Diseño e implementación de un sistema de operación off-line del brazo robótico Mitsubishi RV-2SDB a través de realidad virtual no inmersiva para contribuir en el proceso de aprendizaje de robótica industrial en el laboratorio de Mecatrónica de la Universidad De Las Fuerzas Armadas ESPE sede Latacunga.

Bastidas Guayanay, Daniela Alexandra y Gallegos Velásquez, Dayana Esther

Departamento de Ciencias de la Energía y Mecánica

Carrera de Ingeniería en Mecatrónica

Trabajo de titulación, previo a la obtención del título de Ingeniero en Mecatrónica

Msc. Mendoza Chipantasi, Dario José

23 de agosto 2021

DEPARTAMENTO DE ENERGÍA Y MECÁNICA

CARRERA DE INGENIERÍA MECATRÓNICA

CERTIFICACIÓN

Certifico que el trabajo de titulación, **"Diseño e implementación de un sistema de operación off-line del brazo robótico Mitsubishi RV-2SDB a través de realidad virtual no inmersiva para contribuir en el proceso de aprendizaje de robótica industrial en el laboratorio de Mecatrónica de la Universidad De Las Fuerzas Armadas ESPE sede Latacunga"** fue realizado por las señoritas **Bastidas Guayanay, Daniela Alexandra y Gallegos Velásquez, Dayana Esther** el cual ha sido revisado y analizado en su totalidad por la herramienta de verificación de similitud de contenido; por lo tanto cumple con los requisitos legales, teóricos, científicos, técnicos y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, razón por la cual me permito acreditar y autorizar para que lo sustente públicamente.

Latacunga, 23 agosto del 2021

Firmado electrónicamente por:
DARIO JOSE MENDOZA CHIPANTASI

.....

Msc. Mendoza Chipantasi, Dario José

C. C.: 0603110834

Curiginal

Document Information

Analyzed document	[Tesis] Bastidas Guayanay Daniela Alexandra y Gallegos Velásquez Dayana Esther.pdf (D111681142)
Submitted	8/25/2021 8:20:00 PM
Submitted by	
Submitter email	rtoasa@uisrael.edu.ec
Similarity	4%
Analysis address	dguevara.uta@analysis.urkund.com

Sources included in the report

SA	**Trabajo de titulacion_Barberan.pdf** Document Trabajo de titulacion_Barberan.pdf (D110803899)	1
SA	**tesis_final.pdf** Document tesis_final.pdf (D82820091)	2
W	URL: http://ri.uaemex.mx/bitstream/handle/20.500.11799/108137/secme33422_1.pdf?sequence=1 Fetched: 8/25/2021 8:21:00 PM	2
W	URL: https://www.slideshare.net/jesusgomez39750/guia-r-pidarobotmitsubishirv2aj Fetched: 8/25/2021 8:21:00 PM	1
W	URL: https://docplayer.es/13616403-Control-de-una-fresadora-cnc-de-uso-didactico-instituto-politecnico-nacional-que-para-obtener-el-titulo-de-ingeniero-tesis.html Fetched: 2/15/2020 7:09:02 AM	1
SA	**Carrillo-Huaraca-NAveda-Yépez.pdf** Document Carrillo-Huaraca-NAveda-Yépez.pdf (D47755995)	1
W	URL: https://pdfcoffee.com/fundamentos-de-robotica-2-ed-antonio-barrientos--2-pdf-free.html Fetched: 1/23/2021 12:03:21 PM	4
W	URL: https://docplayer.es/113233561-Control-cinematico-de-robots-serie-con-arduino.html Fetched: 12/7/2019 6:41:15 PM	1

Firmado electrónicamente por:
DARIO JOSE MENDOZA CHIPANTASI

.....

Msc. Mendoza Chipantasi, Dario José

C. C.: 0603110834

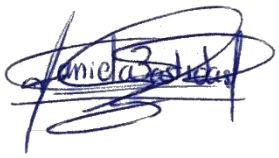
DEPARTAMENTO DE ENERGÍA Y MECÁNICA

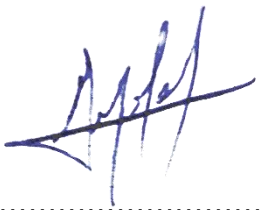
CARRERA DE INGENIERÍA MECATRÓNICA

RESPONSABILIDAD DE AUTORÍA

Nosotras, **Bastidas Guayanay, Daniela Alexandra**, con cédula de ciudadanía N°1725389785 y **Gallegos Velásquez, Dayana Esther**, con cédula de ciudadanía N°2300048929, declaramos que el contenido, ideas y criterios del trabajo de titulación**: "Diseño e implementación de un sistema de operación off-line del brazo robótico Mitsubishi RV-2SDB a través de realidad virtual no inmersiva para contribuir en el proceso de aprendizaje de robótica industrial en el laboratorio de Mecatrónica de la Universidad De Las Fuerzas Armadas ESPE sede Latacunga"** es de nuestra autoría y responsabilidad, cumpliendo con los requisitos legales, teóricos, científicos, técnicos, y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Latacunga, 23 agosto del 2021





…… ……

Bastidas Guayanay, Daniela Alexandra **Gallegos Velásquez, Dayana Esther**

C.C.: 1725389785 C.C.: 2300048929

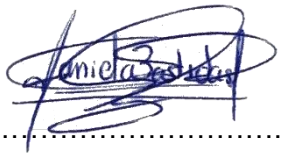
DEPARTAMENTO DE ENERGÍA Y MECÁNICA

CARRERA DE INGENIERÍA MECATRÓNICA

AUTORIZACIÓN DE PUBLICACIÓN

Nosotras, **Bastidas Guayanay, Daniela Alexandra**, con cédula de ciudadanía N°1725389785 y **Gallegos Velásquez, Dayana Esther**, con cédula de ciudadanía N°2300048929, autorizamos a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de titulación: **"Diseño e implementación de un sistema de operación off-line del brazo robótico Mitsubishi RV-2SDB a través de realidad virtual no inmersiva para contribuir en el proceso de aprendizaje de robótica industrial en el laboratorio de Mecatrónica de la Universidad De Las Fuerzas Armadas ESPE sede Latacunga"**, en el Repositorio Institucional, cuyo contenido, ideas y criterios son de nuestra responsabilidad.

Latacunga, 23 agosto del 2021



.....

Bastidas Guayanay, Daniela Alexandra

C.C.: 1725389785

.....

Gallegos Velásquez, Dayana Esther

C.C.: 2300048929

Dedicatoria

Dedico este proyecto de titulación a mi familia por darme fortaleza, apoyo, valentía y coraje de seguir adelante. Mis pilares de vida, mi madre Luz que cada día me demuestra que el cariño y el trabajo van de la mano y la gratitud que generó en mí forjaron una parte esencial de vida, a mi padre Manuel que nunca limito mis habilidades académicas y personales que desarrollaba desde que era una niña con eso me brindo la confianza por trabajar junto a él en cualquier proyecto. A mi hermana Giselle que me enseñó a apreciar que cada caída es una enseñanza y lo importante es saber levantarse. Mi hermano Jordan es para mí un ejemplo de estudio, a mi hermana Fernanda que me demuestra la fuerza de voluntad que tiene y cada día al enseñarme algo nuevo. A mis sobrinos Poleth, David y Dylan que, con su existencia, su cariño y su curiosidad de conocer algo nuevo, han hecho resaltar la lucha y perseverancia para que vean en mi un ejemplo de ser una gran persona y una gran profesional.

Este trabajo es por y para ustedes, apoyando a toda la familia en lo que más pueda.

Bastidas Guayanay, Daniela Alexandra

Dedicatoria

El presente trabajo de investigación es dedicado en primer lugar a Dios, quien me dio fortalezas de seguir cuando veía que no podía más, a toda mi familia, quienes estuvieron apoyándome a pesar de muchas adversidades que hemos pasado, en especial a mi padre que sin su apoyo no habría podido llegar hasta donde me encuentro el día de hoy. También dedico esté trabajo a los grandes docentes que me apoyaron en esta etapa y que pudieron impartirme sus conocimientos, a mis amigos que fueron pilar fundamental en cada momento compartido y finalmente a las personas que llegue a conocer en la etapa final de mi tesis y se convirtieron también en una gran familia.

Gallegos Velásquez, Dayana Esther

Agradecimiento

Estoy agradecida con las personas más importantes en mi vida mis padres y hermanos que fueron un apoyo total en mi formación académica y personal. A mi familia más allegada Fernando padre de sobrinos, tíos, tías, primas y primos por darme un aliento en cada paso desde la escuela, colegio, universidad y la vida. Mi segunda madre Francisca, que me vio crecer y darme todo el amor de mamá. A la señorita Veronica por dar a mi familia y mis hermanos el sustento, para formarnos como profesionales en lo que pudo. Mis amigas y amigos, con los que comparto momentos valiosos y fueron un soporte personal y académico para salir adelante.

A la Universidad de las Fuerzas Armadas ESPE que me brindo el conocimiento fundamental en la formación como Ingeniería Mecatrónica. A los docentes que realmente cumplieron con su rol en mi aprendizaje.

A mi tutor de tesis Dario Mendoza, que nos acompañó en todo el proceso de titulación otorgando conocimiento, opiniones y pautas para sacar este trabajo a flote.

Finalmente, a la persona que a lo largo de mi vida universitaria ha estado conmigo Dayana, mi gran amiga y compañera de tesis que en cada fracaso o éxito a estado ahí.

Bastidas Guayanay, Daniela Alexandra

Agradecimiento

El amor recibido, la paciencia, constancia y dedicación con la que se preocupaban día a día mis padres en el desarrollo de esta tesis, es un sentimiento único y puro de amor, dicho sentimiento se ve reflejado en las acciones como hija que puedo tener, al sentirme realizada por haber alcanzar un sueño más de los tantos planeados para mi vida.

Gracias a mis padres por ser los principales promotores de mis sueños, gracias a ellos por cada día confiar y creer en mí y en mis expectativas, por darme seguridad y confianza en todo lo que realizo, gracias a mis hermanos por darme consejos en este camino llamado vida, me sirvieron mucho para poder crecer como persona, a mi madre por escucharme en todo momento y mostrarme lo valioso que es la palabra de Dios.

Agradezco a la persona más importante en mi vida, mi Padre, quien me ah enseñado a no desistir, quien con su ejemplo me ah demostrado que a pesar de muchas dificultades que podamos enfrentar en la vida, siempre debemos continuar, gracias a él no me rendí en toda esta hermosa etapa universitaria, hoy dedico todo mi esfuerzo y perseverancia a su apoyo incondicional.

Agradezco a los docentes que impartieron sus conocimientos con mucho amor y calidez, gracias por los consejos para la vida profesional y personal, agradezco a mi gran amiga incondicional Daniela, sin su apoyo tampoco pudiera estar aquí, gracias por compartir toda la etapa universitaria junto a mí.

Finalmente agradezco a todas las personas que han estado en esta etapa y en el desarrollo de la Tesis, por sus consejos, su apoyo, y estar conmigo hasta el final.

Gallegos Velásquez, Dayana Esther

Tabla de contenidos

Índice de Tablas

Índice de figuras

Resumen

En el presente trabajo de titulación, contempla la implementación de un sistema de operación off-line del robot Mitsubishi Melfa RV 2SDB a través del motor de videojuegos Unity. Por medio del estudio preliminar se incorporó en el motor de juego un aula virtual, la estructura que sostiene el manipulador y el robot Mitsubishi Melfa RV 2SDB. Para la cinemática directa e inversa del robot se efectuó dos soluciones, primero resolver la configuración por medio de los parámetros de Denavit Hartenberg para hallar la matriz de transformación homogénea, segundo, la resolución por medio modelo diferencial de la jacobiana o jacobiana inversa que relaciona las variaciones entre velocidades articulares y del efector final misma que se efectuó en el controlador para realizar trayectorias. En el simulador colocó 4 entradas para el movimiento del robot una de ellas el Wii Motion Plus y para la programación que realiza el usuario en el interfaz es seleccionar y configurar que tipo de trayecto ya sea punto a punto, línea, arco o circulo esto es gracias a la estructura de herencias y polimorfismos que permite modularidad y reutilización de código todo esto se ejecutó por medio del scripting en C# en Unity. Finalmente se analiza en función a la precisión y eficiencia las trayectorias del simulador, por medio de la comunicación TCP/IP de la controladora con Matlab, y, a su vez se valida la hipótesis referente al proceso de aprendizaje con los estudiantes, midiendo el nivel de conocimiento adquirido después de su uso.

Palabras clave

- **SOFTWARE UNITY**
- **BRAZO ROBÓTICO MITSUBISHI MELFA RV 2SDB**
- **VIDEOJUEGOS**
- **ROBÓTICA INDUSTRIAL**

Abstract

In the present degree work, it contemplates the implementation of an off-line operating system of the robot Mitsubishi Melfa RV 2SDB through the engine of videogames Unity. By means of the preliminary study was incorporated in the game engine virtual classroom, the structure that holds the manipulator and the robot Mitsubishi Melfa RV 2SDB. For the direct and inverse kinematics of the robot two solutions were made, first solving the configuration by means of the Denavit Hartenberg parameters to find the homogeneous transformation matrix, second, the resolution by means of differential model of the Jacobian or Jacobian inverse that relates the variations between articular velocities and the final effector same that was carried out in the controller to realize trajectories. In the simulator, he placed 4 inputs for the robot movement, one of them the Wii Motion Plus, and for the programming that the user performs in the interface is to select and configure what type of path is already point-to-point, line, arc or circle this is thanks to the structure of inheritances and polymorphisms that allows modularity and reuse of code all this was executed by means of C# scripting in Unity. Finally, the trajectories of the simulator are analyzed in terms of accuracy and efficiency, by means of the TCP/IP communication of the controller with Matlab, and, in turn, the hypothesis regarding the learning process with the students is validated, measuring the level of knowledge acquired after use.

Keywords

- **SOFTWARE UNITY**
- **ROBOTIC ARM MITSUBISHI MELFA RV 2SDB**
- **VIDEO GAME**
- **INDUSTRIAL ROBOTICS**

Capítulo I

1. Aspectos generales

1.1 Planteamiento del problema

Los laboratorios convencionales, con toda su infraestructura y equipos, han sido tradicionalmente el lugar predilecto para desarrollar prácticas, pero a su vez el significado de costos mayores se hace presente. Sin embargo, a medida que los modelos educativos se han vuelto más flexibles y enfocados a la competencia de incluir nuevas tecnologías como laboratorios virtuales en los centros de educación permitiendo la accesibilidad y retención al aprendizaje práctico a través de simulaciones (De Antonio et al., 2000)

En el proceso de aprendizaje, las prácticas del laboratorio son un potente recurso pedagógico para desarrollar competencias en grados de ciencia experimental e ingeniería, siendo una herramienta necesaria y significativa. Un laboratorio tradicional tiene sesiones prácticas en un horario fijo programado periódicamente y con un tiempo limitado para la finalización de la actividad, en ocasiones el estudiante siente que no alcanzó todos los objetivos o que no comprendió con suficiencia todas las experiencias del laboratorio; llevando esto a la posterior búsqueda de otros espacios de tiempos disponibles para poder terminar las tareas asignadas, labor que regularmente no puede ser realizada debido a que el laboratorio se encuentra ocupado en otras clases, investigaciones, consultorías o el ingreso no puede ser de manera regular a causa de la modalidad de enseñanza en la que nos encontramos en la actualidad.

En mención a esto la materia de robótica industrial presenta en su syllabus prácticas en dónde el estudiante refuerce los conocimientos teóricos aprendidos, para ello se basan en la utilización de los diferentes manipuladores robóticos que existen en la universidad; dentro de las programaciones presentes se encuentra la modalidad

tradicional on-line y en ella es necesario que, durante el tiempo de desarrollo del programa se disponga del robot físicamente, y consiste en escribir un algoritmo usando una programación textual de robot para la generación de trayectorias, tomando en cuenta que se debe tener conocimiento previo de programación específica del software del robot, haciendo referencia al párrafo anterior es necesario optimizar tiempos de trabajo en las prácticas presenciales para que todos los estudiantes puedan hacer uso del manipulador y dominen las bases fundamentales que integran su funcionamiento, por ello es necesario incluir un modo de operaciones off – line, que corresponde a una simulación previa de la programación del robot en un entorno virtual y posterior a esto las trayectorias generadas puedan ser visualizadas en el brazo físicamente partiendo de los mismos conocimientos que se tiene en la programación on-line.

1.2 Antecedentes

Los avances tecnológicos en el campo de robótica son continuos y esto se debe al buen manejo en la investigación y recursos técnicos de ciertos sectores que se dedican a ello, Ecuador establece en el Art. 284, “Incentivar el conocimiento científico y tecnológico”, y esto logra que las universidades transfieran el conocimiento del estudiante hacia la comunidad.

Es una realidad que, para las nuevas generaciones de estudiantes universitarios, hablar de nuevas tecnologías es hablar del uso del ordenador como herramienta. Es por ello, que el uso de software específico evidenció un papel determinante como herramienta de trabajo y como base de su aprendizaje.

En la última década la realidad virtual ha tenido un crecimiento exponencial elevado, debido a la presentación de nuevas tecnologías accesibles para hacer uso de ella, su exploración ha sido abarcada a realizar entornos virtuales como videojuegos o simulaciones predictivas de procesos industriales, donde se evidencie un acercamiento

virtual de la parte física simulando situaciones finales que se podrían alcanzar sin presentar el riesgo de tener situaciones de peligro que podría afectar al usuario o generar pérdidas en costos en el caso de las industrias. (Fib, 2008)

El aumento de la competencia en la industria y el incremento de la demanda incurre en una intensificación de la productividad. Este hecho ha repercutido notablemente en el entorno de los robots industriales que han dirigido la mirada a la programación off-line, esto ocurre a una mejora en la productividad a través de simulaciones, donde puedan analizar y optimizar tiempos de producción.

La Universidad de las Fuerzas Armadas ESPE-L posee dos marcas de robot manipuladores un Mitsubishi y 3 Robots KUKA. Debido a las altas necesidades industriales percibidos en la actualidad, Mitsubishi Electric propone una gran variedad de modelos de robots. Todos sus robots son potentes, rápidos y compactos, características por las cuales se ha implementado en el laboratorio de Mecatrónica el robot RV-2SDB, posee un controlador CR1D-700 Series y el software recomendado para la programación y simulación es CIROS® que posee el lenguaje Melfa Basic V donde su estructura es un conjunto de instrucciones basado en el lenguaje Basic estándar con programación textual. (Gómez, 2013)

El robot Mitsubishi RV-2SDB puede combinarse sin ninguna complejidad con un gran número de componentes de automatización. Esto debido a que la unidad de control del robot puede comunicarse mediante con algunos puertos de comunicación. (Naranjo & Tello, 2017)

La estación del brazo Robótico Mitsubishi RV-2SDB permite crear un entorno de trabajo donde los estudiantes adquieran criterios industriales, mediante las prácticas que fortalezcan el conocimiento y habilidades de las competencias en la malla

curricular. El proceso de control del robot hasta ahora ha sido únicamente por programación textual es decir mediante un panel donde se deberá programar las posiciones de toda la trayectoria que se desee generar, realizar esta programación en grandes industrias genera un tiempo mayor en relación a las que ya implementan programación guiada es decir generar la trayectoria en base el posicionamiento punto a punto del robot, estos avances tecnológicos deberán ser tomados en cuenta en el proceso de aprendizaje de las futuras generaciones de los estudiantes de la carrera, implementando nuevos métodos de control y mando del brazo robótico Mitsubishi RV-2SDB.

Mediante Acuerdo Ministerial No. 00126 de 11 de marzo de 2020, la ministra de Salud Pública declaró el estado de emergencia sanitaria para impedir la propagación del Covid-19, entre las disposiciones generales esta: “ Como adopción de medidas de prevención en el COVID-19, se promoverá el uso de mecanismos como teletrabajo, teleducación, entre otros, con el objetivo de evitar la propagación del virus. ” , resolvió la suspensión inmediata de las clases a partir del mediodía del jueves 12 de marzo de 2020 (MSP, 2020). Los centros de Educación Superior adoptaron medidas para una nueva modalidad de enseñanza, haciendo uso de herramientas virtuales tales como simuladores, videos y plataformas que les permitan alcanzar el aprendizaje deseado

1.3 Justificación e importancia.

El brazo robótico Mitsubishi MELFA RV-2SDB posee un controlador CR1D-700 series donde se logra programar el movimiento del manipulador a través de posiciones donde el usuario debe indicar paso a paso la ruta (programación textual), los brazos robóticos poseen un software propietario que generalmente utiliza código cerrado, sólo puede programarse desde la consola de programación o la unidad de control, el controlador ofrece un puerto de comunicación para implementar un nuevo método de

operación off-line capaz de generar trayectorias de manera sencilla y fácilmente manipulable.

La programación off-line permite desarrollar, verificar y validar los programas antes de su implementación sin la necesidad de disponer del robot. Esto supone que el programador no necesita el entorno real de trabajo. También elimina el tiempo de inactividad del robot real al momento de probar y ejecutar las trayectorias designadas por los usuarios ya que estas fueron configuradas previamente en el simulador virtual el cual utiliza un software que permite simular los movimientos del robot, utilizando para ello un modelo cinemático. De esta forma se puede disponer de un modelo gráfico de la tarea que permita manipular virtualmente el robot como si se realizará físicamente sobre él, dando mayor oportunidad de aprendizaje en casa para los estudiantes y a la vez se asegura de evitar posibles riesgos al equipo.

Un estudio realizado por la Universidad de Warwick establece que el uso de aplicaciones de realidad virtual ayuda a la retención del conocimiento y optimiza el tiempo en la comprensión, surgiendo un mayor interés en los estudiantes a comparación de las metodologías de aprendizaje como video o lectura. (Allcoat & von Mühlenen, 2018)

La utilización de nuevas tecnologías en el campo de la robótica dispone que la interacción humano máquina debe establecer un ambiente seguro y práctico, es por eso que al crear un entorno de realidad virtual ofrece a los usuarios los recursos tecnológicos multimedia necesarios para desarrollar actividades, ofreciendo una interacción casi real del área de trabajo para asegurar una instrucción académica adecuada y proyectar al desarrollo industrial en diferentes campos para beneficio de la comunidad.

Estudios destacan que la realidad virtual en el enfoque no inmersivo ofrece un entorno que permite una comunicación con mayor aceptación por los usuarios a través de los medios tecnológicos como un computador, dando beneficios de bajo costo en la implementación de dispositivos y de fácil uso. Por lo general el individuo para acoplarse al sistema prefiere manipular con controles el ambiente virtual como mandos, teclados, etc.

Este proyecto propone enfatizar el interés, motivación y participación de estudiantes para controlar y manipular el robot Mitsubishi MELFA RV-2SDB con un aprendizaje previo, programando las trayectorias deseadas, utilizando el modelo tridimensional en simulaciones virtuales del entorno de trabajo en modo off-line para conseguir una previa verificación de los datos realizados y posteriormente ejecutarlos en el manipulador robótico como una nueva estrategia de enseñanza en la docencia, con un sistema de código abierto y ambiente amigable, logrando que los estudiantes puedan reforzar los conocimientos prácticos desde sus hogares.

1.4 Objetivos

1.4.1 Objetivo General

Diseño e implementación de un sistema de operación off-line del brazo robótico Mitsubishi RV-2SDB a través de realidad virtual no inmersiva para contribuir en el proceso de aprendizaje de robótica industrial en el laboratorio de Mecatrónica de la Universidad de las Fuerzas Armadas ESPE Sede Latacunga.

1.4.2 Objetivos específicos

- Investigar sobre el uso e implementación de sistemas de operación off-line en robots manipuladores basados en realidad virtual no inmersiva a través de la

recolección de información técnica para visualizar el tipo de control de un robot manipulador.

- Diseñar el modelo 3D y la representación cinemática del robot manipulador considerando el punto de interés como herramienta que describe el movimiento del sistema para la obtención del controlador adecuado.
- Desarrollar el entorno virtual no inmersivo de un brazo robótico y la comunicación de los datos del dispositivo del mando que genera el movimiento a partir de la programación de un software orientado a la creación de videojuegos para visualizar el comportamiento del robot de manera virtual de acuerdo al modelo real.
- Implementar el sistema de operación off-line a través de la comunicación entre el controlador del brazo robótico Mitsubishi RV-2SDB y el entorno de realidad virtual para visualizar el funcionamiento de las trayectorias generadas.
- Determinar si el uso de realidad virtual en operación off-line del brazo robótico contribuye con el proceso de aprendizaje en robótica industrial mediante pruebas de funcionamiento con los estudiantes de la carrera.

1.5 Hipótesis

¿El diseño e implementación de un sistema de operación off-line del brazo robótico Mitsubishi RV-2SDB a través de realidad virtual no inmersiva contribuirá en el proceso de aprendizaje de robótica industrial en el Laboratorio de Mecatrónica de la Universidad de las Fuerzas Armadas ESPE Sede Latacunga?

1.6 Variables de la investigación

1.6.1 Variable Independiente

Sistema de realidad virtual no inmersiva con operación off-line del brazo robótico Mitsubishi RV-2SDB

1.6.2 Variables Dependientes

Contribución en el proceso de aprendizaje de robótica industrial

Capítulo II

2. Fundamentación teórica e introducción

2.1 Robótica industrial

Es una ciencia aplicada, que se origina a partir de la combinación de tecnología de máquina herramienta e informática, su objetivo principal es ayudar a incrementar productividad, eficiencia y minimizar error en las cadenas de producción.

Las herramientas CAD en los últimos años han tenido un impacto decisivo en el desarrollo de simuladores robóticos. La simulación favorece la investigación previa a su instalación como la asignación de máquinas, tipo de robot, la depuración de rutas y la optimización del tiempo de trabajo. (Mora et al., 2002)

Es así que surge la programación off-line debido a que un simulador puede desarrollar su código en base al entorno virtual, reemplazando la tecnología de programación tradicional, que está siendo ampliamente utilizada en robótica en el campo industrial.

2.2 Espacios virtuales de aprendizaje

Los espacios virtuales de aprendizaje (EVA) nacen a partir del desarrollo de la educación virtual, aquella que se respalda por los avances que presenta día a día la tecnología, se caracterizan por enfatizar el desarrollo de conocimientos en los estudiantes introduciendo metodologías didácticas, que mejora el aprendizaje autónomo y su formación integral de un estudiante, las plataformas que se suelen usar son de fácil acceso con respecto al tiempo y espacio, permitiendo una mejor interacción entre contenido – docente y alumno.

2.3 Manipuladores Robóticos

Un manipulador robótico consiste en una cadena de cuerpos rígidos conocidos como eslabones conectados entre sí por mecanismos de traslación o rotación que funcionan como articulaciones. El efector final forma parte del último eslabón donde le permite tomar objetos dependiendo su arquitectura. (Mendoza, 2019, p. 7)

Un manipulador puede ser analizado de diferentes formas, una de ellas es hacerlo en referencia a su parte mecánica, donde se puede mencionar que está compuestos por 3 partes fundamentales, el brazo para alcanzar una cierta posición dentro del plano xyz, una muñeca para dar mejor movimiento de orientación y un efector final que describe la aplicación o tarea objetiva.

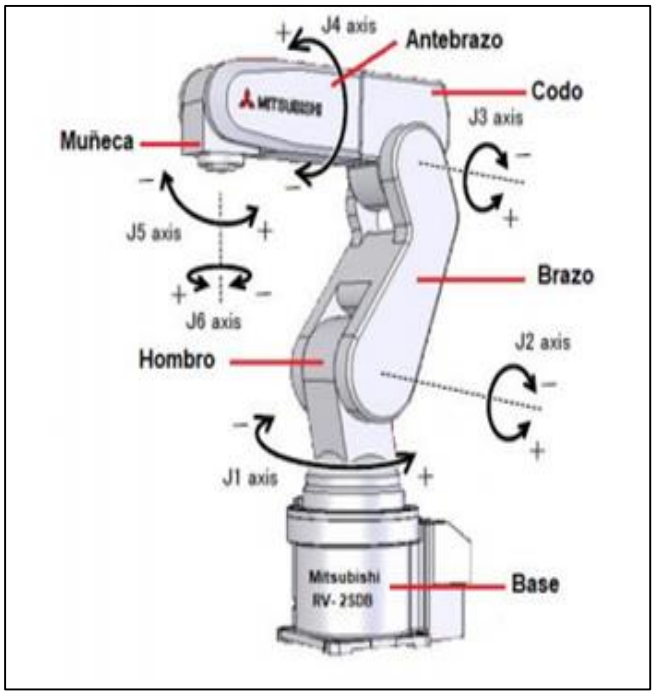
Las articulaciones del manipulador robótico cumplen actividades en específico definido por diferentes trayectorias, “existen varias técnicas de control lineal y no lineal usadas para el control articular de los manipuladores robóticos, considerando el manipulador como un sistema: una-entrada/una-salida (Single Input Single Output / SISO), o como un sistema: múltiples-entradas/múltiples-salidas (Multiple Input Multiple Output MIMO)” (Mendoza, 2019, p. 9)

2.4 Brazo robótico Mitsubishi MELFA RV – 2SDB

El MELFA RV - 2SDB de Mitsubishi Electric es un robot creado para aumentar la capacidad de producción en las diferentes células de trabajo, poseen libertad de movimiento para desarrollar tareas de mayor complejidad en espacios pequeños, alta velocidad y tiempos de ciclo de 0,6 segundos. Es un brazo articulado multi ejes vertical que brinda seguridad y soporte por su posicionamiento de piezas con una gran precisión. (MITSUBISHI ELECTRIC, 2010). Las articulaciones se visualizan en la Figura 1.

Figura 1

Partes del Mitsubishi MELFA RV – 2SDB.



Nota. Ejes y las direcciones de rotaciones que posee el robot Mitsubishi MELFA RV – 2SDB. Tomado de *MELFA ROBOTS Industrial Robot Instruction Manual* [Gráfico], por Mitsubishi Electric Industrial, 2010, Direct Industry.

2.4.1 Características del Mitsubishi MELFA RV – 2SDB

Las características técnicas que muestra el manipulador se muestran en la Tabla 1, también se detallan las configuraciones en la dimensión y rango de operación de las articulaciones en la Figura 2. Especificaciones Técnicas del Robot MELFA RV-2SDB

Tabla 1

Especificaciones Técnicas del Robot MELFA RV-2SDB

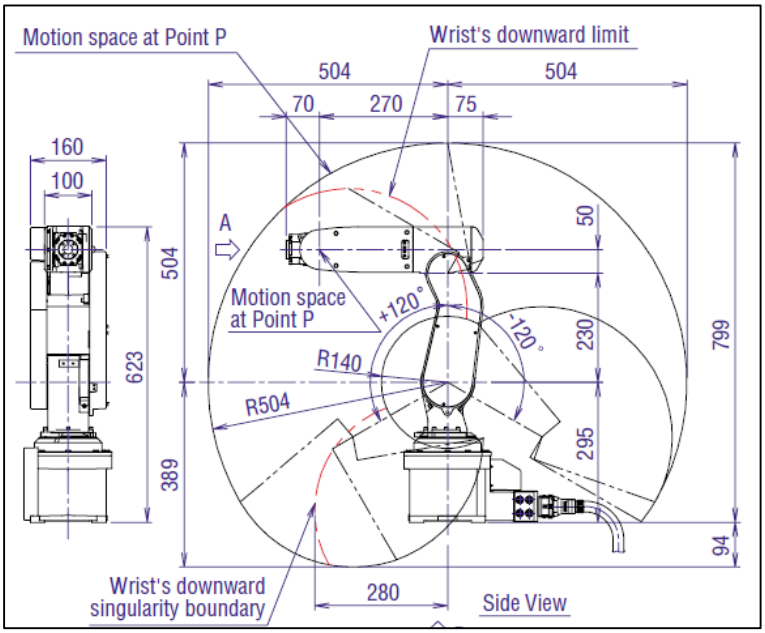
Item	Especificación
Tipo	RV-2SDB

Item		Especificación
Grado de Protección		IP30
Estructura		Brazo Articulado Vertical
Controlador		CR1DA-700
Comunicación del Controlador		RS-232/RS-422/Ethernet/USB
Lenguaje del Robot		MELFA BASIC V
Grados de Libertad		6
Longitud de Brazo (mm)		230+270
Radio de Alcance Máximo (mm)		504
Rango de Operación (deg)	J1	225
	J2	150
	J3	275
	J4	412
	J5	450
	J6	720
Velocidad Máxima (deg/s)	J1	225
	J2	150
	J3	275
	J4	412
	J5	450
	J6	720
Capacidad de Carga (kg)	Nominal	2
	Muñeca (máx.)	3
Repetibilidad (mm)		± 0.02
Masa (kg)		19

Nota. Detalles principales de las especificaciones técnicas del Robot. Tomado de *RV-2SDB / RV-2SQB Series MELFA* [Gráfico], por Mitsubishi Electric Industrial, 2010, Direct Industry

Figura 2

Dimensión y rango de operación del Brazo Robótico



Nota. Detalles de las dimensiones del robot Mitsubishi MELFA RV – 2SDB, en mm. Tomado de *RV-2SDB / RV-2SQB Series MELFA* [Gráfico], por Mitsubishi Electric Industrial, 2010, Direct Industry.

2.4.2 Controlador brazo robótico RV – 2SDB (CR1DA-700)

El controlador de un robot es presentado como el cerebro del sistema, ya que su funcionamiento radica en el sistema de control para las aplicaciones dadas, el brazo robótico Mitsubishi RV - 2SDB tiene un controlador CR1DA – 700 del cual se presentan las siguientes especificaciones más importantes en la Tabla 2 y sus componentes principales en la Figura 3.

Tabla 2

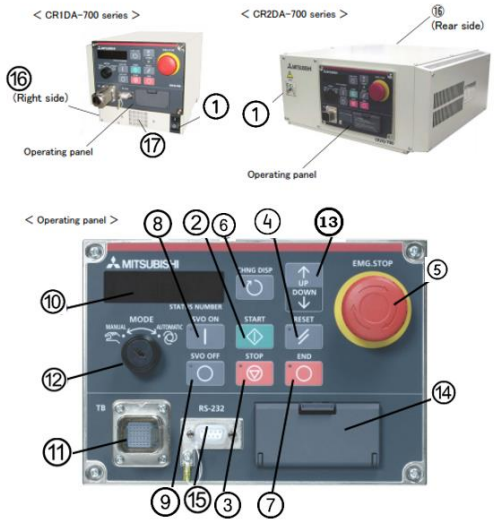
Especificaciones Técnicas del controlador CR1DA - 700

Item		Especificación
Tipo		CR1DA 770
Numero de ejes de control		Simultaneamente 5/6
Lenguaje del Robot		MELFA BASIC V
Protección		IP20
Interfaz	RS – 232C	1 puerto para expansion a conexión con una computadora personal
	Ethernet	1 ranura para Teaching Pendant (T/B), 1 para el usuario
	USB	1 puerto de 2.0
	Ranura dedicada	1 ranura dedicado a neumática
	Ranura de opción	1 para interfaz opcional que se requiera instalar
Fuente de alimentación	Rango de voltaje de entrada	1 fase / AC (180 – 253) V
	Capacidad de Potencia	0,5 KVA
Dimensiónn del esquema(mm)		240(W)*290(D)*200(H)
Masa (kg)		Aproximadamente 9

Nota. Detalles principales de las especificaciones técnicas del controlador. Tomado de *RV-3SD/3SDJ/3SDB/3SDBJ Series Standard Specifications Manual*, por Mitsubishi Electric Industrial, 2010, p. 51, Direct Industry,

Figura 3

Partes del controlador CR1DA – 700



Nota. Partes fundamentales del controlador indicados por ítems con números, el listado correspondiente a cada punto se encuentra en la sección de abajo. Tomado de *RV-3SD/3SDJ/3SDB/3SDBJ Series Standard Specifications Manual* [Gráfico], por Mitsubishi Electric Industrial, 2010, Direct Industry

A continuación, se presenta el listado de cada parte indicada en la Figura 3.

