CENTRO DE INVESTIGACIONES EN OPTICA, A.C.

POSGRADO INTERINSTITUCIONAL DE CIENCIA Y TECNOLOGÍA

GEMELO VIRTUAL DE UN BRAZO ROBÓTICO COLABORATIVO DE 6 GRADOS DE LIBERTAD BAJO LA TÉCNICA DE REALIDAD AUMENTADA

TESIS

QUE PARA OBTENER EL GRADO ACADÉMICO DE

MAESTRA EN CIENCIA Y TECNOLOGÍA EN LA ESPECIALIDAD DE MECATRÓNICA

PRESENTA

ING. LORENA RODRÍGUEZ ISLAS

DIRECTOR

DR. CARLOS ALBERTO PAREDES ORTA

AGUASCALIENTES, AGUASCALIENTES, JUNIO, 2022.

V.°B.° Dr. Carlos Alberto Paredes Orta

AGRADECIMIENTOS

Agradezco de manera especial a mis maestros del Centro de Investigaciones en Óptica, que compartieron conmigo sus conocimientos; gracias a esto me fue posible la elaboración de esta investigación, así como agradecer su temple, tiempo y entrega que tuvieron para que el presente estudio se desarrollara de manera exitosa.

Quiero extender un sincero agradecimiento a el Dr. Carlos Alberto paredes Orta, el Dr. Fernando Martell Chávez y el Dr. Gerardo Ramón Flores Colunga; ya que gracias a su capacidad, experiencia, atención y constante asesoramiento se logró cumplir el objetivo del estudio.

Doy gracias al admirable apoyo y acompañamiento de mi familia en cada momento de mis estudios.

Por último, me gustaría reconocer a mis compañeros y amistades que estuvieron conmigo, me guiaron y asistieron a fortalecer la presente investigación, principalmente a Jonathan Esquivel Hernández por su valiosa orientación que me brindó durante la estancia en el Centro de Investigaciones en Óptica (CIO).

La actual investigación de ingeniería fue posible utilizando la infraestructura de los laboratorios virtuales del CIO y con la beca de posgrado CONACYT (Consejo Nacional de Ciencia y Tecnología).

RESUMEN

Las industrias de hoy requieren mayor rapidez y sistemas aptos para la toma de decisiones. Técnicas emergentes como los robots colaborativos y la realidad aumentada, son los elementos idóneos. Se sabe que los cobots son especialmente seguros para poder trabajar con los humanos y ofrecen gran flexibilidad para adaptarse a cualquier aplicación, gracias a que no requieren de celdas de seguridad. Por otro lado, la realidad aumentada permite agregar información adicional a una imagen del mundo real [1], ofreciendo así una realidad transformada, siendo muy útil en capacitación o en asistencia al operar equipos en las empresas. Sin embargo, el desconocer y no tomar en cuenta correctamente las dimensiones de un cobot al momento de programarlo, así como el mal posicionamiento de su efector final, puede suscitar en rutinas deficientes o colisiones; por lo que el presente estudio desarrollado, busca la validación de la programación de cobots de manera gráfica, es decir, incorpora la realidad aumentada en el gemelo virtual del TechMan® TM5-900, el cual se ejecuta bajo el sistema Unity, utilizando la librería Vuforia para crear la aplicación multiplataforma que puede ejecutarse en cualquier dispositivo electrónico portátil con sistema Android, el cual corre sobre un servidor virtual, comunicando el robot físico y virtual vía Modbus. Así mismo, se describe el cálculo e implementación de la cinemática directa e inversa del cobot de 6 grados de libertad, con el fin de efectuar correctamente las posiciones en cada articulación del robot y así colocarse en la posición final de forma precisa y óptima.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE ANEXOS

1. INTRODUCCIÓN

La creciente popularidad de los robots colaborativos para la automatización de procesos no proviene exclusivamente de sus capacidades, sino que también se debe a que es un producto competitivo [2], por lo que muchas empresas están apostando por su implementación, y es por esto que es fundamental conocer cómo manejar correctamente un robot colaborativo con herramientas rápidas y fáciles, al igual que de la manera más eficiente, es decir, programar las rutinas más óptimas en un cobot con los tiempos más cortos de ejecución, posiciones correctas del efector final y con una interacción humano-robot sencilla de comprender y utilizar; por consiguiente es que surge el presente trabajo de investigación, el cual tiene el fin de crear y simular las rutinas del robot colaborativo de seis grados de libertad TechMan® TM5-900 en el software de Unity, a través de la generación de su gemelo digital realizado en SolidWorks, visualizado de manera totalmente gráfica mediante realidad aumentada, utilizando Vuforia, mostrando su modelado matemático para su control de movimientos y usando el protocolo de comunicación Modbus para su funcionamiento remoto.

El presente documento está conformado por capítulos, donde en el capítulo uno se muestra un panorama general de la investigación, es decir, se habla de los trabajos que le anteceden a este proyecto, se habla de la necesidad de crear un simulador fácil de comprender y manipular para el usuario, manteniendo la esencia física del robot a pesar de utilizarlo de manera remota; esta sección concluye con los objetivos del trabajo. En el capítulo dos se define y se habla de la historia de la industria 4.0, la RA, gemelos digitales y cobots, así como describe los elementos primordiales que se utilizaron para el desarrollo de la aplicación móvil. El capítulo tres presenta la metodología utilizada para cumplir los objetivos establecidos y su respectiva explicación detallada para su desarrollo, control e integración de herramientas. En el capítulo cuatro, se presentan los resultados, donde se muestra la funcionalidad del sistema desarrollado para finalizar con una discusión acerca de sus posibles aplicaciones futuras.

1.1 ANTECEDENTES

La interacción entre robots y humanos es un amplio campo de investigación, ya que tiene importantes aplicaciones beneficiosas en la industria. Un ejemplo de ello es el trabajo denominado “A multimodal interface to control a robot arm via the web: a case study on remote programming” [3], En esta investigación, se presenta la interfaz de usuario y la arquitectura del sistema de un telelaboratorio basado en Internet, que permite controlar y programar de forma remota robots educativos en línea. La interfaz de usuario utiliza técnicas basadas en realidad aumentada y realidad virtual no inmersiva, que mejoran la forma en que los operadores obtienen y transmiten información desde o hacia el escenario robótico. Puesto que la realidad aumentada se describe como una técnica para mejorar la forma en que los operadores interactúan con los escenarios robóticos, ya que se simplifica la recepción y publicación de información desde y hacia el sistema [4], Native Robotics diseñó una aplicación RA móvil: OmniFit, que permite al usuario visualizar un robot UR o una celda de trabajo [5]. La aplicación se ejecuta en dispositivos iPhone y iPad a partir de 2016, en ésta se puede observar una galería de diferentes aplicaciones de cobots, las cuales son personalizables, y donde es necesario especificar los parámetros de la celda robótica, para después iniciar la simulación o visualizar la celda en AR.

En cuanto al estudio denominado “Robot Programming Through Augmented Trajectories in Augmented Reality”, presenta un enfoque en la programación de robots basado en trayectorias aumentadas, usando una pantalla montada en la cabeza de realidad mixta (Microsoft Hololens) y un brazo robótico de 7 grados de libertad, donde se diseña una interfaz robótica de realidad aumentada (RA) con cuatro funciones interactivas para facilitar la tarea de programación del robot: Especificación de trayectoria, vistas previas virtuales del movimiento del robot, visualización de parámetros del robot y reprogramación en línea durante la simulación y ejecución. Este trabajo presenta un estudio de caso industrial como validación [6]. De modo similar, la compañía ABB creó una aplicación basada en RA [7], donde el usuario puede analizar un robot ABB o una celda de trabajo, creando una visualización animada en la versión de escritorio y exportarla.

En la publicación “Augmented reality-assisted robot programming system for industrial Applications” se analiza un sistema de programación de robots asistido por RA, que proporciona una programación de robots más rápida e intuitiva que las técnicas convencionales. ARRPS es diseñado para permitir a los usuarios con pocos conocimientos de programación de robots programar tareas para un robot en serie. Con una interfaz de usuario RA y un puntero de mano para la interacción, los usuarios pueden moverse libremente por la celda de trabajo para definir puntos 3D y rutas para el verdadero robot a seguir. Los datos y algoritmos de los sensores se utilizan para la planificación del movimiento del robot, la detección de colisiones y la validación de planos [8]. De igual forma en la investigación “Concept and architecture for programming industrial robots using augmented reality with mobile devices like microsoft HoloLens” [4], se desarrolla una arquitectura que permite la interacción con sistemas robóticos industriales, lo que significa la visualización de programas de robot y el espacio de trabajo del robot o restricciones de movimiento como planos de seguridad o límites de ángulo de articulación, así como la interacción con el robot, lo que significa manipular el robot cambiando ángulos de articulación o crear programas para el robot objetivo, etc.

En el trabajo “A Study of Cobot Practitioners Needs for Augmented Reality Interfaces” explora las implementaciones basadas en RA en robótica y las clasifica según el tipo de dispositivo utilizado, con el enfoque principal en la categoría menos explorada: RA móvil, donde se desarrolla un prototipo de aplicación RA como una co-interfaz para un TP. Los resultados obtenidos sugieren que los usuarios podrían beneficiarse de las soluciones RA basadas en dispositivos móviles en la fase de puesta en servicio y resolución de problemas durante la vida útil del robot [9].

Por último, Hongchao Fang propone en su investigación denominada “Robot Programming Using Augmented Reality”, un sistema de programación de robots basado en RA (RPAR-II). El sistema RPAR-II ayuda a los usuarios durante el proceso de programación del robot, desde la planificación de tareas hasta la ejecución. En este sistema se emplean metodologías basadas en visión estéreo para el registro de objetos virtuales, así como el seguimiento de posición de dispositivos interactivos [10].

1.2 DEFINICIÓN DEL PROBLEMA

Cuando se selecciona un robot para una tarea especifica, las dimensiones es una de las principales características que se tienen que tomar en cuenta para su adquisición, el no tomar en cuenta correctamente las medidas de un cobot en una línea de producción o desconocerlas, genera rutinas deficientes y aumenta el peligro de colisión o bloqueos de sus mismas ejecuciones con diversos objetos o humanos, así mismo las interfaces de usuario y métodos de interacción para la programación e información de cobots resulta complicada y visualmente limitada para el usuario, lo que provoca que las capacidades del robot no sean aprovechadas al máximo, es decir, para examinar el robot en el espacio de trabajo físico, los efectos de movimiento y rutinas, así como la posición de su efector final de manera gráfica; es necesario que los usuarios recurran a una pantalla para proporcionar visualización de información adicional y operaciones de interacción con el robot, las cuales carecen de esencia del robot de manera gráfica, o en una situación más desfavorable, el usuario termina creando mentalmente el análisis de posición final o rutina del cobot para después proceder a programarlo.

1.3 Justificación

El presente trabajo se basa en el desarrollo de una aplicación móvil con realidad aumentada para el control de un cobot, en la cual se localizó poca bibliografía referente a la investigación sobre RA basada en dispositivos móviles en robótica [9]. Se sabe que un teléfono inteligente o una tableta son mucho más baratos que cualquier otro dispositivo habilitado para RA, y la mayoría de las personas cuentan con uno; de igual forma el utilizar un cobot significa que será amigable con su entorno y los humanos, gracias a sus elementos de seguridad a diferencia de un robot normal que tiene que estar dentro de una celda. Los robots debido a su versatilidad y repetibilidad en realización de tareas de fabricación [8], resultan factibles de implementar en líneas de producción. El desarrollo de un sistema que incorpora la realidad aumentada en un gemelo virtual del robot colaborativo, permite ser manipulado de forma remota, ya que es una aplicación móvil, lo que significa que los usuarios pueden diseñar y probar las rutinas del cobot TechMan® sin necesidad de estar presencialmente.

La realidad aumentada se utiliza en esta investigación, ya que es una tecnología emergente que posibilita ver gráficamente los movimientos del robot que el usuario pretende generar, para después enviar las instrucciones al robot físico y hacerlo funcionar en tiempo real en un espacio de trabajo seguro; esto evitará accidentes, ya que las rutinas generadas por el usuario se pueden ver y reproducir desde diferentes perspectivas y ver claramente cuando el robot colisiona, algo muy beneficioso en los procesos de fabricación ya que se pueden evitar costosos errores en la fase de producción. Por otra parte, el tener un sistema funcionando en tiempo real, significa que la información de la base de datos siempre va a estar actualizada y sincronizada, lo que hace posible el monitoreo y supervisión del funcionamiento del robot. Finalmente, el tener una interfaz gráfica y fácil de visualizar, proporciona a los usuarios que no cuentan con una gran habilidad para programar, hacerlo sencillamente con la posibilidad de reprogramar al cobot y de esta manera reducir costos, ya que no es necesario más personal para realizar esta tarea.

1.4 Hipótesis

La generación de un gemelo virtual del cobot TM5-900, basado en realidad aumentada, permitirá la manipulación del robot físico a partir de una interfaz virtual, en la que se probarán rutinas de movimiento, antes de ser enviadas desde un dispositivo móvil al robot TechMan®.

1.5 OBJETIVOS DE INVESTIGACIÓN

1.5.1 Objetivo general

Generar un gemelo virtual de un robot colaborativo de 6 GDL (Grados De Libertad), utilizando como interfaz la realidad aumentada, para lograr su manipulación mediante un dispositivo móvil y esta sea capaz de enseñar las capacidades del cobot y poder generar rutinas del robot TechMan®.

1.1.2 Objetivos específicos

- Diseñar gemelo virtual.
- Elaborar articulaciones de gemelo virtual.

- Modelar cinemática de gemelo virtual (Directa e inversa).
- Planear trayectorias de movimiento para brazo robótico.
- Realizar control basado en modelo virtual.
- Elaborar interfaz para conexión del sistema de gemelo virtual con RA y robot colaborativo.

2. FUNDAMENTOS TEÓRICOS

En este capítulo se inicia explicando los términos de realidad aumentada, gemelos virtuales, robot colaborativo e industria 4.0, ya que son los elementos principales para el desarrollo del proyecto.

2.1 Industria 4.0

2.1.1 Definición

El concepto de fondo de la Industria 4.0 consiste en la interconexión de todos los subprocesos del entorno productivo de manera automatizada, informatizada e "inteligente". En otras palabras, lo que pretenden los proyectos de Industria 4.0 es crear procesos industriales complejos donde todos los equipos que intervienen estén conectados entre sí, tengan la capacidad de optimizar tiempos y recursos e incluso tomar decisiones de manera autónoma [11].

2.1.2 Historia

Las revoluciones industriales son casos que arrojan luz sobre nuestra vida cotidiana y estas revoluciones han supuesto mejoras significativas. A lo largo de la historia ha habido cuatro grandes revoluciones industriales como base [12], [13].Estas revoluciones son las siguientes:

- Industria 1.0: La incorporación del agua y el vapor al trabajo mecánico basado en la Industria 1.0, ha surgido a finales del siglo XVIII.
- Industria 2.0: Establecimiento de la electricidad basada en líneas de producción en masa para la medición de la mano de obra y promoción, es la base de la revolución industrial. Esta revolución comenzó a principios del siglo XX, especialmente la línea de producción de Henry Ford en paralelo con la idea que

se planteó en este proceso y ha experimentado el desarrollo más significativo; ha sido la base de la línea de producción actual.

- Industria 3.0: Con este movimiento surgido a principios de la década de 1970, las máquinas de producción mecánicas y eléctricas son reemplazadas por dispositivos electrónicos que pueden programarse. El controlador lógico programable (PLC) ha surgido en esta revolución industrial. Hoy en día, actualmente vive la revolución industrial.
- Industria 4.0: Es divulgada por el gobierno alemán y son responsables de los fines de fabricación informatizados. El término Industria 4.0 se introdujo por primera vez en 2011, en la Feria de Hannover y este término se presenta bajo el título de producción inteligente [14].

