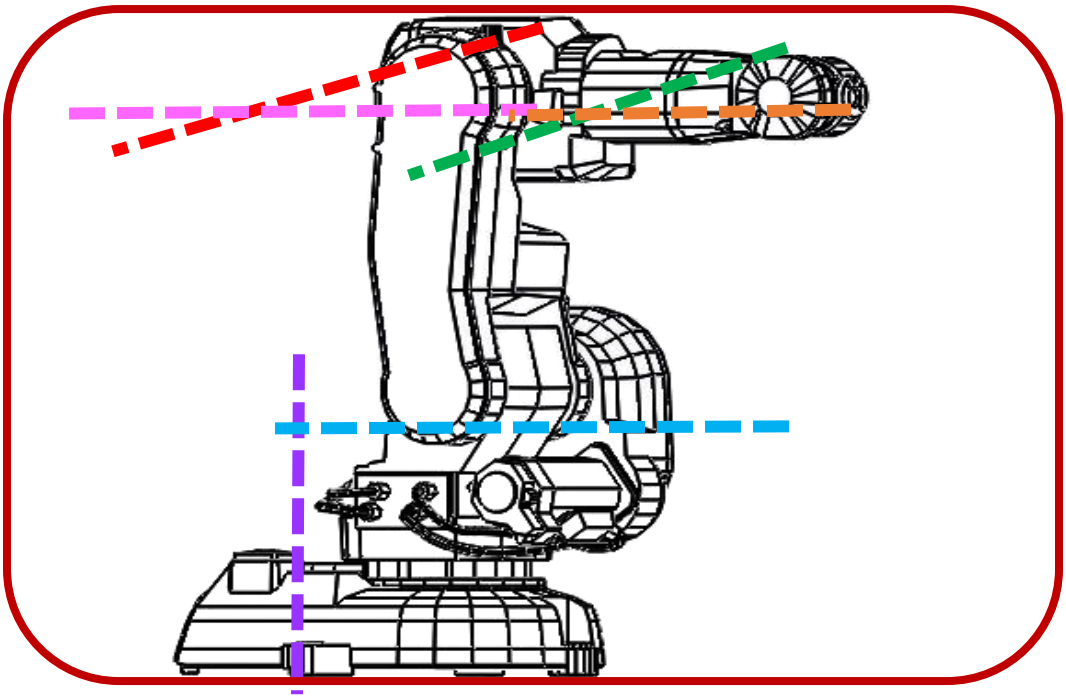


Ilustración 34. *Paso D-H 3*

DESCRIPCIÓN DE LOS EJES DE ARTICULACIÓN		
COLOR	**DESCRIPCIÓN**	**ARTICULACIÓN**
	Eje de la articulación 1	Rotativa
	Eje de la articulación 2	Rotativa
	Eje de la articulación 3	Rotativa
	Eje de la articulación 4	Rotativa
	Eje de la articulación 5	Rotativa
	Eje de la articulación 6	Rotativa

Tabla 1.10 *Descripción de los ejes de acción de las articulaciones*

D-H 4 Para i de 0 a $n-1$ situar el eje Zi sobre el eje de la articulación $i+1$

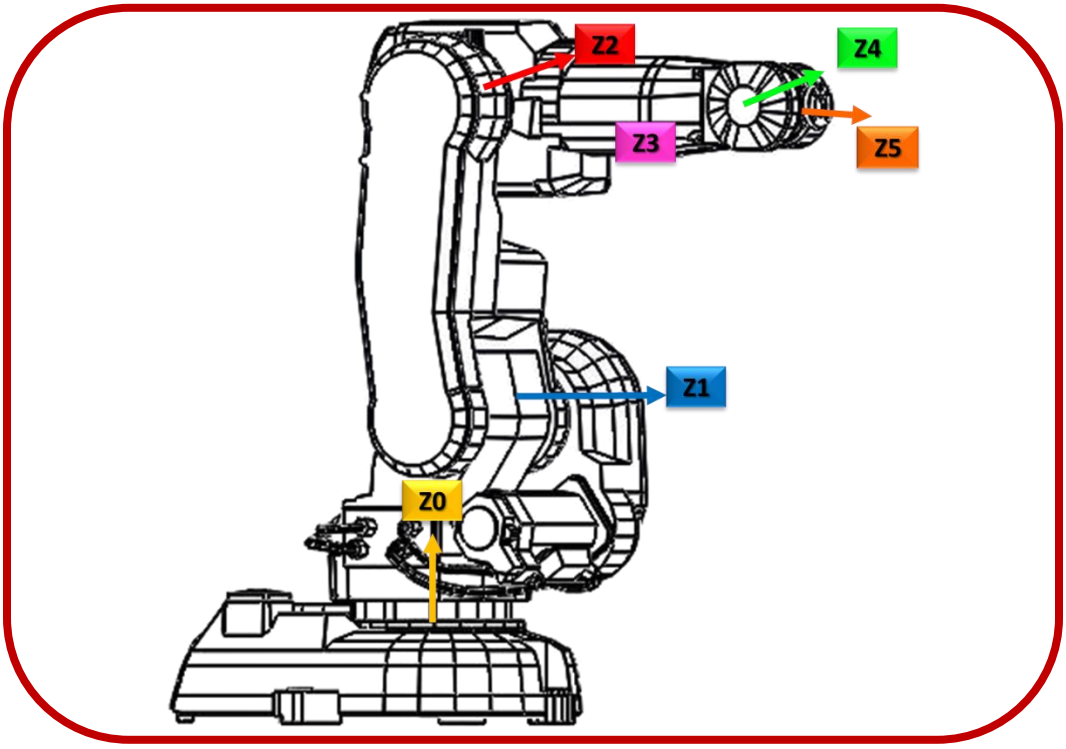


Ilustración 35. *Paso D-H 4*

D-H 5. Situar el origen del sistema de la base $\{S0\}$ en cualquier punto del eje $Z0$. Los ejes $X0$ y $Y0$ se situarán de modo que formen un sistema dextrógiro $z0$.

D-H 6. Para i de 1 a $n-1$, situar el sistema $\{Si\}$ (solidario al eslabón i) en la intersección del eje Zi con la línea normal común a $Zi-1$ y Zi. Si ambos ejes se cortasen se situaría $\{Si\}$ en el punto de corte. Si fuesen paralelos $\{Si\}$ se situaría en la articulación.

D-H 7. Situar Xi en la línea normal común a $Zi-1$ y Zi.

D-H 8. Situar Yi de modo que forme un sistema dextrógiro con Xi y Zi.

D-H 9 Situar el sistema $\{Sn\}$ en el extremo del robot de modo que Zn coincida con la dirección de $Zn-1$ y Zn sea paralelo.

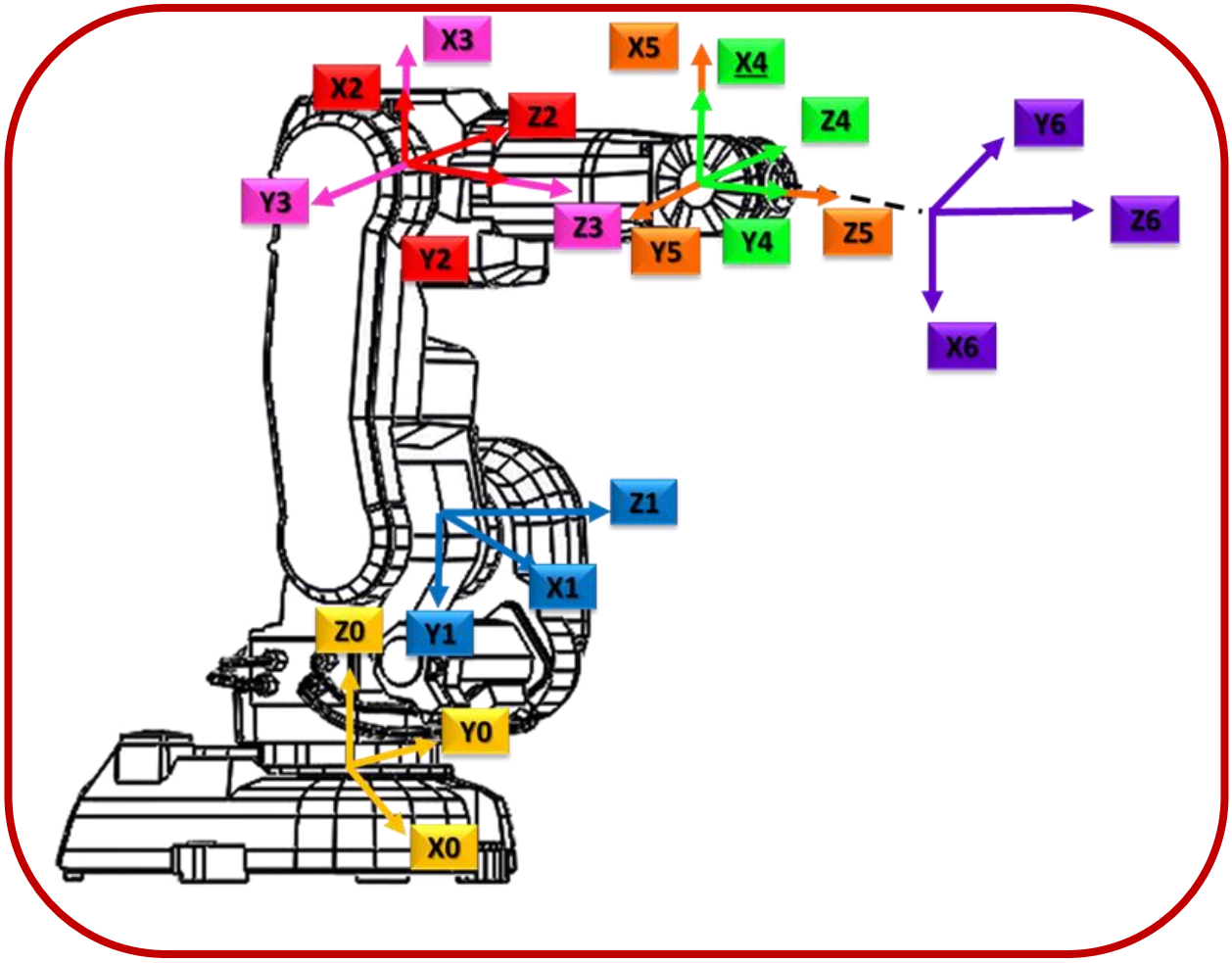


Ilustración 36. *Pasos D-H 5, D-H 6, D-H 7, D-H 8, D-H 9*

D-H 10 Obtener $\boldsymbol{\theta i}$ como el ángulo que hay que girar en torno a **Zi-1** para que **Xi-1** y **Xi** queden paralelos.

D-H 11 Obtener **di** como la distancia medida a lo largo de **Zi-1** que habría que desplazar **{Si-1}** para que **Xi** y **Xi-1** quedasen alineados.

D-H 12 Obtener **ai** como la distancia medida a lo largo de **Xi** (que ahora coincidiría con **Xi-1**) que habría que desplazar el nuevo {Si-1} para que su origen coincidiese con **{Si}.**

D-H 13 Obtener **αi** como el ángulo que habría que girar en torno a **xi** (que ahora coincidiría con **xi-1**), para que el nuevo **{Si-1}** coindícese totalmente con **{Si}.**

Tabla de parámetros Denavit-Hartenberg Robot IRB-140				
Sistema	θi	di (m)	ai (m)	αi
1	θ_1	D1	A1	-90°
2	θ_2-90	0	A2	0
3	θ_3	0	0	-90°
4	θ_4	D4	0	90°
5	θ_5	0	0	-90°
6	θ_6+180	D6	0	0

Tabla 1.11 *Parámetros de Denavit-Hartenberg para el Robot IRB-140*

Con las ilustraciones **37**, **38**, **39** se sustituyeron los valores de las cotas de cada imagen en la tabla **1.11** se obtuvo:

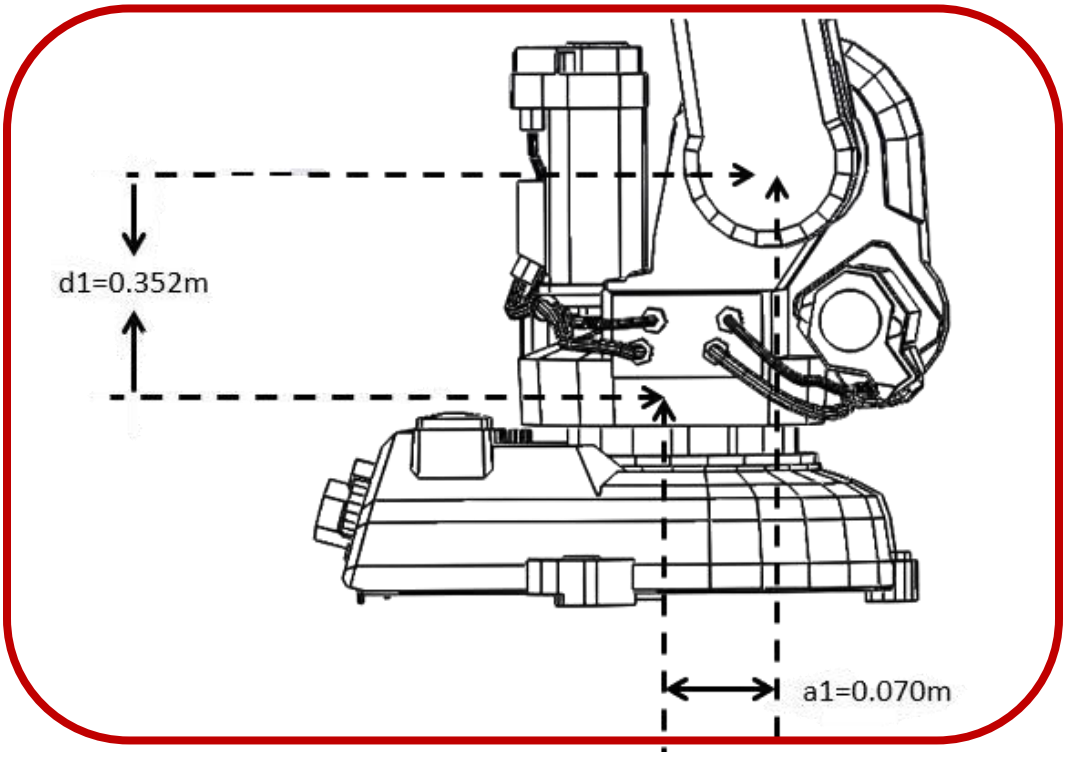


Ilustración 37. *Cotas de d1 y a1*

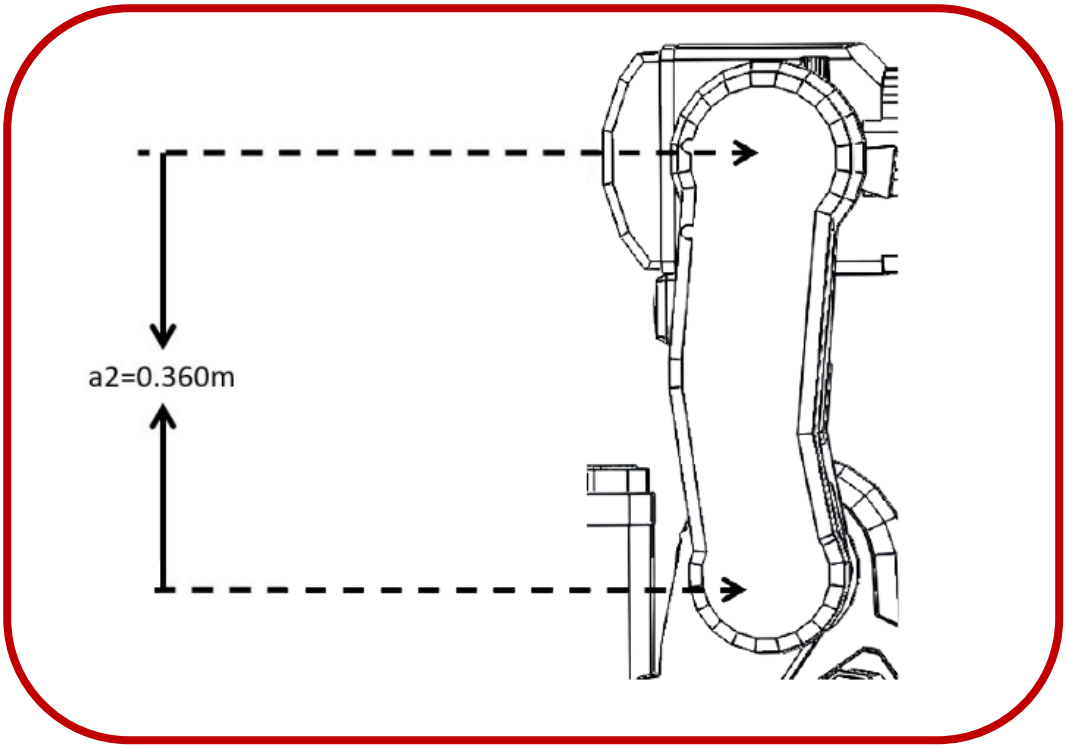


Ilustración 38. *Cota a2*

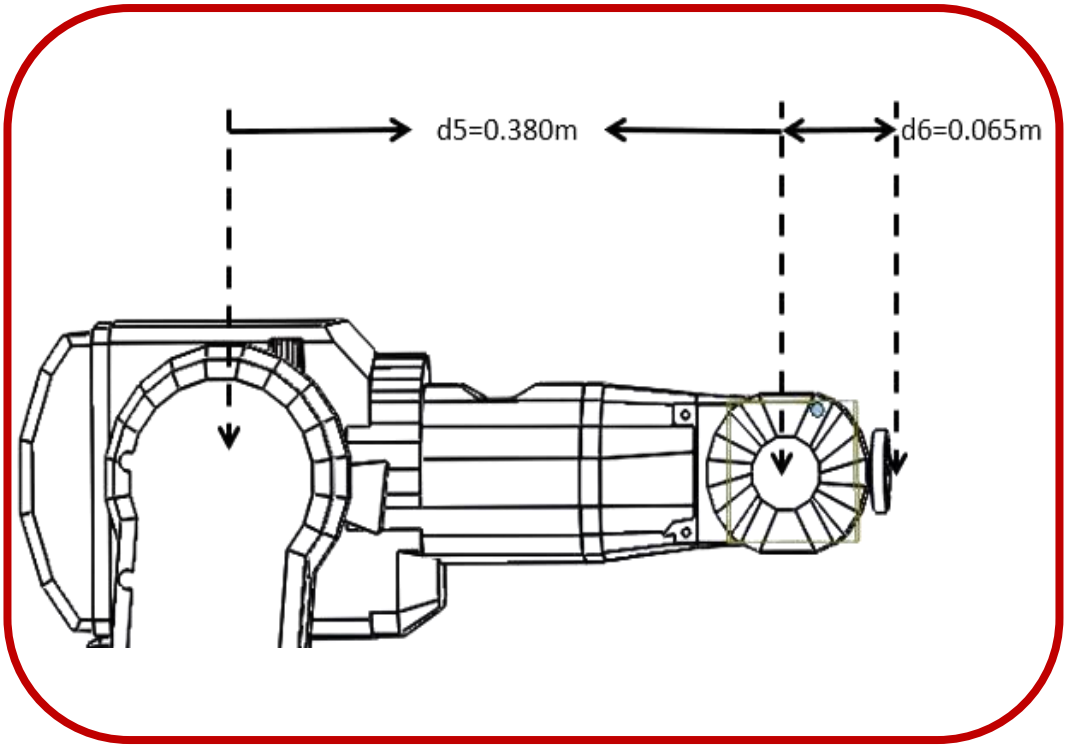


Ilustración 39. *Cotas d5 y d6*

Tabla de parámetros Denavit-Hartenberg Robot IRB-140				
Sistema	θi	di (m)	ai (m)	αi
1	θ_1	0.352	0.070	-90°
2	θ_2-90	0	0.360	0
3	θ_3	0	0	-90°
4	θ_4	0.380	0	90°
5	θ_5	0	0	-90°
6	θ_6+180	0.065	0	0

Tabla 1.12 *Parámetros de Denavit-Hartenberg para el Robot IRB-140 con valores numéricos*

D-H 14 Obtener las matrices de transformación $^{i-1}\mathbf{A}_i$ definidas en ecuación (7).

D-H 15 Obtener la matriz de transformación que relaciona el sistema de la base con el del extremo del robot $\mathbf{T} = {}^{0}\mathbf{A}_1, {}^{1}\mathbf{A}_2 \ldots {}^{n-1}\mathbf{A}_n$.

D-H 16 La matriz $\mathbf{T}$ que define la orientación (submatriz de rotación) y posición (submatriz de traslación) del extremo referido a la base en función de las $\mathbf{n}$ coordenadas articulares.

Para los pasos **D-H 14,15,16** se ocupó la herramienta Matlab para obtener la matriz de transformación homogénea que define la posición y orientación del efector final del robot IRB-140.

En Matlab hay dos maneras para obtener la matriz de transformación homogénea, las cuales son:

- Ingresando los valores uno por uno y sustituyendo en la ecuación (7).
- Utilizando la librería Robotic Toolbox de Matlab de Peter Corke.

A continuación, se presentan los dos métodos que se utilizaron para obtener la matriz de transformación homogénea.

Método 1

Se ingresó los parámetros de Denavit-Hartenberg eslabón por eslabón.

Se definieron los parámetros para el eslabón uno, en la ventana de comandos de Matlab.

```
>> %Parámetros D-H del Eslabón 1

>> t1=0;
>> d1=0.352;
>> a1=0.070;
>> ap1=-90;
```

Para representar una matriz en Matlab cada fila es separada por un “punto y coma (**;**)” y cada valor de la fila es separado por una “coma (**,**)”.

La matriz de transformación homogénea que representa el cambio del sistema del eslabón uno con el de la base, se escribió de la siguiente manera:

```
>>      A01=[cosd(t1),-cosd(ap1)*sind(t1),sind(ap1)*sind(t1),
a1*cosd(t1); sind(t1),cosd(ap1)*cosd(t1),-sind(ap1)*cosd(t1),
a1*sind(t1); 0, sind(ap1),cosd(ap1),d1;0,0,0,1]
```

Como resultado se obtuvo la matriz de transformación que relaciona el sistema 1 con el sistema 0:

```
A01 =

    1.0000         0         0    0.0700
         0         0    1.0000         0
         0   -1.0000         0    0.3520
         0         0         0    1.0000
```

... (8)

Se realizo los mismos pasos con los demás parámetros de la **Tabla 1.11.**

```
>>%Parámetros D-H del Eslabón 2

>> t2=-90;
>> d2=0;
>> a2=0.360;
>> ap2=0;
```

```
>>%Matriz de transformación homogénea 1A2

>>      A12=[cosd(t2),-cosd(ap2)*sind(t2),sind(ap2)*sind(t2),
a2*cosd(t2); sind(t2),cosd(ap2)*cosd(t2),-sind(ap2)*cosd(t2),
a2*sind(t2); 0, sind(ap2),cosd(ap2),d2;0,0,0,1]


          A12 =

                0    1.0000         0         0
          -1.0000         0         0   -0.3600
                0         0    1.0000         0
                0         0         0    1.0000
```

... (9)

```
>>%Parámetros D-H del Eslabón 3

>> t3=0;
>> d3=0;
>> a3=0;
>> ap3=-90;

>>%Matriz de transformación homogénea 2A3

>>      A23=[cosd(t3),-cosd(ap3)*sind(t3),sind(ap3)*sind(t3),
a3*cosd(t3); sind(t3),cosd(ap3)*cosd(t3),-sind(ap3)*cosd(t3),
a3*sind(t3); 0, sind(ap3),cosd(ap3),d3;0,0,0,1]

                  A23 =

                       1     0     0     0
                       0     0     1     0
                       0    -1     0     0
                       0     0     0     1
```

... (10)

```
>>%Parámetros D-H del Eslabón 4

>> t4=0;
>> d4=0.380;
>> a4=0;
>> ap4=90;

>>%Matriz de transformación homogénea 3A4

>>      A34=[cosd(t4),-cosd(ap4)*sind(t4),sind(ap4)*sind(t4),
a4*cosd(t4); sind(t4),cosd(ap4)*cosd(t4),-sind(ap4)*cosd(t4),
a4*sind(t4); 0, sind(ap4),cosd(ap4),d4;0,0,0,1]
```

```
A34 =

    1.0000         0         0         0
         0         0   -1.0000         0
         0    1.0000         0    0.3800
         0         0         0    1.0000
```
… (11)

```
>>%Parámetros D-H del Eslabón 5

>> t5=0;
>> d5=0;
>> a5=0;
>> ap5=-90;

>>%Matriz de transformación homogénea 4A5

>>      A45=[cosd(t5),-cosd(ap5)*sind(t5),sind(ap5)*sind(t5),
a5*cosd(t5); sind(t5),cosd(ap5)*cosd(t5),-sind(ap5)*cosd(t5),
a5*sind(t5); 0, sind(ap5),cosd(ap5),d5;0,0,0,1]

A45 =

     1     0     0     0
     0     0     1     0
     0    -1     0     0
     0     0     0     1
```
… (12)

```
>>%Parámetros D-H del Eslabón 6

>> t6=180;
>> d6=0.065;
>> a6=0;
>> ap6=0;

>>%Matriz de transformación homogénea 5A6

>>      A56=[cosd(t6),-cosd(ap6)*sind(t6),sind(ap6)*sind(t6),
a6*cosd(t6); sind(t6),cosd(ap6)*cosd(t6),-sind(ap6)*cosd(t6),
a6*sind(t6); 0, sind(ap6),cosd(ap6),d6;0,0,0,1]

A56 =

   -1.0000         0         0         0
         0   -1.0000         0         0
         0         0    1.0000    0.0650
         0         0         0    1.0000
```
… (13)

Se obtuvo la matriz de transformación homogénea que relaciona el sistema de la base con el extremo del robot, multiplicando todas las transformaciones obtenidas de manera ascendente y en orden.

```
>> A06=A01*A12*A23*A34*A45*A56

>>%Matriz de transformación homogénea 0A6

          A06 =

                0         0    1.0000    0.5150
                0    1.0000         0         0
           -1.0000         0         0    0.7120
                0         0         0    1.0000
```

... (14)

Método 2

Con este método se obtuvo la misma matriz de Transformación homogénea, pero se utilizó la librería de ***Robotics Toolboox de Peter Corke*** con el fin de simplificar líneas de código.

Se definió cada eslabón de la **Tabla 1.11** con el comando "***Link***". Si la junta es rotativa se coloca 0 en el último parámetro, si es junta prismática se coloca un 1 y además los grados deben estar en radianes.

Para definir un eslabón con el método "***Link***" se debe escribir siguiendo la siguiente estructura.

$$L(i)=\text{Link}\ (\theta i,\ di, ai, \alpha i, \text{tipo de junta})$$

En la ventana de comandos en Matlab se escribió los 6 eslabones que definen al robot IRB-140 con base a la **Tabla 1.12**.

```
>> L(1)=Link([0,0.352,0.070,-pi/2,0]);
>> L(2)=Link([0,0,0.360,0,0]);
>> L(3)=Link([0,0,0,-pi/2,0]);
>> L(4)=Link([0,0.380,0,pi/2,0]);
>> L(5)=Link([0,0,0,-pi/2,0]);
>> L(6)=Link([0,0.065,0,0,0]);
```

Donde:

pi/2: Son 90° en radianes.

L: Es un array donde se almacenan todos los eslabones $L = [L1, L2, L3, ..., L6]$.

Se creo una cadena cinemática de todos los eslabones anteriores usando el comando de la librería "***SerialLink***", esta cadena se almacenó en una variable denominada "robot".

```
>> %Cadena cinemática.
>> robot=SerialLink(L);
```

Se definió una variable **q** donde se almacenan los valores iniciales de $\theta_1, \theta_2, \theta_3, \ldots, \theta_6$ con base en la **Tabla 1.12.**

```
>> %q= [q1 q2 q3 q4 q5 q6]
>> q= [0 -pi/2 0 0 0 pi];
```

Se utilizó el comando "***fkine***" para obtener la matriz de transformación homogénea que relaciona el sistema 6 con el sistema 0.

```
>>% Matriz de transformación Homogénea
>> robot.fkine(q)

          ans =
                    0         0         1    0.515
                    0         1         0        0
                   -1         0         0    0.712
                    0         0         0        1
```

... (15)

El resultado fue la misma matriz de trasformación homogénea **(14)** que se obtuvo con el primer método. La ventaja de usar esta librería además de usar menos líneas de código es que también permite obtener un plóter o gráfico de la cadena cinemática. Permitiendo observar si la cadena es correcta o alguno de los parámetros ingresados fue erróneo. El comando que permite realizar el gráfico es "***plot***". En el "**ploter**" las articulaciones se representan de color rojo y los eslabones de color azul.

```
>> %Plóter de la cadena cinemática.
>> robot.plot(q)
```

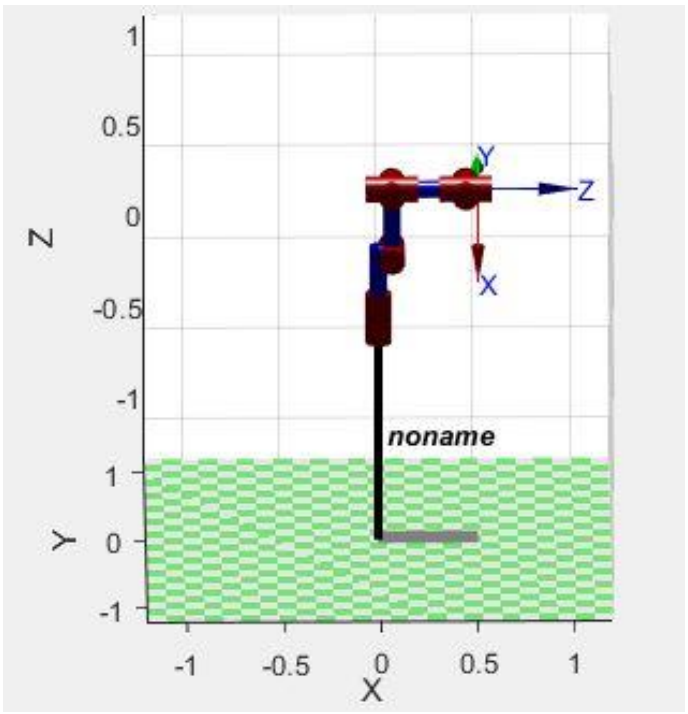


Ilustración 40. *Gráfica de la cadena cinemática que representa la estructura robot IRB-140*

2.4.2 CINEMÁTICA INVERSA DE LOS MANIPULADORES

Como ya se ha mencionado en capítulos anteriores la cinemática inversa trata de encontrar el valor de las variables articulares $\mathbf{q = [q1\ q2\ q3 ... qn]}$. Correspondientes a una posición y orientación del efector final.

En comparación de la cinemática directa, la cinemática inversa es mucho más compleja de calcular por las siguientes razones:

- Las ecuaciones en las variables articulares por lo general no son lineales por ello no es posible encontrar una solución explicita.
- Las ecuaciones dependen de la configuración del robot.
- Puede haber múltiples soluciones.
- Puede haber soluciones infinitas en el caso de manipuladores redundantes.

- En algunas ocasiones no hay soluciones adecuadas debido a la configuración del robot. [48] [10]

Pero la mayoría de estos manipuladores poseen cinemáticas simples que facilitan la resolución del problema cinemático inverso. Tal es el caso del robot IRB-140 de la marca ABB, que cuenta con 6 articulaciones, de las cuales, las 3 primeras funcionan para posicionar al robot y las últimas 3 articulaciones sirven para orientar el efector final o la herramienta de trabajo del robot. [10]

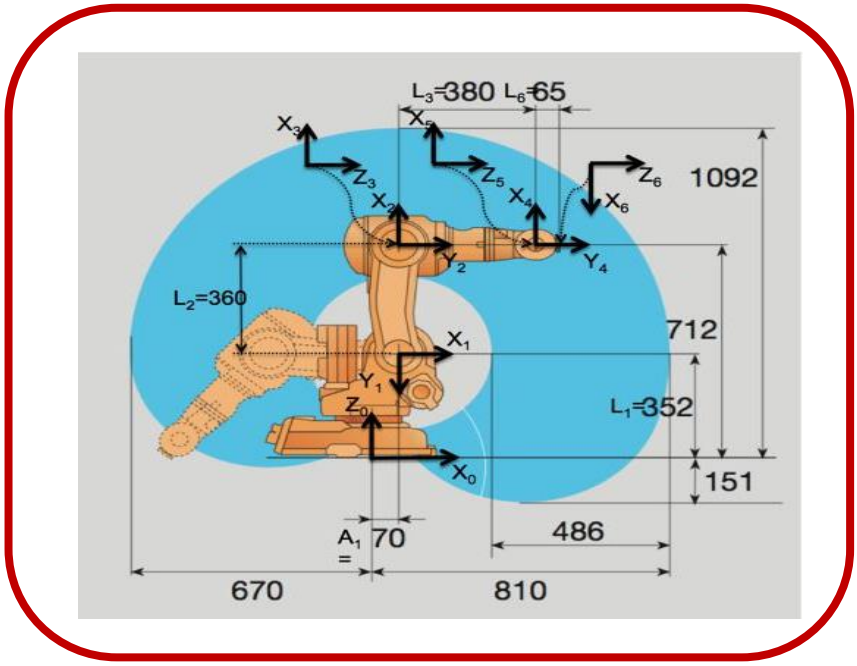


Ilustración 41. *Representación de los sistemas de referencia y las dimensiones del robot. Recuperado en:*
Payá, C. L & otros. Uso MATLAB en robótica y visión por computador pág. 52.

Se asumió que la orientación y la posición del efector final (**Sistema 6**) está dada por una matriz de transformación Homogénea (**T**):

$$T = \begin{bmatrix} \overrightarrow{x_6} & \overrightarrow{y_6} & \overrightarrow{z_6} & \vec{p} \\ 0 & 0 & 0 & 1 \end{bmatrix} \dots (16)$$

Donde:

$\overrightarrow{x_6}, \overrightarrow{y_6}, \overrightarrow{z_6}$: Definen la orientación del Sistema **S6.**

$\vec{p}$: Define la posición del Sistema **S6.**

Escrita de otra manera la matriz **(16).**

$$\mathrm{T} = \begin{bmatrix} \vec{R}_6^0 & & & \overrightarrow{P_6^0} \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (17)$$

Donde:

$\overrightarrow{\boldsymbol{R}_6^0}$: Submatriz de Rotación del efector final (Orientación), con respecto al sistema de coordenadas de la base.

$\overrightarrow{\boldsymbol{P}_6^0}$**:** Posición del efector final (Traslación), con respecto al sistema de coordenadas de la base.

Para poder realizar el desacoplo cinemático al manipulador, es necesario analizarlo desde la posición de la muñeca del robot pues este punto solamente depende del valor de las primeras tres articulaciones **q1, q2, q3**. No importa el valor de las otras variables **q4, q5, q6** la posición en la muñeca siempre será la misma. La muñeca de los robots o también conocida como muñeca de Euler se caracteriza porque los últimos ejes de acción **Z3, Z4, Z5** de las articulaciones **q4, q5, q6** interceptan en un mismo punto (Véase la **Ilustración 42**). [48] [30]

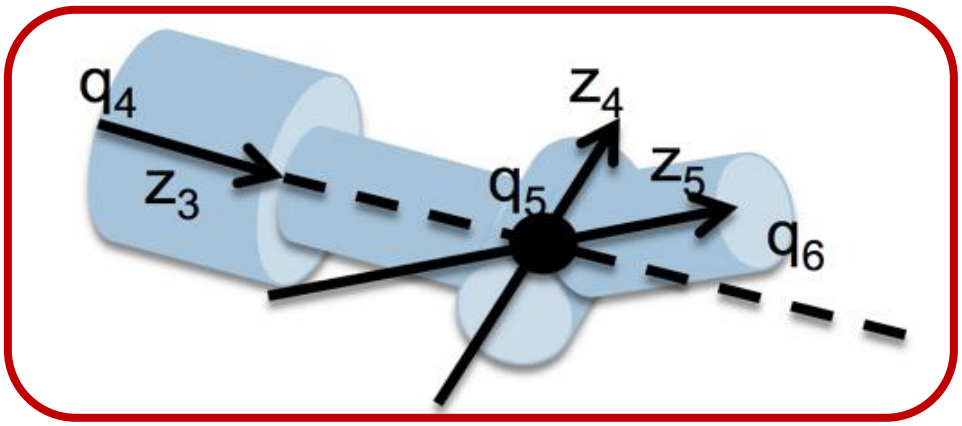


Ilustración 42. *Intersección de los ejes de acción de las articulaciones q1, q2, q3. Recuperado en:*
Payá, C. L & otros. Uso MATLAB en robótica y visión por computador pág. 59.

Una vez conocida la posición ($\vec{p}$) del robot, se calcula la posición de la muñeca ($\overrightarrow{p_m}$) para después obtener la solución de las primeras tres articulaciones.

Se calculó vectorialmente la relación que hay entre el punto de la muñeca con el punto del efector final.

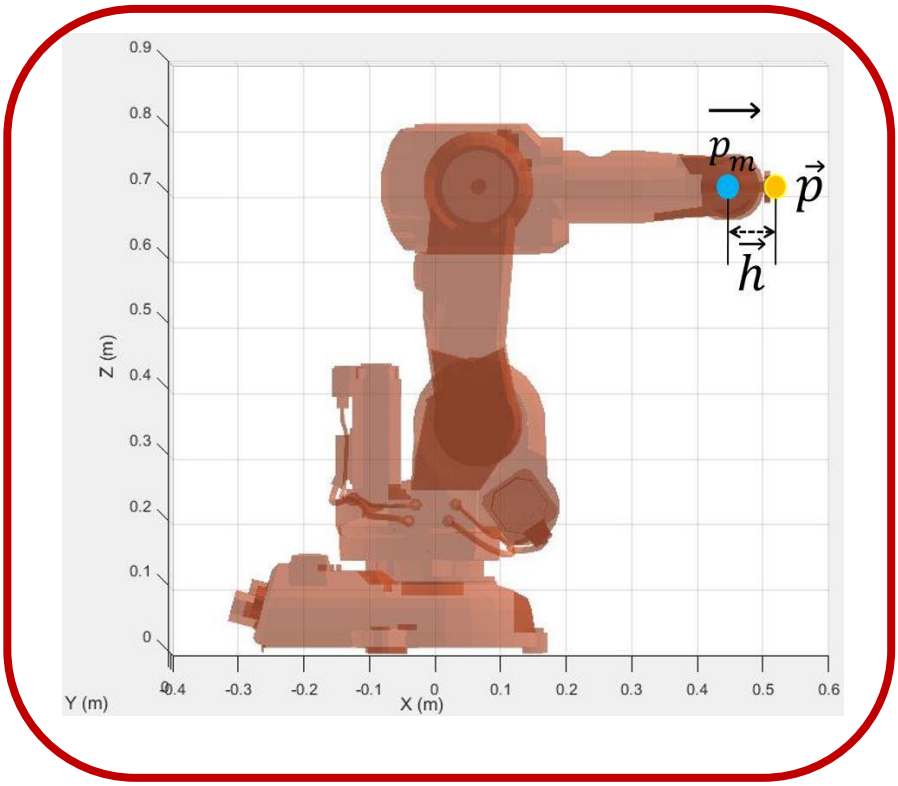


Ilustración 43. *Relación entre la posición de la muñeca y del efector final.*

La posición de la muñeca ($\overrightarrow{p_m}$) más un vector $\mathbf{h}$ ($\vec{h}$) da como resultado la posición del efector final ($\vec{p}$).

$$\vec{p} = \overrightarrow{p_m} + \overrightarrow{h_0} \ldots (18)$$

Donde:

$\vec{p}$: La posición del efector final.

$\overrightarrow{p_m}$: Posición de la muñeca

$\overrightarrow{\boldsymbol{h_0}}$**:** Es un vector que va desde el origen del **sistema 5** (punto de la muñeca) al **sistema 6** (punto del efector final) expresados en el **sistema 0** (la base).

Si $\vec{h}$ en vez de ser expresado en el **sistema 0** ($\overrightarrow{h_0}$) se expresó en el sistema 6 ($\overrightarrow{h_6}$) como se observa en la **Ilustración 43**. Entonces $\overrightarrow{h_6}$ es la distancia L_6 proyectada a lo largo del eje Z_6 . Por lo tanto:

$$\overrightarrow{h_6} = L_6 * \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \ldots (19)$$

Para expresar a $\vec{h}$ en el **sistema 0** ($\overrightarrow{h_0}$) se multiplicó la submatriz de orientación del efector final por el vector $\vec{h}$ extresado en el sistema 6 ($\overrightarrow{h_6}$)

$$\overrightarrow{h_0} = R_6^0 * L_6 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \ldots (20)$$

La matriz de transformación homogénea se conforma de un cambio de orientación con nueve componentes (R_{11} … R_{33}) conocida como la submatriz de orientación y una posición de tres componentes (P_X , P_y , P_Z).

$$\mathrm{T} = \begin{bmatrix} R_{11} & R_{12} & R_{13} & P_x \\ R_{21} & R_{22} & R_{23} & P_Y \\ R_{31} & R_{32} & R_{33} & P_Z \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (21)$$

Por lo tanto:

$$\overrightarrow{h_0} = R_6^0 * L_6 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = L_6 \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = L_6 \begin{pmatrix} R_{13} \\ R_{23} \\ R_{33} \end{pmatrix} \ldots (22)$$

Donde:

$\begin{pmatrix} \boldsymbol{R_{13}} \\ \boldsymbol{R_{23}} \\ \boldsymbol{R_{33}} \end{pmatrix}$**:** Es una proyección del eje $\overrightarrow{z_6}$ en el **sistema 0.**

Sustituyendo $\overrightarrow{h_0}$ en ecuación (18) y despejando $\overrightarrow{p_m}$:

$$\overrightarrow{p_m} = \vec{p} - L_6 \overrightarrow{z_6} \ldots (23)$$

Donde:

$\overrightarrow{p_m}$: Es la posición de la muñeca con respecto al sistema 0.

$\vec{p}$: Es la posición del efector final $\vec{p} = [P_x\ P_y\ P_z]^T$

L_6: Longitud entre el sistema de referencia S5 con el sistema S6

$\overrightarrow{z_6}$: Proyección del eje $\overrightarrow{z_6}$ en el sistema 0, $\overrightarrow{z_6} = [R_{13}\ R_{23}\ R_{33}]^T$

Una vez obtenida la posición de la muñeca con respecto al sistema 0 se obtuvo las primeras tres ecuaciones para las articulaciones (**q1, q2, q3**).

Solución para la articulación q1

Existen varios métodos para resolver la cinemática inversa de un manipulador serial. "Los métodos geométricos permiten obtener normalmente los valores de las primeras variables articulares, que son las que consiguen posicionar el robot. Para ello utilizan relaciones trigonométricas y geométricas sobre los elementos del robot. Se suele recurrir a la resolución de triángulos formados por los elementos y articulaciones del robot… Como alternativa para resolver el mismo problema se puede recurrir a manipular directamente las ecuaciones correspondientes al problema cinemático directo". [10]

Se utilizó Matlab y la librería ARTE Toolbox y para tener mejor visualización del manipulador. Se ocupó el siguiente código:

```
>> init_lib
>> robot=load_robot('ABB','IRB140')
>> robot.graphical.draw_transparent=1
>>   drawrobot3d   (robot,[0.523599   0   0   0   0   0   ])% q1=30°=0.523599rad (para mover el robot en su primera articulación 30°)
```

Con ayuda de Power Point se dibujó el triángulo rectángulo para calcular q1.

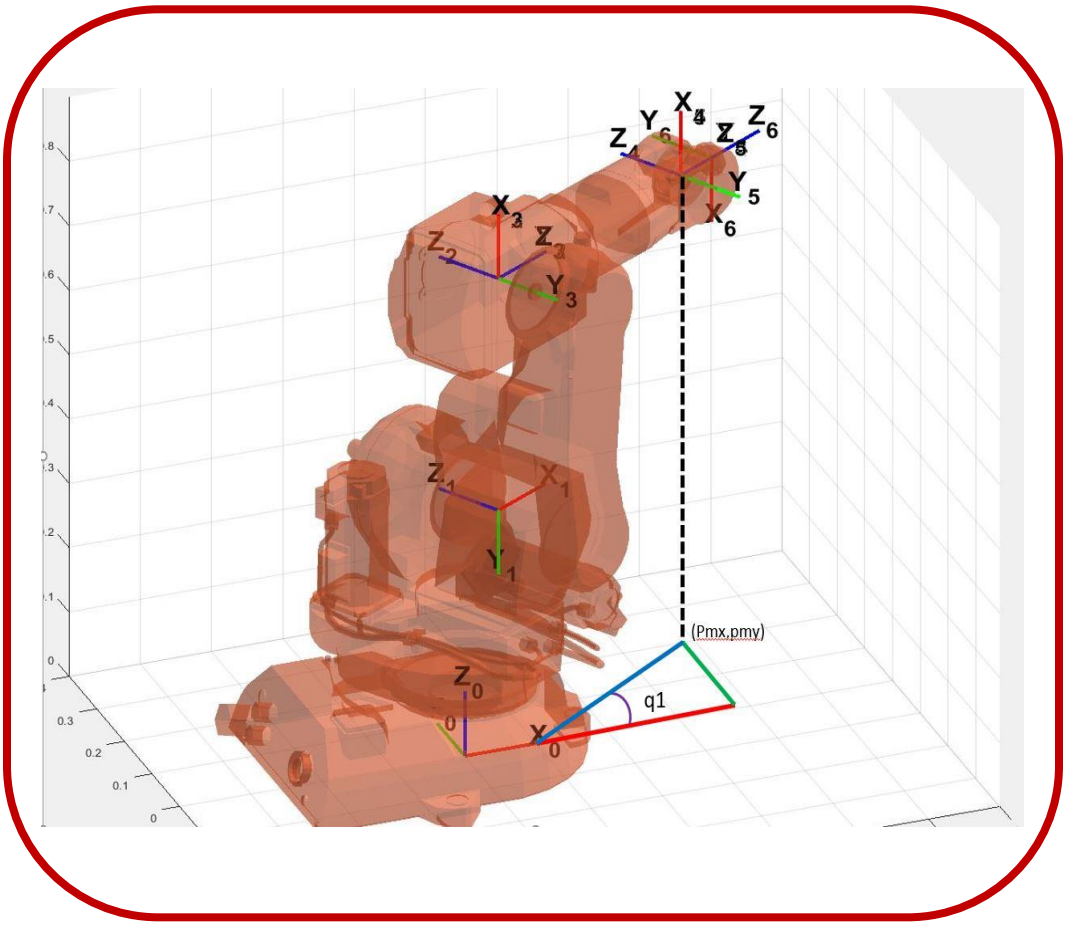


Ilustración 44. *Robot IRB-140 con la librería ARTE Robotics Matlab con un giro en la articulación 1.*

La muñeca del robot se proyecta hacia un punto del plano (X0, Y0) el cual se denominó como (Pmx, Pmy), con este punto se puede formar un triángulo rectángulo (Véase **Ilustración 44**). Usando la función trigonométrica **arctan** (arco tangente) para resolver **q1** se obtuvo:

$$q1 = arctan\left(\frac{Pmy}{Pmx}\right) \ldots (24)$$

"Existen más de una posible solución para q1…Dada la naturaleza periódica de las funciones trigonométricas". [63] Por lo tanto, hay otra posible solución:

$$q1^{*} = q1 + K\pi \ldots (25)$$

Para encontrar la solución de las articulaciones **q1** y **q2** se ingresó el siguiente código en la ventana comandos de Matlab y se obtuvo otra vista del manipulador (**Ilustración 45**).

```
>> init_lib
>> robot=load_robot('ABB','IRB140')
>> robot.graphical.draw_transparent=1
>> drawrobot3d(robot,[0 0.523599 0 0 0 0 ])% q2=30°=0.523599rad
```

Con ayuda de la **Ilustración 45** se creó dos representaciones alámbricas (Véase **Ilustración 46 e Ilustración 47**).

Solución de la articulación q2.

Se tomó en cuenta que la posición de la muñeca ($\overrightarrow{p_m}$) ahora está en relación con el **sistema 1** (**x1, y1, z1**) por lo tanto $\overrightarrow{p_m}^1$ se define como:

$$\overrightarrow{p_m}^1 = (A_1^0)^{-1} * \overrightarrow{p_m} \ldots (26)$$

Donde:

$\overrightarrow{\boldsymbol{p_m}}^{\mathbf{1}}$: Vector de la muñeca del robot referenciado en el sistema 1

$\left(\boldsymbol{A_1^0}\right)^{-\mathbf{1}}$: Matriz Inversa entre del sistema 1 y el sistema 0.

$\overrightarrow{\boldsymbol{p_m}}$: Vector de la muñeca del robot en el Sistema 0.

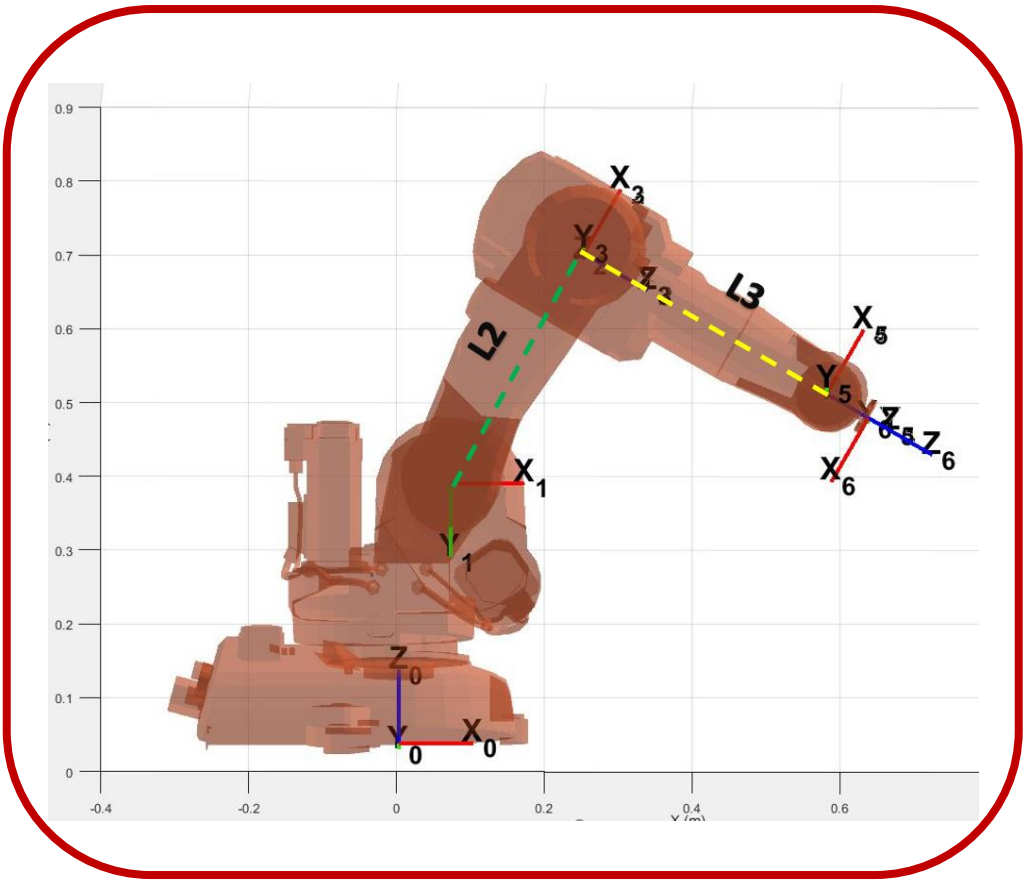


Ilustración 45. *Robot IRB-140 con la librería ARTE Robotics Matlab con un giro en la articulación 2.*

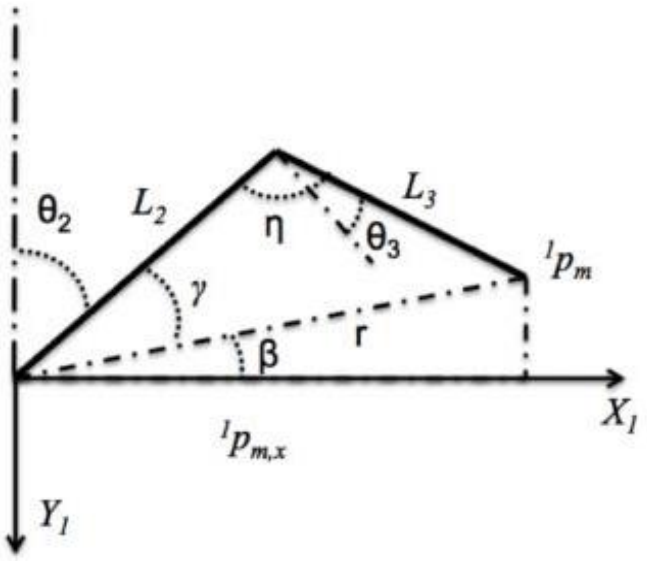


Ilustración 46. *Representación alámbrica codo arriba del robot IRB-140.*

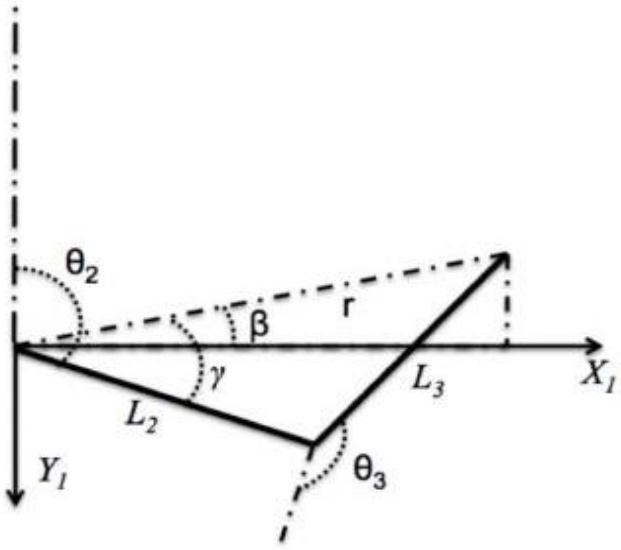


Ilustración 47. *Representación alámbrica codo abajo del robot IRB-140.*

En las **Ilustraciones 46 y 47** se observó que la articulación 2 (**q2=$\boldsymbol{\theta_2}$**) inicialmente tiene 2 posibles soluciones también conocidas como codo arriba y codo abajo, por lo tanto:

$$q2 = 90^0 - \gamma - \beta \ldots (27)$$

$$q2^* = 90^0 + \gamma - \beta \ldots (28)$$

Donde:

$q2$: Solución de la articulación 2 para la configuración codo arriba.