1. Interruptor ENCENDIDO
2. Botón START
3. Botón STOP
4. Botón RESET
5. Interruptor EMERGENCIA
6. Botón DISP
7. Botón Final
8. Botón SVO.ON
9. Botón SVO.OFF
10. Número de Estado, pantalla del panel

11. Conexión del conector T/B
12. Interruptor de llave de modo
13. Botón ARRIBA/ABAJO
14. Cubierta de Interfaz
15. Conector RS – 232
16. Cubierta Terminal
17. Filtro

Cabe recalcar en este apartado una de las características esenciales del controlador es el lenguaje de programación, como se menciona en la Tabla 2 se usa MELFA-BASIC.

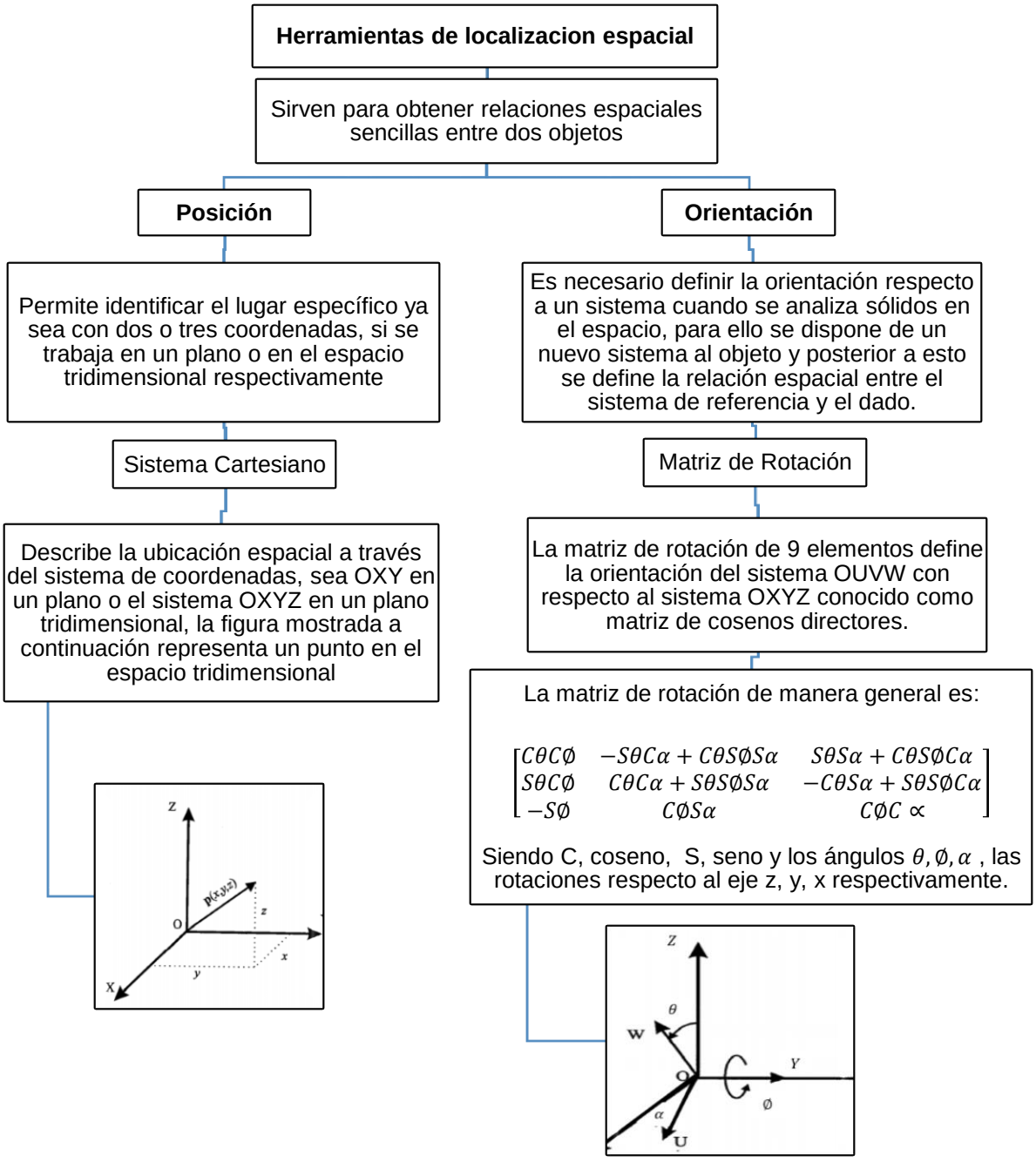
En este lenguaje de programación se estructura como un conjunto de instrucciones cuyo flujo de proceso se realiza en BASIC estándar. El aspecto de un programa es un conjunto e instrucciones propias del sistema del Robot entre sentencias ya conocidas BASIC. Se obtiene una forma de programación intuitiva de programación, sencilla incluso para aquellos usuarios con pocos conocimientos. (Fernandez, s/f)

2.5 Herramientas de localización espacial

Según Barrientos et al. (2007), las herramientas de localización espacial pueden ser descritas como el mapa presentado a continuación.

Figura 4

Marco organizativo de herramientas de localización espacial



Nota. El sistema de localización espacial viene relacionado con la configuración del robot.

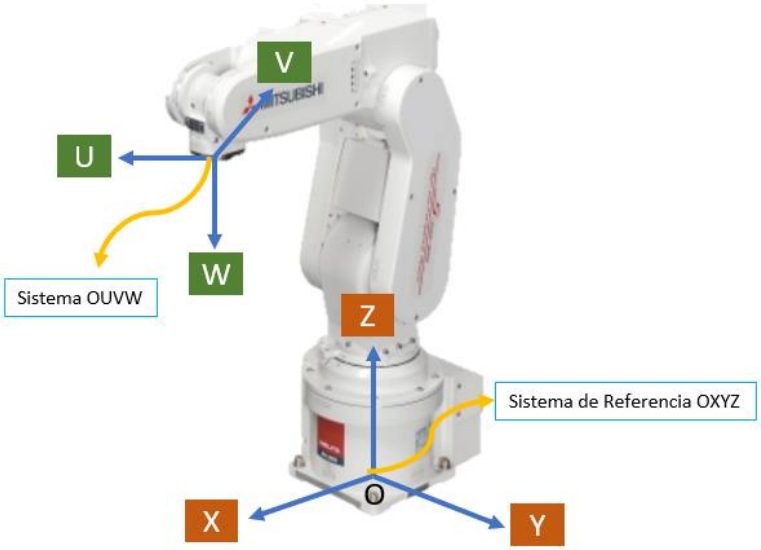
2.6 Control cinemático de un robot manipulador

Estudia su movimiento en relación con un sistema de referencia. No es necesario considerar las fuerzas involucradas. (Barrientos et al., 2007)

El estudio del control cinemático se apoya en la generación de movimientos frente a un sistema de referencia describiendo analíticamente las coordenadas correspondientes del efector final del robot y el movimiento espacial en función del tiempo. En robótica, hay dos maneras de poder manipular un robot mediante el modelo cinemático directo y el modelo cinemático inverso. (Barrientos et al., 2007)

Figura 5

Configuración del sistema de referencia OXYZ y OUVW



Nota. Configuración de los sistemas de referencia base OXYZ y un punto en el espacio para el sistema OUVW del brazo robótico Mitsubishi RV-2SDB.

2.6.1 Control cinemático directo

La cinemática directa, se basa en encontrar la posición y orientación del extremo o efector final del robot mediante la matriz homogénea de transformación denominada T, dicha matriz se compone de variables que están en función a los ángulos de sus

articulaciones del manipulador que permite describir la localización de cada uno de los eslabones basándose en un sistema de referencia situado en la base del robot.

Existen diferentes métodos para resolver la cinemática directa, los mismos se describen a continuación:

- **Resolución mediante matrices homogéneas**

Consiste en encontrar una matriz que relacione los valores de las articulaciones con la ubicación espacial del extremo del robot, para dicha localización se seleccionan coordenadas cartesianas para la posiciones y ángulos de Euler para representar la orientación.

Un robot está formado por cada articulación – eslabón, donde n constituye los grados de libertad (GDL), cada eslabón se asocia a un sistema de referencia propio a él, gracias a la matriz transformación homogénea se puede manifestar las translaciones y rotaciones correspondientes entre distintos eslabones del robot, si se mantiene dos eslabones consecutivos se denomina matriz ${}^{i-1}A_i$.

Con, ${}^{0}A_1$ es la matriz de referencia solidario al primer eslabón con respecto al sistema de referencia solidario base, ${}^{1}A_2$ describe el segundo eslabón respecto del primero, etc. (Barrientos et al., 2007, p. 95)

Así, cuando se desea relacionar todos los grados de libertad, a la matriz ${}^{0}A_n$ suele denominarse T, para un robot de 6 grados de libertad viene dada por:

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

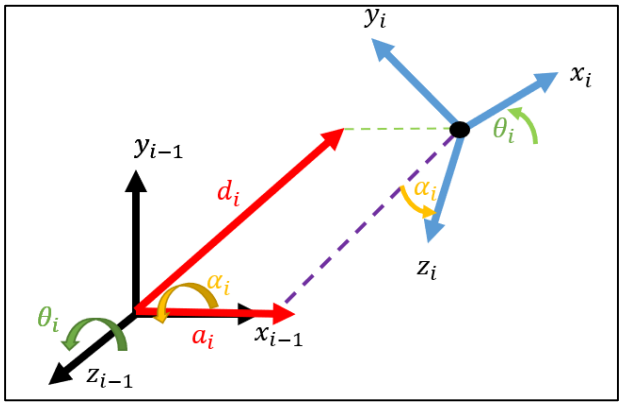
Como el producto de matrices no es conmutativo, las transformaciones se dan en siguiente orden:

$$^{i-1}A_i = T(z,\theta_i)\,T(0,0,d_i)\,T(a_i,0,0)\,T(x,\alpha_1) \quad (2)$$

Gráficamente el procedimiento se logra visualizar en la Figura 6:

Figura 6

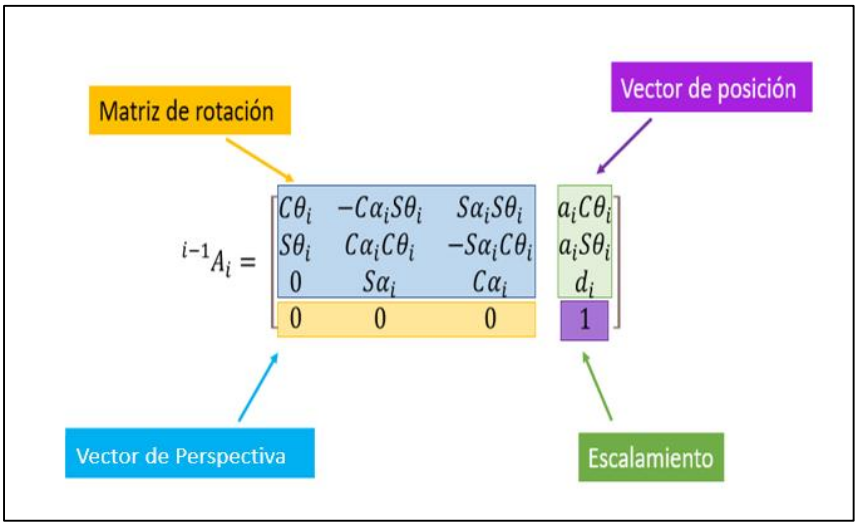
Procedimiento geométrico de la MTH



Haciendo ese producto se obtiene la matriz mostrada en la Figura 7.

Figura 7

Matriz de Transformación Homogénea



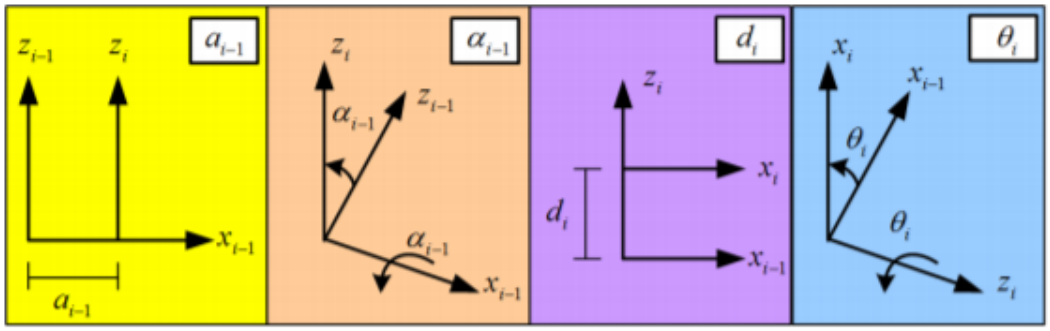
Nota. Aquellos parámetros $\theta_i, \alpha_i, d_i, a_i$ se obtienen del algoritmo D-H para cada eslabón i.

- **Algoritmo para obtener los Parámetros Denavit-Hartenberg**

Método matricial para determinar la ubicación de cada sistema de coordenadas de cada eslabón y que estará vinculado a la cadena articulada para obtener de forma sistemática la ecuación cinemática de la cadena completa. Según Barrientos et al. (2007) deduce a una tabla con cuatro parámetros de D-H $(\theta_i,\ d_i, a_i,\ \alpha_i)$ que dependen únicamente de las características geométricas de cada eslabón y de las articulaciones como se muestra en la Figura 8.

Figura 8

Descripción gráfica de los parámetros D-H



Nota. Tomado de Robótica: Análisis, modelado, control e implementación (p.65), por Hernández, et al., 2015, OmniaScience.

Para el cálculo cinemático se logra utilizar herramientas matemáticas que ayuden a definir la posición y orientación a partir de matrices homogéneas, como por ejemplo al utilizar los parámetros de Denavit-Hartenberg (D-H) obtenidos para la representación geométrica de la cadena cinemática del posicionamiento del brazo robótico.

- **Resolución mediante Cuaterniones**

Un cuaternio constituido por 4 elementos, definido como Q (q0, q1, q2, q3) representan las coordenadas del cuaternio en una base {**e, i, j, k**}, logrando la

orientación relativa del sistema O'UVW con respecto a otro, **e** se define el escalar del cuaternio y los componentes **i, j, k** los vectores. (Barrientos et al., 2007)

Además, tiene la capacidad de independizar los ángulos de giro frente a un objeto, logrando eliminar el efecto Gimbal Lock al utilizar ángulos de Euler.

2.6.2 Control cinemático inverso

Consiste en buscar los valores de las coordenadas de cada articulación del robot, es decir $q = [q_1, q_2, \ldots . . q_n]$ en base a un punto determinado en el espacio en el que se encuentra situado el extremo del robot (Barrientos et al., 2007).

El análisis cinemático inverso conlleva un estudio donde se considera la configuración del robot al momento de resolver y obtener ecuaciones, para conseguir mejores resultados es óptimo hacerlo mediante una solución cerrada, es decir encontrar una relación explícita de la forma:

$$q_k = f_k(x, y, z, \alpha, \beta, \gamma) \quad k = 1 \ldots . n \tag{3}$$

Las soluciones cerradas presentan ventajas en comparación a las soluciones con métodos iterativos ya que aquellos pueden presentar deficiencia en la convergencia y pueden tener obstáculos con la velocidad del robot.

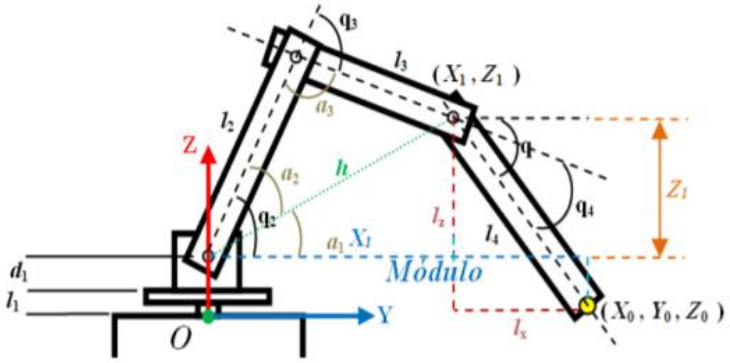
Cuando se tienen robots con pocos grados de libertad, lo más fácil es resolver el problema por métodos geométricos ya que permite posicionarlo a través de relaciones trigonométricas y geométricas; sin embargo, existirán muchos manipuladores que son más usados en diversas aplicaciones y los mismos disponen de más grados de libertad haciendo que su resolución se realice por otros métodos presentados a continuación:

- **Resolución por métodos geométricos**

Como ya se mencionó en el apartado anterior, este método funciona netamente para robots con pocos grados de libertad, el procedimiento a seguir es buscar relaciones geométricas en las que intervendrán las coordenadas del extremo del robot, sus coordenadas articulares y las dimensiones físicas de sus elementos, un ejemplo del análisis geométrico se muestra en la Figura 9.

Figura 9

Relaciones geométricas de un brazo robótico de 3 GDL



Nota. Tomado de “Generación de trayectorias para el brazo robótico (ArmX)” (p.61), por D. Milanés, y A. Castilla, 2016, Revista de Ingeniería Electrónica, Automática y Comunicaciones, 37.

- **Resolución a partir de la transformación homogénea**

Este método parte de la matriz de transformación homogénea encontrada de manera directa ya que ahora se conoce la posición y orientación de la ubicación espacial del punto determinado en el extremo del robot (en el efector final) y se desea encontrar los valores de cada articulación, tomando en cuenta que se llega a obtener 12 ecuaciones para un robot de 6 grados de libertad y lo específico es optar por poseer solo 6 ecuaciones, lo que conlleva a una toma de decisiones sobre cuál de ellas será la

mejor opción, los inconvenientes se presentan cuando se trabaja en trayectorias que describen puntos en tiempo real.

- **Desacoplo cinemático**

Este método es aplicable donde los 3 últimos grados de libertad del robot manipulador, es decir las articulaciones que dan orientación del extremo del manipulador localizado en el espacio intersecan en un mismo punto.

Se llama desacoplo porque separa la resolución en dos partes: posición y orientación, para encontrar la posición en el espacio dado un punto xyz se calcula mediante métodos ya mencionados ya que su análisis conlleva a solo encontrar los valores de los 3 primeros grados de libertad, en su mayoría utilizan el método geométrico para encontrar (q_1, q_2, q_3). A partir de los datos de orientación y las variables ya calculadas se consiguen encontrar las últimas 3 articulaciones (q_4, q_5, q_6) únicamente tomando en cuenta las matrices de rotación de las mismas.

- **Resolución a través de la matriz Jacobiana inversa**

"La matriz jacobiana inversa permite conocer las velocidades articulares necesarias para obtener las velocidades determinadas en el extremo del robot. Para obtener la matriz jacobiana inversa se parte de aquella obtenida en el análisis directo invirtiendo simbólicamente la matriz" (Barrientos et al., 2007, p.139) como se muestra en la ecuación (4).

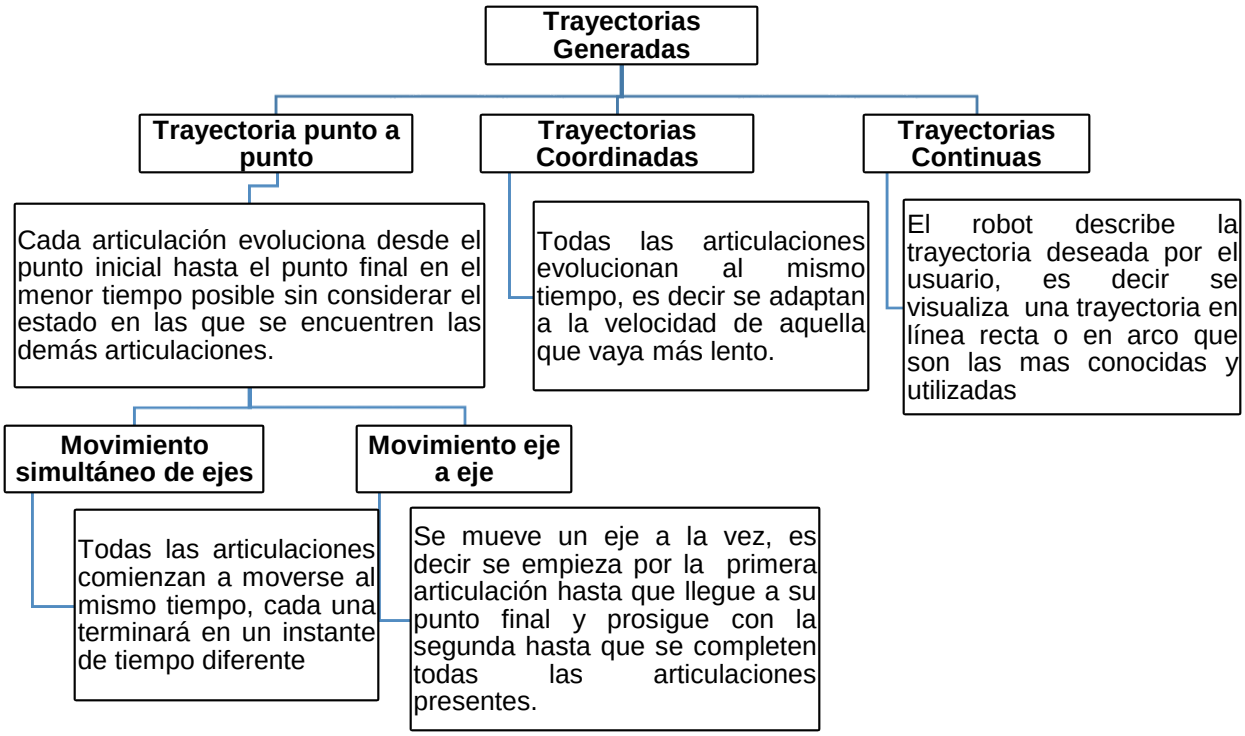
$$\begin{bmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_n \end{bmatrix} = J^{-1} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \end{bmatrix} \tag{4}$$

2.6.3 Trayectorias generadas en un control cinemático

Según Barrientos et al., (2007) existen diferentes trayectorias que pueden describir un manipulador robótico a la hora de ir desde un punto inicial a punto final determinado, en base a eso se determinan ciertos tipos de las mismas, cada una se implementa ya sea por sencillez a partir del control cinemático o por mejor factibilidad en determinadas aplicaciones, las trayectorias se describen en el siguiente mapa conceptual:

Figura 10

Marco organizativo del tipo trayectorias



Nota. Tipos de trayectorias que el robot puede realizar.

2.7 Programación off-line

La programación offline, o simulación, se utiliza más a menudo en la investigación de la robótica para asegurar que los algoritmos de control avanzado

funcionan correctamente antes de pasarlos a un robot real, en base a la definición también puede ser llamado como un “GEMELO DIGITAL”. Suele usarse en la industria para reducir el tiempo de inactividad en la línea de producción, mejorar su eficiencia y predecir errores.

La programación off-line, se debe realizar utilizando un paquete de software que permita simular los movimientos del robot, utilizando para ello un modelo cinemático y en ocasiones dinámico del robot. De esta forma se puede disponer de un modelo gráfico de la tarea que permita manipular virtualmente el robot como sí se realizara físicamente sobre él.

2.8 Realidad virtual

El término “Realidad Virtual” suele asociarse a casi todo aquello que tiene que ver con imágenes en tres dimensiones generadas por ordenador y con la interacción de los usuarios con este ambiente gráfico. Ello supone la existencia de un complejo sistema electrónico para proyectar espacios visuales en 3D y para enviar y recibir señales con información sobre la actuación del usuario, quien, con un sistema de este tipo, puede sentir que se encuentra inmerso en un "mundo virtual". (Otón et al., 2018)

Un mundo virtual es un modelo matemático que describe un “espacio tridimensional”, dentro de este “espacio” están contenidos objetos que pueden representar cualquier cosa, desde una simple entidad geométrica, por ejemplo, un cubo o una esfera, hasta una forma compleja, como puede ser un desarrollo arquitectónico, un nuevo estado físico de la materia o el modelo de una estructura genética (Otón et al., 2018). La RV (realidad virtual) se puede clasificar en inmersiva y no inmersiva.

2.8.1 Realidad virtual inmersiva

Requiere material avanzado y especializado como son cascos y gafas, logrando ampliar la sensación de inmersión con otro tipo de periféricos como accesorios como guantes, pulseras, traje, etc. Lo que hace que el individuo sienta ser parte de ese mundo virtual al completo.

Un ejemplo de aplicación es en las industrias donde se puede observar los procesos que se realizan de forma virtual, como se muestra en la Figura 11, se está haciendo uso de realidad virtual inmersiva ya que enfoca la visualización por medio de gafas especiales.

Figura 11

Realidad Virtual inmersiva en procesos industriales

Nota. Realidad virtual inmersiva de industrias reales. Tomado de *Realidad virtual y realidad aumentada. Tecnología que pide paso en la industria.* [Gráfico], por Ángel Gil Pérez, 2016.

2.8.2 Realidad virtual no inmersiva

Los sistemas no inmersivos son aquellos donde el monitor es la puerta hacia el mundo virtual. Su interacción principal es por medio de teclado, ratón o un mando en el que se pueda comunicar con el mundo virtual. Destinado a sistemas sencillos, menos costosos y adecuados por ejemplo a visualizaciones científicas y experimentaciones.

2.9 Mandos para trabajar con motores gráficos

Los dispositivos de realidad virtual se los considera parte esencial en la era de la nueva tecnología, los mandos ofrecen la posibilidad de implementar una retroalimentación háptica como parte de una interacción multimodal usuario - máquina, las gafas o el casco pueden quedarse en poco si no se dispone de un mando que ayude a interactuar con la realidad.

El dispositivo de mando se muestra como una herramienta que ayuda a captura los movimientos de una máquina y los proyecta al motor gráfico de desarrollo de videojuego.

Según Guevara & Martínez (2018) “los parámetros para la selección del dispositivo de mando podrían ser según su:

- Fácil Instalación y portabilidad
- Buena respuesta de movimiento
- Compatible con el motor de desarrollo
- Costo accesible
- Interacción sencilla” (p. 77).

Figura 12

Mandos de realidad virtual

Nota. Dispositivos de juego. Tomado de *Juego de iconos de joystick y tecnología de gadgets para jugadores*.[Imagen], por KittyVector, 2020, Freepik.

El objetivo principal del uso de un mando físico que trabaja con realidad virtual en el desarrollo del proyecto es realizar ciertos movimientos del brazo robótico a través del mismo, dando el equivalente entre cada articulación del brazo con las diferentes opciones de proyección de movimiento que el mando físico puede presentar, algunos de estos dispositivos se muestran en la Figura 12.

2.10 Tecnología en la educación

Las herramientas tecnológicas requieren que los educadores puedan diseñar entornos efectivos y comprensibles en el campo de la instrucción. La realidad virtual (RV) permite entregar al alumno experiencias didácticas que ayuden a percibir de mejor manera los conceptos impartidos en un aula de clase.

La creación de un mundo virtual interactivo permite tener una versión virtual simulada de un robot existente en un laboratorio real. (Zaldívar, 2003)

2.11 Simulador de realidad virtual en robótica

El desarrollo de aplicaciones de realidad virtual es un fenómeno de tendencia, su uso en la educación impartida en el área de robótica puede considerarse como una de las evoluciones naturales de la instrucción asistida por computadora, con la intención de mejorar la experiencia de aprendizaje y entender de mejor manera las secuencias de movimiento planteadas, seguidas de una evaluación que permita al alumno visualizar la simulación del modelo real, con eso los estudiantes de robótica podrán corregir y hacer cambios en caso de requerirlo. El resultado es un robot virtual con posiciones y movimientos que coinciden con los del robot real.

Las tecnologías disponibles para la programación en la actualidad son las que utilizan simulación virtual. El uso de modelos virtuales del entorno de trabajo y la simulación del propio robot pueden aportar ventajas para empresas y programadores e

incluso en situaciones donde el uso de laboratorios para instituciones de educación superior sea limitado. Es posible visualizar los movimientos del robot y las piezas de ensamblaje en un entorno virtual tridimensional varios meses antes de que se produzca el prototipo. (Hisour, s/f)

2.12 Interfaz de usuario

Conocida como User Interface (UI), permite al usuario visualizar el entorno simulador y poder interactuar con él. Utilizando elementos que ayuden a otorgar información de manera intuitiva o directa realizando cambios o acciones al sistema.

Según Toledo Castro (2019) para un diseño en la interfaz de un videojuego constituye de una combinación de elementos mediante una clasificación. Destacando dos elementos Diegética y No-diegética entre otros.

- Diegética: Tipo de interfaz tiene como objetivo que el personaje pueda ver, escuchar y tocar elementos dentro del espacio del juego.
- No-diegética: Tipo de interfaz fuera del espacio de juego, tiene como objetivo informar e interactuar con el jugador, pero el personaje del juego no es consiente.

2.13 Herramientas computacionales

2.13.1 Software de desarrollo de videojuegos

Denominado también motor de juegos, sirve en la creación de video juegos, integra la programación, gráficos generados por computadora, en los cuales se ven reflejadas las acciones programadas para un robot manipulador. El alto nivel de calidad visual que presenta este tipo de software, requiere para su desarrollo aplicaciones de herramientas avanzadas en el manejo de estructuras de código mediante un lenguaje de programación que permite el funcionamiento de los objetos enlazados entre sí y la

generación de trayectorias que los mismos desean describir, en conjunto con la interacción con interfaces gráficas generadas mediante técnicas de diseño 2D y 3D de buena calidad.

2.13.2 Software de modelado 3D

El modelado 3D es el proceso de desarrollo de una representación matemática de cualquier objeto tridimensional, permiten la creación y manipulación de gráficos 3D.

Autodesk 3DS MAX

Es un Software de modelado, texturización y renderizado 3D para visualización de diseños, juegos y animación. La transferencia de los datos 3D se importa con alta precisión.

La arquitectura basada en plugin puede proporcionar instrucciones ya sea para programas de animación 3D más utilizado, en la creación de videojuegos, arquitectura, películas, etc. Importa archivos CAD y BIM para crear una estructura general y mejorar los detalles de su diseño 3D. Autodesk 3DS Max posee un Game Exporter específicamente para que los usuarios exporten modelos y clips de animaciones en formato *fbx* a su motor de juego de manera más efectiva. (Autodesk, 2021)

Tabla 3

Requerimientos del Sistema en la Instalación De Autodesk 3DS Max

	Software
Sistema Operativo	Microsoft® Windows® 10. de 64 bits.
	Hardware
CPU	Procesador Intel® o AMD® de múltiples núcleos de 64 bits con conjunto de instrucciones SSE4.2

RAM	4 GB de RAM como mínimo (se recomiendan 8 GB o más)
Espacio del Disco	9 GB de espacio libre en disco para la instalación
Dispositivo Señalador	Ratón de tres botones

Nota. Especificaciones del software 3DS Max. Tomado de New Features In 3ds Max 2021, por Autodesk, 2021, Autodesk.

2.13.3 Software de cómputo numérico

El Software ayuda a resolver cálculos numéricos. Los Métodos Numéricos son técnicas algorítmicas basadas en operaciones aritméticas simples para la solución de problemas matemáticos. En el mercado actual varias empresas ofrecen softwares de acceso libre o comercial. (Rodríguez, 2009). Estos programas permiten realizar el cálculo numérico de las iteraciones en el proceso de obtención de la cinemática del robot con el uso de diferentes funciones que se puedan plantear.

Matlab

Este es un software de cálculo numérico ingenieril, ayuda a ejecutar códigos basados en matemáticas computacionales (análisis iterativo y los procesos de diseño), con un lenguaje de programación para desarrollar algoritmos, analizar datos y crear modelos y aplicaciones. (The MathWorks, 2021), una característica fundamental del software para el desarrollo del proyecto es poder leer archivos de texto que se encuentren en diferentes formatos para obtener la información necesaria y desarrollar códigos que el usuario desee generar.

Matlab cuenta con protocolos de transmisión TCP que sirven para la comunicación con un cliente TCP/IP, en este caso el cliente es la controladora *CR1DA – 771* que de igual manera posee un puerto ethernet como se indica en la Tabla 3 lo que permite el envío de datos de control al robot por medio de este protocolo de comunicación.

Peter Corke Robotics Toolbox es un paquete de robótica que proporciona la caja de herramientas de Matlab, posee variedad de funciones que son útiles para investigar y simular robots de brazos clásicos, como su cinemática, dinámica y generación de trayectorias. El paquete tiene funciones que ayudan a tener un mejor análisis de la representación espacial, como matrices, cuaterniones, giros y ángulos de Euler. Ayuda a la manipulación y conversión entre tipos de datos (vectores, transformaciones homogéneas y cuaterniones de unidades) necesarios para representar posiciones y direcciones tridimensionales. (Corke, 2017)

También se puede verificar los datos de los parámetros D-H de un robot manipulador obtenidos de diferentes fuentes para validar la información.

Capítulo III

3. Diseño y selección de componentes

3.1 Método cualitativo por puntos

El método implica asignar principales factores determinantes de una localización, para asignarle valores ponderados de peso relativo de acuerdo con la importancia que se le atribuye el criterio del investigador. Recomiendan los siguientes pasos para clasificar los factores cualitativos:

- Desarrollar una lista de factores relevantes o parámetros a evaluar.
- Asignar un peso a cada factor mostrando su importancia relativa asignado por el juicio del investigador, la suma de los pesos debe ser 1.00.
- Asignar una escala común del 0 a 10 para cada factor (0 representa que No Existe o No Hay y 10 representa que Existe o Si Hay)
- Cada sitio potencial se califica de acuerdo con la escala especificada y luego la calificación se multiplica por el peso.
- Adicionar la puntuación para cada sitio y luego seleccione el factor cualitativo con la puntuación más alta. (Baca, 2010, p.99)

Tabla 4

Método Cualitativo por Puntos

Factor Relevante	Peso Asignado	LUGAR					
		A		B		C	
		Calif.	Pond.	Calif.	Pond.	Calif.	Pond.
Materia Prima Disponible	0,35	5	1,75	5	1,75	4	1,4
Mano de Obra	0,1	8	0,8	3	0,3	3	0,3
Costo de Insumo	0,25	7	1,75	8	2	7	1,75
Costo de Vida	0,1	2	0,2	4	0,4	7	0,7
Mercado	0,2	5	1	8	1,6	6	1,2
Total	1		5,5		6,05		5,35

Nota. Ejemplo del Método cualitativo por puntos. Tomado de Estudio Técnico, por (Baca, 2010).