  Los principales pilares de la industria 4.0 son:

  - Simulación
  - Fabricación aditiva
  - Sistemas de integración horizontales y verticales
  - Ciberseguridad
  - Big Data y análisis
  - Internet industria de las cosas
  - Computación en la nube
  - Robots autónomos
  - Realidad aumentada (RA)

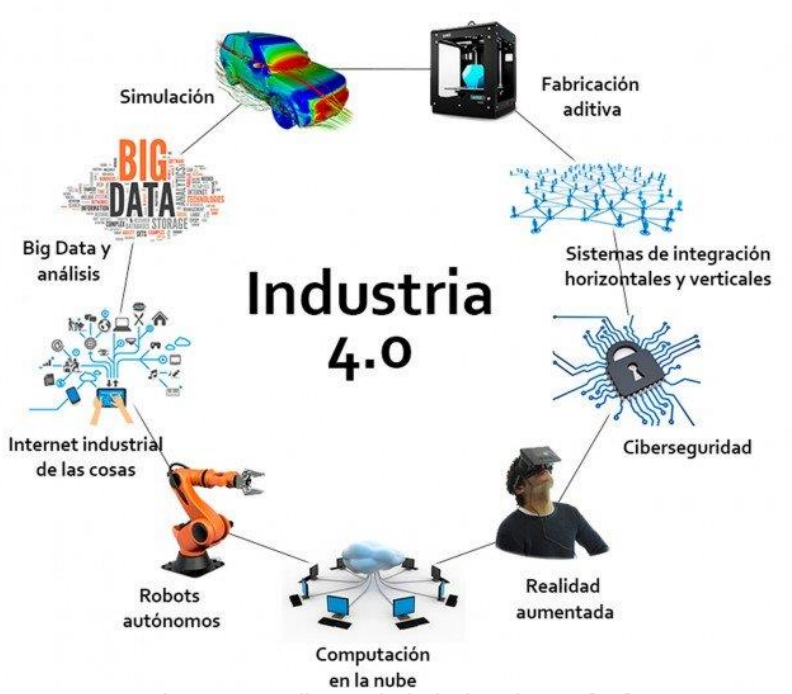

Figura 2.1: Pilares de la industria 4.0 [15].

Debido a que los robots, la realidad aumentada y los gemelos digitales, se encuentran dentro de los pilares de la industria 4.0 y son elementos esenciales en este proyecto, en las próximas secciones se profundizan estos temas.

2.2 REALIDAD AUMENTADA

2.2.1 Definición

La realidad aumentada es una tecnología que integra señales captadas del mundo real (típicamente video y audio) con señales generadas por computadores (objetos gráficos tridimensionales); las hace corresponder para construir nuevos mundos coherentes, complementados y enriquecidos, hace coexistir objetos del mundo real y objetos del mundo virtual en el ciberespacio [16].

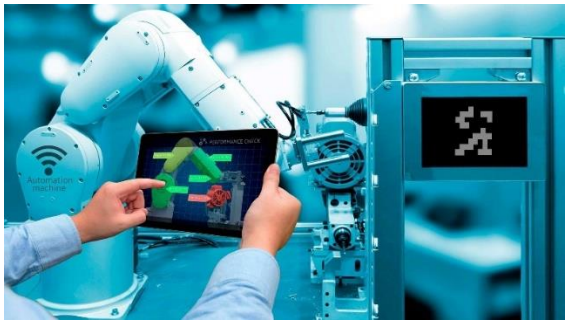
Figura 2.2: Representación de la realidad aumentada [17].

2.2.2 Historia

Conforme al artículo de la revista UNAM sobre la realidad aumentada, los inicios de la realidad aumentada se remontan a 1960, cuando Sutherland usó un dispositivo de despliegue de imágenes tridimensionales de tipo casco, para visualizar gráficos tridimensionales [16].

El concepto de realidad aumentada se introdujo en 1992 para denotar una pantalla transparente que Caudel y Mizell habían diseñado como parte de un proyecto de investigación, donde describieron el concepto de la siguiente manera: " Esta tecnología emergente se utiliza para “incrementar” la visualización del usuario con la información representada gráficamente de la tarea actual, por lo que nos referimos a la tecnología como “realidad aumentada”. Es importante *distinguir* entre realidad aumentada y realidad virtual ya que estos dos conceptos no son lo mismo. En la realidad virtual, los usuarios están completamente inmersos en un mundo virtual y no pueden ver el mundo real que los rodea. Por el contrario, la realidad aumentada fusiona los mundos virtual y real superponiendo información virtual sobre la percepción del usuario del mundo real. Azuma define la realidad aumentada como un sistema que tiene tres características: (a) la capacidad de combinar objetos reales y virtuales, (b) la capacidad de ser interactivo en tiempo real y (c) la capacidad de utilizar objetos 3D. Cabe señalar que la realidad aumentada incluye más sentidos además de lo visual, y se aplica potencialmente a todos los sentidos, incluidos el oído, el tacto y el olfato. Ejemplos de información que puede superponerse en el entorno real para crear realidad aumentada son imágenes, audio, video y sensaciones táctiles o hápticas. Hoy en día, la amplia adopción de la realidad aumentada se observa principalmente en los juegos, los deportes y el turismo, y las nuevas soluciones dirigidas a estas áreas comerciales se están desarrollando rápidamente. La implementación de la realidad aumentada se basa generalmente en alguna forma de ancla del mundo real utilizada para la navegación y para proporcionar información basada en el contexto de información al usuario [18].

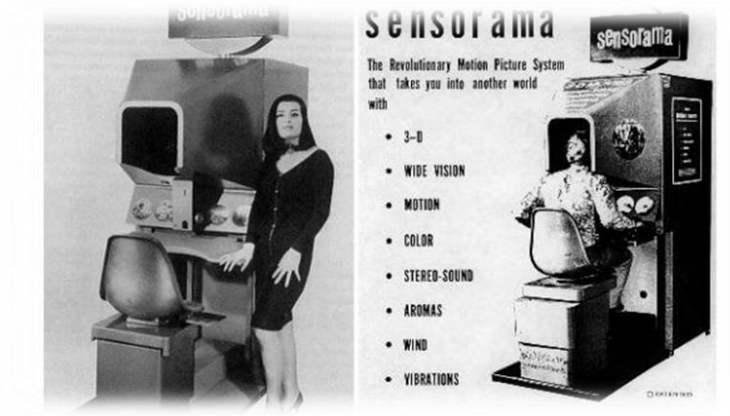

Figura 2.3: Antecedentes de la realidad aumentada [19].

2.2.3 Niveles

Existen diferentes niveles de realidad aumentada según su grado de complejidad o de fusión con la realidad [1].

- **Nivel 0** – *Physical World Hyper Linking* o enlazado con el mundo físico: este nivel se caracteriza por el empleo de imágenes en 2D como códigos de barras que sirven como enlaces a otros contenidos. Es la forma más básica de realidad aumentada.
- **Nivel 1** – *Marker Based AR* o realidad aumentada con marcadores: se emplean aplicaciones que pueden reconocer patrones en 2D o 3D simples, como figuras en blanco y negro, formas o dibujos esquemáticos.
- **Nivel 2** – *AR without markers* o realidad aumentada sin marcadores: No necesitan marcadores, sino que utilizan sistemas como la brújula digital para conocer la localización del usuario y proyectar imágenes virtuales de interés sobre la realidad cotidiana.
- **Nivel 3** – *Augmented vision* o visión aumentada: son dispositivos de alta tecnología que permiten una experiencia totalmente inmersiva y fusionada con la realidad [20, 21].

Después de analizar los tres niveles que existen en la realidad aumentada, se puede decir que el nivel que se está utilizando en esta investigación, es el nivel uno, ya que utilizamos un marcador (Figura 3.7) para la generación de RA.

2.2.4 Funcionamiento

La realidad aumentada agrupa aquellas tecnologías que permiten la superposición en tiempo real de:

- Imágenes
- Marcadores
- Información

Generados de forma virtual sobre el mundo físico. Así se crea un entorno en el la información y los objetos virtuales se relacionan con el entorno físico [22].

Esto ofrece una experiencia diferente al usuario y puede llegar a pensar que los objetos forman parte de su realidad cotidiana. Por ello son muchos los que la consideran una nueva lente para ver al mundo, y de ahí el calificativo de “aumentada”. Los adelantos de los dispositivos, y especialmente los de los teléfonos inteligentes han dado lugar a que hoy en día podamos disfrutar de esta tecnología de forma generalizada.

Pero el elemento esencial es la conectividad permanente que potencia el uso de estos dispositivos. Para ello son necesarios cuatro ingredientes básicos:

1. El primero de ellos es un elemento que **capture las imágenes reales** como, por ejemplo, una cámara.
2. También se necesita un elemento sobre el que **proyectar las imágenes reales con las sintetizadas**, como por ejemplo un ordenador, un móvil o una consola.
3. Además, es necesario un **elemento de procesamiento** o varios de ellos que funcionen conjuntamente. Esto sirve para interpretar la información del mundo real que recibe el usuario. Después genera la información virtual que cada servicio necesite y la mezcla de forma adecuada.
4. Finalmente, también es necesario un **elemento activador** de la realidad aumentada. En un mundo perfecto sería la imagen que estén visualizando los usuarios, ya que a partir de ahí el sistema debería reaccionar. Sin embargo, como es un proceso complejo, se utilizan otros elementos que lo sustituyen, como por ejemplo GPS (Global Positioning System, es decir, Sistema de Posicionamiento Global), acelerómetros o brújulas. Incluso en un futuro estos elementos podrían llegar a eliminarse. ¿Cómo? Proyectando la información para que el ojo pudiera verla sobre la retina, unas gafas o alguna técnica holográfica avanzada [23].

En la Figura 2.4 que se muestra a continuación, se puede observar lo anteriormente mencionado, en resumen, mediante un bosquejo.

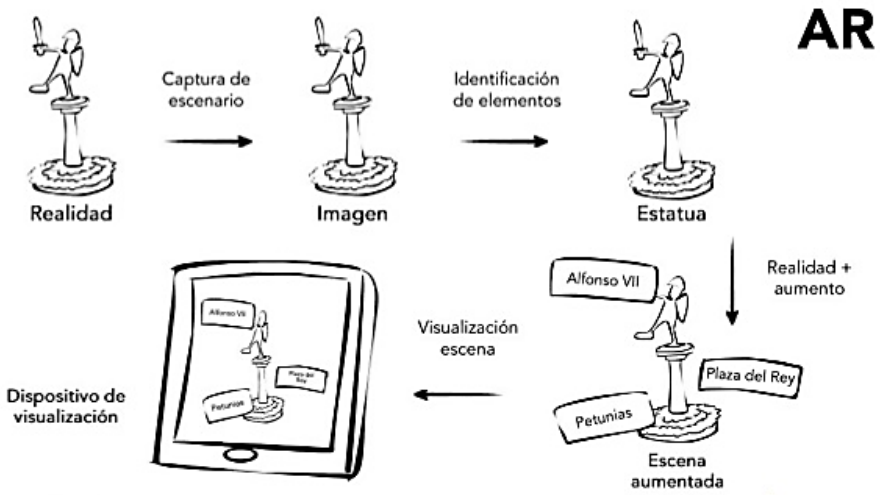

Figura 2.4: Esquemático del funcionamiento de realidad aumentada [24].

2.3 GEMELO DIGITAL

2.3.1 Definición

Los gemelos digitales son representaciones virtuales de los componentes del elemento físico en un proxy virtual mediante un modelo matemático que simula un objeto en el mundo digital [25].

Figura 2.5: Representación de gemelo virtual [26].

2.3.2 Historia

Según la universidad de Valladolid, con base al trabajo de fin de grado “Los gemelos digitales en la Industria 4.0”, el origen de los gemelos digitales se remonta a la década

de los años 80, cuando la NASA comenzó a diseñar simulaciones que reflejaban el comportamiento de sus naves y equipos para asegurar el éxito de las misiones y la integridad física de los tripulantes. La motivación que llevó a la agencia aeroespacial estadounidense a profundizar en esta tecnología fue el accidente que sufrió en 1970 la misión Apolo 13 (donde nació la famosa frase: “Houston, tenemos un problema”), que finalmente no acabó en tragedia porque los ingenieros y el equipo de control del Apolo 13 replicaron, desde la estación, un módulo gemelo de la nave para probar, en tiempo récord, soluciones a los problemas de supervivencia que estaban teniendo los astronautas. Cuando los niveles de dióxido de carbono en el Apolo 13 estaban empezando a ser insostenibles, desde Houston consiguieron crear un sistema de purificación de oxígeno con los mismos materiales que tenían los tripulantes, trasladándoles a éstos las instrucciones para que lo creasen en el espacio.

No obstante, no es hasta el año 2003 cuando empieza a aplicarse como tal el término digital twin, gracias a una ponencia del ingeniero informático Michael Grieves en la Universidad de Michigan sobre la gestión del ciclo de vida del producto (en inglés, Product Lifecycle Management PLM). Con el paso de los años y el avance de las tecnologías de Big Data, IoT y almacenamiento en la nube, entre otras, se ha podido extender el ámbito de aplicación de los gemelos digitales a otros campos y sectores [25].

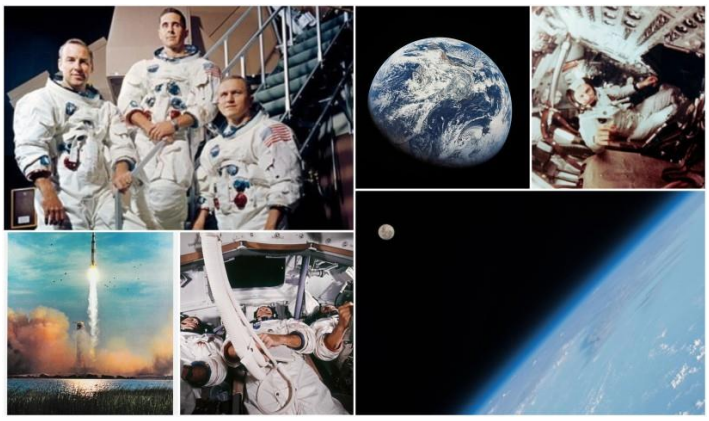

Figura 2.6: Inicios del gemelo virtual [27].

2.4 ROBOT COLABORATIVO

2.4.1 Definición

Un robot colaborativo es un brazo robótico creado para trabajar junto a los humanos en una cadena de producción, de ahí que también sean conocidos como robots colaborativos [28].

2.4.2 Historia

Acorde Universal Robots (Fabricante Danés de brazos robóticos colaborativos), el término robot ha sido acuñado hasta 1921, por el escritor checo Karel Capek, donde la palabra proviene de robota que significa servidumbre. El primer robot industrial fue presentado hasta 16 años después, el cual consistió de un diseño similar a una grúa con cinco ejes de movimiento y este funcionaba con un motor eléctrico. Su función era para colocar bloques de madera en maneras preprogramados. Sin embargo, no fue hasta 1956 que se patentó el primer robot por la compañía americana Unimation. El desarrollo de los robots comenzó a acelerarse a medida que nacieron nuevas empresas y se agrandaron muchas líneas de producción. A principios de los años 70, ABB Robotics y Kuka Robotics ambas compañías europeas, comercializaron sus robots eléctricos controlados por un microprocesador. Para la siguiente década, en 1980's, los robots ya eran más comunes en la industria y era hora de que se iniciara a desarrollar nuevas ideas [29].

Según la empresa Harmonic Drive, los cobots (robots colaborativos), juegan un papel determinante en proyectos de futuro tales como la industria 4.0, que tiene por objetivo la amplia digitalización de la producción industrial en Alemania. Para ello, los cobots no solo deben servir como prometedores componentes de la producción y hacer más cómoda la actividad diaria de los empleados, sino que también deben facilitar y optimizar la recogida de datos a nivel empresarial interno. Los datos mencionados sirven para el desarrollo eficiente y efectivo de la producción.

El desarrollo continuo de cobots se debe en gran parte al aspecto de la seguridad. Para ello, actualmente los cobots están provistos, entre otros elementos, de sensores de fuerza, sensores capacitivos y una variedad de sistemas de cámara que recogen informaciones en tiempo real y las transmiten a un software. Si el software detecta un fallo o un riesgo, debe poder interrumpir inmediatamente el funcionamiento del aparato [1].