$q2^*$: Solución para la articulación 2 para la configuración codo abajo.

Se utilizó el teorema de Pitágoras para obtener **r**, así que **r** quedo definida como:

$$r = \sqrt{(\overrightarrow{p_{mx}}^1)^2 + (\overrightarrow{p_{my}}^1)^2} \ldots (29)$$

Para obtener en ángulo $\boldsymbol{\beta}$ se utilizó la identidad trigonométrica de la tangente:

$$\beta = arctan\left(\frac{\overrightarrow{p_{my}}^1}{\overrightarrow{p_{mx}}^1}\right) \ldots (30)$$

Se aplicó el teorema de cosenos para obtener el ángulo $\boldsymbol{\gamma}$:

$$L_3^2 = L_2^2 + r^2 - 2rL_2\cos(\gamma) \ldots (31)$$

Se despejó $\boldsymbol{\gamma}$ de ecuación (31):

$$\gamma = arccos\left(\frac{L_2^2 + r^2 - L_3^2}{2rL_2}\right) \ldots (32)$$

Conocidos los ángulos $\boldsymbol{\beta}$ y $\boldsymbol{\gamma}$ se sustituyeron en las ecuaciones ecuación (27) y (28) para encontrar las dos posibles soluciones de la articulación 2:

$$q2 = 90^0 - arccos\left(\frac{L_2^2 + r^2 - L_3^2}{2rL_2}\right) - arctan\left(\frac{Pmy}{Pmx}\right) \ldots (33)$$

$$q2^{*} = 90^{0} - arccos\left(\frac{L_2^2 + r^2 - L_3^2}{2rL_2}\right) - arctan\left(\frac{Pmy}{Pmx}\right) \ldots (34)$$

Solución de la articulación q3

Al igual que la articulación 2, la articulación 3 tiene dos posibles soluciones por la configuración codo arriba, codo abajo.

Entonces:

$$q3 = 90^{0} - \eta \ldots (35)$$

$$q3^{*} = \eta - 270° \ldots (36)$$

Donde:

$q3$: Solución de la articulación 3 para la configuración codo arriba.

$q3^{*}$: Solución para la articulación 3 para la configuración codo abajo.

Se obtiene $\boldsymbol{\eta}$ utilizando la ley de cosenos:

$$r^2 = L_2^2 + L_3^2 - 2L_2L_3\cos(\eta) \ldots (37)$$

Se despejó $\boldsymbol{\eta}$ de la ecuación (37):

$$\eta = arccos\left(\frac{L_2^2 + L_3^2 - r^2}{2L_2L_3}\right) \ldots (38)$$

Conocido el ángulo $\boldsymbol{\eta}$ se sustituyó en las ecuaciones (34) y (35)

$$q3 = 90^{0} - \ arccos\left(\frac{L_2^2 + L_3^2 - r^2}{2L_2L_3}\right) \ldots (39)$$

$$q3^{*} = arccos\left(\frac{L_2^2 + L_3^2 - r^2}{2L_2L_3}\right) - 270° \ldots (40)$$

Solución de las articulaciones q4, q5, q6

Para la solución de las articulaciones 4, 5 y 6 se tuvo en cuenta dos factores:

- La matriz de transformación homogénea **(T)** que define al robot, es conocida.
- Las articulaciones q1, q2, q3 también ya son conocidas.

Una vez conocido estos dos factores se sabe que:

$$T = A_1^0 * A_2^1 * A_3^2 * A_4^3 * A_5^4 * A_6^5 \ldots (41)$$

La submatriz $\boldsymbol{A_1^0}$ depende del valor de la articulación **q1,** así como $\boldsymbol{A_2^1}$ de **q2** y $\boldsymbol{A_3^2}$ de **q3.** Y como estás tres articulaciones ya fueron calculadas anteriormente, se conocen $\boldsymbol{A_1^0}$, $\boldsymbol{A_2^1}$, $\boldsymbol{A_3^2}$, por lo tanto:

$$(A_3^2)^{-1} * (A_3^2)^{-1} * (A_3^2)^{-1} * T = A_4^3 * A_5^4 * A_6^5 = Q \ldots (42)$$

Con los parámetros de Denavit-Hartenberg de la **Tabla 1.10** se obtuvo las submatrices $\boldsymbol{A_4^3}$, $\boldsymbol{A_5^4}$, $\boldsymbol{A_6^5}$.

Entonces:

$$A_4^3 = \begin{bmatrix} Cos(q4) & 0 & Sin(q4) & 0 \\ Sin(q4) & 0 & -\mathrm{Cos}(q4) & 0 \\ 0 & 1 & 0 & L_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (43)$$

$$A_5^4 = \begin{bmatrix} Cos(q5) & 0 & -Sin(q5) & 0 \\ Sin(q5) & 0 & \mathrm{Cos}(q5) & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (44)$$

$$A_6^5 = \begin{bmatrix} -Cos(q6) & Sin(q6) & 0 & 0 \\ -Sin(q6) & -\mathrm{Cos}(q6) & 0 & 0 \\ 0 & 0 & 1 & L_6 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (45)$$

Se multiplicó $\boldsymbol{A_4^3},\ \boldsymbol{A_5^4},\ \boldsymbol{A_6^5}$:

$$Q = \begin{bmatrix} -C(q4)c(q5)c(c6) + s(q4)s(q6) & c(q4)c(q5)s(q6) + s(q4)c(q6) & -c(q4)s(q5) & -L_6c(q4)s(q5) \\ -s(q4)c(q5)c(c6) - c(q4)s(q6) & s(q4)c(q5)s(q6) - c(q4)c(q6) & -s(q4)s(q5) & -L_6s(q4)s(q5) \\ -s(q5)c(q6) & s(q5)s(q6) & C(q5) & L_6c(q5) + L_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (46)$$

Este método exige elegir con cuidado las ecuaciones de la matriz **Q** para extraer las soluciones **q4, q5, q6,** buscando siempre la tangente de cada uno de las articulaciones **(tan(q4), tan(q5), tan(q6)).**

La ecuación $Q_{33} = \cos(q5)$ dice si la solución es redundante o no, esto es, si q5 es mayor a 0 entonces las soluciones de q4 y q6 se pueden solucionar de la siguiente manera.

$$\text{Si } Q_{33}>0 \text{ entonces } q5 \neq 0 \ldots (47)$$

Se sabe que la identidad trigonométrica para obtener valor de un ángulo con la tangente es:

$$\tan(\theta i) = \frac{\sin(\theta i)}{\cos(\theta i)} \ldots (48)$$

Se obtuvo **q4** si **q5≠0,** se tomaron Q_{23} y Q_{13}:

$$\frac{Q_{23}}{Q_{13}} = \frac{-\sin(q_4)\sin(q_5)}{-\cos(q_4)\sin(q_5)} = \frac{-\sin(q_4)}{-\cos(q_4)} = \tan(q4) \ldots (49)$$

Se despejo **q4** de ecuación (49):

$$q_4 = arctan\left(\frac{Q_{23}}{Q_{13}}\right) \ldots (50)$$

Solución de **q6** si **q5≠0**:

$$\frac{-Q_{32}}{Q_{31}} = \frac{-\sin(q_5)\sin(q_6)}{-\sin(q_5)\cos(q_6)} = \frac{\sin(q_6)}{\cos(q_6)} = \tan(q6) \ldots (51)$$

Se despejó **q6** de ecuación (51):

$$q_6 = arctan\left(\frac{-Q_{32}}{Q_{13}}\right) \ldots (52)$$

Para hallar la tangente de la articulación 5 se requiere del seno y del coseno de **q5**. Y como en este caso q5 es mayor que cero, se tomó como coseno (**q5**) el valor de la matriz Q_{33} por lo tanto:

$$Q_{33}=\cos(q5) \ldots (53)$$

Como ya se conoce **q6** se tomó la ecuación Q_{32}, para obtener el **seno (q6)**, por lo tanto:

$$Q_{32}= (sen(q5))((sen(q6)) \ldots (54)$$

Se despejó la ecuación (54) y se obtuvo:

$$\left(\frac{Q_{32}}{(sen(q6)}\right) = (sen(q5)) \ldots\ldots (55)$$

Conocido **cos(q5)** y **sin(q5)** se obtuvo **tan(q5):**

$$\frac{\sin(q5)}{\cos(q_5)} = \tan(q5) \ldots (56)$$

Los problemas surgen cuando **q5≈ 0** porque las ecuaciones (50) y (52) dan como resultado un valor indeterminado por lo tanto no es posible calcular el valor de las articulaciones **q4** y **q6** con las ecuaciones utilizadas anteriormente.

Por lo tanto, para encontrar el valor de las articulaciones **q4**, **q6,** se tuvo que manipular otras ecuaciones ($\boldsymbol{Q_{12}, Q_{21}, Q_{11}, Q_{22}}$) para obtener el valor de **q4** y **q6**

$$\frac{Q_{12}-Q_{21}}{-Q_{11}-Q_{21}} = \frac{[c(q4)c(q5)s(q6)+s(q4)c(q6)]-[-s(q4)c(q5)c(c6)-c(q4)s(q6)]}{-[-C(q4)c(q5)c(c6)+s(q4)s(q6)]-[s(q4)c(q5)s(q6)-c(q4)c(q6)]} \ldots (57)$$

Se factorizó el denominador y el numerador:

$$\frac{Q_{12}-Q_{21}}{-Q_{11}-Q_{21}} = \frac{(1+c(q5))(c(q4)s(q6)+s(q4)c(q6))}{(1+c(q5))(c(q4)c(q6)-s(q4)s(q6))} \ldots (58)$$

Se sabe que:

$$\sin(x+y) = \sin(x)\cos(y) + \cos(x)\sin(y) \ldots (59)$$

$$\cos(x+y) = \cos(x)\cos(y) - \sin(x)\sin(y) \ldots (60)$$

Simplificando ecuación (58) y utilizando las identidades trigonométricas (59) y (60) se obtuvo:

$$\frac{Q_{12}-Q_{21}}{-Q_{11}-Q_{21}} = \frac{\sin(q6+q4)}{\cos(q6+q4)} \ldots (61)$$

Por lo tanto, puede haber infinitas soluciones para **q6** y **q4** que permiten orientar al extremo del robot si $\mathbf{q5} \approx \mathbf{0}$.

Si $\mathbf{q4=0}$ entonces:

$$(q6) = arctan\left(\frac{Q_{12}-Q_{21}}{-Q_{11}-Q_{21}}\right) \ldots (64)$$

La herramienta de Matlab, con la librería ARTE tiene programada la cinemática inversa de todos los robots manipuladores con los que cuenta. Basta con tan solo unas líneas de código para resolver la cinemática inversa y regrese 8 posibles soluciones para que el efector final se posicione y se oriente en un punto determinado. Esta librería ya tiene programadas todos los cálculos matemáticos para resolver **q1, q2, q3, q4, q5, q6** como se obtuvieron anteriormente, además de tener las condiciones para casos degenerados (cuando $\boldsymbol{Q_{33}} \approx \mathbf{0}$).

Cinemática inversa del robot IRB-140 con ARTE Robotic Toolbox.

Se ingreso el siguiente código para graficar el robot IRB-140:

```
>> init_lib
>> robot=load_robot('ABB','IRB140')
```

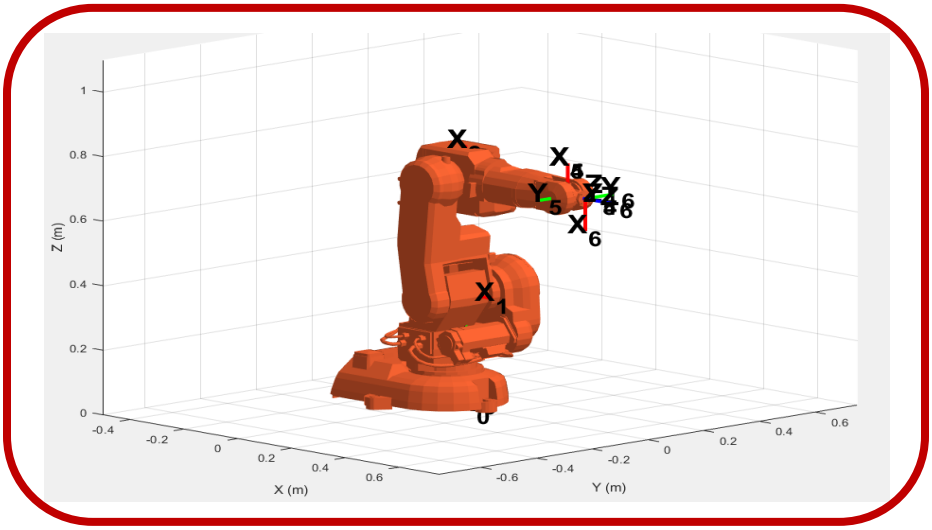


Ilustración 48. *Robot IRB-140 con la librería ARTE Robotics Matlab con en posición "HOME".*

El efector final del robot se encuentra en la posición ($x=0.515, y=0, z=0.712$), con la cinemática inversa se pueden obtener distintas soluciones para llegar al mismo punto, pero con diferente valor en sus articulaciones ($q1, q2, q3, q4, q5, q6$).

Después de obtener el "**plotter**" o gráfico del robot se inserta el siguiente código para poder controlar el manipulador con el "**teach pendant**" o controlador.

```
>> teach
```

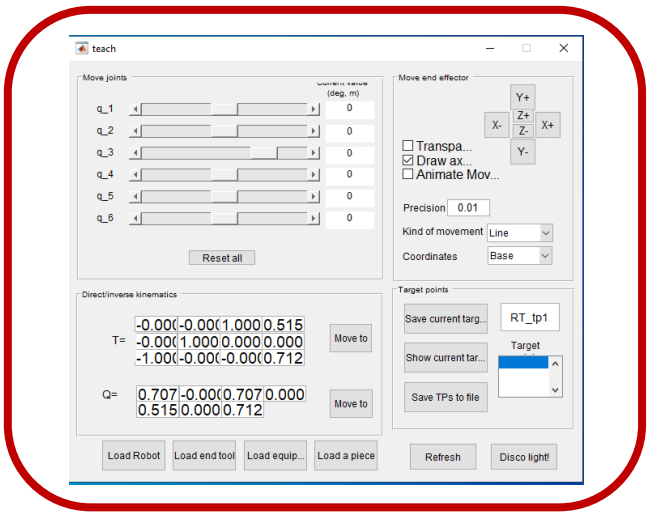


Ilustración 49. *"Teach Pendant" con la librería ARTE Robotics Matlab.*

Se extrajo el valor de las articulaciones del robot en la posición en la que se muestra en la **Ilustración 48**.

```
>> robot.q

ans =

     0 %Valor de la articulación 1
     0 %Valor de la articulación 2
     0 %Valor de la articulación 3
     0 %Valor de la articulación 4
     0 %Valor de la articulación 5
     0 %Valor de la articulación 6
```

… (65)

Se obtuvo la matriz de transformación homogénea que describe al robot en esa posición, con el siguiente código.

```
>> T=directkinematic(robot,robot.q)

T =

   -0.0000   -0.0000    1.0000    0.5150
   -0.0000    1.0000    0.0000    0.0000
   -1.0000   -0.0000   -0.0000    0.7120
         0         0         0    1.0000
```

… (66)

Con la función "***inversekinematic***" y la matriz de transformación homogénea, el programa realiza todos los cálculos necesarios para lanzar 8 posibles soluciones (El código está limitado a solo 8 soluciones, pero puede haber más) para que el manipulador llegue al mismo punto, pero con diferente valor en sus articulaciones.

```
>> soluciones=inversekinematic(robot,T)

soluciones =

    0.0000    0.0000    0.0000    0.0000    3.1416    3.1416    3.1416    3.1416
    0.0000    0.0000    1.6248    1.6248   -1.5278   -1.5278   -0.4027   -0.4027
   -0.0000   -0.0000   -3.1416   -3.1416   -0.4786   -0.4786   -2.6630   -2.6630
         0   -3.1416   -3.1416         0    0.0000   -3.1416    0.0000   -3.1416
         0         0   -1.5168    1.5168   -1.1352    1.1352   -0.0759    0.0759
   -0.0000    3.1416    3.1416   -0.0000    3.1416   -0.0000    3.1416   -0.0000
```

… (67)

Cada columna representa una solución. Para demostrar si la solución es válida, es necesario obtener la matriz de transformación homogénea con cada solución. Si la matriz (**T1, T2, ..., T8**) es igual a la matriz **T** (66) entonces la solución es válida. [48]

Se obtuvo la matriz de transformación homogénea y además el gráfico con la configuración que toma el robot para cada solución.

Se obtuvo la matriz de transformación homogénea **T1** (se obtuvieron números negativos en algunos términos de la matriz por cuestiones del redondeo) y se comparó con la matriz **T** (66). Se observó que da como resultado la misma matriz (**T1=T2**) por lo tanto la primera solución es válida.

```
>> T1=directkinematic(robot,soluciones(:,1))
>> drawrobot3d(robot,soluciones(:,1))
```

```
T1 =

   -0.0000   -0.0000    1.0000    0.5150
   -0.0000    1.0000    0.0000    0.0000
   -1.0000   -0.0000   -0.0000    0.7120
         0         0         0    1.0000
```

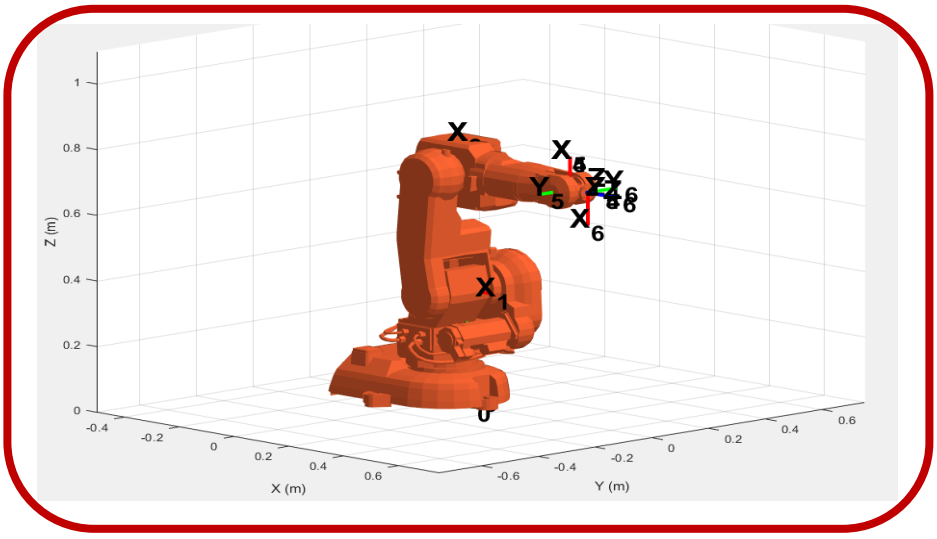


Ilustración 50. *Robot IRB-140 tomando el valor en sus articulaciones de la solución 1*

Se realizó el mismo procedimiento, pero con las otras siete soluciones.

Solución 2:

```
    0.0000 %Valor de la articulación 1 en radianes, en grados=0°
    0.0000 %Valor de la articulación 2 en radianes, en grados=0°
   -0.0000 %Valor de la articulación 3 en radianes, en grados=0°
   -3.1416 %Valor de la articulación 4 en radianes, en grados=-180°
         0 %Valor de la articulación 5 en radianes, en grados=0°
    3.1416 %Valor de la articulación 6 en radianes, en grados=180°
```

```
>> T2=directkinematic(robot,soluciones(:,2))
>> drawrobot3d(robot,soluciones(:,2))
```

```
T2 =

   -0.0000   -0.0000    1.0000    0.5150
   -0.0000    1.0000    0.0000    0.0000
   -1.0000   -0.0000   -0.0000    0.7120
         0         0         0    1.0000
```

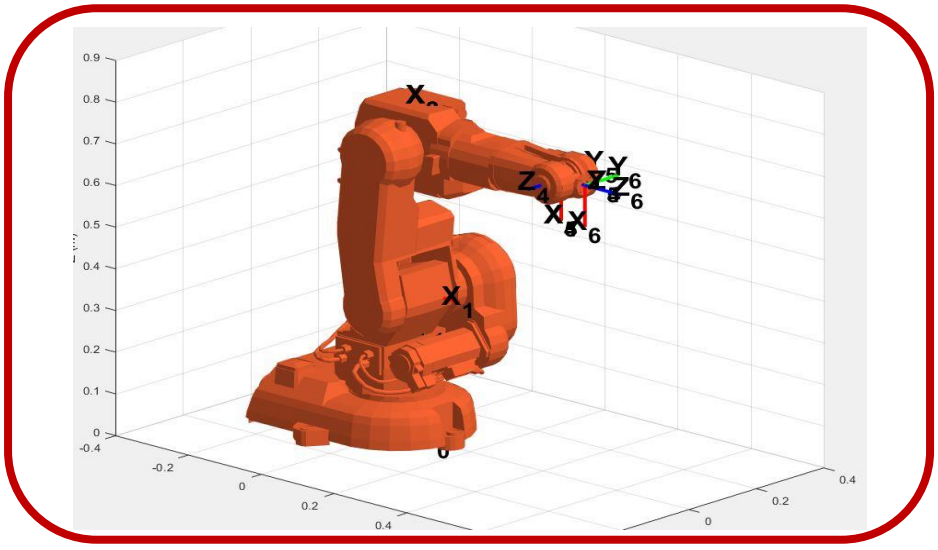


Ilustración 51. *Robot IRB-140 tomando el valor en sus articulaciones de la solución 2*

Solución 3:

```
 0.0000 %Valor de la articulación 1 en radianes, en grados=0°
 1.6248 %Valor de la articulación 2 en radianes, en grados=93°
-3.1416 %Valor de la articulación 3 en radianes, en grados=-180°
-3.1416 %Valor de la articulación 4 en radianes, en grados=-180°
-1.5168 %Valor de la articulación 5 en radianes, en grados=87°
 3.1416 %Valor de la articulación 6 en radianes, en grados=180°
```

```
>> T3=directkinematic(robot,soluciones(:,3))
>> drawrobot3d(robot,soluciones(:,3))
```

```
T3 =

   -0.0000   -0.0000    1.0000    0.5150
   -0.0000    1.0000    0.0000    0.0000
   -1.0000   -0.0000   -0.0000    0.7120
         0         0         0    1.0000
```

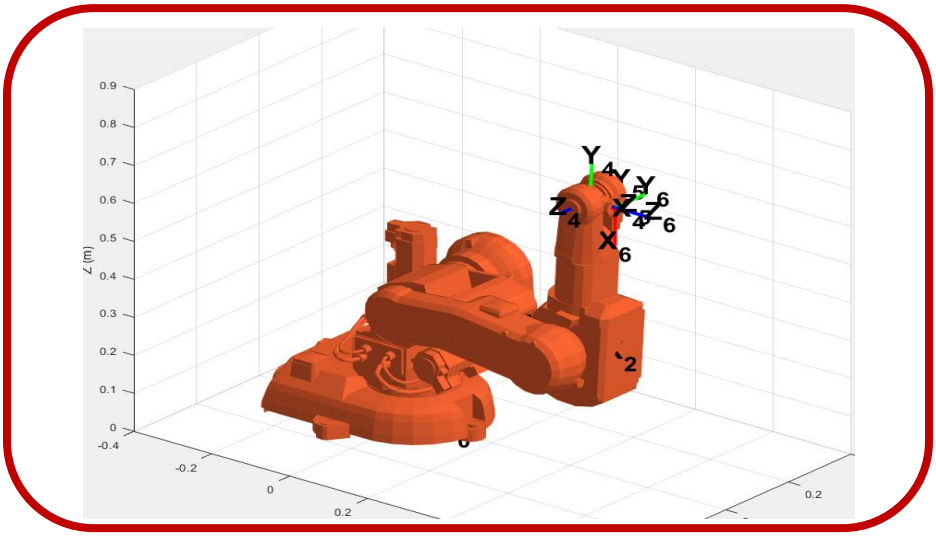


Ilustración 52. *Robot IRB-140 tomando el valor en sus articulaciones de la solución 3*

Solución 4:

```
 0.0000 %Valor de la articulación 1 en radianes, en grados=180°
 1.6248 %Valor de la articulación 2 en radianes, en grados=93°
-3.1416 %Valor de la articulación 3 en radianes, en grados=180°
      0 %Valor de la articulación 4 en radianes, en grados=0°
 1.5168 %Valor de la articulación 5 en radianes, en grados=87°
-0.0000 %Valor de la articulación 6 en radianes, en grados=0°
```

```
>> T4=directkinematic(robot,soluciones(:,4))
>> drawrobot3d(robot,soluciones(:,4))
```

```
T4 =

   -0.0000   -0.0000    1.0000    0.5150
   -0.0000    1.0000    0.0000    0.0000
   -1.0000   -0.0000   -0.0000    0.7120
         0         0         0    1.0000
```

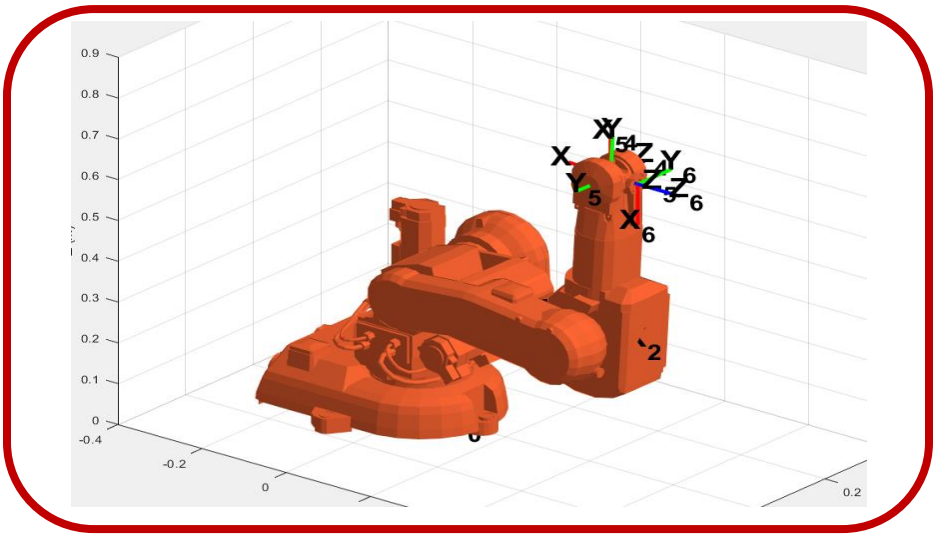


Ilustración 53. *Robot IRB-140 tomando el valor en sus articulaciones de la solución 4*

Solución 5:

```
 3.1416 %Valor de la articulación 1 en radianes, en grados=180°
-1.5278 %Valor de la articulación 2 en radianes, en grados=-87.5°
-0.4786 %Valor de la articulación 3 en radianes, en grados=-27.5°
 0.0000 %Valor de la articulación 4 en radianes, en grados=0°
-1.1352 %Valor de la articulación 5 en radianes, en grados=-65°
 3.1416 %Valor de la articulación 6 en radianes, en grados=180°
```

```
>> T5=directkinematic(robot,soluciones(:,5))
>> drawrobot3d(robot,soluciones(:,5))
```

```
T5 =

    0.0000   -0.0000    1.0000    0.5150
   -0.0000    1.0000    0.0000   -0.0000
   -1.0000   -0.0000    0.0000    0.7120
         0         0         0    1.0000
```

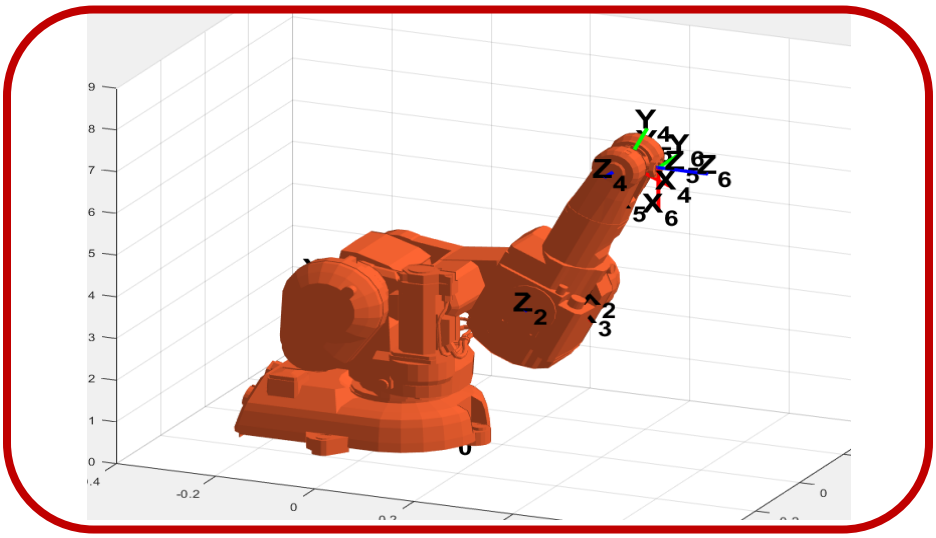


Ilustración 54. *Robot IRB-140 tomando el valor en sus articulaciones de la solución 5*

Solución 6:

```
 3.1416 %Valor de la articulación 1 en radianes, en grados=180°
-1.5278 %Valor de la articulación 2 en radianes, en grados=-87.5°
-0.4786 %Valor de la articulación 3 en radianes, en grados=-27.5°
-3.1416 %Valor de la articulación 4 en radianes, en grados=-180°
 1.1352 %Valor de la articulación 5 en radianes, en grados=65°
-0.0000 %Valor de la articulación 6 en radianes, en grados=0°
```

```
>> T6=directkinematic(robot,soluciones(:,6))
>> drawrobot3d(robot,soluciones(:,6))
```

```
T6 =

    0.0000    0.0000    1.0000    0.5150
   -0.0000    1.0000   -0.0000   -0.0000
   -1.0000   -0.0000    0.0000    0.7120
         0         0         0    1.0000
```

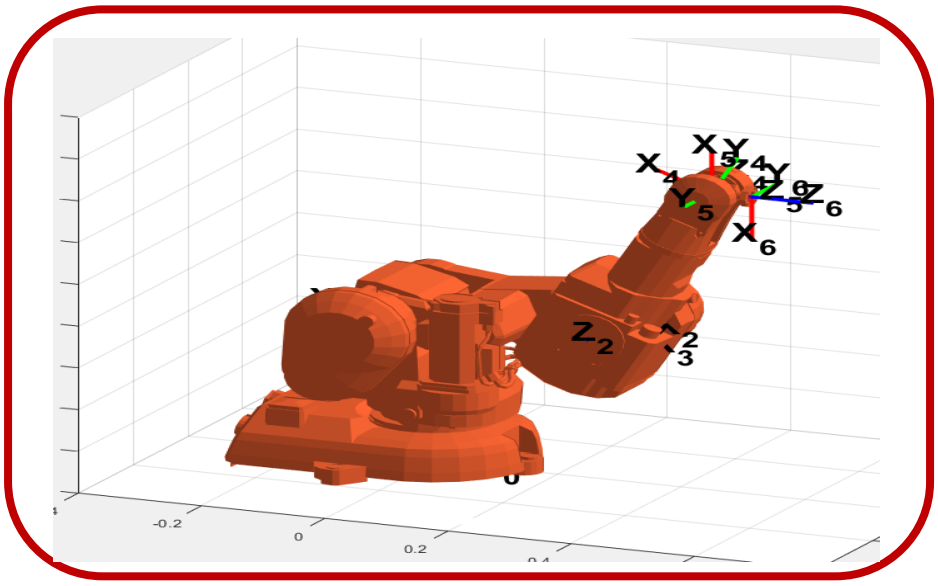


Ilustración 55. *Robot IRB-140 tomando el valor en sus articulaciones de la solución 6*

Solución 7:

```
 3.1416 %Valor de la articulación 1 en radianes, en grados=180°
-0.4027 %Valor de la articulación 2 en radianes, en grados=-23°
-2.6630 %Valor de la articulación 3 en radianes, en grados=-152°
 0.0000 %Valor de la articulación 4 en radianes, en grados=0°
-0.0759 %Valor de la articulación 5 en radianes, en grados=4.34°
 3.1416 %Valor de la articulación 6 en radianes, en grados=180°
```

```
>> T7=directkinematic(robot,soluciones(:,7))
>> drawrobot3d(robot,soluciones(:,7))
```

```
T7 =

   -0.0000   -0.0000    1.0000    0.5150
   -0.0000    1.0000    0.0000   -0.0000
   -1.0000   -0.0000   -0.0000    0.7120
         0         0         0    1.0000
```

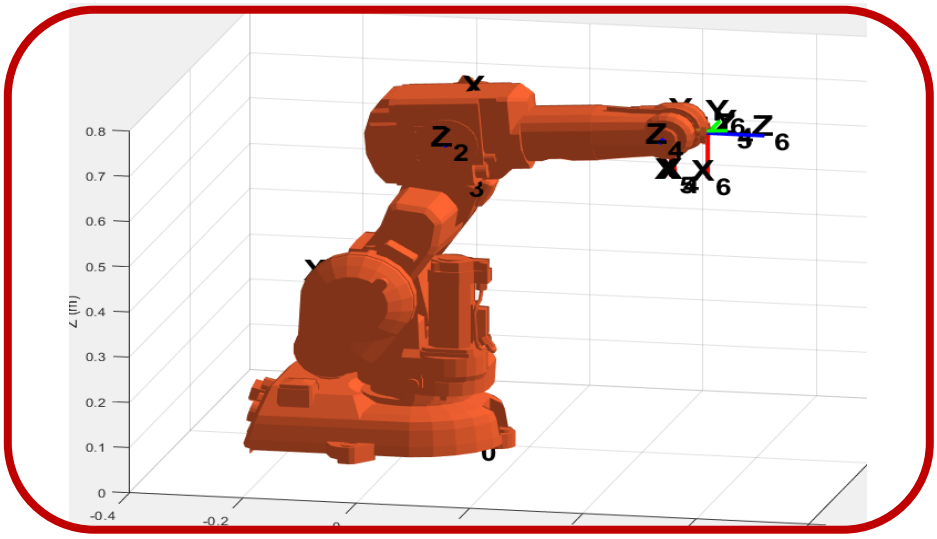


Ilustración 56. *Robot IRB-140 tomando el valor en sus articulaciones de la solución 7*

Solución 8:

```
 3.1416 %Valor de la articulación 1 en radianes, en grados=180°
-0.4027 %Valor de la articulación 2 en radianes, en grados=-23°
-2.6630 %Valor de la articulación 3 en radianes, en grados=-152°
-3.1416 %Valor de la articulación 4 en radianes, en grados=-180°
 0.0759 %Valor de la articulación 5 en radianes, en grados=4.34°
-0.0000 %Valor de la articulación 6 en radianes, en grados=0°
```

```
>> T8=directkinematic(robot,soluciones(:,8))
>> drawrobot3d(robot,soluciones(:,8))
```

```
T8 =

   -0.0000   -0.0000    1.0000    0.5150
   -0.0000    1.0000    0.0000   -0.0000
   -1.0000   -0.0000   -0.0000    0.7120
         0         0         0    1.0000
```

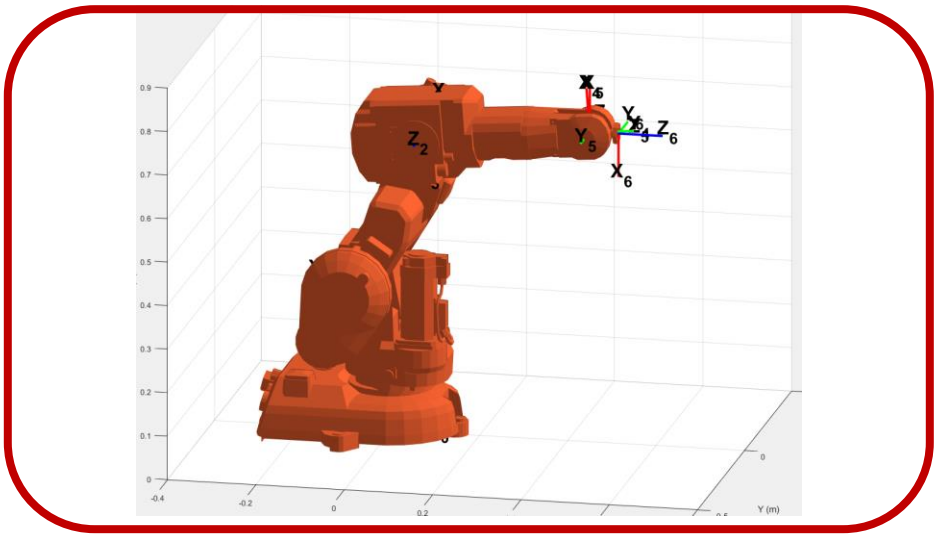


Ilustración 57. *Robot IRB-140 tomando el valor en sus articulaciones de la solución 8*

Las matrices homogéneas de las 8 soluciones $\mathbf{T1, T2, T3, T4, T5, T6, T7, T8}$ son las mismas a $\mathbf{T}$, por lo tanto, cualquier solución para llegar al punto ($\mathbf{x=0.515, y=0, z=0.712}$) es válida.

Dependiendo de la tarea que realizará el manipulador, de su campo de trabajo y otras variables como el peso de la herramienta final, los motores, la orientación deseada del efector final, etc., se elegirá la mejor configuración que tomará el robot para llegar a dicho punto.

3. CAPÍTULO 3: DESARROLLO DE DOS SIMULADORES VIRTUALES

El capítulo dos muestra cómo resolver la cinemática directa e inversa del manipulador IRB-140, en el software de Matlab se realizó las operaciones con tan solo insertar unas cuantas líneas de código, además se pudo observar de manera gráfica el manipulador, los ejes de coordenadas de cada sistema y el movimiento de las articulaciones.

Pero una de las desventajas de usar el software de Matlab, es que para poder hacer uso de esta herramienta se debe contar con conocimiento de programación. Por lo cual se tendría que impartir un curso para enseñar a los alumnos de la FES Aragón que cursan la materia de Robótica a programar en Matlab, esta no es una buena opción, pues se tendría que tomar tiempo de las 64 horas que dura el curso en el semestre, esto llevaría a no cubrir con todo el plan de estudios. Otra desventaja que presenta el usar Matlab es la repetición de las líneas de código para ejecutar un comando, lo cual hace tediosa la programación, porque si se llegase a equivocar en una sola letra, número o paréntesis el comando no se ejecuta y se tiene que ingresar de nuevo hasta que esté bien escrito.

Pero como se mencionó en el **Capítulo 2.1.1,** Matlab cuenta con la herramienta GUI que permite desarrollar aplicaciones interactivas para que los usuarios puedan hacer uso de la herramienta ahorrando tiempo y la necesidad de aprender Matlab.

En todo este capítulo se presenta el desarrollo de las dos aplicaciones que tienen como objetivo enseñar a los alumnos a comprender y resolver cinemática directa e inversa de los robots industriales.

3.1 DESARROLLO DE LA APLICACIÓN 1

El objetivo de esta primera aplicación es ayudar a los alumnos a que puedan comprobar si los parámetros de Denavit-Hartenberg obtenidos en clase fueron correctos utilizando gráficos en 3D. También ayudarlos a observar el comportamiento de las articulaciones primaticas o rotacionales del robot, dándoles valores a las articulaciones, además de que hallar la matriz de transformación homogénea que define al robot.

Además, la aplicación cuenta con algunos robots de ejemplo para que puedan observar cómo funciona la aplicación.

Para realizar la aplicación se insertó el siguiente código en la ventana de comandos de Matlab.

```
>> guide
```

En el recuadro que aparece, se seleccionó la opción "**Blank GUI (Default)**".

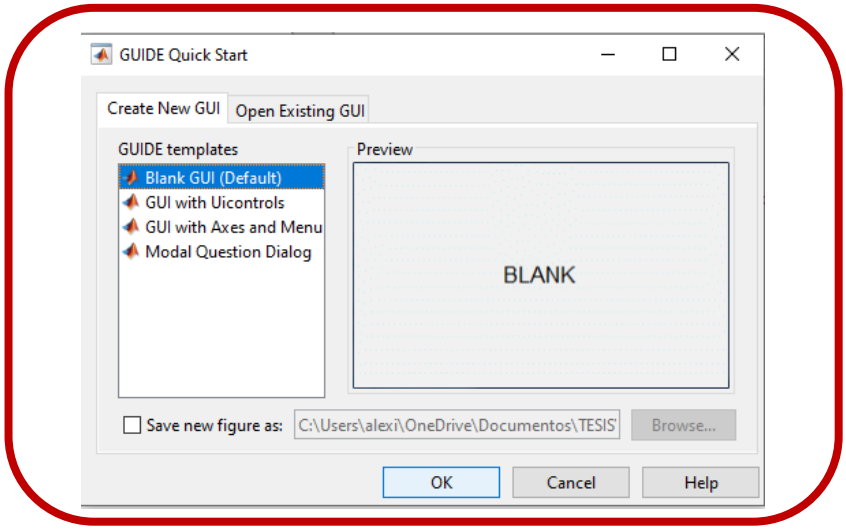


Ilustración 58. *Abrir una nueva GUI desde 0*

Al dar en el botón ok para continuar aparece una nueva ventana cuadriculada, es aquí donde se comenzó a desarrollar la parte visual con la que interactuara el alumno con el programa.

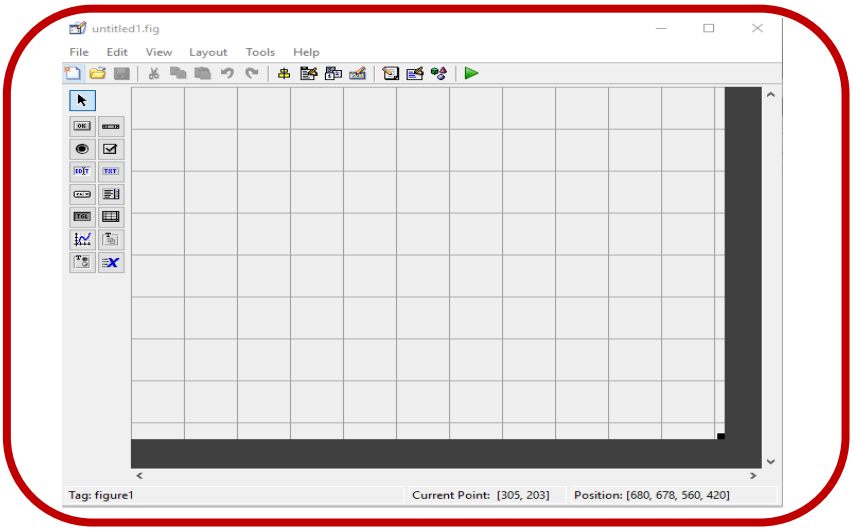


Ilustración 59. *Interface GUI desde 0*

Con la barra de herramientas que aparece del lado izquierdo de la interface GUI se colocaron los botones, gráficas, y con el Inspector se hizo la parte del diseño y las etiquetas de identificación de cada uno de los elementos. Dando como resultado la interface visual o “Frontend” de la aplicación (Véase **Ilustración 62**).

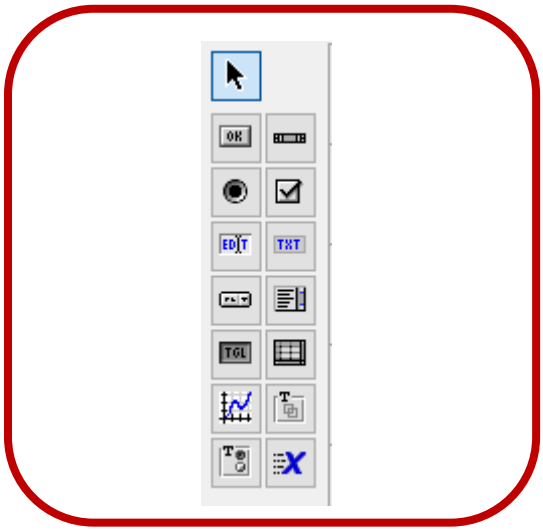


Ilustración 60. *Barra de herramientas de GUI*

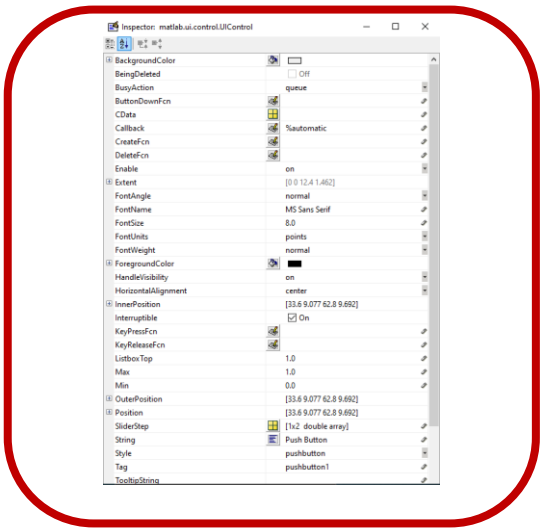


Ilustración 61. *Inspector para dar formato y diseño a las GUI de Matlab*

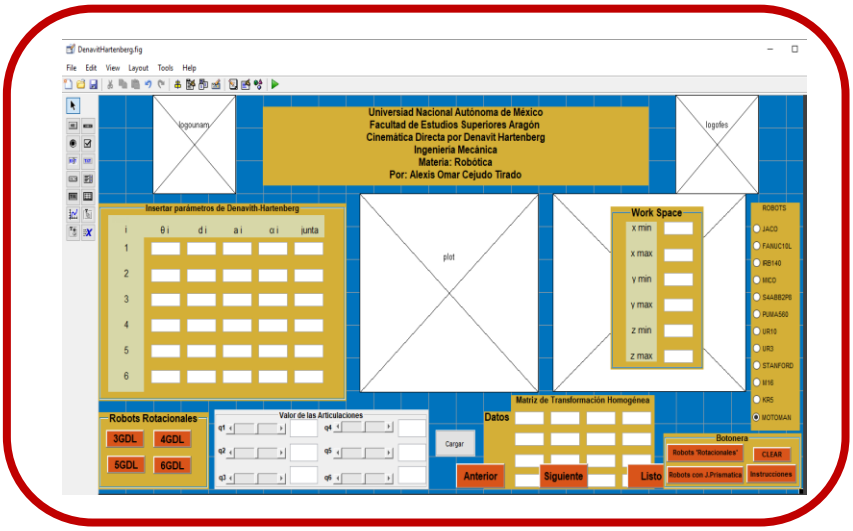


Ilustración 62. *Interface Visual de la aplicación*

Para que una aplicación funcione, se tiene que realizar el "Backend" o la programación, que hará los llamados a las funciones cada que el usuario de "click" a un botón o ingrese algún valor en un campo de texto (Véase **Apéndice 7** donde se muestra el código desarrollado para esta aplicación).

Para que una aplicación desarrollada en GUI de Matlab pueda ser utilizada en computadoras donde no es necesario tener instalado Matlab. Se crea el paquete que almacenará todos los archivos para crear el ejecutable.

En la ventana de comandos de Matlab se escribió el siguiente código

```
>> deploytool
```

Abre una ventana del compilador de Matlab y se seleccionó la primera opción "**Application Compiler**" una vez ahí se seleccionó el archivo principal que contiene todo el código y además se seleccionó todos los archivos secundarios para que la aplicación funcione.

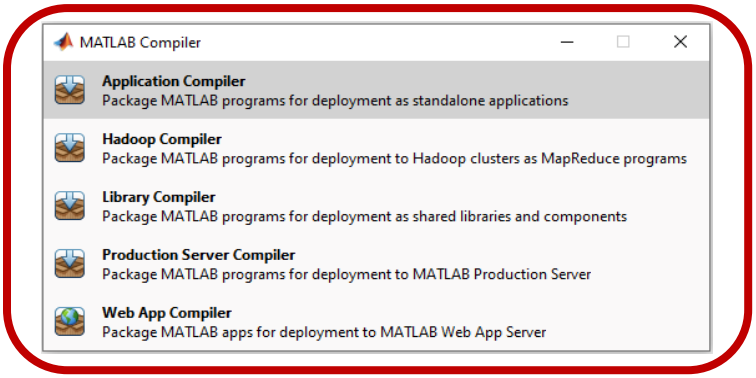


Ilustración 63. *Matlab Compiler*

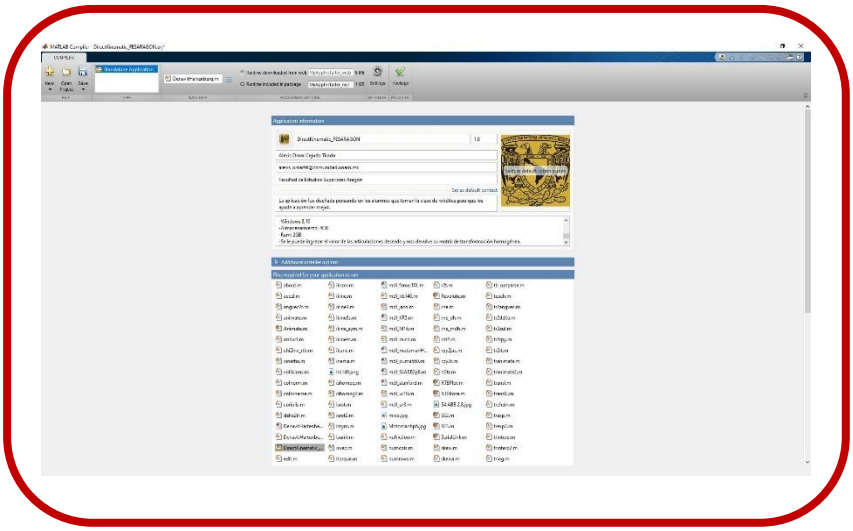


Ilustración 64. *Empaquetado para crear el ejecutable.*

Una vez creado, se instaló la aplicación en el ordenador (Véase **Apéndice 8** para observar el proceso de instalación de la aplicación).