3.2 Selección del motor de juegos para realidad virtual no inmersiva

El motor de juegos para el desarrollo de realidad virtual es el software principal para realizar los cálculos de la cinemática del robot, visualizar los movimientos del Brazo Robótico Mitsubishi RV-2SDB y generar archivos de comunicación para enlazarse con otros programas, de preferencia que sea un software libre.

A continuación, se presenta el detalle de los dos motores de desarrollo de videojuegos más importantes y usados en la actualidad, posterior a eso se procede a hacer la selección del más adecuado para el proyecto.

3.2.1 Unreal Engine

Es un motor de juego de edición en tiempo real, presenta características realistas en función de sus renderizados, efectos dinámicos, panorámicos y varias

animaciones de diferentes aspectos que quieran presentar, el software es gratuito y se lo puede descargar desde su página principal.

A continuación, se enlista ciertas características determinantes para el desarrollo del proyecto

- El software posee un complemento denominado Datasmith que ayuda a importar alrededor de 25 formatos 3D diferentes e incluso escenas de forma automática para su conversión y uso dentro del software.
- Permite la detección continua de colisiones dividiéndola en dos formas, colisión simple en base al volumen de la forma y colisión compleja donde se toma en cuenta netamente la geometría del objeto.
- Posee funciones relacionadas a las juntas de un cuerpo rígido, esta tiene la capacidad de realizar diferentes movimientos según el tipo de junta designada.
- Soporta dispositivos de entradas para la interacción con el motor gráfico, sin embargo, está limitado a ciertos dispositivos de mandos como los gamepads, si se requiere enlazar dispositivos diferentes se deben crear las los blueprints que permitan su comunicación.
- Posee la funcionalidad de Tonemapper que proporciona colores similares a películas.
- Posee diferentes funciones para dar un realce gráfico realista tales como porcentaje de pantalla, resolución dinámica, aproximadamente tiene 15 funciones importantes en renderizado y texturas para mejorar la calidad de visualización de los objetos y dispone alrededor de 16 funciones posteriores al proceso desarrollado en los efectos visuales que ayudan a determinar un mejor nivel de calidad en los gráficos

- No dispone de tecnología en multiprocesamiento
- El lenguaje de programación usado es C++ y blueprints, en C++ genera archivos .cpp asignado para implementación en el motor y .h que define métodos, objetos o variables que se utilizan para implementarse, en funciones deben declararse de varias librerías a comparación en C# que incluye funciones de cualquier cosa en una sola. Blueprints es un sistema scripting visual con funciones ya establecidas.
- Unreal Engine posee dentro de su plataforma información que ayuda a los programadores a desarrollar sus proyectos de mejor manera, esta documentación se presenta como un tutorial, sin embargo, el apoyo sobre una comunidad destinada netamente a este software es escasa.

3.2.2 Unity

Es un motor de desarrollo de videojuegos multiplataforma, tiene una versión gratuita de descarga y es muy usado en la actualidad por varios desarrolladores de este campo, presenta una interfaz gráfica fácil de comprender, sencilla y cómoda.

A continuación, se enlistan ciertas características determinantes para el desarrollo del proyecto

- El programa es capaz de soportar formatos de importación de modelo tales como: fbx y .obj, estos dos son tipos genéricos los cuales pueden ser usados por la mayoría de softwares de modelado 3D, aparte de eso Unity acepta formatos propios de 7 softwares, es decir puede importar 9 tipos de formatos 3D diferentes.
- Permite la detección continua de colisiones en base a un método basado en barrido y un método especulativo, también simula el fenómeno físico de la gravedad.

- Posee dos funciones relacionadas a las juntas de un cuerpo rígido, esta tiene la capacidad de realizar diferentes movimientos según el tipo de junta designada, lo que da un realce realista y un mayor dinamismo a un objeto 3D.
- Soporta varios dispositivos de entradas para la interacción con el motor gráfico que suelen usarse de manera convencional tales como teclado, joypad, cámaras web, micrófono, gafas, pantallas táctiles, etc. Sin embargo, cabe mencionar que existen funciones predeterminadas para algunos periféricos, lo que facilita su implementación con el proyecto que se esté desarrollando.
- Permite importar texturas en forma de imágenes dentro del proyecto.
- Posee varios ajustes para determinar el nivel de calidad de los gráficos tales como Texture Quality, AntiAliasing, Soft Particles, Shadows y LOD (nivel de detalle) entre 0 - +infinito.
- Dispone de tecnología DOTS, permite ejecutar los proyectos de una manera eficiente, es decir fps más altos, tiene menos consumo de batería ya que es un sistema que integra multiprocesamiento.
- El lenguaje de programación usado es C# y es orientada a objetos, sus partes principales de la sintaxis se deriva en variables, funciones y clases.
- Unity posee una comunidad propia de apoyo y dentro de su plataforma facilita información que ayuda a los programadores a desarrollar de mejor manera sus proyectos.

Para la selección del motor de juego adecuado se determinaron los factores más relevantes en base a las características requeridas para el proyecto y el criterio de los diseñadores, obteniendo así los siguientes puntos:

- **Exportación e importación de modelos**: Se mide la cantidad de formatos con los que el software permite importar y exportar modelos 3D.
- **Simulación de modelos matemáticos**: Determina los fenómenos físicos soportados por el Software provenientes de modelos matemáticos para que sean propuestos en él
- **Nivel de realidad de los gráficos:** Evalúa la realidad que muestra la gráfica, se determinan funciones de textura, iluminación, perspectivas, anti-aliasing y nivel de detalle de los objetos.
- **Opciones de multiprocesamiento**: Presenta funciones o tecnología que permita al software desarrolla de mejor manera, sus flujos con multiprocesamiento, dando un realce al rendimiento.
- **Dispositivos de entrada:** Valora el uso de dispositivos de entradas y a su vez que sean de fácil manipulación para el usuario, los dispositivos pueden ser periféricos como: Mouse, guantes y joysticks que permiten la navegación e interacción dentro del medio ambiente virtual
- **Documentación y comodidad:** Evalúa la documentación necesaria que el software presenta frente a múltiples proyectos desarrollados dentro de una comunidad de programadores orientados en este campo, este factor es uno de los más importantes a considerar ya que se requiere de la mayor cantidad de información respecto a funciones relacionadas con la cinemática de un robot manipulador.

Tabla 5

Selección del motor de juegos para realidad virtual no inmersiva

Factor Relevante	Peso Asignado	Lugar			
		Unreal Engine		Unity	
		Calif.	Pond.	Calif.	Pond.
Numero de Formatos 3D que permite importar	0,1	9	0.9	8	0,8
Funcionalidad para junturas y fenómenos físicos	0,2	7	1,4	8	1.6
Funcionalidad para dar realismo a los gráficos	0,1	9	0,9	7	0,7
Posee tecnología en multiprocesamiento	0,1	5	0,5	9	0,9
Facilidad en enlazar con dispositivos de entrada	0,2	7	1,4	9	1,8
Documentación y Comunidad	0, 3	7	2,1	10	3
Total	1		7,2		8.8
Selección			NO		SI

De acuerdo con la Tabla 5 la alternativa ideal para el motor de juegos es Unity debido facilidad de uso y la integración de dispositivos de entrada que ayuda a cumplir con los requerimientos solicitados.

3.2 Selección del mando para realidad virtual no inmersiva

El mando para realidad virtual no inmersiva será el encargado de trasladar por medio de los botones los movimientos respectivos de cada eje del brazo robótico Mitsubishi RV-2SDB, hay que tener en cuenta el tipo de comunicación que tendrá con la

computadora, también si se permite la funcionalidad de los botones como entradas al simulador virtual, la autonomía, velocidad de comunicación y como esta se ajusta a la comodidad del usuario.

3.2.1 Wii Remote

El Wii Remote es un dispositivo inalámbrico que utiliza comunicación Bluetooth estándar, cuenta con 9 botones de entrada y posee internamente acelerómetros que permite identificador los movimientos que el usuario realiza con él, en la Figura 13 se puede visualizar el dispositivo como tal.

Figura 13

Wii Remote

Nota. Control del Nintendo Wii “Wii Remote”. Tomado de *Wii Remote* [Imagen], por Nintendo Wiki Fandom, s/f, FANDOM.

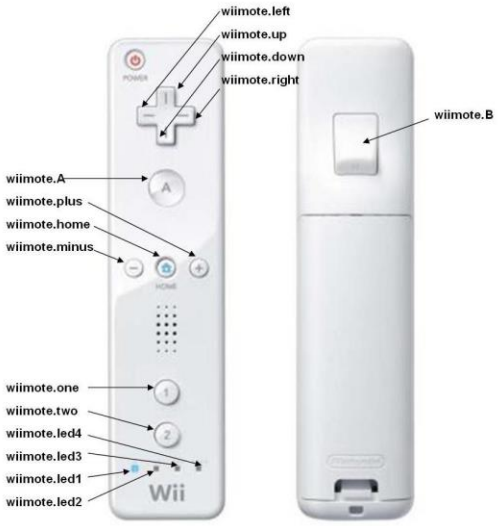
Características técnicas del Wii Remote.

- Microcontrolador Broadcom BCM2042 Bluetooth
- Chip EEPROM de 16 kB
- Acelerómetro lineal de tres ejes ADXL330.

- Fuente de alimentación 2 pilas AA, dependiendo del tipo y como esta se usa puede llegar a durar alrededor de 30 horas.
- 4 LEDS del jugador o LED del reproductor remoto de Wii Remote.
- Posee vibración a los movimientos
- La conexión inalámbrica cubre aproximadamente 10 metros
- Posee un diseño ergonómico de fácil manipulación para que pueda ser usado por profesionales y principiantes.
- Tiene un peso de 0.46 lb

Figura 14

Elementos básicos del Wii Remote



Nota. Detalle de los elementos que se encuentran en Wii Remote. Tomado de *Wiimote*[Imagen], por WiiBrew, 2020, Wiibrew.

Elementos básicos del Wii Remote

- Wii Motion Plus: cambio de cabeceo / guiñada / alabeo
- Datos básicos de los botones (A, B, +, -, 1, 2, D-Pad, botones de inicio)

Cabe destacar que uno de los motores de desarrollo virtual Unity posee una librería diseñada por el usuario flafla2 de manera gratuita en la plataforma GITHUB (Biagioli et al., 2016), que permite la comunicación entre el software y el mando Wii Remote capaz de interpretar todos los datos que el dispositivo puede enviar.

3.2.2 HTC Vive controllers

Los controladores HTC Vive posee un diseño ergonómico, cómodo y de fácil manejo. Permiten comunicarse de forma inalámbrica a través de bluetooth, dispone de 24 sensores, un trackpad multifunción y una retroalimentación táctil para aplicaciones desarrolladas en sistemas de realidad virtual, en la Figura 15 se puede visualizar este tipo de dispositivo

Figura 15

HTC Vive Controller

Nota. HTC Vive Controller. Tomado de *Controller HTC VIVE* [Imagen], por HTC Corporation, 2018, VIVE.

Características técnicas del HTC Vive Controller

- Microcontrolador NXP Semiconductors 11U37F ARM Cortex-M0
- Microprocesador Invensense MPU-6500
- Combinación de acelerómetro y giroscopio de 6 ejes

- Batería de polímero de litio de 3.85 V, 3.69 Wh y 960 mAh, llega a durar 4 horas.
- Micro M25P40 Memoria flash serial de 4 Mb
- Regulador de voltaje lineal Semiconductor 61AKE6U L00075B
- Cargador de batería TI BQ24158 I
- Tiene un peso de 0.7425 lb

Figura 16

Desmontaje de HTC Vive Controller

Nota. HTC Vive Controller desarmado. Tomado de *HTC Vive Teardown* [Imagen], por (iFixit, 2016b), iFixit.

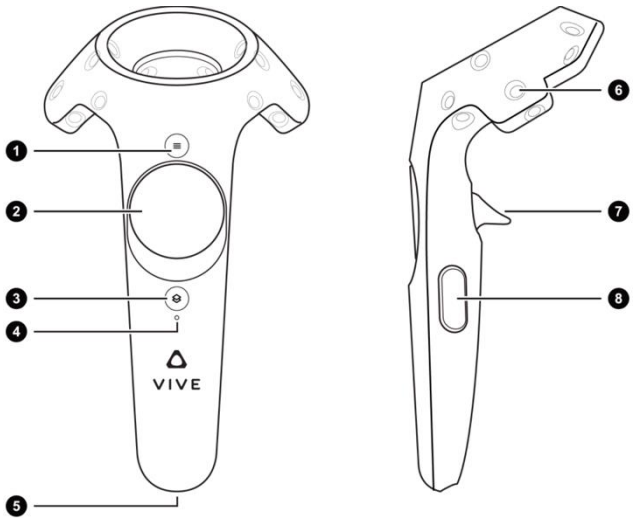
Elementos básicos del HTC Vive Controller

1. Botón de Menú
2. Trackpad
3. Botón de sistema
4. Luz de estado

5. adaptador de carga USB
6. Sensor de seguimiento
7. Gatillo
8. Botón de agarre

Figura 17

Elementos del HTC Vive Controller



Nota. Detalle de los elementos que se encuentran en la HTC Vive Controller. Tomado de *Acerca de los controladores VIVE* [Imagen], por HTC Corporation, s/f, VIVE.CC BY 2.0

Unity ofrece una librería sencilla, una API Open VR que está ya colocada en el motor de juego, esta es apta para colocar en su interfaz las asignaciones del sistema de mando HTC Vive. A través de su documentación permite ingresar los comandos para su posterior mapeo del dispositivo.

3.2.3 Oculus Touch Controllers

Mandos desarrollados por Oculus VR para sistemas de realidad virtual. Este par de controladores con seguimiento que permiten una presencia intuitiva de las manos en

la realidad virtual. Brindan la sensación de que las manos virtuales son las reales. Requiere sensores para el movimiento, el dispositivo se puede apreciar en la Figura 18.

Figura 18

Oculus Touch Controllers

Nota. Oculus Touch Controllers. Tomado de *Características del Oculus Rift S* [Imagen], por Oculus VR, s/f, Oculus.CC BY 2.0

Características técnicas

- Nordic Semiconductor nRF51822 Bluetooth Smart y SoC patentado de 2,4 GHz
- Controlador de sensores de capacitancia Analog Devices AD7147
- Acelerómetro y giroscopio combinado de 6 ejes
- Microprocesador Invensense MP651
- Dos controladores táctiles
- Cada controlador táctil incluye 24 LEDs IR para seguimiento
- Posee un motor de vibración háptica
- Función de una batería AA
- Posee un peso de 1lb
- Las pilas tienen una duración entre 20 - 30 horas

Figura 19

Desmontaje de Oculus Touch

Nota. Oculus Touch Controllers desarmado. Tomado de *Oculus Touch Teardown* [Imagen], por (iFixit, 2016a), iFixit. CC BY 2.0

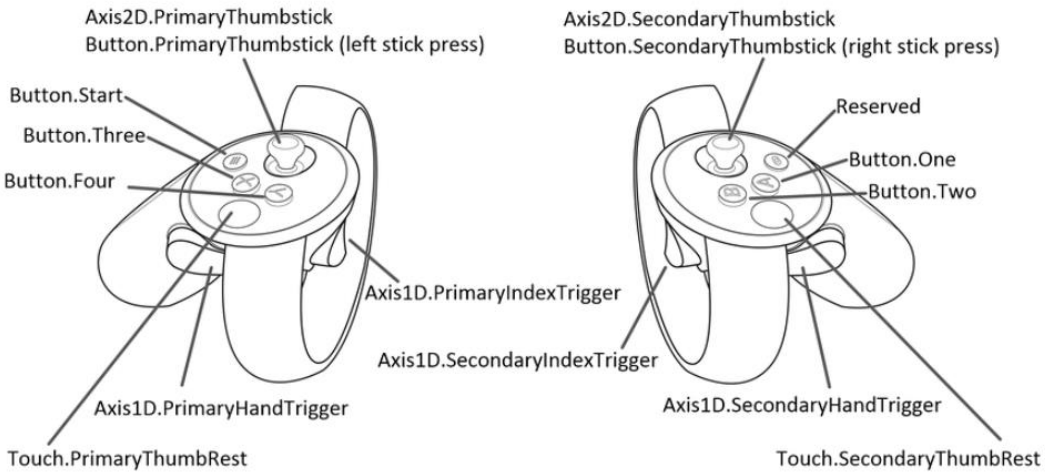
Elementos básicos del Oculus Touch Controllers

En cada mando contiene**:**

- 2 botones de acción
- 2 gatillos
- 1 stick de movimiento
- 1 botón de acceso al menú.

Figura 20

Elementos del Oculus Touch Controller



Nota. Detalle de los en la Oculus Touch Controller. Tomado de *Entrada para controladores OpenVR* [Imagen], por Unity Technologies, 2017, Unity Documentación. CC BY 2.0

El hardware de entrada de Oculus Rift es compatible con el editor nativo de Unity a través de la API Open VR es el mismo para HTC Vive y Oculus Touch Controller.

A continuación, en la Tabla 6 se muestra una comparativa entre los diferentes dispositivos de mando.

Tabla 6

Comparación Wii Remote vs Oculus Touch vs HTC Vive Controllers

Características	Wii Remote	Oculus Touch Controllers	HTC Vive Controllers
Soporte para comunicación	Hidapi (librería externa)	Soporte Windows	Soporte Windows

Características	Wii Remote	Oculus Touch Controllers	HTC Vive Controllers
Frecuencia de actualización	100Hz	72 Hz, 80 Hz o 90 Hz	250Hz
Uso	Media	Sencilla	Media
Comunicación	Bluetooth	Bluetooth	Bluetooth
Baterías	2 baterías AA	2 baterías AA	1 batería de litio
Autonomía Baterías	30 horas	20 – 30 horas	4 horas
Dispositivos	1 mando	2 mandos	1 o 2 mandos
Costo de mando	40 dólares	188 dólares	150 dólares (1 mando) 300 dólares (2 mandos)
Portabilidad	Baja	Media	Media
Área o distancia de operación	0.8m- 10m	3.3m x 1.5m	4.6m x 4.6m
Librerías públicas para Unity	Flafla2 Unity-WiiMote	Open RV (Unity)	Open RV (Unity)

De acuerdo con la Tabla 7 se vinculan todos los datos proporcionados por los desarrolladores de los dispositivos para luego presentar los parámetros de selección de mandos requeridos.

Tabla 7

Selección del mando para el motor de juego

Factor Relevante	Peso Asignado	Lugar					
		Wii Remote		Oculus Touch Controllers		HTC Vive Controllers	
		Calif	Pond	Calif	Pond	Calif	Pond
Compatibilidad con el motor de juego	0,25	9	2,25	10	2,5	10	2,5
Bajo Costo	0,25	10	2,5	8	2	7	1,75
Rapidez en Comunicación	0,2	8	1,6	7	1,4	9	1,8
Portabilidad en referencia a su Peso	0,1	9	0,9	7	0,7	8	0,8
Autonomía de Batería	0,1	9	0,9	7	0,7	8	0,8
Facilidad en uso	0,1	7	1,4	8	1,6	8	1,6
TOTAL	1		9,3		8,9		8,95
SELECCIÓN			SI		NO		NO

En referencia a lo ilustrado en la Tabla 7 la implementación a la propuesta se lo realiza con el mando Wii Remote para realidad virtual no inmersiva, el peso que conllevó al resultado fue el costo del dispositivo, la autonomía, su portabilidad, la rapidez de comunicación y al obtener el estudio de la compatibilidad con el motor de juego Unity, ayuda a los usuarios a llevar proyectos a ser mucho más económicos.

Capítulo IV

4. Modelación matemática y algoritmos de control

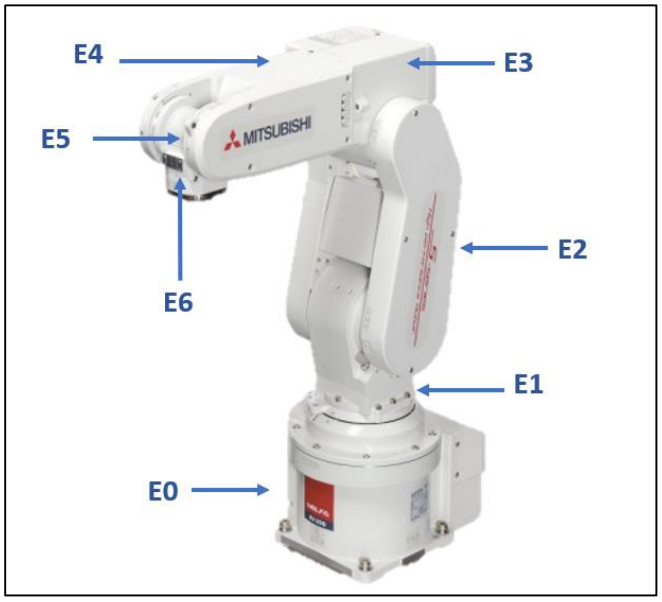
4.1 Obtención de los parámetros D-H del brazo robótico Mitsubishi RV – 2SDB

Para obtener los parámetros Denavit Hartenberg (D-H) se deben seguir una serie de pasos los cuales se muestran a continuación, es necesario recalcar que la Figura 2 mostrado en el capítulo 2 ya que se muestran las dimensiones del brazo robótico Mitsubishi, las mismas que son necesarias para el cálculo de los parámetros del robot.

1. Se procede a enumerar los eslabones del brazo robótico.

Figura 21

Enumeración de los eslabones del brazo robótico RV2SDB



2. Como siguiente paso se enumera las articulaciones.

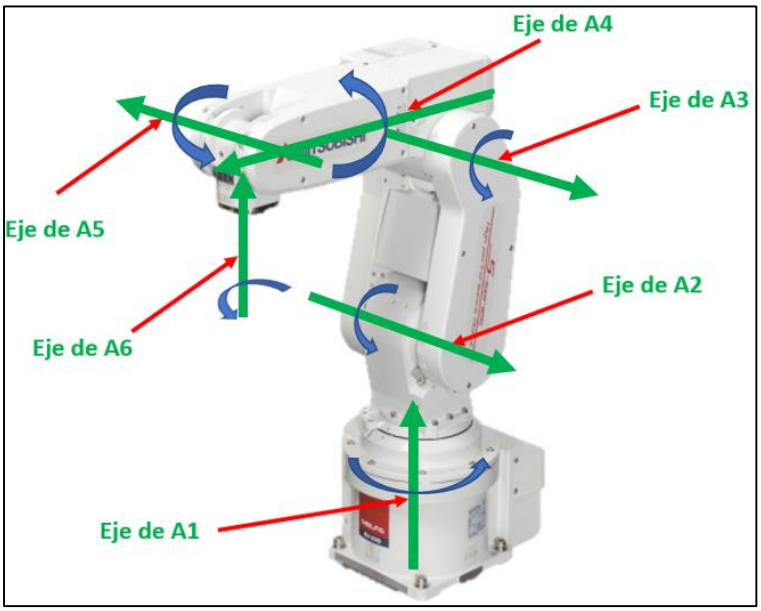
Figura 22

Enumeración de las articulaciones del brazo robótico RV2SDB

3. Localizar los ejes de cada articulación, es decir buscar el eje en el cual se está realizando el movimiento de giro o desplazamiento.

Figura 23

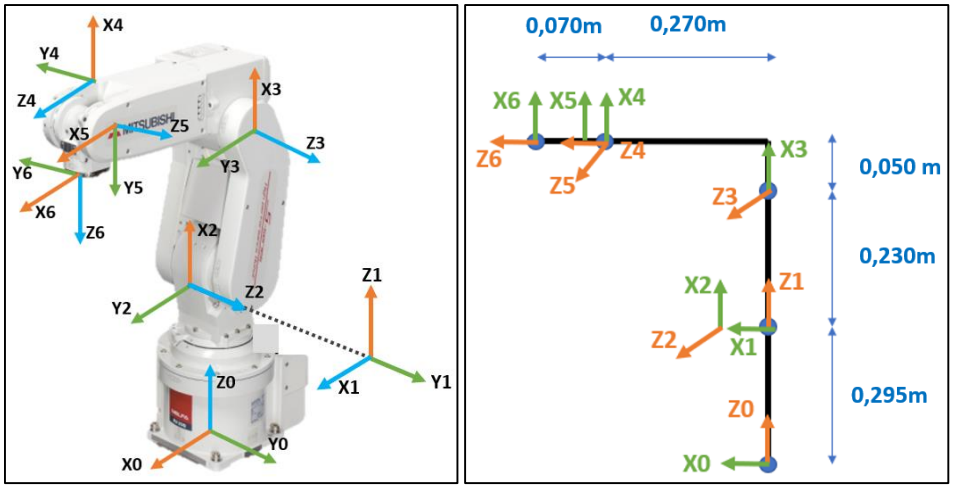
Localización de los ejes de cada articulación del manipulador



4. Ubicar los sistemas de cada articulación tomando en cuenta la ley de la mano derecha, también se hace un diagrama con sus respectivas dimensiones para poder determinar todos parámetros necesarios.

Figura 24

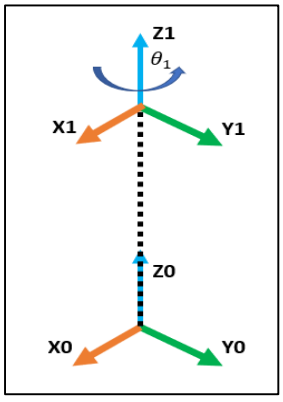
Ejes de cada articulación del brazo robótico RV2SDB



5. Se procede a armar la tabla con los parámetros DH según lo explicado en el capítulo 2, se procederá a hacer el análisis para las 3 primeras articulaciones:
   - Articulación 1

Figura 25

Parámetros gráficos DH de la primera articulación



θ Será θ_1 por la configuración inicial del robot, por tanto, no se necesita de un ángulo de rotación en el Z1 ya que X1 Y X0 se encuentran alineados.

a Será 0, ya que este representa la distancia del X0 al X1, como se encuentran alineados, no representa ningún desplazamiento en dicho eje.

d Será 0.295, ya que este representa la distancia del Z0 al Z1, como se vio en la Figura 2, tiene un desplazamiento de 0.295 m.

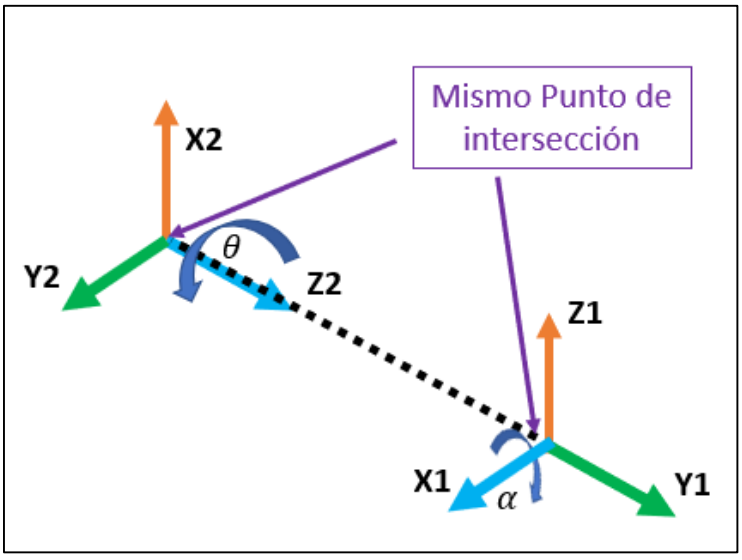
α Será 0, ya que representa el ángulo que puede rotar en el eje X0 para alinear Z1 y Z0, como se encuentran alineados el ángulo será 0°.

J	θ	d	a	α
1	θ_1	0.295	0	0

- Articulación 2

Figura 26

Parámetros gráficos DH de la segunda articulación



θ Será $\theta_2 - 90$,ya que es el ángulo que puede rotar en el eje Z2 para alinear X2 y X1, el ángulo de rotación es 90° y está en sentido antihorario.

a Será 0, ya que este representa la distancia del X2 al X1, como se encuentran en el mismo punto de intersección, no representa ningún desplazamiento en dicho eje.

d Será 0, ya que este representa la distancia del Z2 al Z1, como se vio en la Figura 2 se encuentran alineados, no representa ningún desplazamiento en dicho eje.

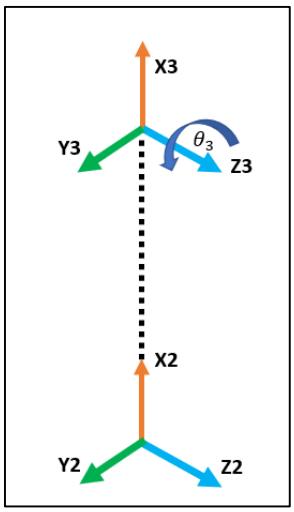
α Será -90, ya que representa el ángulo que puede rotar en el eje X1 para alinear Z1 y Z2, el ángulo de rotación es 90° y está en sentido antihorario.

J	θ	d	a	α
2	$\theta_2 - 90$	0	0	-90

- Articulación 3

Figura 27

Parámetros gráficos DH de la tercera articulación



θ Será θ_3 por la configuración inicial del robot, por tanto, no se necesita de un ángulo de rotación en el Z3 ya que X2 Y X3 se encuentran alineados.

a Será 0.230, ya que este representa la distancia del X2 al X3, como se vio en la Figura 2, tiene un desplazamiento de 0.230 m.

d Será 0, ya que este representa la distancia del Z0 al Z1, como se vio en la Figura 2 representa a la encuentran alineados, no representa ningún desplazamiento en dicho eje.

α Será 0, ya que representa el ángulo que puede rotar en el eje X2 para alinear Z3 y Z2, como se encuentran alineados el ángulo será 0°.