2.4.3 TECHMAN® TM5-900

El elemento principal de la investigación es el cobot TM5-900 (Figura 2.7), el cual cuenta con las siguientes características principales [30]:

- Robot Colaborativo 6-ejes
- Alcance: 900 mm
- Carga útil: 4 kg
- Alimentación: 100-240 VCA, 50-60Hz
- Seguridad: ISO10218-1 / ISO TS15066
- Incluye: Controlador (Control Box)
- Sistema de Visión embebido 5MPx
- Velocidad típica: 1.4 m/s

Cabe mencionar que para este desarrollo se contó con la infraestructura y recursos necesarios del laboratorio virtual del CIO (Centro de Investigaciones en Óptica, A.C.), así como con las licencias de los softwares UNITY y Vuforia.
El objetivo principal de todos estos elementos es desarrollar una herramienta que elimine las barreras entre el un robot colaborativo (cobot) físico y el virtual.

Figura 2.7: Robot colaborativo de seis grados de libertad, modelo TM5-900 [31].

A continuación, se exponen sus ventajas principales [32].

- **Visión de robots**
  - Sistema de visión Integrado.
  - TM Robot está equipado con un sistema de visión integrado, el cual integra perfectamente el hardware y el software.

- Muchas funciones estándar de visión para robot están ya integradas en el sistema: coincidencia de patrones, localización de objeto, mejoramiento de imagen, lectura de código de barras, reconocimiento de color, etc.

- **Interfaz de usuario**
  - La interfaz de usuario está basada en un diagrama de flujo gráfico. El operador puede programar todas las funciones de visión ya integradas, desde la IGU (Interfaz Gráfica de Usuario). Además, el tradicional y pesado Teaching Pendant cableado como "cordón umbilical" ha desaparecido. El software TMflow® puede ser operado en PC´s, laptops o tabletas.
- **Programación**
  - Se pueden bloquear aquellos ejes seleccionados, para permitir el ajuste en planos definidos para entonces hacer el ajuste fino en las coordenadas por medio de una edición de aquellas coordenadas dentro del software de programación.
- **Seguro**
  - Fuerza limitante.
  - TM Robot está certificado bajo las normas ISO 10218-1 & ISO/TS 15066 y cumple los requerimientos de seguridad para la operación conjunta entre humanos y robots colaborativos, permitiendo al robot ser programado con límites tanto de velocidad como de fuerza.
  - TM Robot es serio acerca de la seguridad en cada aspecto del diseño de todo el sistema del robot, a través de hardware, software y diseño operativo.
  - Diseño ergonómico.
  - TM Robot está físicamente diseñado para ser seguro para su entorno. Un acabado y bordes totalmente suaves, son parte de toda la experiencia colaborativa.

Los **componentes** que integran al robot colaborativo son los siguientes:

- Caja de control
- Mando del robot
- Red del cliente
- Red de área local

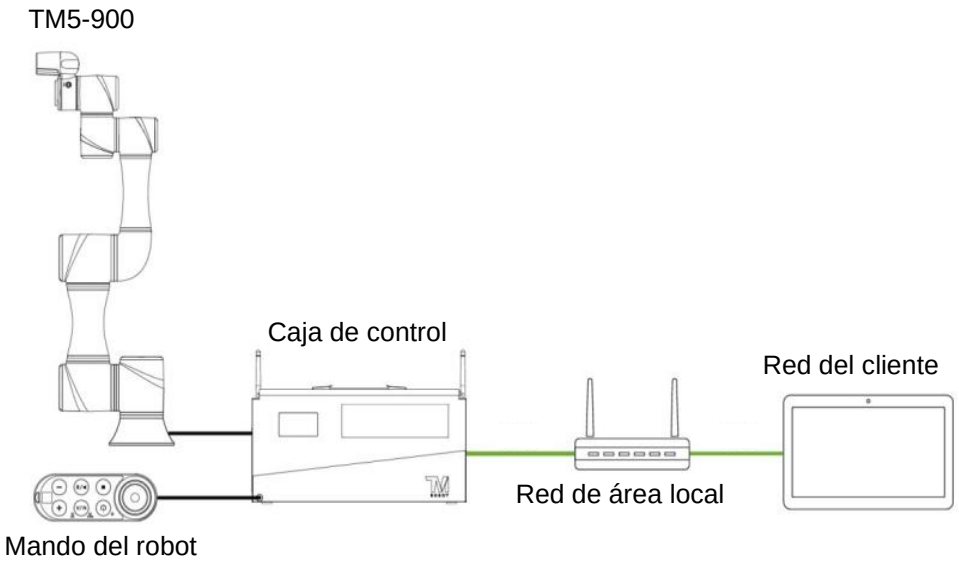

Figura 2.8: Elementos de cobot TechMan® TM5-900.

2.5 SOFTWARES Y HERRAMIENTAS DE DESARROLLO UTILIZADAS

2.5.1 SolidWorks

SolidWorks es un software tipo CAD, de diseño mecánico, que utilizando un entorno gráfico basado en Microsoft Windows permite de manera intuitiva y rápida la creación de Modelos sólidos en 3D, Ensamblajes y Dibujos. Se basa en el modelado paramétrico, reduciendo el esfuerzo necesario en modificar y crear variantes en el diseño, ya que las cotas y relaciones usadas para realizar operaciones se almacenan en el modelo [33].

Figura 2.9: Logotipo de SolidWorks [34].

2.5.2 Blender

Blender permite crear modelados en 3D; es un programa gratuito al que cualquier usuario puede acceder. Además de ser gratuito, es de código abierto, es decir que la colaboración del programa es completamente abierta para poder conseguir más beneficios en la herramienta de modelado 3D [35].

Figura 2.10: Distintivo de Blender [36].

2.5.3 Unity

Unity es una plataforma de desarrollo que se utiliza generalmente para el diseño de videojuegos. Sin embargo, ofrece muchas más opciones, tal es el caso para el que se utilizó en el presente trabajo de investigación, es decir, para realidad aumentada. Unity permite aplicar funciones como renderizado fotorrealista, físicas avanzadas, optimización para diferentes tipos de dispositivos. Una vez que la app ha sido creada, se puede utilizar en numerosos dispositivos móviles [21].

Figura 2.11: Emblema de Unity [37].

2.5.4 Vuforia

Vuforia Engine es un kit de desarrollo de software (SDK) para crear aplicaciones de Realidad Aumentada. Los desarrolladores pueden agregar fácilmente funciones avanzadas de visión por computadora a cualquier aplicación, lo que les permite reconocer imágenes y objetos e interactuar con espacios en el mundo real. Unity Editor es una plataforma de creación popular y útil para crear experiencias de realidad aumentada de vanguardia para dispositivos portátiles y anteojos digitales. El motor Vuforia se agrega fácilmente a cualquier proyecto [38].

Figura 2.12: Símbolo de librería Vuforia [39].

2.5.5 Android studio

Android Studio es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de apps para Android y está basado en IntelliJ IDEA.

Figura 2.13: Emblema de Android Studio [40].

2.5.6 Modbus

Modbus está diseñado para permitir a equipos industriales tales como Controladores Lógicos Programables (PLCs), computadores, drivers para motores y otros tipos de dispositivos físicos de entrada/salida comunicarse sobre una red [41].

Figura 2.14: Protocolo de comunicación utilizado.

Modbus/TCP básicamente encapsula una trama Modbus dentro de una trama TCP en una manera simple como se muestra en la Figura 2.15.

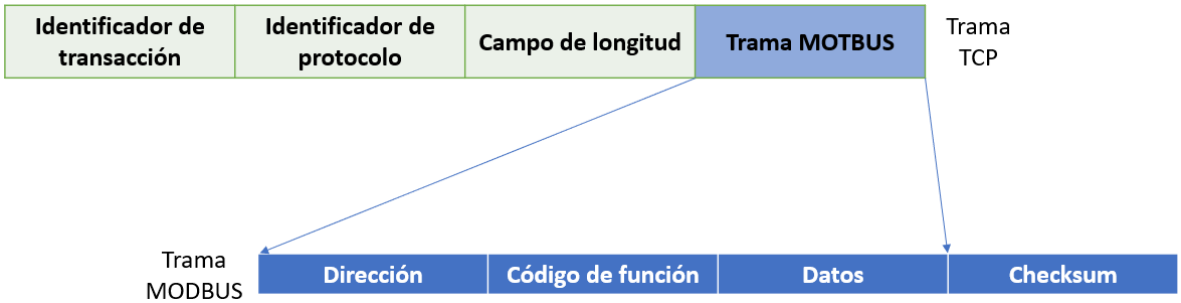

Figura 2.15: Trama TCP/Modbus.

2.5.7 TMflow®

TMflow® es una interfaz de usuario de programación del robot TM5-900 basada en diagramas de flujo.

Figura 2.16: Interfaz de usuario de programación de robot TM5-900.

3. METODOLOGÍA

Este proyecto se desarrolla bajo el sistema Unity 2018, donde se utiliza la librería Vuforia [42], para la generación de la aplicación basada en RA, la cual es ejecutada bajo un dispositivo móvil con sistema operativo Android 5, donde se corre sobre un servidor virtual generado en la red del laboratorio virtual del CIO. A continuación, se muestra la metodología utilizada para la realización del proyecto (Figura 3.1).

Figura 3.1: Metodología general del proyecto

3.1 DISEÑO CAD 3D

Se realizó el modelado del cobot a escala real, mediante el software de diseño Solidworks, donde se crearon las piezas, para terminar con el ensamblaje de todos los elementos, en la Figura 3.2 se observa el resultado del modelo final.

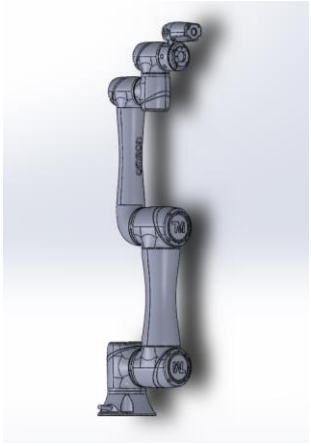

Figura 3.2: Modelado de robot de seis grados de libertad en SOLIDWORKS.

3.2 Desarrollo de cobot en UNITY

Posteriormente se procedió a exportar el modelo 3D que se realizó en Solidworks a la plataforma de desarrollo denominada Unity. En la siguiente imagen (Figura 3.3) se contempla la implementación lograda en la plataforma de desarrollo.

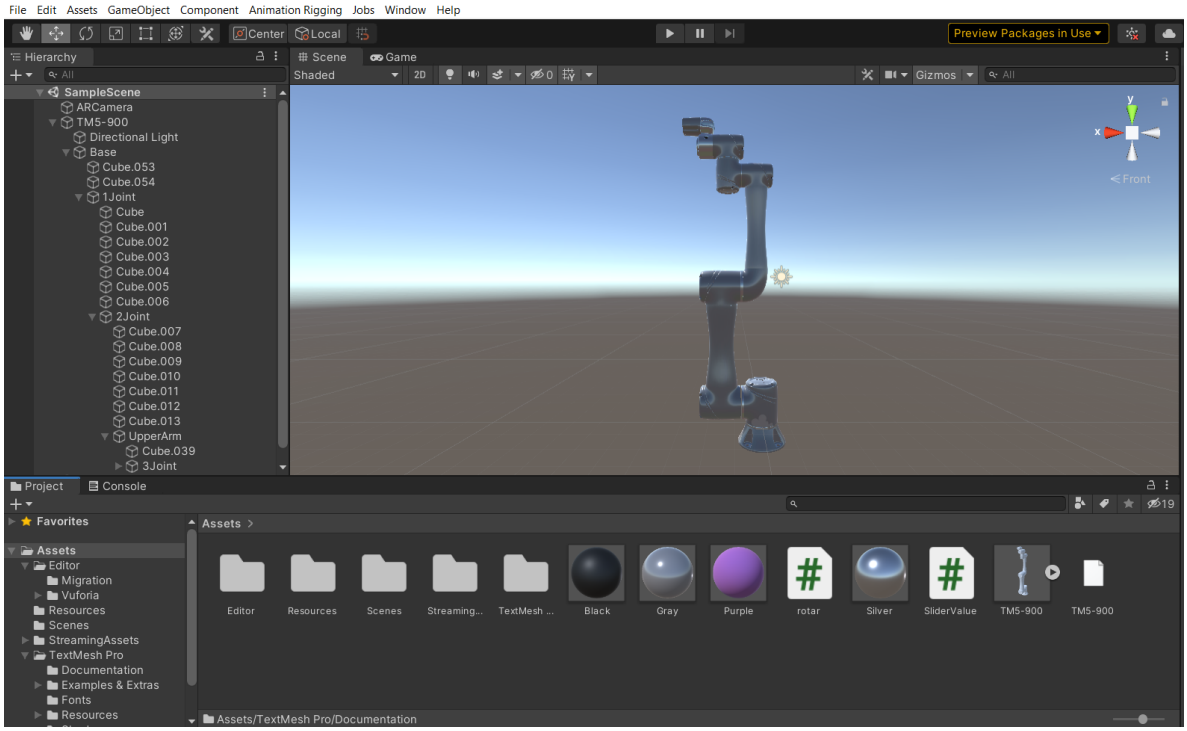

Figura 3.3: Implementación de cobot en Unity.

Cabe mencionar que cada pieza del cobot se separó por un GameObject, de igual forma estas piezas esta organizadas por grado de libertad, esto favorece a mantener una estructura limpia en la realización del proyecto.

En el diagrama que se muestra a continuación (Figura 3.4) se analizan los seis grados de libertad del cobot.

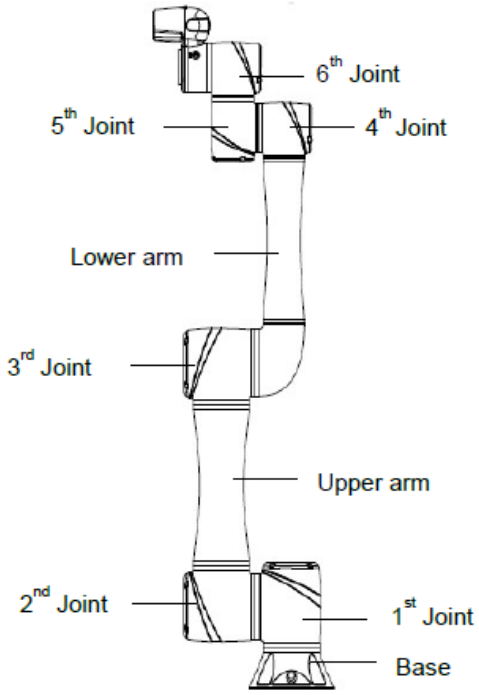

Figura 3.4: Grados de libertad del cobot [31].

Después de estructurar cada pieza del cobot en Unity, se aplicó el material, con la finalidad de crear un efecto realista. Luego se procedió a la elaboración de huesos (Figura 3.5). Los huesos tienen como propósito actuar como articulaciones, los cuales son la unión de dos grados de libertad, y así, a partir de influencias en la geometría del modelo, permitirá el movimiento y simulación de la animación del modelo 3D [43].

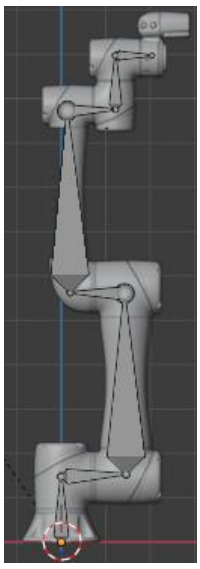

Figura 3.5: Creación de huesos de robot de seis grados de libertad.

3.3 Generación de aplicación

Con el fin de conseguir una animación ordenada y amigable con el usuario, se agregaron sliders en cada grado de libertad del cobot, por medio de Unity con la herramienta Canvas y la realización de un Script; de esta manera el usuario podrá observar el límite de grados que existe en cada articulación del cobot y así percatarse de los diversos movimientos permitidos para realizar en el cobot. Al momento de ejecutar la aplicación mediante Unity, el usuario observará en tiempo real los grados precisos que está rotando el cobot a través de los sliders (Figura 3.6).