3.2 DESARROLLO DE LA APLICACIÓN 2

Se desarrollo de la segunda aplicación para acercar a los alumnos a observar los conceptos básicos de la robótica en robots industriales utilizados actualmente. Como ya se había mencionado anteriormente el conseguir un robot de esta magnitud es muy costoso para la institución.

Este simulador tiene como objetivo, desarrollar la cinemática directa obteniendo la matriz de transformación homogénea de cada robot, además de obtener la cinemática inversa y que los alumnos observen que un robot puede llegar a determinado punto con diferentes valores en sus articulaciones. Además de mostrar los sistemas de referencia de cada articulación para que los alumnos puedan obtener los parámetros de Denavit-Hartenberg de los manipuladores.

También tiene la opción de cargar efectores finales y campos de trabajo para que el robot pueda simular alguna tarea con diferentes puntos. Como los utiliza el código RAPID. Este es un lenguaje de programación para robots de la marca ABB. [34]

Esta segunda aplicación está desarrollada con la librería ARTE Robotics, sólo que, se realizaron algunos cambios para que esta librería sea de fácil manejo con tan solo unos “cliks”, además que sea entendible para que se pueda obtener la

cinemática directa e inversa de una manera más sencilla, y por último que pueda ejecutarse la librería sin necesidad de tener Matlab.

El desarrollo de esta aplicación fue el siguiente:

En la ventana de comandos se insertó el código siguiente, para abrir una Guide nueva.

```
>> guide
```

Se seleccionó la opción "**Blank GUI (Default)**".

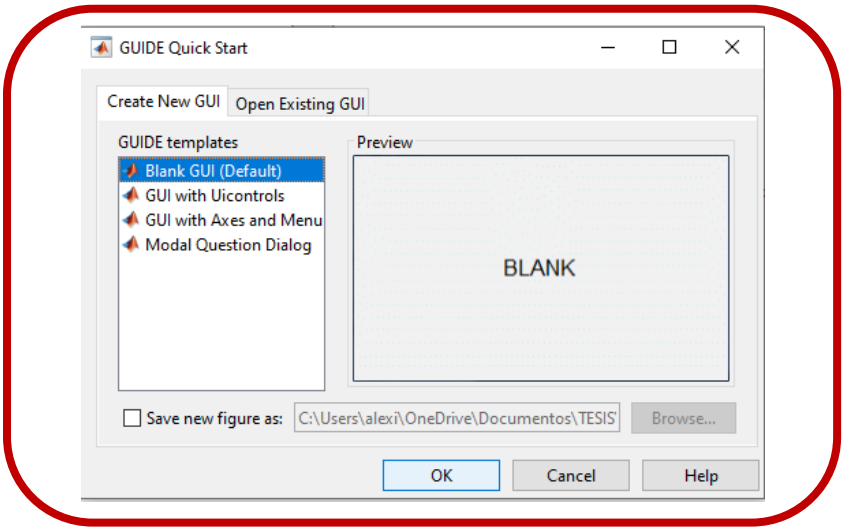


Ilustración 65. *Abrir una nueva GUI desde 0*

Una vez adentro se comenzó a dar formato y color a los botones y paneles de la aplicación con ayuda del inspector y la barra de herramientas (**Ilustración 60** y **61**). El "frontend" o la Interfaz Visual quedo de la siguiente manera:

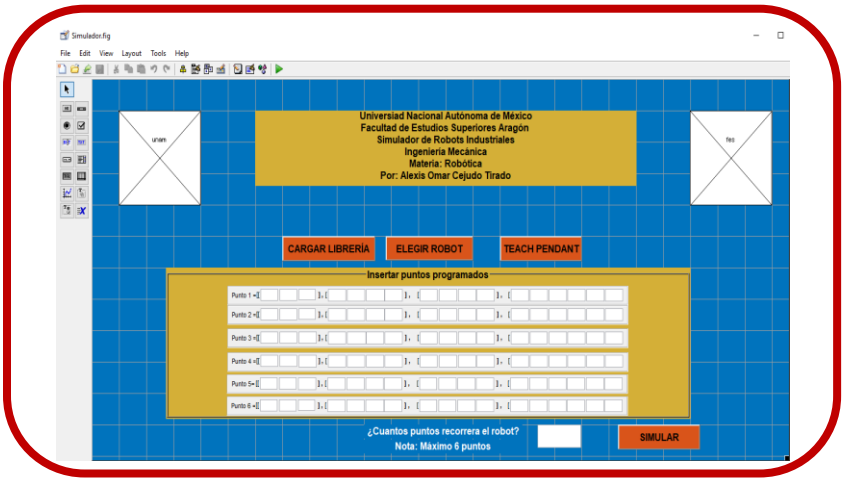


Ilustración 66. *Frontend o Interface Visual*

Para que la aplicación funcione correctamente, se realizó en “Backend” o código del programa para que el usuario con tan solo dar clic en los botones, la aplicación realice el llamado a las funciones necesarias para hacer uso de la librería ARTE y sea de fácil acceso (Véase **Apéndice 9** para observar el código).

Al igual que la aplicación anterior se creó el paquete para que esta aplicación se pueda utilizar sin tener instalado el software de Matlab.

En la ventana de comandos de Matlab se escribió el siguiente código:

```
>> deploytool
```

En el compilador de Matlab se seleccionó la opción “**Applicaction Compiler**” y se almacenó todos los archivos necesarios para que la aplicación funcione correctamente.

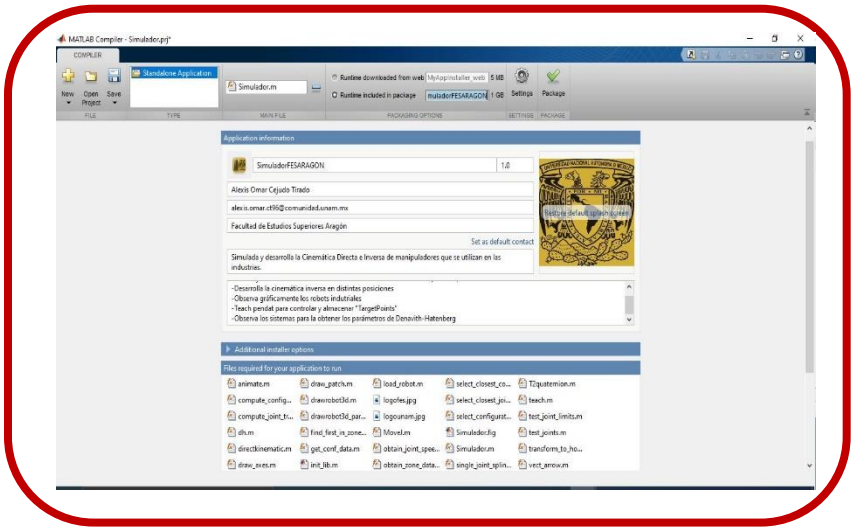


Ilustración 67. *Empaquetado para crear el ejecutable de la aplicación 2.*

Después se realizó la instalación en el ordenador para que se pueda hacer uso del simulador. (Véase **Apéndice 10** para ver el proceso de instalación de la aplicación 2).

4. CAPÍTULO 4: RESULTADOS DE LAS APLICACIONES CON LA TEORÍA

En este capítulo se muestra el funcionamiento de las dos aplicaciones, utilizando el mismo robot que se describió en el **Capítulo 2.4**. Se comprobó que ambas aplicaciones funcionen de manera correcta, esto se hizo comparando las ecuaciones que se obtuvieron en los **capítulos 2.4.1** y **2.4.2.**

4.1 PRUEBA DE LA APLICACIÓN 1

Con la primera aplicación se comprobó si los parámetros de Denavit-Hatenberg que se obtuvieron en la **Tabla 1.12** sean correctos. Esto se corroboro con la matriz de transformación homogénea desde la aplicación 1 y comparó con las obtenidas en el **Capítulo 2.4.1** en las **Matrices (14)** y **(15)**.

Se ejecutó la aplicación que se instaló el ordenador.

Ilustración 68. *Inicio de la aplicación*

Una vez adentro en el recuadro "**Insertar parámetros de Denavit-Hartenberg**" se ingresó los parámetros de la **Tabla 1.12**. Siguiendo los siguientes pasos:

1. En la Panel de botones, denominado "**Botonera**" se seleccionó la opción robot "**Rotacionales**" (porque el robot solamente cuenta con juntas rotacionales). Esto activó otra botonera con el nombre "**Robots Rotacionales**"

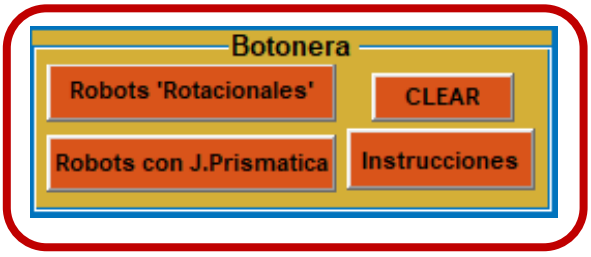


Ilustración 69. *Panel de Botones "Botonera"*

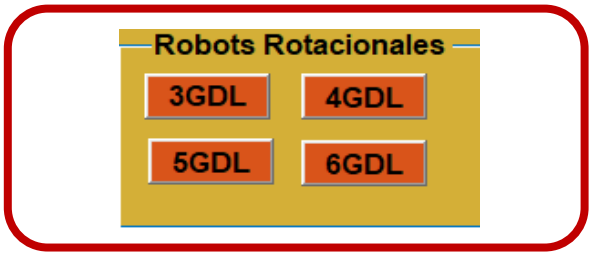


Ilustración 70. *Panel de Botones "Robots Rotacionales"*

2. Después se ingresó los parámetros de la **Tabla 1.12** con las condiciones siguientes:
   2.1 Al ser una junta rotacional el parámetro **Theta i (θi)** tomo el valor **0**
   2.2 En la columna denominada "**junta**" al ser juntas rotacionales tomo el valor de **0.**

Por lo tanto, el panel "**Insertar parámetros de Denavit-Hartenber**" quedó de la siguiente manera.

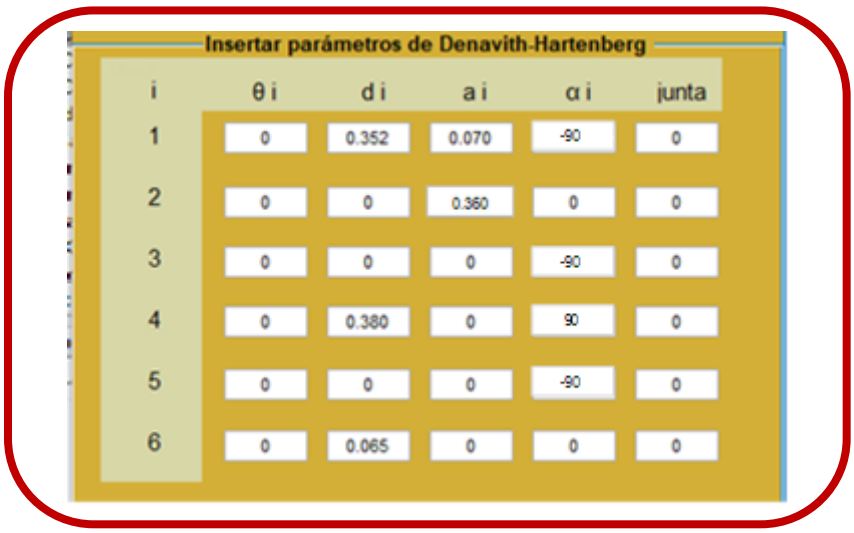


Ilustración 71. *Parámetros del robot en la aplicación*

3. En el panel de los “sliders” con el nombre “**Valor de las articulaciones**” se colocó los giros que se necesitaban para coincidir los ejes X_{i-1} con el eje X_i al cómo se obtuvo en la **Tabla 1.12.** En este caso **q2** y **q6** tienen un giro de -90° y 180° respectivamente.

θi
θ_1
θ_2-90
θ_3
θ_4
θ_5
θ_6+180

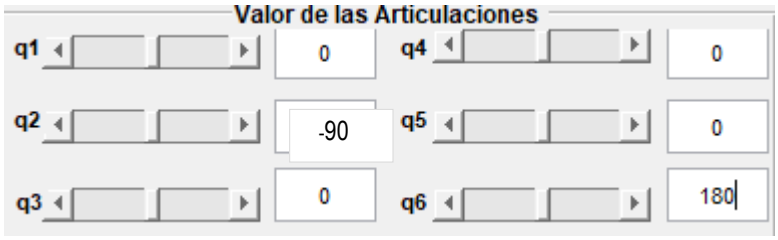


Ilustración 72. *Valor de las Articulaciones del Robot en el Slider*

Una vez ingresados los parámetros del robot se dio clic en el botón **6GDL** en la botonera “**Robots Rotacionales**”. Dando como resultado una representación gráfica del robot (Donde los cilindros en color rojo representan la ubicación de las articulaciones y los de color azul muestran la representación de los eslabones).

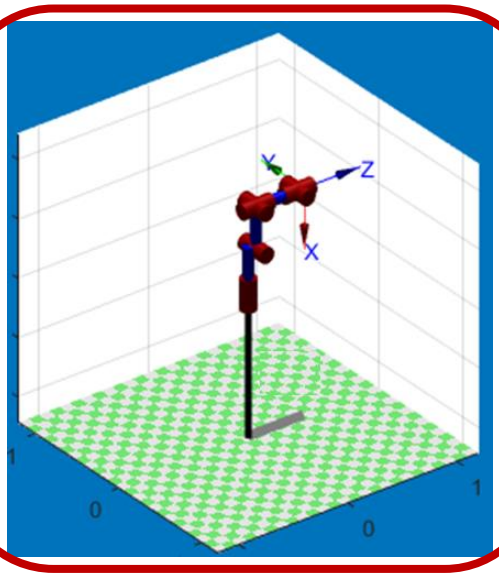


Ilustración 73. *Representación gráfica del robot IRB-14***0**

Al mismo tiempo se desarrolló la cinemática directa del robot, mostrando su matriz de transformación homogénea.

Ilustración 74. *Matriz de transformación Homogénea obtenida con la aplicación*

Se comparó la matriz que se obtuvo con la aplicación contra las del **Capítulo 2.4.1** (**Matrices** (**14**) y (**15**)) se notó que son la misma matriz por lo tanto la aplicación funciona correctamente.

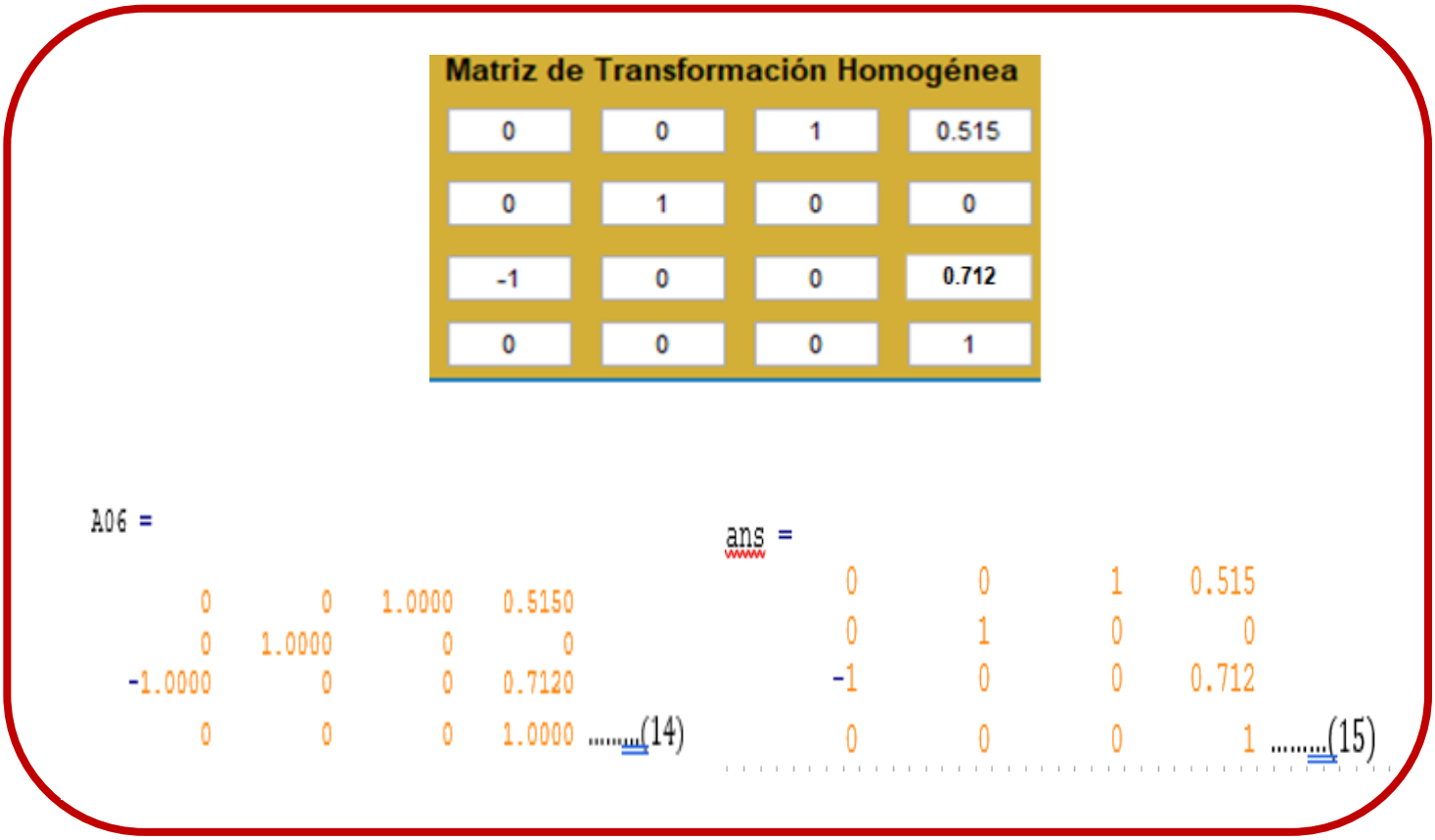


Ilustración 75. *Comparación de resultados*

4.2 PRUEBA DE LA APLICACIÓN 2

Primero se abrió la aplicación que se instaló en el ordenador.

Ilustración 76. *Inicio de la Aplicación 2*

Una vez en la aplicación se presionó en el botón "**Cargar Librería**" para cargar todos los archivos necesarios para que la aplicación funcione correctamente.

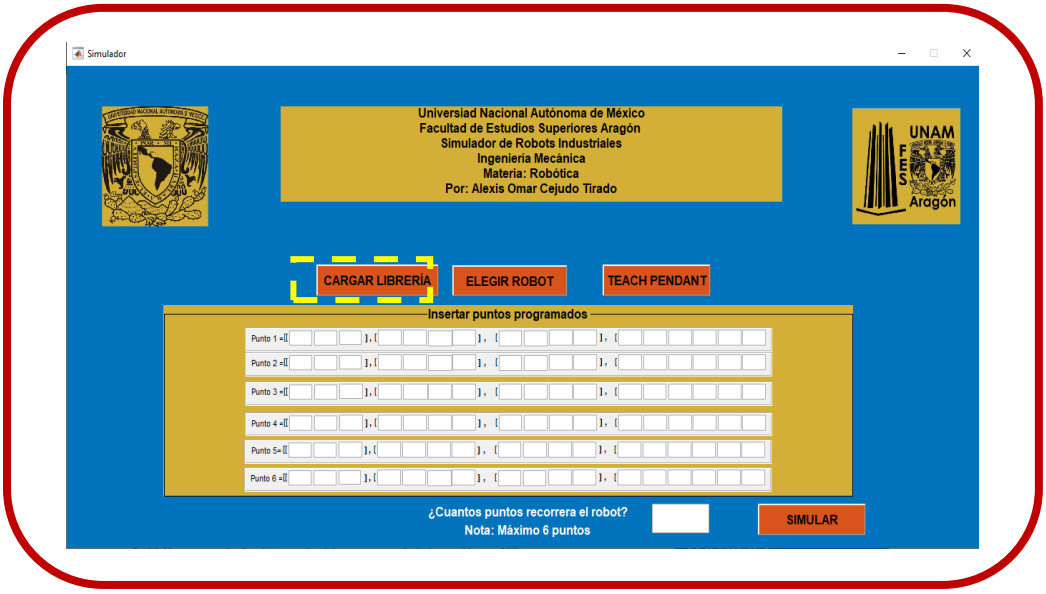


Ilustración 77. *Inicio de la Librería Arte*

Se selecciono el robot dando clic en el botón “**Elegir Robot**”.

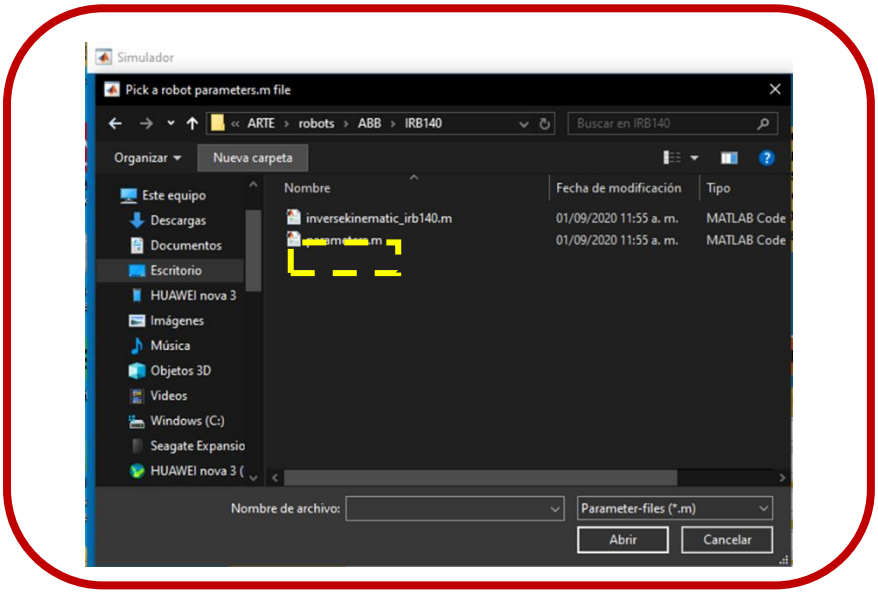


Ilustración 78. *Selección de los parámetros del Robot*

Una vez que cargo el robot y apareció el gráfico, se presionó el botón “**Teach Pendant**” para cargar el controlador del robot.

Para dar un mejor ejemplo de la teoría con la práctica se cargó un “**gripper**” para agarrar objetos.

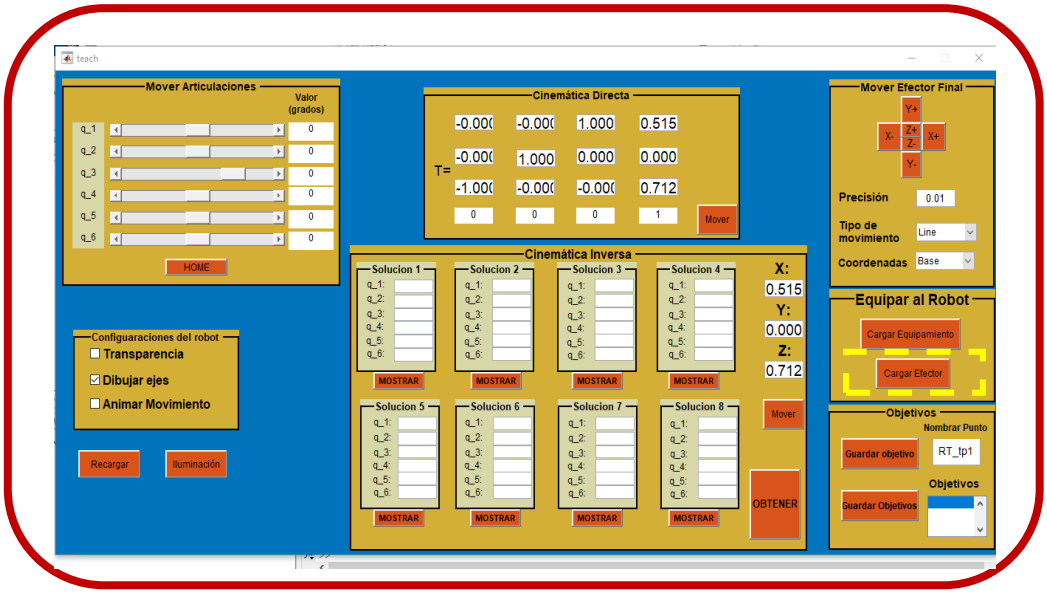


Ilustración 79. *Cargar efector Final al Robot*

Ahora el robot está listo para realizar su estudio de cinemática directa.

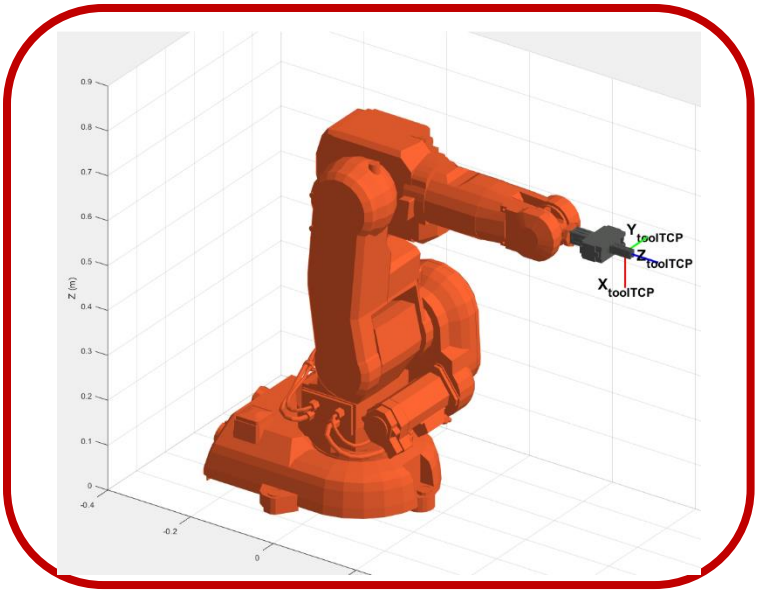


Ilustración 80. *Robot con Gripper*

Primero se comprobó que la matriz de transformación homogénea que da la aplicación 2 sea correcta para la posición home como se muestra en la **Ilustración 80**.

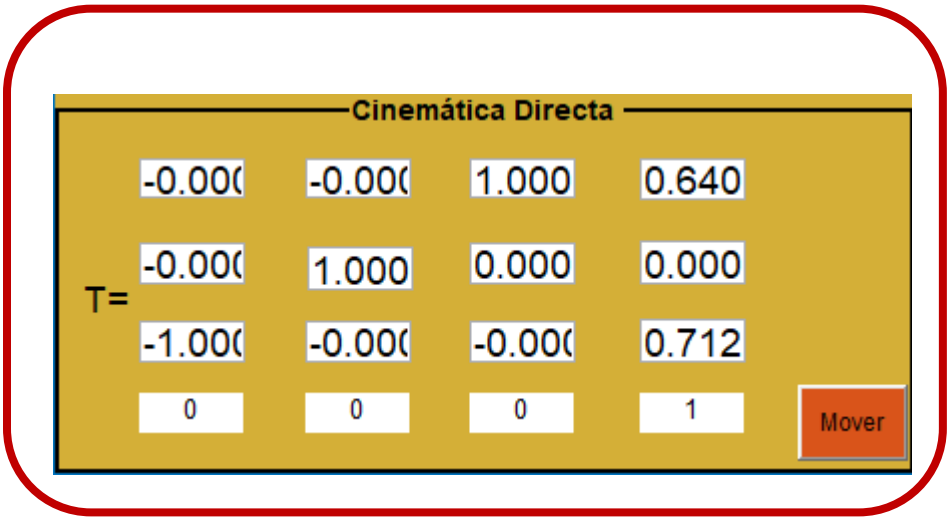


Ilustración 81. *Matriz de Transformación Homogénea con Gripper*

A comparación con la matriz de transformación homogénea de la **Ilustración 75,** la posición en **X** aumenta por las dimensiones del "gripper" (**Ilustración 82**).

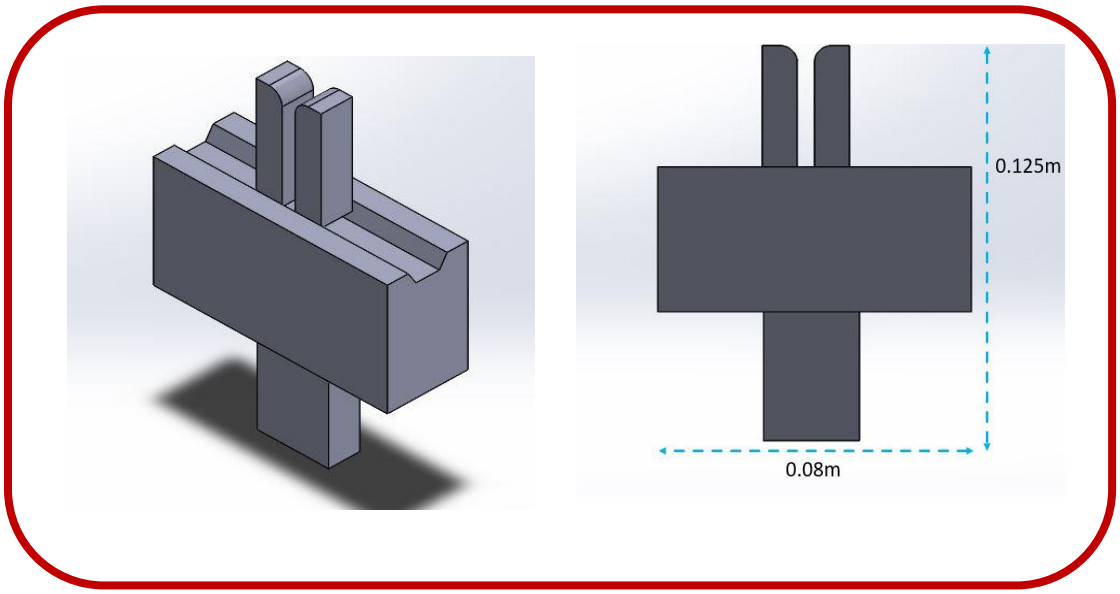


Ilustración 82. *Dimensiones del Gripper equipado*

Por lo cual el sistema 6 ahora será colocado en la punta del “gripper”, por lo tanto, las dimensiones cambiaron en el último eslabón.

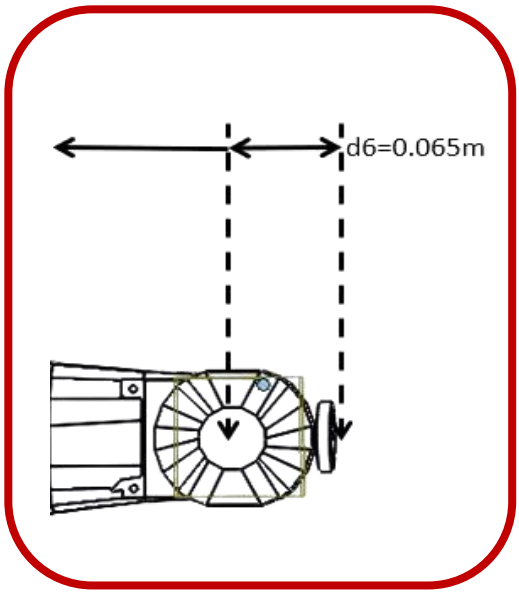


Ilustración 83. *Dimensión sin gripper*

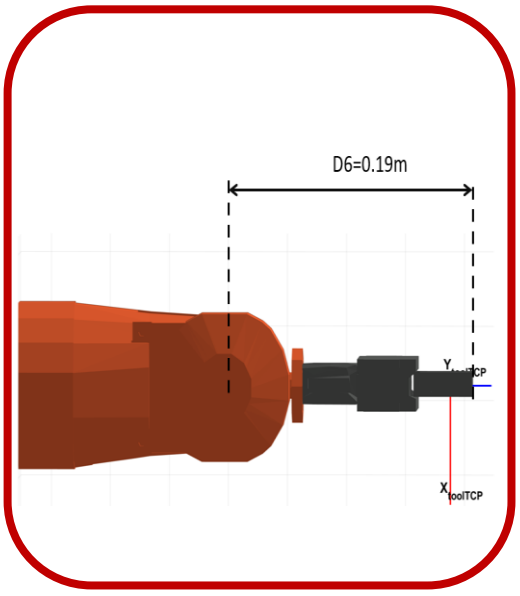


Ilustración 84. *Dimensión con gripper*

Como primer paso se obtuvo los parámetros de Denavit-Hartenberg con ayuda de los sistemas de referencia que ya tiene incluido el robot.

Parámetros entre el sistema **S0** (Sistema inercial) y el sistema **S1**.

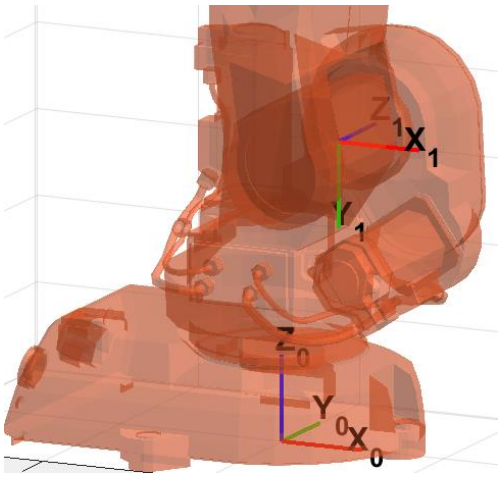
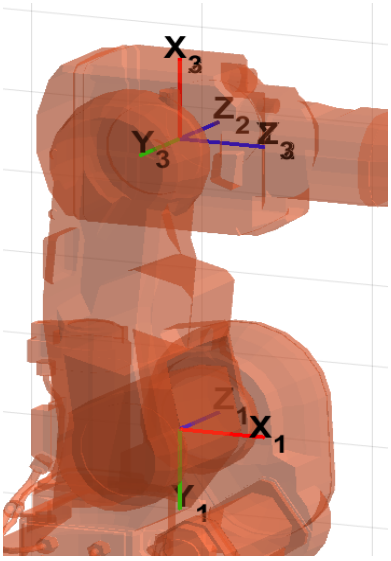


Tabla de parámetros Denavit-Hartenberg Robot IRB-140				
Sistema	θi	di (m)	ai (m)	αi
1	θ_1	D1	A1	-90°

Parámetros entre el sistema **S1** y **S2.**

Tabla de parámetros Denavit-Hartenberg Robot IRB-140				
Sistema	θi	di (m)	ai (m)	αi
1	θ_1	D1	A1	-90°
2	$\theta_{2+}90$	0	A2	0



Parámetros entre el sistema **S2** y **S3.**

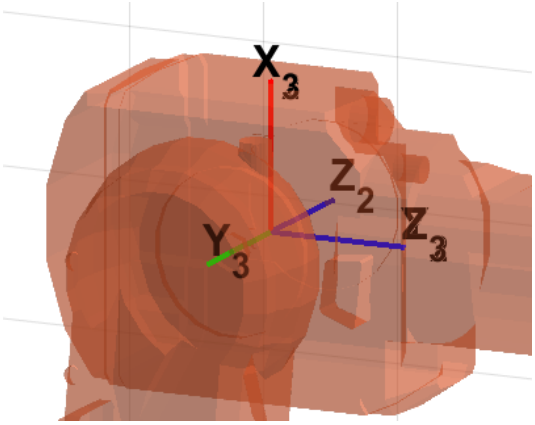
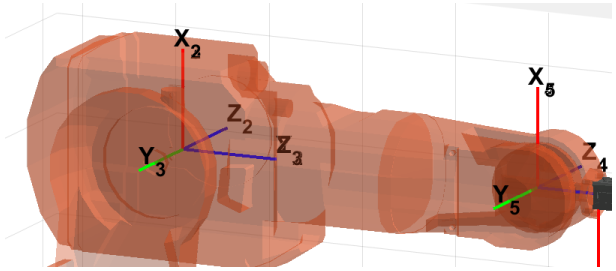


Tabla de parámetros Denavit-Hartenberg Robot IRB-140				
Sistema	θi	di (m)	ai (m)	αi
1	θ_1	D1	A1	-90°
2	θ_2-90	0	A2	0
3	θ_3	0	0	-90°

Parámetros entre el sistema **S3** y **S4.**

Tabla de parámetros Denavit-Hartenberg Robot IRB-140				
Sistema	θi	di (m)	ai (m)	αi
1	θ_1	D1	A1	-90°
2	θ_2-90	0	A2	0
3	θ_3	0	0	-90°
4	θ_4	D4	0	90°



Parámetros entre el sistema **S4** y **S5.**

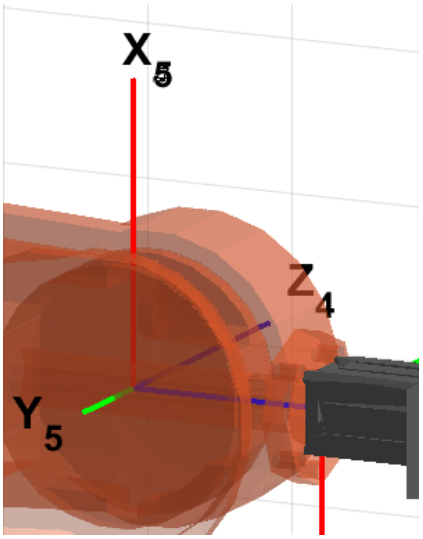


Tabla de parámetros Denavit-Hartenberg Robot IRB-140				
Sistema	θi	di (m)	ai (m)	αi
1	θ_1	D1	A1	-90°
2	θ_2-90	0	A2	0
3	θ_3	0	0	-90°
4	θ_4	D4	0	90°
5	θ_5	0	0	-90°

Parámetros entre el sistema **S5** y **S6** (En la imagen de abajo el **sistema S6** cambia al "**sistema tool"** ubicado en la punta del gripper).

Tabla de parámetros Denavit-Hartenberg Robot IRB-140				
Sistema	θi	di (m)	ai (m)	αi
1	θ_1	D1	A1	-90°
2	θ_2-90	0	A2	0
3	θ_3	0	0	-90°
4	θ_4	D4	0	90°
5	θ_5	0	0	-90°
6	θ_6+180	D6	0	0

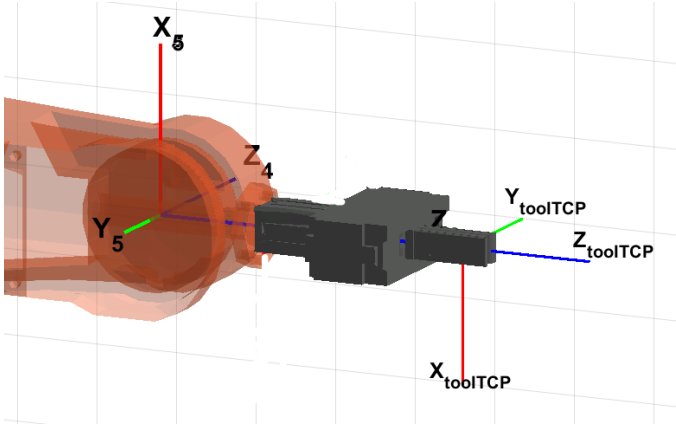


Tabla de parámetros Denavit-Hartenberg Robot IRB-140				
Sistema	θi	di (m)	ai (m)	αi
1	θ_1	0.352	0.070	-90°
2	θ_2-90	0	0.352	0
3	θ_3	0	0	-90°
4	θ_4	0.380	0	90°
5	θ_5	0	0	-90°
6	θ_6+180	0.19	0	0

Tabla 1.13. *Parámetros de DH obtenidos con la aplicación 2*

La **Tabla 1.13** y la **Tabla 1.12** son muy parecidas a diferencia del valor del último eslabón y esto es porque se agregó un "gripper" como efector final.

Se tomó todas las bases del **Capítulo 2.4.1.1** para pasar de un sistema **Si-1** a un sistema **Si** por 4 transformaciones.

1. Una rotación por el eje Zi-1 (θi).
2. Una translación a lo largo de eje Zi-1(di)
3. Una translación a lo largo del eje Xi (ai)
4. Rotación por el eje Xi (αi).

$$^{i-1}A_i = T(z, \theta_i)\ T(0, 0, d_i)\ T(a_i, 0, 0)\ T(x, \alpha_i) \ldots (68)$$

Entonces la matriz que define el cambio de base del sistema **Si-1** al sistema **Si** es:

$$^{i-1}\mathbf{A}_i = \begin{bmatrix} C\theta i & -C\alpha i S\theta i & S\alpha i S\theta i & aiC\theta i \\ S\theta i & C\alpha i C\theta i & -S\alpha i C\theta i & aiS\theta i \\ 0 & S\alpha i & C\alpha i & di \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (69)$$

Primero se realizó el cambio de base entre el sistema inercial **(S0)** con el sistema **S1**, se sabe que, para que el robot tome la posición de "**Home**" $\boldsymbol{\theta_1 = \theta_2 = \theta_3 = \theta_4 = \theta_5 = \theta_6 = 0°}$

Por lo tanto, se sustituyeron los primeros parámetros del sistema 1 en la **Ecuación (68).**

Sistema	$\theta 1$	d1 (m)	a1(m)	α1
1	θ_1=0°	0.352	0.070	90°

$$ {}^{0}\mathbf{A}_{1} = \begin{bmatrix} C(0) & -C(-90)S(0) & S(-90)S(0) & 0.070C(0) \\ S(0) & C(-90)C(0) & -S(-90)C(0) & 0.070S(0) \\ 0 & S(-90) & C(-90) & 0.352 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (70) $$

Se sabe que, $\mathbf{Sen\ (0) = 0, Sen\ (90) = 1, Cos\ (0) = 1, Cos\ (90) = 0}$. La matriz que define el cambio de base del sistema $\mathbf{S0}$ al sistema $\mathbf{S1}$ quedo de la siguiente manera:

$$ {}^{0}\mathbf{A}_{1} = \begin{bmatrix} 1 & 0 & 0 & 0.070 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0.352 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (71) $$

Se realizó el mismo procedimiento, pero con los valores correspondientes a cada sistema de la **Tabla 1.13**.

Cambio de base entre **sistema 1** y **sistema 2**

Sistema	$\theta 2$	d2 (m)	a2 (m)	α2
2	θ_2=0°-90°=-90°	0	0.360	0

$$ {}^{1}\mathbf{A}_{2} = \begin{bmatrix} C(-90) & -C(0)S(-90) & S(0)S(-90) & 0.360C(-90) \\ S(-90) & C(0)C(-90) & -S(0)C(-90) & 0.360S(-90) \\ 0 & S(0) & C(0) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (72) $$

$$^{1}\mathbf{A}_{2} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & -0.360 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \dots (73)$$

Cambio de base entre **sistema 2** y **sistema 3**

Sistema	$\theta 3$	d3 (m)	a3 (m)	α3
3	θ_3=0°	0	0	-90°

$$^{2}\mathbf{A}_{3} = \begin{bmatrix} C(0) & -C(-90)S(0) & S(-90)S(0) & 0\,C(0) \\ S(0) & C(-90)C(0) & -S(-90)C(0) & 0\,S(0) \\ 0 & S(-90) & C(-90) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \dots (74)$$

$$^{2}\mathbf{A}_{3} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \dots (75)$$

Cambio de base entre **sistema 3** y **sistema 4**

Sistema	$\theta 4$	d4 (m)	a4 (m)	α4
4	θ_4=0°	0.380	0	90°

$$^{3}\mathbf{A}_{4} = \begin{bmatrix} C(0) & -C(90)S(0) & S(90)S(0) & 0\,C(-0) \\ S(0) & C(90)C(0) & -S(90)C(0) & 0\,S(-0) \\ 0 & S(90) & C(90) & 0.380 \\ 0 & 0 & 0 & 1 \end{bmatrix} \dots (76)$$

$$^{3}\mathbf{A}_{4} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0.380 \\ 0 & 0 & 0 & 1 \end{bmatrix} \dots (77)$$

Cambio de base entre **sistema 4** y **sistema 5**

Sistema	$\theta 5$	d5 (m)	a5 (m)	α5
5	θ_5=0°	0	0	-90°

$$^{4}\mathbf{A}_{5} = \begin{bmatrix} C(0) & -C(-90)S(0) & S(-90)S(0) & 0C(0) \\ S(0) & C(-90)C(0) & -S(-90)C(0) & 0S(0) \\ 0 & S(-90) & C(-90) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (78)$$

$$^{4}\mathbf{A}_{5} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (79)$$

Cambio de base entre **sistema 5** y **sistema 6** (Efector final). Se sabe que, **Sen (180°) =0 y Cos (180°) =-1**.

Sistema	$\theta 6$	d6(m)	a6 (m)	α6
6	θ_6=0°+180=180°	0.19	0	0

$$^{5}\mathbf{A}_{6} = \begin{bmatrix} C(180) & -C(0)S(180) & S(0)S(180) & 0C(180) \\ S(180) & C(0)C(180) & -S(0)C(180) & 0S(180) \\ 0 & S(0) & C(0) & 0.19 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (80)$$

$$^{5}\mathbf{A}_{6} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0.19 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (81)$$

Se obtuvo la matriz que define la orientación y posición del **sistema S6** (efector final) respecto al **sistema S0** (base o sistema inercial), para esto se multiplicó todas las matrices de $^{0}\mathbf{A}_{1}$ hasta $^{5}\mathbf{A}_{6}$ en orden sucesivo.

$$T = {}^{0}\mathbf{A}_{1} * {}^{1}\mathbf{A}_{2} * {}^{2}\mathbf{A}_{3} * {}^{3}\mathbf{A}_{4} * {}^{4}\mathbf{A}_{5} * {}^{5}\mathbf{A}_{6} \ldots (82)$$

$$^0A_2 = {}^0A_1 * {}^1A_2$$

$$^0A_1 \qquad\qquad ^1A_2 \qquad\qquad ^0A_2$$

$$\begin{bmatrix} 1 & 0 & 0 & 0.070 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0.352 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & -0.360 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0.070 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0.712 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^0A_3 = {}^0A_2 * {}^2A_3$$

$$^0A_2 \qquad\qquad ^2A_3 \qquad\qquad ^0A_3$$

$$\begin{bmatrix} 0 & 1 & 0 & 0.070 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0.712 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0.070 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0.712 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^0A_4 = {}^0A_3 * {}^3A_4$$

$$^0A_3 \qquad\qquad ^3A_4 \qquad\qquad ^0A_4$$

$$\begin{bmatrix} 0 & 0 & 1 & 0.070 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0.712 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0.380 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0.45 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0.712 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^{0}A_5 = {}^{0}A_4 * {}^{4}A_5$$

$$\underset{^{0}A_4}{\begin{bmatrix} 0 & 1 & 0 & 0.45 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0.712 \\ 0 & 0 & 0 & 1 \end{bmatrix}} * \underset{^{4}A_5}{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}} = \underset{^{0}A_5}{\begin{bmatrix} 0 & 0 & 1 & 0.45 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0.712 \\ 0 & 0 & 0 & 1 \end{bmatrix}}$$

$$T = {}^{0}A_6 = {}^{0}A_5 * {}^{5}A_6$$

$$\underset{^{0}A_5}{\begin{bmatrix} 0 & 0 & 1 & 0.45 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0.712 \\ 0 & 0 & 0 & 1 \end{bmatrix}} * \underset{^{5}A_6}{\begin{bmatrix} -1 & 0 & 0 & 0.45 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0.19 \\ 0 & 0 & 0 & 1 \end{bmatrix}} = \underset{T}{\begin{bmatrix} 0 & 0 & 1 & 0.64 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0.712 \\ 0 & 0 & 0 & 1 \end{bmatrix}}$$

$$T = \begin{bmatrix} 0 & 0 & 1 & 0.64 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0.712 \\ 0 & 0 & 0 & 1 \end{bmatrix} \dots\dots (83)$$

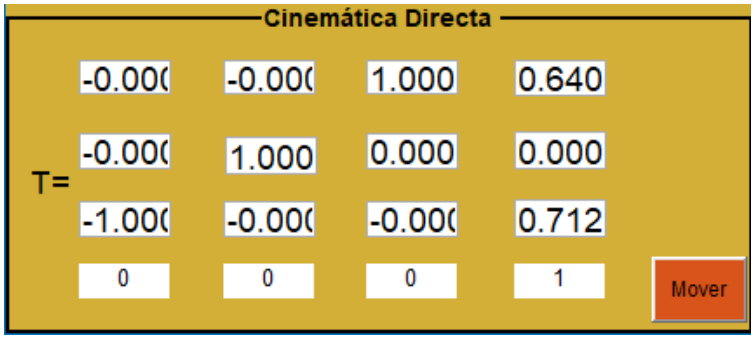


Ilustración 85. *Parámetros de DH obtenidos con la aplicación 2*

Se observó la misma matriz de transformación homogénea, sólo que, la que se obtuvo con la aplicación, tiene valores de **-0** esto es debido a cuestiones de redondeo en la programación.

En ambas matrices se observa que la posición a la que llegó el efector final fue en **x=0.640** en **y=0.000** y en **z=0.712**. y la orientación del efector final está dada por la matriz **noa**.

$$noa = \begin{vmatrix} n_x & o_x & a_x \\ n_y & o_y & a_y \\ n_z & o_z & a_z \end{vmatrix} = \begin{vmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{vmatrix} .....(84)$$

Donde:

- El **vector a** es la dirección en Z del efector final
- El **vector o** es perpendicular a el vector a y a las pinzas del “gripper”.
- El **vector n** es el resultado del producto cruz entre el **vector o** y el **vector a** (**n=oxa**)

En este caso:

- El **Vector a** tiene componentes en **x=1**, **y=0**, **z=0** (**a= (1,0,0)**) por lo cual este vector es paralelo al **eje x** de la base.
- El **vector o** tiene componentes en **x=0 y= 1** y **z= 0 (o=(0,1,0))** este vector es paralelo al **eje y** de la base
- El producto cruz entre el **vector o** y **a** da como resultado del vector n, donde **n=oxa**, **n=(0,1,0) x (1,0,0)= (0 0 -1)**, da como resultado un vector perpendicular al plano formado entre los **vectores o** y **a**. Este vector es paralelo al **eje Z** en dirección negativa.

En la **ilustración 86** se muestra de manera visual el **sistema noa** que se obtuvo con ayuda de la aplicación.

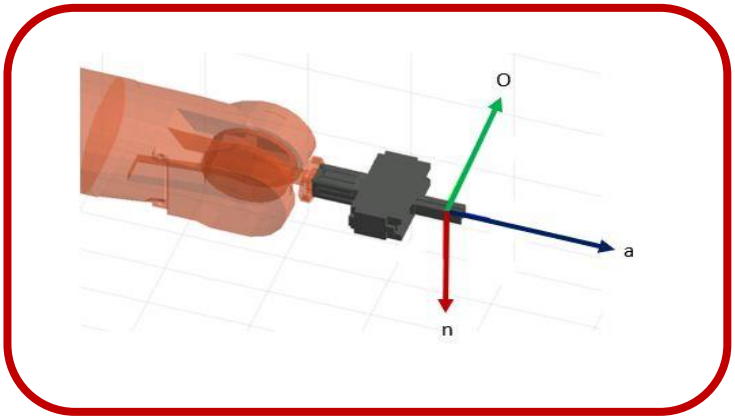


Ilustración 86. *Matriz noa en el efector final*

Además de poder observar la cinemática directa, la aplicación permite obtener la cinemática inversa.