J	θ	d	a	α
3	θ_2	0	0.230	0

Se realiza el mismo procedimiento para las demás articulaciones obteniendo los siguientes parámetros DH mostrados en la Tabla 8

Tabla 8

Parámetros DH del Robot MELFA RV-2SDB

J	θ	d	a	α
1	θ_1	0.295	0	0
2	$\theta_2 - 90°$	0	0	-90
3	θ_3	0	0.230	0
4	θ_4	0.270	0.5	-90
5	$\theta_5 + 90$	0	0	90°
6	θ_6	0.070	0	-90

4.2 Cinemática directa del robot RV-2SDB modelamiento matemático

La cinemática directa se lo realiza a través de los parámetros Denavit Hartenberg del robot mediante la matriz de transformación homogénea (MTH) general expresada a continuación:

$$ {}^{n-1}A_n = \begin{pmatrix} cos\theta_n & -sin\theta_n cos\alpha_n & sin\theta_n sin\alpha_n & a_n cos\theta_n \\ sin\theta_n & cos\theta_n cos\alpha_n & -cos\theta_n sin\alpha_n & a_n sin\theta_n \\ 0 & sin\alpha_n & cos\alpha_n & d_n \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (5) $$

Se procede a reemplazar valores para cada juntura analizada en la Tabla 8.

$$ {}^{0}A_1 = \begin{pmatrix} cos\theta_1 & -sin\theta_1 \cos(0) & sin\theta_1 sin(0) & (0)cos\theta_1 \\ sin\theta_1 & cos\theta_1 cos(0) & -cos\theta_1 sin(0) & (0)sin\theta_1 \\ 0 & sin(0) & cos(0) & 0.295 \\ 0 & 0 & 0 & 1 \end{pmatrix} $$

$$ {}^{0}A_1 = \begin{pmatrix} cos\theta_1 & -sin\theta_1 & 0 & 0 \\ sin\theta_1 & cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0.295 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (6) $$

$$ {}^{1}A_2 $$

$$ = \begin{pmatrix} cos(\theta_2 - 90) & -sin(\theta_2 - 90)\cos(-90) & sin(\theta_2 - 90)sin(-90) & (0)cos(\theta_2 - 90) \\ sin(\theta_2 - 90) & cos(\theta_2 - 90)cos(-90) & -cos(\theta_2 - 90)\sin(-90) & (0)sin(\theta_2 - 90) \\ 0 & \sin(-90) & \cos(-90) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} $$

$$ {}^{1}A_2 = \begin{pmatrix} sin\theta_2 & cos\theta_2 & 0 & 0 \\ -cos\theta_2 & sin\theta_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (7) $$

$$ {}^{2}A_3 = \begin{pmatrix} cos\theta_3 & -sin\theta_3 cos(-90) & sin\theta_3 \, sin(-90) & (0.050)cos\theta_3 \\ sin\theta_3 & cos\theta_3 cos(-90) & -cos\theta_3 sin(-90) & (0.050)sin\theta_3 \\ 0 & sin(-90) & cos(-90) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} $$

$$ {}^{2}A_{3} = \begin{pmatrix} cos\theta_3 & 0 & -sin\theta_3 & (0.050)cos\theta_3 \\ sin\theta_3 & 0 & cos\theta_3 & (0.050)sin\theta_3 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (8) $$

$$ {}^{3}A_{4} = \begin{pmatrix} cos\theta_4 & -sin\theta_4\cos(90) & sin\theta_4 sin(90) & (0)cos\theta_4 \\ sin\theta_4 & cos\theta_4 cos(90) & -cos\theta_4 sin(90) & (0)sin\theta_4 \\ 0 & sin(90) & cos(90) & 0.270 \\ 0 & 0 & 0 & 1 \end{pmatrix} $$

$$ {}^{3}A_{4} = \begin{pmatrix} cos\theta_4 & 0 & sin\theta_4 & 0 \\ sin\theta_4 & 0 & -cos\theta_4 & 0 \\ 0 & 1 & 0 & 0.270 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (9) $$

$$ {}^{4}A_{5} = \begin{pmatrix} cos\theta_5 & -sin\theta_5\cos(-90) & sin\theta_5 sin(-90) & (0)cos\theta_5 \\ sin\theta_5 & cos\theta_5 cos(-90) & -cos\theta_5 sin(-90) & (0)sin\theta_5 \\ 0 & sin(-90) & cos(-90) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} $$

$$ {}^{4}A_{5} = \begin{pmatrix} cos\theta_5 & 0 & -sin\theta_5 & 0 \\ sin\theta_5 & 0 & cos\theta_5 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (10) $$

$$ {}^{5}A_{6} = \begin{pmatrix} cos\theta_6 & -sin\theta_6\cos(0) & sin\theta_6 sin(0) & (0)cos\theta_6 \\ sin\theta_6 & cos\theta_6 cos(0) & -cos\theta_6 sin(0) & (0)sin\theta_6 \\ 0 & sin(0) & cos(0) & 0.070 \\ 0 & 0 & 0 & 1 \end{pmatrix} $$

$$ {}^{5}A_{6} = \begin{pmatrix} cos\theta_6 & -sin\theta_6 & 0 & 0 \\ sin\theta_6 & cos\theta_6 & 0 & 0 \\ 0 & 0 & 1 & 0.070 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (11) $$

La siguiente matriz T resultante es el producto entre (6),(7),(8),(9),(10),(11):

$$ T = {}^{0}A_{6} = {}^{0}A_{1} * {}^{1}A_{2} * {}^{2}A_{3} * {}^{3}A_{4} * {}^{4}A_{5} * {}^{5}A_{6} \quad (12) $$

En la matriz dichos valores son representados para X, Y y Z los valores de (p_x, p_y, p_z) respectivamente, y la submatriz de 3*3 conformada por los

elementos n, o, a representan la matriz de rotación final. Definida como el vector de aproximación $\mathbf{a} = (a_x, a_y, a_z)$, un vector perpendicular $\mathbf{o} = (o_x, o_y, o_z)$,con referencia a la pinza y el vector de normal $\mathbf{n} = (n_x, n_y, n_z)$.

$$ {}^{0}_{6}A = \begin{pmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{13} $$

Dónde:

$$ \begin{aligned} X = {} & \frac{23 * \cos(\theta_1)\sin(\theta_2)}{100} - \frac{7\sin(\theta_5)\left[\sin(\theta_1)\sin(\theta_4) + \cos(\theta_4)(\sigma_2 + \sigma_1)\right]}{100} \\ & - \frac{7\cos(\theta_5)\left[\cos(\theta_1)\sin(\theta_2)\sin(\theta_3) - \cos(\theta_1)\cos(\theta_2)\cos(\theta_3)\right]}{100} \\ & - \frac{7\cos(\theta_1)\cos(\theta_2)\cos(\theta_3)}{100} + \frac{\sigma_2}{20} + \frac{\sigma_1}{20} \end{aligned} \tag{14} $$

Siendo

$$ \sigma_1 = \cos(\theta_1)\cos(\theta_3)\sin(\theta_2) $$

$$ \sigma_2 = \cos(\theta_1)\cos(\theta_2)\sin(\theta_3) $$

$$ \begin{aligned} Y = {} & \frac{23 * \sin(\theta_1)\sin(\theta_2)}{100} + \frac{7\sin(\theta_5)\left[\cos(\theta_1)\sin(\theta_4) + \cos(\theta_4)(\sigma_2 + \sigma_1)\right]}{100} \\ & - \frac{7\cos(\theta_5)\left[\sin(\theta_1)\sin(\theta_2)\sin(\theta_3) - \cos(\theta_2)\cos(\theta_3)\sin(\theta_1)\right]}{100} \\ & - \frac{7\sin(\theta_1)\sin(\theta_2)\sin(\theta_3)}{100} + \frac{\sigma_3}{20} + \frac{\sigma_4}{20} + \frac{7\cos(\theta_2)\cos(\theta_3)\sin(\theta_1)}{100} \end{aligned} \tag{15} $$

Siendo

$$ \sigma_3 = \cos(\theta_3)\sin(\theta_1)\sin(\theta_2) $$

$$ \sigma_4 = \cos(\theta_2)\sin(\theta_1)\sin(\theta_3) $$

$$Z = \frac{23 * \cos(\theta_2)}{100} + \frac{\sigma_5}{20} - \frac{7\cos(\theta_2)\sin(\theta_3)}{100} - \frac{7\cos(\theta_3)\sin(\theta_2)}{100} - \frac{\sigma_6}{20} - \frac{7\cos(\theta_5)\left[\cos(\theta_2)\sin(\theta_3) + \cos(\theta_3)\sin(\theta_2)\right]}{100} - \frac{7\cos(\theta_4)\sin(\theta_5)(\sigma_6 - \sigma_5)}{100} + \frac{59}{200} \quad (16)$$

Siendo

$$\sigma_5 = \sin(\theta_2)\sin(\theta_3)$$

$$\sigma_6 = \cos(\theta_2)\cos(\theta_3)$$

4.2.1 Matriz de rotación

Los ángulos de rotación de Euler Yaw Ø, Pitch θ y Roll ψ (guiñada, cabeceo y alabeo) son de tipo cardán ya que, al ser descritos por la localización del extremo del brazo robótico genera una sucesión rotativa en sus ejes $\boldsymbol{xyz}$ respectivamente, esta matriz se obtuvo de la matriz homogénea de la ecuación (12).

$$Rxyz = \begin{bmatrix} n_x & o_x & a_x \\ n_y & o_y & a_y \\ n_z & o_z & a_z \end{bmatrix}$$

Dada la ecuación

$$Rxyz = Rx(\text{Ø})Ry(\theta)Rz(\psi) \quad (17)$$

Se obtiene la matriz de rotación correspondiente de cada eje

$$Rxyz = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\text{Ø}) & -sin(\text{Ø}) \\ 0 & sin(\text{Ø}) & cos(\text{Ø}) \end{bmatrix} \begin{bmatrix} cos(\theta) & 0 & sin(\theta) \\ 0 & 1 & 0) \\ -sin(\theta) & 0 & cos(\theta) \end{bmatrix} \begin{bmatrix} cos(\psi) & -sin(\psi) & 0 \\ sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (18)$$

Siendo que el producto de la matriz de rotación de (18).

$$Rxyz = \begin{bmatrix} \cos(\theta)cos(\psi) & -\cos(\theta)sin(\psi) & \sin(\theta) \\ \cos(\text{Ø})sin(\psi) + \sin(\text{Ø})\sin(\theta)cos(\psi) & \cos(\text{Ø})cos(\psi) - \sin(\theta)\sin(\text{Ø})sin(\psi) & -\cos(\theta)sin(\text{Ø}) \\ \sin(\text{Ø})sin(\psi) - \cos(\text{Ø})cos(\psi)sin(\theta) & cos(\psi)sin(\text{Ø}) + \cos(\text{Ø})\sin(\theta)sin(\psi) & cos(\text{Ø})\cos(\theta) \end{bmatrix}$$

Al trabajar con ángulos de Euler XYZ se definen las ecuaciones para la rotación $(Yaw\ \emptyset, Pitch\ \ \theta\ y\ Roll\ \psi)$

$$Rxyz(\emptyset, \theta, \psi) = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} atan2(-a_y, a_z) \\ atan2(a_x, \sqrt{{n_x}^2 + {o_x}^2} \\ atan2(-o_x, n_x) \end{bmatrix} \quad (19)$$

4.2.2 Comprobación de los Parámetros DH y la MTH

Dados los parámetros D-H de la 9 y la ecuación (5) de la matriz de transformación homogénea general. Se ingresa los datos en Matlab para obtener la MTH del brazo robótico Mitsubishi Melfa RV – 2SDB, obteniendo como respuesta las ecuaciones (12),(14),(15),(16) y (19).

Figura 28

Matriz de transformación en Matlab

```
L1=295; L2=230; L3=50; L4=270; L6=70; L5=0;
%%  Matriz General
syms th d a alpha
A= [ cos(th) -cos(alpha)*sin(th) sin(alpha)*sin(th) a*cos(th),
     sin(th) cos(alpha)*cos(th) -sin(alpha)*cos(th) a*sin(th),
     0       sin(alpha)          cos(alpha)             d     ,
     0        0                  0                  1  ]
%% Link1
syms th1 L1
A1=subs(A,{th, d, a, alpha},{th1,L1,0,-pi/2});

%% Link2
syms th2 L2
A2=subs(A,{th, d, a, alpha},{th2-pi/2, 0, L2, 0});
%% Link3
syms th3 L3
A3=subs(A,{th, d, a, alpha},{th3, 0, L3, -pi/2});
%% Link2
syms th4  L4
A4=subs(A,{th, d, a, alpha},{th4, L4, 0, pi/2});
%% Link5
syms th5
A5=subs(A,{th, d, a, alpha},{th5, 0, 0, -pi/2});
%% Link
syms th6  L6
A6=subs(A,{th, d, a, alpha},{th6, L6, 0, 0});
%% Matrices
T1=simplify(A1);
T2=simplify(A1*A2);
T3=simplify(A1*A2*A3);
T4=simplify(A1*A2*A3*A4);
T5=simplify(A1*A2*A3*A4*A5);
%% Matriz de Tranformación Homogénea
T=simplify(A1*A2*A3*A4*A5*A6);
```

Nota. Obtención de la MTH del brazo robótico mediante programación en Matlab.

Insertar los parámetros obtenidos de la 9 en la librería de Peter Corke llamada Robotics Toolbox.

Figura 29

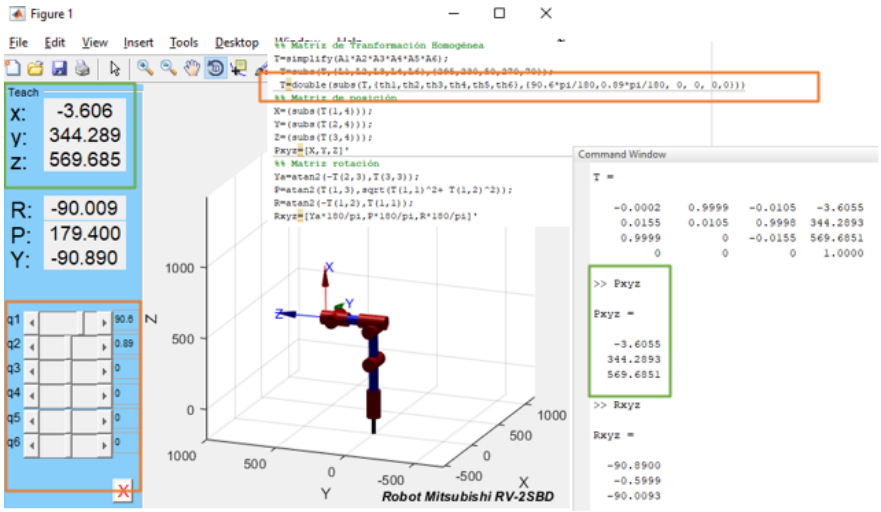
Parámetros D-H en librería Robotics Toolbox

```
%Librería Robotics Toolbox Peter Corke
L1=295; L2=230; L4=270; L6=70;L3=50;
% CREATE LINK
L(1)=Link([0 L1 0 -pi/2]);
L(2)=Link([0-pi/2 0 L2 0]);
L(3)=Link([0 0 L3 -pi/2]);
L(4)=Link([0 L4 0 pi/2]);
L(5)=Link([0 0 0 -pi/2]);
L(6)=Link([0 L6 0 0]);
Robot=SerialLink(L,'name','Robot Mitsubishi RV-2SBD', 'offset', [0 -pi/2 0 0 0 0 ])
```

Para verificar si los datos de posición y orientación de Robotics Toolbox, la MTH propuesta y el brazo Mitsubishi Melfa RV – 2SDB son los mismos, para ello se establece un valor definido en sus articulaciones.

Figura 30

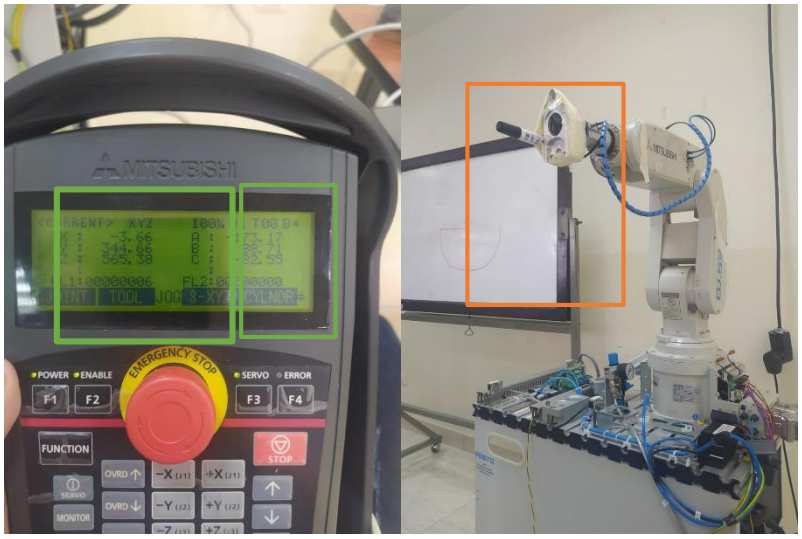
Ejemplo 1. Comprobación con Peter Corke y matriz resultante (MTH)



Nota. Los datos en posición en XYZ son (-3.6, 344.28, 569.6), se verifica que los resultados obtenidos son los mismo con la librería de Peter Corke.

Figura 31

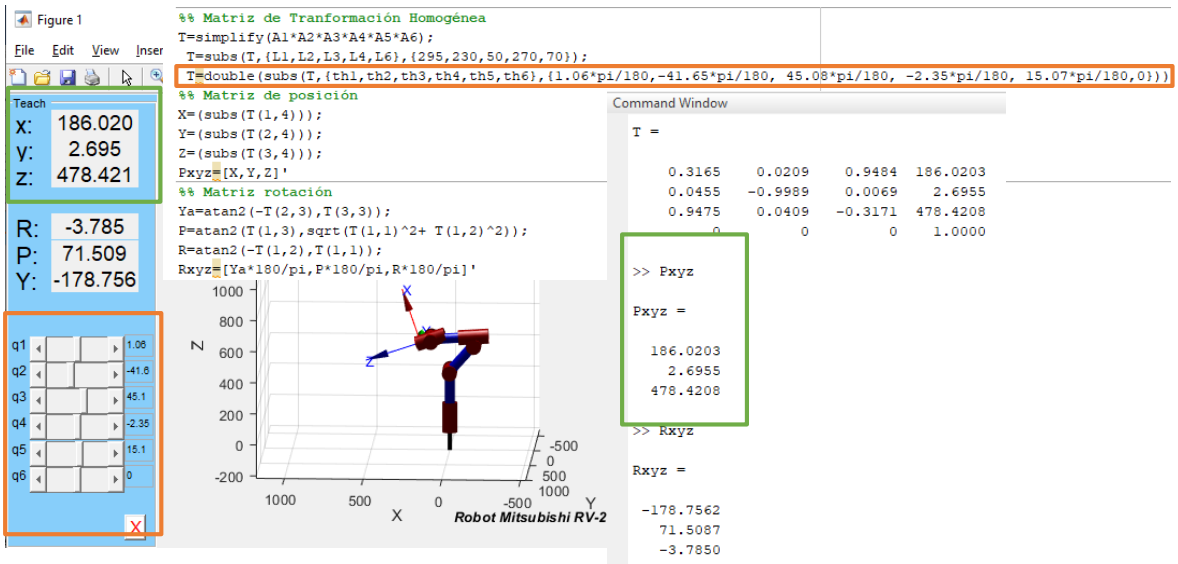
Ejemplo 1. Comprobación del T/B del brazo Mitsubishi RV – 2SDB



Nota. Los datos en posición del manipulador en XYZ son (-3.66, 344.28, 565.3).

Figura 32

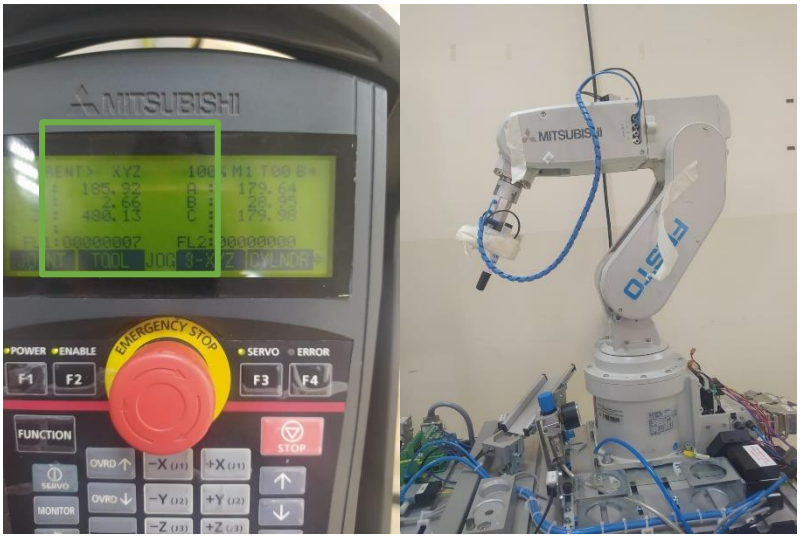
Ejemplo 2. Comprobación con Peter Corke y matriz resultante



Nota. Los datos en posición en XYZ son (186.0, 2.6, 478.4), se verifica que los resultados obtenidos son los mismo con la librería de Peter Corke

Figura 33

Ejemplo 2. Comprobación del T/B del brazo Mitsubishi RV – 2SDB



Nota. Los datos en posición del manipulador en XYZ son (185.9, 2.6, 480).

4.3 Cinemática inversa: a partir de la matriz Jacobiana inversa

La cinemática inversa se puede solucionar con los métodos presentados en la sección 2.6.2 a medida que aumenta el número de grados de libertad el cálculo es más avanzado, es por ello que se ocupa el modelo diferencial de la jacobiana que relaciona las velocidades articulares $\dot{q} = (\dot{q}_1, \dot{q}_2, \dot{q}_3, \dot{q}_4, \dot{q}_5, \dot{q}_6)$ con respecto a las velocidades de localización del extremo del robot en posición $\dot{h} = (\dot{x}, \dot{y}, \dot{z})$. El modelo cinemático del brazo robótico está representado en la ecuación.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = J(q) \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \\ \dot{q}_5 \\ \dot{q}_6 \end{bmatrix} \tag{20}$$

$$\dot{h} = J(q)\dot{q} \tag{21}$$

El modelo cinemático define velocidades angulares dada los valores de posición del efector final en la siguiente ecuación.

$$\dot{q} = J(q)^{-1}\dot{h} \tag{22}$$

Donde J es la matriz Jacobiana analítica que depende de las derivadas de posición XYZ de las ecuaciones (14),(15),(16).

$$J(q) = \begin{bmatrix} \frac{\partial x}{\partial q_1} & \frac{\partial x}{\partial q_2} & \frac{\partial x}{\partial q_3} & \frac{\partial x}{\partial q_4} & \frac{\partial x}{\partial q_5} & \frac{\partial x}{\partial q_6} \\ \frac{\partial y}{\partial q_1} & \frac{\partial y}{\partial q_2} & \frac{\partial y}{\partial q_3} & \frac{\partial y}{\partial q_4} & \frac{\partial y}{\partial q_5} & \frac{\partial y}{\partial q_6} \\ \frac{\partial z}{\partial q_1} & \frac{\partial z}{\partial q_2} & \frac{\partial z}{\partial q_3} & \frac{\partial z}{\partial q_4} & \frac{\partial z}{\partial q_5} & \frac{\partial z}{\partial q_6} \end{bmatrix}_{(3x6)} \tag{23}$$

Sabiendo que $J(q)$ no es una matriz cuadrada no se puede definir una matriz inversa debido a que el sistema es redundante donde posee más columnas que filas, obteniendo varias soluciones, por ende, es necesario definir la matriz como pseudoinversa “por la derecha”, viene dado por la ecuación.

$$J(q)^{+} = J(q)^{T}_{(6x3)} * (J(q)_{(3x6)} * J(q)^{T}_{(6x3)})^{-1} \tag{24}$$

Cumpliendo la propiedad

$$J(q) * J(q)^{+} = I$$

Además, al generar un sistema en lazo cerrado es necesario describir la acumulación del error descrita en la ecuación (25) al producir un movimiento no deseado, donde se añade el control a la ecuación

(22) la diferencia entre los valores de posición deseada y la real.

$$h_e = h_d - h_r \tag{25}$$

$$\dot{q} = J(q)^{+}(\dot{h} + K(h_e)) \tag{26}$$

Siendo

- h_e Matriz de errores
- h_d Matriz posiciones deseadas
- h_r Matriz posiciones reales
- $\dot{h} = 0$ Derivada de la posición del efector final (valor entero)
- J^{+} Jacobiana pseudoinversa
- K Matriz de ganancias

Ley de control cinemático de posición en lazo cerrado.

$$\dot{q}_{(k)} = J(q_{(k)})^{+}\, K * h_e \tag{27}$$

Para obtener la posición de las articulaciones se realiza el método de integración rectangular a partir de la ecuación

$$q_{(k+1)} = q_{(k)} + \dot{q}_{(k)} dt \tag{28}$$

Donde

- dt : intervalo de muestra (delta time Unity $0.25\ seg$. Apróx.)
- $q_{(k+1)}$: ángulo deseado de la articulación
- $q_{(k)}$: ángulo actual de la articulación

Con la ecuación (2829), se puede generar la cinemática inversa del robot como la generación trayectorias en el simulador.

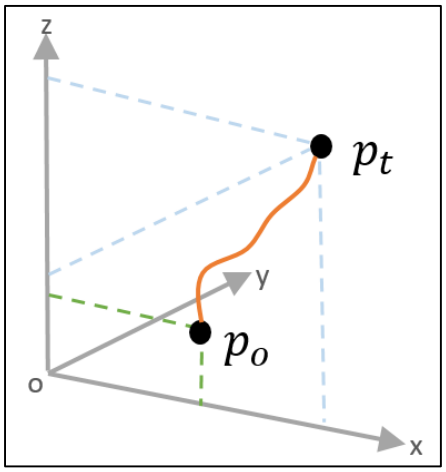
4.4 Trayectorias

4.4.1 Punto a punto

Se genera un movimiento simultáneo en sus articulaciones, sin importar la trayectoria desde p_o a p_t.

Figura 34

Trayectoria punto a punto



- p_o Punto inicial
- p_t Punto final

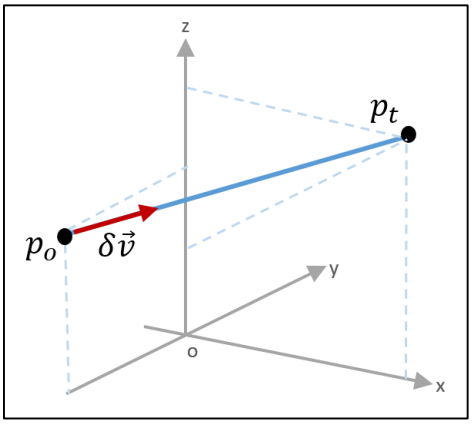
Esta trayectoria se incorpora en la ecuación la ecuación (28)

4.4.2 Línea

Esta trayectoria continua, genera movimientos simultáneos en las juntas en un instante dt, la trayectoria lineal se determina por la recta que cruza en el espacio con punto y dirección $p = (p_o, \vec{v})$.

Figura 35

Trayectoria lineal



Dada la ecuación vectorial de la recta.

$$(x, y, z) = \delta(v_1, v_2, v_3) + (x_o, y_o, z_o)$$

$$p = \delta\vec{v} + p_o \tag{2930}$$

Donde $\vec{v} = \frac{p_t - p_o}{\|p_t - p_o\|_2}$

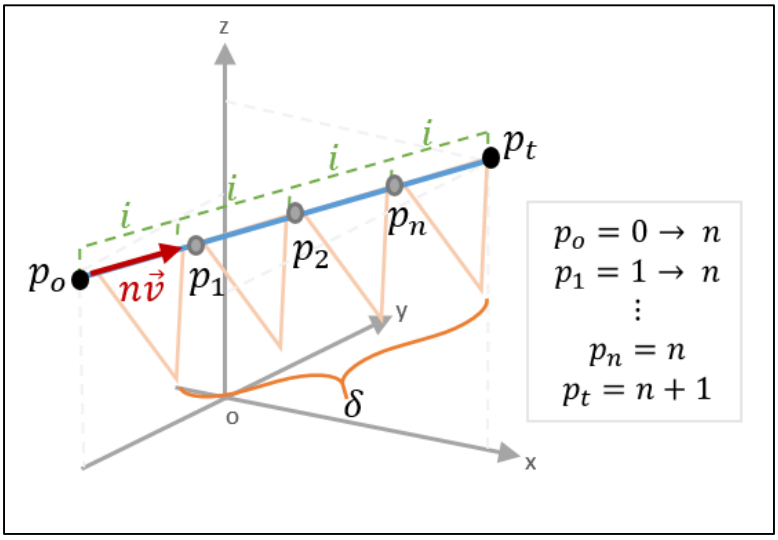
- $\vec{v}$ Vector director
- p_o Punto inicial que pasa la recta
- p_t Punto final que pasa la recta
- $\|p_t - p_o\|_2$ Distancia euclídea
- δ Avance lineal es un valor real que determina cada coordenada de un punto cualquiera $p(xyz)$ de la recta, en este caso se lo conocerá como cada iteración n de la subdivisión.

Sabiendo que para poder completar la trayectoria lineal de p_o a p_t se genera n puntos, estos puntos se crean a partir de las subdivisiones de la recta. Modificando δ con un valor determinado de la ecuación (2930), si su avance llega a muy grande es

necesario determinar una relación eficiente entre movimientos cortos y poca la carga computacional, que para el cálculo se tomó un con un avance de $i = 40mm$ constante en la ecuación (3031)

Figura 36

Trayectoria lineal modificada



Las subdivisiones vienen dadas por

$$\delta = \frac{\vec{v}}{i} \tag{3031}$$

$$\therefore n = 0,1,2,\dots,\delta$$

Donde n es la iteración desde 0 hasta la subdivisión δ y también representa cada punto generado aparte de (p_o, p_t)

Predeterminando 40 mm como una relación eficiente entre movimientos cortos y sin mucha carga computacional para el cálculo

Dando como resultado la siguiente ecuación

$$p_{(n)} = i * n * \vec{v} + p_o \tag{3132}$$

De la ecuación (3132) se genera un vector de puntos para incorporar en la ecuación (2833)

$$p_{(n)} = \begin{bmatrix} p_o \\ p_1 \\ \vdots \\ p_n \\ p_t \end{bmatrix}$$

4.4.3 Arco círculo

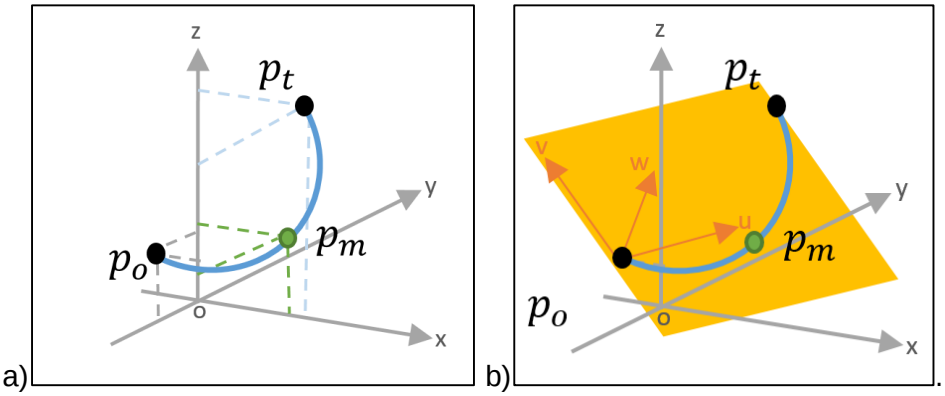
Para obtener los puntos proyectados de una trayectoria circular en el espacio 3D dados 3 puntos: punto inicial, punto medio, punto final (p_o, p_m,p_t), es necesario determinar las ecuaciones que ayuden al cálculo de los parámetros de un círculo. En un sistema 3D las ecuaciones que se anunciarán no pueden intervenir en esta referencia 3D debido a que la posición de sus puntos busca encontrar un sistema de referencia 2D. Unity con sus capacidades de jerarquía comparte la configuración de sus transformaciones (posición y rotación) de cada objeto (punto) asignado y posee herramientas para el cálculo como la función plane (plano 2D). El procedimiento es el siguiente

Se crea un plano 2D(XY) que incluya los 3 puntos, se obtiene la referencia a la normal del plano y se guarda la orientación inicial del sistema UVW (padre).

Se procede a modificar las transformaciones (p_o, p_m,p_t), en un plano XY y donde p_o, el vector vertical z se orienta en la misma dirección de la norma del plano

Figura 37

Trayectoria circular

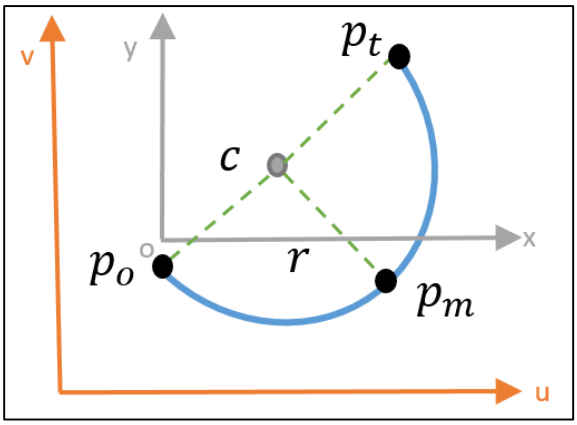


Nota. a) Trayectoria circular de 3 puntos. b) Plano 2D incluye los 3 puntos y a su vez la orientación inicial de (p_o, p_m,p_t) que se lo guarda en el sistema UVW.

Dado que el sistema se encuentra en un plano 2D se determinar las ecuaciones en forma de matrices para obtener el centro y el radio del círculo, deshaciendo la componente z.

Figura 38

Parámetros del círculo



Nota. Plano 2D XY, que contiene los 3 puntos para el cálculo de los parámetros del círculo

$$\begin{bmatrix} p_o \\ p_m \\ p_t \end{bmatrix} = \begin{bmatrix} x_o & y_o & 1 \\ x_m & y_m & 1 \\ x_t & y_t & 1 \end{bmatrix}$$

Dada la ecuación general del círculo

$$x^2 + y^2 + Ax + By + C = 0 \tag{3234}$$

$$Ax + By + C = -(x^2 + y^2) \tag{3335}$$

Dados tres puntos (p_o, p_m,p_t) se calcula la ecuación de circunferencia, mediante un sistema lineal en las siguientes matrices

$$\begin{bmatrix} x_o & y_o & 1 \\ x_m & y_m & 1 \\ x_t & y_t & 1 \end{bmatrix} \begin{bmatrix} A \\ B \\ \mathrm{C} \end{bmatrix} = \begin{bmatrix} -(x_o^2 + y_o^2) \\ -(x_m^2 + y_m^2) \\ -(x_t^2 + y_t^2) \end{bmatrix} \tag{3436}$$

Despejando las incógnitas A, B, C.

$$\begin{bmatrix} A \\ B \\ \mathrm{C} \end{bmatrix} = \begin{bmatrix} x_o & y_o & 1 \\ x_m & y_m & 1 \\ x_t & y_t & 1 \end{bmatrix}^{-1} \begin{bmatrix} -(x_o^2 + y_o^2) \\ -(x_m^2 + y_m^2) \\ -(x_t^2 + y_t^2) \end{bmatrix} \tag{3537}$$

Se desea los parámetros del círculo, se parte de la ecuación canónica.

$$(x - h)^2 + (y - k)^2 = r^2 \tag{3638}$$

Desagrupando los términos de la ecuación (3638) e igualando con la ecuación (3234)

$$x^2 - 2hx + h^2 + y^2 - 2ky + k^2 - r^2 = x^2 + y^2 + Ax + By + C \tag{3739}$$

Se tiene las siguientes igualdades.

$$\begin{bmatrix} A \\ B \\ \mathrm{C} \end{bmatrix} = \begin{bmatrix} -2\mathrm{h} \\ -2\mathrm{k} \\ h^2 + k^2 - r^2 \end{bmatrix} \tag{3840}$$

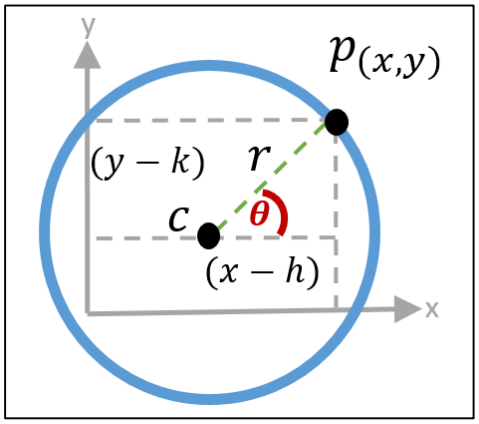
Los parámetros del centro y radio del círculo son:

$$h = -\frac{A}{2} \qquad k = -\frac{B}{2} \quad r = \sqrt{h^2 + k^2 - C} \tag{3941}$$

Para obtener los *n* puntos de la trayectoria se identifica por la ecuación paramétrica del círculo.

Figura 39

Detalle de la ecuación paramétrica del círculo



$$x = h + r * \cos(\theta) \tag{4042}$$

$$y = k + r * \text{sen}(\theta) \tag{4143}$$

$$\theta \ \in [0, 2\pi]$$

Siendo θ el ángulo del punto a través de la circunferencia con respecto al eje x.

Cada punto será parte de las subdivisiones (δ) de la trayectoria para determinar la relación eficiente entre movimientos cortos y poca carga computacional, el ángulo determinado es un avance de $i = 15\,mm$ constante en cada iteración (n puntos) hasta p_t.

Las subdivisiones vienen dadas por un valor entero:

$$\delta = \frac{2\Pi r}{i} \tag{4244}$$

$$\delta \in \mathbb{N}$$

$$\therefore n = 0,1,2, \dots, \delta \tag{4345}$$

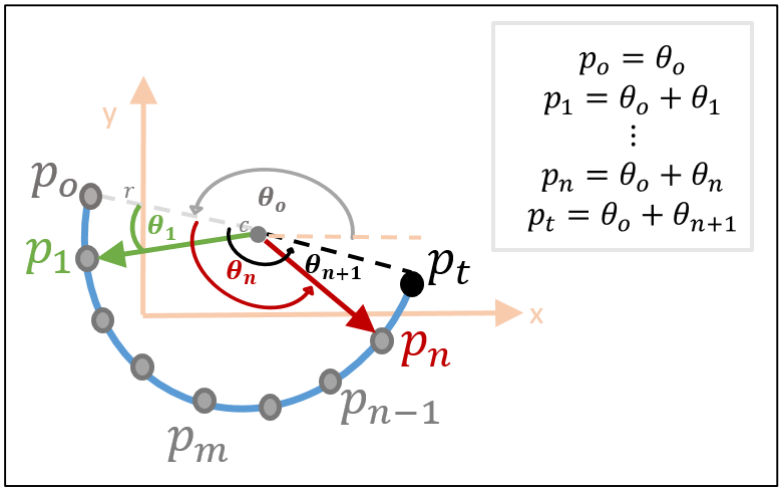
n es cada iteración desde 0 hasta la subdivisión δ, entonces cada punto se genera por un ángulo en cada iteración aparte de (p_o, p_t).

Las iteraciones ayudan a generar un ángulo para un nuevo punto, como se muestra en la ecuación (4446) y la Figura 40.

$$\theta_n = \left(\frac{i}{r}\right) * n \tag{4446}$$

Figura 40

Generación de puntos en la trayectoria circular



Nota. La trayectoria se genera a partir de las divisiones de acuerdo al ángulo θ_n.