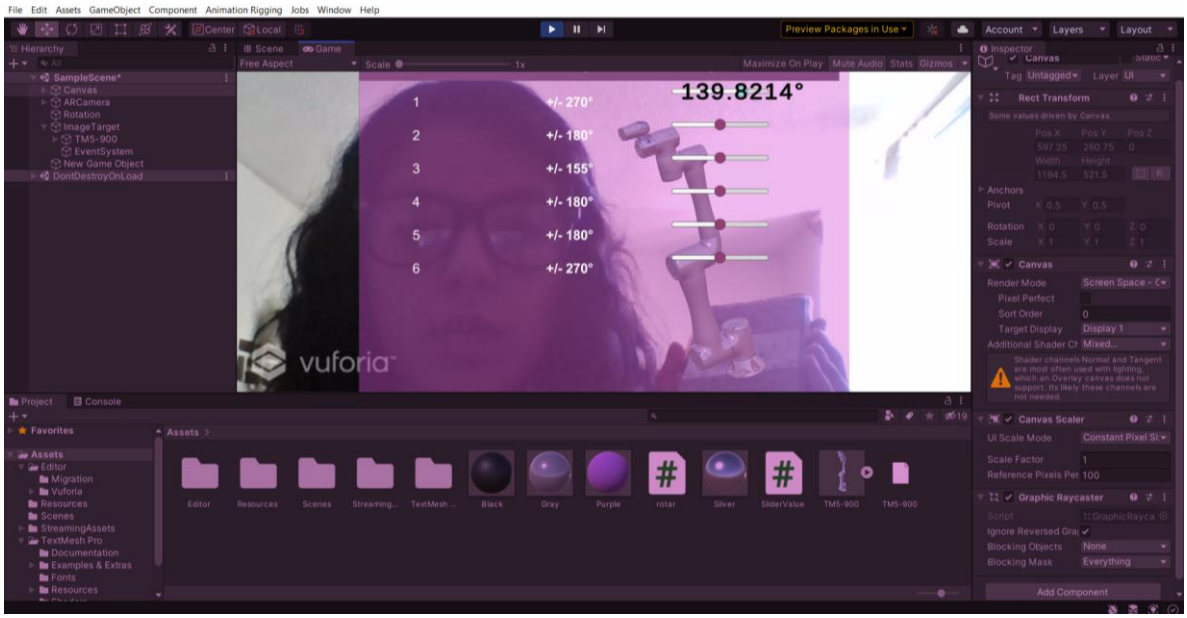

Figura 3.6: Ejecución de Sliders en Unity.

Con el objetivo de crear la realidad aumentada, se configuraron y activaron en Unity los permisos necesarios para utilizar Vuforia, con lo que se permite el desarrollo de la aplicación con RA.

3.4 Marcador de realidad aumentada

Para la generación de realidad aumentada se utilizó un marcador (Figura 3.7), el cual le proporciona a la aplicación de RA una clave activadora sobre dónde posicionar el contenido de RA.

Figura 3.7 Marcador utilizado para la generación de realidad aumentada en Unity [44].

Al momento de ejecutar el SDK (Software Development Kit), y leer el marcador, se ve reflejado el modelo de seis grados de libertad en RA con una proyección de tamaño real, junto con todas las características que se desarrollaron y aplicaron (Figura 3.8), y así lograr el análisis de las diferentes posiciones del cobot mediante su gemelo virtual y realidad aumentada antes de correr sus rutinas en físico.

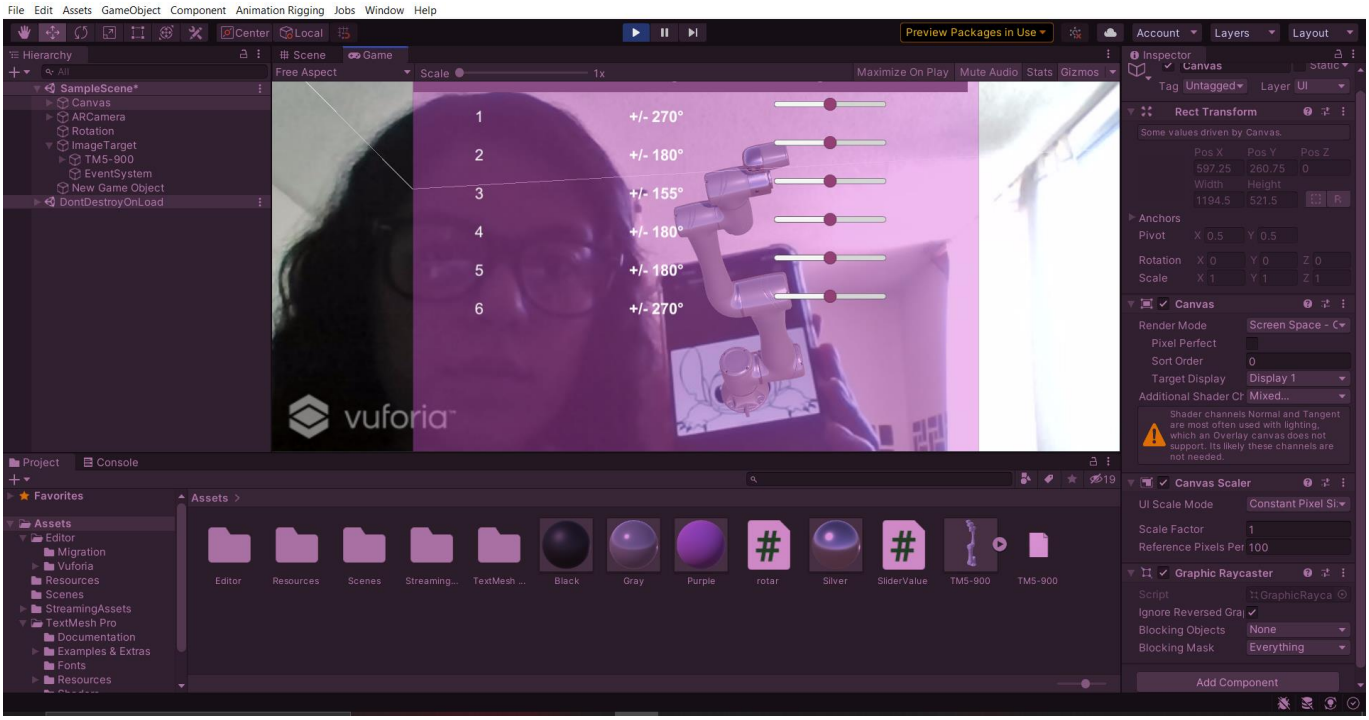

Figura 3.8: Ejecución de aplicación en Unity.

3.5 Modelado matemático

Resulta esencial exponer la cinemática directa e inversa del cobot; ya que son necesarias para crear sus rutinas a partir de su base de datos; es por eso que en seguida se presenta la metodología del modelo matemático.

Para realizar el cómputo de la cinemática, se necesita conocer tanto las medidas como los rangos de ángulos que tiene permitido cada grado de libertad, es por esto que se presentan a continuación los grados y distancias del cobot (Tabla 3.1).

Tabla 3.1: Grados y distancias del cobot.

Grados		**Distancias** *(mm)*	
q_1	*±270°*	d_1	*145.20*
q_2	*±180°*	a_2	*429*
q_3	*±155°*	a_3	*411.50*
q_4	*±180°*	d_4	*106*
q_5	*±180°*	d_5	*106*
q_6	*±270°*	d_6	*113.15*

Con el propósito de puntualizar donde se encuentra cada distancia establecida, se presenta un gráfico de las dimensiones del cobot (Figura 3.9).

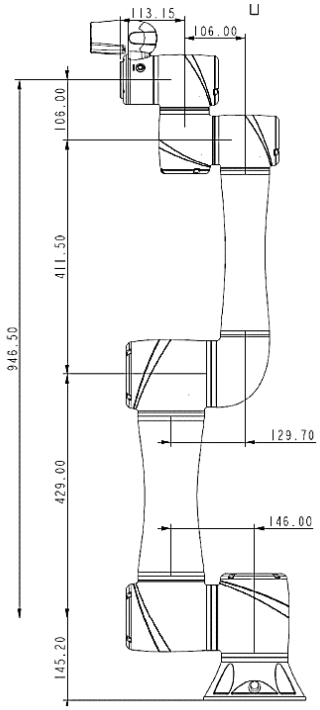

Figura 3.9: Dimensiones en milímetros de robot de 6 grados de libertad [31].

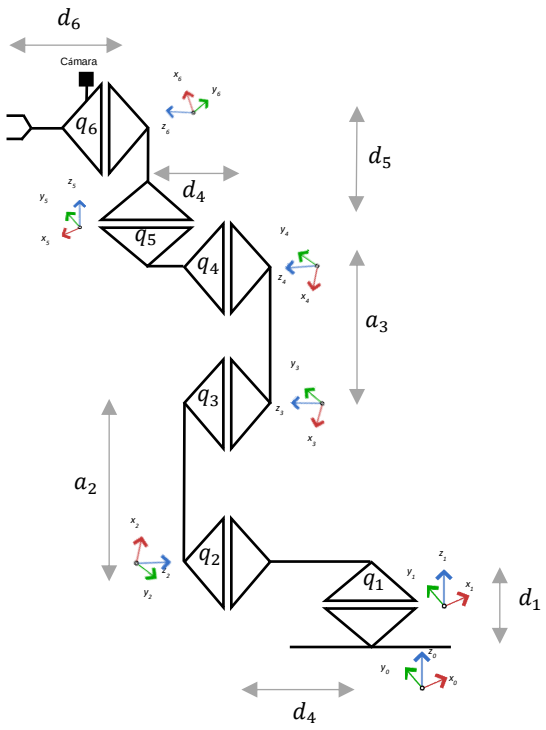

Figura 3.10: Estructura cinemática de robot de 6 grados de libertad.

Simultáneamente se elaboró un diagrama (Figura 3.10) con la intención de visualizar más fácilmente cada grado de libertad del cobot, así como sus rotaciones y todos los elementos indispensables para el desarrollo del álgebra necesaria para la obtención de la cinemática directa e inversa, cabe mencionar que existen distintos procedimientos para solucionar la cinemática directa e inversa, pero en el caso presente se hizo uso de las matrices de transformación homogénea, usando el método de la representación sistemática de Denavit Hartenberg.

Aunque la cinemática del robot se puede solucionar geométricamente, el método propuesto ofrece la ventaja de conocer tanto la posición final de manipulador como la posición de cada una de sus articulaciones [45]. Por lo tanto, el siguiente paso para la obtención de la cinemática directa e inversa, es la elaboración de una tabla (Tabla 3.2) con los parámetros de Denavit Hartenberg (DH).

Tabla 3.2: Parámetros DH.

i	α_i	a_i	d_i	θ_i	
1	$\frac{\pi}{2}$	0	d_1	θ_1	${}^{0}A_1$
2	0	a_2	0	θ_2	${}^{1}A_2$
3	0	a_3	0	θ_3	${}^{2}A_3$
4	$\frac{\pi}{2}$	0	d_4	θ_4	${}^{3}A_4$
5	$-\frac{\pi}{2}$	0	d_5	θ_5	${}^{4}A_5$
6	0	0	d_6	θ_6	${}^{5}A_6$

Donde:

- ***i:*** Articulación.
- $\boldsymbol{\alpha_i}$**:** Rotación en el eje z para coincidir con el marco de referencia de la siguiente junta.
- $\boldsymbol{a_i}$**:** Distancia de desplazamiento horizontal de la primera junta a la próxima.
- $\boldsymbol{d_i}$**:** Distancia de desplazamiento vertical (eje z) de la primera junta a la que le sigue.
- $\boldsymbol{\theta_i}$**:** Rotación en el eje z para coincidir con el marco de referencia de la posterior.

3.5.1 Cinemática Directa

La cinemática directa reside en hallar el valor de la posición final del cobot, el resultado es una función de los valores articulares, esto es el cálculo de traslación o rotación de las articulaciones [45].

$$ {}^{i-1}T_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \cos\alpha_i & \sin\theta_i \sin\alpha_i & \alpha_i \cos\theta_i \\ \sin\theta_i & \cos\theta_i \cos\alpha_i & -\cos\theta_i \sin\alpha_i & \alpha_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1) $$

Para la evaluación de ${}^{i-1}A_i$ se deben definir los parámetros de Denavit-Hartenberg, los cuales se basan exclusivamente en las características geométricas de cada enlace y los sistemas de coordenadas en cada uno. Las características de los parámetros son [45]:

- θ_i rotación alrededor del eje Z_{i-1}
- d_i translación a lo largo de $Z_{i-1\ eje}$

- a_i translación lo largo del eje X_i
- A_i rotación alrededor del eje X_i

Asimismo, es necesario definir las siguientes variables:

- r_{kj}: Representa los elementos rotacionales de la matriz de transformación.
- c_i: cos θ_i
- s_i: sin θ_i
- c_{ij}: cos $(\theta_i + \theta_j)$
- s_{ij}: sin $(\theta_i + \theta_j)$

$$ {}^{0}T_{6} = {}^{0}A_{1}\ {}^{1}A_{2}\ {}^{2}A_{3}\ {}^{3}A_{4}\ {}^{4}A_{5}\ {}^{5}A_{6} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2) $$

Reemplazando los valores de la Tabla 3.2, en la Eq. 1, tenemos [46]:

$$ r_{11} = c_1c_{234}c_5c_6 + c_6s_1s_5 - c_1s_{234}s_6 \quad (3) $$

$$ r_{21} = c_{234}c_5c_6s_1 - c_1c_6s_5 - s_1s_{234}s_6 \quad (4) $$

$$ r_{31} = c_5c_6s_{234} + c_{234}s_6 \quad (5) $$

$$ r_{12} = -c_1c_{234}c_5s_6 - s_1s_5s_6 - c_1c_6s_{234} \quad (6) $$

$$ r_{22} = -c_{234}c_5s_1s_6 + c_1s_5s_6 - c_6s_1s_{234} \quad (7) $$

$$ r_{32} = -c_5s_{234}s_6 + c_{234}c_6 \quad (8) $$

$$ r_{13} = -c_1c_{234}s_5 + c_5s_1 \quad (9) $$

$$ r_{23} = -c_{234}s_1s_5 - c_1c_5 \quad (10) $$

$$ r_{33} = -s_{234}s_5 \quad (11) $$

Las coordenadas del punto final son:

$$ p_x = -c_1c_{234}s_5d_6 + c_5s_1d_6 + c_1s_{234}d_5 + s_1d_4 + c_1c_{23}\alpha_3 + c_1c_2\alpha_2 \quad (12) $$

$$ p_y = -c_{234}s_1s_5d_6 - c_1c_5d_6 + s_1s_{234}d_5 - c_1d_4 + c_{23}s_1\alpha_3 + c_2s_1\alpha_2 \quad (13) $$

$$ p_z = -s_{234}s_5d_6 - c_{234}d_5 + s_{23}\alpha_3 + s_2\alpha_2 + d_1 \quad (14) $$

3.5.2 Cinemática Inversa

En la cinemática inversa de un robot se obtienen los valores articulares (ángulos de las juntas), necesario para posicionar un punto en el espacio referenciado al sistema de coordenadas global del cobot [45].

Como primer paso se debe encontrar q_1, q_2, q_3, tal que el centro de la muñeca sea localizado. Posteriormente, con las variables calculadas en el primer paso, evaluar R_0^3, y finalmente se encuentra el conjunto de ángulos de Euler correspondiente a la matriz de rotación.

Las ecuaciones desarrolladas aplicando los pasos mencionados [46] se muestran a continuación, cabe mencionar que *K, B, C* y *D*, son variables auxiliares para simplificar las ecuaciones.

$$A_1=p_x-d_6r_{13} \tag{15}$$

$$B_1=d_6r_{23}-p_x \tag{16}$$

$$\theta_1=atan2(A_1, B_1)\pm atan2\left(\sqrt{A_1^2+B_1^2-d_4^2}, d_4\right) \tag{17}$$

$$c_5=s_1r_{13}-c_1r_{21} \tag{18}$$

$$s_5=\sqrt{(s_1r_{11}-c_1r_{21})^2+(s_1r_{12}-c_1r_{22})^2} \tag{19}$$

$$\theta_5=\pm atan2(s_5,\ c_5) \tag{20}$$

$$c_6=\frac{s_1r_{11}-c_1r_{21}}{s_5} \tag{21}$$

$$s_6=\frac{c_1r_{22}-s_1r_{12}}{s_5} \tag{22}$$

$$\theta_6=atan2(s_6, c_6) \tag{23}$$

$$A_{234}= c_5c_6 \tag{24}$$

$$B_{234}=s_6 \tag{25}$$

$$C_{234}= c_1r_{11}+s_1r_{21} \tag{26}$$

$$D_{234}= r_{31} \tag{27}$$

$$\theta_{234}=atan2(A_{234}D_{234}-B_{234}C_{234}, A_{234}C_{234}+B_{234}D_{234}) \tag{28}$$

$$K_S=p_z-d_1+c_{234}d_5+s_{234}s_5d_6 \tag{29}$$

$$K_c=c_1p_x+s_1p_y-s_{234}d_5+c_{234}s_5d_6 \tag{30}$$

$$c_3=\frac{K_s^2+K_c^2-\alpha_2^2-\alpha_3^2}{2\alpha_2\alpha_3} \tag{31}$$

$$s_3=\sqrt{1-c_3^2} \tag{32}$$

$$\theta_3=\pm atan2(s_3,c_3) \tag{33}$$

$$\theta_2=atan2(K_S, K_c)-atan2(\alpha_3s_3, \alpha_3c_3+\alpha_2) \tag{34}$$

$$\theta_4=\theta_{234}-\theta_2-\theta_3 \quad (35)$$

Para verificar la metodología utilizada y las ecuaciones calculadas [47], las ecuaciones se implementaron en la plataforma de programación y cálculo numérico MATLAB®, donde se obtuvo el siguiente gráfico, ver Figura 3.11.