Como ya se mencionó en capítulos anteriores la cinemática inversa obtiene el valor de las articulaciones que se necesitan para llegar a un punto deseado, en este caso lo único que se conoce son las coordenadas en $\mathbf{X, Y, Z}$ del punto al que se quiere llegar y la **orientación** deseada del efector final.

Para ver si la aplicación funciona correctamente, se movió el manipulador a un punto diferente al de "Home" ($\mathbf{X=515, Y=0, Z=0.712}$), el punto que se inserto está dentro del campo de trabajo del manipulador

$$\mathbf{X= 0.700}$$

$$\mathbf{Y=0.100}$$

$$\mathbf{Z=0.500}$$

Y la orientación del efector final con la llegó el robot fue:

$$\begin{vmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{vmatrix}$$

Se presionó el botón “**mover”** y el manipulador se colocó en el punto y la orientación dada.

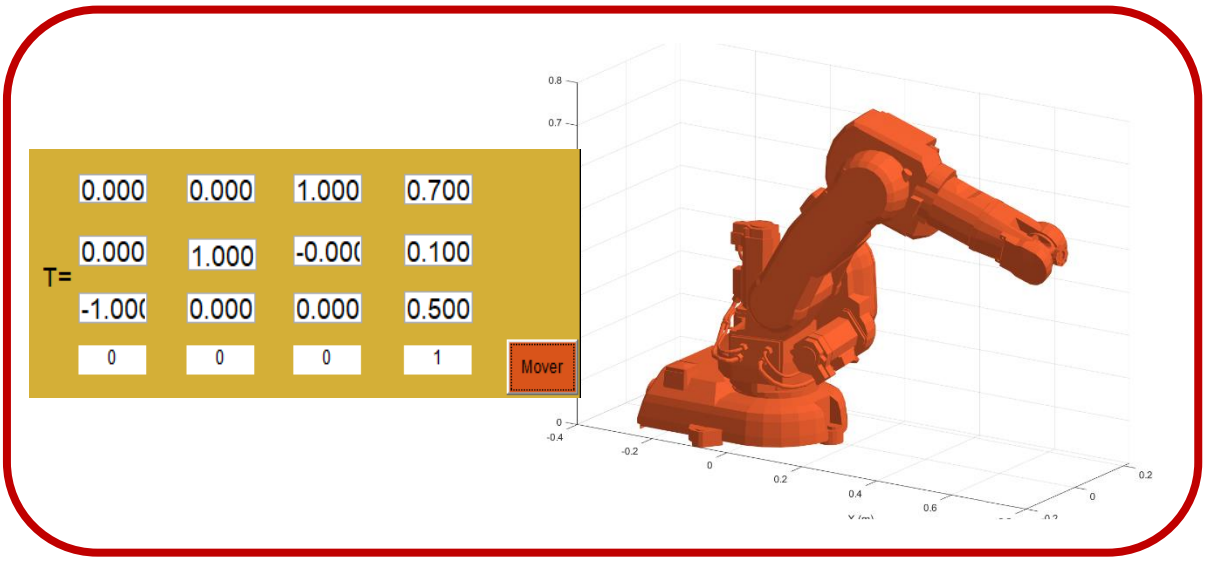


Ilustración 87. *Robot colocado en la posición y orientación dada*

Por lo tanto, la **matriz T** que define la posición y orientación es conocida.

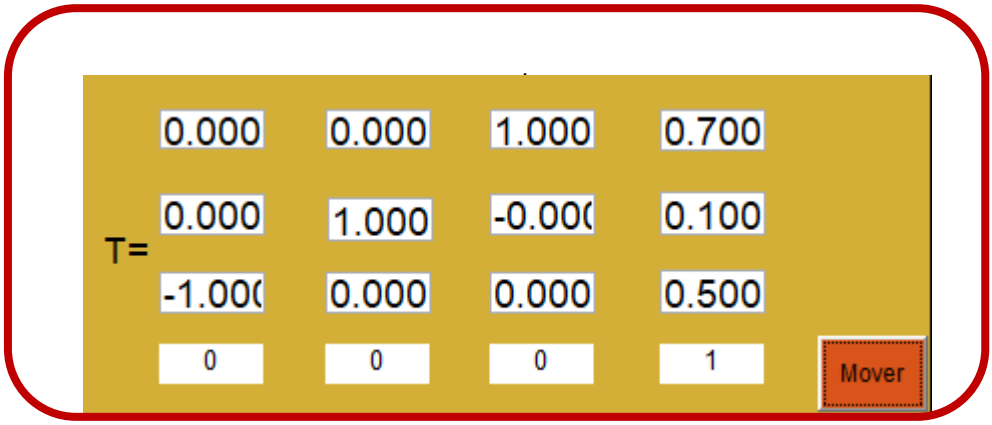


Ilustración 88. *Matriz de Transformación Homogénea conocida.*

Pero para saber el valor de las articulaciones que deben tener para llegar a esa posición y orientación se presionó el botón **“Obtener”** de la aplicación y nos lanzó 8 posibles soluciones para llegar a ese punto.

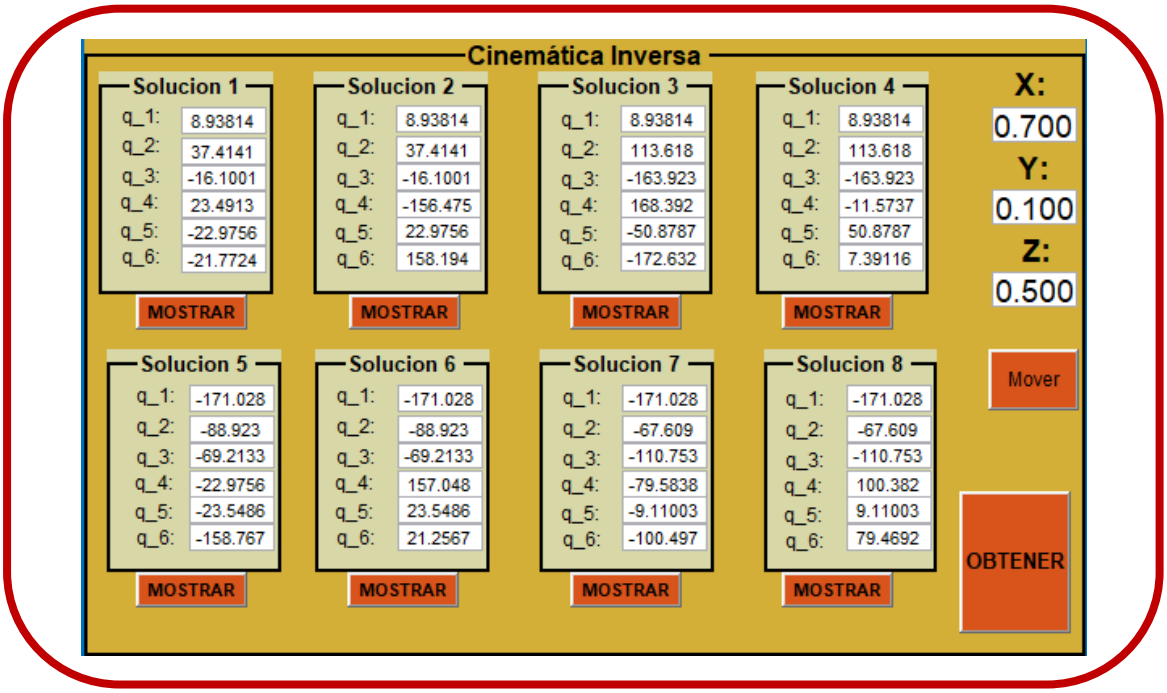


Ilustración 89. *Posibles Soluciones de la cinemática inversa*

Se validaron los resultados de la aplicación con las ecuaciones que se obtuvieron en el **capítulo 2.4.2** para obtener **q1, q2, q3, q4, q5** y **q6.**

Como se observó en el **capítulo 2.4.2** para hacer el análisis cinemático inverso de un robot que cuenta con una muñeca de Euler, primero se encuentra el valor de las primeras 3 articulaciones para posicionar al robot, esto se hizo tomando como referencia el punto de la muñeca.

Se tomaron los valores del robot de acuerdo a las especificaciones del creador **ilustración 91**.

Lo primero que se realizó fue encontrar el punto de la muñeca en coordenadas de la base.

Se tomó la ecuación **(22)** del **capítulo 2.4.1**:

$$\overrightarrow{p_m} = \vec{p} - L_6 \overrightarrow{z_6} \ldots (85)$$

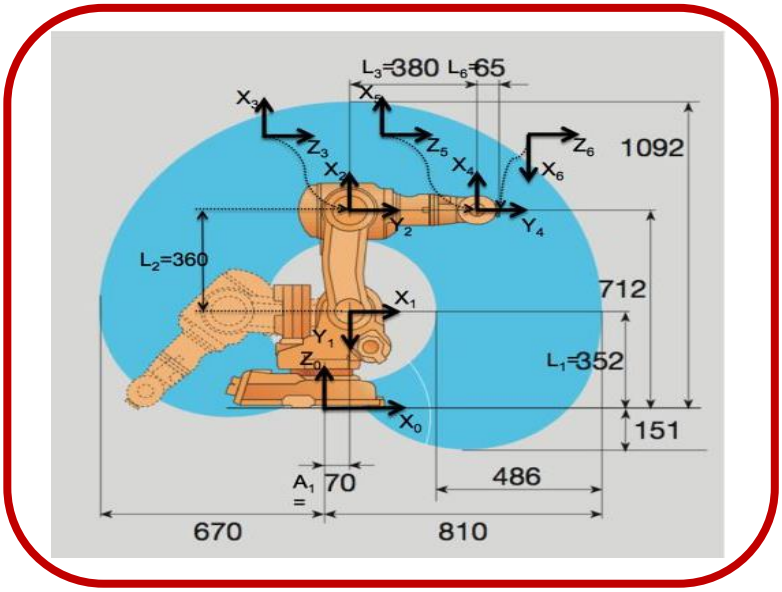


Ilustración 90. *Medidas del robot*

Donde:

$\overrightarrow{p_m}$: El punto de la muñeca en coordenadas de la base.

$\vec{p}$: Coordenadas del punto del efector final con respecto a la base. En este caso **(0.7, 0.1, 0.5).**

$L_6\,\overrightarrow{z_6}$: Longitud 6 **(0.065 m)** multiplicando a **z6** que son los elementos del eje z del efector final con respecto a la base **(1,0,0)**

Se sustituyó en ecuación **(85) y** se obtuvo:

$$\overrightarrow{p_m} = \begin{pmatrix} 0.7 \\ 0.1 \\ 0.5 \end{pmatrix} - 0.065 \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0.635 \\ 0.1 \\ 0.5 \end{pmatrix} \ldots (86)$$

Para encontrar **q1** se utilizó la ecuación **(24):**

$$q1 = arctan\left(\frac{Pmy}{Pmx}\right) = arctan\left(\frac{0.1}{0.635}\right) = 8.95° \ldots (87)$$

Se tomo como referencia las **ilustraciones 46** y **47**, el robot puede llegar al punto con una configuración codo arriba o codo abajo por lo cual para **q2** y **q3** tienen dos posibles soluciones. Pero para este paso se tuvo que encontrar el punto de la muñeca con respecto al sistema de referencia 1.

Se utilizó la ecuación **(26).**

$$\overrightarrow{p_m}^1 = (A_1^0)^{-1} * \overrightarrow{p_m} \ldots (88)$$

Donde:

$\overrightarrow{p_m}$: Es el punto de la muñeca en coordenadas de la base.

$\overrightarrow{p_m}^1$: Es el punto de la muñeca con respecto al sistema de referencia 1.

$(A_1^0)^{-1}$: Es la inversa de la matriz de transformación entre el sistema 1 y el sistema 0.

Como ya se sabe el valor de la articulación 1, se puede obtener la matriz de transformación entre el sistema 1 y el sistema 0, con los parámetros que se obtuvieron en el **capítulo 2.4.1**.

Tabla de parámetros Denavit-Hartenberg Robot IRB-140				
Sistema	θi	di (m)	ai (m)	αi
1	θ_1=8.95	0.352	0.070	-90°

Donde:

αi, ai y di : Son constantes

$\boldsymbol{\theta_1}$: Es el valor de **q1** que se obtuvo anteriormente (8.95°)

Se sustituyeron los valores en ecuación **(7)** del **capítulo 2.4.1**

$$^0\mathbf{A_1} = \begin{bmatrix} C(8.95) & -C(-90)S(8.95) & S(-90)S(8.95) & 0.070C(8.95) \\ S(8.95) & C(-90)C(8.95) & -S(-90)C(8.95) & 0.070S(8.95) \\ 0 & S(-90) & C(-90) & 0.352 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (89)$$

$$^{0}\mathbf{A}_{1} = \begin{bmatrix} 0.9878 & 0 & -0.1556 & 0.070 \\ 0.1556 & 0 & 0.9878 & 0.0109 \\ 0 & -1 & 0 & 0.352 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (90)$$

Se obtuvo la inversa de matriz **(90)**.

$$(^{0}\mathbf{A}_{1})^{-1} = \begin{bmatrix} 0.9878 & 0 & 0.1556 & -0.070 \\ 0 & 0 & 0 & 0.352 \\ 0.1556 & -1 & 0.9878 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (91)$$

Se multiplicó la matriz **(91)** por el vector homogéneo de la muñeca en coordenadas de la base **(86).**

$$\overrightarrow{p_m}^{1} = \begin{bmatrix} 0.9878 & 0 & 0.1556 & -0.070 \\ 0 & 0 & 0 & 0.352 \\ 0.1556 & -1 & 0.9878 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{pmatrix} 0.635 \\ 0.1 \\ 0.5 \\ 1 \end{pmatrix} = \begin{pmatrix} 0.572813 \\ -0.148 \\ -0.00026 \\ 1 \end{pmatrix} \ldots (92)$$

Una vez que se halló el punto de la muñeca con respecto al sistema 1, se utilizó las ecuaciones **27** y **28** para hallar las dos posibles soluciones, codo arriba y codo abajo en la articulación 2.

$$q2 = 90^{0} - arccos\left(\frac{L_2^2 + r^2 - L_3^2}{2rL_2}\right) - arctan\left(\frac{Pmy}{Pmx}\right) \ldots \text{Para codo arriba}$$

$$q2^{*} = 90^{0} - arccos\left(\frac{L_2^2 + r^2 - L_3^2}{2rL_2}\right) - arctan\left(\frac{Pmy}{Pmx}\right) \ldots \text{Para codo abajo}$$

Se calculó r:

$$r = \sqrt{p_{mx}^2 + P_{my}^2} = \sqrt{(0.572813)^2 + (-0.148)^2} = 0.5162 \ldots (93)$$

Se sustituyó en las ecuaciones **(27)** y **(28):**

$$q2 = 90^{0} - arccos\left(\frac{(0.360)^2 + (0.59162)^2 - (0.380)^2}{2(0.59162)(0.360)}\right) - arctan\left(\frac{-0.148}{0.5728}\right)$$

$$q2 = 90^0 - 38.1 - (-14.49) \ldots (94)$$

El valor de $q2$**=-14.49,** cambia a positivo porque el ángulo se encuentra en el cuarto cuadrante.

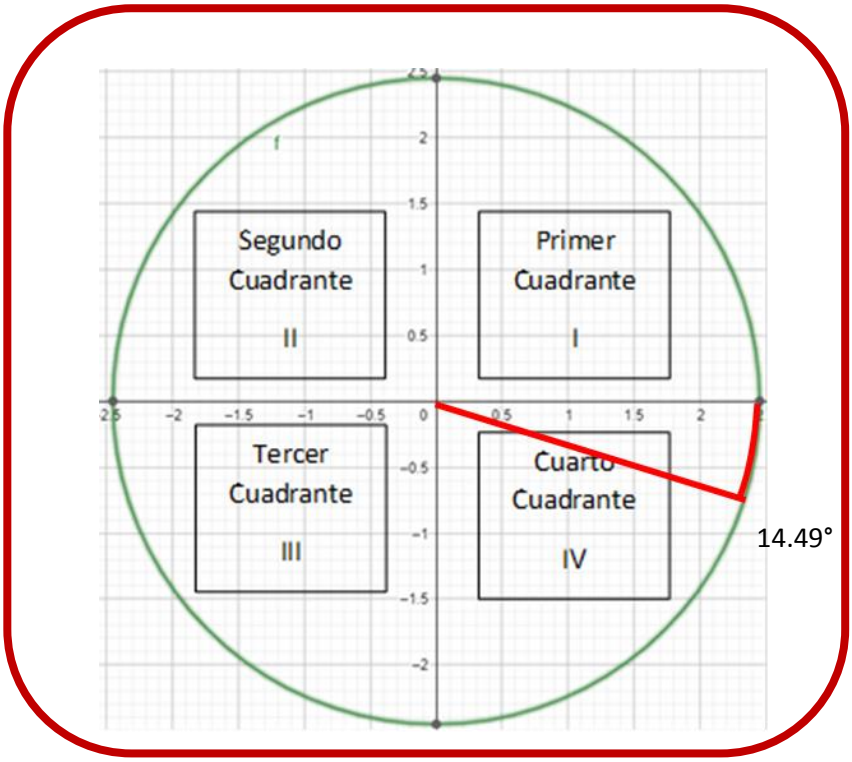


Ilustración 91. *Angulo en el cuarto cuadrante*

Por lo tanto:

$$q2 = 90^0 - 38.1 - (14.49)$$

$$q2 = 37.41 \ldots (95)$$

Para la configuración codo abajo:

$$q2^* = 90^0 + 38.1° - 14.49°$$

$$q2^* = 113.61° \ldots (96)$$

Se realizó lo mismo para obtener el valor de la articulación 3 pero ahora se tomó las ecuaciones **(39)** y **(40)**

$$q3 = 90^0 - \ arccos\left(\frac{L_2^2 + L_3^2 - r^2}{2L_2L_3}\right) \text{... Codo arriba}$$

$$q3^* = arccos\left(\frac{L_2^2 + L_3^2 - r^2}{2L_2L_3}\right) - 270° \text{... Codo abajo}$$

Una vez que se halló el valor de las primeras tres articulaciones, se encontró las soluciones de **q4, q5, q6**, después se compararon con los resultados de la **primera solución** que lanzó la aplicación. Para esto se ocupó el método algebraico que se utilizó en el **capítulo 2.4.1.**

Como se sabe:

$$T = A_1^0 * \ A_2^1 * \ A_3^2 * \ A_4^3 * A_5^4 * A_6^5$$

La matriz **T** está dada por la multiplicación de todas las transformaciones desde el sistema 0 al sistema 6. En este caso se conoce T, y como ya se obtuvo el valor de las primeras tres articulaciones también se conocen $A_1^0 * \ A_2^1 * \ A_3^2$, por lo cual:

$$(A_3^2)^{-1} * (A_2^1)^{-1} * (A_0^1)^{-1} * T = \ A_4^3 * A_5^4 * A_6^5$$

Se calculó las trasformaciones $A_{2,}^1 A_3^2$ con sus respectivos parámetros de Denavit Hartenberg.

Para A_2^1:

Sistema	θi	di (m)	ai (m)	αi
2	θ_2-90=37.41-90=-52.6°	0	0.360	0

$$^1A_2 = \begin{bmatrix} C(-52.6) & -C(0)S(-52.6) & S(0)S(-52.6) & 0.36C(-52.6) \\ S(-52.6) & C(0)C(-52.6) & -S(0)C(-52.6) & 0.36S(-52.6) \\ 0 & S(0) & C(0) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{... } (99)$$

$$^{1}A_{2} = \begin{bmatrix} 0.6074 & 0.7944 & 0 & 0.2187 \\ 0.7944 & 0.6074 & 0 & -0.286 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (100)$$

Para A_3^2:

Sistema	θi	Di (m)	ai (m)	αi
3	$\theta_{3=}-16.13$	0	0	-90°

$$^{2}A_{3} = \begin{bmatrix} C(-16.13) & -C(-90)S(-16.13) & S(-90)S(-16.13) & 0C(-16.13) \\ S(-16.13) & C(-90)C(-16.13) & -S(-90)C(-16.13) & 0S(-16.13) \\ 0 & S(-90) & C(-90) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (101)$$

$$^{2}A_{3} = \begin{bmatrix} 0.9606 & 0 & 0.2778 & 0 \\ -0.2778 & 0 & 0.9606 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (102)$$

Se obtuvo la matriz inversa de A_2^1, A_3^2

$$(^{1}A_{2})^{-1} = \begin{bmatrix} 0.6074 & -0.7944 & 0 & -0.36 \\ 0.7944 & 0.6074 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (103)$$

$$(^{2}A_{3})^{-1} = \begin{bmatrix} 0.9606 & -0.2778 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0.2778 & 0.9606 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (104)$$

Se multiplicó todas las matrices en orden y se obtuvo:

$$({}^{2}A_{3})^{-1} * ({}^{1}A_{2})^{-1} = T1$$

$$\begin{bmatrix} 0.9606 & -0.2778 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0.2778 & 0.9606 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 0.6074 & -0.7944 & 0 & -0.36 \\ 0.7944 & 0.6074 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

$$\begin{bmatrix} 0.3629 & -0.9318 & 0 & -0.3458 \\ 0 & 0 & -1 & 0 \\ 0.9318 & 0.3629 & 0 & -0.1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T1 * ({}^{0}A_{1})^{-1} = T2$$

$$\begin{bmatrix} 0.3629 & -0.9318 & 0 & -0.3458 \\ 0 & 0 & -1 & 0 \\ 0.9318 & 0.3629 & 0 & -0.1 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 0.9878 & 0 & 0.1556 & -0.070 \\ 0 & 0 & 0 & 0.352 \\ 0.1556 & -1 & 0.9878 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

$$\begin{bmatrix} 0.3585 & 0.0565 & 0.9318 & -0.6992 \\ 0.1556 & -0.9878 & 0 & 0 \\ 0.9205 & 0.1450 & -0.3629 & -0.0375 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T2 * T = Q$$

$$\begin{bmatrix} 0.3585 & 0.0565 & 0.9318 & -0.6992 \\ 0.1556 & -0.9878 & 0 & 0 \\ 0.9205 & 0.1450 & -0.3629 & -0.0375 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 0 & 0 & 1 & 0.700 \\ 0 & 1 & 0 & 0.100 \\ -1 & 0 & 0 & 0.500 \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

$$Q = \begin{bmatrix} -0.9318 & 0.0565 & 0.3585 & 0.0233 \\ 0 & -0.9878 & 0.1556 & 0.0101 \\ 0.3629 & 0.1450 & 0.9205 & 0.4399 \\ 0 & 0 & 0 & 1 \end{bmatrix} \ldots (105)$$

La matriz Q **(105)** se igualó con la multiplicación entre $\boldsymbol{A_4^3}$, $\boldsymbol{A_5^4}$, $\boldsymbol{A_6^5}$, y la que se obtuvo en el **capítulo 2.4.2** la matriz **(46).**

$$Q = \begin{bmatrix} -C(q4)c(q5)c(c6) + s(q4)s(q6) & c(q4)c(q5)s(q6) + s(q4)c(q6) & -c(q4)s(q5) & -L_6c(q4)s(q5) \\ -s(q4)c(q5)c(c6) - c(q4)s(q6) & s(q4)c(q5)s(q6) - c(q4)c(q6) & -s(q4)s(q5) & -L_6s(q4)s(q5) \\ -s(q5)c(q6) & s(q5)s(q6) & C(q5) & L_6c(q5) + L_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Donde **Q**:

$$Q = \begin{bmatrix} Q11 & Q12 & Q13 & Q14 \\ Q21 & Q22 & Q23 & Q24 \\ Q31 & Q32 & Q33 & Q34 \\ Q41 & Q42 & Q43 & Q44 \end{bmatrix} \ldots (106)$$

La igualdad quedó de la siguiente manera:

$$\begin{bmatrix} -0.9318 & 0.0565 & 0.3585 & 0.0233 \\ 0 & -0.9878 & 0.1556 & 0.0101 \\ 0.3629 & 0.1450 & 0.9205 & 0.4399 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -C(q4)c(q5)c(c6) + s(q4)s(q6) & c(q4)c(q5)s(q6) + s(q4)c(q6) & -c(q4)s(q5) & -L_6c(q4)s(q5) \\ -s(q4)c(q5)c(c6) - c(q4)s(q6) & s(q4)c(q5)s(q6) - c(q4)c(q6) & -s(q4)s(q5) & -L_6s(q4)s(q5) \\ -s(q5)c(q6) & s(q5)s(q6) & C(q5) & L_6c(q5) + L_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\ldots (107)$$

Como se vio en el **capítulo 2.4.1** Existe una condición que dice: si $\mathbf{Q_{33} = \cos(q5) \approx 0}$ entonces será una solución redundante. Por lo cual existirán infinitas soluciones para **q4** y **q6**. Pero en este caso $\mathbf{Q_{33} > 0}$ por lo cual no es una solución redundante y se encontró **q4** y **q6** con ecuaciones **(50)** y **(51)**

$$q_4 = arctan\left(\frac{Q_{23}}{Q_{13}}\right)$$... Solución para q4

$$q_6 = arctan\left(\frac{-Q_{32}}{Q_{13}}\right)$$..... solución para q6

Se sustituyó los valores de la matriz Q:

$$q_4 = arctan\left(\frac{0.1556}{0.3585}\right) = 23.46° \ldots (108)$$

$$q_6 = arctan\left(\frac{-(0.1450)}{0.3629}\right) = -21.78° \ldots (109)$$

Se halló **q5** con la ecuación **(56)**

$$\frac{\sin(q5)}{\cos(q_5)} = \tan(q5)$$

Como ya se conoce **cos(q5) =Q_{33}**, se obtuvo **sin(q5)** con ecuación **(55)**

$$\left(\frac{Q_{32}}{(sen(q6)}\right) = sen(q5)$$

Se sustituyó los valores de la matriz **Q** y de **q6**

$$sen(q5) = \left(\frac{0.1450}{(sin(-21.78)}\right) = -0.39078° \ldots (110)$$

Se sustituyó **sin(q5)** y **cos(q5)** en ecuación **(56)**

$$q_5 = arctan\left(\frac{-0.39078}{0.9205}\right) = -23°$$

El valor de las articulaciones que se obtuvieron con las ecuaciones del **capítulo 2.4.2** son:

Valor de las articulaciones					
q1	**q2**	**q3**	**q4**	**q5**	**q6**
8.95°	**37.41°**	**-16.13°**	**23.46°**	**-23°**	**-21.78°**

Tabla 1.14 *Articulaciones obtenidas con la teoría*

Se comparó las articulaciones obtenidas matemáticamente, con las de la solución 2 de la aplicación y se observó que se obtuvieron valores semejantes, esto debido a la perdida de milésimas en los cálculos. Ya que la aplicación toma muchos más decimales para realizar los cálculos, por lo tanto, tiene valores muchos más precisos.

Se comprobó las articulaciones que se obtuvieron matemáticamente con las de la primera solución de la aplicación 2 utilizando la aplicación 1.

Primero se introdujeron los parámetros de Denavit-Hartenber del robot.

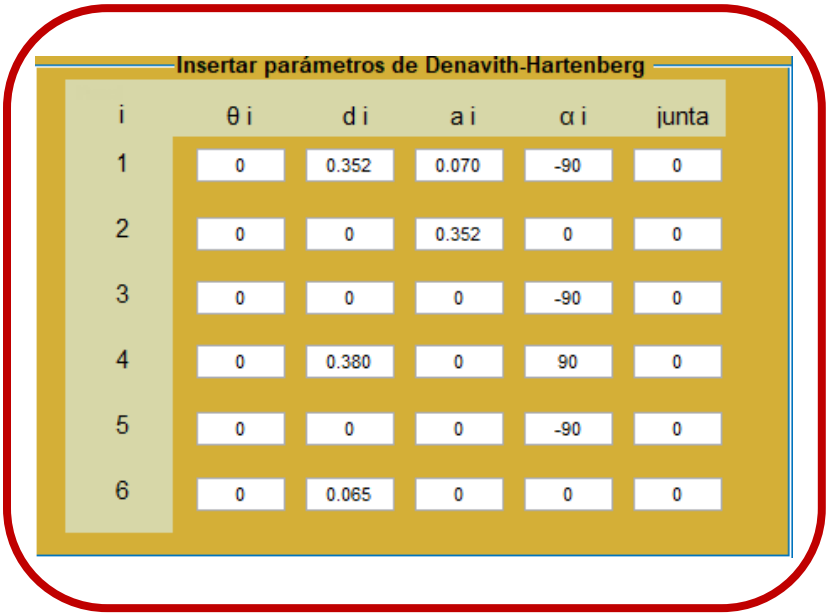


Ilustración 92. *Parámetros de D-H*

En el Slider se colocaron las articulaciones obtenidas matemáticamente utilizando las ecuaciones de la teoría.

Valor de las articulaciones					
q1	**q2**	**q3**	**q4**	**q5**	**q6**
8.95°	**37.41°-90=52.59**	**-16.13°**	**23.46°**	**-23°**	**-21.78°+180=158.23**

Ilustración 93. *Matriz de Transformación Homogénea con articulaciones obtenidas matemáticamente*

También se colocaron en el Slider el valor as articulaciones de la solución 1 de la aplicación 2

Valor de las articulaciones					
q1	**q2**	**q3**	**q4**	**q5**	**q6**
8.93814°	**37.4141°-90=52.5859**	**-16.1001°**	**23.4613°**	**-22.9756°**	**-21.7724°+180=158.2276**

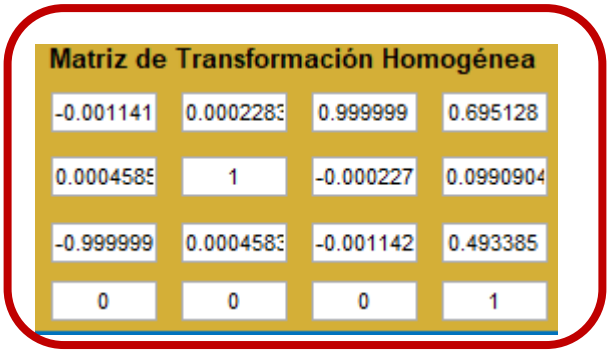


Ilustración 94. *Matriz de Transformación Homogénea con las articulaciones obtenidas con la aplicación 2*

Se notó que las articulaciones que se obtuvieron con la aplicación 2 son más precisas acercándose más las a la matriz (T) deseada.

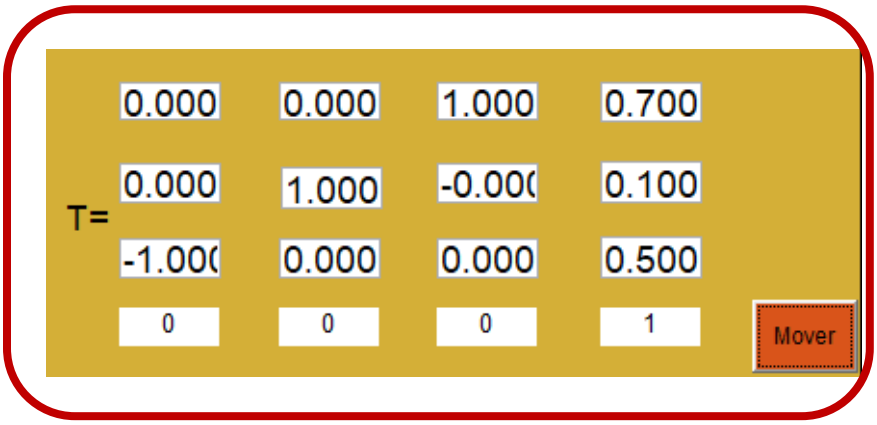


Ilustración 95. *Matriz de Transformación Homogénea deseada.*

Pero la aplicación 2 redondea los valores al momento de insértalos en los “sliders” de cada articulación.

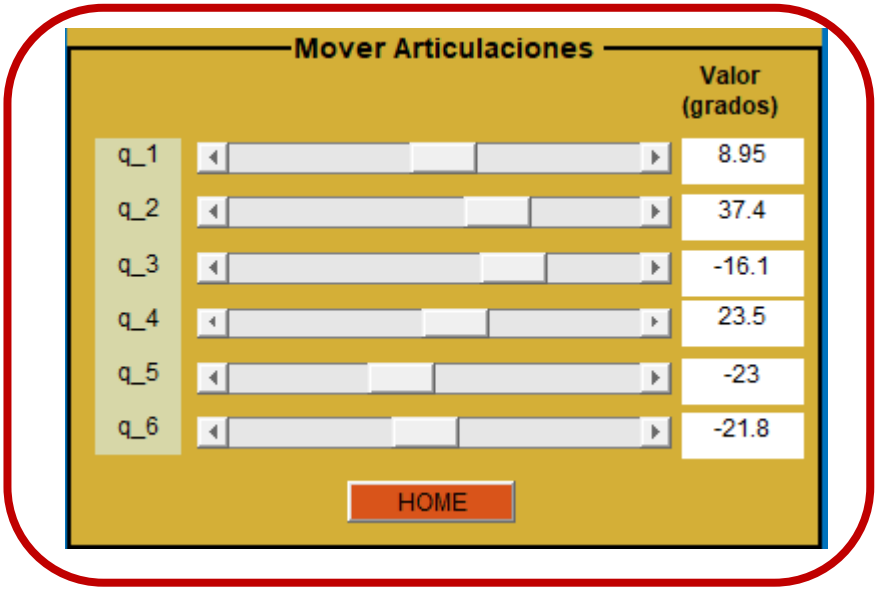


Ilustración 96. *Matriz de Transformación Homogénea deseada.*

Por lo tanto, la segunda aplicación encuentra de manera rápida y precisa la cinemática inversa del robot IRB-140. La ventaja de usar la aplicación es que no solamente encuentra una solución si no 7 soluciones más para cada punto que se desee, esto sirve para demostrar en las clases de Robótica, que el robot puede llegar a un lugar, no solo con una configuración, si no con varias.

5. CAPÍTULO 5: PRUEBA DE LOS SIMULADORES CLASE EN LINEA

Alumnos de la Carrera Ingeniería Mecánica en la FES Aragón y de Ingeniería Mecatrónica en Ciudad Universitaria, llevan en común varias materias, una de ellas es la robótica. Los compañeros de la Facultad de Ingeniería la cursan a partir del noveno semestre mientras que en la Facultad de Estudios Superiores Aragón la estudian a partir de séptimo semestre para alumnos con especialidad en Biomecánica y optativa para alumnos de otros módulos. [25] [27]

El programa de estudio de la materia de robótica en ambas facultades se imparten los temas de cinemática directa e inversa. [26] [28]

Ambas carreras se vieron afectadas por la contingencia sanitaria provocada por el virus SARS-COV-2 por lo cual ambas fueron impartidas en línea por el Dr. en Mecatrónica Patricio Martínez Zamudio.

Para poner a prueba las aplicaciones en las clases en línea, se formaron dos grupos de estudio. El primero consto de 14 personas de la Carrera Ingeniería Mecatrónica y el segundo estuvo formado de 12 personas de la Carrera Ingeniería Mecánica. Ambos grupos cursaron la materia en línea por lo cual ambos ya tenían conocimiento de los temas de la cinemática directa e inversa. Ambos fueron elegidos para comparar cual era la mejor manera de aprender, con o sin simuladores virtuales.

Para observar que tanto habían aprendido durante el transcurso del semestre con las herramientas que les brindo el Dr. Patricio Martínez en los temas ya antes mencionados, se realizó un ejercicio propedéutico el cual consistía en obtener la cinemática directa con parámetros de Denavit-Hartenberg y la cinemática inversa del robot IRB-140 (Véase **Apéndice 11** para observar el ejercicio propedéutico realizado).

Se dio un tiempo de realización del ejercicio de 1 hora 30 minutos. Se tomaron 5 criterios a evaluar (la tabla 1.15 muestra los criterios a evaluar y el puntaje en cada uno).

Criterios por evaluar	Puntaje		
Definición de la cinemática directa	Definió correctamente el concepto (2 puntos)	Definió el concepto, pero no del todo correcto (1 punto)	Definió incorrectamente o no realizó este punto (0 puntos)
Definición de la cinemática inversa	Definió correctamente el concepto (2 puntos)	Definió el concepto, pero no del todo correcto (1 punto)	Definió incorrectamente o no realizó este punto (0 puntos)
Tabla de Parámetros de Denavit-Hartenber.	Encontró todos los parámetros de Denavit Hartenber (2 puntos)	Encontró los parámetros, pero no le todo correctos (1 punto)	No encontró ningún parámetro o no realizó el ejercicio (0 puntos)
Matriz de Transformación Homogénea que define la posición y orientación del efector final	Encontró correctamente la matriz (2 puntos)	Encontró la matriz, pero no del todo correcta (1 punto)	No realizó el ejercicio (0 puntos)
Cinemática Inversa	Encontró la cinemática inversa del robot (2 puntos)	Encontró la cinemática inversa del robot, pero no del todo correcta (2 puntos)	No realizó el ejercicio

Tabla 1.15. *Criterios para evaluar ejercicio propedéutico*

Los puntos para una evaluación perfecta eran de 10 como máximo. A continuación, se presentan los resultados obtenidos en ambos grupos:

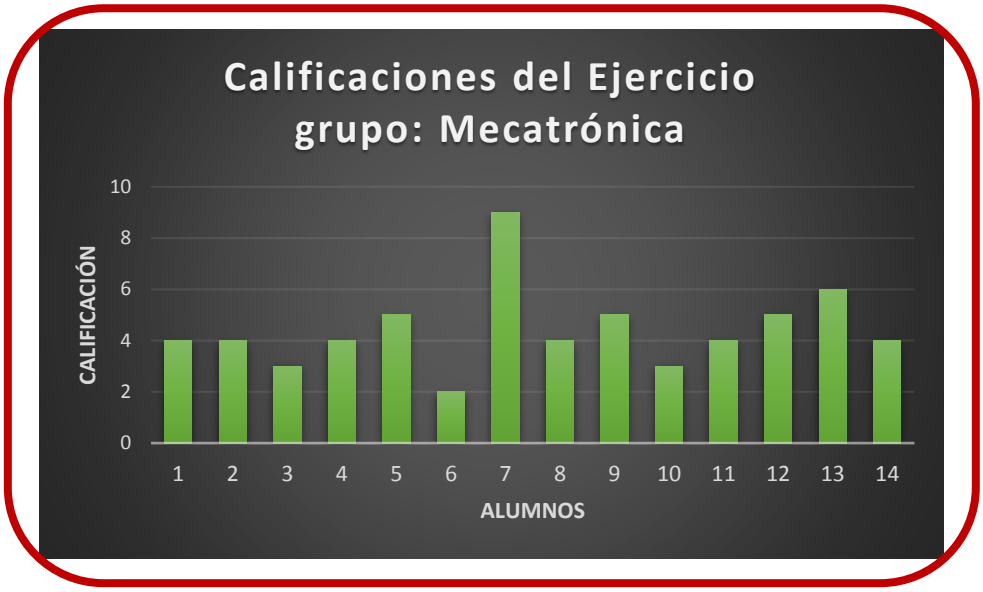


Ilustración 97. *Calificaciones del grupo de mecatrónica*

Se observó que los alumnos del grupo de mecatrónica obtuvieron calificaciones reprobatorias en la mayoría de sus integrantes, solamente hubo dos personas que pasaron el ejercicio propedéutico con calificación aprobatoria, una de ellas obtuvo un nueve como calificación y la otra obtuvo una calificación baja pero aprobatoria de seis. Los demás compañeros se quedaron por debajo del puntaje mínimo para aprobar (6 puntos).

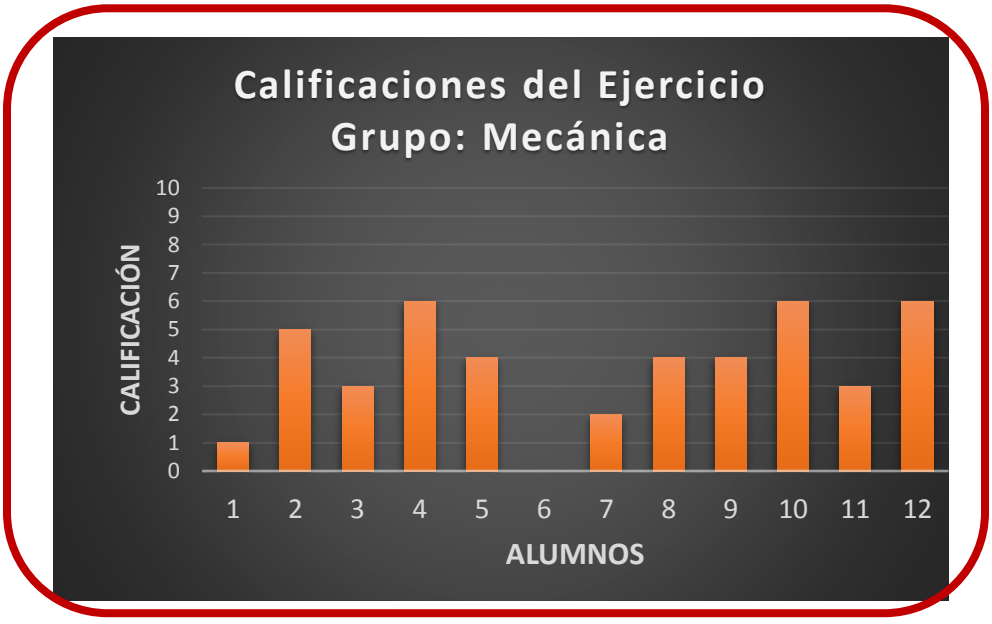


Ilustración 98. *Calificaciones del grupo de mecánica*

Por otro lado, el grupo de Ingeniería Mecánica pasaron tres personas, pero todas con una calificación aprobatoria baja (6 puntos), el resto permaneció por debajo del puntaje.

En ambos grupos se observó que la teoría y los conceptos los tienen claros, porque sabían de lo que se trataba el ejercicio y de lo que tenían que hacer. Pero tuvieron muchos problemas al encontrar la tabla de los parámetros de Denavit-Hartenberg, hubo casos en los que no alcanzaron a realizar este punto, otros llegaron a la tabla, pero no del todo correcta, se equivocaban mucho a la hora de obtener el giro **alfa(α)** en el **eje Xi** para cada escalón, dando signos contrarios a los que realmente eran y si estos parámetros son incorrectos, al encontrar la matriz de transformación homogénea dará como resultado una incorrecta. Fue tanto el tiempo que tardaron en realizar el ejercicio de cinemática directa, que la mayoría de los integrantes no alcanzó a realizar el ejercicio de la cinemática inversa, los pocos que la realizaron se apoyaron de un código que el profesor les enseño durante el curso.

Al finalizar el ejercicio se les pregunto a los alumnos por qué no pudieron resolver el ejercicio correctamente o que fue lo que les hizo tardarse tanto. Se notó que la mayoría de ellos se confunden mucho a la hora de obtener los parámetros de Denavit- Hartenber porque no logran visualizar los movimientos del robot pues no cuentan con una vista 3D. Por lo tanto, la mayoría colocaba los ejes de referencia de cada sistema de una manera incorrecta. Y fue tanto el tiempo que ocuparon realizando la cinemática directa que se olvidaron en realizar la cinemática inversa.

Para observar si los simuladores desarrollados en esta tesis ayudan a resolver los problemas de visualización, también, si permiten obtener la cinemática directa e inversa más rápida y clara a como la obtenían en clases, y además para ver si son una buena herramienta para las clases en línea. Se compartió por medio de la plataforma virtual Google Classroom un “link” a cada integrante de ambos grupos, con un video explicando como descargar e instalar las aplicaciones.

La clase del miércoles 20 de enero del 2021, se mostró a los alumnos como utilizar las aplicaciones desarrollando la cinemática directa e inversa del robot IRB 140.

Además, se les mostró todas las funciones de la aplicación y como todos los conceptos sobre estos temas están aplicados en los robots que la industria utiliza.

Al final de la clase los alumnos de ambos grupos desarrollaron un cuestionario con escala de Likert (Véase **apéndice 12** para observar el cuestionario realizado al grupo) donde:

- Muy de acuerdo son 5 puntos
- De acuerdo son 4 puntos
- Ni de acuerdo ni en desacuerdo 3 puntos
- En desacuerdo 2 puntos
- Muy en desacuerdo 1 punto

El cuestionario constó de 16 reactivos por lo cual la calificación más satisfactoria a obtener es de 80 puntos. La finalidad de este cuestionario fue evaluar a las aplicaciones y observar si son efectivas para enseñar cinemática directa e inversa.

Las siguientes gráficas circulares muestran los porcentajes para cada pregunta. Las primeras dos muestran si los alumnos aprendieron de una mejor manera los temas de cinemática directa e inversa con ayuda de las dos aplicaciones. Para el grupo de Ingeniería Mecánica el 58% del total de alumnos están muy de acuerdo y el 48% de ellos están de acuerdo. Para el grupo de Mecatrónica solamente del 13% de los alumnos no están de acuerdo ni en desacuerdo con que hayan aprendido mejor con las aplicaciones.

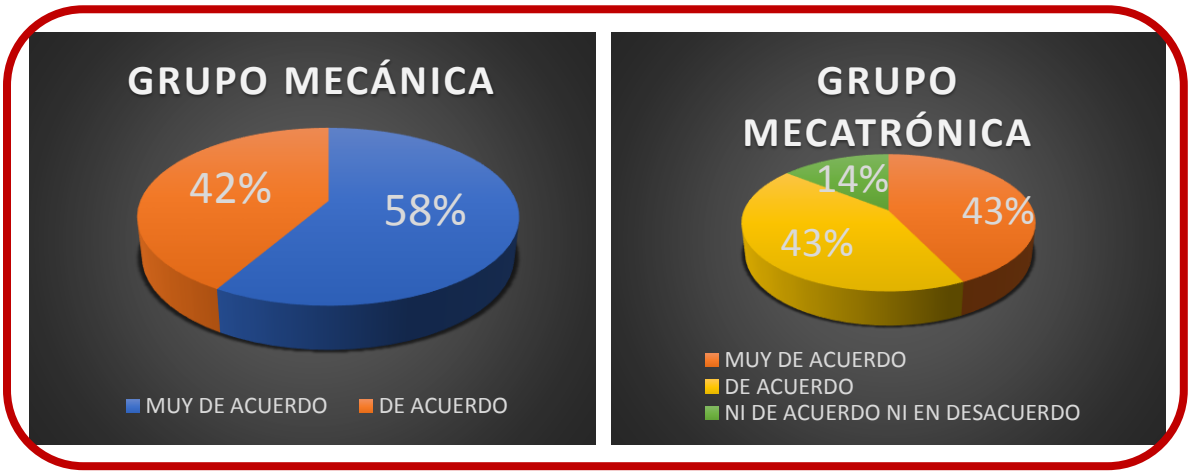


Ilustración 99. *Resultados pregunta 1*

Ambos grupos están de acuerdo y muy acuerdo con que las aplicaciones ayudan a obtener mejor los parámetros de Denavit-Hartenberg a como los obtenían en sus clases.

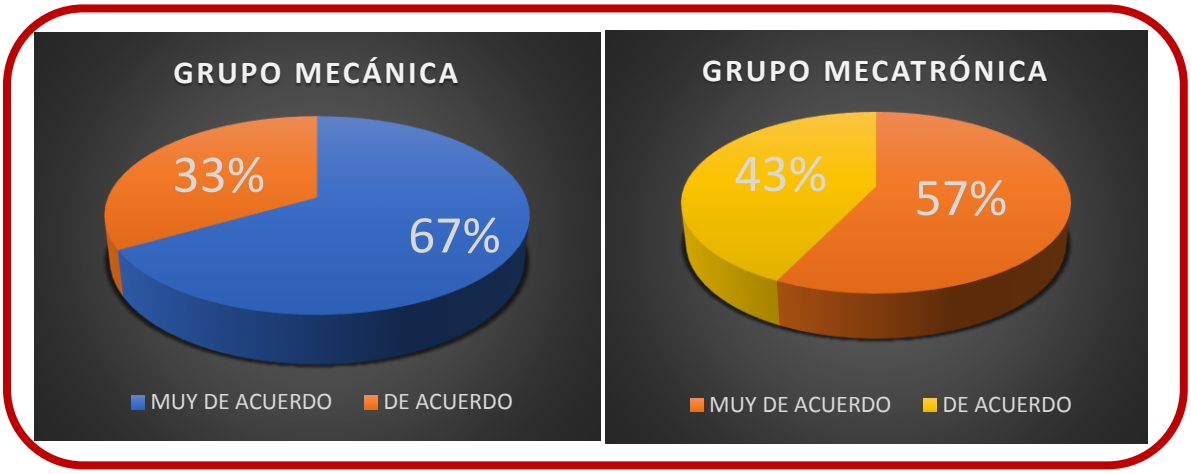


Ilustración 100. *Resultados pregunta 2*

Se les pregunto si las aplicaciones ayudan a reforzar los conocimientos de los temas de cinemática directa e inversa. A lo cual se obtuvieron buenos porcentajes en acuerdo y muy de acuerdo para ambos grupos, solo un pequeño porcentaje de ellos no están de acuerdo ni en desacuerdo.

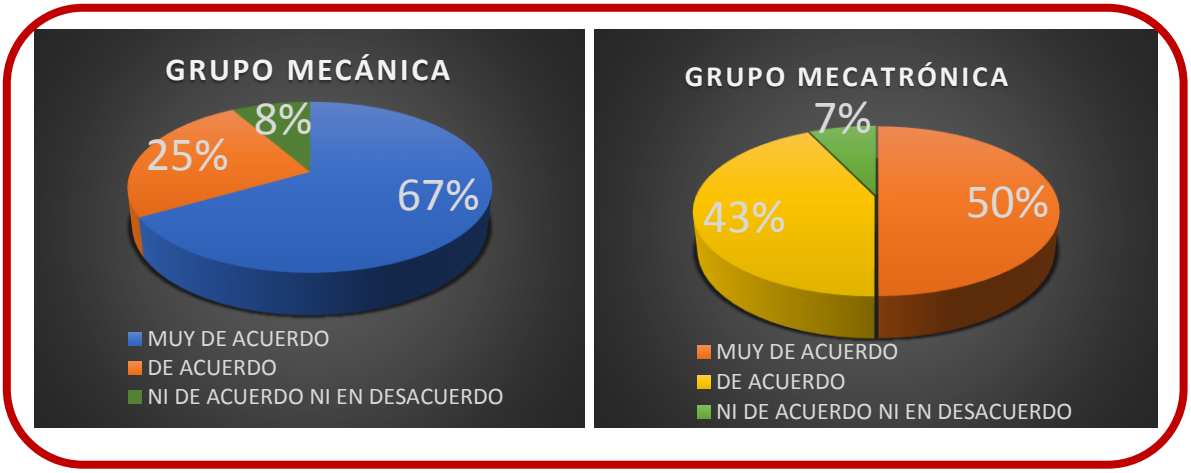


Ilustración 101. *Resultados pregunta 3*

Los resultados obtenidos en la pregunta 4 nos dice que las aplicaciones ayudaron a cubrir las dudas que se presentaron durante el transcurso del semestre ya que se obtuvieron buenos porcentajes en los incisos de acuerdo y muy de acuerdo en ambos grupos.