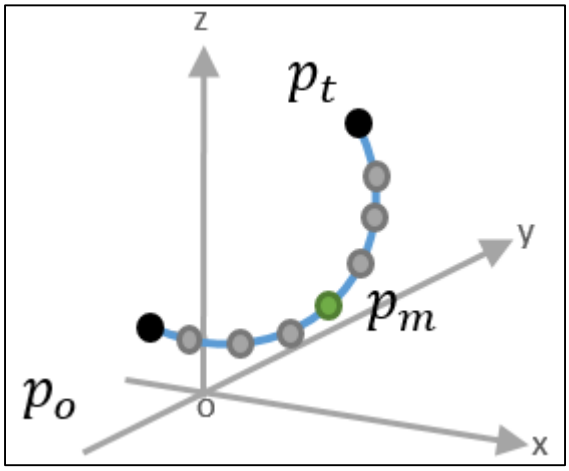
Dando como resultado la siguiente ecuación

$$p_{(n)} = (h + r * \cos(\theta_o + \theta_n), k + r * \text{sen}(\theta_o + \theta_n)) \tag{4547}$$

Finalmente, el vector de puntos generados $p_{(n)}$ y los puntos (p_o, p_m,p_t) que se encuentran en la dirección a la normal del plano XY vuelven al estado original de orientación en el plano de 3D XYZ al multiplicar por sistema padre UVW .

Figura 41

Trayectoria circular modificada



Nota. La trayectoria retoma las coordenadas iniciales en sistema 3D.

El vector de puntos generados de la ecuación (4547)(4547) se incorpora como una trayectoria en la ecuación (2848).

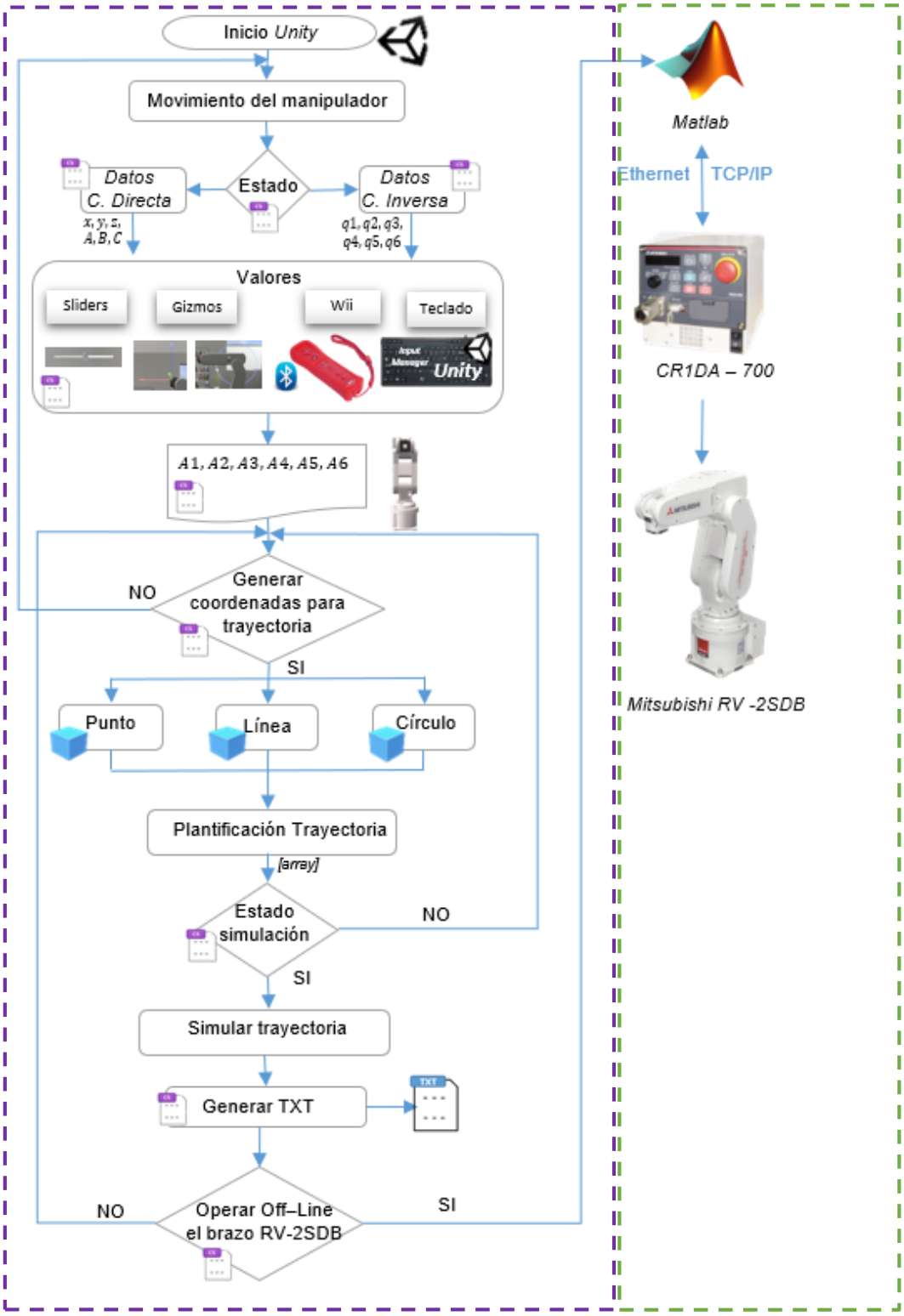
Capítulo V

5. Implementación del sistema off-line a través de realidad virtual no inmersiva

Para la implementación del sistema de operación Off – line se identifican dos etapas principales, la primera es la implementación del sistema de realidad virtual no inmersivo y la segunda la comunicación con la controladora del brazo RV - 2SDB para que pueda ejecutar los movimientos, en este capítulo se desarrolla las dos etapas contemplando todo el sistema, en la Figura 40 se representa el funcionamiento general.

Figura 42

Flujograma del sistema de operación off-line



Nota. Procedimiento del sistema de operación off – line a través de realidad virtual, se indica el proceso desde que el usuario usa el simulador hasta que pueda ejecutar dichas trayectorias en el brazo físico.

5.1 Escenas y ventanas de realidad virtual no inmersiva

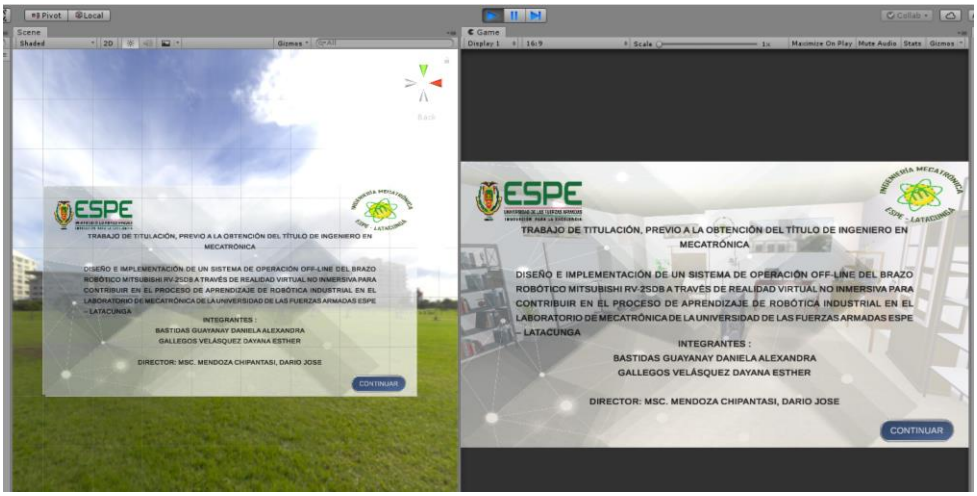
5.1.1 Creación del inicio

- **Presentación**

La presentación donde se define el título del proyecto y el botón de “Continuar” direcciona a la pantalla de menú.

Figura 43

Pantalla de Inicio



- **Menú**

En el menú de inicio se encuentran 3 botones que permiten ir a las siguientes opciones.

Figura 44

Pantalla de menú de opciones

En esta Pantalla se presentan 4 botones a las siguientes opciones.

1. “Iniciar” se dirige a la escena donde se encuentra el Robot Manipulador y el ambiente virtual.
2. “Instrucciones” es una ventana emergente que despliega todas las indicaciones del funcionamiento del robot (movimiento) y de cada elemento del HMI incorporado en la pantalla de juego. En la sección 5.1.6 se detallará esta opción.
3. “Salir” direcciona el fin del Juego y se activa la función “Quit” de Unity.
4. “Clase” muestra un video educativo donde se explica conceptos generales de robótica industrial.

- **Clase robótica**

En esta pantalla inicia una presentación de nociones básicas de robótica y con los conocimientos adquiridos puedan desarrollar el simulador.

Figura 45

Pantalla clase de robótica

Nota. Presentación de conocimientos básicos en robótica el cual se basó para poder realizar el simulador y ver el funcionamiento del mismo

5.1.2 Creación del ambiente virtual

5.1.2.1 Aula virtual

Para la escena principal se crea un laboratorio virtual a partir de un modelo 3D con el software SketchUp. A continuación, en la imagen se muestra el entorno realizado.

Figura 46

Aula virtual en SketchUp 2020

SketchUp permite crear ambientes arquitectónicos, soporta aristas y elementos poligonales permite exportar estas entidades de objeto 3D, para que Unity pueda reconocerlo.

Figura 47

Exportar modelo 3D de SketchUp 2020

Nota. Cargar archivo como modelo 3D, como se mencionó en la sección 3.2.2 debido a que Unity soporta archivos .fbx u obj.

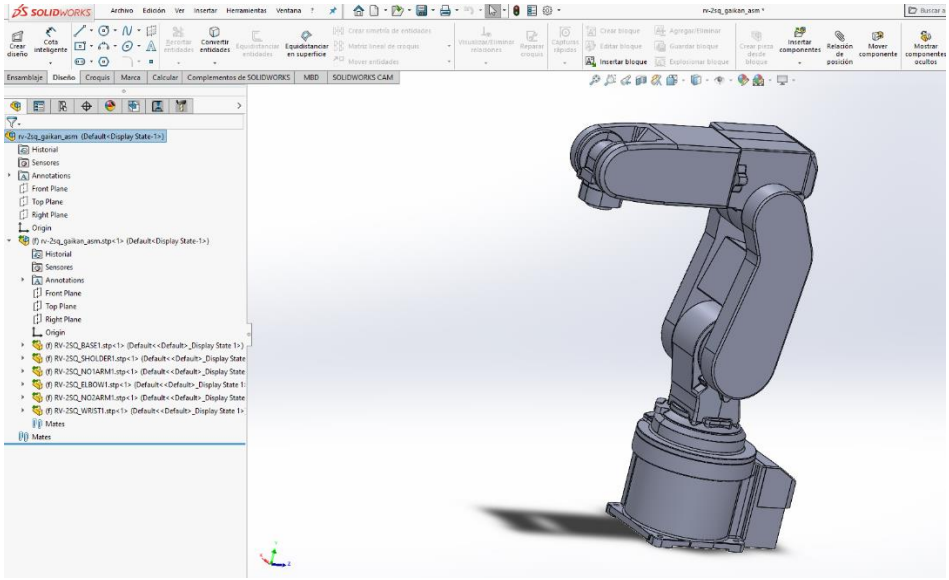
5.1.2.2 Modelo 3D Robot Mitsubishi Melfa RV-2SDB

En la búsqueda sistemática del estado del arte sobre el Robot Mitsubishi Melfa RV-2SDB se encontró que la página oficial MITSUBISHI ELECTRIC AUTOMATION,

tiene un apartado que ofrece modelos de robots del tipo *RV-2SQ/2SD* con el formato .dxf y .step lo cual no fue necesario realizar la modelación en 3D en función de sus planos, solamente se procede a importarlo.

Figura 48

Modelo 3D CAD Robot Mitsubishi Melfa RV-2SDB

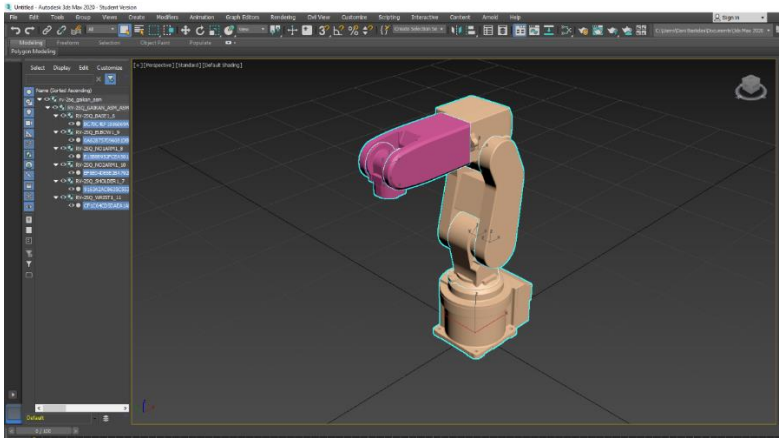


Nota. CAD .step RV-2SQ/2SD Tomado de *Industrial/Collaborative Robots-MELFA Robots CAD Data Downloads* [CAD], por Mitsubishi Electric Corporation, 2010*,*

Se procede a **i**mportar el CAD. step a formato tipo .fbx (objeto 3D) por medio del software **3DS MAX**. En el software se puede especificar las partes que conforma el brazo robótico, verificando que no existan elementos innecesarios para la visualización en objeto.

Figura 49

Modelo del brazo robótico en 3DS MAX

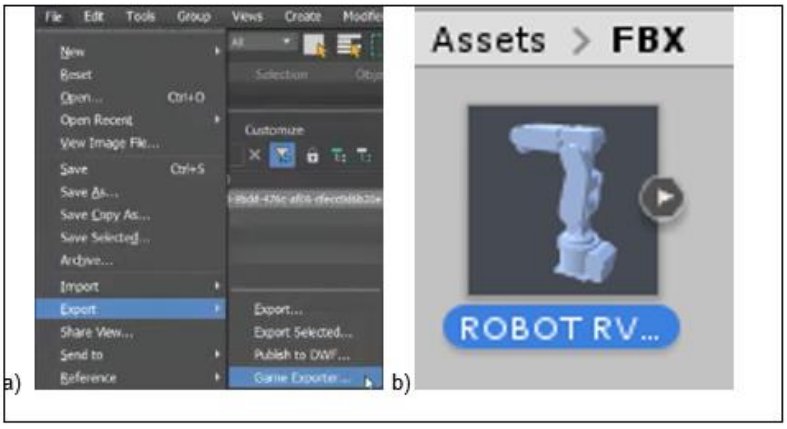


Nota. Selección de elementos que conforman el brazo robótico. La dimensión del modelo 3D son las mismas al manual de especificaciones técnicas del Robot Mitsubishi Melfa RV-2SDB.

El objeto previamente terminando se procede a exportar en la opción de **Archivo, Exportar, Game Exporter**, se coloca como referencia el eje z, debido a que en el estudio de los parámetros D-H es necesario determinar este eje como punto de inicio en rotación.

Figura 50

Exportar modelo a formato .fbx



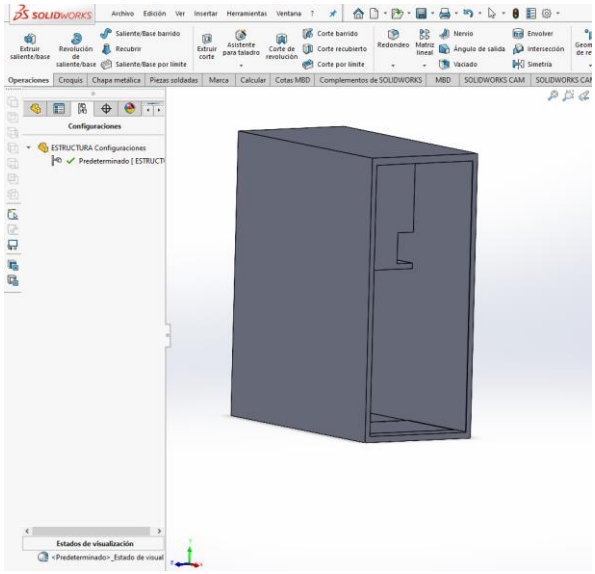
Nota. a) Exportar el modelo desde 3DS MAX como .fbx b) Importar modelo .fbx a Unity

5.1.3 Estructura mesa

El robot es parte del módulo MPS que ofrece Festo, al sugerir que el simulador tenga las mismas características de todos los elementos útiles que conforman el brazo, se partió de la obtención de medidas del brazo robótico que se encuentra en el laboratorio de mecatrónica y posteriormente se realizó el modelado tridimensional en el software Solidworks el modelo de la mesa que contiene al Robot Mitsubishi Melfa RV-2SDB.

Figura 51

Mesa de módulo MPS del Robot



Nota. Modelo de la mesa con las mismas dimensiones que contiene al robot.

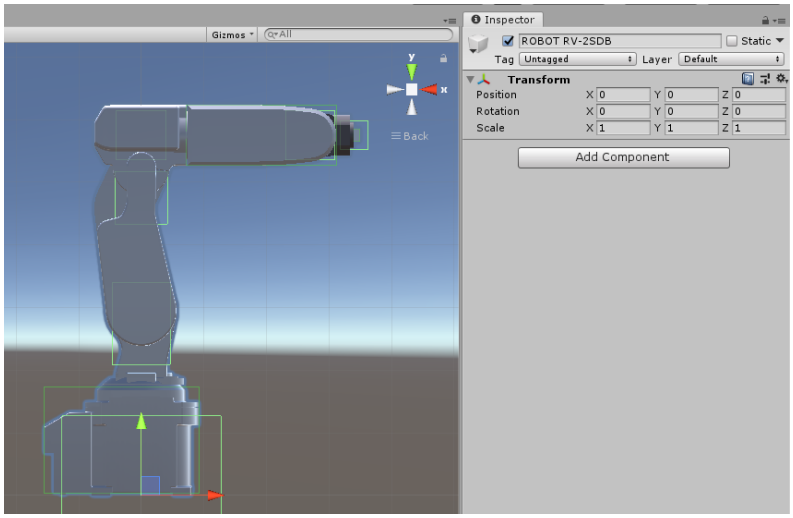
De igual manera la mesa se exporta como objeto 3D en 3DMAX y se ingresa a la escena virtual en Unity.

5.1.4 Ubicación, texturización y renderización de los objetos en la escena

Al colocar los objetos en la escena se debe considerar la escala a trabajar, Unity toma en cuenta que una unidad del espacio en el mundo corresponde a un metro, ejemplo si en el inspector (sistema de referencia) un objeto está en la posición $(0, 1, 0)$ se refiere a que está ubicado a un metro de altura con respecto al sistema fijo del mundo. En el caso del robot su base se encuentra en el origen de la escena $(0,0,0)$, y sus dimensiones están a escala real, lo cual ayuda a determinar la ubicación exacta de sus articulaciones y comparar con los parámetros D-H ya detallados en la sección 4.1.

Figura 52

Robot en el ambiente virtual



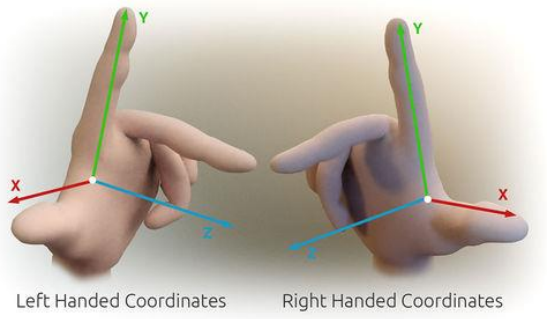
Nota. El brazo robótico se encuentra ubicado en las coordenadas base del mundo virtual.

Al importar el brazo robótico a Unity, este actúa como un solo cuerpo y las articulaciones forman parte de ello. Para que exista una independencia en el movimiento de cada articulación del robot, Unity puedan realizar movimientos por medio de relaciones internas conocidas como jerarquía padre-hijo.

Se debe tomar en cuenta que Unity al igual que otros softwares de gráficos 3D hace referencia a un sistema de coordenadas de la mano izquierda, una representación gráfica se muestra en la Figura 53.

Figura 53

Coordenadas de mano izquierda y derecha



Nota. Sistema de coordenadas de mano izquierda (en referencia Unity 3D) y derecha. Tomado de *3D Cartesian Coordinate Handedness* [Gráfico],por Primalshell, 2013, Wiki CC BY 2.0.

Cada articulación ($A1, A2, A3, A4, A5, A6$) se encuentra dentro de un GameObject como se muestra en la Figura 54.

Figura 54

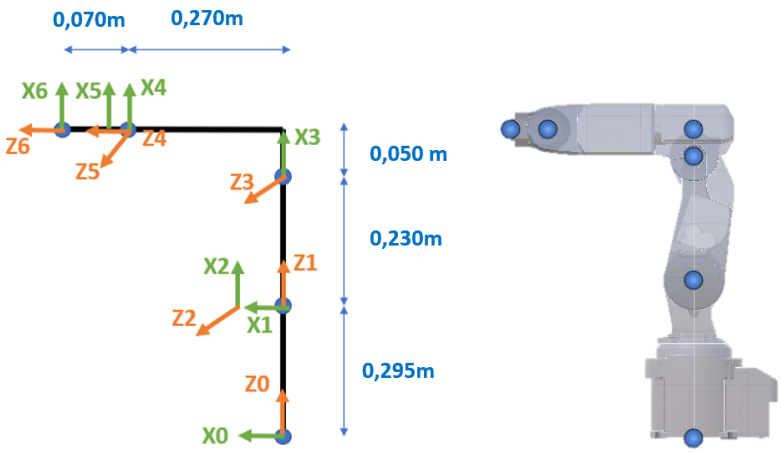
Jerarquía de las articulaciones del robot

Nota. Niveles de jerarquía de las articulaciones, siendo dependientes de sus movimientos desde A6 hasta A1.

Visualmente en el entorno virtual se encuentra distribuido de la siguiente manera.

Figura 55

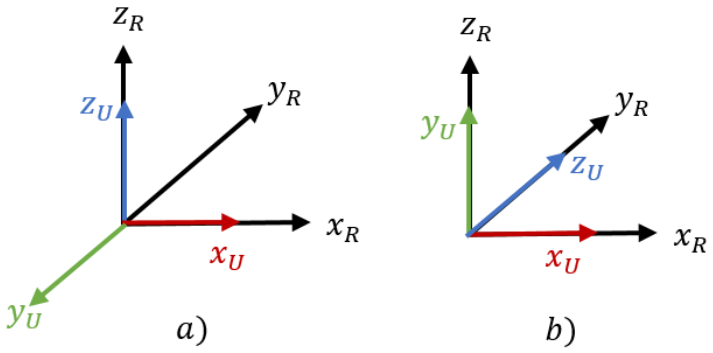
Articulaciones de acuerdo a los parámetros D-H



Unity normalmente el eje Y apunta hacia arriba, para migrar al sistema diestro en Unity 3D se invierte el orden de los vértices como se encuentra en la Figura 56.

Figura 56

Migración de sistema de referencias



Nota. a) Sistema de coordenadas regla mano derecha (X_R, Y_R, Z_R) con el sistema de coordenadas regla mano izquierda de Unity3D (X_U, Y_U, Z_U). b) Al rotar el eje x en sentido de horario, logra una equivalencia de los vértices de X_R, Y_R, Z_R, pasan a X_U, Z_U, Y_U

Se sitúan todos los objetos en la escena y se agregan materiales y texturas en cada elemento, esto se complementa para generar ya una superficie renderizada. Para dar soporte visual en el ambiente y se puedan observar más elementos en el exterior se incluye un skybox (una imagen exterior que se relaciona con la escena de un laboratorio virtual) se inserta según los pasos de (GameDevTraum, 2020b).

Se fijan los puntos para la iluminación en el entorno y por último se añade 3 cámaras (frente, superior y lateral del robot) que determina los diferentes ángulos de visualización que se encuentre el robot manipulador en el modo de juego.

Figura 57

Visualización del ambiente virtual

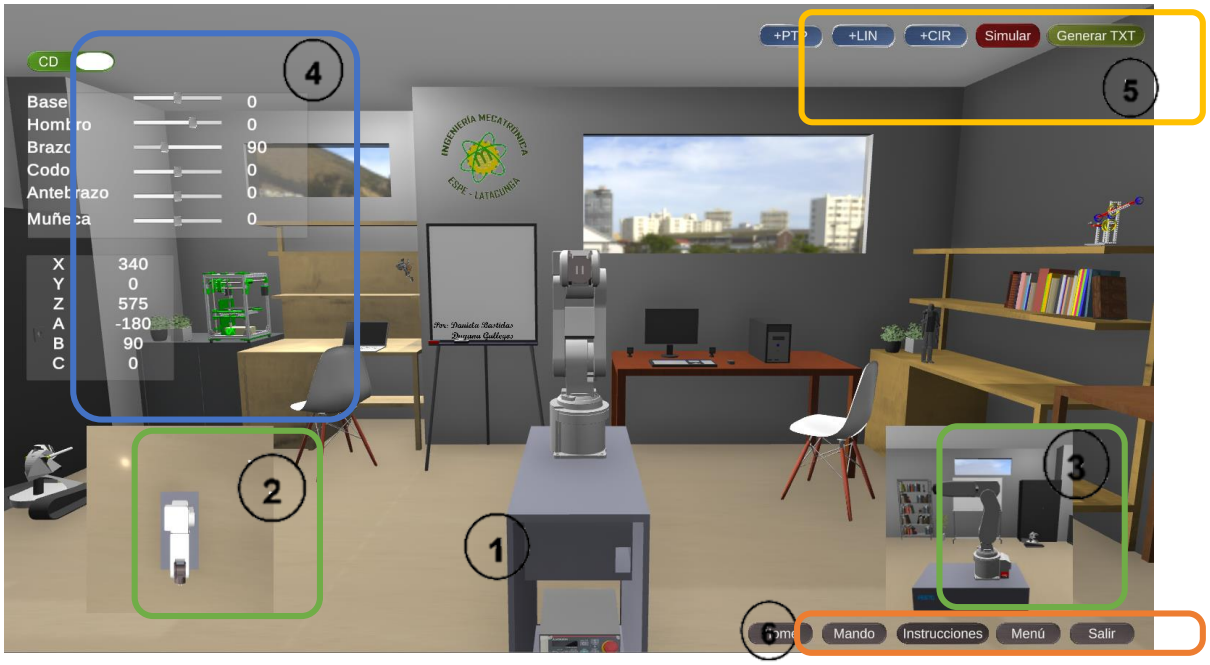
Nota. Escena renderizada con materiales y texturas, iluminación dentro y fuera del aula virtual, se visualiza las cámaras ubicadas cerca del robot y en el fondo esta una imagen de una comunidad universitaria como skybox.

5.1.5 Interfaz de usuario

La interfaz de usuario es un conjunto de controles que dan soporte visual como interactivo para el usuario, Unity proporciona un objeto llamado Canvas encargado en mantener todos los elementos UI del motor de juego.

Figura 58

Interfaz del simulador virtual



1. Cámara principal configurada a la vista del espectador en primer plano.
2. Cámara secundaria configurada a la vista superior del robot en segundo plano, sí el usuario selecciona esta cámara, la visualización general cambia a primer plano del espectador.
3. Cámara secundaria configurada a la vista lateral derecho del robot en segundo plano, sí el usuario selecciona esta cámara, la visualización general cambia a primer plano del espectador.
4. Panel cinemática del robot, puede ser cinemática directa o inversa dependiendo el estado del toogle (switch de dos posiciones 0,1)

5. Panel de trayectorias, el usuario crea el número de trayectorias a generar.
6. Sistema de comandos iniciales.

5.1.6 Instrucciones para la interacción con el simulador

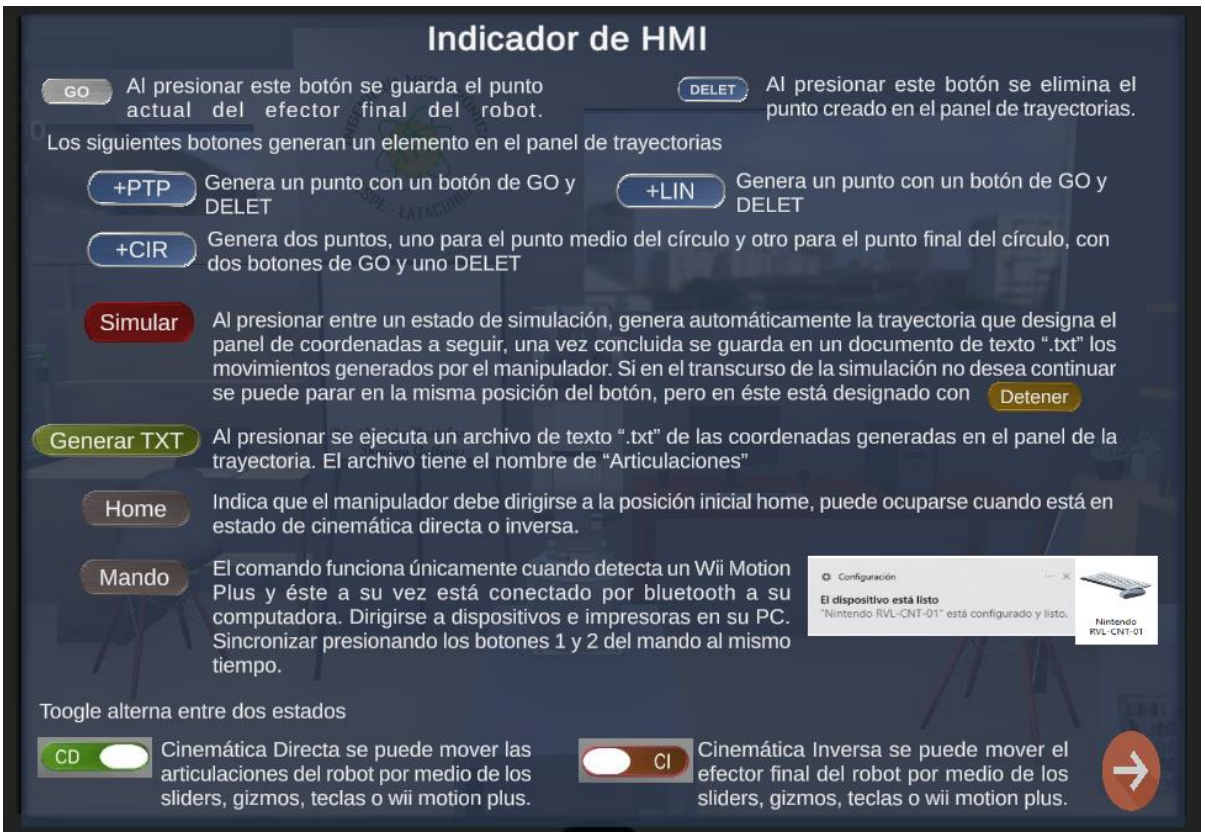
A continuación se muestran 3 pantallas con instrucciones del funcionamiento del simulador. En cada pantalla se detalla las caracteristicas especificas de cada elemento.

5.1.6.1 Indicador de botones.

En la primera pantalla de instrucciones se muestra el funcionamiento de cada botón de la interfaz de la Figura 57.

Figura 59

Instrucciones de la Pantalla #1



Nota. Indicador del HMI, a continuación se detalla cada elemento de la UI.

- GO, al presionar este botón se guarda el punto actual del efector final del robot.
- DELET, al presionar este botón se elimina el punto creado en el panel de trayectorias.

  Los siguientes botones generan un elemento en el panel de trayectorias.

- +PTP, genera un punto con un botón de Go y Delet
- +LIN, genera un punto con un botón de Go y Delet
- +CIR, genera 2 puntos, uno para el punto medio del círculo y otro para el punto final del círculo, con dos botones de Go y uno Delet
- Simular, al presionar entre un estado de simulación, genera automáticamente la trayectoria que designa el panel de coordenadas a seguir, una vez concluida se guarda en un documento de texto ".txt" los movimientos generados por el manipulador. Si en el transcurso de la simulación no desea continuar se puede parar en la misma posición del botón, pero en éste está designado con Detener.
- Generar TXT, al presionar se ejecuta un archivo de texto ".txt" de las coordenadas generadas en el panel de la trayectoria. El archivo tiene el nombre de "Articulaciones"
- Home, indica que el manipulador debe dirigirse a la posición inicial home, puede ocuparse cuando está en estado de cinemática directa o inversa.
- Mando, el comando funciona únicamente cuando detecta un Wii Motion Plus y éste a su vez está conectado por bluetooth a su computadora. Dirigirse a dispositivos e impresoras en su PC. Sincronizar presionando los botones 1 y 2 del mando al mismo tiempo. En la siguiente sección se detalla como sincronizar

Toogle alterna entre dos estados

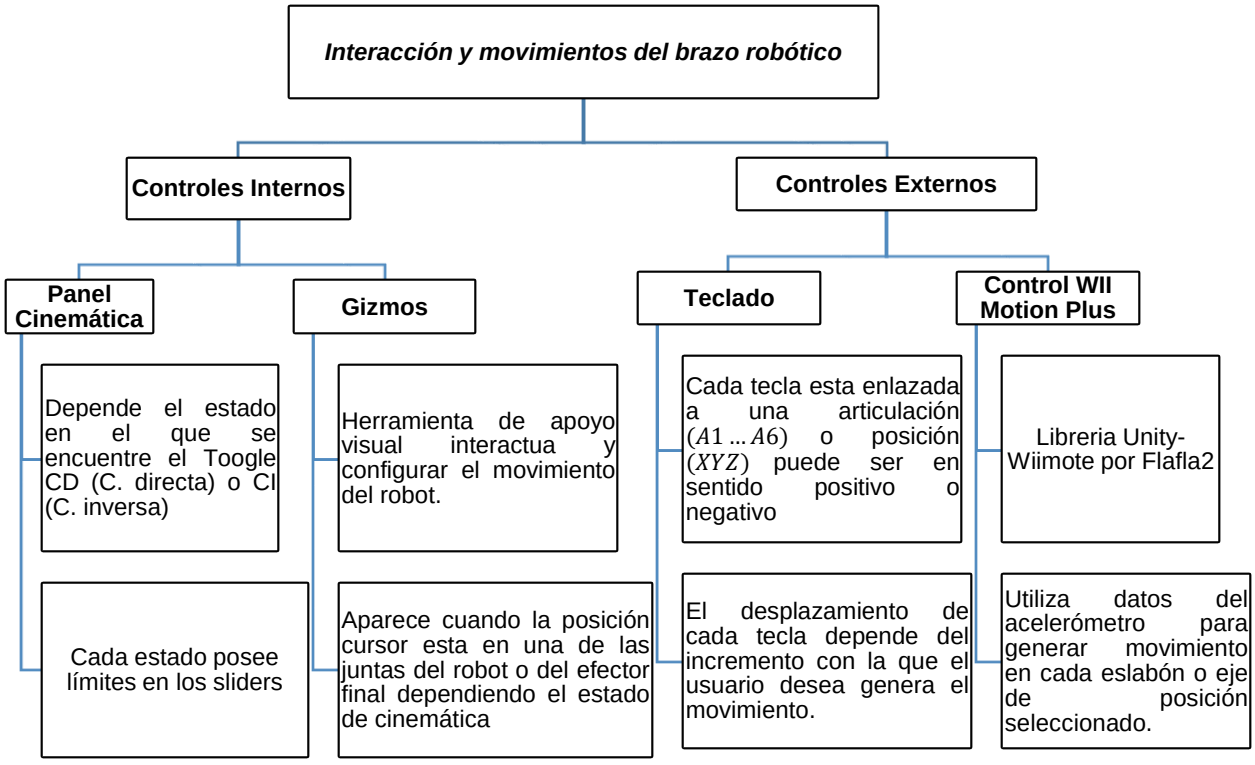
- CD, cinemática directa se puede mover las articulaciones del robot por medio de los sliders, gizmos, teclas o Wii Motion Plus.
- CI, cinemática inversa se puede mover el efector final del robot por medio de los sliders, gizmos, teclas o wii Motion plus.