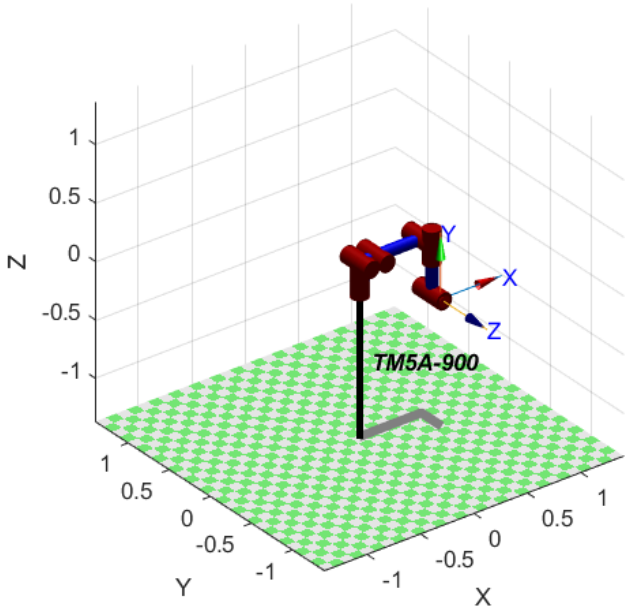

Figura 3.11: Gráfico obtenido con Matlab de cobot TM5-900.

Con este gráfico (Figura 3.11), podemos validar que las ecuaciones desarrolladas son correctas, ya que se puede ver la configuración cinemática del robot donde se describen los marcos de referencia, así como el marco de referencia de la posición del efector final.

3.6 Base de datos

Posteriormente se procedió a la creación de base de datos, donde como primer paso se instaló XAMPP (Figura 3.12), el cual es un paquete de instalación independiente de plataforma, software libre, que consiste principalmente en el sistema de gestión de bases de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script: PHP y Perl [48].

Figura 3.12: Software de sistema de gestión de bases de datos MySQL.

Para acceder a phpMyAdmin (Figura 3.13) desde XAMPP, es necesario tener Apache y MySQL ejecutándose a la par en el panel de control de XAMPP. Para levantar los servicios de XAMPP, nos conducimos a abrir phpMyAdmin escribiendo: http://localhost:81/phpmyadmin/index.php?route=/database/structure&server=1&db=robot en el navegador, donde se creó la base de datos.

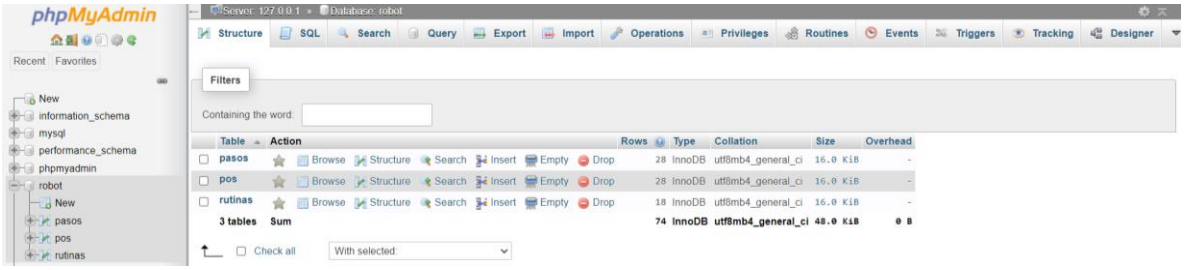

Figura 3.13: Interfaz de phpmyAdmin.

Inicialmente al nombre de la base de datos se designó como “robot”, para después crear tres tablas (Figura 3.14) con los siguientes nombres:

- **Pos:** Esta tabla se encarga de guardar las posiciones de los ángulos del robot donde se encuentra posicionado en ese momento.
- **Rutinas:** Se ocupa de almacenar todas las rutinas que son creadas.
- **Pasos:** Realiza el desglose de los pasos de cada una de las rutinas que se generen.

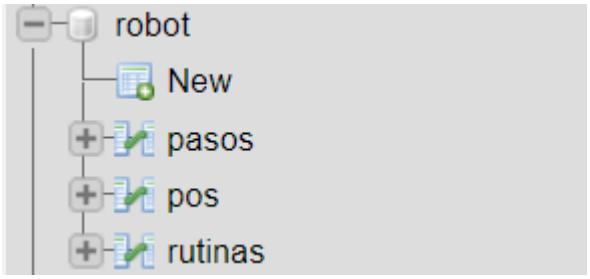

Figura 3.14: Diagrama de árbol de base de datos.

En diagrama de la Figura 3.15, que se observa al término de este párrafo; muestra el modelo entidad relación de la base de datos, la cual traza el flujo de la información, donde

se almacena una rutina, y ésta puede tener varios pasos y a la vez, cada uno de los pasos tiene una serie de ángulos.

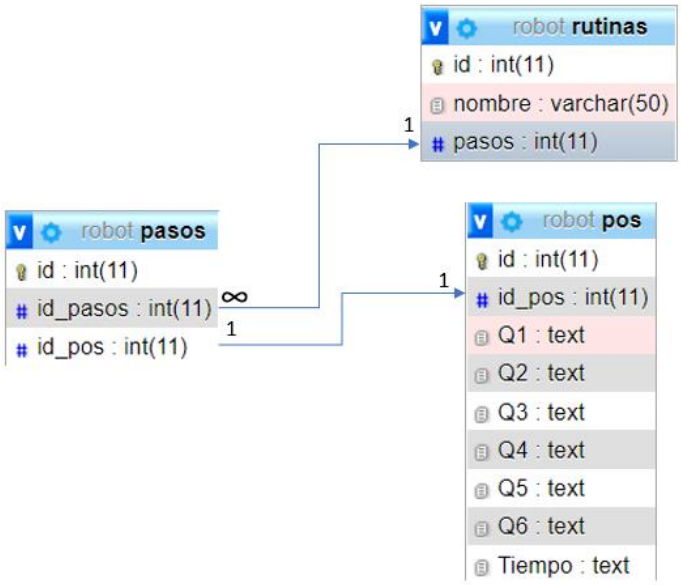

Figura 3.15: Entidad relación de base de datos.

En el esquema que se encuentra en seguida (Figura 3.16), se analiza cómo funciona la consulta de la base de datos, es decir, primero se selecciona la rutina, después la tabla de pasos y finalmente la posición.

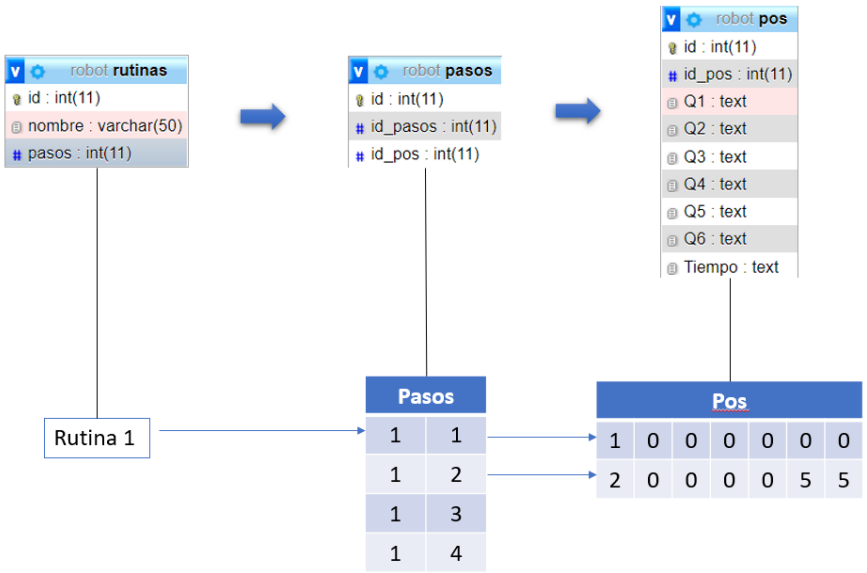

Figura 3.16: Consulta de rutina en base de datos.

Con la culminación de las tablas, la base de datos está lista para iniciar a utilizarla.

3.6.1 Enlace de base de datos con Unity

En la siguiente sección se describe como se ejecutó la conexión entre la base de datos y Unity, ya que es donde se encuentra el diseño virtual del cobot.

Se inicia con la elaboración de archivos para consultar la base de datos en formato php (Figura 3.17).

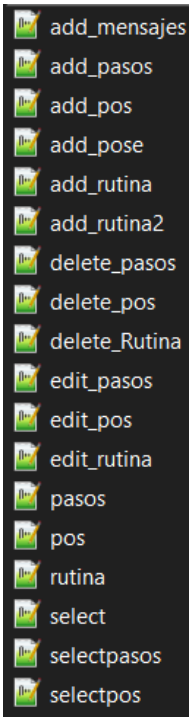

Figura 3.17: Script para base de datos php.

Para esto se creó un archivo de inserción, el cual tiene la finalidad de ingresar los registros en la base de datos (Figura 3.18).

```php
<?php
  require_once('includes/load.php');
  $page_title = 'Agregar pasos';
?>
<?php
  if(isset($_POST['add_pasos'])){

    $req_fields = array('id_pasos','id_pos');
    validate_fields($req_fields);

    if(empty($errors)){
           $id_pasos   = remove_junk($db->escape($_POST['id_pasos']));
       $id_pos   = $db->escape($_POST['id_pos']);

        $query = "INSERT INTO pasos (";
        $query .="id_pasos,id_pos";
        $query .=") VALUES (";
        $query .=" '{$id_pasos}', '{$id_pos}'";
        $query .=")";
        if($db->query($query)){
          //sucess
          $session->msg('s'," Se agrego los pasos");
          redirect('pasos.php', false);
        } else {
          //failed
          $session->msg('d',' No se pudo crear los pasos.');
          redirect('add_pasos.php', false);
        }
    } else {
      $session->msg("d", $errors);
       redirect('add_pasos.php',false);
    }
  }
?>
<div class="row">
  </div>

  <div class="card shadow mb-4">
  <div class="card-header py-3">
    <h6 class="m-0 font-weight-bold text-primary">
        Agregar pasos
    </h6>
    </div>
    <div class="card-body">
```

PHP Hypertext Preprocessor file length : 1,815 lines : 63 Ln : 1 Co

Figura 3.18: Fracción de generación de script “add_pasos.php”.

Después se efectuó la actualización y consulta de base de datos (Figura 3.19) mediante los Scripts php (add, delete y edit), es decir, altas, bajas y cambios (ABC).

Cabe mencionar que cada archivo php hace una conexión a la base de datos y a Unity, primero Unity solicita un “Web Request”, posteriormente el “Web request” al servidor local que en este caso se realizó con XAMPP y este ejecuta los archivos php que hacen la consulta de la base de datos de MySQL.

admin.cs CalclKsldr.cs Consulta1.cs

```
240   public void SendposeBtn()
241   {
242       int i = Int32.Parse(inicio.text);
243       int f = Int32.Parse(fin.text);
244       if (i >= f)
245       {
246           StartCoroutine(PostScores(Q1.text, Q2.text, Q3.text, Q4.text, Q5.text, Q6.text,tiempo.text, inicio.text));
247           v1.gameObject.SetActive(true);
248           v3.gameObject.SetActive(false);
249       }
250       else
251       {
252           StartCoroutine(PostScores(Q1.text, Q2.text, Q3.text, Q4.text, Q5.text, Q6.text,tiempo.text, inicio.text));
253           i += 1;
254           inicio.text = i.ToString();
255       }
256   }
257
258   public IEnumerator PostScores(string Q1, string Q2, string Q3, string Q4, string Q5, string Q6, string tiempo, string inicio)
259   {
260       string post_url = "http://localhost:81/robot/add_pose.php?Q1=" + UnityWebRequest.EscapeURL(Q1) + "&Q2="
261             + UnityWebRequest.EscapeURL(Q2) + "&Q3="
262             + UnityWebRequest.EscapeURL(Q3) + "&Q4="
263             + UnityWebRequest.EscapeURL(Q4) + "&Q5="
264             + UnityWebRequest.EscapeURL(Q5) + "&Q6="
265             + UnityWebRequest.EscapeURL(Q6) + "&tiempo="
266            + UnityWebRequest.EscapeURL(tiempo) + "&inicio="
267               + UnityWebRequest.EscapeURL(inicio);
268       Debug.Log(post_url);
269       UnityWebRequest hs_post = UnityWebRequest.Get(post_url);
270       yield return hs_post.SendWebRequest();
271       if (hs_post.error != null)
272           Debug.Log("error: "
```

Figura 3.19: Script de Unity para conexión de base de datos.

3.7 Optimización de Canvas en Unity.

Se diseñaron Canvas en Unity para poder operar al cobot TM5-900, buscando que sea intuitiva para el usuario (Figura 3.20).

La ventana 1 es la primera ventana que el usuario visualizará. En este canvas:

- Se agregan nuevas rutinas para el cobot.
- Se actualizan las rutas recién agregadas.
- Se direcciona a visualizar la rutina seleccionada por el usuario.

Figura 3.20: Ventana 1 en Unity de interfaz de usuario.

En la ventana 2 (Figura 3.21), le permite al usuario:

- **Nombre**: Agregar nombre personalizado a la nueva rutina que está insertando.
- **Número de pasos:** Poner el número de pasos que tendrá la nueva rutina.
- **Regresar:** Regresa a la ventana 1.
- **Guardar:** Guarda el nombre y número de pasos en la base de datos.

Este Canvas es mostrado al dar clic en el botón “Agregar nueva ruta” de la ventana 1.

Figura 3.21: Ventana 2 en Unity de interfaz de usuario.

En la ventana 3 (Figura 3.22) el usuario selecciona la posición de cada grado de libertad del robot a través de los Sliders, ya sea mediante la columna de Sliders de cinemática directa o inversa; de igual forma en esta ventana es posible configurar los siguientes elementos:

- **Segundos:** Se inserta el tiempo que quiere que tarde en realizar la pose.
- **Regresar:** Regresa a la ventana 2.
- **Guardar:** Guarda la posición de cada uno de los seis grados de libertad que el usuario seleccionó en cada número de pose, así como el tiempo insertado.

Este Canvas se visualiza al dar clic en el boto de “Guardar” de la ventana 2.