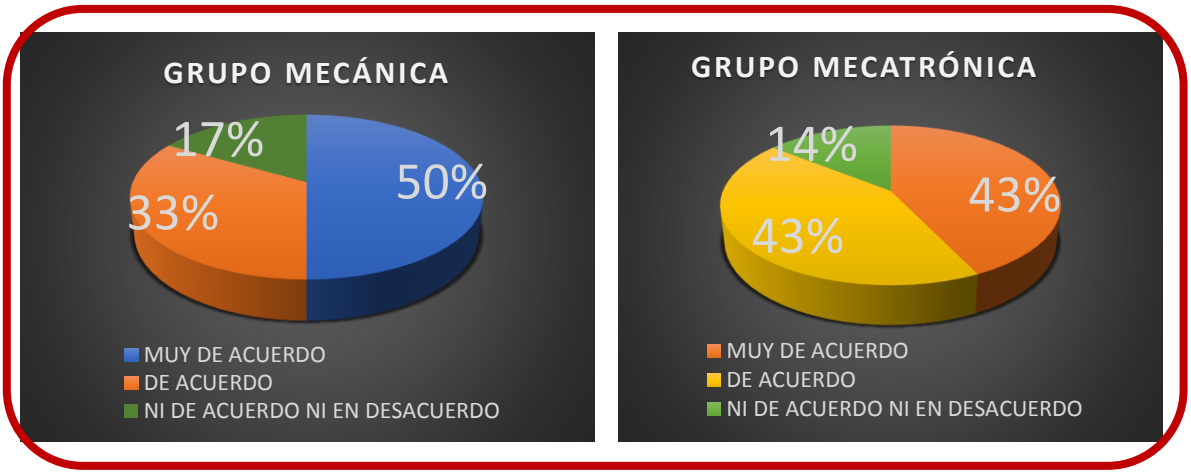


Ilustración 102. *Resultados pregunta 4*

Como ya se había mencionado anteriormente uno de los grandes problemas que tuvieron los alumnos a la hora de obtener los parámetros de Denavit-Hartenberg en el ejercicio propedéutico, fue que no lograron observar los movimientos ni los ejes de cada sistema. Se les pregunto, si con ayuda de las aplicaciones pudieron observar mejor los sistemas y los movimientos del robot. Se obtuvieron buenos porcentajes en ambos grupos estando de acuerdo y muy de acuerdo con esto.

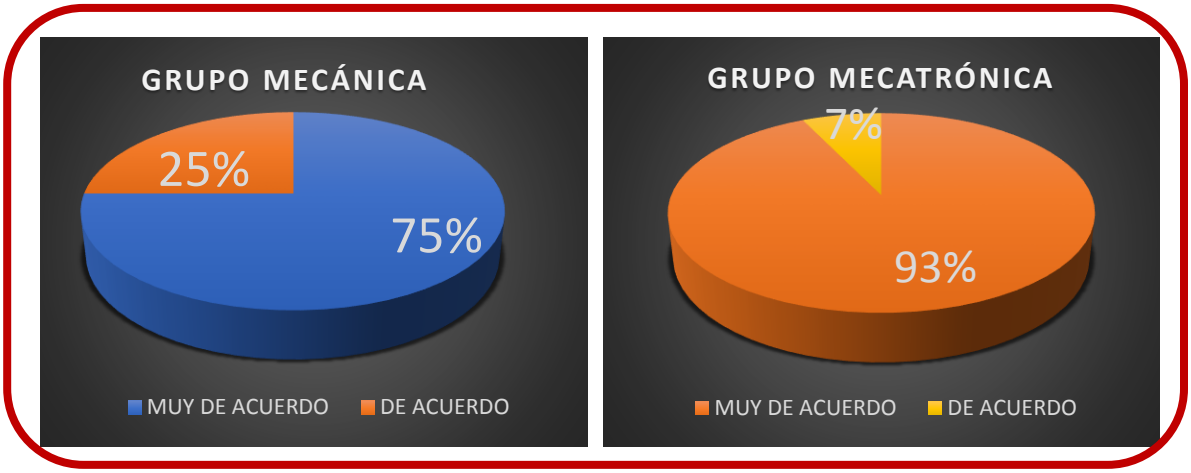


Ilustración 103. *Resultados pregunta 5*

Con los datos obtenidos en la pregunta 6, los alumnos están de acuerdo y muy de acuerdo con que la aplicación puede ser utilizada en las futuras generaciones para enseñar cinemática directa e inversa.

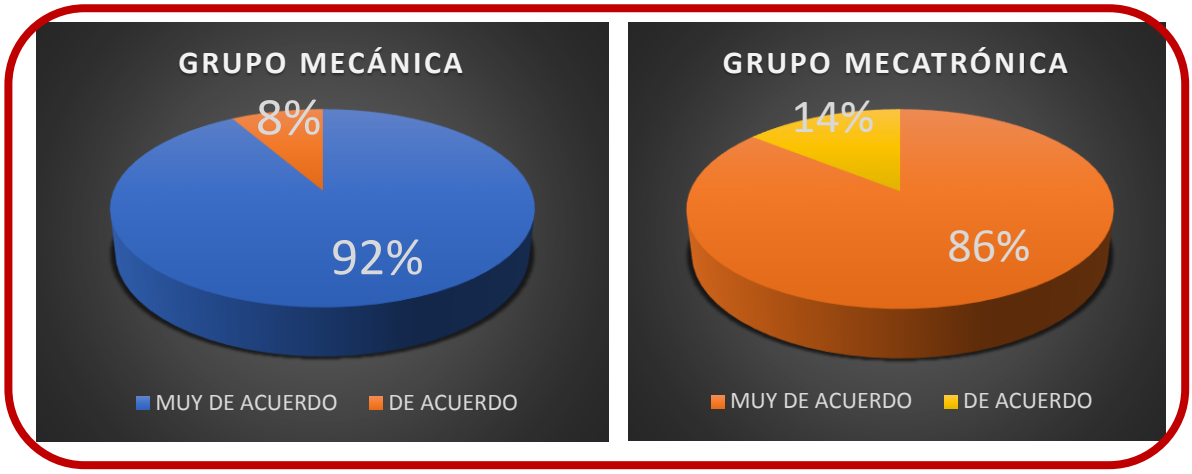


Ilustración 104. *Resultados pregunta 6*

Al igual que los resultados de la pregunta 6 los alumnos están de acuerdo y muy de acuerdo que no solamente estas aplicaciones pueden ser utilizadas en clases en línea si no también en clases presenciales.

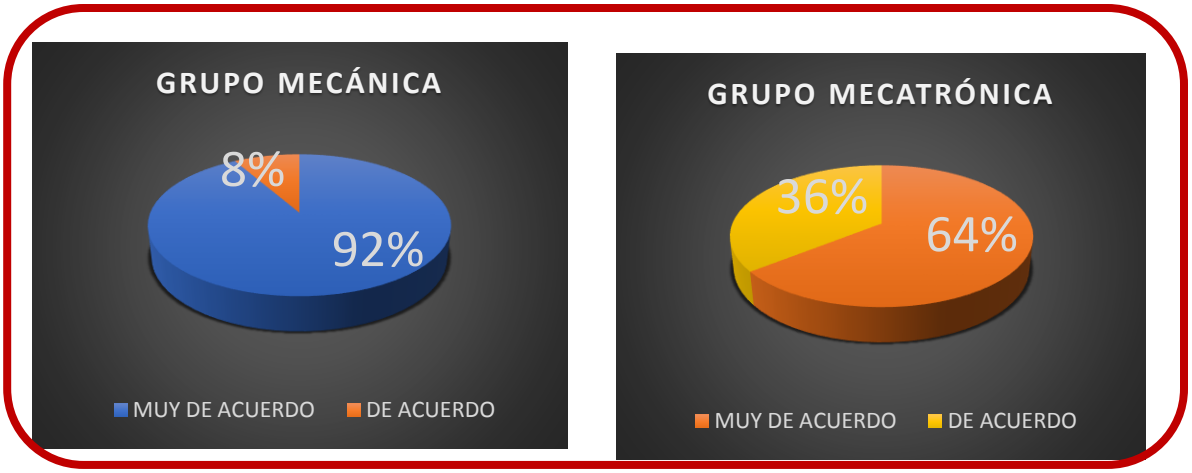


Ilustración 105. *Resultados pregunta 7*

Se observó que los alumnos estuvieron de acuerdo y muy acuerdo con la facilidad de usar las aplicaciones, por lo cual las aplicaciones son amigables y de fácil entendimiento para los que la utilicen. Pero siempre y cuando se tenga conocimiento previo a la materia de robótica.

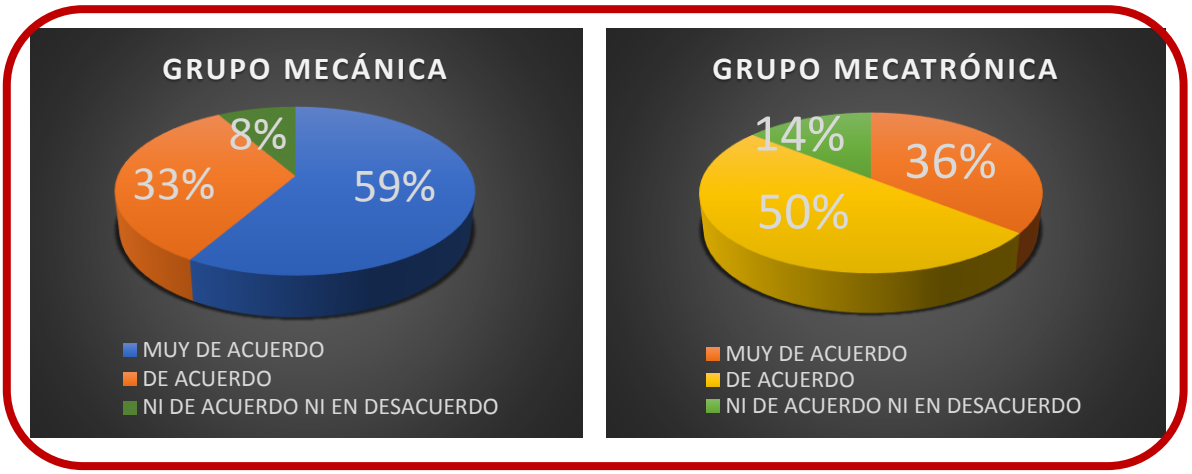


Ilustración 106. *Resultados pregunta 8*

Para ver si las aplicaciones fueron fáciles de instalar se les pregunto a los alumnos al respecto, lo cual para ambos grupos la mayoría estuvo de acuerdo y muy de acuerdo, pero para este punto hubo una mayor parte en el grupo de mecatrónica que tuvo problemas al instalar el software.

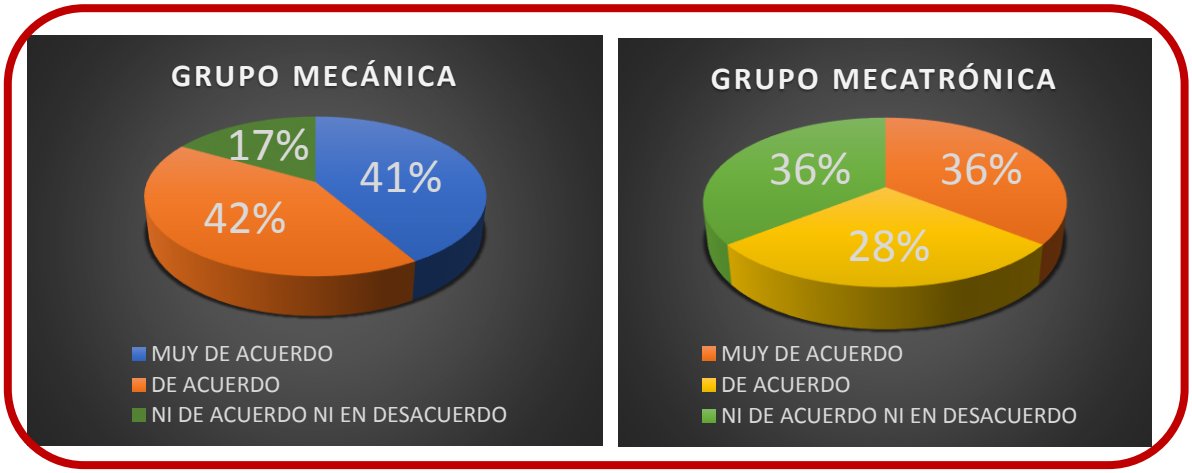


Ilustración 107. *Resultados pregunta 9*

La aplicación tuvo problemas al ejecutarse, una gran parte de ambos grupos, tardo en abrir la aplicación, esto debido al tipo de computadora con el que contaban, para los alumnos con una computadora con buen procesador y una memoria RAM de alta gama la aplicación se ejecutó rápido y sin tardarse.

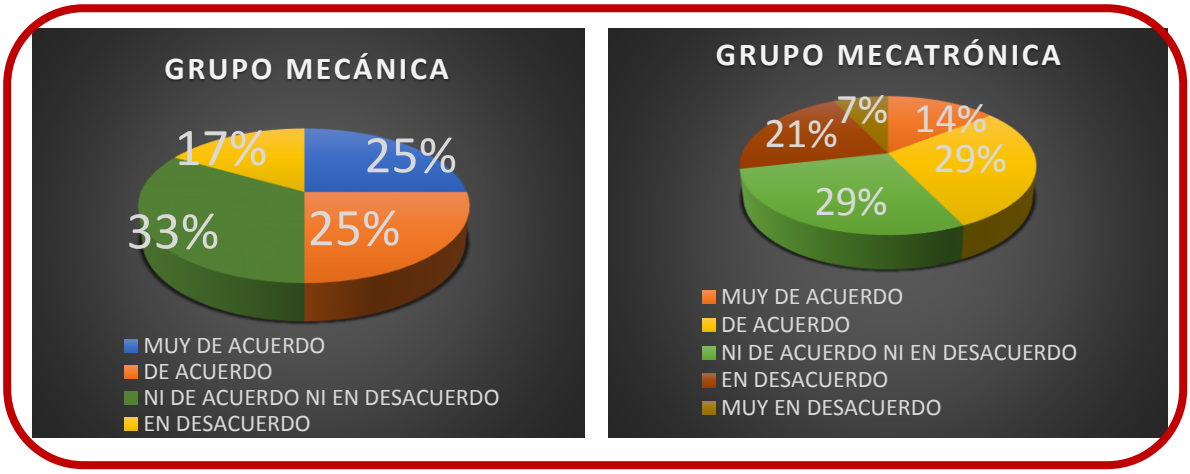


Ilustración 108. *Resultados pregunta 10*

Se observó en el cuestionario que los alumnos no contaron o utilizaron aplicaciones similares durante el transcurso del semestre. Hubo un gran porcentaje en los incisos de acuerdo y muy de acuerdo en este punto.

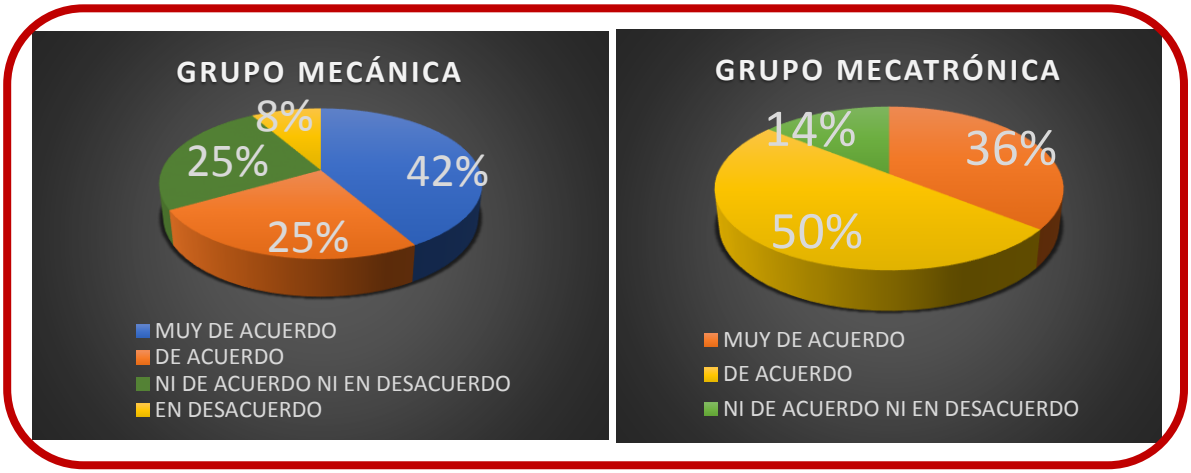


Ilustración 109. *Resultados pregunta 11*

Se notó en ambos grupos que obtener la cinemática inversa fue más rápido usando estas aplicaciones pues el porcentaje de alumnos están de acuerdo y muy de acuerdo con este punto.

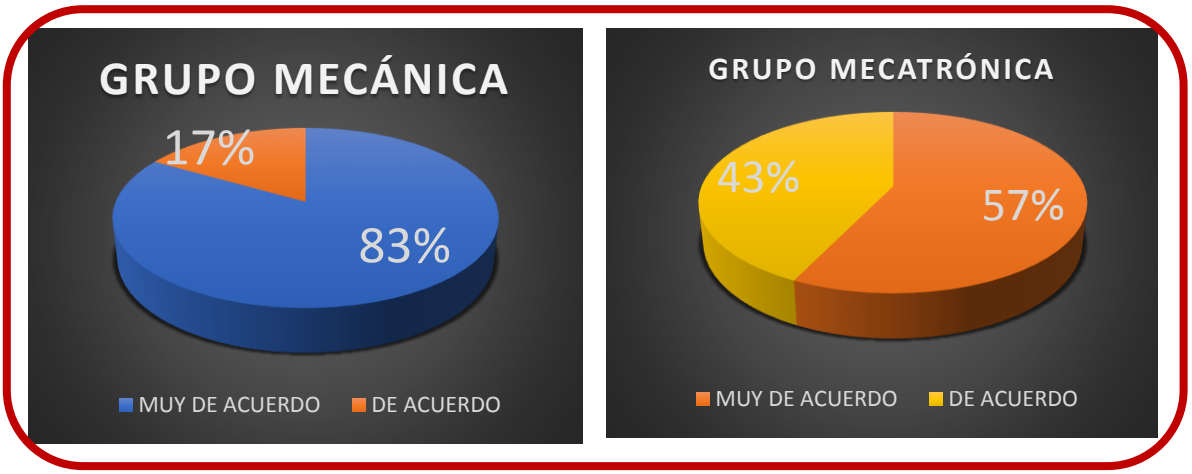


Ilustración 110. *Resultados pregunta 12*

Para observar si esta clase de herramientas despierta el interés a la asignatura en los alumnos, se observó, que en el grupo de Mecánica la mayoría de los alumnos estuvieron muy de acuerdo y acuerdo con esto. Pero para el grupo de Mecatrónica hubo un mayor porcentaje que no estuvo ni de acuerdo ni en desacuerdo al respecto.

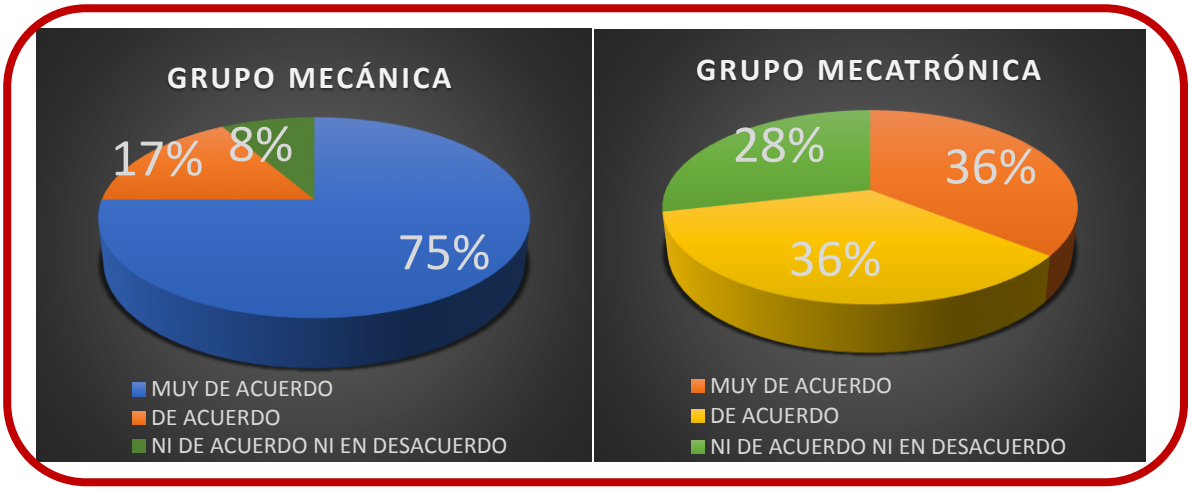


Ilustración 111. *Resultados pregunta 13*

La mayoría de los alumnos les gustaría aprender otros temas de la robótica como la Dinámica utilizando un simulador como los desarrollados. Pues en ambos grupos la mayoría de los alumnos estuvieron de acuerdo y muy de acuerdo con este punto.

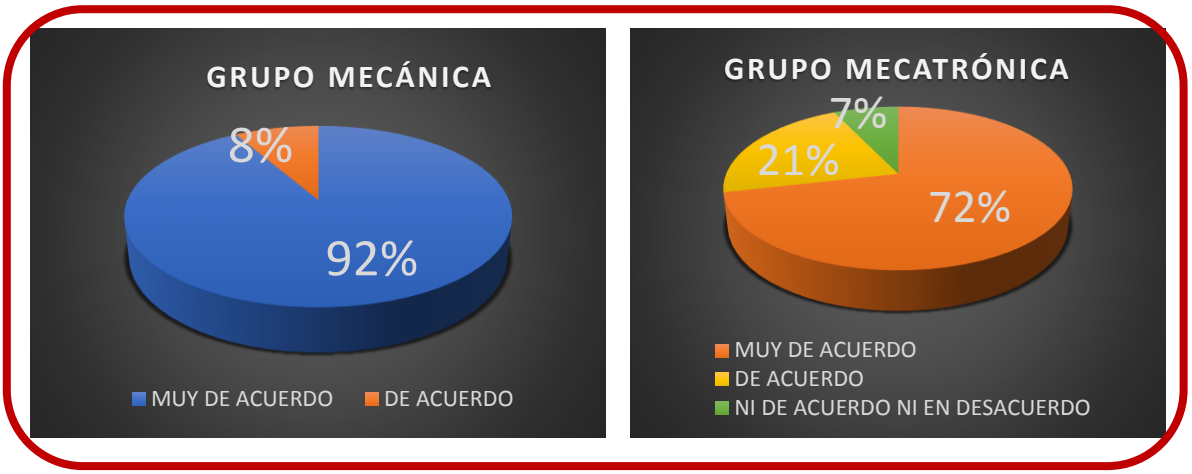


Ilustración 112. *Resultados pregunta 14*

Enseñar usando la tecnología es cada vez más evidente, ambos grupos están de acuerdo y muy de acuerdo con que se enseñe usando la tecnología para este tipo de materias.

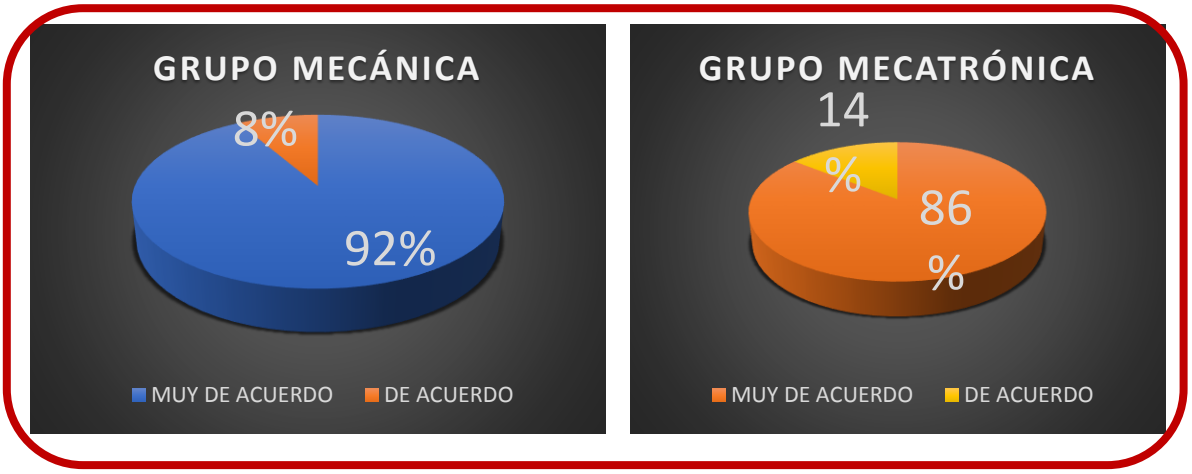


Ilustración 113. *Resultados pregunta 15*

El grupo de Mecánica opina que el usar simuladores los acerca al mundo industrial, llevando los conceptos aprendidos a la práctica. Al igual que en el grupo de Mecatrónica, pero con una pequeña diferencia en las personas que no están de acuerdo ni en desacuerdo con esto.

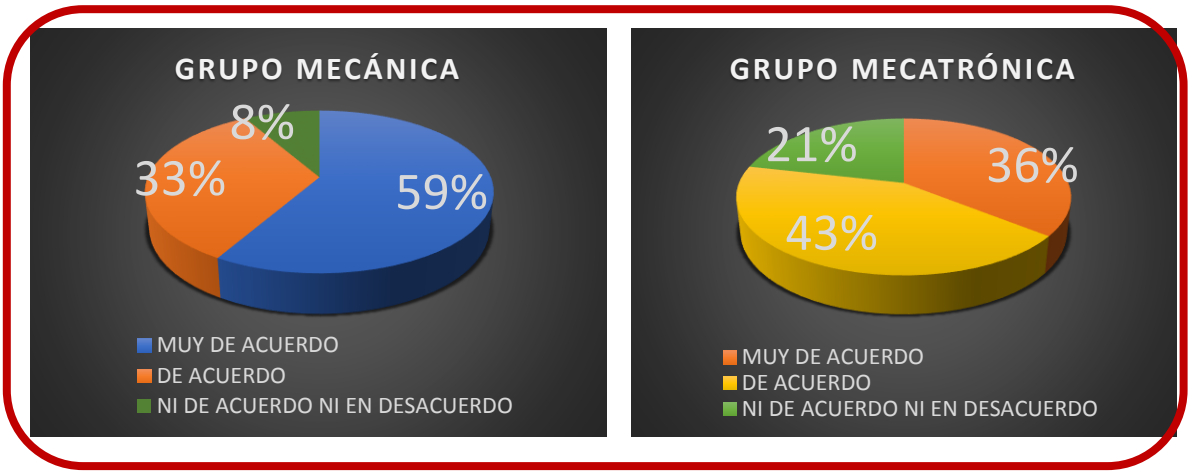


Ilustración 114. *Resultados pregunta 16*

Se tomó todos los puntos de cada cuestionario y se hizo una base de datos con Excel se obtuvieron las siguientes gráficas que muestran el puntaje de cada alumno que les dio a los simuladores.

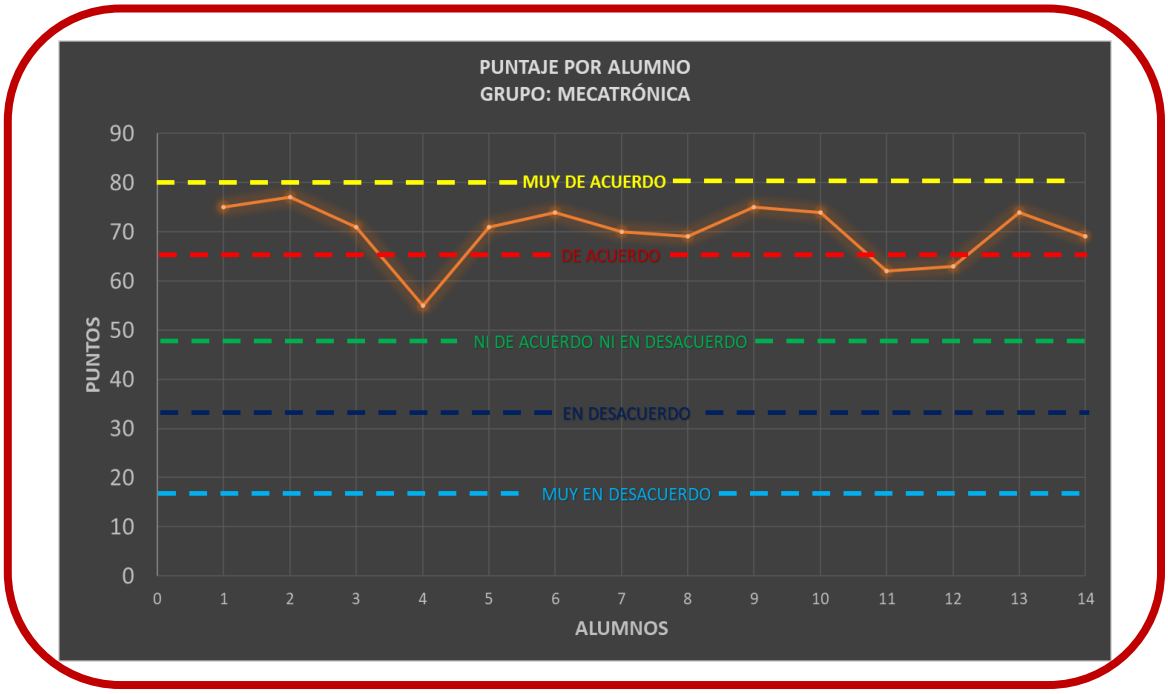


Ilustración 115. *Puntaje a los simuladores por el grupo mecatrónica*

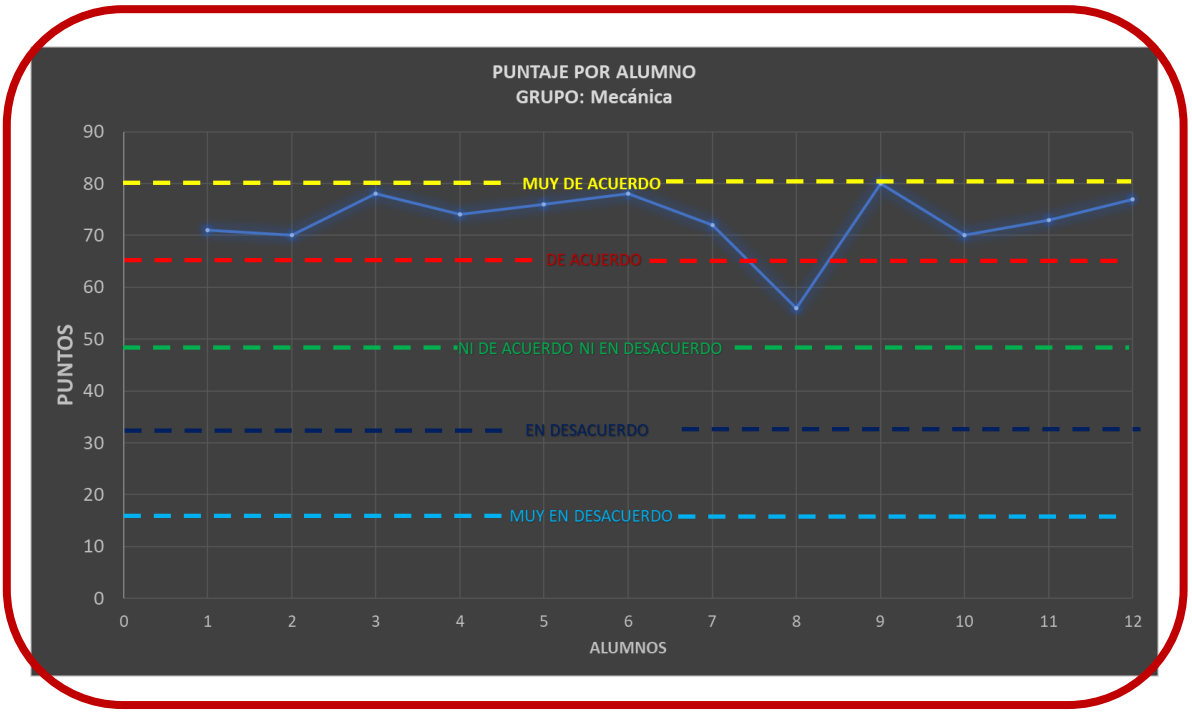


Ilustración 116. *Puntaje a las aplicaciones por el grupo mecánica*

Se observo en ambas gráficas que la mayor parte de los puntajes se encuentran entre 64 y 80 puntos. Por lo cual las aplicaciones desarrolladas son buenas herramientas para enseñar los temas de cinemática directa e inversa.

6. CONCLUSIÓN

En relación a los resultados obtenidos al poner a prueba los dos simuladores con el grupo de Mecánica de la FES Aragón y con el de Mecatrónica de Ciudad Universitaria. Se observó que los alumnos en ambos grupos comprendieron de una mejor manera los temas de cinemática directa e inversa, ayudándolos a resolver las dudas que surgieron durante el transcurso del semestre en línea, también las aplicaciones les bridaron mejores herramientas para poder visualizar los movimientos del robot, además los acercó al mundo real, pues los robots con los que cuenta la aplicación están a la venta y se utilizan en las industrias actualmente.

Se observó que las aplicaciones no solamente ayudan a observar mejor los sistemas de referencia de cada articulación, sino que, ayudan a comprender mejor los conceptos de cinemática inversa y directa. Los alumnos dieron comentarios en los cuales felicitaban a la aplicación por ser buena herramienta para aprender estos temas de la robótica, tanto es así, que les hubiera gustado aprender otros temas de la materia como la dinámica, jacobianos, etc. con esta clase de herramientas.

Los compañeros de ambos grupos de estudio comentaron que la aplicación sería una buena herramienta de apoyo para el docente, con estos softwares el profesor podría enseñar de una mejor manera estos temas y hacer que los alumnos los comprendan rápido y así cubrir con el temario de la materia.

Además, con los estudios realizados se notó que las aplicaciones no solamente pueden ayudar para las clases en línea, si no también, ahora que los alumnos regresen a clases presenciales, estas herramientas también pueden ser utilizadas por el profesor para enseñar robótica presencialmente.

Como trabajo a futuro se puede crear un plan para que los alumnos puedan llevar estos softwares instalados en sus computadoras a clases, o, tomar las clases en una sala de computación para hacer uso de estas herramientas. También, en cuanto las aplicaciones, se les pueden realizar mejoras, ya que tienen algunos problemas en la programación, porque se tuvieron inconvenientes al instalarlos, además algunos robots no cargaban todos sus eslabones. También, el diseño de las

aplicaciones no fue el mejor, ya que, muchos alumnos no les agrado los colores ni como estaba distribuido el programa. Por lo cual hay mucho que mejorar para resolver estos problemas.

Una opción para solucionar algunos de estos problemas es ejecutando las aplicaciones desde Matlab. Por lo cual como trabajo a futuro se puede compartir el código para que los alumnos instalen el software desde Matlab.

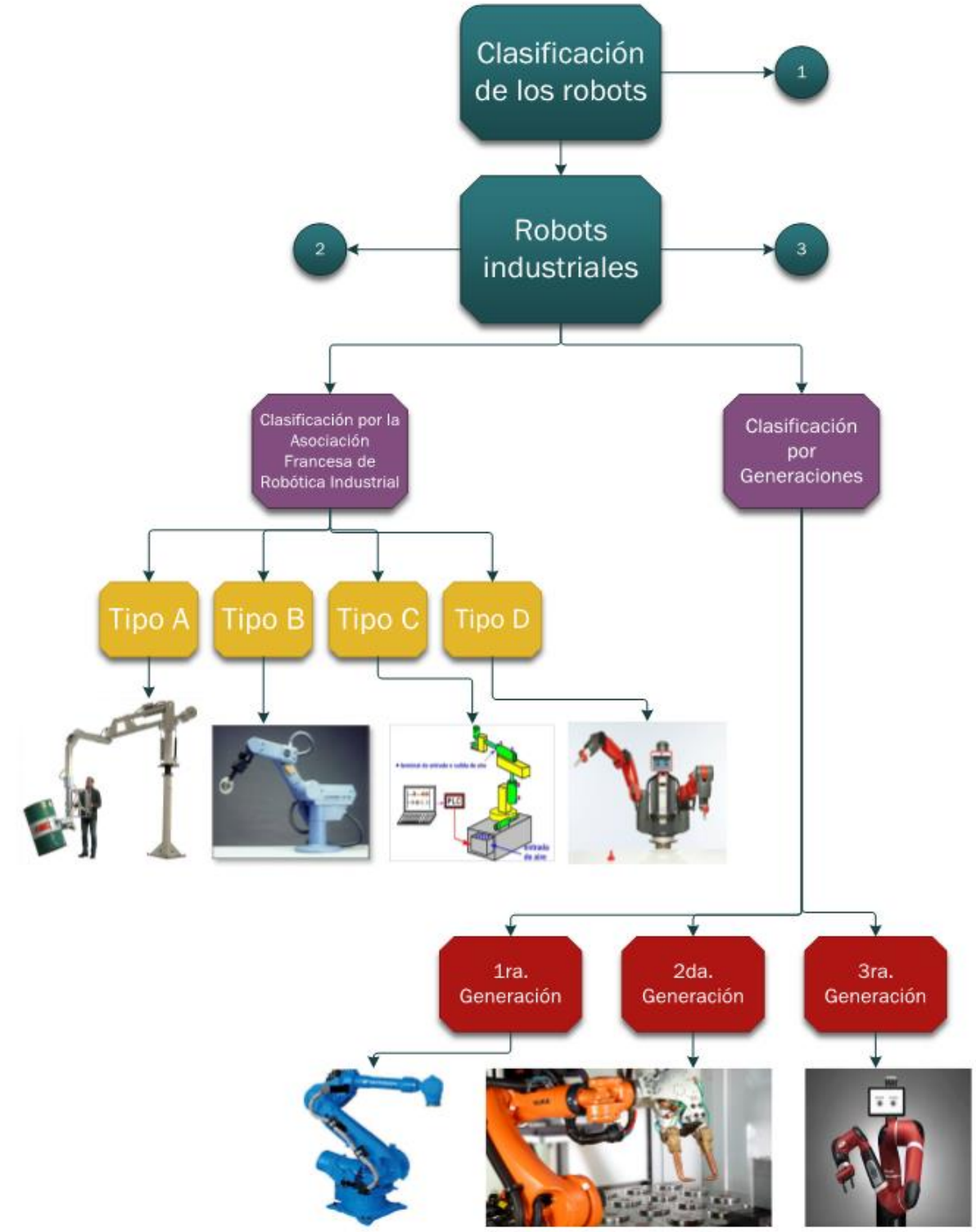
Finalmente se notó que los simuladores son una herramienta que conducen a la innovación y al mejoramiento del proceso enseñanza-aprendizaje y beneficien a los alumnos, y tienen mayor impacto para dar clases en línea. El material que se trabajó con los alumnos fue recibido con buenas puntuaciones a las aplicaciones. Por lo tanto, es momento de actualizarse, la tecnología avanza cada vez más, es momento de dejar de ver a la tecnología como un enemigo y hay que empezar a verla como un aliado, para enseñar a las futuras generaciones las materias que se imparten en la Universidad, en este caso como lo fue la robótica.

APÉNDICES

Apéndice 1 Datos del Ing. Jesús. Vendedor de Cobots.

Apéndice 2 Folleto informativo del Cobot.

Apéndice 3. Clasificación de los Robots.



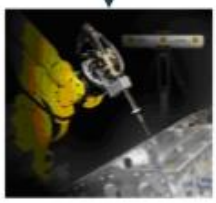
2

Clasificación por la Federación Internacional de Robótica

Robot secuencial.

Robot de trayectoria controlable

Robot adptativo



Robot telemanipulado

Telemanipulador

3

Clasificación por sus configuraciones

Cartesiano

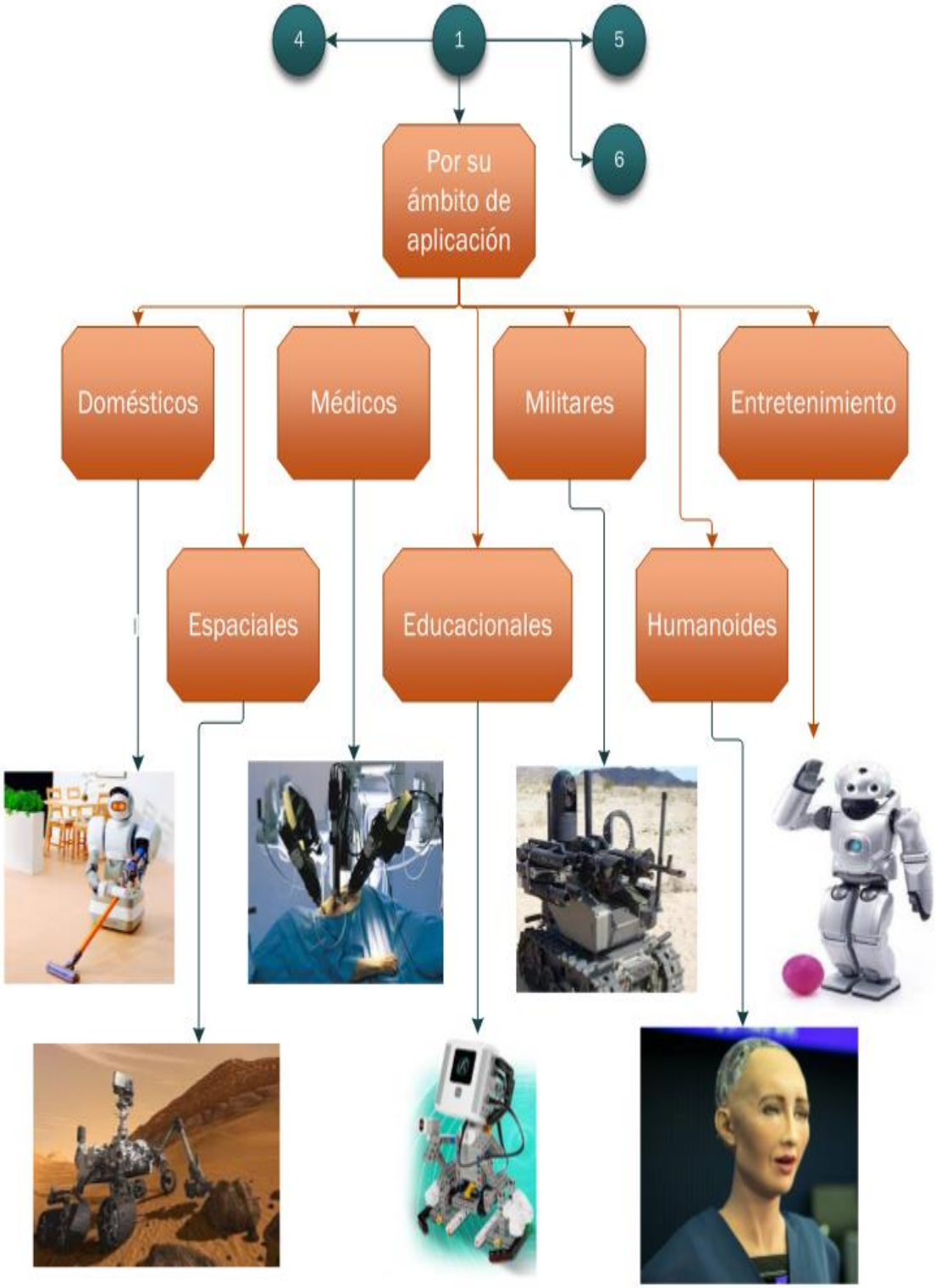
Paralelo

Cilíndrico

Angular

Mixto

Polar


4
1
5
6
Por su ámbito de aplicación
Domésticos
Médicos
Militares
Entretenimiento
Espaciales
Educacionales
Humanoides

4

Clasificación en función del entorno de trabajo

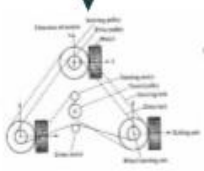
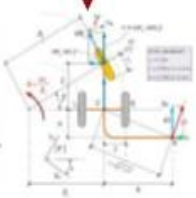
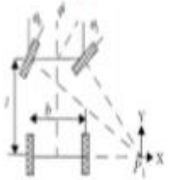
- Estacionarios
- De suelo
  - Orugas
  - Patas
  - Ruedas
    - Diferencial
    - Sincrona
    - Triciclo
    - Ackerman
    - Omnidireccionales
- Marinos
- Aéreos
- Microgravedad
- Entornos peligrosos










Apéndice 4 Datos del Vendedor del Robot IRB-120 ABB

Miguel Bórquez
Channel Manager
Robotics Mexico

miguel-angel.borquez@mx.abb.com
Phone: +52 (444) 870 8117
Cell: +52 (444) 204 9316

https://new.abb.com/channel-partners

ABB México S.A. de C.V.
E-Building
Av. Central #310
Parque Logístico
San Luis Potosí, S.L.P.
C.P. 78395
Mexico

Apéndice 5 Catálogo de robots para simular con RST

ROBOTS PARA SIMULAR EN ROBOTIC SYSTEM TOOLBOX	
NOMBRE DEL ROBOT	DESCRIPCIÓN
"abbIrb120"	ABB IRB 120 6-axis robot
"abbIrb120T"	ABB IRB 120T 6-axis robot
"abbIrb1600"	ABB IRB 1600 6-axis robot
"abbYuMi"	ABB YuMi 2-armed robot
"amrPioneer3AT"	Adept MobileRobots Pioneer 3-AT mobile robot
"amrPioneer3DX"	Adept MobileRobots Pioneer 3-DX mobile robot
"amrPioneerLX"	Adept MobileRobots Pioneer LX mobile robot
"atlas"	Boston Dynamics ATLAS® Humanoid robot
"clearpathHusky"	Clearpath Robotics Husky mobile robot
"clearpathJackal"	Clearpath Robotics Jackal mobile robot
clearpathTurtleBot2"	Clearpath Robotics TurtleBot 2 mobile robot
"fanucLRMate200ib"	FANUC LR Mate 200iB 6-axis robot
"fanucM16ib"	FANUC M-16iB 6-axis robot
"frankaEmikaPanda"	Franka Emika Panda 7-axis robot
"kinovaGen3"	KINOVA® Gen3 robot
"kinovaJacoJ2N6S200"	KINOVA JACO® 2-fingered 6 DOF robot with non-spherical wrist
"kinovaJacoJ2N6S300"	KINOVA JACO® 3-fingered 6 DOF robot with non-spherical wrist
"kinovaJacoJ2N7S300"	KINOVA JACO® 3-fingered 7 DOF robot with non-spherical wrist
"kinovaJacoJ2S6S300"	KINOVA JACO® 3-fingered 6 DOF robot with spherical wrist
"kinovaJacoJ2S7S300"	KINOVA JACO® 3-fingered 7 DOF robot with spherical wrist
"kinovaJacoTwoArmExample"	Two KINOVA JACO® 3-fingered 6 DOF robots with non-spherical wrist
"kinovaMicoM1N4S200"	KINOVA MICO® 2-fingered 4 DOF robot
"kinovaMicoM1N6S200"	KINOVA MICO® 2-fingered 6 DOF robot
"kinovaMicoM1N6S300"	KINOVA MICO® 3-fingered 6 DOF robot
"kinovaMovo"	KINOVA MOVO® 2-armed mobile robot
"rethinkBaxter"	Rethink Robotics Baxter 2-armed robot
"robotisOP2"	ROBOTIS OP2 Humanoid robot
"robotisOpenManipulator"	ROBOTIS OpenMANIPULATOR 4-axis robot with gripper
"robotisTurtleBot3Burger"	ROBOTIS TurtleBot 3 Burger robot
"robotisTurtleBot3Waffle"	ROBOTIS TurtleBot 3 Waffle robot
"robotisTurtleBot3WafflePi"	ROBOTIS TurtleBot 3 Waffle Pi robot
"universalUR3"	Universal Robots UR3 6-axis robot
"universalUR5"	Universal Robots UR5 6-axis robot
"universalUR10"	Universal Robots UR10 6-axis robot
"valkyrie"	NASA Valkyrie Humanoid robot
"willowgaragePR2"	Willow Garage PR2 2-armed mobile robot
"yaskawaMotomanMH5"	Yaskawa Motoman MH5 6-axis robot

Apéndice 6 Catálogo de Robots que fabrica la marca ABB

Articulated robots	Payload (kg)	Reach (m)	Pos. rep. (mm)	Mounting	Protection	Axes	Controller	Remark
 IRB 1100	4 4	0.475 0.58	0.01	Any angle	IP40 Option: Clean Room ISO 4, IP67	6	OmniCore™	
 IRB 120	3	0.58	0.01	Any angle	Std: IP30 Option: Clean Room ISO 5, Food Grade	6	IRC5	
 IRB 1200	5 7	0.90 0.70	0.02 0.025	Any angle	Std: IP40. Option: Foundry Plus 2, IP67, Clean Room ISO 3, Food Grade	6	IRC5	
 IRB 140 IRB 140T	6	0.80	0.03	Floor, wall, inverted, tilted	Std: IP67. Option: Foundry Plus 2, Clean Room ISO 6, SteamWash	6	IRC5	
 IRB 1410	5	1.44	0.02	Floor		6	IRC5	
 IRB 1520ID	4	1.50	0.05	Floor, inverted	Std: IP40	6	IRC5	
 IRB 1600	6 10	1.20/1.45 1.20/ 1.45	0.02 - 0.05	Floor, wall, inverted, tilted, shelf	Std: IP54 Option: Foundry Plus 2 with IP67	6	IRC5	
 IRB 1660ID	4 6	1.55	0.02	Floor, wall, inverted, tilted	Std: IP40	6	IRC5	
 IRB 2400	12 20	1.55	0.03	Floor, inverted	Std: IP54 Option: Foundry Plus 2 with IP67	6	IRC5	
 IRB 2600	12/12 20	1.65/1.85 1.65	0.04	Floor, wall, inverted, titled, shelf	Std: IP67, IP54 (axis 4) Option: Foundry Plus 2	6	IRC5	
 IRB 2600ID	15 8	1.85 2.00	0.02 - 0.03	Floor, wall, shelf, titled, inverted	Std: IP67 for base and lower arm. IP54 for upper arm	6	IRC5	
 IRB 4400 IRB 4400L	60 10	1.96 2.55	0.06 0.05	Floor	Std: IP54 Option: IP67, Foundry Plus 2	6	IRC5	
 IRB 460	110	2.40	0.20	Floor	Std: IP67	4	IRC5	
 IRB 4600	20/40/ 45/60	2.5/2.55/ 2.05/2.05	0.05 - 0.06	Floor, inverted, tilted, shelf	Std: IP67 Option: Foundry Plus 2, Foundry Prime 2	6	IRC5	

 IRB 660	180 250	3.15	0.05	Floor	Std: IP67	4	IRC5	

Articulated robots	Payload (kg)	Reach (m)	Pos. rep. (mm)	Mounting	Protection	Axes	Controller	Remark
 IRB 6700	150/155/175 200/205/235 245/300	3.20/2.85/3.05 2.60/2.80/2.65 3.00/2.70	0.05 - 0.10	Floor	Std: IP67 Option: Foundry Plus 2	6	IRC5	Available with LeanID
 IRB 6700INV	245 300	2.90 2.60	0.10 0.05	Inverted	Std: IP67 Option: Foundry Plus 2	6	IRC5	Available with LeanID
 IRB 6790	205 235	2.80 2.65	0.05	Floor	Std: IP69 Option: Foundry Prime 3	6	IRC5	
 IRB 760	450	3.18	0.05	Floor	Std: IP67	4	IRC5	
 IRB 7600	150/325/ 340/400/ 500	3.50/3.10*/ 2.80*/2.55*/ 2.55	0.10 - 0.30	Floor	Std: IP67 Option: Foundry Plus 2	6	IRC5	*Available with LeanID
 IRB 8700	550 800	4.20 3.50	0.05 - 0.10	Floor	Std: IP67 Option: Foundry Plus 2	6	IRC5	Available with LeanID

YuMi® family	Payload (kg)	Reach (m)	Pos. rep. (mm)	Mounting	Protection	Axes	Controller	Remark
 IRB 14000	0.50	0.559	0.02	Table	Std: IP30 and Clean Room ISO 5	14	Embedded IRC5	Safety: PL b Cat B
 IRB 14050	0.50	0.559	0.02	Any angle	Std: IP30 and Clean Room ISO 5	7	OmniCore™	Estop and PStop: PL d Cat 3 Option: SafeMove2

Parallel robots	Payload (kg)	Reach (m)	Pos. rep. (mm)	Mounting	Protection	Axes	Controller	Remark
 IRB 360 FlexPicker®	1 3 6 8	1.13/1.60 1.13 1.60 1.13	0.03 - 0.09	Inverted	Std: IP54/67/69K Option: Clean Room, Wash-Down, Stainless	4	IRC5	

SCARA robots	Payload (kg)	Reach (m)	Pos. rep. (mm)	Mounting	Protection	Axes	Controller	Remark
 IRB 910SC	3 (max 6)	0.45 0.55 0.65	0.01	Table	Std: IP20	4	IRC5	
 IRB 910INV	1 (max 3) 3 (max 6)	0.35 0.55	0.01 0.015	Inverted	Std: IP30. Option: Clean Room ISO 5, IP54	4	OmniCore™	

Paint robots		Payload (kg)	Reach (m)	Pos. rep. (mm)	Mounting	Protection	Axes	Controller	Remark
	IRB 52	7	1.20/1.45	0.15	Floor, wall, inverted, tilted	Std: IP67, Ex	6	IRC5P	
	IRB 5500-22	13	2.97	0.15	Floor, wall, inverted, tilted	Std: IP67, Ex	6	IRC5P	
	IRB 5500-23	13	2.97 Rail travel length: 2-15	0.15	Clean-wall rail	Std: IP67, Ex	7	IRC5P	
	IRB 5500-25	13	2.97 Rail travel length: 2-15	0.15	Elevated, Robot: tilted, upright and inverted	Std: IP67, Ex	7	IRC5P	
	IRB 5500-27	13	2.97	0.15	Wall, floor, inverted	Std: IP67, Ex	7	IRC5P	
	IRB 5510	13	2.56	0.15	Floor	Std: IP67, Ex	6	IRC5P	
	IRB 5350	7	1.35 Rail travel length: 3-10	0.15	Floor, rail	Std: IP67, Ex	3/4	IRC5P	

Compliant with ISO 9283.