5.1.6.2 Instrucciones de la interacción y movimientos del robot.

Existen 4 formas de modificar el comportamiento del brazo robótico, estos controles son internos (controlan los movimientos por medio de la interfaz) o externos (son dispositivos que envían datos por medio de comunicación a la PC), y se lo define en el siguiente organigrama.

Figura 60

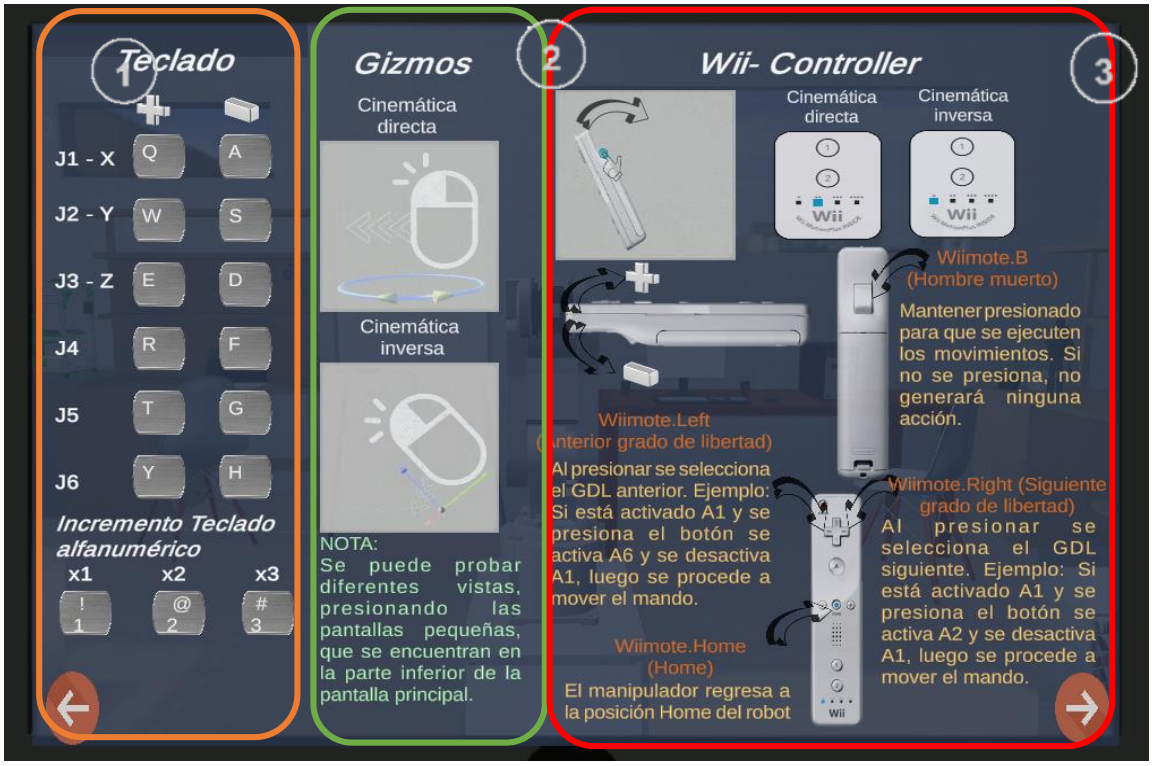
Marco organizativo del movimiento del robot



La segunda pantalla de la Figura 61, se indica los elementos de entrada aparte de los sliders que hablara en la siguiente sección, dichos elementos permiten mover los ángulos de las articulaciones del robot o la posición del efector final.

Figura 61

Instrucciones de la Pantalla #2



Nota. **1.** Entrada de datos por teclado. **2.** Entrada de datos por Gizmos por medio de la pantalla. **3.** Entrada de datos por Wii Motion Plus.

5.1.6.2.1 Panel Cinemática.

En la Figura 62 se muestra un control interno para el movimiento del robot, que es el panel cinemático que dependerá de su estado directo o inverso, este panel está conformado por sliders y los datos de salida cinemática.

Figura 62

Partes del panel cinemático

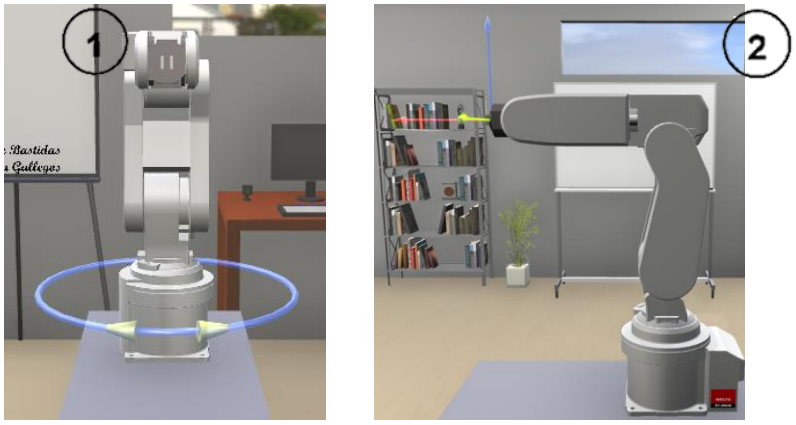
Nota. **1.** Panel de cinemática directa. **2.** Panel de cinemática inversa.

5.1.6.2.2 Gizmos.

Otra manera de manipular el robot por medio de las articulaciones o el efector final, proporcionando mayor comodidad al usuario en la escena.

Figura 63

Proyección de Gizmos en el simulador



Nota. **1.** Gizmos en estado de cinemática directa, se proyecta solo en las articulaciones. **2.** Gizmos en estado de cinemática inversa, se proyecta en el efector final.

Para realizar los movimientos se necesita colocar el puntero del mouse en las articulaciones o el efector final del robot y se desplaza de izquierda a derecha o viceversa en la pantalla del juego.

5.1.6.2.3 Teclado.

Las teclas designadas en la Figura 61 en el estado de CD (Cinemática Directa) o CI (Cinemática Inversa) ayuda al movimiento del robot. Donde los signos $+$ y $-$ representa el sentido antihorario y horario respectivamente.

Teclado alfanumérico	Incremento CD (Ángulo en articulación)	Incremento CI(Desplazamiento en xyz)
1	$10\ mm * dt$	$1\ mm * dt$
2	$100\ mm * dt$	$10\ mm * dt$
3	$1000\ mm * dt$	$100\ mm * dt$

Al pulsar una tecla alfanumérica se genera el incremento para los botones teclas “Q,A,W,S,E,D,R,F,T,G,Y,H”. Donde dt es el intervalo de muestra.

5.1.6.2.4 Control Wii Motion Plus (WMP).

A continuación se detalla la conexión al simulador virtual, y su las caracteristicas del Wii motion Plus, para su posterior uso.

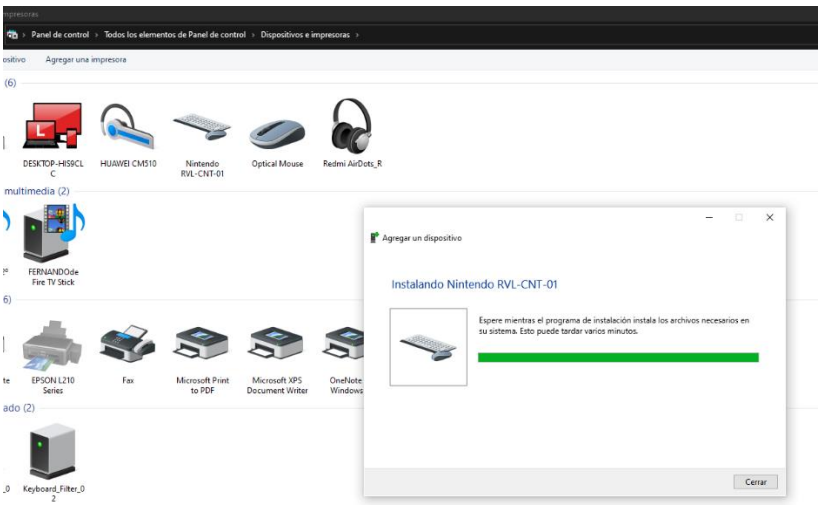
- **Configuración comunicación del WMP en PC**

Para poder utilizar el mando es necesario configurarlo, para ello realizamos los siguientes pasos:

1. En el PC, dirigirse al panel de control en “**Dispositivos e impresoras”** para agregar un nuevo dispositivo.
2. En el WMP, Para establecer la comunicación bluetooth entre la PC y el mando, mantener presionados los botones del WII “**1**” y “**2**”.
3. En el PC, Seleccionar el dispositivo **Nintendo RVL-CNT-01**
4. En el PC, Posteriormente da inicio a la instalación del dispositivo en la PC

Figura 64

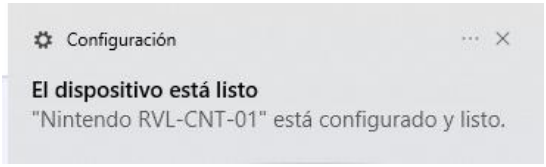
Configuración para comunicación WII



5. Para confirmar que el dispositivo se enlazó adecuadamente debe aparecer en la pantalla de inicio la notificación de configuración **“El dispositivo está listo”**

Figura 65

Indicador del dispositivo conectado



Nota. Este apartado indicara que realmente el dispositivo se ha conectado con éxito.

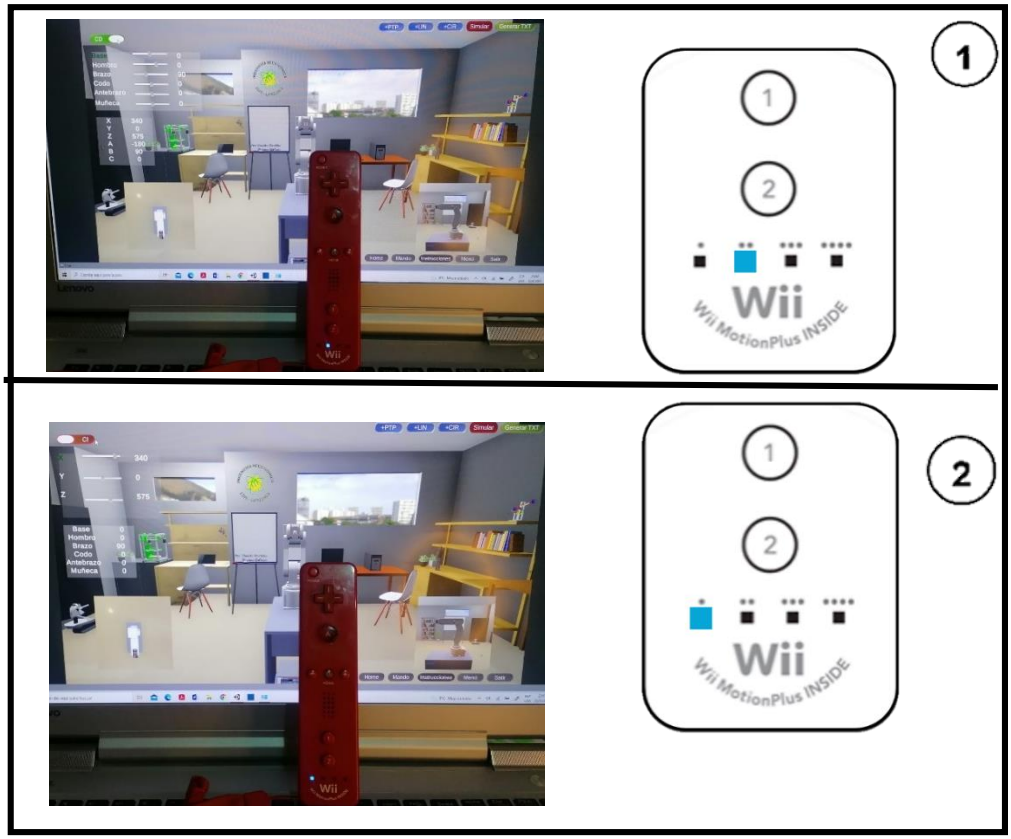
- **Comunicación WMP a Unity**

Ahora que está conectando el WMP a la PC por medio de bluetooth, se conectará el mando a Unity, para ello realizamos los siguientes pasos:

1. Presionar el botón “**Mando”** en la UI de Unity, funciona siempre y cuando detecta el dispositivo conectado. El mando iniciará la comunicación con Unity encendiendo de manera secuencial el primer led hasta el cuarto led del WMP.
2. El estado en el que se encuentra el simulador (cinemático directo o cinemático inverso), dependerá el led indicador que se encienda. Como se ve a continuación.

Figura 66

Estado de cinemática del Wii Motion Plus



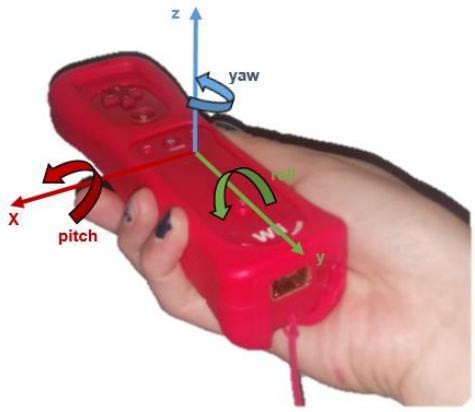
Nota. 1. Cinemática directa. 2. Cinemática inversa

- **Movimientos del manipulador con el WMP**

El sistema de coordenadas del WMP tiene arquitectura diferente al manipulador como se muestra en la Figura 67.

Figura 67

Coordenadas de referencia de Wii Motion Plus



Nota. Arquitectura del WMP, donde las rotaciones de sistema xyz.

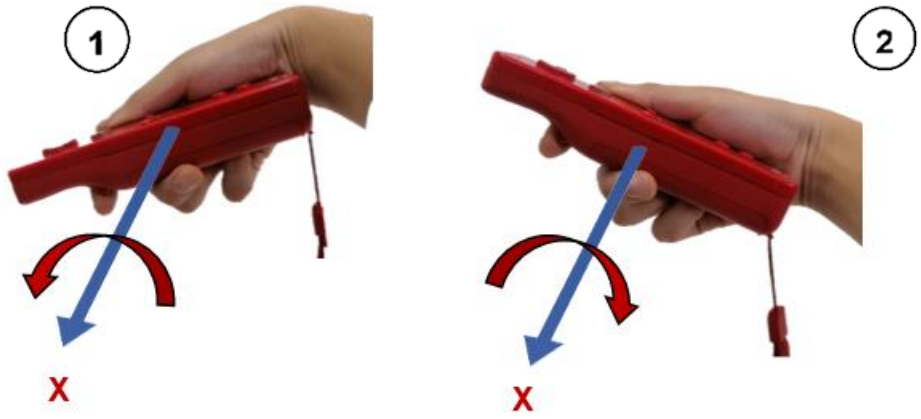
En la Figura 61, detalla los botones del WMP, y la función que realiza en el manipulador robótico del simulador. Como se presenta a continuación.

- Wiimote.Left Anterior grado de Libertad; al presionar se selecciona el GDL anterior. Ejemplo:Si está activado A1 y se presiona el botón se activa A6 y se desactiva A1, luego se procede a mover el mando.
- Wiimote.Right Siguiente grado de Libertad ; al presionar se selecciona el GDL siguiente. Ejemplo: Si está activado A1 y se presiona el botón se activa A2 y se desactiva A1, luego se procede a mover el mando.
- Wiimote.Home El manipulador regresa a la posición Home del robot
- Wiimote.B Hombre muerto. Mantener presionado para que se ejecuten los movimientos. Si no se presiona, no generará ninguna acción.

Para relacionar el movimiento de las articulaciones o el efector final con el Wii se ocupa la velocidad angular (pitchspeed) y girar horario o antihorario sobre el eje X, el movimiento se ejecuta siempre y cuando el botón de Hombre muerto B, esté presionado.

Figura 68

Rotación del Wii para movimiento del robot



Nota. 1. Antihorario (positivo) 2. Horario (negativo).

Figura 69

Funcionamiento del WMP en articulaciones

Nota. El indicador para conocer que articulación o eje del efector final se mueve es cuando el texto del mismo cambia de color a verde.

5.1.6.3 Instrucciones de límites del manipulador en simulación.

En esta pantalla se detallan las características técnicas del robot en el simulador para que se mantenga presente, como seguridad se añaden límites en el valor de las juntas como también en los valores de posición, como se muestra a continuación.

Figura 70

Instrucciones de la Pantalla #3

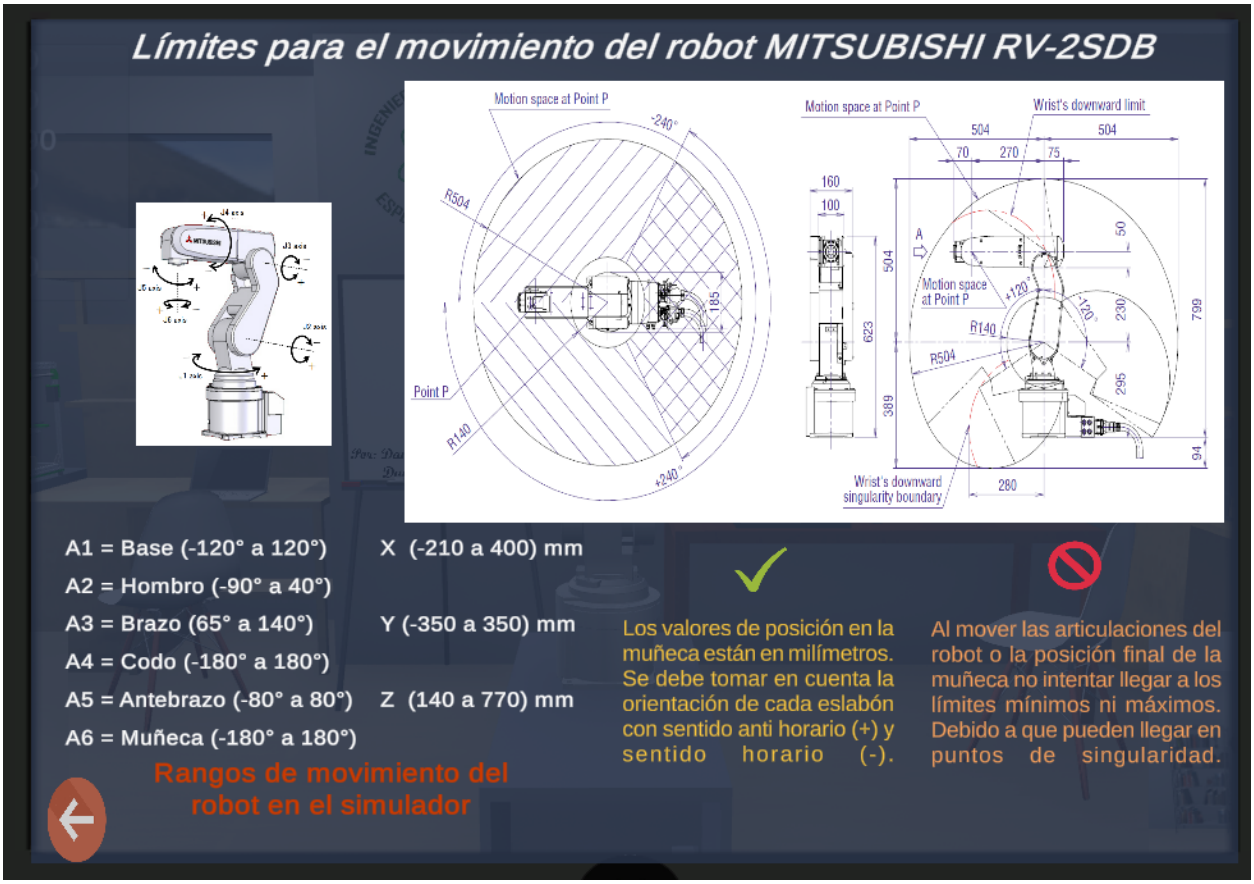


Tabla 9

Límites impuestos del brazo robótico en el simulador

KD	IK
Base (-120° a 120°)	X (-210 a 400) mm
Hombro (-90° a 40°)	
Brazo (65° a 140°)	Y (-350 a 350) mm
Codo (-180° a 180°)	
Antebrazo (-80° a 80°)	Z (-140 a 770) mm
Muñeca (-180° a 180°)	

Los valores de posición en la muñeca están en milímetros. Se debe tomar en cuenta la orientación de cada eslabón con sentido anti horario (+) y sentido horario (-).

Al mover las articulaciones del robot o la posición final de la muñeca **no** intentar llegar a los límites mínimos ni máximos. Debido a que pueden llegar en puntos de singularidad.

5.2 Programación y scripting del sistema de realidad virtual no inmersivo

5.2.1 Movimiento del brazo robótico

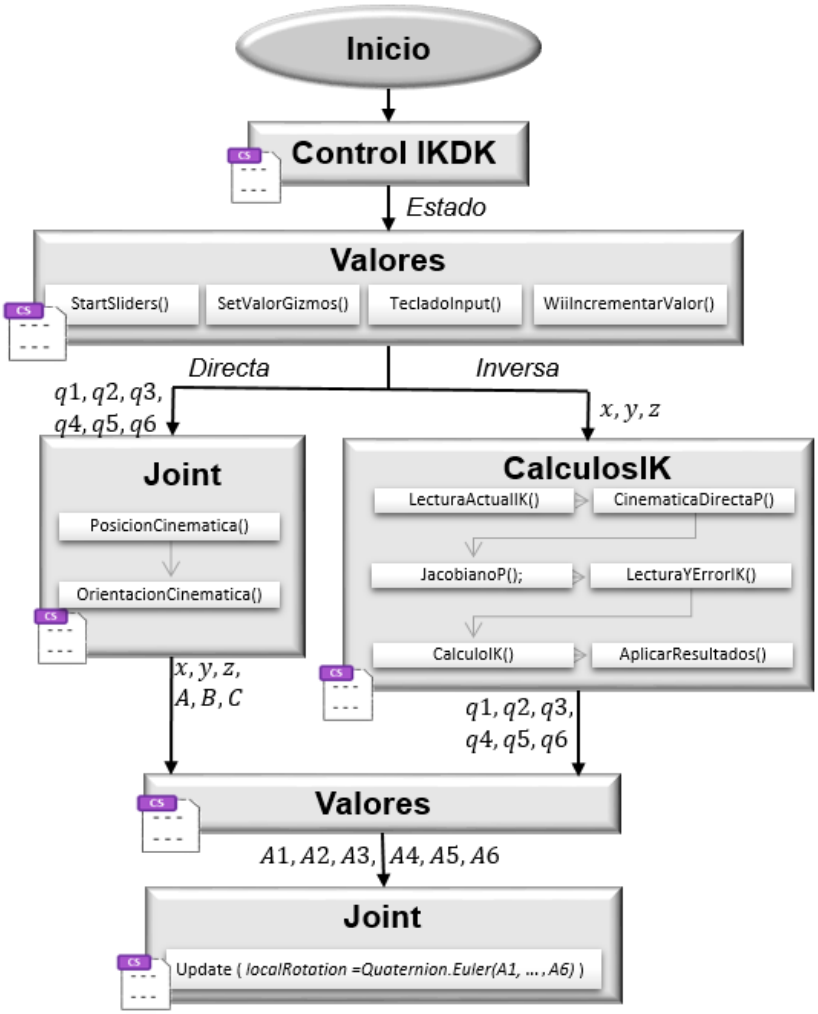
En el siguiente esquema se detallan las clases usadas en el scripting en Unity 3D.

Cada clase están heredados de MonoBehaviour, esto es un conjunto de variables, objetos y funciones proporcionados por Unity 3D. MonoBehaviour es como la clase que les permite inicializarse automáticamente a través del método Start y ejecutar automáticamente el método Update. (GameDevTraum, 2020a)

Se detalla en cada clase los métodos más importantes que realizan las acciones del simulador.

Figura 71

Programación para el movimiento cinemático del robot



Nota. Cada cuadro representa una clase complementando el movimiento cinemático del robot.

- Control IKDK es la clase que realiza un cambio de estado entre cinemática directa, Cinemática Inversa y Simulación. Por lo cual se activan o desactivan algunos componentes propios de cada estado.

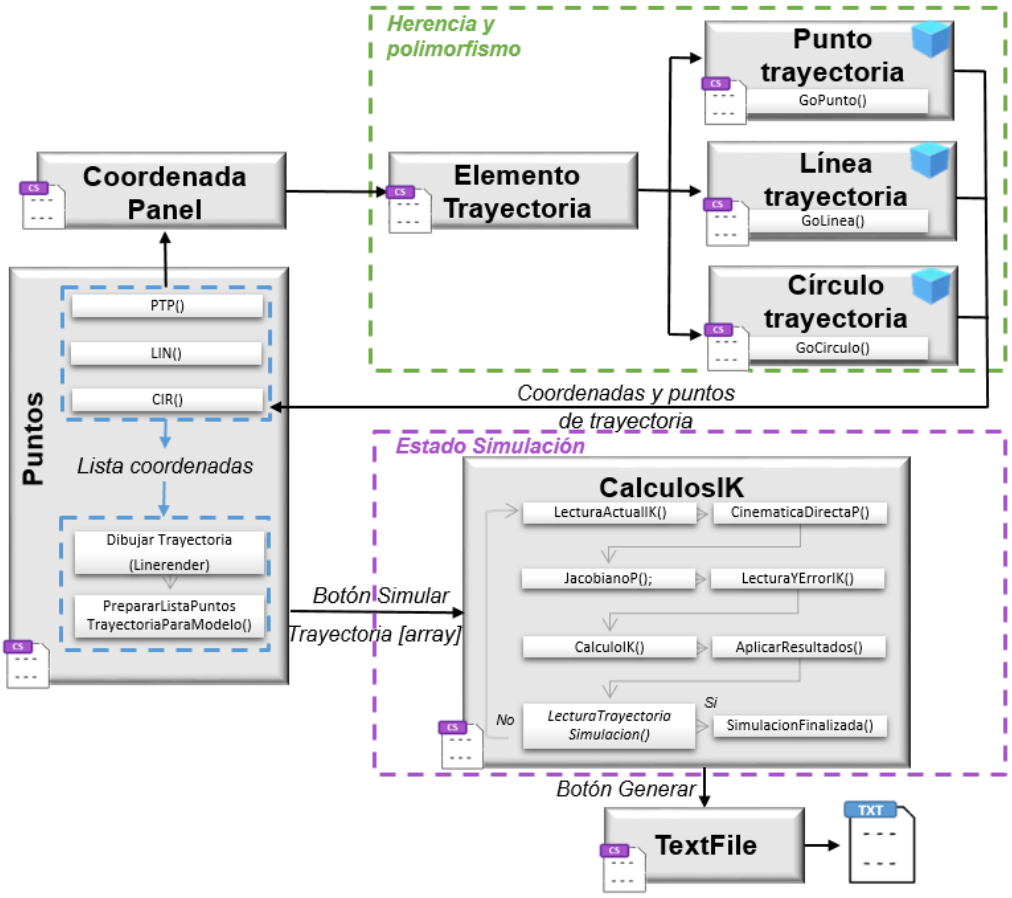
- Valores: esta clase realiza la trasferencia de datos ya sea de lectura o escritura de las articulaciones y posiciones asegurando la existencia de un solo control en todo el sistema de entradas para el movimiento del brazo robótico. Esta clase provee funciones para controlar las variables del sistema, permitiendo que exista distintos métodos de control sin que se bloqueen entre sí.
- **Joint:** genera un movimiento de rotación en cada articulación agregado en un game object del robot, así como también el método de posición y orientación cinemática que realiza cálculos matemáticos de la cinemática directa, aquí se encuentran tanto la matriz de transformación homogénea como la matriz de orientación.
- **CalculoIK:** realiza el algoritmo de control cinemático inverso tomado de la sección del modelamiento matemático

5.2.2 Generación de trayectoria

Las trayectorias deseadas por el usuario, se complementan con este scripting. Se detalla a continuación la programación para generar las trayectorias.

Figura 72

Programación para la generación de trayectorias.

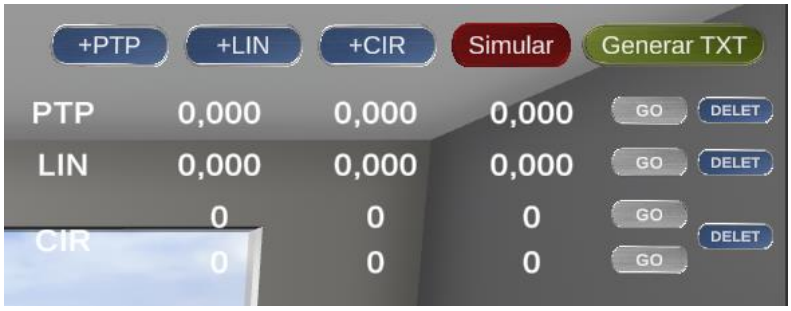


Nota. Cada cuadro representa una clase para generar las trayectorias deseadas.

- **Puntos:** En esta clase, las primeras funciones hacen referencia a qué es lo que desea el usuario, puede ser generar uno o varios (punto, línea o círculo). A Estos objetos se les asigna un script, al principio serán de coordenadas nulas como se observa en la Figura 73. excepto cuando el usuario ingresa el valor en la coordenada de cada objeto generado. Por lo cual se necesita de la clase coordenada panel.

Figura 73

Clase puntos para elementos de trayectoria



Nota. Se crean los puntos para cada trayectoria deseada.

- **Coordenada panel:** esta clase diferencia el tipo de objeto (PTP, LIN O CIR) que creó el usuario en el panel de trayectorias, lo cual asigna una instancia de coordenadas. Internamente pide posiciones del punto inicial, final y los puntos intermedios de la trayectoria, pero estos se generarán a partir de elemento trayectoria, es por eso que se detalla una parte fundamental en esta sección. Como se muestra en la Figura 74.

Figura 74

Clase coordenada panel

Nota. Ingreso de puntos como datos en cada trayectoria ya sea punto, línea y circulo.

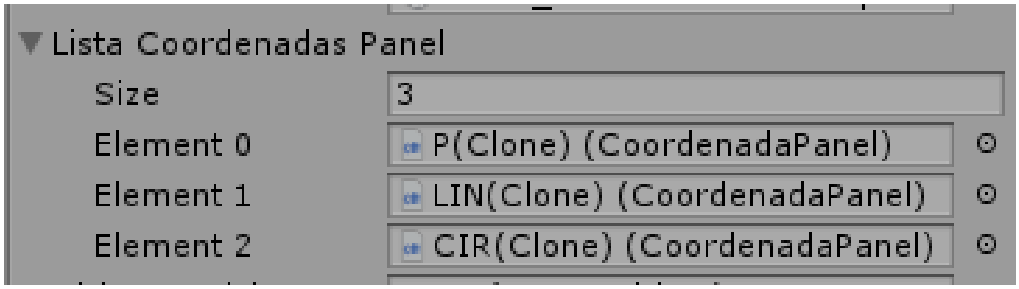
- **Herencia y polimorfismo.** En el recuadro de la Figura 72, la herencia permite generar una estructura en la que se tiene clases que derivan de otras clases, por un lado, se tiene las súper clases que agrupan variables y funciones generales y por otro lado las subclases que tienes variables y funciones específicas, esto permite aumentar la abstracción y reutilizar código. El polimorfismo es la capacidad de que objetos de distinta clase respondan a un mismo mensaje o función, esto permite modificar el código dependiendo del objeto, el sistema propuesto se obtiene a partir del artículo de (Bravo et al., 2021) donde realiza un paquete modificable del mismo.
- **Elemento trayectoria:** las superclases heredan las clases (punto trayectoria, línea trayectoria y círculo trayectoria) esta clase identifica el tipo de trayectoria y como se representa en el espacio para poder dibujar la trayectoria
- **Punto trayectoria, Línea trayectoria y Círculo trayectoria**: conocidas como el tipo de trayectoria, estas clases generan instancias objetos (prefaps) donde se coloca la coordenada xyz actual del efector final (obtenido directamente desde Unity) y hasta la posición que se desea recorrer a partir de la clase coordenada panel accionadas por la función de Go (procedimiento, pasos ingresados por usuario), a su vez estas clases poseen métodos para el cálculo matemático de cada trayectoria descrita en el cap. 4 en la sección 4.4. de trayectorias, al concluir se produce una lista de puntos de trayectoria para que se ejecute la simulación. Como se muestra en la Figura 75.
- **Puntos**: Otra función de esta clase es que genera una lista de coordenadas para la trayectoria Figura 76, desde coordenada panel tomando los

elementos (PTP LIN, CIR) hasta puntos necesarios almacenar y enviar por medio de un array hacia el estado de simulación y a su vez se puede visualizar como es la proyección del movimiento del manipulador como la Figura 77.

- **CalculosIK:** Esta clase se describió previamente pero adicional se cambia el estado de inversa a simulación, donde se ejecuta la función de lectura de trayectoria de n puntos creados.

Figura 75

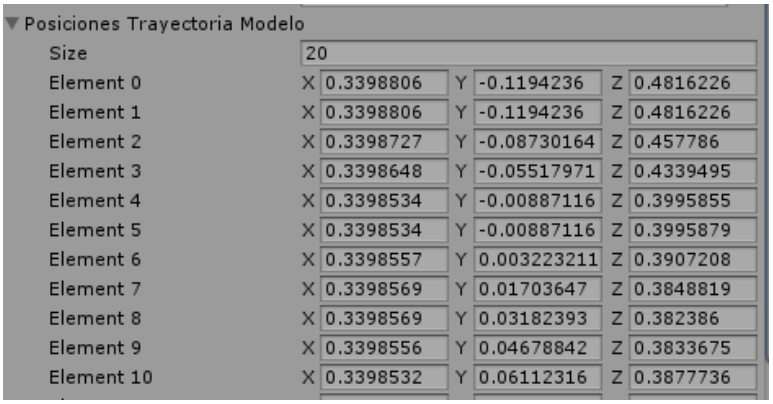
Clase coordenada panel



Nota. En cada elemento de esta lista están las coordenadas de Punto, línea o círculo donde se asigna una clase específica para generar su trayectoria, gracias a la herencia y polimorfismo.