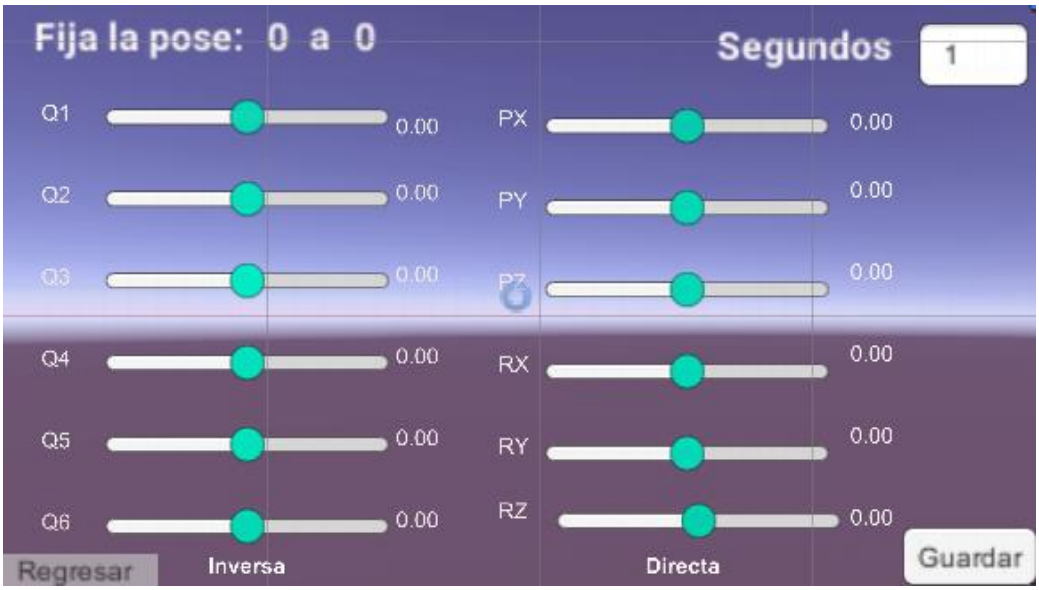

Figura 3.22: Ventana 3 en Unity de interfaz de usuario.

En la ventana 4 (Figura 3.23) se observa con realidad aumentada la rutina del robot que el usuario seleccionó e insertó, con las siguientes opciones:

- **Regresar:** Regresa a la ventana 1.
- **Parar:** Pausa la rutina en el momento que el usuario lo desee y al dar nuevamente clic en el botón se restablece la reproducción en el tiempo donde se dejó pausada.
- **Reproducir:** Visualiza nuevamente la rutina completa del cobot.

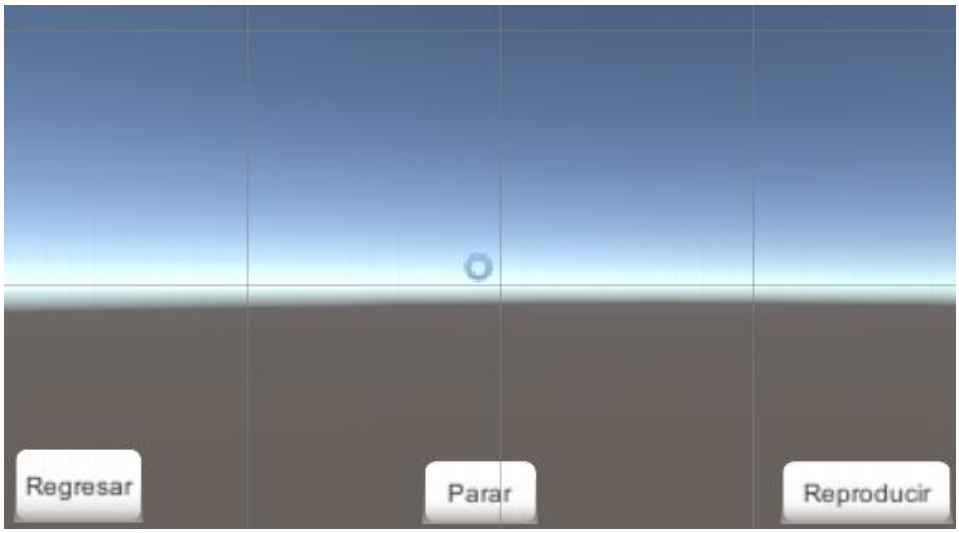

Figura 3.23: Ventana 4 en Unity de interfaz de usuario.

Así es como se observa la interfaz de “Proyecto” de Unity, con los Scripts creados (Figura 3.24).

Figura 3.24: Ventana de proyecto en Unity.

En el esquema que se observa seguidamente (Figura 3.25), se describe paso a paso como es el funcionamiento de creación de rutinas, es decir, lo que recientemente se acaba de describir mediante la generación de “rutina 7”.

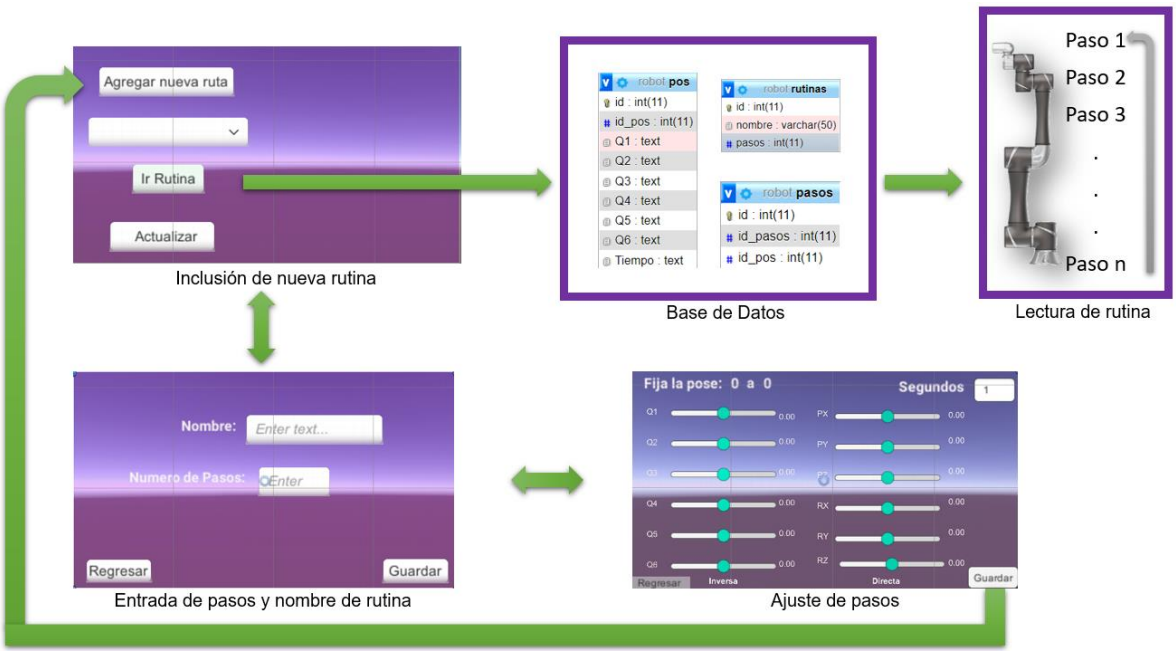

Figura 3.25: Esquema de ingreso de rutinas.

3.8 Comunicación de robot físico y virtual

Para efectuar la manipulación remota del robot físico mediante un dispositivo móvil, es necesario desarrollar una aplicación, la cual se realizó con el software de Unity.

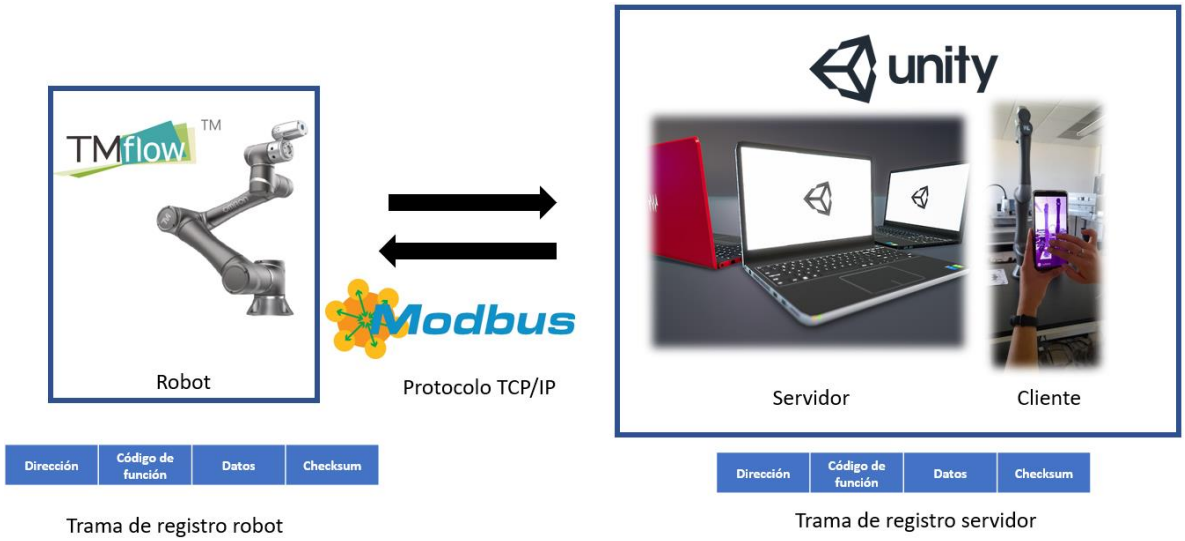

Figura 3.26: Esquema de comunicación entre robot físico y virtual.

De igual manera para lograr operar el robot tanto virtual como físico, es imprescindible comunicarlos entre sí (Figura 3.26). Para esto se utilizó el protocolo de comunicación abierto denominado Modbus, así mismo se utilizó EasyModbus (Figura 3.27), el cual

permite enviar y recibir comandos sin programación compleja, es decir, solo se necesitan unas pocas líneas de códigos para leer o escribir datos desde o hacia un dispositivo [49].

Figura 3.27: Modbus Client.

Este protocolo se empleó vía cliente, con el cual se estuvieron realizando pruebas y visualizar los datos que arroja el robot. Modbus Server Simulator hace que el desarrollo de software sea rápido y fácil.

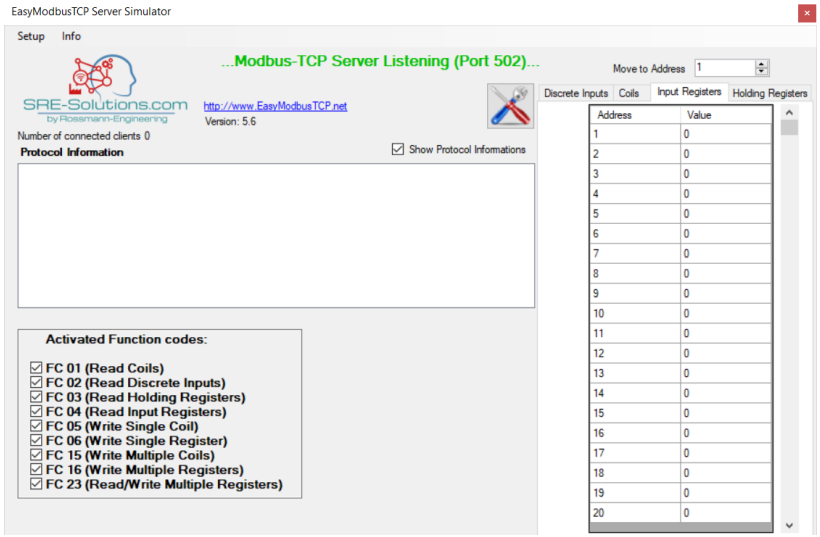

Figura 3.28: Modbus TCP Server.

3.8.1 Configuración de comunicación en TMFlow®.

A continuación, se presenta la configuración TCP de Modbus, donde básicamente su funcionamiento depende de leer las coordenadas de direcciones del robot (Figura 3.29) establecidas por el fabricante del robot.

Robot Coordinate	FC	$Address_{10}$	$Address_{16}$	Type	R/W	Note1	Note2
X (Cartesian coordinates)	04	7001~7002	1B59~1B5A	Float	R	Dword	mm
Y (Cartesian coordinates)	04	7003~7004	1B5B~1B5C	Float	R	Dword	mm
Z (Cartesian coordinates)	04	7005~7006	1B5D~1B5E	Float	R	Dword	mm
Rx (Cartesian coordinates)	04	7007~7008	1B5F~1B60	Float	R	Dword	degree
Ry (Cartesian coordinates)	04	7009~7010	1B61~1B62	Float	R	Dword	degree
Rz (Cartesian coordinates)	04	7011~7012	1B63~1B64	Float	R	Dword	degree
Joint 1	04	7013~7014	1B65~1B66	Float	R	Dword	degree
Joint 2	04	7015~7016	1B67~1B68	Float	R	Dword	degree
Joint 3	04	7017~7018	1B69~1B6A	Float	R	Dword	degree
Joint 4	04	7019~7020	1B6B~1B6C	Float	R	Dword	degree
Joint 5	04	7021~7022	1B6D~1B6E	Float	R	Dword	degree
Joint 6	04	7023~7024	1B6F~1B70	Float	R	Dword	degree

Robot Stick	FC	$Address_{10}$	$Address_{16}$	Type	R/W	Note
Project Running Speed	04	7101	1BBD	Int16	R	%
M/A Mode	04	7102	1BBE	Int16	R	A:1; M:2
Play/Pause	05	7104	1BC0	Bool	W	
Stop	05	7105	1BC1	Bool	W	Bottom down: 1
Stick+	05	7106	1BC2	Bool	W	Bottom up: 0
Stick-	05	7107	1BC3	Bool	W	

Figura 3.29: Coordenadas del Robot TM en lista de Modbus.

Estas direcciones son necesarias para poder interpretar las coordenadas del robot mediante TMflow®, el cual cuenta con un bloque para estructurar los dispositivos Modbus (Figura 3.30).

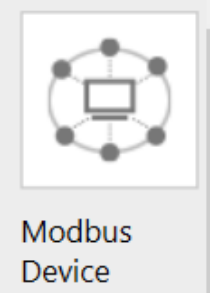

Figura 3.30: Bloque Modbus en TMflow®.

Dentro de esta interfaz se configuran los parámetros relevantes para el dispositivo TCP, como se muestra en la Figura 3.31.

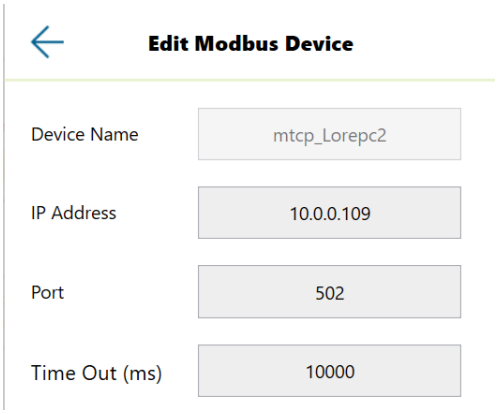

Figura 3.31: Configuración de parámetros para el para el dispositivo TCP.

La dirección IP se obtuvo mediante el Símbolo del sistema de Windows (CMD), la cual es la que se va a conectar al robot (Figura 3.32); esta dirección es necesario establecerla en el script de Unity.

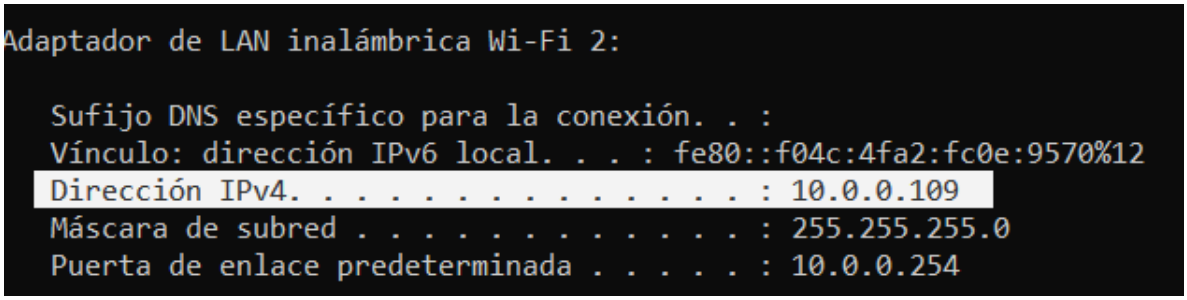

Figura 3.32: Dirección IP local de servidor.

Una vez completada la configuración, se agrega la ubicación previa a la lectura y escritura del dispositivo (Figura 3.33), es decir, se agrega la dirección de entrada correspondiente a cada joint del robot, conforme a la lista de direcciones, así como el tipo de variable de cada grado de libertad, el cual corresponde a flotante.