Apéndice 7 Código de la aplicación uno

```
function varargout = DenavitHartenberg(varargin)

% DENAVITHARTENBERG MATLAB code for DenavitHartenberg.fig
%      DENAVITHARTENBERG, by itself, creates a new DENAVITHARTENBERG or raises the existing
%      singleton*.
%
%      H = DENAVITHARTENBERG returns the handle to a new DENAVITHARTENBERG or the handle to
%      the existing singleton*.
%
%      DENAVITHARTENBERG('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in DENAVITHARTENBERG.M with the given input arguments.
%
%      DENAVITHARTENBERG('Property','Value',...) creates a new DENAVITHARTENBERG or raises
the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before DenavitHartenberg_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to DenavitHartenberg_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help DenavitHartenberg

% Last Modified by GUIDE v2.5 11-Sep-2020 14:00:56
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @DenavitHartenberg_OpeningFcn, ...
                   'gui_OutputFcn',  @DenavitHartenberg_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDI
% --- Executes just before DenavitHartenberg is made visible.
function DenavitHartenberg_OpeningFcn(hObject, eventdata, handles, varargin)
% Instrucciones
global txt1 txt2 txt3 txt4 txt5

txt1=["Bienvenido"
    " "
    "Está es una aplicación para apoyarte a realizar la cinemática directa de manipuladores
seriales con juntas rotativas o prismáticas con parámetros de Denavith-Hartenberg. La
aplicación cuenta con:"
    " "
    "•    Robots de Ejemplo: Son robots industriales a los cuales se les puede dar
movimiento a las articulaciones y ver su comportamiento."
    " "
    "•    Cinemática directa: Obtiene la matriz de transformación homogénea de manipuladores
seriales, solamente insertando los parámetros de Denavith-Hartenberg."
    " "
    "•    Grafica manipuladores de 3,4,5,6 GDL, con pura junta rotativa o prismática o la
combinación de ambos."
    " "
    "•    Panel para manipular las articulaciones y darle movimiento al gráfico 3D."];
```

```
txt2=["Robots de Ejemplo"
    " "
    "La aplicación cuenta con un panel para escoger entre doce robots industriales, algunos
son utilizados actualmente, otros tienen un poco más de tiempo, pero son de gran ayuda para
poder visualizar la cinemática directa de los manipuladores."
    " "
    'Puedes acceder a ellos dando click en el circulo selector del panel con el título
"ROBOTS". Aparecerá un gráfico mostrando la configuración y una foto del robot de la vida
real.'
    " "
    'Para poder darle movimiento a las articulaciones. En el panel que tiene como título
"Valor de las articulaciones", puede ingresar los valores manualmente o con la barra
deslizante mover al valor de la articulación(nes) deseado (los valores estan en grados y
pueden ser tanto negativos como positivos). Después de elegir el valor de cada articulación,
presione el botón cargar y vea cómo cambia el gráfico.'
    " "
    "En la parte de abajo aparecerá un recuadro con el titulo “Datos”, en este recuadro se
muestran el nombre,la compañia, los gdl con los que cuenta y su configuración."];

txt3=["Cinemática Directa"
    " "
    "Para obtener la matriz de transformación homogénea de robots seriales primero debe
seleccionar una de las siguientes opciones:"
    " "
    "•    Si la configuración del robot cuenta con puras juntas rotacionales presione el
botón “Robots Rotaciones”."
    " "
    "•    Si la configuración cuenta con puras juntas prismáticas o la combinación entre
rotacionales y prismáticas presione el botón “Robots con J. Prismática”"
    " "
    "En el panel con el título “Insertar parámetros de Denavith-Hartenberg” ingrese los
valores para cada eslabón, siguiendo estas normas:"
    " "
    "•    Si la junta es rotacional theta ingrese 0 y en junta ingrese 0 (thetai=0 y
junta=0)"
    " "
    "•    Si la junta es prismática d(di) ingrese 0 y en junta ingrese 1 (di=0 y junta=1)"];
txt4=["Nota:"
    " "
    "•    Como mínimo se pueden ingresar 3 eslabones como máximo 6 eslabones"
    " "
    "•    Solo permite valores numéricos."
    " "
    "Cuando termine de ingresar los parámetros presione un botón dependiendo el caso:"
    " "
    "• 3GDL: Si desea graficar un robot de 3GDL (Grados de Libertad)."
    " "
    "• 4GDL: Si desea graficar un robot de 4GDL."
    " "
    "• 5GDL: Si desea graficar un robot de 5GDL."
    " "
    "• 6GDL: Y por último desea graficar un robot de 6GDL."];
txt5=["Manipulación de articulaciones"
    " "
    "Para manipular las articulaciones y ver su comportamiento, ingrese valores de manera
manual o moviendo la barra deslizante. Cuando ya tenga los valores deseados para cada
articulación vuelva a presionar el botón (3,4,5,6GDL dependiendo el caso)."];
set(handles.instrucciones,'String',txt1)
global q workspace comparador q1 q2 q3 q4 q5 q6
q1=0;
q2=0;
q3=0;
q4=0;
q5=0;
q6=0;
comparador=0;
q=[0 0 0 0 0 0]
set(handles.q1,'String',num2str(q(1)));
set(handles.q2,'String',num2str(q(2)));
set(handles.q3,'String',num2str(q(3)));
set(handles.q4,'String',num2str(q(4)));
```

```
set(handles.q5,'String',num2str(q(5)));
set(handles.q6,'String',num2str(q(6)));
workspace=[-1 1 -1 1 -1 1];
set(handles.xmin,'String',num2str(workspace(1)));
set(handles.xmax,'String',num2str(workspace(2)));
set(handles.ymin,'String',num2str(workspace(3)));
set(handles.ymax,'String',num2str(workspace(4)));
set(handles.zmin,'String',num2str(workspace(5)));
set(handles.zmax,'String',num2str(workspace(6)));
%Contador de instrucciones
global count
count = 0;
%Cargar Logos
axes(handles.logounam);
unam=imread('logounam.jpg');
imshow(unam)
axes(handles.logofes);
fes=imread('logofes.jpg');
imshow(fes)
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);
function varargout = DenavitHartenberg_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
% --- Executes on button press in rotacional.
function rotacional_Callback(hObject, eventdata, handles)
set(handles.robotsr,'visible','on')
set(handles.robotsp,'visible','off')
set(handles.workspace,'visible','off')
set(handles.datos,'visible','off')
set(handles.matriz,'visible','on')
set(handles.cargar,'visible','off')
axes(handles.plot)
cla
axes(handles.robot)
cla
% --- Executes on button press in clear.
function clear_Callback(hObject, eventdata, handles)
axes(handles.plot)
cla
axes(handles.robot)
cla
% --- Executes on button press in prismatica.
function prismatica_Callback(hObject, eventdata, handles)
set(handles.robotsr,'visible','off')
set(handles.datos,'visible','off')
set(handles.robotsp,'visible','on')
set(handles.workspace,'visible','on')
set(handles.matriz,'visible','on')
set(handles.cargar,'visible','off')
axes(handles.plot)
cla
axes(handles.robot)
cla

% --- Executes on button press in boton3gdl.
function boton3gdl_Callback(hObject, eventdata, handles)
set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
%Eslabon 1
t1_get=str2double(get(handles.edit1,'String'));
t1= t1_get*(pi/180); %Conversion de grados a radianes
d1=str2double(get(handles.edit2,'String'));
a1=str2double(get(handles.edit3,'String'));
ap1_get=str2double(get(handles.edit4,'String'));
ap1= ap1_get*(pi/180); %Conversion de grados a radianes
j1=str2double(get(handles.edit5,'String'));
L(1)=Link([t1,d1,a1,ap1,j1]); %Crear el enlace

%Eslabon 2
t2_get=str2double(get(handles.edit6,'String'));
```

```
t2= t2_get*(pi/180); %Conversion de grados a radianes
d2=str2double(get(handles.edit7,'String'));
a2=str2double(get(handles.edit8,'String'));
ap2_get=str2double(get(handles.edit9,'String'));
ap2= ap2_get*(pi/180); %Conversion de grados a radianes
j2=str2double(get(handles.edit10,'String'));
L(2)=Link([t2,d2,a2,ap2,j2]); %Crear el enlace

%Eslabon 3
t3_get=str2double(get(handles.edit11,'String'));
t3= t3_get*(pi/180); %Conversion de grados a radianes
d3=str2double(get(handles.edit12,'String'));
a3=str2double(get(handles.edit13,'String'));
ap3_get=str2double(get(handles.edit14,'String'));
ap3= ap3_get*(pi/180); %Conversion de grados a radianes
j3=str2double(get(handles.edit15,'String'));
L(3)=Link([t3,d3,a3,ap3,j3]); %Crear el enlace

%Mostrar pantallas
set(handles.plot,'visible','on')
set(handles.robot,'visible','off')

%Dibujar el robot
global q
robot=SerialLink(L,'name','3GDL');
a_get=str2double(get(handles.q1,'String'));
a=a_get*(pi/180);%Convertir de gradosa radianes
b_get=str2double(get(handles.q2,'String'));
b=b_get*(pi/180);%Convertir de gradosa radianes
c_get=str2double(get(handles.q3,'String'));
c=c_get*(pi/180);%Convertir de gradosa radianes
q_3art=[a b c];
axes(handles.robot);
cla
set(handles.robot,'visible','off')
axes(handles.plot);
cla
robot.plot(q_3art)
%Matriz de transformación Homogénea
T1=[cosd(a_get),-cosd(ap1_get)*sind(a_get),sind(ap1_get)*sind(a_get), a1*cosd(a_get);
sind(a_get),cosd(ap1_get)*cosd(a_get),-sind(ap1_get)*cosd(a_get), a1*sind(a_get); 0,
sind(ap1_get),cosd(ap1_get),d1;0,0,0,1];
T2=[cosd(b_get),-cosd(ap2_get)*sind(b_get),sind(ap2_get)*sind(b_get), a2*cosd(b_get);
sind(b_get),cosd(ap2_get)*cosd(b_get),-sind(ap2_get)*cosd(b_get), a2*sind(b_get); 0,
sind(ap2_get),cosd(ap2_get),d2;0,0,0,1];
T3=[cosd(c_get),-cosd(ap3_get)*sind(c_get),sind(ap3_get)*sind(c_get), a3*cosd(c_get);
sind(c_get),cosd(ap3_get)*cosd(c_get),-sind(ap3_get)*cosd(c_get), a3*sind(c_get); 0,
sind(ap3_get),cosd(ap3_get),d3;0,0,0,1];
T=T1*T2*T3
%Fila 1
set(handles.edit46,'String',T(1,1))
set(handles.edit47,'String',T(1,2))
set(handles.edit48,'String',T(1,3))
set(handles.edit49,'String',T(1,4))
%Fila 2
set(handles.edit50,'String',T(2,1))
set(handles.edit51,'String',T(2,2))
set(handles.edit52,'String',T(2,3))
set(handles.edit53,'String',T(2,4))
%Fila 3
set(handles.edit54,'String',T(3,1))
set(handles.edit55,'String',T(3,2))
set(handles.edit56,'String',T(3,3))
set(handles.edit57,'String',T(3,4))
%Fila 4
set(handles.edit58,'String',T(4,1))
set(handles.edit59,'String',T(4,2))
set(handles.edit60,'String',T(4,3))
set(handles.edit61,'String',T(4,4))
% --- Executes on button press in boton4gdl.
function boton4gdl_Callback(hObject, eventdata, handles)
```

```
set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
%Eslabon 1
t1_get=str2double(get(handles.edit1,'String'));
t1= t1_get*(pi/180); %Conversion de grados a radianes
d1=str2double(get(handles.edit2,'String'));
a1=str2double(get(handles.edit3,'String'));
ap1_get=str2double(get(handles.edit4,'String'));
ap1= ap1_get*(pi/180); %Conversion de grados a radianes
j1=str2double(get(handles.edit5,'String'));
L(1)=Link([t1,d1,a1,ap1,j1]); %Crear el enlace
%Eslabon 2
t2_get=str2double(get(handles.edit6,'String'));
t2= t2_get*(pi/180); %Conversion de grados a radianes
d2=str2double(get(handles.edit7,'String'));
a2=str2double(get(handles.edit8,'String'));
ap2_get=str2double(get(handles.edit9,'String'));
ap2= ap2_get*(pi/180); %Conversion de grados a radianes
j2=str2double(get(handles.edit10,'String'));
L(2)=Link([t2,d2,a2,ap2,j2]); %Crear el enlace
%Eslabon 3
t3_get=str2double(get(handles.edit11,'String'));
t3= t3_get*(pi/180); %Conversion de grados a radianes
d3=str2double(get(handles.edit12,'String'));
a3=str2double(get(handles.edit13,'String'));
ap3_get=str2double(get(handles.edit14,'String'));
ap3= ap3_get*(pi/180); %Conversion de grados a radianes
j3=str2double(get(handles.edit15,'String'));
L(3)=Link([t3,d3,a3,ap3,j3]); %Crear el enlace
%Eslabon 4
t4_get=str2double(get(handles.edit16,'String'));
t4= t4_get*(pi/180); %Conversion de grados a radianes
d4=str2double(get(handles.edit17,'String'));
a4=str2double(get(handles.edit18,'String'));
ap4_get=str2double(get(handles.edit19,'String'));
ap4= ap4_get*(pi/180); %Conversion de grados a radianes
j4=str2double(get(handles.edit20,'String'));
L(4)=Link([t4,d4,a4,ap4,j4]); %Crear el enlace
%Mostrar pantallas
set(handles.plot,'visible','on');
%Dibujar el robot
global q
robot=SerialLink(L,'name','3GDL');
a_get=str2double(get(handles.q1,'String'));
a=a_get*(pi/180);%Convertir de gradosa radianes
b_get=str2double(get(handles.q2,'String'));
b=b_get*(pi/180);%Convertir de gradosa radianes
c_get=str2double(get(handles.q3,'String'));
c=c_get*(pi/180);%Convertir de gradosa radianes
d_get=str2double(get(handles.q4,'String'));
d=d_get*(pi/180);%Convertir de gradosa radianes
q_4art=[a b c d];
axes(handles.robot)
cla
set(handles.robot,'visible','off')
axes(handles.plot);
cla
robot.plot(q_4art);
%Matriz de transformación Homogénea
T1=[cosd(a_get),-cosd(ap1_get)*sind(a_get),sind(ap1_get)*sind(a_get), a1*cosd(a_get);
sind(a_get),cosd(ap1_get)*cosd(a_get),-sind(ap1_get)*cosd(a_get), a1*sind(a_get); 0,
sind(ap1_get),cosd(ap1_get),d1;0,0,0,1]
T2=[cosd(b_get),-cosd(ap2_get)*sind(b_get),sind(ap2_get)*sind(b_get), a2*cosd(b_get);
sind(b_get),cosd(ap2_get)*cosd(b_get),-sind(ap2_get)*cosd(b_get), a2*sind(b_get); 0,
sind(ap2_get),cosd(ap2_get),d2;0,0,0,1]
T3=[cosd(c_get),-cosd(ap3_get)*sind(c_get),sind(ap3_get)*sind(c_get), a3*cosd(c_get);
sind(c_get),cosd(ap3_get)*cosd(c_get),-sind(ap3_get)*cosd(c_get), a3*sind(c_get); 0,
sind(ap3_get),cosd(ap3_get),d3;0,0,0,1]
T4=[cosd(d_get),-cosd(ap4_get)*sind(d_get),sind(ap4_get)*sind(d_get), a4*cosd(d_get);
sind(d_get),cosd(ap4_get)*cosd(d_get),-sind(ap4_get)*cosd(d_get), a4*sind(d_get); 0,
sind(ap4_get),cosd(ap4_get),d4;0,0,0,1]
```

```
T=T1*T2*T3*T4
%Fila 1
set(handles.edit46,'String',T(1,1))
set(handles.edit47,'String',T(1,2))
set(handles.edit48,'String',T(1,3))
set(handles.edit49,'String',T(1,4))
%Fila 2
set(handles.edit50,'String',T(2,1))
set(handles.edit51,'String',T(2,2))
set(handles.edit52,'String',T(2,3))
set(handles.edit53,'String',T(2,4))
%Fila 3
set(handles.edit54,'String',T(3,1))
set(handles.edit55,'String',T(3,2))
set(handles.edit56,'String',T(3,3))
set(handles.edit57,'String',T(3,4))
%Fila 4
set(handles.edit58,'String',T(4,1))
set(handles.edit59,'String',T(4,2))
set(handles.edit60,'String',T(4,3))
set(handles.edit61,'String',T(4,4))
% --- Executes on button press in boton5gdl.
function boton5gdl_Callback(hObject, eventdata, handles)
set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
%Eslabon 1
t1_get=str2double(get(handles.edit1,'String'));
t1= t1_get*(pi/180); %Conversion de grados a radianes
d1=str2double(get(handles.edit2,'String'));
a1=str2double(get(handles.edit3,'String'));
ap1_get=str2double(get(handles.edit4,'String'));
ap1= ap1_get*(pi/180); %Conversion de grados a radianes
j1=str2double(get(handles.edit5,'String'));
L(1)=Link([t1,d1,a1,ap1,j1]); %Crear el enlace
%Eslabon 2
t2_get=str2double(get(handles.edit6,'String'));
t2= t2_get*(pi/180); %Conversion de grados a radianes
d2=str2double(get(handles.edit7,'String'));
a2=str2double(get(handles.edit8,'String'));
ap2_get=str2double(get(handles.edit9,'String'));
ap2= ap2_get*(pi/180); %Conversion de grados a radianes
j2=str2double(get(handles.edit10,'String'));
L(2)=Link([t2,d2,a2,ap2,j2]); %Crear el enlace
%Eslabon 3
t3_get=str2double(get(handles.edit11,'String'));
t3= t3_get*(pi/180); %Conversion de grados a radianes
d3=str2double(get(handles.edit12,'String'));
a3=str2double(get(handles.edit13,'String'));
ap3_get=str2double(get(handles.edit14,'String'));
ap3= ap3_get*(pi/180); %Conversion de grados a radianes
j3=str2double(get(handles.edit15,'String'));
L(3)=Link([t3,d3,a3,ap3,j3]); %Crear el enlace
%Eslabon 4
t4_get=str2double(get(handles.edit16,'String'));
t4= t4_get*(pi/180); %Conversion de grados a radianes
d4=str2double(get(handles.edit17,'String'));
a4=str2double(get(handles.edit18,'String'));
ap4_get=str2double(get(handles.edit19,'String'));
ap4= ap4_get*(pi/180); %Conversion de grados a radianes
j4=str2double(get(handles.edit20,'String'));
L(4)=Link([t4,d4,a4,ap4,j4]); %Crear el enlace
%Eslabon 5
t5_get=str2double(get(handles.edit21,'String'));
t5= t5_get*(pi/180); %Conversion de grados a radianes
d5=str2double(get(handles.edit22,'String'));
a5=str2double(get(handles.edit23,'String'));
ap5_get=str2double(get(handles.edit24,'String'));
ap5= ap5_get*(pi/180); %Conversion de grados a radianes
j5=str2double(get(handles.edit25,'String'));
L(5)=Link([t5,d5,a5,ap5,j5]); %Crear el enlace
%Mostrar pantallas
```

```
set(handles.plot,'visible','on')
set(handles.robot,'visible','off')
%Dibujar el robot
global q
robot=SerialLink(L,'name','3GDL');
a_get=str2double(get(handles.q1,'String'))
a=a_get*(pi/180);%Convertir de gradosa radianes
b_get=str2double(get(handles.q2,'String'))
b=b_get*(pi/180);%Convertir de gradosa radianes
c_get=str2double(get(handles.q3,'String'))
c=c_get*(pi/180);%Convertir de gradosa radianes
d_get=str2double(get(handles.q4,'String'))
d=d_get*(pi/180);%Convertir de gradosa radianes
e_get=str2double(get(handles.q5,'String'))
e=e_get*(pi/180);%Convertir de gradosa radianes
q_5art=[a b c d e]
axes(handles.robot)
cla
set(handles.robot,'visible','off')
axes(handles.plot);
cla
robot.plot(q_5art);
%Matriz de transformación Homogénea
T1=[cosd(a_get),-cosd(ap1_get)*sind(a_get),sind(ap1_get)*sind(a_get), a1*cosd(a_get);
sind(a_get),cosd(ap1_get)*cosd(a_get),-sind(ap1_get)*cosd(a_get), a1*sind(a_get); 0,
sind(ap1_get),cosd(ap1_get),d1;0,0,0,1]
T2=[cosd(b_get),-cosd(ap2_get)*sind(b_get),sind(ap2_get)*sind(b_get), a2*cosd(b_get);
sind(b_get),cosd(ap2_get)*cosd(b_get),-sind(ap2_get)*cosd(b_get), a2*sind(b_get); 0,
sind(ap2_get),cosd(ap2_get),d2;0,0,0,1]
T3=[cosd(c_get),-cosd(ap3_get)*sind(c_get),sind(ap3_get)*sind(c_get), a3*cosd(c_get);
sind(c_get),cosd(ap3_get)*cosd(c_get),-sind(ap3_get)*cosd(c_get), a3*sind(c_get); 0,
sind(ap3_get),cosd(ap3_get),d3;0,0,0,1]
T4=[cosd(d_get),-cosd(ap4_get)*sind(d_get),sind(ap4_get)*sind(d_get), a4*cosd(d_get);
sind(d_get),cosd(ap4_get)*cosd(d_get),-sind(ap4_get)*cosd(d_get), a4*sind(d_get); 0,
sind(ap4_get),cosd(ap4_get),d4;0,0,0,1]
T5=[cosd(e_get),-cosd(ap5_get)*sind(e_get),sind(ap5_get)*sind(e_get), a5*cosd(e_get);
sind(e_get),cosd(ap5_get)*cosd(e_get),-sind(ap5_get)*cosd(e_get), a5*sind(e_get); 0,
sind(ap5_get),cosd(ap5_get),d5;0,0,0,1]
TT1*T2*T3*T4*T5
%Fila 1
set(handles.edit46,'String',T(1,1))
set(handles.edit47,'String',T(1,2))
set(handles.edit48,'String',T(1,3))
set(handles.edit49,'String',T(1,4))
%Fila 2
set(handles.edit50,'String',T(2,1))
set(handles.edit51,'String',T(2,2))
set(handles.edit52,'String',T(2,3))
set(handles.edit53,'String',T(2,4))
%Fila 3
set(handles.edit54,'String',T(3,1))
set(handles.edit55,'String',T(3,2))
set(handles.edit56,'String',T(3,3))
set(handles.edit57,'String',T(3,4))
%Fila 4
set(handles.edit58,'String',T(4,1))
set(handles.edit59,'String',T(4,2))
set(handles.edit60,'String',T(4,3))
set(handles.edit61,'String',T(4,4))
% --- Executes on button press in boton6gdl.
function boton6gdl_Callback(hObject, eventdata, handles)
set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
%Eslabon 1
t1_get=str2double(get(handles.edit1,'String'));
t1= t1_get*(pi/180); %Conversion de grados a radianes
d1=str2double(get(handles.edit2,'String'));
a1=str2double(get(handles.edit3,'String'));
ap1_get=str2double(get(handles.edit4,'String'));
ap1= ap1_get*(pi/180); %Conversion de grados a radianes
j1=str2double(get(handles.edit5,'String'));
```

```
L(1)=Link([t1,d1,a1,ap1,j1]); %Crear el enlace
%Eslabon 2
t2_get=str2double(get(handles.edit6,'String'));
t2= t2_get*(pi/180); %Conversion de grados a radianes
d2=str2double(get(handles.edit7,'String'));
a2=str2double(get(handles.edit8,'String'));
ap2_get=str2double(get(handles.edit9,'String'));
ap2= ap2_get*(pi/180); %Conversion de grados a radianes
j2=str2double(get(handles.edit10,'String'));
L(2)=Link([t2,d2,a2,ap2,j2]); %Crear el enlace
%Eslabon 3
t3_get=str2double(get(handles.edit11,'String'));
t3= t3_get*(pi/180); %Conversion de grados a radianes
d3=str2double(get(handles.edit12,'String'));
a3=str2double(get(handles.edit13,'String'));
ap3_get=str2double(get(handles.edit14,'String'));
ap3=ap3_get*(pi/180); %Conversion de grados a radianes
j3=str2double(get(handles.edit15,'String'));
L(3)=Link([t3,d3,a3,ap3,j3]); %Crear el enlace
%Eslabon 4
t4_get=str2double(get(handles.edit16,'String'));
t4= t4_get*(pi/180); %Conversion de grados a radianes
d4=str2double(get(handles.edit17,'String'));
a4=str2double(get(handles.edit18,'String'));
ap4_get=str2double(get(handles.edit19,'String'));
ap4= ap4_get*(pi/180); %Conversion de grados a radianes
j4=str2double(get(handles.edit20,'String'));
L(4)=Link([t4,d4,a4,ap4,j4]); %Crear el enlace
%Eslabon 5
t5_get=str2double(get(handles.edit21,'String'));
t5= t5_get*(pi/180); %Conversion de grados a radianes
d5=str2double(get(handles.edit22,'String'));
a5=str2double(get(handles.edit23,'String'));
ap5_get=str2double(get(handles.edit24,'String'));
ap5= ap5_get*(pi/180); %Conversion de grados a radianes
j5=str2double(get(handles.edit25,'String'));
L(5)=Link([t5,d5,a5,ap5,j5]); %Crear el enlace

%Eslabon 6
t6_get=str2double(get(handles.edit26,'String'));
t6= t6_get*(pi/180); %Conversion de grados a radianes
d6=str2double(get(handles.edit27,'String'));
a6=str2double(get(handles.edit28,'String'));
ap6_get=str2double(get(handles.edit29,'String'));
ap6= ap6_get*(pi/180); %Conversion de grados a radianes
j6=str2double(get(handles.edit30,'String'));
L(6)=Link([t6,d6,a6,ap6,j6]); %Crear el enlace
%Mostrar pantallas
set(handles.plot,'visible','on')
set(handles.robot,'visible','off')
%Dibujar el robot
global q
robot=SerialLink(L,'name','3GDL');
a_get=str2double(get(handles.q1,'String'))
a=a_get*(pi/180);%Convertir de gradosa radianes
b_get=str2double(get(handles.q2,'String'))
b=b_get*(pi/180);%Convertir de gradosa radianes
c_get=str2double(get(handles.q3,'String'))
c=c_get*(pi/180);%Convertir de gradosa radianes
d_get=str2double(get(handles.q4,'String'))
d=d_get*(pi/180);%Convertir de gradosa radianes
e_get=str2double(get(handles.q5,'String'))
e=e_get*(pi/180);%Convertir de gradosa radianes
f_get=str2double(get(handles.q6,'String'))
f=f_get*(pi/180);%Convertir de gradosa radianes
q_6art=[a b c d e f]
axes(handles.robot)
cla
set(handles.robot,'visible','off')
axes(handles.plot);
cla
```

```
robot.plot(q_6art);
a3
d3
ap3
c_get

%Matriz de Transformación Homogénea.
T1=[cosd(a_get),-cosd(ap1_get)*sind(a_get),sind(ap1_get)*sind(a_get), a1*cosd(a_get);
sind(a_get),cosd(ap1_get)*cosd(a_get),-sind(ap1_get)*cosd(a_get), a1*sind(a_get); 0,
sind(ap1_get),cosd(ap1_get),d1;0,0,0,1]
T2=[cosd(b_get),-cosd(ap2_get)*sind(b_get),sind(ap2_get)*sind(b_get), a2*cosd(b_get);
sind(b_get),cosd(ap2_get)*cosd(b_get),-sind(ap2_get)*cosd(b_get), a2*sind(b_get); 0,
sind(ap2_get),cosd(ap2_get),d2;0,0,0,1]
T3=[cosd(c_get),-cosd(ap3_get)*sind(c_get),sind(ap3_get)*sind(c_get), a3*cosd(c_get);
sind(c_get),cosd(ap3_get)*cosd(c_get),-sind(ap3_get)*cosd(c_get), a3*sind(c_get); 0,
sind(ap3_get),cosd(ap3_get),d3;0,0,0,1]
T4=[cosd(d_get),-cosd(ap4_get)*sind(d_get),sind(ap4_get)*sind(d_get), a4*cosd(d_get);
sind(d_get),cosd(ap4_get)*cosd(d_get),-sind(ap4_get)*cosd(d_get), a4*sind(d_get); 0,
sind(ap4_get),cosd(ap4_get),d4;0,0,0,1]
T5=[cosd(e_get),-cosd(ap5_get)*sind(e_get),sind(ap5_get)*sind(e_get), a5*cosd(e_get);
sind(e_get),cosd(ap5_get)*cosd(e_get),-sind(ap5_get)*cosd(e_get), a5*sind(e_get); 0,
sind(ap5_get),cosd(ap5_get),d5;0,0,0,1]
T6=[cosd(f_get),-cosd(ap6_get)*sind(f_get),sind(ap6_get)*sind(f_get), a6*cosd(f_get);
sind(f_get),cosd(ap6_get)*cosd(f_get),-sind(ap6_get)*cosd(f_get), a6*sind(f_get); 0,
sind(ap6_get),cosd(ap6_get),d6;0,0,0,1]
T=T1*T2*T3*T4*T5*T6
%Fila 1
set(handles.edit46,'String',T(1,1))
set(handles.edit47,'String',T(1,2))
set(handles.edit48,'String',T(1,3))
set(handles.edit49,'String',T(1,4))
%Fila 2
set(handles.edit50,'String',T(2,1))
set(handles.edit51,'String',T(2,2))
set(handles.edit52,'String',T(2,3))
set(handles.edit53,'String',T(2,4))
%Fila 3
set(handles.edit54,'String',T(3,1))
set(handles.edit55,'String',T(3,2))
set(handles.edit56,'String',T(3,3))
set(handles.edit57,'String',T(3,4))
%Fila 4
set(handles.edit58,'String',T(4,1))
set(handles.edit59,'String',T(4,2))
set(handles.edit60,'String',T(4,3))
set(handles.edit61,'String',T(4,4))

function edit1_Callback(hObject, eventdata, handles)
function edit1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit2_Callback(hObject, eventdata, handles)
function edit2_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit3_Callback(hObject, eventdata, handles)
function edit3_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit4_Callback(hObject, eventdata, handles)
function edit4_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit5_Callback(hObject, eventdata, handles)
```

```
function edit5_CreateFcn(hObject, eventdata, handles)
e a white background on Windows.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit6_Callback(hObject, eventdata, handles)
function edit6_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit7_Callback(hObject, eventdata, handles)
function edit7_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit8_Callback(hObject, eventdata, handles)
function edit8_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit9_Callback(hObject, eventdata, handles)
function edit9_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit10_Callback(hObject, eventdata, handles)
function edit10_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit11_Callback(hObject, eventdata, handles)
function edit11_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit12_Callback(hObject, eventdata, handles)
function edit12_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit13_Callback(hObject, eventdata, handles)
function edit13_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit14_Callback(hObject, eventdata, handles)
function edit14_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit15_Callback(hObject, eventdata, handles)
function edit15_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit16_Callback(hObject, eventdata, handles)
```

```
function edit16_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit17_Callback(hObject, eventdata, handles)
function edit17_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit18_Callback(hObject, eventdata, handles)
function edit18_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit19_Callback(hObject, eventdata, handles)
function edit19_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit20_Callback(hObject, eventdata, handles)
function edit20_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit21_Callback(hObject, eventdata, handles)
function edit21_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit22_Callback(hObject, eventdata, handles)
function edit22_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit23_Callback(hObject, eventdata, handles)
function edit23_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit24_Callback(hObject, eventdata, handles)
function edit24_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit25_Callback(hObject, eventdata, handles)
function edit25_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit26_Callback(hObject, eventdata, handles)
function edit26_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit27_Callback(hObject, eventdata, handles)
function edit27_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit28_Callback(hObject, eventdata, handles)
function edit28_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit29_Callback(hObject, eventdata, handles)
function edit29_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit30_Callback(hObject, eventdata, handles)
function edit30_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% --- Executes during object creation, after setting all properties.
function boton3gdl_CreateFcn(hObject, eventdata, handles)
function slider_q1_Callback(hObject, eventdata, handles)
global q
q(1)=get(handles.slider_q1,'value')
set(handles.q1,'String',num2str(q(1)));
function slider_q1_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
% --- Executes on slider movement.
function slider_q4_Callback(hObject, eventdata, handles)
global q
q(4)=get(handles.slider_q4,'value')
set(handles.q4,'String',num2str(q(4)));
function slider_q4_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
% --- Executes on slider movement.
function slider_q2_Callback(hObject, eventdata, handles)
global q
q(2)=get(handles.slider_q2,'value');
set(handles.q2,'String',num2str(q(2)));
function slider_q2_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end

% --- Executes on slider movement.
function slider_q5_Callback(hObject, eventdata, handles)
global q
q(5)=get(handles.slider_q5,'value')
set(handles.q5,'String',num2str(q(5)));
function slider_q5_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
function slider_q3_Callback(hObject, eventdata, handles)
global q
q(3)=get(handles.slider_q3,'value')
set(handles.q3,'String',num2str(q(3)));
function slider_q3_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
% --- Executes on slider movement.
function slider_q6_Callback(hObject, eventdata, handles)
global q
q(6)=get(handles.slider_q6,'value')
set(handles.q6,'String',num2str(q(6)));
function slider_q6_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
function q1_Callback(hObject, eventdata, handles)
function q1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function q4_Callback(hObject, eventdata, handles)
```

```
function q4_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function q2_Callback(hObject, eventdata, handles)
function q2_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function q5_Callback(hObject, eventdata, handles)
function q5_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function q3_Callback(hObject, eventdata, handles)
function q3_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function q6_Callback(hObject, eventdata, handles)
function q6_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% --- Executes on button press in robot1.
function robot1_Callback(hObject, eventdata, handles)
set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
set(handles.workspace,'visible','off')
set(handles.datos,'visible','on')
set(handles.matriz,'visible','off')
set(handles.cargar,'visible','on')
%Mostrar pantallas
set(handles.plot,'visible','on')
set(handles.robot,'visible','on')
mdl_jaco
axes(handles.plot);
cla
set(handles.q1,'String',round(str2double('4.71238')*57.2958))
set(handles.q2,'String',round(str2double('3.141592')*57.2958))
set(handles.q3,'String',round(str2double('3.141592')*57.2958))
set(handles.q4,'String',round(str2double('4.71238')*0))
set(handles.q5,'String',round(str2double('4.71238')*0))
set(handles.q6,'String',round(str2double('4.71238')*0))
qr=[4.712388980384690,3.141592653589793,3.141592653589793,0,0,0];
jaco.plot(qr)
global comparador
comparador=1
axes(handles.robot);
cla
jaco_img=imread('jaco.jpg');
imshow(jaco_img);
datos=char('Nombre: Jaco',char(13),'Compañia: Kinova',char(13),'GDL: 6','Configuración:
RRRRRR');
set(handles.datos,'String',datos)
% --- Executes on button press in robot2.
function robot2_Callback(hObject, eventdata, handles)
set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
set(handles.workspace,'visible','off')
set(handles.datos,'visible','on')
set(handles.matriz,'visible','off')
set(handles.cargar,'visible','on')
%Mostrar pantallas
set(handles.plot,'visible','on')
set(handles.robot,'visible','on')
mdl_fanuc10L
axes(handles.plot);
cla
set(handles.q1,'String',round(str2double('0')*57.2958))
set(handles.q2,'String',round(str2double('-1.57079')*57.2958))
```

```
set(handles.q3,'String',round(str2double('0')*57.2958))
set(handles.q4,'String',round(str2double('0')*57.2958))
set(handles.q5,'String',round(str2double('0')*57.2958))
set(handles.q6,'String',round(str2double('0')*57.2958))
q0=[0,-1.570796326794897,0,0,0,0];
R.plot(q0);
axes(handles.robot);
cla
fanuc_img=imread('fanuc10L.png');
imshow(fanuc_img);
datos=char('Nombre: FanucAM120iB/10L',char(13),'Compañia: Robot Works',char(13),'GDL:
6','Configuración: RRRRRR');
set(handles.datos,'String',datos)
global comparador
comparador=2;
% --- Executes on button press in robot3.
function robot3_Callback(hObject, eventdata, handles)
set(handles.workspace,'visible','off')
set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
set(handles.datos,'visible','on')
set(handles.matriz,'visible','off')
%Mostrar pantallas
set(handles.plot,'visible','on')
set(handles.robot,'visible','on')
set(handles.cargar,'visible','on')
global comparador
comparador=3;
mdl_irb140
axes(handles.plot);
cla
set(handles.q1,'String',round(str2double('0')*57.2958))
set(handles.q2,'String',round(str2double('-1.57079')*57.2958))
set(handles.q3,'String',round(str2double('3.1415')*57.2958))
set(handles.q4,'String',round(str2double('0')*57.2958))
set(handles.q5,'String',round(str2double('0')*57.2958))
set(handles.q6,'String',round(str2double('-1.57079')*57.2958))
q0=[0,-1.570796326794897,3.141592653589793,0,0,-1.570796326794897];
irb140.plot(q0);
axes(handles.robot);
cla
irb140_img=imread('irb140.png');
imshow(irb140_img);
datos=char('Nombre: IRB-140',char(13),'Compañia: ABB',char(13),'GDL: 6','Configuración:
RRRRRR');
set(handles.datos,'String',datos)
% --- Executes on button press in robot4.
function robot4_Callback(hObject, eventdata, handles)
set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
set(handles.workspace,'visible','off')
set(handles.datos,'visible','on')
set(handles.matriz,'visible','off')
%Mostrar pantallas
set(handles.plot,'visible','on')
set(handles.robot,'visible','on')
set(handles.cargar,'visible','on')

global comparador
comparador=4;
mdl_mico
axes(handles.plot);
cla
set(handles.q1,'String',round(str2double('4.71239')*57.2958))
set(handles.q2,'String',round(str2double('3.14159')*57.2958))
set(handles.q3,'String',round(str2double('3.14159')*57.2958))
set(handles.q4,'String',round(str2double('0')*57.2958))
set(handles.q5,'String',round(str2double('0')*57.2958))
set(handles.q6,'String',round(str2double('3.141592')*57.2958))
qr=[4.712388980384690,3.141592653589793,3.141592653589793,0,0,3.141592653589793];
mico.plot(qr);
```

```
axes(handles.robot);
cla
mico_img=imread('mico.jpg');
imshow(mico_img);
datos=char('Nombre: Mico',char(13),'Compañia: Kinova',char(13),'GDL: 6','Configuración:
RRRRRR');
set(handles.datos,'String',datos)
% --- Executes on button press in robot5.
function robot5_Callback(hObject, eventdata, handles)
set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
set(handles.workspace,'visible','off')
set(handles.datos,'visible','on')
set(handles.matriz,'visible','off')
%Mostrar pantallas
set(handles.plot,'visible','on')
set(handles.robot,'visible','on')
set(handles.cargar,'visible','on')
global comparador
comparador=5;
mdl_S4ABB2p8
axes(handles.plot);
cla
set(handles.q1,'String',round(str2double('0')*57.2958))
set(handles.q2,'String',round(str2double('-1.57079')*57.2958))
set(handles.q3,'String',round(str2double('0')*57.2958))
set(handles.q4,'String',round(str2double('0')*57.2958))
set(handles.q5,'String',round(str2double('0')*57.2958))
set(handles.q6,'String',round(str2double('-1.570796')*57.2958))
qr=[0,-1.570796326794897,0,0,0,-1.570796326794897];
s4.plot(qr);
axes(handles.robot);
cla
s4abb2_img=imread('S4 ABB 2.8.jpg');
imshow(s4abb2_img);
datos=char('Nombre: IRB6400',char(13),'Compañia: ABB',char(13),'GDL: 6','Configuración:
RRRRRR');
set(handles.datos,'String',datos)
% --- Executes on button press in robot6.
function robot6_Callback(hObject, eventdata, handles)
set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
set(handles.workspace,'visible','off')
set(handles.datos,'visible','on')
set(handles.matriz,'visible','off')
%Mostrar pantallas
set(handles.plot,'visible','on')
set(handles.robot,'visible','on')
set(handles.cargar,'visible','on')
global comparador
comparador=6;
mdl_puma560
axes(handles.plot);
cla
set(handles.q1,'String',round(str2double('0')*57.2958))
set(handles.q2,'String',round(str2double('0.7853')*57.2958))
set(handles.q3,'String',round(str2double('3.1415')*57.2958))
set(handles.q4,'String',round(str2double('0')*57.2958))
set(handles.q5,'String',round(str2double('0.7853')*57.2958))
set(handles.q6,'String',round(str2double('0')*57.2958))
qn=[0,0.785398163397448,3.141592653589793,0,0.785398163397448,0];
p560.plot(qn);
axes(handles.robot);
cla
puma560_img=imread('puma560.png');
imshow(puma560_img);
datos=char('Nombre: PUMA560',char(13),'Compañia: Unimation',char(13),'GDL:
6','Configuración: RRRRRR');
set(handles.datos,'String',datos)
% --- Executes on button press in robot7.
function robot7_Callback(hObject, eventdata, handles)
```

```
set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
set(handles.workspace,'visible','off')
set(handles.datos,'visible','on')
set(handles.matriz,'visible','off')
%Mostrar pantallas
set(handles.plot,'visible','on')
set(handles.robot,'visible','on')
set(handles.cargar,'visible','on')
global comparador
comparador=7;
mdl_ur10
axes(handles.plot);
cla
set(handles.q1,'String',round(str2double('3.1415')*57.2958))
set(handles.q2,'String',round(str2double('0')*57.2958))
set(handles.q3,'String',round(str2double('0')*57.2958))
set(handles.q4,'String',round(str2double('0')*57.2958))
set(handles.q5,'String',round(str2double('1.5709')*57.2958))
set(handles.q6,'String',round(str2double('0')*57.2958))
qr=[3.141592653589793,0,0,0,1.570796326794897,0];
ur10.plot(qr);
axes(handles.robot);
cla
ur10_img=imread('ur10.png');
imshow(ur10_img);

datos=char('Nombre: UR10 (Cobot)',char(13),'Compañia: Universal Robots',char(13),'GDL:
6','Configuración: RRRRRR');
set(handles.datos,'String',datos)
% --- Executes on button press in robot8.
function robot8_Callback(hObject, eventdata, handles)
set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
set(handles.workspace,'visible','off')
set(handles.datos,'visible','on')
set(handles.matriz,'visible','off')
%Mostrar pantallas
set(handles.plot,'visible','on')
set(handles.robot,'visible','on')
set(handles.cargar,'visible','on')
global comparador
comparador=8;
mdl_ur3
axes(handles.plot);
cla
set(handles.q1,'String',round(str2double('3.1415')*57.2958))
set(handles.q2,'String',round(str2double('0')*57.2958))
set(handles.q3,'String',round(str2double('0')*57.2958))
set(handles.q4,'String',round(str2double('0')*57.2958))
set(handles.q5,'String',round(str2double('1.5707')*57.2958))
set(handles.q6,'String',round(str2double('0')*57.2958))
qr=[3.141592653589793,0,0,0,1.570796326794897,0];
ur3.plot(qr);
axes(handles.robot);
cla
ur3_img=imread('ur3.png');
imshow(ur3_img);
datos=char('Nombre: UR3 (Cobot)',char(13),'Compañia: Universal Robots',char(13),'GDL:
6','Configuración: RRRRRR');
set(handles.datos,'String',datos)
% --- Executes on button press in robot9.
function robot9_Callback(hObject, eventdata, handles)
set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
set(handles.workspace,'visible','off')
set(handles.datos,'visible','on')
set(handles.matriz,'visible','off')
%Mostrar pantallas
set(handles.plot,'visible','on')
set(handles.robot,'visible','on')
```

```
set(handles.cargar,'visible','on')
xmax= 2;
xmin=-2;
ymax= 2;
ymin=-2;
zmax= 2;
zmin= -2;
global comparador
comparador=9;
mdl_stanford
axes(handles.plot);
cla
set(handles.q1,'String',round(str2double('0')*57.2958))
set(handles.q2,'String',round(str2double('1.57079')*57.2958))
set(handles.q3,'String',str2double('1.10'))
set(handles.q4,'String',round(str2double('0')*57.2958))
set(handles.q5,'String',round(str2double('0')*57.2958))
set(handles.q6,'String',round(str2double('0')*57.2958))
qz=[0,1.5708,1.10,0,0,0];
stanf.plot(qz,'workspace',[xmin xmax ymin ymax zmin zmax]);
axes(handles.robot);
cla
stanford_img=imread('stanford.jpg');
imshow(stanford_img);
datos=char('Nombre: Stanford',char(13),'Compañia: Robot Works',char(13),'GDL:
6','Configuración: RRPRRR');
set(handles.datos,'String',datos)
% --- Executes on button press in robot10.
function robot10_Callback(hObject, eventdata, handles)
set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
set(handles.workspace,'visible','off')
set(handles.datos,'visible','on')
set(handles.matriz,'visible','off')
set(handles.cargar,'visible','on')
%Mostrar pantallas
set(handles.plot,'visible','on')
set(handles.robot,'visible','on')
global comparador
comparador=10;
mdl_M16
axes(handles.plot);
cla
set(handles.q1,'String',round(str2double('0')*57.2958))
set(handles.q2,'String',round(str2double('-1.5707')*57.2958))
set(handles.q3,'String',round(str2double('0')*57.2958))
set(handles.q4,'String',round(str2double('0')*57.2958))
set(handles.q5,'String',round(str2double('-3.1415')*57.2958))
set(handles.q6,'String',round(str2double('3.1415')*57.2958))
qd=[0,-1.570796326794897,0,0,-3.141592653589793,3.141592653589793];
m16.plot(qd);
axes(handles.robot);
cla
m16_img=imread('m16.png');
imshow(m16_img);
datos=char('Nombre: M16',char(13),'Compañia: Fanuc',char(13),'GDL: 6','Configuración:
RRRRRR');
set(handles.datos,'String',datos)
% --- Executes on button press in robot11.
function robot11_Callback(hObject, eventdata, handles)

set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
set(handles.workspace,'visible','off')
set(handles.datos,'visible','on')
set(handles.matriz,'visible','off')
%Mostrar pantallas
set(handles.plot,'visible','on')
set(handles.robot,'visible','on')
set(handles.cargar,'visible','on')
global comparador
```

```
comparador=11;
mdl_KR5
axes(handles.plot);
cla
set(handles.q1,'String',round(str2double('0.7853')*57.2958))
set(handles.q2,'String',round(str2double('-1.0471')*57.2958))
set(handles.q3,'String',round(str2double('0.7839')*57.2958))
set(handles.q4,'String',round(str2double('0.5235')*57.2958))
set(handles.q5,'String',round(str2double('0.7839')*57.2958))
set(handles.q6,'String',round(str2double('0.5235')*57.2958))
qk1=[0.785398163397448,1.047197551196598,0.785398163397448,0.523598775598299,0.7853981633974
48,0.523598775598299];
KR5.plot(qk1);
axes(handles.robot);
cla
kr5_img=imread('KR5.png');
imshow(kr5_img);
datos=char('Nombre: KR5',char(13),'Compañia: KUKA',char(13),'GDL: 6','Configuración:
RRRRRR');
set(handles.datos,'String',datos)
% --- Executes on button press in robot12.
function robot12_Callback(hObject, eventdata, handles)
set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
set(handles.workspace,'visible','off')
set(handles.datos,'visible','on')
set(handles.matriz,'visible','off')

%Mostrar pantallas
set(handles.plot,'visible','on')
set(handles.robot,'visible','on')
set(handles.cargar,'visible','on')
global comparador
comparador=12;

mdl_motomanHP6
axes(handles.plot);
cla
set(handles.q1,'String',round(str2double('0')*57.2958))
set(handles.q2,'String',round(str2double('-1.57079')*57.2958))
set(handles.q3,'String',round(str2double('0')*57.2958))
set(handles.q4,'String',round(str2double('0')*57.2958))
set(handles.q5,'String',round(str2double('-1.5707')*57.2958))
set(handles.q6,'String',round(str2double('0')*57.2958))
q0=[0,-1.570796326794897,0,0,-1.570796326794897,0];
hp6.plot(q0);

axes(handles.robot);
cla
hp6_img=imread('Motomanhp6.jpg');
imshow(hp6_img);

datos=char('Nombre: Motoman HP6',char(13),'Compañia: Yaskawa',char(13),'GDL:
6','Configuración: RRRRRR');
set(handles.datos,'String',datos)
% --- Executes on button press in pushbutton11.
function pushbutton11_Callback(hObject, eventdata, handles)
function pushbutton12_Callback(hObject, eventdata, handles)
function pushbutton13_Callback(hObject, eventdata, handles)
function pushbutton14_Callback(hObject, eventdata, handles)
function robotsp_CreateFcn(hObject, eventdata, handles)
function boton3GDLP_Callback(hObject, eventdata, handles)
set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
global validador1 validador2 validador3 validador4
global validador5 validador6
validador1=str2double(get(handles.edit5,'String'));
validador2=str2double(get(handles.edit10,'String'));
validador3=str2double(get(handles.edit15,'String'));
%Eslabon 1
t1_get=str2double(get(handles.edit1,'String'));
```

```
t1= t1_get*(pi/180); %Conversion de grados a radianes
d1=str2double(get(handles.edit2,'String'));
a1=str2double(get(handles.edit3,'String'));
ap1_get=str2double(get(handles.edit4,'String'));
ap1= ap1_get*(pi/180); %Conversion de grados a radianes
j1=str2double(get(handles.edit5,'String'));
L(1)=Link([t1,d1,a1,ap1,j1]); %Crear el enlace
%Eslabon 2
t2_get=str2double(get(handles.edit6,'String'));
t2= t2_get*(pi/180); %Conversion de grados a radianes
d2=str2double(get(handles.edit7,'String'));
a2=str2double(get(handles.edit8,'String'));
ap2_get=str2double(get(handles.edit9,'String'));
ap2= ap2_get*(pi/180); %Conversion de grados a radianes
j2=str2double(get(handles.edit10,'String'));
L(2)=Link([t2,d2,a2,ap2,j2]); %Crear el enlace
%Eslabon 3
t3_get=str2double(get(handles.edit11,'String'));
t3= t3_get*(pi/180); %Conversion de grados a radianes
d3=str2double(get(handles.edit12,'String'));
a3=str2double(get(handles.edit13,'String'));
ap3_get=str2double(get(handles.edit14,'String'));
ap3=ap3_get*(pi/180); %Conversion de grados a radianes
j3=str2double(get(handles.edit15,'String'));
L(3)=Link([t3,d3,a3,ap3,j3]); %Crear el enlace
%Mostrar pantallas
set(handles.plot,'visible','on')
set(handles.robot,'visible','off')

%Workspace
global xmin xmax ymin ymax zmin zmax
xmin=str2double(get(handles.xmin,'String'));
xmax=str2double(get(handles.xmax,'String'));
ymin=str2double(get(handles.ymin,'String'));
ymax=str2double(get(handles.ymax,'String'));
zmin=str2double(get(handles.zmin,'String'));
zmax=str2double(get(handles.zmax,'String'));

%Dibujar el robot
robot=SerialLink(L,'name','3GDL');
a=str2double(get(handles.q1,'String'));
b=str2double(get(handles.q2,'String'));
c=str2double(get(handles.q3,'String'));
q_3art=[a b c];
axes(handles.robot);
cla
set(handles.robot,'visible','off');
axes(handles.plot);
cla
robot.plot(q_3art,'workspace',[xmin xmax ymin ymax zmin zmax]);

%Ploter del robot
robot=SerialLink(L,'name','3GDL');
a_get=str2double(get(handles.q1,'String'));
if (validador1==0)
    a=a_get*(pi/180); %Conversion de grados a radianes
else
    a=a_get;
end
b_get=str2double(get(handles.q2,'String'));
if (validador2==0)
    b=b_get*(pi/180); %Conversion de grados a radianes
else
    b=b_get;
end
c_get=str2double(get(handles.q3,'String'));
if (validador3==0)
    c=c_get*(pi/180); %Conversion de grados a radianes
else
    c=c_get;
end
```

```
q_3art=[a b c]
axes(handles.robot)
cla
set(handles.robot,'visible','off')
axes(handles.plot);
cla
robot.plot(q_3art,'workspace',[xmin xmax ymin ymax zmin zmax]);
%Matriz de Transformación Homegénea
if (validador1==0)
    T1=[cosd(a_get),-cosd(ap1_get)*sind(a_get),sind(ap1_get)*sind(a_get), a1*cosd(a_get);
sind(a_get),cosd(ap1_get)*cosd(a_get),-sind(ap1_get)*cosd(a_get), a1*sind(a_get); 0,
sind(ap1_get),cosd(ap1_get),d1;0,0,0,1]
else
    T1=[cosd(t1_get),-cosd(ap1_get)*sind(t1_get),sind(ap1_get)*sind(t1_get),
a1*cosd(t1_get); sind(t1_get),cosd(ap1_get)*cosd(t1_get),-sind(ap1_get)*cosd(t1_get),
a1*sind(t1_get); 0, sind(ap1_get),cosd(ap1_get),a_get;0,0,0,1]
end
if (validador2==0)
    T2=[cosd(b_get),-cosd(ap2_get)*sind(b_get),sind(ap2_get)*sind(b_get), a2*cosd(b_get);
sind(b_get),cosd(ap2_get)*cosd(b_get),-sind(ap2_get)*cosd(b_get), a2*sind(b_get); 0,
sind(ap2_get),cosd(ap2_get),d2;0,0,0,1]
else
    T2=[cosd(t2_get),-cosd(ap2_get)*sind(t2_get),sind(ap2_get)*sind(t2_get),
a2*cosd(t2_get); sind(t2_get),cosd(ap2_get)*cosd(t2_get),-sind(ap2_get)*cosd(t2_get),
a2*sind(t2_get); 0, sind(ap2_get),cosd(ap2_get),b_get;0,0,0,1]
end
if (validador3==0)
    T3=[cosd(c_get),-cosd(ap3_get)*sind(c_get),sind(ap3_get)*sind(c_get), a3*cosd(c_get);
sind(c_get),cosd(ap3_get)*cosd(c_get),-sind(ap3_get)*cosd(c_get), a3*sind(c_get); 0,
sind(ap3_get),cosd(ap3_get),d3;0,0,0,1]
else
    T3=[cosd(t3_get),-cosd(ap3_get)*sind(t3_get),sind(ap3_get)*sind(t3_get),
a3*cosd(t3_get); sind(t3_get),cosd(ap3_get)*cosd(t3_get),-sind(ap3_get)*cosd(t3_get),
a3*sind(t3_get); 0, sind(ap3_get),cosd(ap3_get),c_get;0,0,0,1]
end
T=T1*T2*T3
%Fila 1
set(handles.edit46,'String',T(1,1))
set(handles.edit47,'String',T(1,2))
set(handles.edit48,'String',T(1,3))
set(handles.edit49,'String',T(1,4))
%Fila 2
set(handles.edit50,'String',T(2,1))
set(handles.edit51,'String',T(2,2))
set(handles.edit52,'String',T(2,3))
set(handles.edit53,'String',T(2,4))
%Fila 3
set(handles.edit54,'String',T(3,1))
set(handles.edit55,'String',T(3,2))
set(handles.edit56,'String',T(3,3))
set(handles.edit57,'String',T(3,4))
%Fila 4
set(handles.edit58,'String',T(4,1))
set(handles.edit59,'String',T(4,2))
set(handles.edit60,'String',T(4,3))
set(handles.edit61,'String',T(4,4))
% --- Executes on button press in boton4GDLP.
function boton4GDLP_Callback(hObject, eventdata, handles)
set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
global validador1 validador2 validador3 validador4
validador1=str2double(get(handles.edit5,'String'));
validador2=str2double(get(handles.edit10,'String'));
validador3=str2double(get(handles.edit15,'String'));
validador4=str2double(get(handles.edit20,'String'));
%Eslabon 1
t1_get=str2double(get(handles.edit1,'String'));
t1= t1_get*(pi/180); %Conversion de grados a radianes
d1=str2double(get(handles.edit2,'String'));
a1=str2double(get(handles.edit3,'String'));
ap1_get=str2double(get(handles.edit4,'String'));
```

```
ap1= ap1_get*(pi/180); %Conversion de grados a radianes
j1=str2double(get(handles.edit5,'String'));
L(1)=Link([t1,d1,a1,ap1,j1]); %Crear el enlace
%Eslabon 2
t2_get=str2double(get(handles.edit6,'String'));
t2= t2_get*(pi/180); %Conversion de grados a radianes
d2=str2double(get(handles.edit7,'String'));
a2=str2double(get(handles.edit8,'String'));
ap2_get=str2double(get(handles.edit9,'String'));
ap2= ap2_get*(pi/180); %Conversion de grados a radianes
j2=str2double(get(handles.edit10,'String'));
L(2)=Link([t2,d2,a2,ap2,j2]); %Crear el enlace
%Eslabon 3
t3_get=str2double(get(handles.edit11,'String'));
t3= t3_get*(pi/180); %Conversion de grados a radianes
d3=str2double(get(handles.edit12,'String'));
a3=str2double(get(handles.edit13,'String'));
ap3_get=str2double(get(handles.edit14,'String'));
ap3=ap3_get*(pi/180); %Conversion de grados a radianes
j3=str2double(get(handles.edit15,'String'));
L(3)=Link([t3,d3,a3,ap3,j3]); %Crear el enlace
%Eslabon 4
t4_get=str2double(get(handles.edit16,'String'));
t4= t4_get*(pi/180); %Conversion de grados a radianes
d4=str2double(get(handles.edit17,'String'));
a4=str2double(get(handles.edit18,'String'));
ap4_get=str2double(get(handles.edit19,'String'));
ap4= ap4_get*(pi/180); %Conversion de grados a radianes
j4=str2double(get(handles.edit20,'String'));
L(4)=Link([t4,d4,a4,ap4,j4]); %Crear el enlace
%Mostrar pantallas
set(handles.plot,'visible','on')
set(handles.robot,'visible','off')
%Workspace
global xmin xmax ymin ymax zmin zmax
xmin=str2double(get(handles.xmin,'String'));
xmax=str2double(get(handles.xmax,'String'));
ymin=str2double(get(handles.ymin,'String'));
ymax=str2double(get(handles.ymax,'String'));
zmin=str2double(get(handles.zmin,'String'));
zmax=str2double(get(handles.zmax,'String'));
%Dibujar el robot
robot=SerialLink(L,'name','3GDL');
a=str2double(get(handles.q1,'String'));
b=str2double(get(handles.q2,'String'));
c=str2double(get(handles.q3,'String'));
d=str2double(get(handles.q4,'String'));
q_4art=[a b c d];
axes(handles.robot);
cla
set(handles.robot,'visible','off');
axes(handles.plot);
cla
robot.plot(q_4art,'workspace',[xmin xmax ymin ymax zmin zmax]);
%Ploter del robot
robot=SerialLink(L,'name','3GDL');
a_get=str2double(get(handles.q1,'String'));
if (validador1==0)
    a=a_get*(pi/180); %Conversion de grados a radianes
else
    a=a_get;
end
b_get=str2double(get(handles.q2,'String'));
if (validador2==0)
    b=b_get*(pi/180); %Conversion de grados a radianes
else
    b=b_get;
end

c_get=str2double(get(handles.q3,'String'));
if (validador3==0)
```

```
    c=c_get*(pi/180); %Conversion de grados a radianes
else
    c=c_get;
end
d_get=str2double(get(handles.q4,'String'))
if (validador4==0)
    d=d_get*(pi/180) %Conversion de grados a radianes
else
    d=d_get
end
q_4art=[a b c d]
axes(handles.robot)
cla
set(handles.robot,'visible','off')
axes(handles.plot);
cla
robot.plot(q_4art,'workspace',[xmin xmax ymin ymax zmin zmax]);
%Matriz de Transformación Homegénea
if (validador1==0)
    T1=[cosd(a_get),-cosd(ap1_get)*sind(a_get),sind(ap1_get)*sind(a_get), a1*cosd(a_get);
sind(a_get),cosd(ap1_get)*cosd(a_get),-sind(ap1_get)*cosd(a_get), a1*sind(a_get); 0,
sind(ap1_get),cosd(ap1_get),d1;0,0,0,1]
else
    T1=[cosd(t1_get),-cosd(ap1_get)*sind(t1_get),sind(ap1_get)*sind(t1_get),
a1*cosd(t1_get); sind(t1_get),cosd(ap1_get)*cosd(t1_get),-sind(ap1_get)*cosd(t1_get),
a1*sind(t1_get); 0, sind(ap1_get),cosd(ap1_get),a_get;0,0,0,1]
end

if (validador2==0)
    T2=[cosd(b_get),-cosd(ap2_get)*sind(b_get),sind(ap2_get)*sind(b_get), a2*cosd(b_get);
sind(b_get),cosd(ap2_get)*cosd(b_get),-sind(ap2_get)*cosd(b_get), a2*sind(b_get); 0,
sind(ap2_get),cosd(ap2_get),d2;0,0,0,1]
else
    T2=[cosd(t2_get),-cosd(ap2_get)*sind(t2_get),sind(ap2_get)*sind(t2_get),
a2*cosd(t2_get); sind(t2_get),cosd(ap2_get)*cosd(t2_get),-sind(ap2_get)*cosd(t2_get),
a2*sind(t2_get); 0, sind(ap2_get),cosd(ap2_get),b_get;0,0,0,1]
end
if (validador3==0)
    T3=[cosd(c_get),-cosd(ap3_get)*sind(c_get),sind(ap3_get)*sind(c_get), a3*cosd(c_get);
sind(c_get),cosd(ap3_get)*cosd(c_get),-sind(ap3_get)*cosd(c_get), a3*sind(c_get); 0,
sind(ap3_get),cosd(ap3_get),d3;0,0,0,1]
else
    T3=[cosd(t3_get),-cosd(ap3_get)*sind(t3_get),sind(ap3_get)*sind(t3_get),
a3*cosd(t3_get); sind(t3_get),cosd(ap3_get)*cosd(t3_get),-sind(ap3_get)*cosd(t3_get),
a3*sind(t3_get); 0, sind(ap3_get),cosd(ap3_get),c_get;0,0,0,1]
end
if (validador4==0)
    T4=[cosd(d_get),-cosd(ap4_get)*sind(d_get),sind(ap4_get)*sind(d_get), a4*cosd(d_get);
sind(d_get),cosd(ap4_get)*cosd(d_get),-sind(ap4_get)*cosd(d_get), a4*sind(d_get); 0,
sind(ap4_get),cosd(ap4_get),d4;0,0,0,1]
else
    T4=[cosd(t4_get),-cosd(ap4_get)*sind(t4_get),sind(ap4_get)*sind(t4_get),
a4*cosd(t4_get); sind(t4_get),cosd(ap4_get)*cosd(t4_get),-sind(ap4_get)*cosd(t4_get),
a4*sind(t4_get); 0, sind(ap4_get),cosd(ap4_get),d_get;0,0,0,1]
end
T=T1*T2*T3*T4
%Fila 1
set(handles.edit46,'String',T(1,1))
set(handles.edit47,'String',T(1,2))
set(handles.edit48,'String',T(1,3))
set(handles.edit49,'String',T(1,4))
%Fila 2
set(handles.edit50,'String',T(2,1))
set(handles.edit51,'String',T(2,2))
set(handles.edit52,'String',T(2,3))
set(handles.edit53,'String',T(2,4))
%Fila 3
set(handles.edit54,'String',T(3,1))
set(handles.edit55,'String',T(3,2))
set(handles.edit56,'String',T(3,3))
set(handles.edit57,'String',T(3,4))
```

```
%Fila 4
set(handles.edit58,'String',T(4,1))
set(handles.edit59,'String',T(4,2))
set(handles.edit60,'String',T(4,3))
set(handles.edit61,'String',T(4,4))
% --- Executes on button press in boton5GDLP.
function boton5GDLP_Callback(hObject, eventdata, handles)
global validador1 validador2 validador3 validador4
global validador5 validador6
validador1=str2double(get(handles.edit5,'String'))
validador2=str2double(get(handles.edit10,'String'))
validador3=str2double(get(handles.edit15,'String'))
validador4=str2double(get(handles.edit20,'String'))
validador5=str2double(get(handles.edit25,'String'))
validador6=str2double(get(handles.edit30,'String'))
set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
%Eslabon 1
t1_get=str2double(get(handles.edit1,'String'));
t1= t1_get*(pi/180); %Conversion de grados a radianes
d1=str2double(get(handles.edit2,'String'));
a1=str2double(get(handles.edit3,'String'));
ap1_get=str2double(get(handles.edit4,'String'));
ap1= ap1_get*(pi/180); %Conversion de grados a radianes
j1=str2double(get(handles.edit5,'String'));
L(1)=Link([t1,d1,a1,ap1,j1]); %Crear el enlace
%Eslabon 2
t2_get=str2double(get(handles.edit6,'String'));
t2= t2_get*(pi/180); %Conversion de grados a radianes
d2=str2double(get(handles.edit7,'String'));
a2=str2double(get(handles.edit8,'String'));
ap2_get=str2double(get(handles.edit9,'String'));
ap2= ap2_get*(pi/180); %Conversion de grados a radianes
j2=str2double(get(handles.edit10,'String'));
L(2)=Link([t2,d2,a2,ap2,j2]); %Crear el enlace
%Eslabon 3
t3_get=str2double(get(handles.edit11,'String'));
t3= t3_get*(pi/180); %Conversion de grados a radianes
d3=str2double(get(handles.edit12,'String'));
a3=str2double(get(handles.edit13,'String'));
ap3_get=str2double(get(handles.edit14,'String'));
ap3=ap3_get*(pi/180); %Conversion de grados a radianes
j3=str2double(get(handles.edit15,'String'));
L(3)=Link([t3,d3,a3,ap3,j3]); %Crear el enlace
%Eslabon 4
t4_get=str2double(get(handles.edit16,'String'));
t4= t4_get*(pi/180); %Conversion de grados a radianes
d4=str2double(get(handles.edit17,'String'));
a4=str2double(get(handles.edit18,'String'));
ap4_get=str2double(get(handles.edit19,'String'));
ap4= ap4_get*(pi/180); %Conversion de grados a radianes
j4=str2double(get(handles.edit20,'String'));
L(4)=Link([t4,d4,a4,ap4,j4]); %Crear el enlace
%Eslabon 5
t5_get=str2double(get(handles.edit21,'String'));
t5= t5_get*(pi/180); %Conversion de grados a radianes
d5=str2double(get(handles.edit22,'String'));
a5=str2double(get(handles.edit23,'String'));
ap5_get=str2double(get(handles.edit24,'String'));
ap5= ap5_get*(pi/180); %Conversion de grados a radianes
j5=str2double(get(handles.edit25,'String'));
L(5)=Link([t5,d5,a5,ap5,j5]); %Crear el enlace
%Mostrar pantallas
set(handles.plot,'visible','on')
set(handles.robot,'visible','off')
%Workspace
global xmin xmax ymin ymax zmin zmax
xmin=str2double(get(handles.xmin,'String'))
xmax=str2double(get(handles.xmax,'String'))
ymin=str2double(get(handles.ymin,'String'))
ymax=str2double(get(handles.ymax,'String'))
```

```
zmin=str2double(get(handles.zmin,'String'))
zmax=str2double(get(handles.zmax,'String'))
%Dibujar el robot
robot=SerialLink(L,'name','3GDL')
a_get=str2double(get(handles.q1,'String'))
if (validador1==0)
    a=a_get*(pi/180) %Conversion de grados a radianes
else
    a=a_get
end
b_get=str2double(get(handles.q2,'String'))
if (validador2==0)
    b=b_get*(pi/180) %Conversion de grados a radianes
else
    b=b_get
end
c_get=str2double(get(handles.q3,'String'))
if (validador3==0)
    c=c_get*(pi/180) %Conversion de grados a radianes
else
    c=c_get
end
d_get=str2double(get(handles.q4,'String'))
if (validador4==0)
    d=d_get*(pi/180) %Conversion de grados a radianes
else
    d=d_get
end
e_get=str2double(get(handles.q5,'String'))
if (validador5==0)
   e=e_get*(pi/180) %Conversion de grados a radianes
else
    e=e_get
end
q_5art=[a b c d e]
axes(handles.robot)
cla
set(handles.robot,'visible','off')
axes(handles.plot);
cla
robot.plot(q_5art,'workspace',[xmin xmax ymin ymax zmin zmax]);
%Matriz de Transformación Homogénea
if (validador1==0)
    T1=[cosd(a_get),-cosd(ap1_get)*sind(a_get),sind(ap1_get)*sind(a_get), a1*cosd(a_get);
sind(a_get),cosd(ap1_get)*cosd(a_get),-sind(ap1_get)*cosd(a_get), a1*sind(a_get); 0,
sind(ap1_get),cosd(ap1_get),d1;0,0,0,1]
else
    T1=[cosd(t1_get),-cosd(ap1_get)*sind(t1_get),sind(ap1_get)*sind(t1_get),
a1*cosd(t1_get); sind(t1_get),cosd(ap1_get)*cosd(t1_get),-sind(ap1_get)*cosd(t1_get),
a1*sind(t1_get); 0, sind(ap1_get),cosd(ap1_get),a_get;0,0,0,1]
end
if (validador2==0)
    T2=[cosd(b_get),-cosd(ap2_get)*sind(b_get),sind(ap2_get)*sind(b_get), a2*cosd(b_get);
sind(b_get),cosd(ap2_get)*cosd(b_get),-sind(ap2_get)*cosd(b_get), a2*sind(b_get); 0,
sind(ap2_get),cosd(ap2_get),d2;0,0,0,1]
else
    T2=[cosd(t2_get),-cosd(ap2_get)*sind(t2_get),sind(ap2_get)*sind(t2_get),
a2*cosd(t2_get); sind(t2_get),cosd(ap2_get)*cosd(t2_get),-sind(ap2_get)*cosd(t2_get),
a2*sind(t2_get); 0, sind(ap2_get),cosd(ap2_get),b_get;0,0,0,1]
end
if (validador3==0)
    T3=[cosd(c_get),-cosd(ap3_get)*sind(c_get),sind(ap3_get)*sind(c_get), a3*cosd(c_get);
sind(c_get),cosd(ap3_get)*cosd(c_get),-sind(ap3_get)*cosd(c_get), a3*sind(c_get); 0,
sind(ap3_get),cosd(ap3_get),d3;0,0,0,1]
else
    T3=[cosd(t3_get),-cosd(ap3_get)*sind(t3_get),sind(ap3_get)*sind(t3_get),
a3*cosd(t3_get); sind(t3_get),cosd(ap3_get)*cosd(t3_get),-sind(ap3_get)*cosd(t3_get),
a3*sind(t3_get); 0, sind(ap3_get),cosd(ap3_get),c_get;0,0,0,1]
end
if (validador4==0)
```

```
    T4=[cosd(d_get),-cosd(ap4_get)*sind(d_get),sind(ap4_get)*sind(d_get), a4*cosd(d_get);
sind(d_get),cosd(ap4_get)*cosd(d_get),-sind(ap4_get)*cosd(d_get), a4*sind(d_get); 0,
sind(ap4_get),cosd(ap4_get),d4;0,0,0,1]
else
    T4=[cosd(t4_get),-cosd(ap4_get)*sind(t4_get),sind(ap4_get)*sind(t4_get),
a4*cosd(t4_get); sind(t4_get),cosd(ap4_get)*cosd(t4_get),-sind(ap4_get)*cosd(t4_get),
a4*sind(t4_get); 0, sind(ap4_get),cosd(ap4_get),d_get;0,0,0,1]
end
if(validador5==0)
    T5=[cosd(e_get),-cosd(ap5_get)*sind(e_get),sind(ap5_get)*sind(e_get), a5*cosd(e_get);
sind(e_get),cosd(ap5_get)*cosd(e_get),-sind(ap5_get)*cosd(e_get), a5*sind(e_get); 0,
sind(ap5_get),cosd(ap5_get),d5;0,0,0,1]
else
    T5=[cosd(t5_get),-cosd(ap5_get)*sind(t5_get),sind(ap5_get)*sind(t5_get),
a5*cosd(t5_get); sind(t5_get),cosd(ap5_get)*cosd(t5_get),-sind(ap5_get)*cosd(t5_get),
a5*sind(t5_get); 0, sind(ap5_get),cosd(ap5_get),e_get;0,0,0,1]
end
T=T1*T2*T3*T4*T5
%Fila 1
set(handles.edit46,'String',T(1,1))
set(handles.edit47,'String',T(1,2))
set(handles.edit48,'String',T(1,3))
set(handles.edit49,'String',T(1,4))
%Fila 2
set(handles.edit50,'String',T(2,1))
set(handles.edit51,'String',T(2,2))
set(handles.edit52,'String',T(2,3))
set(handles.edit53,'String',T(2,4))
%Fila 3
set(handles.edit54,'String',T(3,1))
set(handles.edit55,'String',T(3,2))
set(handles.edit56,'String',T(3,3))
set(handles.edit57,'String',T(3,4))
%Fila 4
set(handles.edit58,'String',T(4,1))
set(handles.edit59,'String',T(4,2))
set(handles.edit60,'String',T(4,3))
set(handles.edit61,'String',T(4,4))
% --- Executes on button press in boton6GDLP.
function boton6GDLP_Callback(hObject, eventdata, handles)
global validador1 validador2 validador3 validador4
global validador5 validador6
validador1=str2double(get(handles.edit5,'String'))
validador2=str2double(get(handles.edit10,'String'))
validador3=str2double(get(handles.edit15,'String'))
validador4=str2double(get(handles.edit20,'String'))
validador5=str2double(get(handles.edit25,'String'))
validador6=str2double(get(handles.edit30,'String'))
set(handles.titulo,'visible','off')
set(handles.instrucciones,'visible','off')
%Eslabon 1
t1_get=str2double(get(handles.edit1,'String'));
t1= t1_get*(pi/180); %Conversion de grados a radianes
d1=str2double(get(handles.edit2,'String'));
a1=str2double(get(handles.edit3,'String'));
ap1_get=str2double(get(handles.edit4,'String'));
ap1= ap1_get*(pi/180); %Conversion de grados a radianes
j1=str2double(get(handles.edit5,'String'));
L(1)=Link([t1,d1,a1,ap1,j1]); %Crear el enlace
%Eslabon 2
t2_get=str2double(get(handles.edit6,'String'));
t2= t2_get*(pi/180); %Conversion de grados a radianes
d2=str2double(get(handles.edit7,'String'));
a2=str2double(get(handles.edit8,'String'));
ap2_get=str2double(get(handles.edit9,'String'));
ap2= ap2_get*(pi/180); %Conversion de grados a radianes
j2=str2double(get(handles.edit10,'String'));
L(2)=Link([t2,d2,a2,ap2,j2]); %Crear el enlace
%Eslabon 3
t3_get=str2double(get(handles.edit11,'String'));
t3= t3_get*(pi/180); %Conversion de grados a radianes
```

```
d3=str2double(get(handles.edit12,'String'));
a3=str2double(get(handles.edit13,'String'));
ap3_get=str2double(get(handles.edit14,'String'));
ap3=ap3_get*(pi/180); %Conversion de grados a radianes
j3=str2double(get(handles.edit15,'String'));
L(3)=Link([t3,d3,a3,ap3,j3]); %Crear el enlace
%Eslabon 4
t4_get=str2double(get(handles.edit16,'String'));
t4= t4_get*(pi/180); %Conversion de grados a radianes
d4=str2double(get(handles.edit17,'String'));
a4=str2double(get(handles.edit18,'String'));
ap4_get=str2double(get(handles.edit19,'String'));
ap4= ap4_get*(pi/180); %Conversion de grados a radianes
j4=str2double(get(handles.edit20,'String'));
L(4)=Link([t4,d4,a4,ap4,j4]); %Crear el enlace
%Eslabon 5
t5_get=str2double(get(handles.edit21,'String'));
t5= t5_get*(pi/180); %Conversion de grados a radianes
d5=str2double(get(handles.edit22,'String'));
a5=str2double(get(handles.edit23,'String'));
ap5_get=str2double(get(handles.edit24,'String'));
ap5= ap5_get*(pi/180); %Conversion de grados a radianes
j5=str2double(get(handles.edit25,'String'));
L(5)=Link([t5,d5,a5,ap5,j5]); %Crear el enlace
%Eslabon 6
t6_get=str2double(get(handles.edit26,'String'));
t6= t6_get*(pi/180); %Conversion de grados a radianes
d6=str2double(get(handles.edit27,'String'));
a6=str2double(get(handles.edit28,'String'));
ap6_get=str2double(get(handles.edit29,'String'));
ap6= ap6_get*(pi/180); %Conversion de grados a radianes
j6=str2double(get(handles.edit30,'String'));
L(6)=Link([t6,d6,a6,ap6,j6]); %Crear el enlace
%Mostrar pantallas
set(handles.plot,'visible','on')
set(handles.robot,'visible','off')
%Workspace
global xmin xmax ymin ymax zmin zmax
xmin=str2double(get(handles.xmin,'String'))
xmax=str2double(get(handles.xmax,'String'))
ymin=str2double(get(handles.ymin,'String'))
ymax=str2double(get(handles.ymax,'String'))
zmin=str2double(get(handles.zmin,'String'))
zmax=str2double(get(handles.zmax,'String'))
%Dibujar el robot
robot=SerialLink(L,'name','3GDL')
a_get=str2double(get(handles.q1,'String'))
if (validador1==0)
    a=a_get*(pi/180) %Conversion de grados a radianes
else
    a=a_get
end
b_get=str2double(get(handles.q2,'String'))
if (validador2==0)
    b=b_get*(pi/180) %Conversion de grados a radianes
else
    b=b_get
end
c_get=str2double(get(handles.q3,'String'))
if (validador3==0)
    c=c_get*(pi/180) %Conversion de grados a radianes
else
    c=c_get
end
d_get=str2double(get(handles.q4,'String'))
if (validador4==0)
    d=d_get*(pi/180) %Conversion de grados a radianes
else
    d=d_get
end
e_get=str2double(get(handles.q5,'String'))
```

```
if (validador5==0)
   e=e_get*(pi/180) %Conversion de grados a radianes
else
    e=e_get
end
f_get=str2double(get(handles.q6,'String'))
if (validador6==0)
    f=f_get*(pi/180) %Conversion de grados a radianes
else
    f=f_get
end
q_6art=[a b c d e f]
axes(handles.robot)
cla
set(handles.robot,'visible','off')
axes(handles.plot);
cla
robot.plot(q_6art,'workspace',[xmin xmax ymin ymax zmin zmax]);
%Matriz de Transformación Homogénea
if (validador1==0)
    T1=[cosd(a_get),-cosd(ap1_get)*sind(a_get),sind(ap1_get)*sind(a_get), a1*cosd(a_get);
sind(a_get),cosd(ap1_get)*cosd(a_get),-sind(ap1_get)*cosd(a_get), a1*sind(a_get); 0,
sind(ap1_get),cosd(ap1_get),d1;0,0,0,1]
else
    T1=[cosd(t1_get),-cosd(ap1_get)*sind(t1_get),sind(ap1_get)*sind(t1_get),
a1*cosd(t1_get); sind(t1_get),cosd(ap1_get)*cosd(t1_get),-sind(ap1_get)*cosd(t1_get),
a1*sind(t1_get); 0, sind(ap1_get),cosd(ap1_get),a_get;0,0,0,1]
end

if (validador2==0)
    T2=[cosd(b_get),-cosd(ap2_get)*sind(b_get),sind(ap2_get)*sind(b_get), a2*cosd(b_get);
sind(b_get),cosd(ap2_get)*cosd(b_get),-sind(ap2_get)*cosd(b_get), a2*sind(b_get); 0,
sind(ap2_get),cosd(ap2_get),d2;0,0,0,1]
else
    T2=[cosd(t2_get),-cosd(ap2_get)*sind(t2_get),sind(ap2_get)*sind(t2_get),
a2*cosd(t2_get); sind(t2_get),cosd(ap2_get)*cosd(t2_get),-sind(ap2_get)*cosd(t2_get),
a2*sind(t2_get); 0, sind(ap2_get),cosd(ap2_get),b_get;0,0,0,1]
end
if (validador3==0)
    T3=[cosd(c_get),-cosd(ap3_get)*sind(c_get),sind(ap3_get)*sind(c_get), a3*cosd(c_get);
sind(c_get),cosd(ap3_get)*cosd(c_get),-sind(ap3_get)*cosd(c_get), a3*sind(c_get); 0,
sind(ap3_get),cosd(ap3_get),d3;0,0,0,1]
else
    T3=[cosd(t3_get),-cosd(ap3_get)*sind(t3_get),sind(ap3_get)*sind(t3_get),
a3*cosd(t3_get); sind(t3_get),cosd(ap3_get)*cosd(t3_get),-sind(ap3_get)*cosd(t3_get),
a3*sind(t3_get); 0, sind(ap3_get),cosd(ap3_get),c_get;0,0,0,1]
end

if (validador4==0)
    T4=[cosd(d_get),-cosd(ap4_get)*sind(d_get),sind(ap4_get)*sind(d_get), a4*cosd(d_get);
sind(d_get),cosd(ap4_get)*cosd(d_get),-sind(ap4_get)*cosd(d_get), a4*sind(d_get); 0,
sind(ap4_get),cosd(ap4_get),d4;0,0,0,1]
else
    T4=[cosd(t4_get),-cosd(ap4_get)*sind(t4_get),sind(ap4_get)*sind(t4_get),
a4*cosd(t4_get); sind(t4_get),cosd(ap4_get)*cosd(t4_get),-sind(ap4_get)*cosd(t4_get),
a4*sind(t4_get); 0, sind(ap4_get),cosd(ap4_get),d_get;0,0,0,1]
end

if(validador5==0)
    T5=[cosd(e_get),-cosd(ap5_get)*sind(e_get),sind(ap5_get)*sind(e_get), a5*cosd(e_get);
sind(e_get),cosd(ap5_get)*cosd(e_get),-sind(ap5_get)*cosd(e_get), a5*sind(e_get); 0,
sind(ap5_get),cosd(ap5_get),d5;0,0,0,1]
else
    T5=[cosd(t5_get),-cosd(ap5_get)*sind(t5_get),sind(ap5_get)*sind(t5_get),
a5*cosd(t5_get); sind(t5_get),cosd(ap5_get)*cosd(t5_get),-sind(ap5_get)*cosd(t5_get),
a5*sind(t5_get); 0, sind(ap5_get),cosd(ap5_get),e_get;0,0,0,1]
end
if(validador6==0)
    T6=[cosd(f_get),-cosd(ap6_get)*sind(f_get),sind(ap6_get)*sind(f_get), a6*cosd(f_get);
sind(f_get),cosd(ap6_get)*cosd(f_get),-sind(ap6_get)*cosd(f_get), a6*sind(f_get); 0,
sind(ap6_get),cosd(ap6_get),d6;0,0,0,1]
```

```
else
    T6=[cosd(t6_get),-cosd(ap6_get)*sind(t6_get),sind(ap6_get)*sind(t6_get),
a6*cosd(t6_get); sind(t6_get),cosd(ap6_get)*cosd(t6_get),-sind(ap6_get)*cosd(t6_get),
a6*sind(t6_get); 0, sind(ap6_get),cosd(ap6_get),f_get;0,0,0,1]
end
T=T1*T2*T3*T4*T5*T6
%Fila 1
set(handles.edit46,'String',T(1,1))
set(handles.edit47,'String',T(1,2))
set(handles.edit48,'String',T(1,3))
set(handles.edit49,'String',T(1,4))
%Fila 2
set(handles.edit50,'String',T(2,1))
set(handles.edit51,'String',T(2,2))
set(handles.edit52,'String',T(2,3))
set(handles.edit53,'String',T(2,4))
%Fila 3
set(handles.edit54,'String',T(3,1))
set(handles.edit55,'String',T(3,2))
set(handles.edit56,'String',T(3,3))
set(handles.edit57,'String',T(3,4))
%Fila 4
set(handles.edit58,'String',T(4,1))
set(handles.edit59,'String',T(4,2))
set(handles.edit60,'String',T(4,3))
set(handles.edit61,'String',T(4,4))
function xmin_Callback(hObject, eventdata, handles)
function xmin_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function xmax_Callback(hObject, eventdata, handles)
function xmax_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function ymin_Callback(hObject, eventdata, handles)
function ymin_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function ymax_Callback(hObject, eventdata, handles)
function ymax_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function zmin_Callback(hObject, eventdata, handles)
function zmin_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function zmax_Callback(hObject, eventdata, handles)
function zmax_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit43_Callback(hObject, eventdata, handles)
function edit43_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit44_Callback(hObject, eventdata, handles)
function edit44_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit46_Callback(hObject, eventdata, handles)
function edit46_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end
function edit47_Callback(hObject, eventdata, handles)
function edit47_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit48_Callback(hObject, eventdata, handles)
function edit48_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit49_Callback(hObject, eventdata, handles)
function edit49_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit50_Callback(hObject, eventdata, handles)
function edit50_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit51_Callback(hObject, eventdata, handles)
function edit51_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit52_Callback(hObject, eventdata, handles)
function edit52_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit53_Callback(hObject, eventdata, handles)
function edit53_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit54_Callback(hObject, eventdata, handles)
function edit54_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit55_Callback(hObject, eventdata, handles)
function edit55_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit56_Callback(hObject, eventdata, handles)
function edit56_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit57_Callback(hObject, eventdata, handles)
function edit57_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit58_Callback(hObject, eventdata, handles)
function edit58_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit59_Callback(hObject, eventdata, handles)
function edit59_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit60_Callback(hObject, eventdata, handles)
function edit60_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end
function edit61_Callback(hObject, eventdata, handles)
function edit61_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% --- Executes on button press in cargar.
function cargar_Callback(hObject, eventdata, handles)
global comparador  q1 q2 q3 q4 q5 q6
if (comparador==1)
    set(handles.titulo,'visible','off')
    set(handles.instrucciones,'visible','off')
    set(handles.workspace,'visible','off')
    set(handles.datos,'visible','on')
    set(handles.matriz,'visible','off')
    %Mostrar pantallas
    set(handles.plot,'visible','on')
    set(handles.robot,'visible','on')
    mdl_jaco
    axes(handles.plot);
    cla
    q1_get=str2num(get(handles.q1,'String'));
    q1= q1_get*(pi/180); %Conversion de grados a radianes
    q2_get=str2num(get(handles.q2,'String'));
    q2= q2_get*(pi/180); %Conversion de grados a radianes
    q3_get=str2num(get(handles.q3,'String'));
    q3= q3_get*(pi/180); %Conversion de grados a radianes
    q4_get=str2num(get(handles.q4,'String'));
    q4= q4_get*(pi/180); %Conversion de grados a radianes
    q5_get=str2num(get(handles.q5,'String'));
    q5= q5_get*(pi/180); %Conversion de grados a radianes
    q6_get=str2num(get(handles.q6,'String'));
    q6= q6_get*(pi/180); %Conversion de grados a radianes
    q0=[q1,q2,q3,q4,q5,q6];
    jaco.plot(q0)
    axes(handles.robot);
    cla
    jaco_img=imread('jaco.jpg');
    imshow(jaco_img);
    datos=char('Nombre: Jaco',char(13),'Compañia: Kinova',char(13),'GDL: 6','Configuración:
RRRRRR');
    set(handles.datos,'String',datos)
elseif (comparador==2)
    set(handles.titulo,'visible','off')
    set(handles.instrucciones,'visible','off')
    set(handles.workspace,'visible','off')
    set(handles.datos,'visible','on')
    set(handles.matriz,'visible','off')
    %Mostrar pantallas
    set(handles.plot,'visible','on')
    set(handles.robot,'visible','on')
    mdl_fanuc10L
    axes(handles.plot);
    cla
    q1_get=str2num(get(handles.q1,'String'));
    q1= q1_get*(pi/180); %Conversion de grados a radianes
    q2_get=str2num(get(handles.q2,'String'));
    q2= q2_get*(pi/180); %Conversion de grados a radianes
    q3_get=str2num(get(handles.q3,'String'));
    q3= q3_get*(pi/180); %Conversion de grados a radianes
    q4_get=str2num(get(handles.q4,'String'));
    q4= q4_get*(pi/180); %Conversion de grados a radianes
    q5_get=str2num(get(handles.q5,'String'));
    q5= q5_get*(pi/180); %Conversion de grados a radianes
    q6_get=str2num(get(handles.q6,'String'));
    q6= q6_get*(pi/180); %Conversion de grados a radianes
    q0=[q1,q2,q3,q4,q5,q6];
    R.plot(q0)
    axes(handles.robot);
    cla
```

```
    fanuc_img=imread('fanuc10L.png');
    imshow(fanuc_img);
    datos=char('Nombre: FanucAM120iB/10L',char(13),'Compañia: Robot Works',char(13),'GDL:
6','Configuración: RRRRRR');
    set(handles.datos,'String',datos)
elseif (comparador==3)
    set(handles.workspace,'visible','off')
    set(handles.titulo,'visible','off')
    set(handles.instrucciones,'visible','off')
    set(handles.datos,'visible','on')
    set(handles.matriz,'visible','off')
    %Mostrar pantallas
    set(handles.plot,'visible','on')
    set(handles.robot,'visible','on')
    mdl_irb140
    axes(handles.plot);
    cla
    q1_get=str2num(get(handles.q1,'String'));
    q1= q1_get*(pi/180); %Conversion de grados a radianes
    q2_get=str2num(get(handles.q2,'String'));
    q2= q2_get*(pi/180); %Conversion de grados a radianes
    q3_get=str2num(get(handles.q3,'String'));
    q3= q3_get*(pi/180); %Conversion de grados a radianes
    q4_get=str2num(get(handles.q4,'String'));
    q4= q4_get*(pi/180); %Conversion de grados a radianes
    q5_get=str2num(get(handles.q5,'String'));
    q5= q5_get*(pi/180); %Conversion de grados a radianes
    q6_get=str2num(get(handles.q6,'String'));
    q6= q6_get*(pi/180); %Conversion de grados a radianes
    q0=[q1,q2,q3,q4,q5,q6];
    irb140.plot(q0);
    axes(handles.robot);
    cla
    irb140_img=imread('irb140.png');
    imshow(irb140_img);
    datos=char('Nombre: IRB-140',char(13),'Compañia: ABB',char(13),'GDL: 6','Configuración:
RRRRRR');
    set(handles.datos,'String',datos)
elseif(comparador==4)
    set(handles.titulo,'visible','off')
    set(handles.instrucciones,'visible','off')
    set(handles.workspace,'visible','off')
    set(handles.datos,'visible','on')
    set(handles.matriz,'visible','off')
    %Mostrar pantallas
    set(handles.plot,'visible','on')
    set(handles.robot,'visible','on')
    mdl_mico
    axes(handles.plot);
    cla
    q1_get=str2num(get(handles.q1,'String'));
    q1= q1_get*(pi/180); %Conversion de grados a radianes
    q2_get=str2num(get(handles.q2,'String'));
    q2= q2_get*(pi/180); %Conversion de grados a radianes
    q3_get=str2num(get(handles.q3,'String'));
    q3= q3_get*(pi/180); %Conversion de grados a radianes
    q4_get=str2num(get(handles.q4,'String'));
    q4= q4_get*(pi/180); %Conversion de grados a radianes
    q5_get=str2num(get(handles.q5,'String'));
    q5= q5_get*(pi/180); %Conversion de grados a radianes
    q6_get=str2num(get(handles.q6,'String'));
    q6= q6_get*(pi/180); %Conversion de grados a radianes
    q0=[q1,q2,q3,q4,q5,q6];
    mico.plot(q0);
    axes(handles.robot);
    cla
    mico_img=imread('mico.jpg');
    imshow(mico_img);
    datos=char('Nombre: Mico',char(13),'Compañia: Kinova',char(13),'GDL: 6','Configuración:
RRRRRR');
    set(handles.datos,'String',datos)
```

```
elseif(comparador==5)
    set(handles.titulo,'visible','off')
    set(handles.instrucciones,'visible','off')
    set(handles.workspace,'visible','off')
    set(handles.datos,'visible','on')
    set(handles.matriz,'visible','off')
    %Mostrar pantallas
    set(handles.plot,'visible','on')
    set(handles.robot,'visible','on')
    mdl_S4ABB2p8
    axes(handles.plot);
    cla
    q1_get=str2num(get(handles.q1,'String'));
    q1= q1_get*(pi/180); %Conversion de grados a radianes
    q2_get=str2num(get(handles.q2,'String'));
    q2= q2_get*(pi/180); %Conversion de grados a radianes
    q3_get=str2num(get(handles.q3,'String'));
    q3= q3_get*(pi/180); %Conversion de grados a radianes
    q4_get=str2num(get(handles.q4,'String'));
    q4= q4_get*(pi/180); %Conversion de grados a radianes
    q5_get=str2num(get(handles.q5,'String'));
    q5= q5_get*(pi/180); %Conversion de grados a radianes
    q6_get=str2num(get(handles.q6,'String'));
    q6= q6_get*(pi/180); %Conversion de grados a radianes
    q0=[q1,q2,q3,q4,q5,q6];
    s4.plot(q0);
    axes(handles.robot);
    cla
    s4abb2_img=imread('S4 ABB 2.8.jpg');
    imshow(s4abb2_img);
    datos=char('Nombre: IRB6400',char(13),'Compañia: ABB',char(13),'GDL: 6','Configuración:
RRRRRR');
    set(handles.datos,'String',datos)
elseif(comparador==6)
    set(handles.titulo,'visible','off')
    set(handles.instrucciones,'visible','off')
    set(handles.workspace,'visible','off')
    set(handles.datos,'visible','on')
    set(handles.matriz,'visible','off')
    %Mostrar pantallas
    set(handles.plot,'visible','on')
    set(handles.robot,'visible','on')
    mdl_puma560
    axes(handles.plot);
    cla
    q1_get=str2num(get(handles.q1,'String'));
    q1= q1_get*(pi/180); %Conversion de grados a radianes
    q2_get=str2num(get(handles.q2,'String'));
    q2= q2_get*(pi/180); %Conversion de grados a radianes
    q3_get=str2num(get(handles.q3,'String'));
    q3= q3_get*(pi/180); %Conversion de grados a radianes
    q4_get=str2num(get(handles.q4,'String'));
    q4= q4_get*(pi/180); %Conversion de grados a radianes
    q5_get=str2num(get(handles.q5,'String'));
    q5= q5_get*(pi/180); %Conversion de grados a radianes
    q6_get=str2num(get(handles.q6,'String'));
    q6= q6_get*(pi/180); %Conversion de grados a radianes
    q0=[q1,q2,q3,q4,q5,q6];
    p560.plot(q0);
    axes(handles.robot);
    cla
    puma560_img=imread('puma560.png');
    imshow(puma560_img);
    datos=char('Nombre: PUMA560',char(13),'Compañia: Unimation',char(13),'GDL:
6','Configuración: RRRRRR');
    set(handles.datos,'String',datos
elseif(comparador==7)
    set(handles.titulo,'visible','off')
    set(handles.instrucciones,'visible','off')
    set(handles.workspace,'visible','off')
    set(handles.datos,'visible','on')
```

```
    set(handles.matriz,'visible','off')
    %Mostrar pantallas
    set(handles.plot,'visible','on')
    set(handles.robot,'visible','on')
    mdl_ur10
    axes(handles.plot);
    cla
    q1_get=str2num(get(handles.q1,'String'));
    q1= q1_get*(pi/180); %Conversion de grados a radianes
    q2_get=str2num(get(handles.q2,'String'));
    q2= q2_get*(pi/180); %Conversion de grados a radianes
    q3_get=str2num(get(handles.q3,'String'));
    q3= q3_get*(pi/180); %Conversion de grados a radianes
    q4_get=str2num(get(handles.q4,'String'));
    q4= q4_get*(pi/180); %Conversion de grados a radianes
    q5_get=str2num(get(handles.q5,'String'));
    q5= q5_get*(pi/180); %Conversion de grados a radianes
    q6_get=str2num(get(handles.q6,'String'));
    q6= q6_get*(pi/180); %Conversion de grados a radianes
    q0=[q1,q2,q3,q4,q5,q6];
    ur10.plot(q0);
    axes(handles.robot);
    cla
    ur10_img=imread('ur10.png');
    imshow(ur10_img);
    datos=char('Nombre: UR10 (Cobot)',char(13),'Compañia: Universal Robots',char(13),'GDL:
6','Configuración: RRRRRR');
    set(handles.datos,'String',datos)
elseif(comparador==8)
    set(handles.titulo,'visible','off')
    set(handles.instrucciones,'visible','off')
    set(handles.workspace,'visible','off')
    set(handles.datos,'visible','on')
    set(handles.matriz,'visible','off')
    %Mostrar pantallas
    set(handles.plot,'visible','on')
    set(handles.robot,'visible','on')
    mdl_ur3
    axes(handles.plot);
    cla
    q1_get=str2num(get(handles.q1,'String'));
    q1= q1_get*(pi/180); %Conversion de grados a radianes
    q2_get=str2num(get(handles.q2,'String'));
    q2= q2_get*(pi/180); %Conversion de grados a radianes
    q3_get=str2num(get(handles.q3,'String'));
    q3= q3_get*(pi/180); %Conversion de grados a radianes
    q4_get=str2num(get(handles.q4,'String'));
    q4= q4_get*(pi/180); %Conversion de grados a radianes
    q5_get=str2num(get(handles.q5,'String'));
    q5= q5_get*(pi/180); %Conversion de grados a radianes
    q6_get=str2num(get(handles.q6,'String'));
    q6= q6_get*(pi/180); %Conversion de grados a radianes
    q0=[q1,q2,q3,q4,q5,q6];
    ur3.plot(q0);
    axes(handles.robot);
    cla
    ur3_img=imread('ur3.png');
    imshow(ur3_img);
    datos=char('Nombre: UR3 (Cobot)',char(13),'Compañia: Universal Robots',char(13),'GDL:
6','Configuración: RRRRRR');
    set(handles.datos,'String',datos)
elseif(comparador==9)
    set(handles.titulo,'visible','off')
    set(handles.instrucciones,'visible','off')
    set(handles.workspace,'visible','off')
    set(handles.datos,'visible','on')
    set(handles.matriz,'visible','off')
    %Mostrar pantallas
    set(handles.plot,'visible','on')
    set(handles.robot,'visible','on')
    xmax= 2;
```