Figura 76

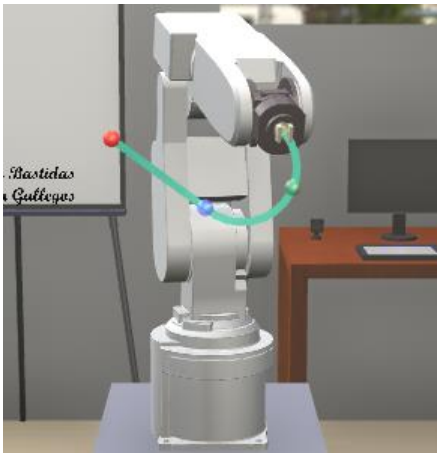
Array puntos trayectoria



Nota. Lista de puntos para la trayectoria almacenados en clase Puntos

Figura 77

Proyección de trayectoria

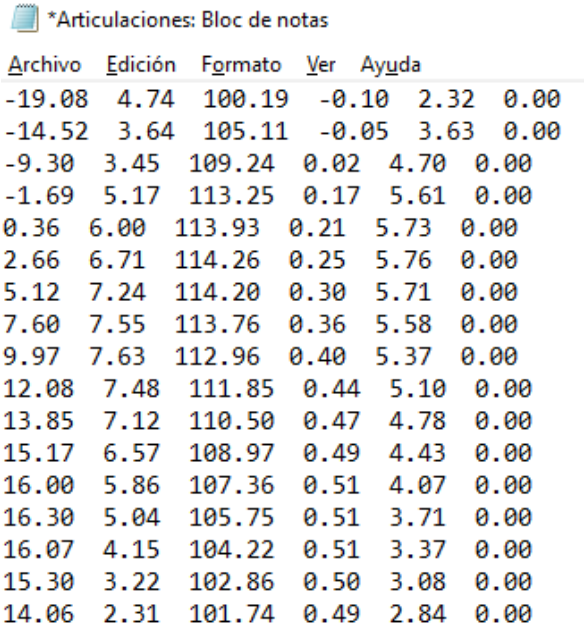


Nota. La esfera roja es el elemento PTP. La esfera azul es el elemento LIN. La esfera verde es el elemento del punto medio del círculo CIR y la esfera amarilla es el elemento del punto final del círculo CIR.

- **TextFile:** esta clase posee una función File para generar tres archivos de texto ".TXT"". El primer archivo llamado "Articulaciones" es array de los valores de las articulaciones para poder ejecutarlo en matlab. El segundo archivo llamado "Coordenas " se encuentran los puntos XYZ generados de la trayectoria. El tercer archivo "Instrucciones" son los elementos que el usuario coloca en el panel de trayectorias (PTP, LIN, CIRC). Todos estos archivos se guardan la carpeta *Robot Melfa RV-2SDB_Data* donde se encuentra el proyecto del simulador.

Figura 78

Archivo TXT de la trayectoria generada



Nota. Los datos del archovo de textor ".txt", son valores de las articulaciones en las posiciones dadas.

5.2.3 Mandos externos del simulador

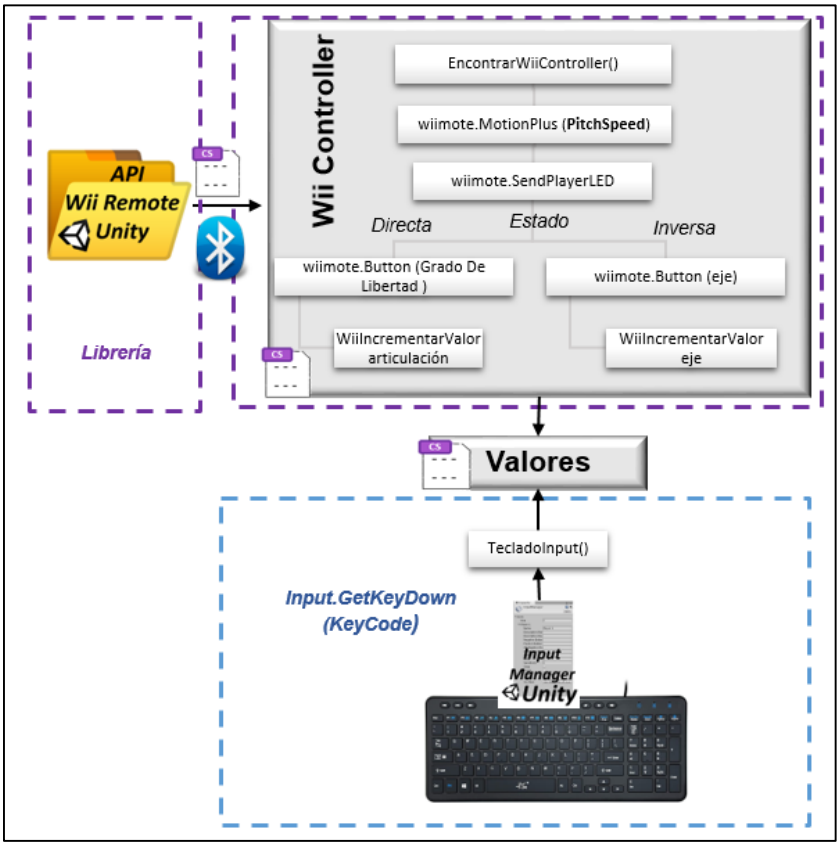
Gracias a la API obtenida por (Biagioli et al., 2016) se realiza la comunicación ya detallada en la sección Wii Remote 3.2.1.

La clase Wii Controller se centra en obtener información del paquete de motion plus para obtener la velocidad del acelerómetro en el parámetro pitch, con sus datos se genera el movimiento ya sea de articulaciones como de la posición de los ejes dependiento del estado en el que se encuentren .

En la clase valores se incorporó otro mando, el teclado con la función de input proporcinado directamente de Unity, es un administrador de dispositivos de entrada referido como KEY.

Figura 79

Programación para mandos externos del simulador



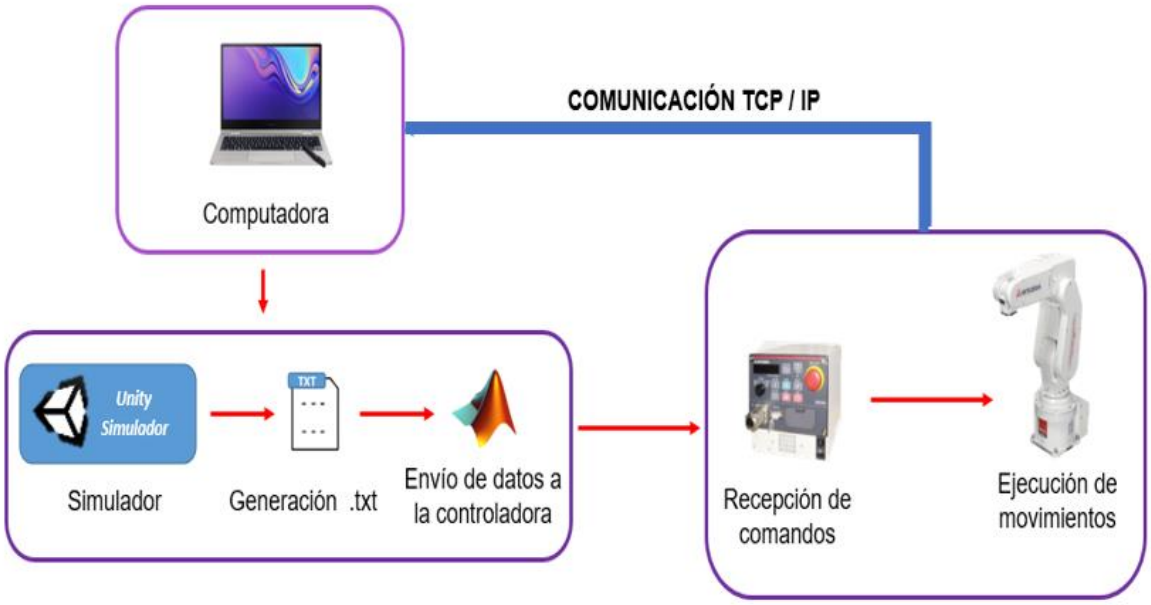
Nota. Los controles externos contienen librerias independientes para modificar los valores en la manipulación del robot.

5.3 Comunicación con el brazo robótico Mitsubishi RV 2SDB

El brazo robótico Mitsubishi tiene una controladora CR1DA – 700, la misma dispone de diferentes tipos de conexiones para que pueda enlazarse con otros dispositivos, la comunicación con el simulador se la realiza mediante TCP / IP, de manera general el software no inmersivo de realidad virtual entrega un archivo .txt el cual es leído por el software Matlab, procediendo a enviar dichos datos a la controladora para que ejecute las trayectorias generadas, en la Figura 80 se logra visualizar este sistema.

Figura 80

Operación off – line del brazo robótico Mitsubishi RV 2SDB



Nota. La operación off – line se realiza con una programación fuera de línea con el simulador para que después los datos sean leídos en el software de Matlab y envíe la información a la controladora del brazo físico.

5.3.1 Comunicación TCP / IP Controladora – Software

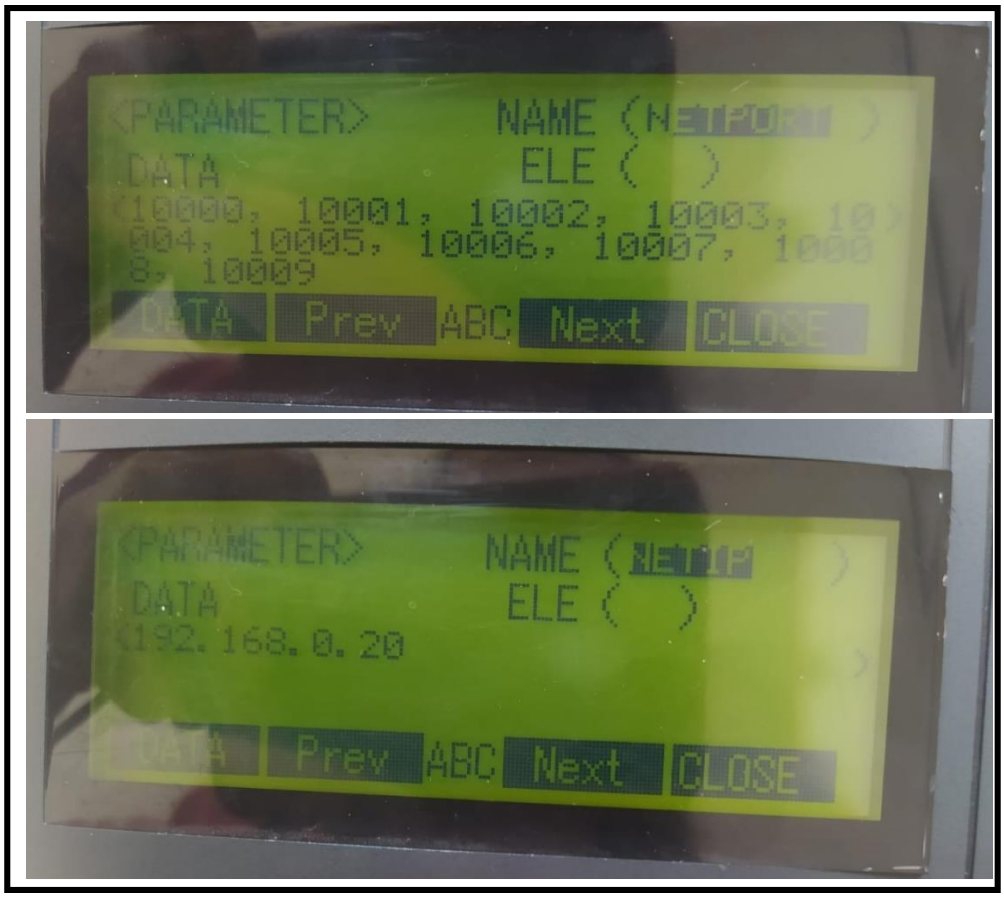
La comunicación TCP/IP establece el intercambio de datos entre dos equipos, cada equipo debe tener su dirección IP designada, en el caso de la controladora el parámetro se lo puede identificar en el manual o en el Teaching Pendant, siguiendo los siguientes pasos:

1. Encender la controladora.
2. Seleccionar el botón Power del TB
3. Pulsar el botón 3 perteneciente al ítem de parámetros.

4. Buscar los parámetros siguientes: NETIP, NETPORT, como se indica a Figura 81.

Figura 81

Parámetros NETIP, NETPORT de la controladora CR1D 700

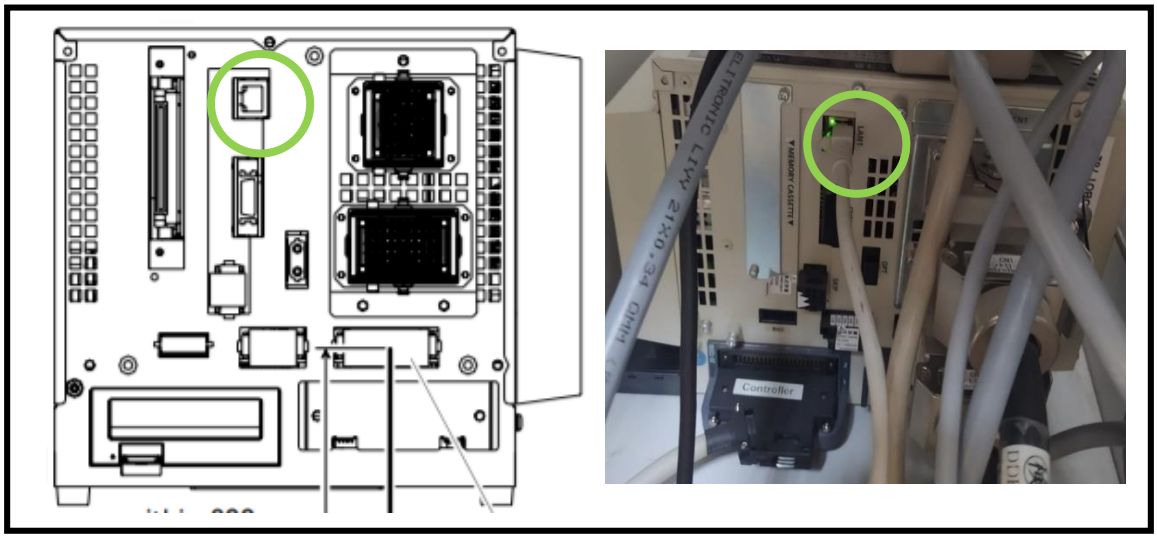


Nota. Parámetros NETIP, NETPORT, necesarios para poder establecer la comunicación TCP/IP entre la controladora y la computadora del usuario.

La IP y el puerto de la controladora se ingresan en el software de Matlab ya que con eso se direcciona hacia donde se envían los datos. Es necesario tener conectado el cable ethernet, un conector va hacia el puerto LAN de la controladora CR1D 700, dicho puerto se encuentra en la parte posterior, el otro conector se conecta a la computadora del usuario.

Figura 82

Ubicación del puerto LAN en la controladora CR1D-700



Nota. Los círculos verdes representan la ubicación del puerto LAN en la controladora del robot Mitsubishi RV – 2SDB

Para poder establecer la comunicación con la computadora, la controladora debe estar en modo automático, los paros de emergencia deben estar desactivados y el botón de Enable del TB (ubicado en la parte posterior, color blanco) de igual manera debe estar apagado, caso contrario las alarmas se activan y no puede realizarse la conexión.

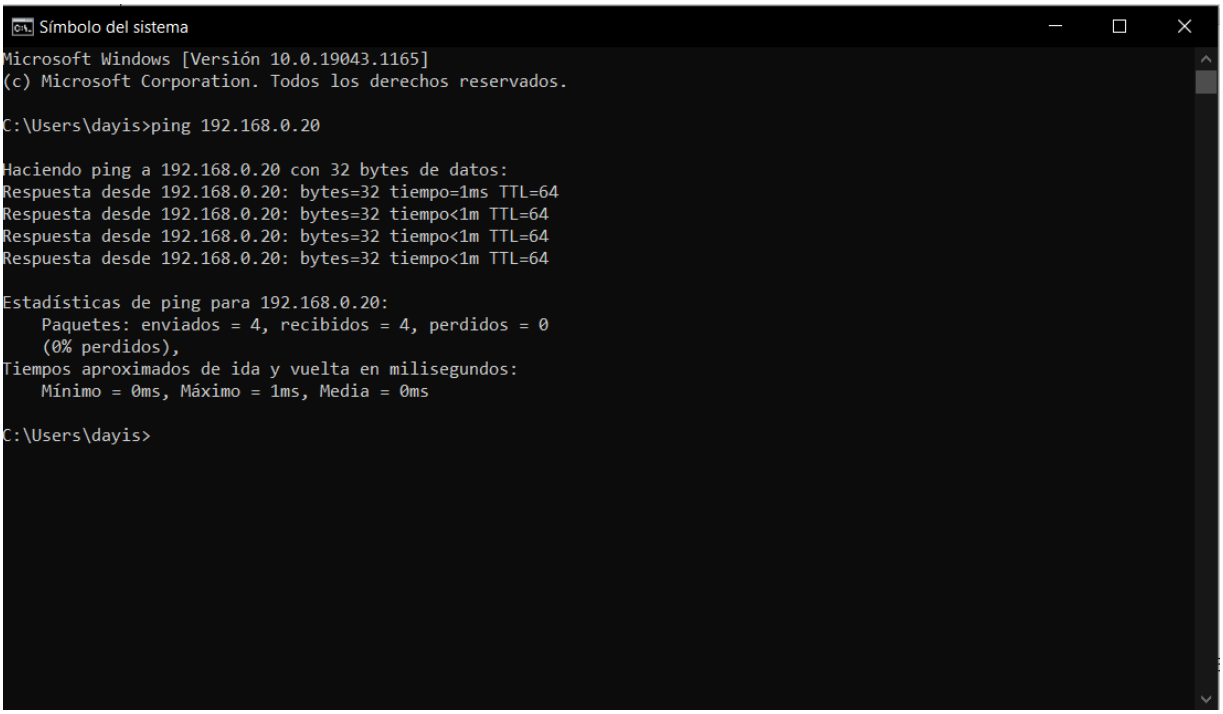
Una vez cumplido todas las indicaciones para el enlace entre la computadora y la controladora, se procede a verificar si existe envío y recepción de datos, realizando los siguientes pasos:

1. Ir al buscador de Windows y escirbir la palabra cmd.
2. Se direcciona a la lista de comandos de Windows

3. Escribir el siguiente comando ping 192.168.0.20, si la conexión es correcta se tiene lo que indica la figura 83, con esto se procede a realizar la comunicación con el software Matlab.

Figura 83

CMD, ping a la IP 192.168.0.20



En el caso de que no se recibiera el mensaje como se muestra en la Figura 83, se procede a establecer una nueva IP estática para la computadora del usuario, para ello se realiza lo siguiente:

1. Ir a el centro de redes y recursos compartidos, seleccionar RED ETHERNET e ir a las propiedades de la misma.
2. Dar doble clic en el protocolo TCP/IPV4
3. Seleccionar “Usar la siguiente dirección IP” y se procede a ingresar los parámetros dados a continuación:

Dirección IP: 192.168.0.50

Máscara de Subred: 255.255.255.0

Puerta de Enlace predeterminada: 162.168.0.1

5.3.2 Software Matlab

Para poder establecer un envió de datos en el software de Matlab con protocolo de comunicación TCP / IP es necesario describir los siguientes comandos que permiten dicha apertura.

Figura 84

Comandos TCP/IP Matlab

```
%Creacion de la variables en la cual almacenaremos el objeto
%de la comunicacion TCP/IP
syms t;
t=tcpip("192.168.0.20",10001);
%Permite abrir el puerto de comunicación/IMPORTANTE/
fopen(t);
```

Nota. En el parámetro del comando tcpip se describe la IP a la cual se quiere enlazar y el puerto, dichos parámetros fueron mencionados en la sección 5.3.1

En el programa se realiza la lectura del archivo .txt generado por el simulador, cabe recalcar que el archivo contiene una lista de posiciones de las junturas del brazo robótico, estas posiciones representan la trayectoria descrita en el simulador.

5.3.2.1 Comandos de comunicación para la controladora CR1D 700

Para enviar información de manera correcta a la controladora desde el software es necesario hacerlo mediante una instrucción, aquella es enviada y recibida en código ASCII, para ello la sintaxis correcta de definirla mediante software Matlab es:

$$fwrite(t, 1; 1; \quad 'Instrucción' \quad)$$

Dónde:

- La variable t representa el puerto de comunicación.
- El parámetro 1;1 representa que se está enviando una instrucción de movimiento.
- ‘Instrucción’, define el comando que quiere que realice la controladora.

La controladora interpreta comandos de MELFA BASIC V o también lenguaje COSIROP, los utilizados en el desarrollo de este trabajo se muestran en la siguiente tabla.

Tabla 10

Comandos de comunicación con la controladora CR1D 700

INSTRUCCIÓN	OPERACION
OPEN=USERTOOL	Da la apertura hacia la controladora
CNTLON	Inicializa el movimiento del brazo, indica que se enviará un dato de control
CNTLOFF	Finaliza el movimiento del brazo, indica que se cierra el envío de datos de control
SRVON	Enciende los servomotores
SRVOFF	Apaga los servomotores
RSTALRM	Resetea las alarmas
OVRD=30.0	Indica el porcentaje de la velocidad al que ejecutará los movimientos el brazo, en este caso es 30%

INSTRUCCIÓN	OPERACION
EXECJCOSIROP= (J1, J2, J3, J4, J5, J6)	Indica los ángulos en grados al que se desea llegar de cada junta

En la Figura 85 se detallan los comandos para leer el archivo .txt con la función importdata, y luego se inicializa al brazo robótico con su respectivo encendido de servomotores y resteo de alarmas, posterior a eso se le envía a una posición HOME determinada.

Figura 85

Inicialización del brazo e importación del archivo .txt

```
%Importación del archivo .txt
m=importdata('E:\TESIS\PC\PC\Articulaciones.txt');
%Obtener numeros de datos recibidos
[f,c]=size(m);
numdatos=f;
%Comandos enviados a la controladora CR1D
fprintf(t,'1;1;OPEN=USERTOOL');
pause(0.25)
fprintf(t,'1;1;CNTLON');
pause(0.25)
fwrite(t,'1;1;SRVON')
pause(0.25)
fwrite(t,'1;1;RSTALRM')
pause(2)
fwrite(t,'1;1;OVRD=30.0')
pause(2)
%Enviar a la posición de HOME
fwrite(t,'1;1;EXECJCOSIROP= (0.00,0.00,90.00,0.00,0.00,0.00)')
pause(4)
fwrite(t,'1;1;EXECMOV JCOSIROP')
pause(2)
```

Se especifica la dirección de la carpeta donde se encuentra el archivo .txt

En la Figura 86 se muestran los comandos para la ejecución de los datos recibidos del archivo .txt, es decir se envían los valores de las juntas en cada punto determinado de la trayectoria realizada.

Figura 86

Comandos para enviar las juntas J1, J2, J3, J4, J5, J6 en cada posición

```
%Ejecutar los movimientos de todos los datos recibidos, através del envío
%de las juntas
for i=1:numdatos

    Punto=m(i,:);

    J1=Punto(1,1);
    J2=Punto(1,2);
    J3=Punto(1,3);
    J4=Punto(1,4);
    J5=Punto(1,5);
    J6=Punto(1,6);
    i=i+1;


    fwrite(t,['1;1;EXECJCOSIROP = ( ' num2str(J1) ' , ' num2str(J2) ' , ' num2str(J3) ' ,'
    num2str(J4) ' , ' num2str(J5) ' , ' num2str(J6) ' )'])
    fwrite(t,'1;1;EXECMOV JCOSIROP')
    pause(2)

end
```

En la Figura 87 se detallan los comandos que envían de nuevo a la posición de Home, después de haberse realizado toda la trayectoria, posterior a eso se cierra la comunicación con la controladora.

Figura 87

Finalización de la comunicación con la controladora CR1D 700

```
% Enviar de nuevo al HOME
fwrite(t,'1;1;EXECJCOSIROP = (0.00,0.00,90.00,0.00,0.00,0.00)')
pause(2)
fwrite(t,'1;1;EXECMOV JCOSIROP')
pause(2)
%Finalizar comunicación
fwrite(t,'1;1;SRVOFF')
fprintf(t,'1;1;CNTLOFF');
```

Capítulo VI

6. Pruebas y resultados

En este capítulo se detalla las pruebas realizas para analizar el correcto funcionamiento del simulador virtual en función de las trayectorias generadas analizando la precisión y eficiencia que posee, y, a su vez la validación de hipótesis referente al proceso de aprendizaje mediante la implementación del sistema de operación off – line con los estudiantes, dónde se mide el nivel de conocimiento adquirido después del uso del mismo.

6.1 Precisión y eficiencia

Para el análisis del correcto funcionamiento del sistema de operación off- line del brazo robótico RV 2SDB frente al funcionamiento del brazo físico es necesario validar dos variables fundamentales, precisión y eficiencia, a partir de las trayectorias generadas y midiendo los tiempos en llegar a dichos puntos respectivamente.

Para el análisis de precisión se realiza el siguiente procedimiento:

1. Realizar diferentes trayectorias correspondientes a los tipos de movimientos presentados en el simulador LIN y CIRC. PTP no puede ser analizado ya que cada controladora busca la trayectoria más rápida por ende no será la misma para comparar.
2. De cada trayectoria, el simulador ejecuta 5 archivos generados diferentes, equivalente se realiza la misma trayectoria programada en el Teach Pendant del brazo físico RV-2SDB.
3. Generar la trayectoria realizada en el Teach Pendant y graficarla con un marcador enlazado a la herramienta final sobre un pizarrón designado.

4. Ejecutar las cinco trayectorias formadas de cada movimiento del simulador, tomar los puntos x, y, z de la trayectoria que se van generando en la controladora y posterior a eso dibujarlos en un software que permita visualizar la gráfica generada.
5. Tomar medidas específicas que permitan comparar lo realizado por el simulador y lo realizado por la propia controladora del brazo RV 2SDB.
6. Realizar validación de la hipótesis mediante la prueba estadística $t - student$.

Para el análisis de eficiencia se realiza el siguiente procedimiento:

1. Determinar 10 puntos diferentes en el simulador con el movimiento tipo PTP, para cada punto se genera un archivo .txt.
2. Se generan los 10 puntos en el brazo físico y se miden los tiempos para llegar a cada punto.
3. Se realizan los mismos puntos dentro de la programación de la controladora del brazo físico con el tipo de movimiento punto a punto.
4. Se ejecuta 10 puntos programados y se miden los tiempos en llegar de cada uno.
5. Se aplica el análisis estadística t – student para dos muestras.

6.1.1 Movimiento tipo LIN

Este tipo de movimiento genera una trayectoria en línea recta, para ello se efectuó una línea de 200 mm, a continuación, el procedimiento realizado:

1. Ejecutar una línea en el simulador, se procede a tomar cinco muestras de la misma trayectoria en archivos .txt diferentes.

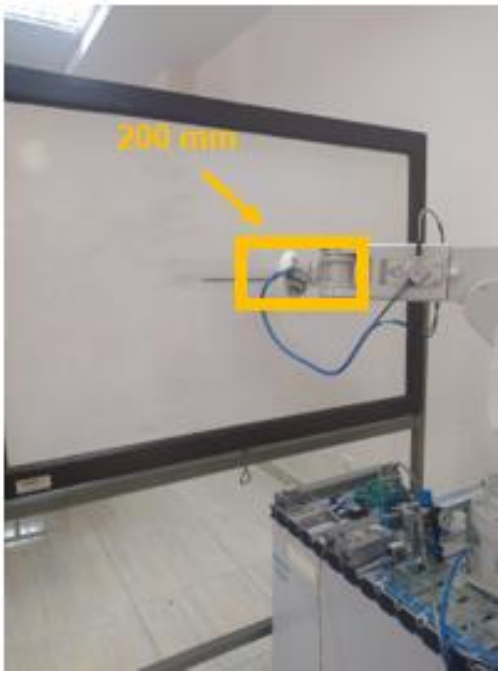
Figura 88

Generación de una línea de 200 mm en el simulador.

2. Se realiza la misma trayectoria programada directamente en el Teach Pendant del brazo físico.

Figura 89

Comandos para enviar las juntas J1, J2, J3, J4, J5, J6 en cada posición



3. Siendo un comando propio de la controladora la longitud de dicha línea es de 200 mm.
4. Se procede a ejecutar las trayectorias realizadas por el simulador en el brazo físico, se toma todos los puntos de posición “x, y, z” que genera la controladora de dicha trayectoria, esto se puede visualizar en el Teach Pendant, para dicha trayectoria se generaron 6 puntos.

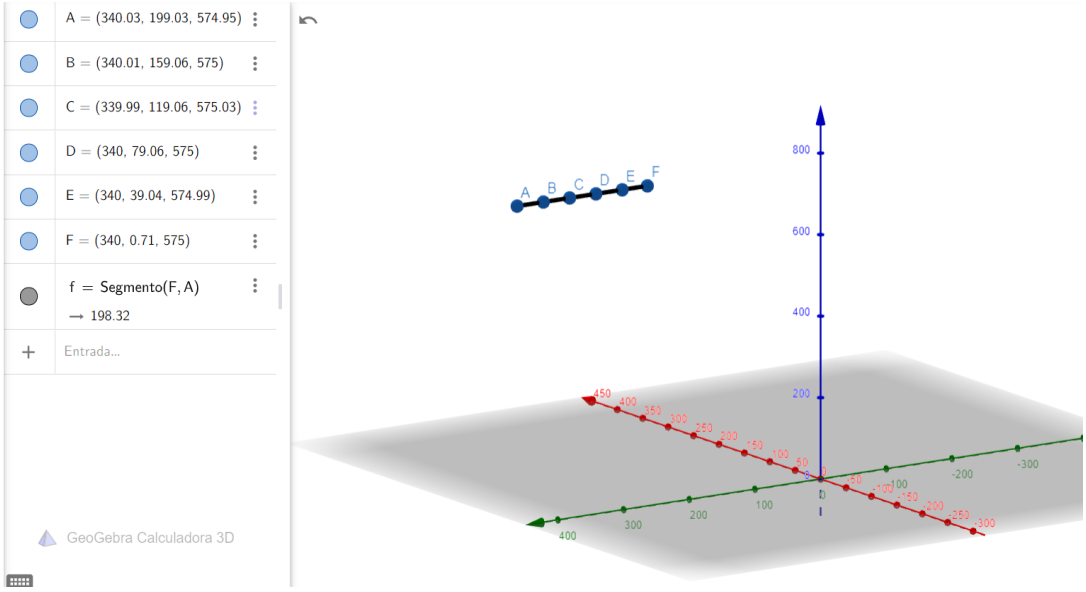
Figura 90

Datos obtenidos de la controladora CR1D 700

5. Dibujar dichos puntos en una graficadora 3D para visualizar la gráfica y medir la dimensión de la línea.

Figura 91

Gráfica de los puntos obtenidos de la controladora para una línea recta



Nota: La dimensión de la línea es de 198.32 mm.

6. Se realiza el mismo procedimiento para todas las muestras. En la tabla 11 se muestra las mediciones obtenidas de las pruebas.

Tabla 11

Pruebas de medición de una trayectoria lineal de 200 mm

Prueba N°	Longitud $X[mm]$
1	198.32
2	197.62
3	200.77
4	200.77
5	198.45

A continuación, se analiza la validación de la hipótesis respecto a la precisión del sistema de operación off – line del brazo RV 2SDB con el método estadístico $t - student$.

1. Definición de la hipótesis

Se describe la hipótesis nula y la hipótesis alternativa según la precisión.

$$H_o: \mu = 200(Medida\ de\ la\ línea\ programada\ en\ el\ robot\ físico)$$

$$H_1: \mu \neq 200(Medida\ de\ la\ línea\ programada\ en\ el\ robot\ físico)$$

H_o No existen diferencias significativas entre el promedio de valores obtenidos en las pruebas de medición de una trayectoria lineal generada por el simulador no inmersivo y la programada en la controladora del brazo físico.

H_1 Existen diferencias significativas entre el promedio de valores obtenidos en las pruebas de medición de una trayectoria lineal generada por el simulador no inmersivo y la programada en la controladora del brazo físico.

2. Ejecución de los cálculos necesarios para el análisis.

Cálculo de la desviación estándar y promedio para los datos obtenidos.

Muestra

$$n = 5$$

Sumatoria de Longitud X

$$\sum X = 995.93$$

Cálculo de la media

$$\bar{\boldsymbol{x}} = \frac{\sum \boldsymbol{X}}{\boldsymbol{n}} \tag{46}$$

$$\bar{x} = \frac{995.93}{5}$$

$$\bar{x} = 199.19$$

Cálculo de la desviación estándar, para ello es necesaria tener la siguiente tabla.

Tabla 12

Datos para el cálculo de la desviación estándar

x_i	$x_i - \bar{x}$	$(x_i - \bar{x})^2$
198.32	−0.87	0.76
197.62	−1.57	2.46
200.77	1.58	2.50
200.77	1.58	2.50
198.45	0.74	0.55

$$\boldsymbol{s^2} = \frac{\sum(\boldsymbol{x_i} - \bar{\boldsymbol{x}})^{\boldsymbol{2}}}{\boldsymbol{n - 1}} \tag{49}$$

$$s^2 = \frac{8.7622}{4}$$

$$s^2 = 2.19055$$

$$s = \sqrt{2.19055}$$

$$s = 1.48$$

3. Establecimiento de la zona de rechazo H_o

Grados de libertad

$$g.l = número\ de\ muestras - 1$$

$$g.l = 5 - 1$$

$$g.l = 4$$

Generalmente el nivel de significancia se define en $\propto = 0.05$, lo que significa que la probabilidad de analizar las diferencias en los datos es de solo el 5%.

Se define el valor crítico para la prueba, dicho dato se lo encuentra en la tabla de distribución t-student de dos colas con grados de libertad.

$$t_{vc} = \pm 2.776$$

4. Cálculo de la prueba t-student

$$\boldsymbol{t = \frac{\bar{x} - \mu}{s\bar{x}}} \tag{48}$$

Cálculo de $s\bar{x}$

$$s\bar{x} = \frac{1.48}{\sqrt{5}} = 0.66$$

$$\boldsymbol{s\bar{x} = \frac{s}{\sqrt{n}}} \tag{50}$$

Cálculo de t

$$t = \frac{199.19 - 200}{0.66}$$

$$t = -1.2272$$

5. Decisión de aceptación o rechazo de H_0

Para rechazar o aceptar la hipótesis se verifica la siguiente comparación:

$$t > t_{vc}, \qquad Rechaza\ H_o$$

$$t < t_{vc}, \qquad Acepta\ H_o$$

Comparación:

$$t = -1.2272; \quad t_{vc} = -2.776$$

$t < t_{vc}$, se acepta H_o

Mediante la prueba t-student de la muestra obtenida, se concluye que no se encontraron diferencias significativas entre el promedio de valores obtenidos en las pruebas de medición de una trayectoria lineal generada por el simulador no inmersivo y la programada en la controladora del brazo físico. Haciendo un promedio, la trayectoria lineal generada por el simulador ($\bar{x} = 199.19$) no mide menos que la programada por la controladora del brazo físico ($\mu = 200$).