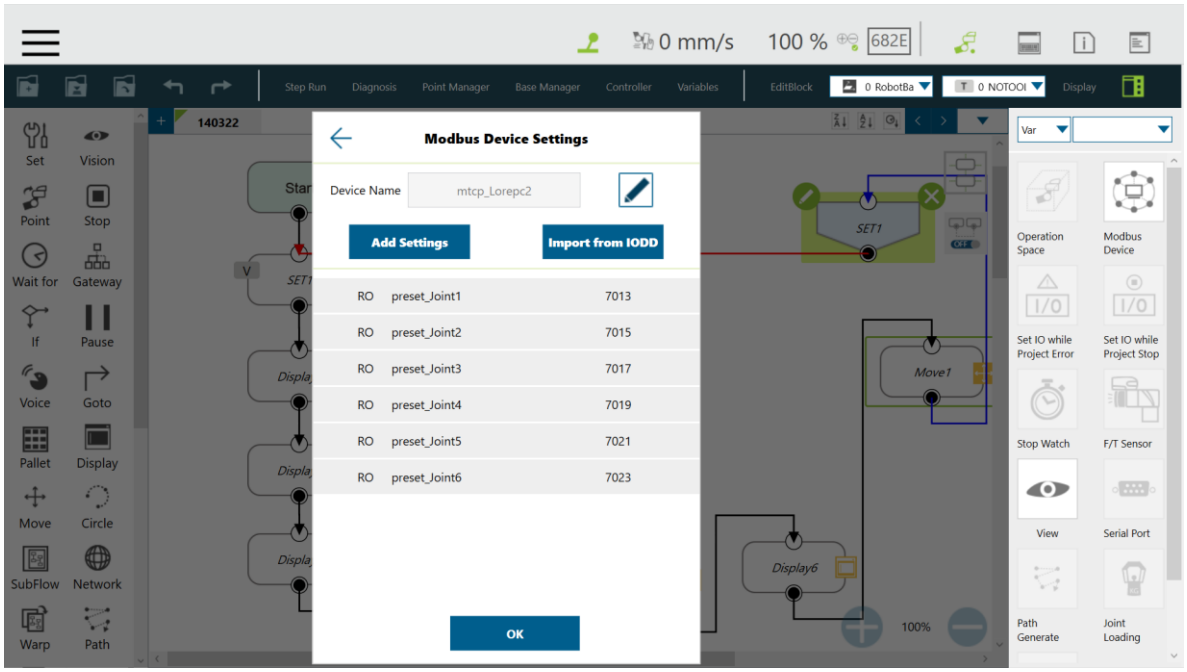

Figura 3.33: Configuracion de direccion de entradas MODBUS.

Posteriormente se usará esta configuración para programar el flujo y leer la posición de las coordenadas actuales del robot. Para esto es necesario crear variables de almacenamiento del valor de las coordenadas en el registro (Figura 3.34). Cabe

mencionar que Big-endian es el byte alto almacenado en la dirección de memoria más baja, por lo cual es necesario selecciona esta opción al agregar la variable de cada joint.

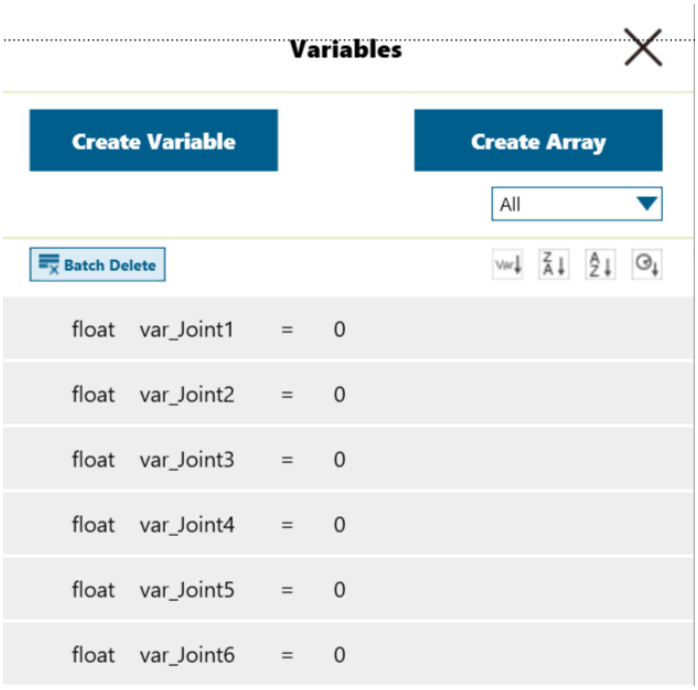

Figura 3.34: Creación de variables para cada grado de libertad.

Las variables var_Joint# de tipo float, obtiene el valor de las coordenadas del robot en su respectiva dirección de cada grado de libertad.

Entonces, para insertar las nuevas variables y las variables obtenidas por Modbus, así como verificar si el valor de cada coordenada obtenida en la dirección de Modbus es correcta, es necesario realizar la edición que se muestra en la Figura 3.35.

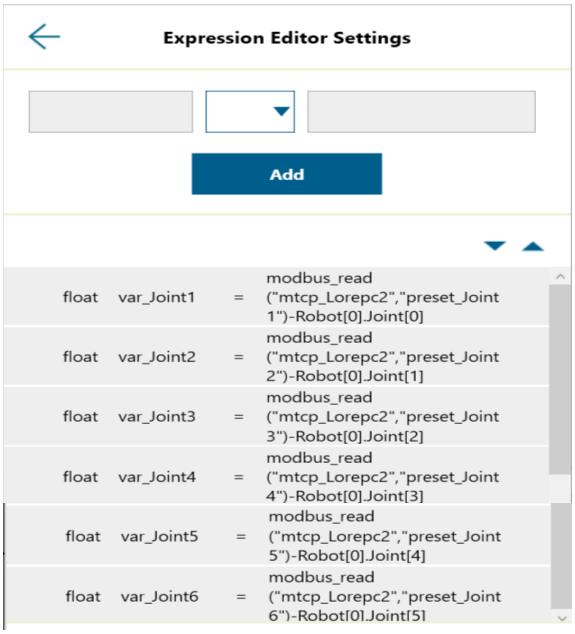

Figura 3.35: Variable obtenida del nodo ESTABLECER para obtener el valor de Modbus.

Para lograr hacer esta configuración, es necesario seleccionar el protocolo Modbus, así como el dispositivo local que se está utilizando, para finalizar con la selección de las variables creadas anteriormente para los 6 grados de libertad (Figura 3.36).

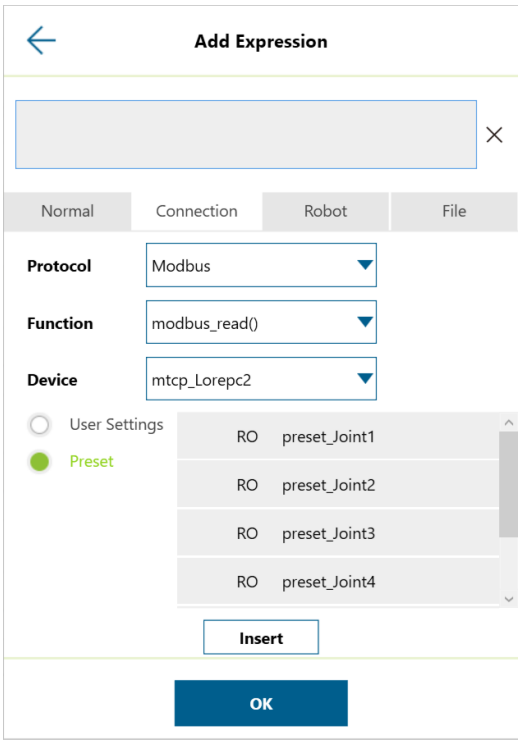

Figura 3.36: Creación de expresiones para variables de grados de libertad.

Dentro de TMflow®, existe un nodo llamado “Conexión” el cual tiene como objetivo realizar la configuración de red del robot físico (Figura 3.37).

Connection

Figura 3.37: Módulo de conexión en TMflow®.

Con el fin de que el robot se comunique con los dispositivos externos a través de la red, se edita el módulo con la dirección IP del dispositivo local (Figura 3.38).

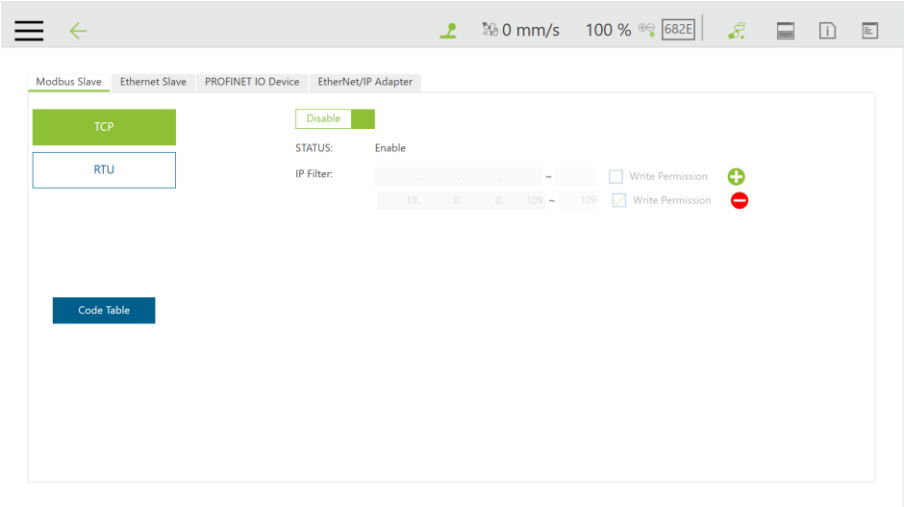

Figura 3.38: Interfaz de configuración IP.

3.9 Programación en TMflow®

En el presente apartado, se presenta el diagrama de bloques que se desarrolló en TMflow®, para lograr la comunicación con Unity, así como mostrar los valores obtenidos. Como se puede ver en la Figura 3.39; es importante mencionar que existe un bloque denominado “SET1” el cual se encarga de establecer los estados de E/S y cambiar el tipo y el valor de las variables. Cuando se atraviesa este nodo, todos los parámetros del nodo cambian al resultado establecido.

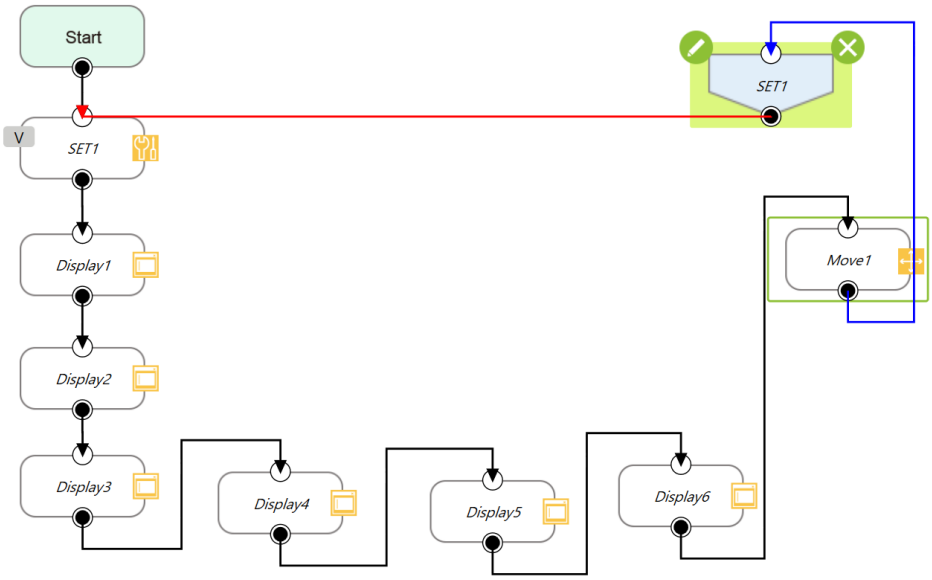

Figura 3.39: Diagrama de bloques en TMflow®.

En el nodo “Mover”, se establecieron los ángulos de seis ejes J1~J6 para determinar la distancia y el ángulo de movimiento del robot, y luego realizar el movimiento relativo desde la posición actual (Figura 3.40).

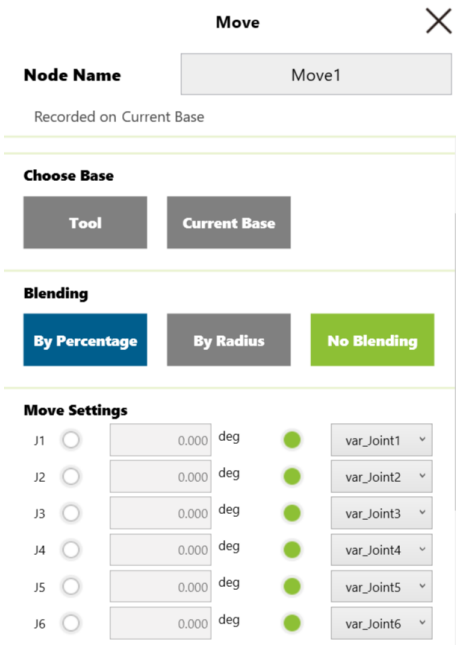

Figura 3.40: Configuración del nodo “Mover”.

La función del nodo “Pantalla” es mostrar la variable o cadena especificada en la pantalla conforme al formato especificado por los usuarios. En este caso se utiliza para mostrar

el valor obtenido por el puerto serie, los parámetros del robot y los resultados de la ejecución (Figura 3.41).

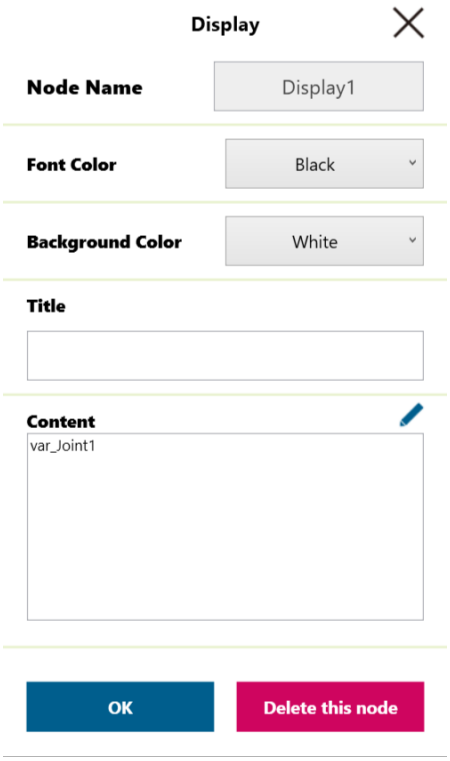

Figura 3.41: Módulo “Pantalla” con variable var_Joint1.

Por último, el valor obtenido se muestra en la pantalla mediante el nodo “Pantalla” y el nodo “Ir a”, se utiliza para actualizar el valor continuamente; en la Figura 3.42 se observa un ejemplo.

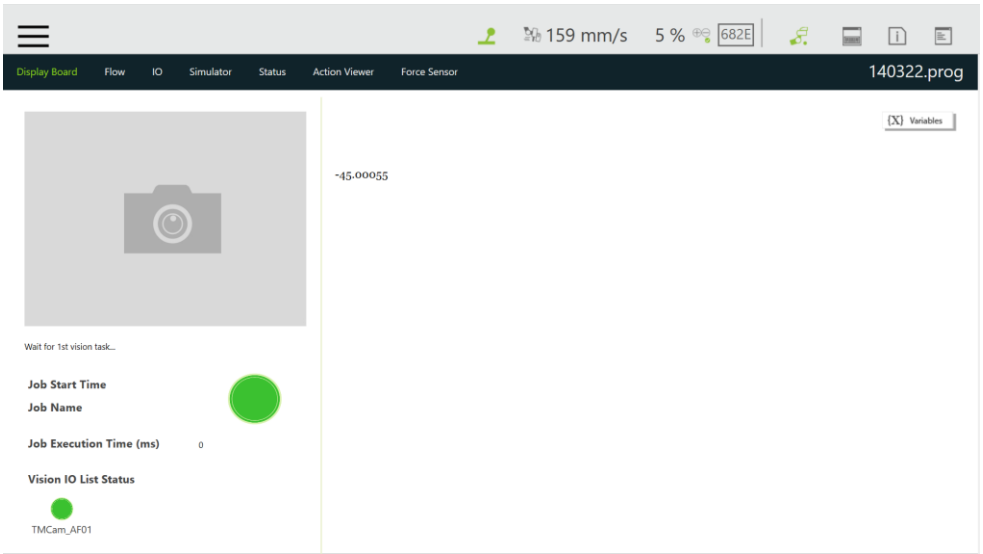

Figura 3.42: Valor obtenido mediante el nodo pantalla.