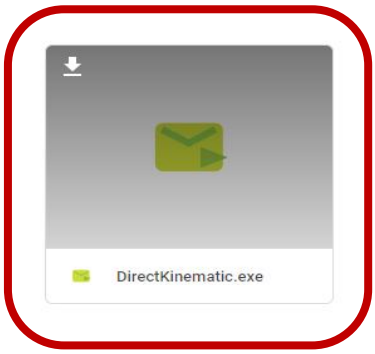
```
    xmin=-2;
    ymax= 2;
    ymin=-2;
    zmax= 2;
    zmin= -2;
    mdl_stanford
    axes(handles.plot);
    cla
    q1_get=str2num(get(handles.q1,'String'));
    q1= q1_get*(pi/180); %Conversion de grados a radianes
    q2_get=str2num(get(handles.q2,'String'));
    q2= q2_get*(pi/180); %Conversion de grados a radianes
    q3=str2num(get(handles.q3,'String'));
%     q3= q3_get*(pi/180); %Conversion de grados a radianes
    q4_get=str2num(get(handles.q4,'String'));
    q4= q4_get*(pi/180); %Conversion de grados a radianes
    q5_get=str2num(get(handles.q5,'String'));
    q5= q5_get*(pi/180); %Conversion de grados a radianes
    q6_get=str2num(get(handles.q6,'String'));
    q6= q6_get*(pi/180); %Conversion de grados a radianes
    q0=[q1,q2,q3,q4,q5,q6];
    stanf.plot(q0,'workspace',[xmin xmax ymin ymax zmin zmax]);
    axes(handles.robot);
    cla
    stanford_img=imread('stanford.jpg');
    imshow(stanford_img);
    datos=char('Nombre: Stanford',char(13),'Compañia: Robot Works',char(13),'GDL:
6','Configuración: RRPRRR');
    set(handles.datos,'String',datos)
elseif(comparador==10)
    set(handles.titulo,'visible','off')
    set(handles.instrucciones,'visible','off')
    set(handles.workspace,'visible','off')
    set(handles.datos,'visible','on')
    set(handles.matriz,'visible','off')
    %Mostrar pantallas
    set(handles.plot,'visible','on')
    set(handles.robot,'visible','on')
    mdl_M16
    axes(handles.plot);
    cla
    q1_get=str2num(get(handles.q1,'String'));
    q1= q1_get*(pi/180); %Conversion de grados a radianes
    q2_get=str2num(get(handles.q2,'String'));
    q2= q2_get*(pi/180); %Conversion de grados a radianes
    q3_get=str2num(get(handles.q3,'String'));
    q3= q3_get*(pi/180); %Conversion de grados a radianes
    q4_get=str2num(get(handles.q4,'String'));
    q4= q4_get*(pi/180); %Conversion de grados a radianes
    q5_get=str2num(get(handles.q5,'String'));
    q5= q5_get*(pi/180); %Conversion de grados a radianes
    q6_get=str2num(get(handles.q6,'String'));
    q6= q6_get*(pi/180); %Conversion de grados a radianes
    q0=[q1,q2,q3,q4,q5,q6];
    m16.plot(q0);
    axes(handles.robot);
    cla
    m16_img=imread('m16.png');
    imshow(m16_img);
    datos=char('Nombre: M16',char(13),'Compañia: Fanuc',char(13),'GDL: 6','Configuración:
RRRRRR');
    set(handles.datos,'String',datos)
elseif(comparador==11)
    set(handles.titulo,'visible','off')
    set(handles.instrucciones,'visible','off')
    set(handles.workspace,'visible','off')
    set(handles.datos,'visible','on')
    set(handles.matriz,'visible','off')
    %Mostrar pantallas
    set(handles.plot,'visible','on')
    set(handles.robot,'visible','on')
```

```
    mdl_KR5
    axes(handles.plot);
    cla
    q1_get=str2num(get(handles.q1,'String'));
    q1= q1_get*(pi/180); %Conversion de grados a radianes
    q2_get=str2num(get(handles.q2,'String'));
    q2= q2_get*(pi/180); %Conversion de grados a radianes
    q3_get=str2num(get(handles.q3,'String'));
    q3= q3_get*(pi/180); %Conversion de grados a radianes
    q4_get=str2num(get(handles.q4,'String'));
    q4= q4_get*(pi/180); %Conversion de grados a radianes
    q5_get=str2num(get(handles.q5,'String'));
    q5= q5_get*(pi/180); %Conversion de grados a radianes
    q6_get=str2num(get(handles.q6,'String'));
    q6= q6_get*(pi/180); %Conversion de grados a radianes
    q0=[q1,q2,q3,q4,q5,q6];
    KR5.plot(q0);
    axes(handles.robot);
    cla
    kr5_img=imread('KR5.png');
    imshow(kr5_img);
    datos=char('Nombre: KR5',char(13),'Compañia: KUKA',char(13),'GDL: 6','Configuración:
RRRRRR');
    set(handles.datos,'String',datos)
elseif(comparador==12)
    set(handles.titulo,'visible','off')
    set(handles.instrucciones,'visible','off')
    set(handles.workspace,'visible','off')
    set(handles.datos,'visible','on')
    set(handles.matriz,'visible','off')
    %Mostrar pantallas
    set(handles.plot,'visible','on')
    set(handles.robot,'visible','on')
    mdl_motomanHP6
    axes(handles.plot);
    cla
    q1_get=str2num(get(handles.q1,'String'));
    q1= q1_get*(pi/180); %Conversion de grados a radianes
    q2_get=str2num(get(handles.q2,'String'));
    q2= q2_get*(pi/180); %Conversion de grados a radianes
    q3_get=str2num(get(handles.q3,'String'));
    q3= q3_get*(pi/180); %Conversion de grados a radianes
    q4_get=str2num(get(handles.q4,'String'));
    q4= q4_get*(pi/180); %Conversion de grados a radianes
    q5_get=str2num(get(handles.q5,'String'));
    q5= q5_get*(pi/180); %Conversion de grados a radianes
    q6_get=str2num(get(handles.q6,'String'));
    q6= q6_get*(pi/180); %Conversion de grados a radianes
    q0=[q1,q2,q3,q4,q5,q6];
    hp6.plot(q0);
    axes(handles.robot);
    cla
    hp6_img=imread('Motomanhp6.jpg');
    imshow(hp6_img);
    datos=char('Nombre: Motoman HP6',char(13),'Compañia: Yaskawa',char(13),'GDL:
6','Configuración: RRRRRR');
    set(handles.datos,'String',datos)
end
% --- Executes on button press in ant.
function ant_Callback(hObject, eventdata, handles)
global txt1 txt2 txt3 txt4 count
  count=count-1
if (count==0)
    set(handles.instrucciones,'String',txt1);
    set(handles.ant,'visible','off')
elseif (count==1)
    set(handles.instrucciones,'String',txt2);
    set(handles.ant,'visible','on')
elseif (count==2)
    set(handles.instrucciones,'String',txt3);
elseif (count==3)
```