6.1.2 Movimiento tipo CIRC

Este tipo de movimiento genera un arco de diferente diámetro, para ello se efectuó una línea de 200 mm, a continuación, el procedimiento realizado:

1. Ejecutar un arco en el simulador, se procede a tomar cinco muestras de la misma trayectoria en archivos .txt diferentes.

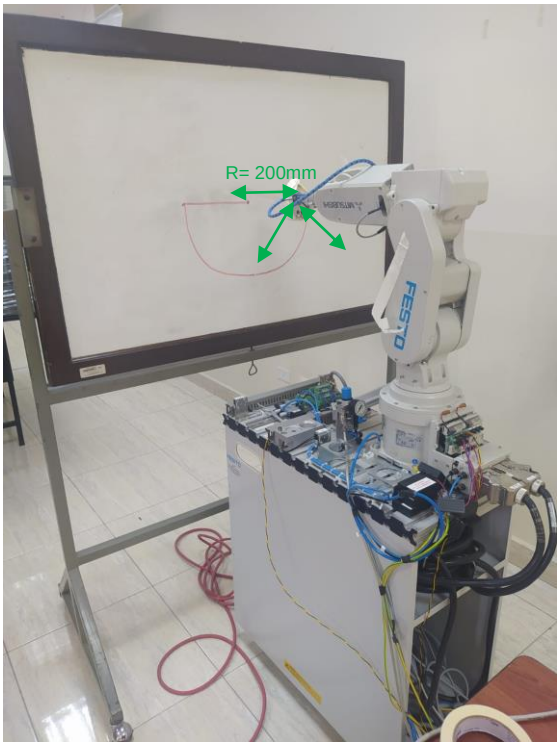
Figura 92

Generación del arco de 200 mm de radio en el simulador.

2. Se realiza la misma trayectoria programada directamente en el Teach Pendant del brazo físico.

Figura 93

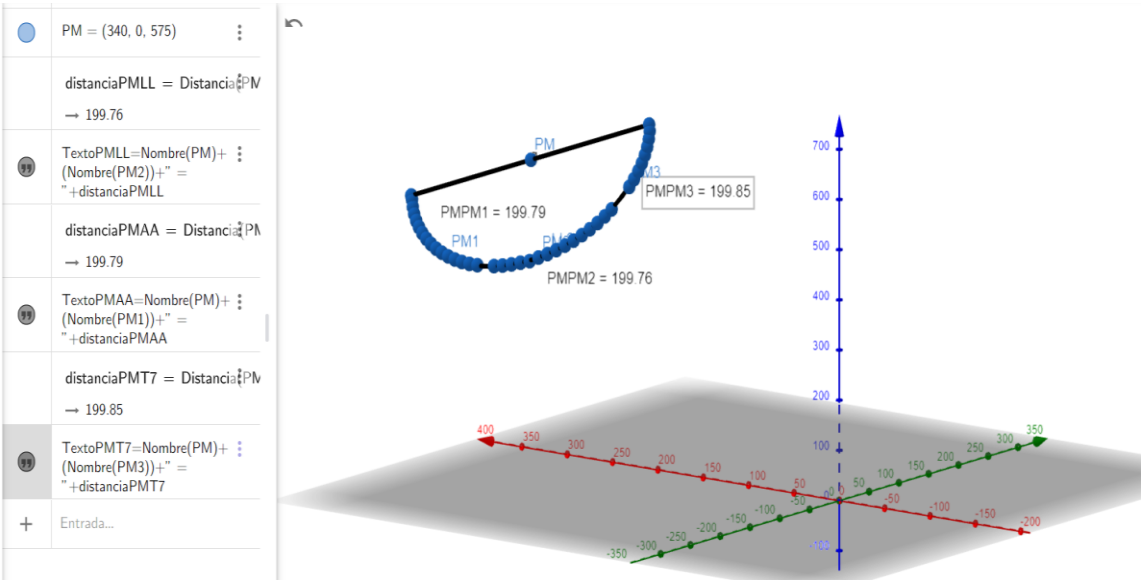
Generación del arco de 200 mm de radio programada en el brazo físico.



3. Siendo un comando propio de la controladora los datos desde el centro del arco hacia diferentes puntos de la trayectoria circular son de 200 mm.
4. Se procede a ejecutar las trayectorias realizadas por el simulador en el brazo físico, se toma todos los puntos de posición “x, y, z” que genera la controladora de dicha trayectoria, esto se puede visualizar en el Teach Pendant, para dicha trayectoria se generaron aproximadamente entre 39 - 42 puntos.
5. Dibujar dichos puntos en una graficadora 3D para visualizar la gráfica y medir la dimensión del radio del arco, y tres medidas desde el centro hacia tres puntos diferentes de la trayectoria.

Figura 94

Gráfica de los puntos obtenidos de la controladora para un arco



Nota: La dimensión del radio es de es de 199.74 mm, las tres medidas hacia puntos diferentes de la trayectoria son 199.79 mm, 199.85 mm y 199.76 mm.

6. Se realiza el mismo procedimiento para todas las muestras.

En la tabla 13 se muestra las mediciones obtenidas de las pruebas.

Tabla 13

Pruebas de medición de una trayectoria circular de radio de 200mm

Prueba N°	Punto 1	Punto 2	Punto 3	Radio	Promedio x
	$[mm]$	$[mm]$	$[mm]$	$[mm]$	$[mm]$
1	199.79	199.85	199.76	199.74	198.78
2	199.42	199.28	199.65	197.66	199
3	199.58	199.46	200.16	199.73	199.73
4	199.82	199.99	200.95	199.75	200.12
5	199.2	199.35	200.33	200.52	199.85

A continuación, se analiza la validación de la hipótesis respecto a la precisión de una trayectoria circular del sistema de operación off – line del brazo RV 2SDB con el método estadístico $t - student$.

1. Definición de la hipótesis

Se describe la hipótesis nula y la hipótesis alternativa según la precisión.

$$H_o: \mu = 200(Medida\ del\ radio\ del\ arco\ programado\ en\ el\ robot\ físico)$$

$$H_1: \mu \neq 200(Medida\ del\ radio\ del\ arco\ programado\ \ en\ el\ robot\ físico)$$

H_o No existen diferencias significativas entre el promedio de valores obtenidos en las pruebas de medición de una trayectoria circular generada por el simulador no inmersivo y la programada en la controladora del brazo físico.

H_1 Existen diferencias significativas entre el promedio de valores obtenidos en las pruebas de medición de una trayectoria circular generada por el simulador no inmersivo y la programada en la controladora del brazo físico.

2. Ejecución de los cálculos necesarios para el análisis.

Cálculo de la desviación estándar y promedio para los datos obtenidos.

Muestra

$$n = 5$$

Sumatoria de Radio promedio X

$$\sum X = 997.48$$

Cálculo de la media

$$\bar{x} = \frac{997.48}{5}$$

$$\bar{x} = 199.49$$

Cálculo de la desviación estándar, para ello es necesaria tener la siguiente tabla.

Tabla 14

Datos para el cálculo de la desviación estándar de una trayectoria circular

x_i	$x_i - \bar{x}$	$(x_i - \bar{x})^2$
198.78	−0.71	0.50
199	−0.49	0.24
200.12	0.63	0.40
199.85	0.36	0.13
199.73	0.24	0.06

$$s^2 = \frac{\sum(x_i - \bar{x})^2}{n - 1}$$

$$s^2 = \frac{1.33}{4}$$

$$s^2 = 0.3325$$

$$s = \sqrt{0.3325}$$

$$s = 0.58$$

3. Establecimiento de la zona de rechazo H_o

Grados de libertad

$$g.l = número\ de\ muestras - 1$$

$$g.l = 5 - 1$$

$$g.l = 4$$

Generalmente el nivel de significancia se define en $\propto = 0.05$, lo que significa que la probabilidad de analizar las diferencias en los datos es de solo el 5%.

Se define el valor crítico para la prueba, dicho dato se lo encuentra en la tabla de distribución t-student de dos colas con grados de libertad.

$$t_{vc} = \pm 2.776$$

4. Cálculo de la prueba t-student

$$t = \frac{\bar{x} - \mu}{s\bar{x}}$$

Cálculo de $s\bar{x}$

$$s\bar{x} = \frac{s}{\sqrt{n}}$$

$$s\bar{x} = \frac{0.58}{\sqrt{5}} = 0.26$$

Cálculo de t

$$t = \frac{199.49 - 200}{0.66}$$

$$t = -0.77$$

5. Decisión de aceptación o rechazo de H_0

Para rechazar o aceptar la hipótesis se verifica la siguiente comparación:

$$t > t_{vc}, \qquad Rechaza\ H_o$$

$$t < t_{vc}, \qquad Acepta\ H_o$$

Comparación:

$$t = -0.77; \quad t_{vc} = -2.776$$

$t < t_{vc}$, se acepta H_o

Mediante la prueba t-student de la muestra obtenida, se concluye que no se encontraron diferencias significativas entre el promedio de valores obtenidos en las pruebas de medición de una trayectoria circular generada por el simulador no inmersivo y la programada en la controladora del brazo físico. Haciendo un promedio, la trayectoria lineal generada por el simulador ($\bar{x} = 199.49$) no mide menos que la programada por la controladora del brazo físico ($\mu = 200$).

6.1.3 Movimiento tipo PTP

Se utiliza el método estadístico t – student para analizar las dos muestras tomadas de los tiempos y determinar si existe diferencias significativas entre las mismas, para ello en la Tabla 15 se muestran los datos obtenidos.

Tabla 15

Tiempos de los puntos ejecutados con el comando PTP

N°	Tiempos de los puntos generados por el simulador	Tiempos de los puntos programados en la controladora del brazo
	[seg]	[seg]
1	1.63	1.86
2	1.53	1.36
3	2.6	3.4
4	1.73	2.3
5	1.6	2.26

N°	Tiempos de los puntos generados por el simulador	Tiempos de los puntos programados en la controladora del brazo
6	1.53	2.06
7	2.3	1.76
8	1.33	1.5
9	1.06	1.03
10	2.1	2.33

1. Planteamiento de la hipótesis nula H_0 y una hipótesis alternativa H_1

H_0: No existen diferencias significativas entre el promedio de los tiempos obtenidos entre los puntos generados por el simulador y los programados por la controladora del brazo.

H_1: Existen diferencias significativas entre el promedio de los tiempos obtenidos entre los puntos generados por el simulador y los programados por la controladora del brazo.

2. Ejecución de los cálculos necesarios para el análisis.

Para efectuar la prueba t, es necesario tener los siguientes parámetros para cada grupo que realizó el test.

- n_1: Cantidad de puntos generados por el simulador virtual.
- n_2: Cantidad de puntos generados a partir de la programación de la controladora del brazo.
- $\overline{x_1}$: Es el promedio de los tiempos en llegar a cada punto generado por el simulador virtual.

- $\overline{x_2}$: Es el promedio de los tiempos en llegar a cada punto programado desde la controladora del brazo.
- ${S_1}^2$: Es la varianza de los tiempos en llegar a cada punto generado por el simulador virtual.
- ${S_2}^2$: Es la varianza de los tiempos en llegar a cada punto programado desde la controladora del brazo.

El procedimiento para calcular dichos parámetros es similar al presentado en ítems anteriores, en la tabla 16 se muestran los respectivos valores.

Tabla 16

Valores necesarios para el análisis t – student respecto a los tiempos.

n_1	n_2	$\overline{x_1}$	$\overline{x_2}$	${S_1}^2$	${S_2}^2$
10	10	1.74	1.98	0.2148	0.43

3. Establecimiento de la zona de rechazo H_o

Grados de libertad

$$g.l = n_1 + n_2 - 2$$

$$g.l = 10 + 10 - 2$$

$$g.l = 18$$

Generalmente el nivel de significancia se define en $\propto = 0.05$.

Se define el valor crítico para la prueba, dicho dato se lo encuentra en la tabla de distribución t-student de dos colas con grados de libertad.

$$t_{vc} = \pm 2.10$$

4. Cálculo de la prueba t-student

$$\boldsymbol{t = \frac{(\overline{x_1} - \overline{x_2})}{s\sqrt{{n_1}^{-1} + {n_2}^{-1}}}} \quad (50)$$

Donde s es la desviación estándar combinada y se calcula de la siguiente manera:

$$\boldsymbol{s^2 = \frac{(n_1 - 1){S_1}^2 + (n_2 - 1){S_2}^2}{n_1 + n_2 - 2}} \quad (51)$$

$$s^2 = 0.3224$$

$$s = \sqrt{s^2} = \sqrt{0.3224}$$

$$s = 0.567$$

Cálculo de t

$$t = \frac{1.74 - 1.98}{0.567\sqrt{0.1 + 0.1}}$$

$$t = -0.94$$

5. Decisión de aceptación o rechazo de H_0

Para rechazar o aceptar la hipótesis se verifica la siguiente comparación:

$$t > t_{vc}, \qquad Rechaza\ H_o$$

$$t < t_{vc}, \qquad Acepta\ H_o$$

Comparación:

$$t = -0.94; \quad t_{vc} = -2.10$$

$t < t_{vc}$, se acepta H_o

Mediante la prueba t-student de las dos muestras obtenidas, se concluye que no se encontraron diferencias significativas entre el promedio de los tiempos obtenidos

entre los puntos generados por el simulador y los programados por la controladora del brazo.

6.2 Validación de la hipótesis

Hipótesis planteada:

¿El diseño e implementación de un sistema de operación off-line del brazo robótico Mitsubishi RV-2SDB a través de realidad virtual no inmersiva contribuirá en el proceso de aprendizaje de robótica industrial en el Laboratorio de Mecatrónica de la Universidad de las Fuerzas Armadas ESPE Sede Latacunga?

Se debe tener variable dependiente e independiente en el proceso de validación de hipótesis, se muestran a continuación:

Variables de la Investigación

- Variable Independiente: Sistema de realidad virtual no inmersiva con operación off-line del brazo robótico Mitsubishi RV-2SDB
- Variable dependiente: Contribución en el proceso de aprendizaje de robótica industrial

Se utiliza el método estadístico t – student para analizar la validación de hipótesis mediante la implementación de un test donde se mide el nivel de conocimiento de robótica industrial en función de conceptos generales, dicho formulario es aplicado a un grupo de estudiantes que aún no reciben la materia de robótica industrial, se segmenta en dos grupos, el primero es aquel que no recibe el sistema de operación off – line del brazo a través de realidad virtual y el segundo si logra hacer uso de él.

Los grupos están conformados por 10 personas y las notas se visualizan en la tabla 17.

Tabla 17

Notas de los estudiantes que realizaron el test.

N°	Notas sin usar el simulador 10/10	Notas haciendo uso del simulador 10/10
1	3	8
2	2	7
3	4	8
4	3	8
5	5	7
6	5	7
7	4	8
8	5	8
9	4	8
10	2	8

6. Planteamiento de la hipótesis nula H_0 y una hipótesis alternativa H_1

H_0: La utilización del sistema de operación off – line a través de realidad virtual no inmersiva no permitió contribuir en el proceso de aprendizaje de robótica industrial en los estudiantes.

H_1: La utilización del sistema de operación off – line a través de realidad virtual no inmersiva permitió contribuir en el proceso de aprendizaje de robótica industrial en los estudiantes.

7. Ejecución de los cálculos necesarios para el análisis.

Para efectuar la prueba t, es necesario tener los siguientes parámetros para cada grupo que realizó el test.

- n_1: Cantidad de pruebas realizadas por el grupo de estudiantes que utilizó el simulador.
- n_2: Cantidad de pruebas realizadas por el grupo de estudiantes que no utilizó el simulador.
- $\overline{x_1}$: Es el promedio de las calificaciones obtenidas de las pruebas de los estudiantes que utilizaron el simulador.
- $\overline{x_2}$: Es el promedio de las calificaciones obtenidas de las pruebas de los estudiantes que no utilizaron el simulador.
- ${S_1}^2$: Es la varianza de las calificaciones obtenidas de las pruebas de los estudiantes que utilizaron el simulador.
- ${S_2}^2$: Es la varianza de las calificaciones obtenidas de las pruebas de los estudiantes que no utilizaron el simulador.

El procedimiento para calcular dichos parámetros es similar al presentado en ítems anteriores, en la Tabla 18 se muestran los respectivos valores.

Tabla 18

Valores necesarios para el análisis t – student

n_1	n_2	$\overline{x_1}$	$\overline{x_2}$	${S_1}^2$	${S_2}^2$
10	10	7.7	3.7	0.23	1.34

8. Establecimiento de la zona de rechazo H_o

Grados de libertad

$$g.l = n_1 + n_2 - 2$$

$$g.l = 10 + 10 - 2$$

$$g.l = 18$$

Generalmente el nivel de significancia se define en $\propto = 0.05$.

Se define el valor crítico para la prueba, dicho dato se lo encuentra en la tabla de distribución t-student de dos colas con grados de libertad.

$$t_{vc} = \pm 2.10$$

9. Cálculo de la prueba t-student

$$t = \frac{(\overline{x_1} - \overline{x_2})}{s\sqrt{{n_1}^{-1} + {n_2}^{-1}}}$$

Donde s es la desviación estándar combinada y se calcula de la siguiente manera:

$$s^2 = \frac{(n_1 - 1){S_1}^2 + (n_2 - 1){S_2}^2}{n_1 + n_2 - 2}$$

$$s^2 = 0.785$$

$$s = \sqrt{s^2} = \sqrt{0.785}$$

$$s = 0.886$$

Cálculo de t

$$t = \frac{7.7 - 3.7}{0.886\sqrt{0.1 + 0.1}}$$

$$t = -10.09$$

10. Decisión de aceptación o rechazo de H_0

Para rechazar o aceptar la hipótesis se verifica la siguiente comparación:

$$t > t_{vc}, \qquad Rechaza\ H_o$$

$$t < t_{vc}, \qquad Acepta\ H_o$$

Comparación:

$$t = -1.009; \quad t_{vc} = -2.10$$

$t < t_{vc}$, se rechaza H_o

Acorde al análisis realizado, el valor crítico de dos colas de distribución es de 2.10 y el valor estadístico de prueba t es mayor, por ende, se rechaza la hipótesis nula H_o, y se acepta la hipótesis alternativa, comprobando que la hipótesis planteada en la investigación es verdadera, asegurando dicha validación con una fiabilidad de 95%, se afirma que la utilización del sistema de operación off – line a través de realidad virtual no inmersiva permitió contribuir en el proceso de aprendizaje de robótica industrial en los estudiantes.

Capítulo VII

7. Conclusiones y recomendaciones

7.1 Conclusiones

- Se diseñó e implementó un sistema de operación off – line del brazo robótico Mitsubishi Melfa RV – 2SDB que contribuye en el proceso de aprendizaje de robótica industrial, mediante realidad virtual no inmersiva en el Laboratorio de Mecatrónica en la Universidad de las Fuerzas Armadas ESPE Sede Latacunga.
- Se determinó que existen diferentes tipos de movimientos para el control cinemático del robot, sabiendo que para la cinemática directa se empleó los parámetros D-H conjuntamente con la matriz de transformación homogénea que ayudó a determinar los valores de posición y orientación del efector final y para la resolución del problema de la cinemática inversa por medio del método de la Jacobiana inversa.
- Para el modelo cinemático del brazo robótico se estableció un control por el método de la Jacobiana inversa o también conocida como el modelo diferencial de la jacobiana analítica, que considera la relación entre las pequeñas variaciones de las velocidades articulares con las velocidades de la localización del extremo del robot, en un intervalo de muestra, obteniendo una matriz no cuadrada que se estableció con el método de la pseudoinversa.
- En la investigación metódica del estado del arte del Robot Mitsubishi Melfa RV-2SDB se encontró que la página oficial de Mitsubishi Electrical contiene un apartado sobre modelos de robots desarrollados en formatos .dxf y. step para desarrolladores de manera gratuita, cosa que no fue necesario realizar diseño y modelado del robot. Sin embargo, el diseño sobre la estructura de la mesa y el aula virtual fue desarrollado por las autoras mediante los softwares como solidworks y sketch up,

respectivamente. Lo cual permitió establecer una inmersión visual al simulador virtual.

- Como dispositivo de mando se escogió el Mando del WII motion Plus esto se calificó por el método cualitativo por puntos donde el peso que conllevó al resultado fue el costo del dispositivo, la autonomía, su portabilidad, la rapidez de comunicación y al obtener el estudio de la compatibilidad con el motor de juego Unity, desarrollado por una librería externa. De igual manera se empleó otro dispositivo externo para la manipulación del brazo robótico que fue el teclado ya que también posee una librería interna en Unity.
- Los softwares 3D poseen sistemas de referencia, pero no todos contienen la misma arquitectura, esto es debido a que no se encuentra definido un sistema de referencia fijo y se basan en el criterio del desarrollador. Unity como motor de juego se guían mediante la regla de la mano izquierda, donde el eje +Y de la coordenada mundial se extiende hacia arriba, en consecuencia, matemáticamente al modelar el brazo robótico Mitsubishi Melfa RV– 2SDB trabaja con el sistema diestro que determina direcciones vectoriales en planos cartesianos para los parámetros D-H y definir la dirección de cada articulación cumpliendo con el criterio del producto vectorial. Por esta razón se procede a migrar del sistema zurdo al diestro. Es decir X_z, Y_z, Z_z, pasan a X_z, Z_z, Y_z.
- Para salvaguardar la integridad del robot y evitar choques se estableció que mediante el proceso de ejecución ciertos puntos generan singularidad donde limita el movimiento del manipulador aunque las configuraciones se encuentran dentro de los rangos de operación de las articulaciones del robot, por ello en el simulador se definen rangos de operación, como resultado las articulaciones varían entre: Base entre (-120° a 120°), Hombro (-90° a 40°), Brazo (65° a 140°), Codo (-180° a 180°),

Antebrazo (-80° a 80°), Muñeca (-180° a 180°). Con estos datos se logra especificar un rango aproximado en las posiciones del robot, tomando en cuenta las dimensiones de sus eslabones con X (-210 a 400) mm, Y (-350 a 350) mm y Z (-140 a 770) mm.

- Se implementó el sistema de operación off – line mediante comunicación TCP / IP de la computadora y la controladora CRD1-700 del brazo robótico, se envió los datos adquiridos del simulador de cada trayectoria generada en función de sus articulaciones J1,J2,J3,J4,J5,J6, con esto se asegura ejecutarlas dentro de los límites establecidos de cada una de ellas y evitar puntos de singularidad, las instrucciones enviadas a la controladora están generadas mediante código ASCII en el software que enlaza dichos equipos.
- Las pruebas realizadas en el capítulo 6 demuestran el óptimo funcionamiento del sistema de operación off – line referente a la precisión que describen las trayectorias, ya sean circulares o lineales, mediante la ejecución de varias pruebas generadas desde el simulador virtual no inmersivo y programadas en la propia controladora del brazo físico, las diferencias de las mismas no fueron significativas, y, por otro apartado tiene una eficiencia similar a la propia controladora del brazo robótico físico RV-2SDB ya que no se visualizó diferencias significativas en los tiempos ejecutados en trayectorias punto a punto.
- Las pruebas aplicadas a los estudiantes de octavo nivel de la carrera de Ingeniería Mecatrónica comprueban que el sistema de operación off – line contribuyó en su proceso de aprendizaje de robótica industrial debido a la interfaz gráfica interactiva que presenta el simulador de realidad virtual no inmersivo, y, a su vez, muestra conceptos fundamentales que se aplican en la enseñanza de la temática en general, dando una alternativa innovadora de estudio para los estudiantes.

7.2 Recomendaciones

- Se recomienda trabajar en el simulador dentro de los límites establecidos, es decir el usuario no debe tratar de llegar hasta el límite máximo tanto en cinemática directa como en inversa, ya que podría llevar a puntos de singularidad y dar error en el control.
- Al momento de hacer la comunicación con el brazo robótico físico es necesario desactivar los paros de emergencia tanto del teach pendant como de la controladora CR1D 700 ya que podrían saltar las alarmas y los datos no puedan ser enviados como tal.
- Al momento de ejecutar los comandos en el brazo físico RV 2SDB no se debe estar dentro del área de trabajo, por eso es necesario estar al menos a 1,5 metros de distancia alrededor del mismo, para precautelar la seguridad del usuario y del robot.
- Se recomienda que, al momento de comunicarse con el WMP a la PC, reconozca la configuración bluetooth como la Figura 65, para iniciar la comunicación en Unity se presiona el botón mando, cerrar, abrir la aplicación nuevamente y presionar el botón mando para conseguir una configuración efectiva.
- Para seguir desarrollando este trabajo se recomienda realizar el estudio para el control de orientación del efector final en la cinemática inversa por medio de cuaterniones que independizan las rotaciones de los ejes yaw, pitch y roll, eliminando el efecto Gimbal Lock.
- Para estudios posteriores Unity ofrece el elemento de componentes físicos, con esto se puede desarrollar sistemas dinámicos del robot manipulador, ingresando parámetros inerciales, gravitacionales, masa, etc, para un mayor realismo en la simulación.

8. Bibliografía

Allcoat, D., & von Mühlenen, A. (2018). Learning in virtual reality: Effects on performance, emotion and engagement. *Research in Learning Technology*, *26*(1063519), 1–13. Recuperado el 17 de febrero de 2021, de https://doi.org/10.25304/rlt.v26.2140

Autodesk. (2021). *New Features In 3ds Max 2021 | 3D Modeling & Rendering Tools | Autodesk*. Recuperado el 17 de febrero de 2021, de https://www.autodesk.com/products/3ds-max/features?support=ADVANCED&plc=3DSMAX&term=1-YEAR&quantity=1

Baca, G. (2010). *Evaluación de Proyectos* (McGRAW-HILL (ed.); 4taEd ed.). Recuperado el 14 de marzo de 2021,https://econforesyproyec.files.wordpress.com/2014/11/evaluacion-de-proyectos-gabriel-baca-urbina-corregido.pdf

Barrientos, A., Peñin, L., Balaguer, C., & Aracil, R. (2007). Fundamentos de Robótica. En McGraw-Hill (Ed.), *Journal of Physics A: Mathematical and Theoretical* (2th ed., Vol. 44, Número 8). Recuperado el 16 de marzo de 2021, https://doi.org/10.1088/1751-8113/44/8/085201

Biagioli, A., Snyder, A., Erdoss, A., & Saffron. (2016, marzo 24). *Unity-Wiimote. Unity3D / C# and a Wii Remote controller.* Recuperado el 12 de junio de 2021, https://github.com/Flafla2/Unity-Wiimote

Bravo, H., Da Vinci's Engine, & GameDevTraum. (2021, junio). *HERENCIA y POLIMORFISMO en programación - C# en Unity*. Recuperado el 12 de junio de 2021, https://davincis-engine.xyz/es/teoria-sobre-programacion/ejemplo-de-herencia-y-polimorfismo-en-programacion-lenguaje-c-en-unity/

Corke, P. (2017). *Robotics, Vision and Control Fundamental Algorithms in MATLAB®* (Springer (ed.)). Recuperado el 12 de junio de 2021, https://doi.org/10.1007/978-3-319-54413-7 ISBN

De Antonio, A., Villalobos, M., & Luna, E. (2000). Cuándo y Cómo usar la Realidad Virtual en la Enseñanza. *Revista de Enseñanza y Tecnología*, 26–36. Recuperado el 12 de enero de 2021, de https://pdfs.semanticscholar.org/97d8/69001b1dffa5c7deaa7e865ff0098595dc5a.pdf

Fernandez, P. (s/f). *Introducción a la programación Melfa Melfa Basic* . Recuperado el 20 de agosto de 2021, de https://www.academia.edu/37034074/Introducción_a_la_programación_Introducción_a_la_programación_Melfa_Melfa_Basic_IV_Basic_IV_MELFA_Robots_MELFA_Robots&as_qdr=y15

Fib. (2008, marzo 13). *Realidad virtual*. Recuperado el 17 de febrero de 2021, de https://www.fib.upc.edu/retro-informatica/avui/realitatvirtual.html

GameDevTraum. (2020a). *Método Update en Unity* . Recuperado el 12 de junio de 2021, https://gamedevtraum.com/es/desarrollo-de-videojuegos/tutoriales-y-soluciones-unity/serie-fundamental-unity/metodo-update-unity/

GameDevTraum. (2020b, diciembre). *Cómo CAMBIAR el SKYBOX en Unity* . Recuperado el 12 de junio de 2021, de https://gamedevtraum.com/es/desarrollo-de-videojuegos/tutoriales-y-soluciones-unity/como-cambiar-el-skybox-en-unity-modificar-cielo/

Gómez, J. (2013, octubre 25). *Guía rápida robot Mitsubishi*. Recuperado el 24 de mayo de 2021, de https://www.slideshare.net/jesusgomez39750/guia-r-

pidarobotmitsubishirv2aj

Guevara, B., & Martínez, A. (2018). *Diseño y desarrollo de un sistema inmersivo de reconocimiento y control de gestos, ostensible por medio de realidad virtual como método de ayuda en la rehabilitación de la capacidad motriz de las extremidades superiores en pacientes con accidente cerebrov* [Universidad de las Fuerzas Armadas ESPE]. Recuperado el 14 de marzo de 2021, http://repositorio.espe.edu.ec/xmlui/handle/21000/14928

Hisour. (s/f). *Simulador de robótica – HiSoUR Arte Cultura Historia*. Recuperado el 17 de febrero de 2021, de https://www.hisour.com/es/robotics-simulator-42971/

HTC Corporation. (s/f). *Acerca de los controladores VIVE [Imagen]*. Recuperado el 14 de marzo de 2021, de https://www.vive.com/mx/support/vive/category_howto/about-the-controllers.html

HTC Corporation. (2018). *Controller HTC VIVE [Imagen]*. Recuperado el 14 de marzo de 2021, https://www.vive.com/eu/accessory/controller2018/?utm_medium=htccom&utm_source=htc&utm_campaign=default_try_vive&cjevent=undefined

iFixit. (2016a). *Oculus Touch Teardown*. Recuperado el 14 de marzo de 2021, https://es.ifixit.com/Desmontaje/Oculus+Touch+Teardown/75109

iFixit. (2016b, marzo 5). *HTC Vive Teardown [Imagen]*. Recuperado el 14 de marzo de 2021, https://es.ifixit.com/Desmontaje/HTC+Vive+Teardown/62213

KittyVector. (2020, julio). *Juego de iconos de joystick y tecnología de gadgets. [Imagen]*. Juego de iconos de joystick y tecnología de gadgets para jugadores. Recuperado el 27 de febrero de 2021, de https://www.freepik.es/vector-premium/juego-iconos-

joystick-tecnologia-gadgets-jugadores-conjunto-ilustraciones-controlador-joysticks-electronicos-video-dispositivos-informaticos-coleccion-consolas-juegos-juegos-digitales-entretenimiento_9505671.htm#page=

Mendoza, M. (2019). *Componentes que conforman un robot manipulador*. 7–9. Recuperado el 05 de abril de 2021, de http://ri.uaemex.mx/bitstream/handle/20.500.11799/108137/secme33422_1.pdf?sequence=1

MITSUBISHI ELECTRIC AUTOMATION. (2010). *RV-2SDB / RV-2SQB Series MELFA [Gráfico]*. 3. Recuperado el 05 de abril de 2021, de https://pdf.directindustry.com/pdf/mitsubishi-electric-automation/rv-2sd-sq-brochure-ver-b/25880-527601.html

Mitsubishi Electric Corporation. (2010, abril 13). *Industrial/Collaborative Robots-MELFA Robots CAD Data Downloads* . Recuperado el 24 de abril de 2021, de https://www.mitsubishielectric.com/fa/download/cad/rbt/robot/index.html

Mora, J., López, M., & Bustillo, M. (2002, septiembre). *SIMULACIÓN EN TIEMPO REAL DE UNA ESTACION DE TRABAJO INDUSTRIAL ROBOTIZADA. - PDF Descargar libre*. 05 de abril de 2021, de Recuperado el 16 de julio de 2021, https://docplayer.es/11464107-Simulacion-en-tiempo-real-de-una-estacion-de-trabajo-industrial-robotizada.html

MSP. (2020, marzo 12). *Acuerdos Ministeriales – Documentos Normativos Coronavirus – Ministerio de Salud Pública*. Recuperado el 19 de enero de 2021, de https://www.salud.gob.ec/acuerdos-ministeriales-documentos-normativos-coronavirus/

Naranjo, X., & Tello, J. (2017). *Diseño e implementación de una pinza flexible basado*

en la tecnología de robótica blanda para manipulación y clasificación de objetos con geometría irregular implementado en el brazo robótico Mitsubishi del laboratorio de Mecatrónica. Universidad de las Fuerzas Armadas ESPE.

Nintendo Wiki Fandom. (s/f). *Wii Remote [Imagen],* . Recuperado el 14 de marzo de 2021, de https://nintendo.fandom.com/es/wiki/Wii_Remote

Oculus VR. (s/f). *Características del Oculus Rift S* . Recuperado el 14 de marzo de 2021, de https://www.oculus.com/rift-s/features/

Otón, S., Martínez, J., & Hilera, J. (2018). *Aplicación de la Realidad Virtual en la enseñanza a través de Internet*. Cuadernos de Documentación Multimedia 8. Recuperado el 16 de marzo de 2021, de https://www.researchgate.net/publication/28076459_Aplicacion_de_la_Realidad_Virtual_en_la_ensenanza_a_traves_de_Internet

Primalshell. (2013, agosto 2). *3D Cartesian Coodinate Handedness [Gráfico]*. Recuperado el 17 de marzo de 2021, de https://upload.wikimedia.org/wikipedia/commons/b/b2/3D_Cartesian_Coodinate_Handedness.jpg

Rodríguez, J. (2009, septiembre 10). *Software de cómputo numérico - KND*. Recuperado el 14 de marzo de 2021, de https://sites.google.com/site/ittgknd/home/1-4-software-de-computo-numerico

The MathWorks, I. (2021). *MATLAB - El lenguaje del cálculo técnico - MATLAB & Simulink*. Recuperado el 14 de abril de 2021, de https://la.mathworks.com/products/matlab.html?s_tid=hp_products_matlab

Toledo Castro, L. E. (2019). *Jugando a la interfaz: la interfaz de usuario como recurso*

en los videojuegos. Recuperado el 14 de marzo de 2021, de https://riunet.upv.es/handle/10251/130040

Unity Technologies. (2017, julio 12). *Entrada para controladores OpenVR*. Recuperado el 14 de marzo de 2021, de https://docs.unity3d.com/560/Documentation/Manual/OpenVRControllers.html

WiiBrew. (2020, julio 24). *Wiimote* . Recuperado el 05 de abril de 2021, de http://wiibrew.org/wiki/Wiimote

Zaldívar, U. (2003). *Robótica Asistida por Teleoperación y Realidad Virtual*. Recuperado el 05 de abril de 2021, de http://www.cs.cinvestav.mx/TesisGraduados/2003/tesisUlisesZ.pdf

Anexos