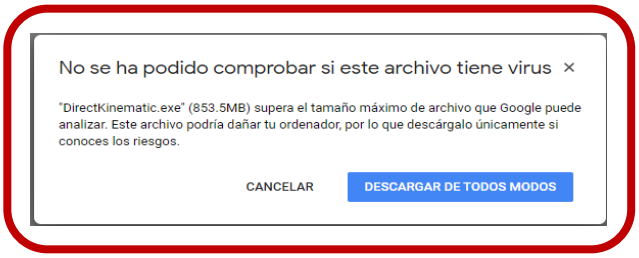
```
    set(handles.instrucciones,'String',txt4);
    set(handles.sig,'visible','on')
end
% --- Executes on button press in sig.
function sig_Callback(hObject, eventdata, handles)
global txt2 txt3 txt4 txt5 count
if (count==0)
    count=count+1
set(handles.instrucciones,'String',txt2);
set(handles.ant,'visible','on')
elseif (count==1)
    count=count+1
    set(handles.instrucciones,'String',txt3);
    set(handles.ant,'visible','on')
elseif (count==2)
    count=count+1
    set(handles.instrucciones,'String',txt4);
elseif (count==3)
    count=count+1
    set(handles.instrucciones,'String',txt5);
    set(handles.sig,'visible','off')
end
% --- Executes on button press in listo.
function listo_Callback(hObject, eventdata, handles)
global count txt1
count=0;
set(handles.instrucciones,'String',txt1);
%Desaparecer
set(handles.instrucciones,'visible','off')
set(handles.sig,'visible','off')
set(handles.ant,'visible','off')
set(handles.titulo,'visible','off')
set(handles.listo,'visible','off')
%Aparecer
set(handles.parametros,'visible','on')
set(handles.load_robots,'visible','on')
set(handles.robots,'visible','on')
set(handles.articulaciones,'visible','on')
set(handles.matriz,'visible','on')
% --- Executes on button press in b_instruc.
function b_instruc_Callback(hObject, eventdata, handles)
%Aparecer
set(handles.instrucciones,'visible','on')
set(handles.sig,'visible','on')
set(handles.ant,'visible','off')
set(handles.titulo,'visible','on')
set(handles.listo,'visible','on')
%Desaparecer
set(handles.parametros,'visible','off')
set(handles.load_robots,'visible','off')
set(handles.robots,'visible','off')
set(handles.articulaciones,'visible','off')
set(handles.matriz,'visible','off')
set(handles.datos,'visible','off')
set(handles.cargar,'visible','off')
set(handles.robotsr,'visible','off')
set(handles.robotsp,'visible','off')
set(handles.workspace,'visible','off')
axes(handles.robot)
cla
axes(handles.plot)
cla
set(handles.plot,'visible','off')
set(handles.robot,'visible','off')
```

Apéndice 8 Pasos para instalar la aplicación número 1

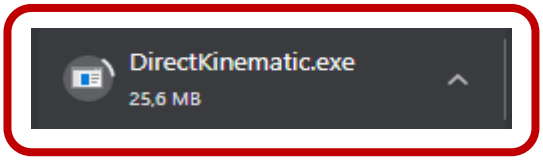
1. Ingrese a su ordenador de preferencia, en este caso se ocupó Google Chrome.
2. Dirigirse a la dirección: https://drive.google.com/drive/folders/1Qb55kJgKhb8YzfF8CopZv4vAK_kfpRMt?usp=sharing
3. Sobre el archivo DirectKinematic.exe dar clic en la flecha para descargar



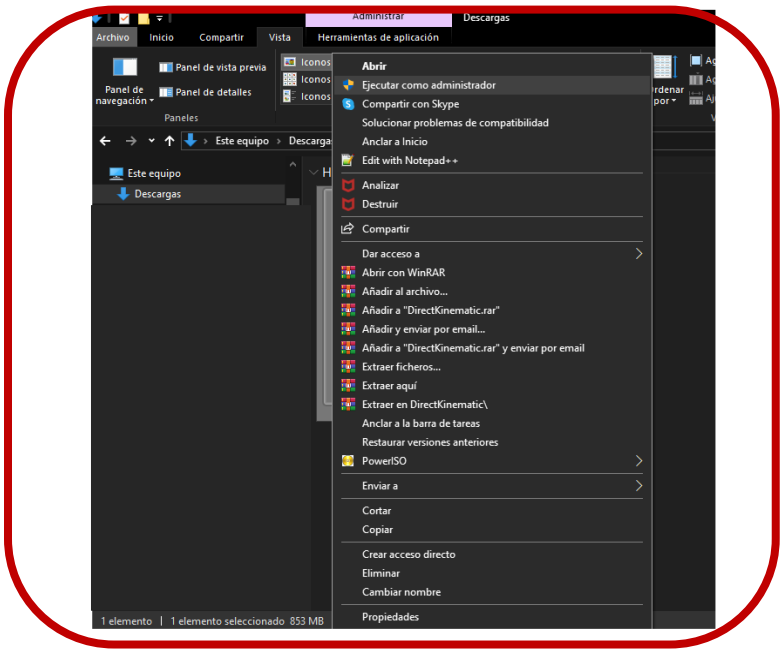
4. Si aparece una alerta diciendo “No se ha podido comprobar si este archivo tiene virus”, dar clic en “DESCARGAR DE TODOS MODOS” y utilizar



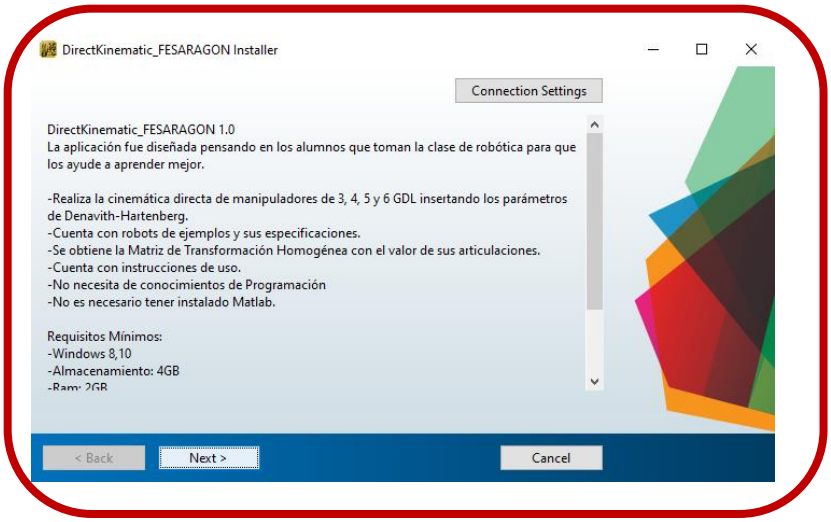
5. Esperar a que se termine la descarga, el archivo pesa alrededor de 854 MB dependiendo de la velocidad del internet de cada quien puede tarde entre 10 min a 60 min.



6. Cuando se termine de descargar nos vamos a la carpeta donde se almacenan las descargas (Aquí depende la configuración de cada usuario, en mi caso me dirijo a la carpeta de descargas). En el archivo se le da click derecho y ejecutar como administrador



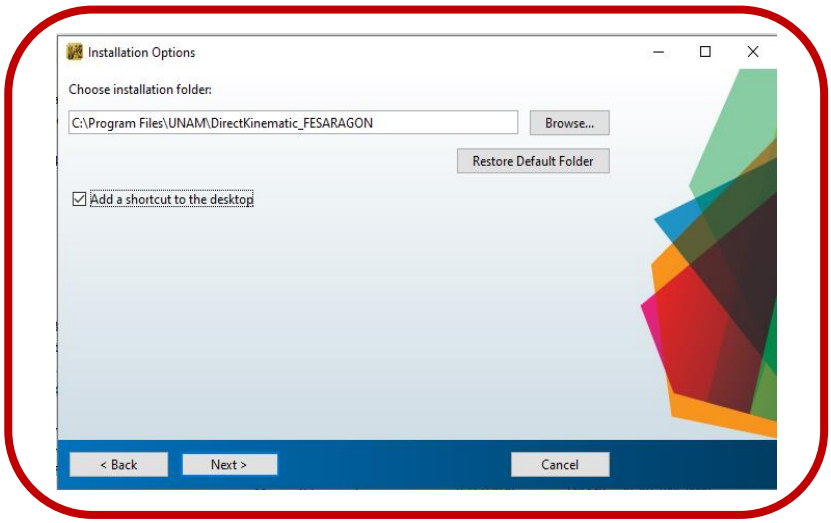
7. Dependiendo la velocidad de tu computadora o del procesador y la memoria RAM que cuente tu ordenador puede en demorar en abrir entre 2 min a 7 min. Cuando se abra el instalador, se abrirá una ventana como la que se muestra a continuación. Y se le da en el botón siguiente



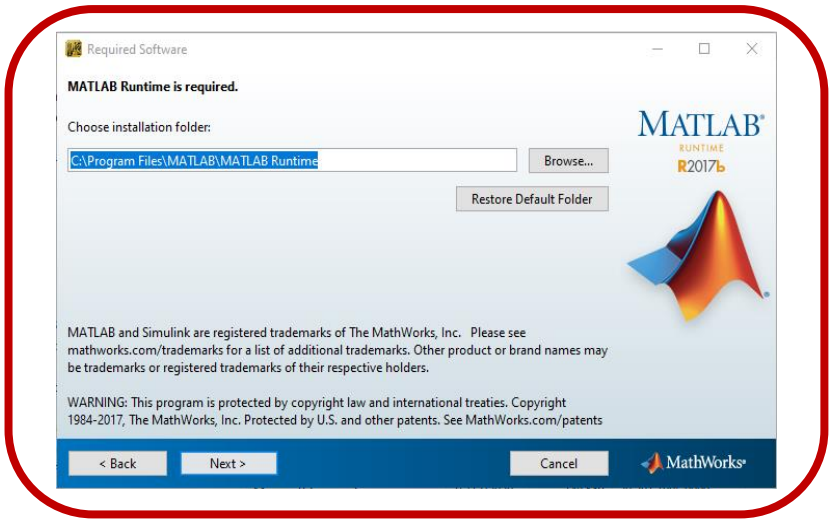
8. Antes de dar en el botón Next se debe cambiar la dirección del folder. La dirección que trae por default es:
   C:\Program Files\Facultad de Estudios Superiores Aragón\DirectKinematic_FESARAGON

   Y se debe cambiar por
   C:\Program Files\UNAM\ DirectKinematic_FESARAGON
   Se palomea la opción “add a shorcut to the desktop”

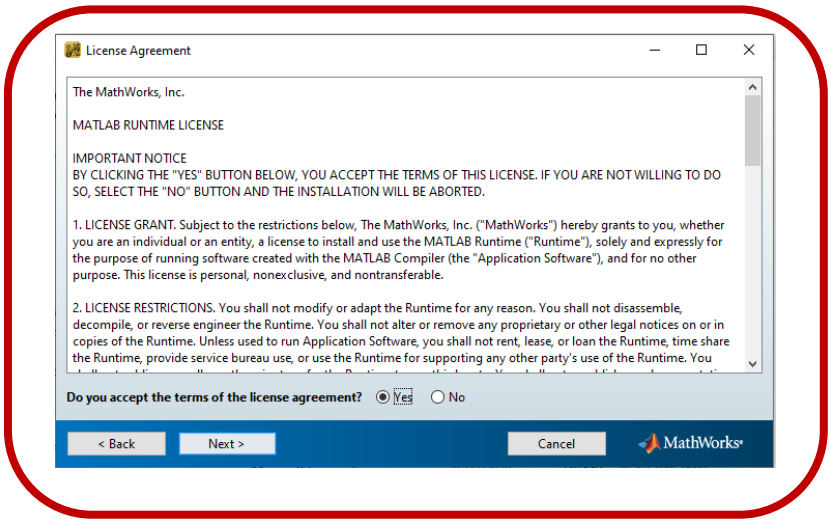
Y se da en el botón de Next >



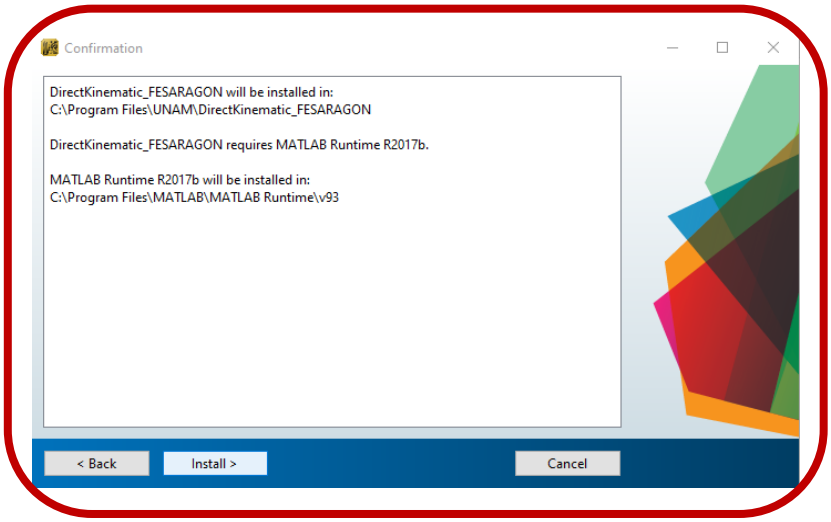
9. Aparece otra pantalla que nos dice que se debe instalar en Runtime de Matlab, dejamos la dirección como viene por default y se le da en el botón Next >



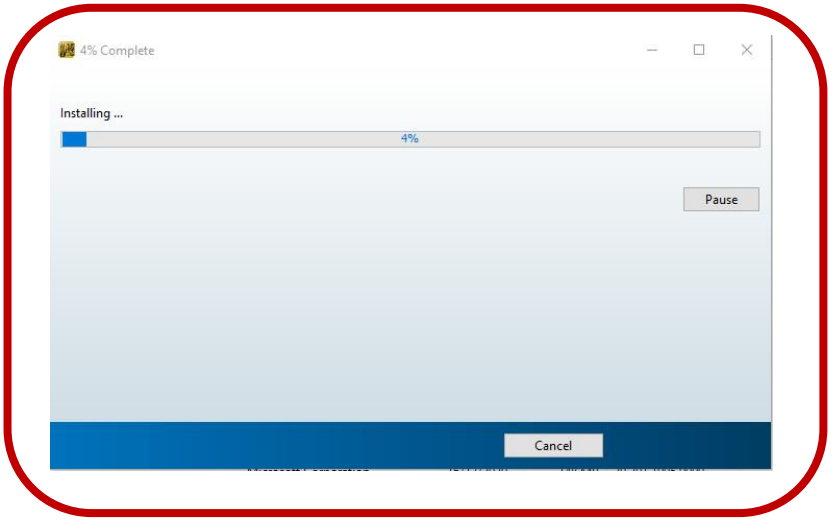
10. Aceptamos términos y condiciones y le damos en el botón siguiente.



11. Aparece un resumen de todo lo que se va a instalar y se da en el botón Install >



12. Comenzará a instalar, esperamos hasta que termine toda la instalación.

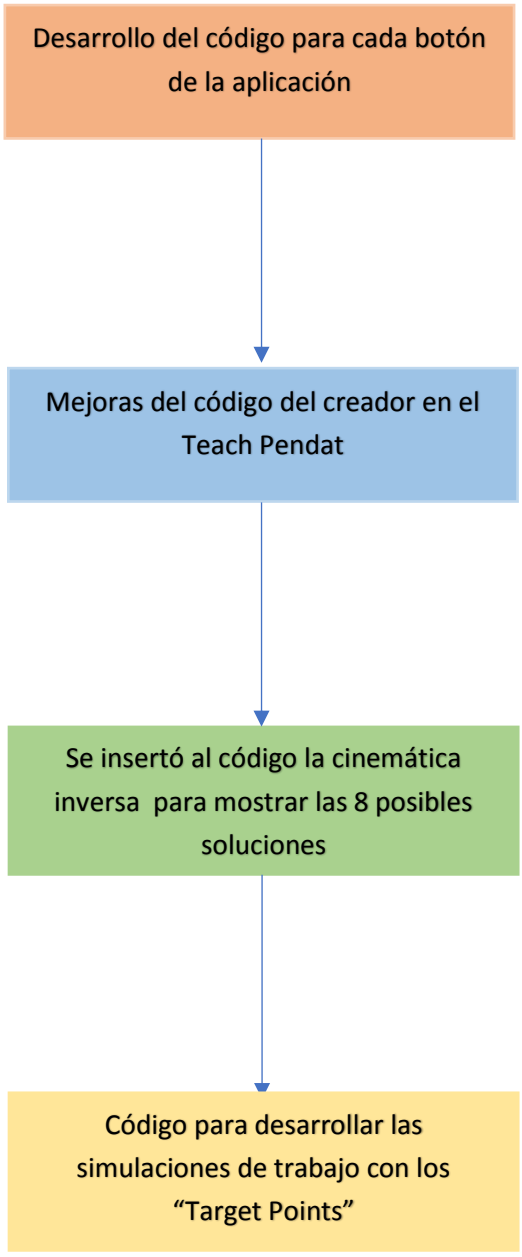


Apéndice 9 Código de la aplicación dos

Se presenta un diagrama de flujo de cómo se desarrolló la aplicación ya que el código es muy extenso para compartirlo en este apéndice.

Para solicitar el código, favor de enviar un correo al creador, con el motivo por el cual lo solicita:

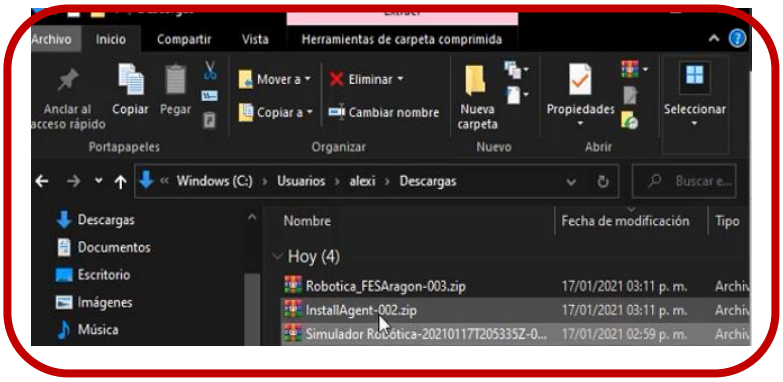
alexis.omar.ct96@gmail.com



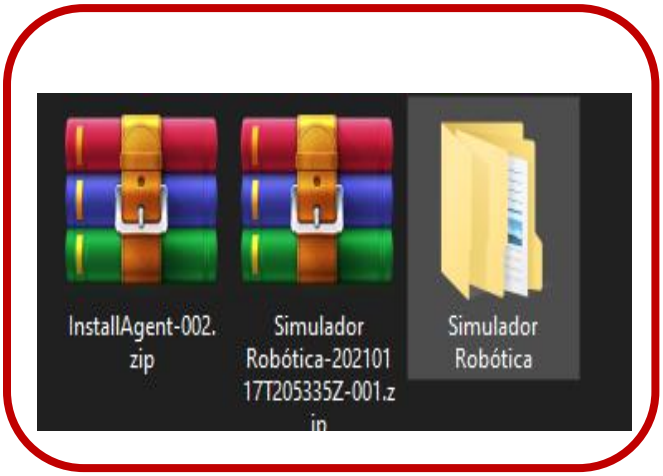
Apéndice 10 Pasos para instalar la aplicación número 2

1. Ingrese a su ordenador de preferencia, en este caso se ocupó Google Chrome.
2. Dirigirse a la dirección: https://drive.google.com/drive/folders/1Qb55kJgKhb8YzfF8CopZv4vAK_kfpRMt?usp=sharing
3. Dar clic en DESCARGAR TODO

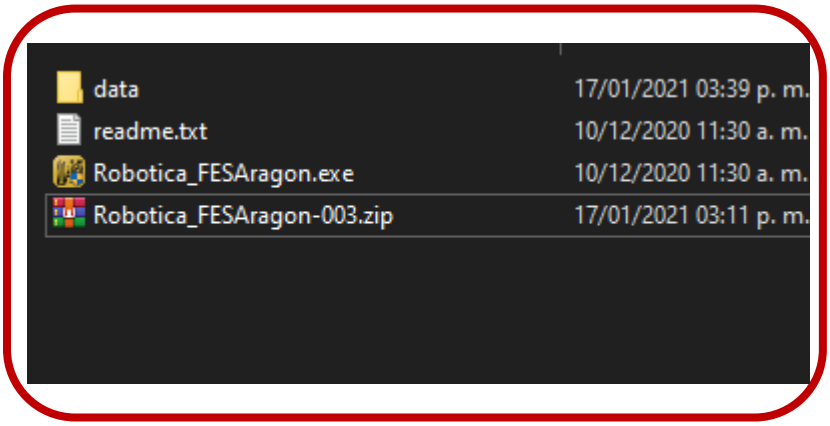
4. Esperar a que termine la descarga de todos los archivos



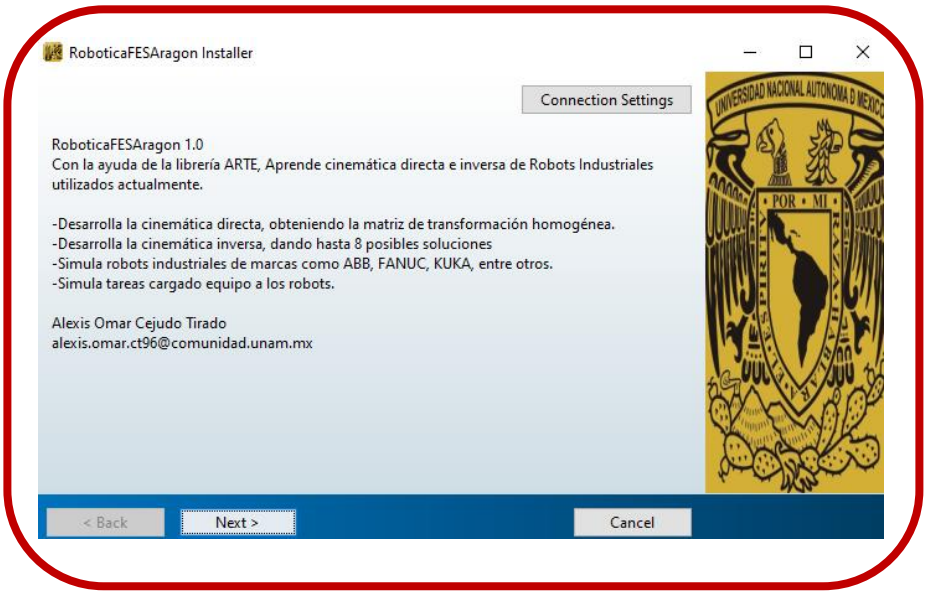
5. Extraemos el archivo Simulador Robotica-2010117120… y Se crea una carpeta con el nombre Simulador Robótica



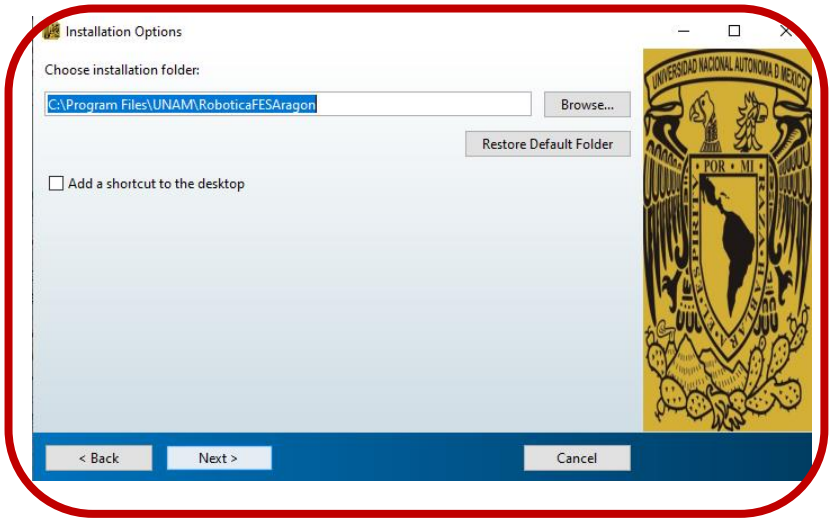
6. Se abre la carpeta Simulador Robótica y en la Carpeta Archivos se pega el archivo “InstallAgent”



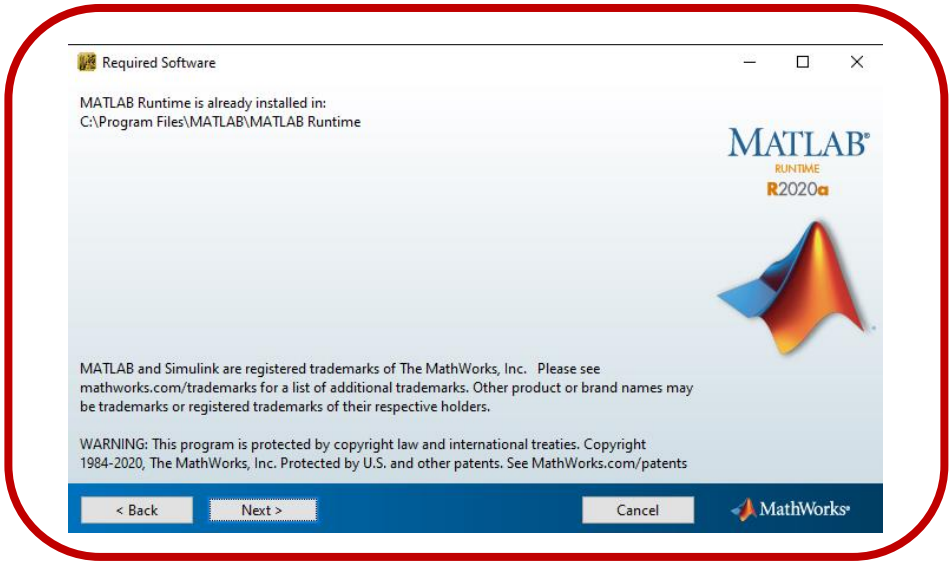
7. Se da clic derecho sobre el archivo Robotica_FESAragon.exe y de da clic en el botón ejecutar como administrador y aparece el instalador. Se da clic en el botón Next >



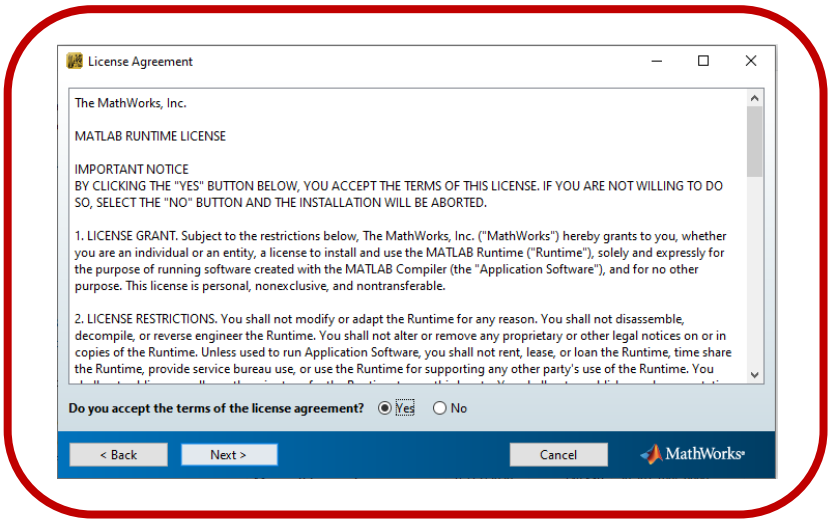
8. Se selecciona la carpeta donde se va a instalar el archivo, y da clic en next



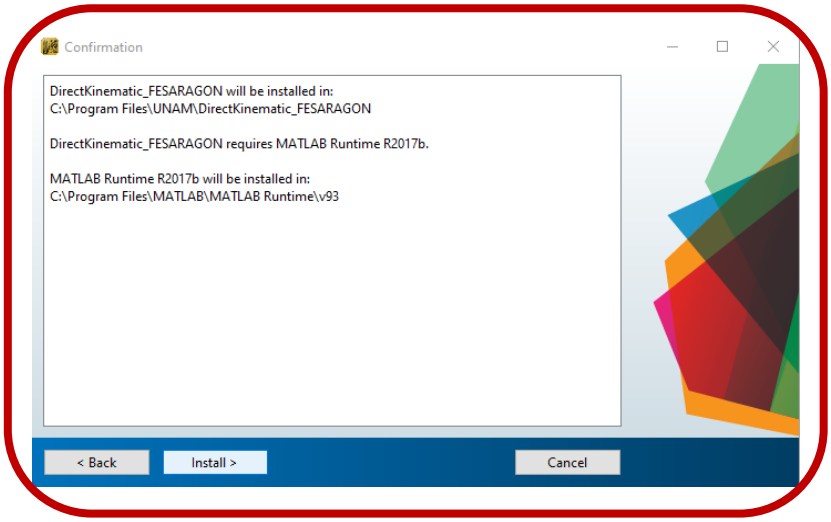
9. Se instala el runtime 2020 para que esta aplicación funcione. Se da clic en Next >



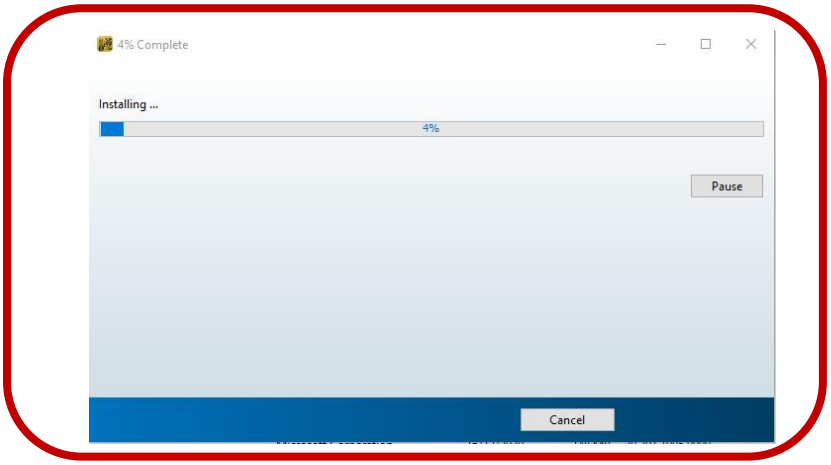
10. Se aceptan los términos y condiciones y da clic en Next >



11. Aparece un resumen de todo lo que se va a instalar y se da en el botón Install>



12. Comenzará a instalar, esperamos hasta que termine toda la instalación. Cuando termine se da clic en Finish



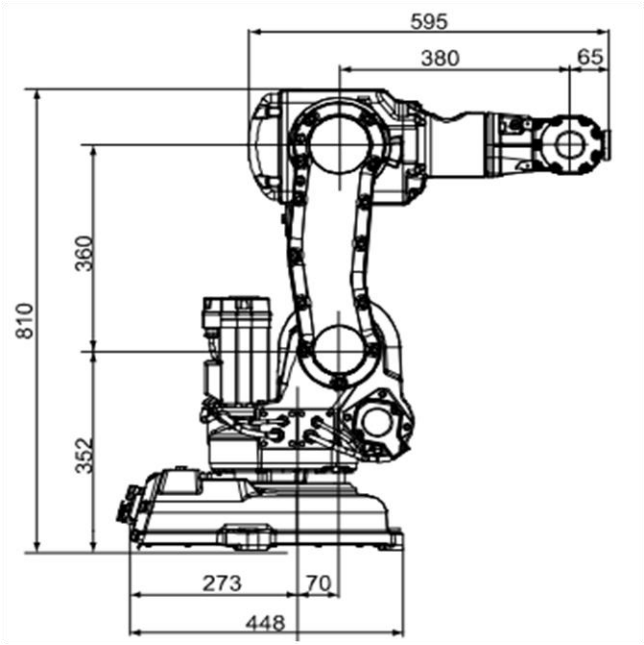
Apéndice 11 Ejercicio Propedéutico

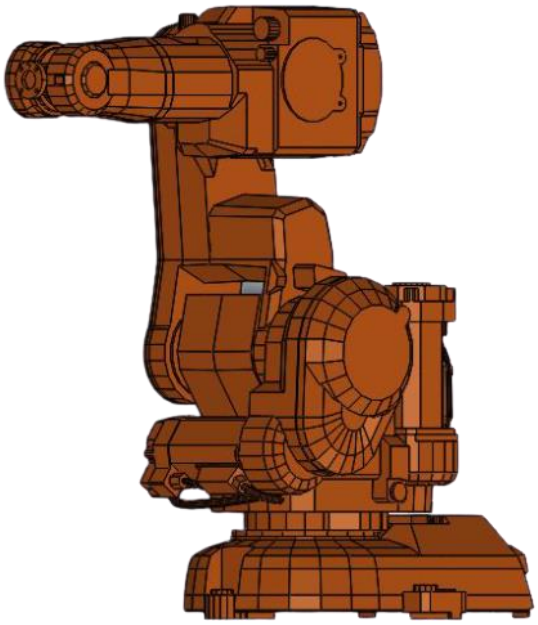
Nombre:

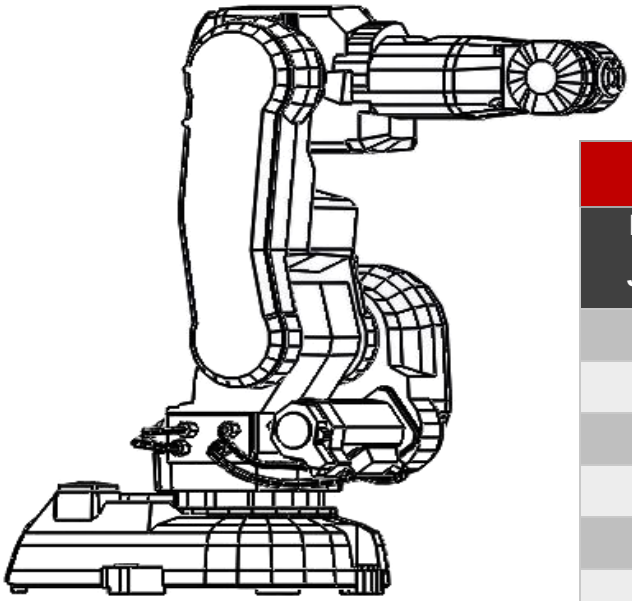
Grupo:

Número de Cuenta:

1. ¿Qué es la cinemática Directa?
2. ¿Qué es la cinemática Inversa?
3. Obtener la cinemática directa del robot por parámetros de Denavit-Hartenber. Y la matriz de transformación homogénea que define el robot. Valor de articulaciones.
   Q1=0, Q2=0, Q3=0, Q4=0, Q5=0, Q6=0
4. Obtener la cinemática inversa del robot.







JUNTAS	
NO.DE JUNTA	**MOVIMIENTO**
1	Rotación
2	Oscilación
3	Oscilación
4	Rotación
5	Oscilación
6	Rotación

Apéndice 12 Cuestionario con escala Likert

Nombre:

Grupo:

Número de Cuenta:

Marca de la computadora con la que cuento:

Tipo: Escritorio/ Laptop

Procesador:

Memoria Ram:

1. Aprendí de una mejor manera los temas de cinemática directa e inversa con ayuda de estas dos aplicaciones.

- Muy de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Muy en desacuerdo

2. Pude obtener más rápido los parámetros de Denavit-Hartenberg a como se obtenía en clase.

- Muy de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Muy en desacuerdo

3. Reforzaron mis conocimientos en conceptos básicos en los temas de cinemática directa e inversa.

- Muy de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Muy en desacuerdo

4. Cubrieron mis dudas sobre conceptos de los temas de cinemática directa e inversa.

- Muy de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Muy en desacuerdo

5. Pude observar los sistemas y los movimientos del robot mejor a como los vi en clase
- Muy de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Muy en desacuerdo

6. Es una excelente herramienta para utilizarla en futuras generaciones que cursen la materia.
- Muy de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Muy en desacuerdo

7. Esta herramienta también se podría utilizar en clases presenciales.
- Muy de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Muy en desacuerdo

8. Las aplicaciones son amigables y de fácil uso
- Muy de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Muy en desacuerdo

9. Pude instalar el software sin ningún problema.
- Muy de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Muy en desacuerdo

10. El programa no se tardó al iniciarlo.
- Muy de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Muy en desacuerdo

11. En clase no usamos alguna aplicación similar
- Muy de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Muy en desacuerdo

12. Pude obtener la cinemática inversa de una manera más rápida
- Muy de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Muy en desacuerdo

13. Este material me despertó interés a la materia de robótica
- Muy de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Muy en desacuerdo

14. Me gustaría aprender los temas de dinámica y programación con una herramienta como esta
- Muy de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Muy en desacuerdo

15. El uso de la tecnología es importante para aprender en esta clase de materias
- Muy de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Muy en desacuerdo

16. Utilizar simuladores me ayudo a acercarme más al mundo industrial.
- Muy de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Muy en desacuerdo

Realiza un comentario apoyándote en esta pregunta:

¿Qué te parecieron las aplicaciones, pudiste observar mejor los conceptos de la cinemática directa, que te gustaría mejorar de estás aplicaciones, crees que con esta herramienta sería más sencillo aprender, crees que te hayan acercado más al mundo industrial?

REFERENCIAS

[1] ABB. (2009). *FlexTrainer*. https://doi.org/ROB0162 EN_A

[2] ABB. (2019a). IRB 2400, M2004, Product specification. In *Data Sheet*. https://search.abb.com/library/Download.aspx?DocumentID=3HAC042195-005&LanguageCode=es&DocumentPartId=&Action=Launch

[3] ABB. (2019b). *Product specification IRB 140*. https://new.abb.com/products/robotics/es/robots-industriales/irb-140

[4] ABB. (2020). *RobotStudio*. https://new.abb.com/products/robotics/es/robotstudio

[5] Aldakin. (2017). *Industria 4.0. Qué es, ventajas e inconvenientes*. http://www.aldakin.com/industria-4-0-que-es-ventajas-e-inconvenientes/

[6] Allen-Bradley. (2018). *Kinetix VP-Series Servo Motors*. https://doi.org/VPC-B1652A-QJ12AS

[7] Antoni Garrell, L. G. (2019). *La industria 4.0 en la sociedad digital* (M. BOOKS (ed.); 1ra Edició).

[8] Army Technologu. (s.f.). *CUTLASS Next Generation Unmanned Ground Vehicle*. Obtenido de https://www.army-technology.com/projects/cutlass-next-generation-unmanned-ground-vehicle/

[9] Asís Roig, R. de. (2015). *UNA MIRADA A LA ROBÓTICA DESDE LOS DERECHOS HUMANOS* (Dykinson (ed.)).

[10] Barrientos, A., Peñin, L. F., Balaguer, C., & Aracil, R. (2007). *Fundamentos de Robótica* (2da.). Concepción Fernández Madrid.

[11] Basco, I., Beliz, G., Coatz, D., & Garnero, P. (2018). *Industria 4.0 fabricando el futuro*.

[12] Bunker DB. (7 de Diciembre de 2018). *Henn Na Hotel: el primer hotel del mundo atendido por robots*. Obtenido de https://bunkerdb.com/blog/entretenimiento/henn-na-hotel-atencion-robots/

[13] Cabero Almenara, J., & Costas, J. (2016). La utilización de simuladores para la formación de los alumnos. In *LA PUBLICIDAD EN IBEROAMÉRICANA: Vol. vol.17*.

[14] Carbajal Rojas, J. H. (2017). *La cuarta revolución industrial o industria 4.0 y su impacto en la educación superior en ingeniería en Latinoamerica y el Caribe*.

[15] Castro, J. (2019). *¿Qué es la Industria 4.0? La transformación tecnológica de nuestro siglo*. Blog Corponet. https://blog.corponet.com.mx/que-es-la-industria-4-0-la-transformacion-tecnologica-de-nuestro-siglo

[16] Contreras G., G. Á., & Carreño M., P. (2012). Simuladores en el Ámbito educativo: Un

Recurso didáctico para la enseñanza. *INGENIUM*, *Vol.25*, 107–119.

[17] Contreras G., G. Á., & Ramírez Montoya, M. S. (2010). *Uso de simuladores como recurso digital para la transferencia de conocimiento*. http://www.udgvirtual.udg.mx/apertura/index.php/apertura/article/view/22/32#resultados

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

[19] Corke, P. (2020). *Acerca de | Peter Corke*. https://petercorke.com/about/

[20] Covarrubias Díaz, A. (2014). *Competencias del facilitador del aprendizaje en línea: revisión del estado del arte*.

[21] Craig, J. J. (2006). *Robótica* (Pearson Educación de México S.A de C.V (ed.); 3ra.).

[22] Departamento de Tecnología Antonio Sequeros. (2016). *ELEMENTOS TRANSMISORES DE MOVIMIENTO*.

[23] Editores. (2019). *Robótica | Estos son los robots de servicio*. https://www.editores-srl.com.ar/revistas/aa/11/ifr_robots_de_servicio

[24] EL UNIVERSAL. (2016). *Robots marinos, el futuro de los oceanos*. https://www.eluniversal.com.mx/articulo/ciencia-y-salud/tecnologia/2016/10/19/los-robots-marinos-y-la-captura-del-adn-son-el-futuro

[25] Facultad de Ingeniería. (2016a). *Facultad de Ingeniería / Ingeniería Mecatrónica Mapa Curricular 2016*. https://www.ingenieria.unam.mx/programas_academicos/licenciatura/mecatronica_plan2016.php

[26] Facultad de Ingeniería. (2016b). *Plan de estudios Ingeniería Mecatrónica*. https://www.ingenieria.unam.mx/programas_academicos/licenciatura/Mecatronica/mecatronica_2016.pdf

[27] FES Aragón. (n.d.-a). *Mapa curricular Ingeniería Mecánica FES Aragón*. Retrieved January 28, 2021, from https://www.aragon.unam.mx/fes-aragon/public_html/documents/licenciaturas/mecanica/mapa-curricular.pdf

[28] FES Aragón. (n.d.-b). *Plan de Estudios Ingeniería Mecánica*. http://ingenieria.aragon.unam.mx/imc/plan_estudios.html

[29] FESTO. (2018). *Cilindros y Actuadores Neumáticos*. https://doi.org/ADN Cilindro Compacto de Doble Efecto

[30] Gil Aparicio, A. (2013). *umh1770 2013-14 Lec014 4-Cinemática inversa parte 4 - YouTube*. https://www.youtube.com/watch?v=d_gnGr86X7g

[31] Gilat, A. (2016). *Matlab Una introducción con ejemplos básicos* (2da Edició).

[32] González, B. (2018). *Industria 4.0: una revolución para las personas*. https://www.youtube.com/watch?v=a0Ycxn-bZak

[33] INNOVATING SOLUTIONS. (2018). *SIMULADORES*. https://www.gmv.com/es/Sectores/Aeronautica/Simulacion/

[34] Khamis, A. (2006). *Lenguaje RAPID*. http://personal.biada.org/~jhorrillo/INTRODUCCIO RAPID.pdf

[35] KOMPASS. (2020). *CILINDROS HIDRÁULICOS*. https://doi.org/CA- II Type

[36] Lili, W., & Eric, R. (2019). *Nuevos simuladores para entrenamiento de médicos residentes*. Gaceta Facultad de Medicina, UNAM. http://gaceta.facmed.unam.mx/index.php/2019/05/22/nuevos-simuladores-para-entrenamiento-de-medicos-residentes/

[37] Martínez Rueda, J. (2007). *SISTEMAS ELÉCTRONICOS Y ELÉCTRICOS de las AERONAVES* (Paraninfo S.A (ed.); Primera Ed).

[38] MathWorks. (2019). *Descripción del producto MATLAB - MATLAB & Simulink - MathWorks España*. https://es.mathworks.com/help/matlab/learn_matlab/product-description.html

[39] MathWorks. (2020a). *GUI de MATLAB - MATLAB & Simulink*. https://es.mathworks.com/discovery/matlab-gui.html

[40] MathWorks. (2020b). *MATLAB - El lenguaje del cálculo técnico - MATLAB & Simulink*. https://es.mathworks.com/products/matlab.html

[41] MathWorks. (2020c). *MATLAB Compiler - MATLAB*. https://es.mathworks.com/products/compiler.html

[42] MathWorks. (2020d). *Modelización y simulación - MATLAB & Simulink*. https://la.mathworks.com/discovery/modeling-and-simulation.html

[43] MathWorks. (2020e). *New License for MATLAB Student R2020a - MathWorks España*. https://es.mathworks.com/store/link/products/student/SV?s_tid=ac_buy_sv_but1

[44] MathWorks. (2020f). *Robótica y sistemas autónomos - MATLAB & Simulink*. https://es.mathworks.com/solutions/robotics.html

[45] MathWorks. (2020g). *Robotics System Toolbox - MATLAB & Simulink*. https://es.mathworks.com/products/robotics.html

[46] Ortiz Germán. (2020). *Robots en el sector productivo | Deloitte México*. https://www2.deloitte.com/mx/es/pages/dnoticias/articles/robots-necesidad-laboral.html

[47] Osorio Villa, P., Blanca, Ángle, F. M., & Alejandro, F. J. (2012). El uso de simuladores educativos para el desarrollo de competencias en la formación universitaria de pregrado.

Revista Q ISSN 1909-2814, *Vol. 7*, 24. http://eds.a.ebscohost.com.pbidi.unam.mx:8080/eds/pdfviewer/pdfviewer?vid=2&sid=39af6a0d-5262-4ccf-a7f6-25276c7f8463%40sessionmgr4007

[48] Payá, C. L., Arturo, A. G., Peidró, V. A., Miguel, J. G. L., & Gárcia, Ó. R. (2018). *Uso MATLAB en robótica y visión por computador* (Universidad Miguel Hernández de Elche (ed.); 1ra ed.).

[49] Portella, A. (2018). *Industria 4.0, una revolución que se retrasa en México*. Forbes México. https://www.forbes.com.mx/industria-4-0-una-revolucion-que-se-retrasa-en-mexico/#:~:text=La Cuarta Revolución Industrial o,no tiene prisa en México.

[50] REAL ACADEMIA ESPAÑOLA. (n.d.). *Diccionario de la lengua española* (23.3 en lí). Retrieved July 25, 2020, from https://dle.rae.es/simulador?m=form

[51] Reyes Cortés, F. (2011). *Robótica Control de Robots Manipuladores* (1ra ed.). Alfaomega.

[52] Riquelme, R. (2019). *México llega con retraso a la Cuarta Revolución Industrial*. El Economista. https://www.eleconomista.com.mx/tecnologia/Mexico-llega-con-retraso-a-la-Cuarta-Revolucion-Industrial-20191009-0055.html

[53] RoboDK. (n.d.). *Pogramación de Robots*. https://robodk.com/doc/es/Robot-Programs.html

[54] Saha kumar, S. (2010). *Introducción a la Robótica* (1ra ed.).

[55] Smart Industry MX. (2018). SIEMENS IMPULSA LA DIGITALIZACION EN MÉXICO, CENTRO AMÉRICA Y EL CARIBE. *Cluster Industrial Enlazando Negocios*, *37*, 22–23.

[56] Solís, A. (2018). "Los robots van a colaborar con los humanos, no a reemplazarlos." *Forbes México*. https://www.forbes.com.mx/los-robots-no-acabaran-con-los-empleos-segun-abb/

[57] Somolinos Sánchez, J. A. (2002). Sensores en Robótica. In Ediciones de la Universidad de Castilla- La Manchaca (Ed.), *AVANCES EN ROBÓTICA Y VISIÓN POR COMPUTADOR* (Primera Ed, p. 64).

[58] Tech Simulation. (2020). *SIMULADORES DE TRANSPORTE*. CloudLabs. http://www.etechsimulation.com/index.php/industrias/simuladores-de-transporte

[59] THOROUGHTEC SIMULATION. (2020). *SIMULADORES MILITARES CYBERWAR*. http://www.thoroughtec.com/es/cyberwar-simuladores-militares/

[60] UNIVERSAL ROBOTS. (n.d.). ▷ *Principales Sectores que Utilizan【Robots Colaborativos】| UR*. Retrieved July 27, 2020, from https://www.universal-robots.com/mx/industrias/

[61] Universidad de Alcalá. (2019). *VENTAJAS Y DESVENTAJAS DE LA INDUSTRIA 4.0*. https://www.masterindustria40.com/ventajas-desventajas-industria-4-0/

[62] Universidad Don Bosco. (2016). *Morfología de Robots* (3ra ed.).

[63] Universidad Miguel Hernández. (2012). *ARTE (A Robotic Toolbox for Education)*. https://arvc.umh.es/arte/index_en.html

[64] Universidad Nacional Autónoma de México. (2020). *LA UNAM EN NUMEROS*. Portal de Estadística Universitaria. http://www.estadistica.unam.mx/numeralia/

[65] Villafranco, G. (2017). *Éstos son los empleos que se perderán por la cuarta revolución industrial*. Forbes México. https://www.forbes.com.mx/estos-los-empleos-se-perderan-la-cuarta-revolucion-industrial/

[66] Zegarra, C., & Moisés, P. (2018). *Industria 4.0: oportunidades y retos en México*. Forbes México. https://www.forbes.com.mx/industria-4-0-oportunidades-y-retos-en-mexico/