3.9.1 Configuración de comunicación en Unity.

El modelo virtual del robot TM5-900 se realizó en Unity, motivo por el cual es necesario lograr la comunicación con este software mediante el protocolo de comunicación Modbus. Para lograr esto se agregó la librería de EasyModbus (Figura 3.43) dentro de la carpeta “Assets”de Unity.

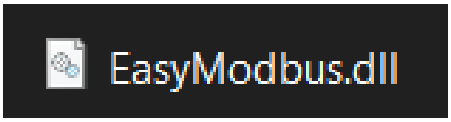

Figura 3.43: Librería EasyModbus.

De igual manera dentro del script de Unity, se mandó llamar la librería para poder utilizarla dentro del script (Figura 3.44).

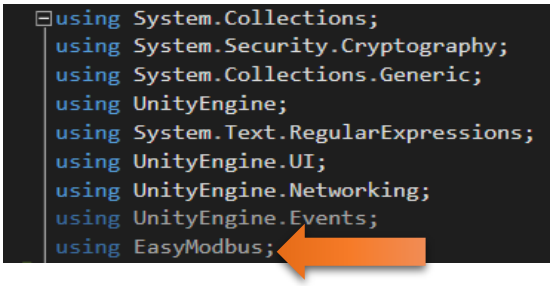

Figura 3.44: Librería EasyModbus en Script.

Seguidamente, se desarrolló dentro del mismo script un conversor de bytes (Figura 3.45), ya que el robot arroja y recibe datos en formato Big endian, es decir, el byte más alto (más significativo) se guarda en la dirección más baja [50].

```
CalclKsldr.cs   collicion.cs   cambioslider.cs   Conversor1.cs   Consulta1.cs
Assembly-CSharp                                  Conversor1
                Script de Unity | 14 referencias
6   public class Conversor1: MonoBehaviour
7   {
                8 referencias
8       public enum RegisterOrder { LowHigh = 0, HighLow = 1 };
                0 referencias
9       public string ToHexString(float f)
10      {
11          var bytes = BitConverter.GetBytes(f);
12          var i = BitConverter.ToInt32(bytes, 0);
13          String valor = i.ToString("X8");
14          //        UnityEngine.Debug.Log("el valor toHexString :" + valor);
15          return valor;
16      }
                0 referencias
17      public int[] float2holding(float v,bool isBigEndian) {
18          var bytes = BitConverter.GetBytes(v);
19          for (int i = 0; i < 4; i++) {
20              Debug.Log($"{i}:{bytes[i].ToString()}");
21          }
22          if (isBigEndian)
23          {
24              //big endian
25              byte[] High = new byte[]{
26              bytes[0],bytes[1]
27          };
28              byte[] Low = new byte[]{
29              bytes[2],bytes[3]
30          };
31              Int16 iHigh = BitConverter.ToInt16(High, 0);
32              Int16 Low1 = BitConverter.ToInt16(Low, 0);
33              UnityEngine.Debug.Log("High :" + iHigh.ToString());
34              UnityEngine.Debug.Log("Low :" + Low1.ToString());
98 %    No se encontraron problemas.
```

Figura 3.45: Script de conversión de bytes en Unity.

Con el objetivo de lograr esta conversión de bytes a Big endian, fue necesario convertirlos a hexadecimal y luego a flotantes, ya que Unity recibe los datos en números flotantes. En la Figura 3.46, se observa un diagrama para una comprensión más gráfica de lo anteriormente descrito.

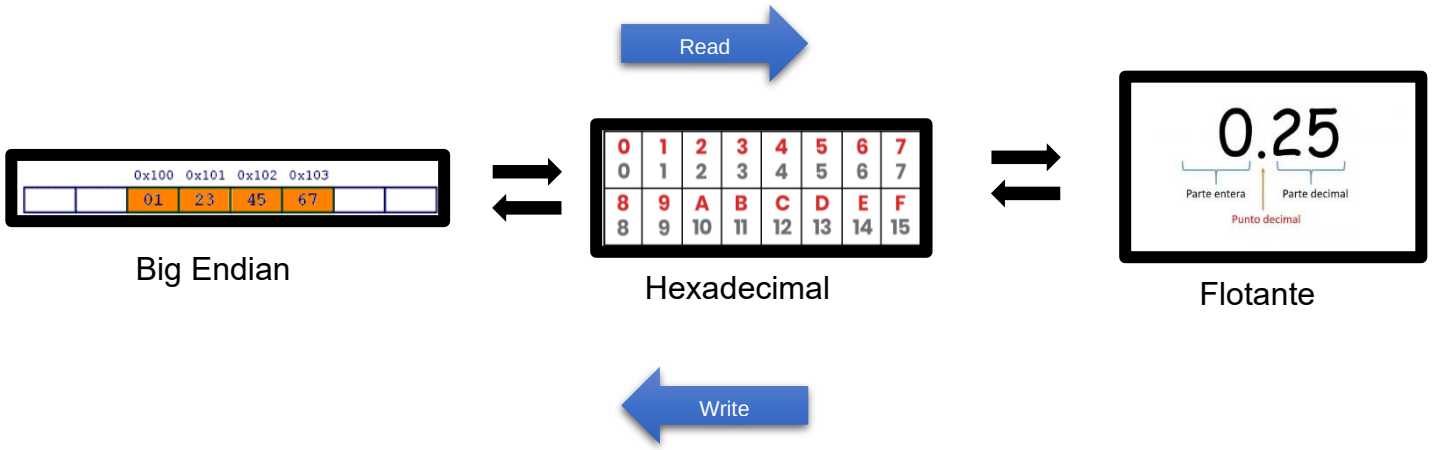

Figura 3.46: Conversión de datos

Posteriormente, se desarrolló y añadió código al script “CalclKsldr”, con el objetivo de poder escribir y leer en el robot (Figura 3.47), es decir mandar los puntos de cada grado de libertad que el usuario elija en la aplicación al robot real, así como también enviar las rutinas y leer la posición actual del robot físico.

CalclKsldr.cs | collicion.cs | cambioslider.cs | Conversor1.cs | Consulta1.cs

Assembly-CSharp | CalclKsldr

```
192    public void RobotFisico() {
193
194        //var ActualTime= Time.realtimeSinceStartup
195        //UnityEngine.Debug.Log($"Tiempo para conexion: {ActualTime}");
196        ModbusClient modbusClient = new ModbusClient("10.0.0.185", 502);
197        modbusClient.Connect();
198        //            modbusClient.WriteSingleRegister(19, 4711);
199        int[] registros = modbusClient.ReadInputRegisters(7013, 12);
200        Texto.text = "";
201        a1 = modbusClient.receiveData;
202        modbusClient.Disconnect();
203        Conversor1 c = new Conversor1();
204        for (int i = 0; i < 12; i++) {
205            if (registros[i] < 0) registros[i] = 65536 + registros[i];
206            UnityEngine.Debug.Log("valor:" + i + " ->" + registros[i]);
207        }
208       int[] a = new int[] { registros[0], registros[1] };
209        L1 =c.Leer(a);
210        a = new int[] { registros[2], registros[3] };
211        L2 = c.Leer(a);
212        a = new int[] { registros[4], registros[5] };
213        L3 = c.Leer(a);
214        a = new int[] { registros[6], registros[7] };
215        L4 = c.Leer(a);
216        a = new int[] { registros[8], registros[9] };
217        L5 = c.Leer(a);
218        a = new int[] { registros[10], registros[11] };
219        L6 = c.Leer(a);
220
221        if (!double.IsNaN(L1))
222            Joints[0].transform.localEulerAngles = new Vector3(0, L1, 0);
223        if (!double.IsNaN(L2))
```

Figura 3.47: Script de lectura y escritura Modbus en Unity.

3.10 Ejecución de lectura y escritura en robot virtual y físico

Una vez realizada la configuración de parámetros para la lectura y escritura tanto en Unity como en TMflow®, se procedió a realizar sus respectivas pruebas y de esta manera comprobar su correcto funcionamiento.

En la Figura 3.48, podemos observar la interfaz de EasyModbus Client ejecutándose, es decir, se visualiza la conexión en la dirección IP 10.0.0.185 en el puerto 502; esta dirección corresponde a la del robot físico. De igual manera podemos ver que tiene una conexión correcta de servidor, así como los bytes que está escribiendo.

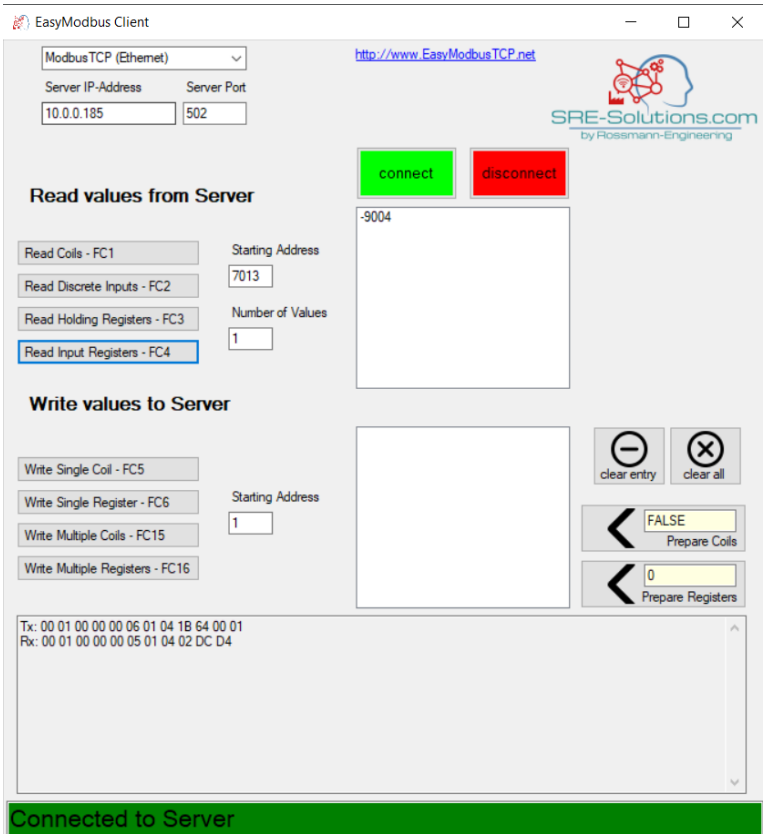

Figura 3.48: Modbus Client en ejecución.

En la Figura 3.49, se contempla la ventana de EasyModbus TCP server ejecutándose de manera correcta, y es en la pestaña de “Holding Registers” donde se observan los datos que se están escribiendo en cada grado de libertad conforme a la dirección que le corresponde según la imagen denominada “Coordenadas del Robot TM en la lista de Modbus”.

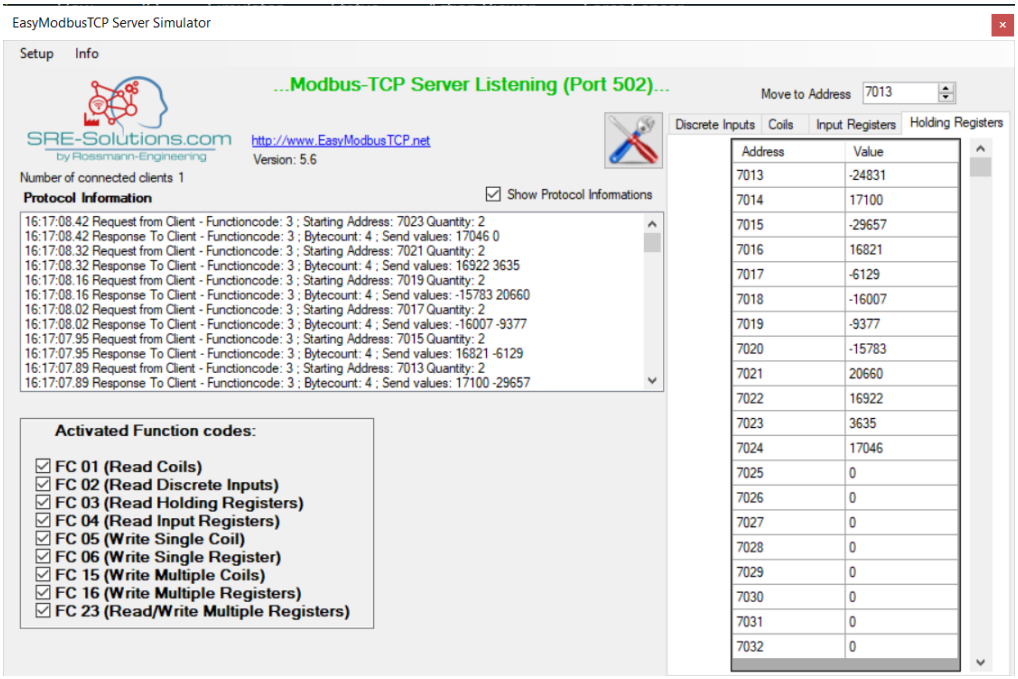

Figura 3.49: Modbus TCP Server en ejecución.

El servidor actúa de depósito de datos y funciona como un sistema gestor de base de datos o aplicaciones [51]. Un servidor hace referencia a un proveedor de servicios, este servidor es la aplicación generada a través de Unity, la cual envía información a los demás agentes de la red [52].

En la Figura 3.50, se observa la interfaz de la aplicación; es en la ventana tres donde se establece la posición del robot en cada grado de libertad, y del lado derecho se muestra el módulo “Controlador”, el cual proporciona a los usuarios control directo del robot desde el control de movimiento, el control de E/S y la configuración FreeBot. El control de movimiento incluye tres pestañas: Articulación, Base y Herramienta, que se corresponden, respectivamente con “Movimiento conforme a la configuración del ángulo de la articulación”, “Movimiento conforme a la base del robot o a la base actual” y “Movimiento conforme a la coordenada de la herramienta”.

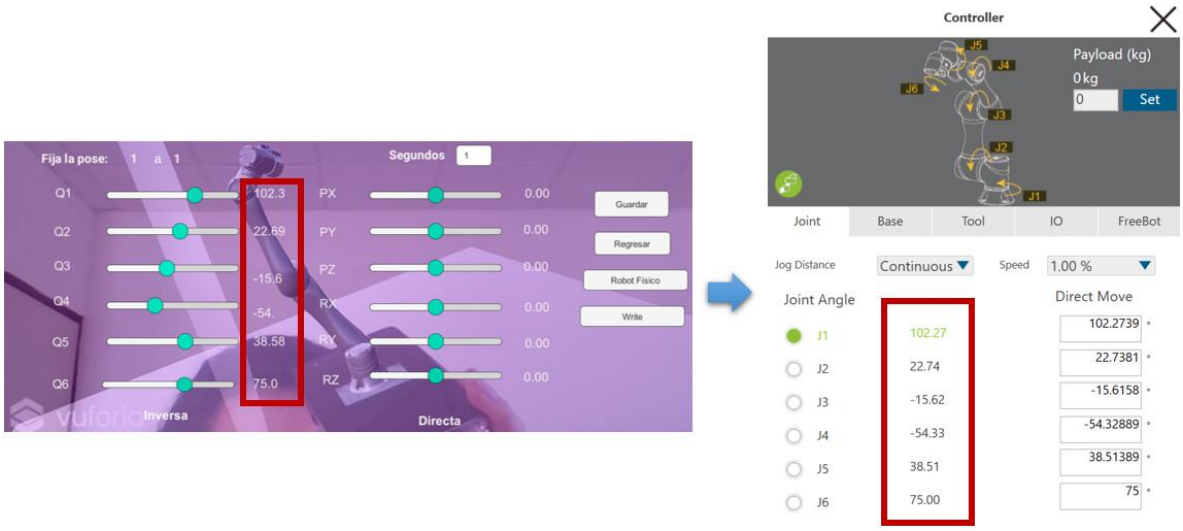

Figura 3.50: Comprobación de datos escritos con el módulo “Controlador”.

Con el módulo “Controlador” comprobamos que los grados establecidos en cada grado de libertad en Unity han sido escritos correctamente en el robot físico (Figura 3.51) ya que coinciden los valores como las posiciones del cobot físico y virtual.

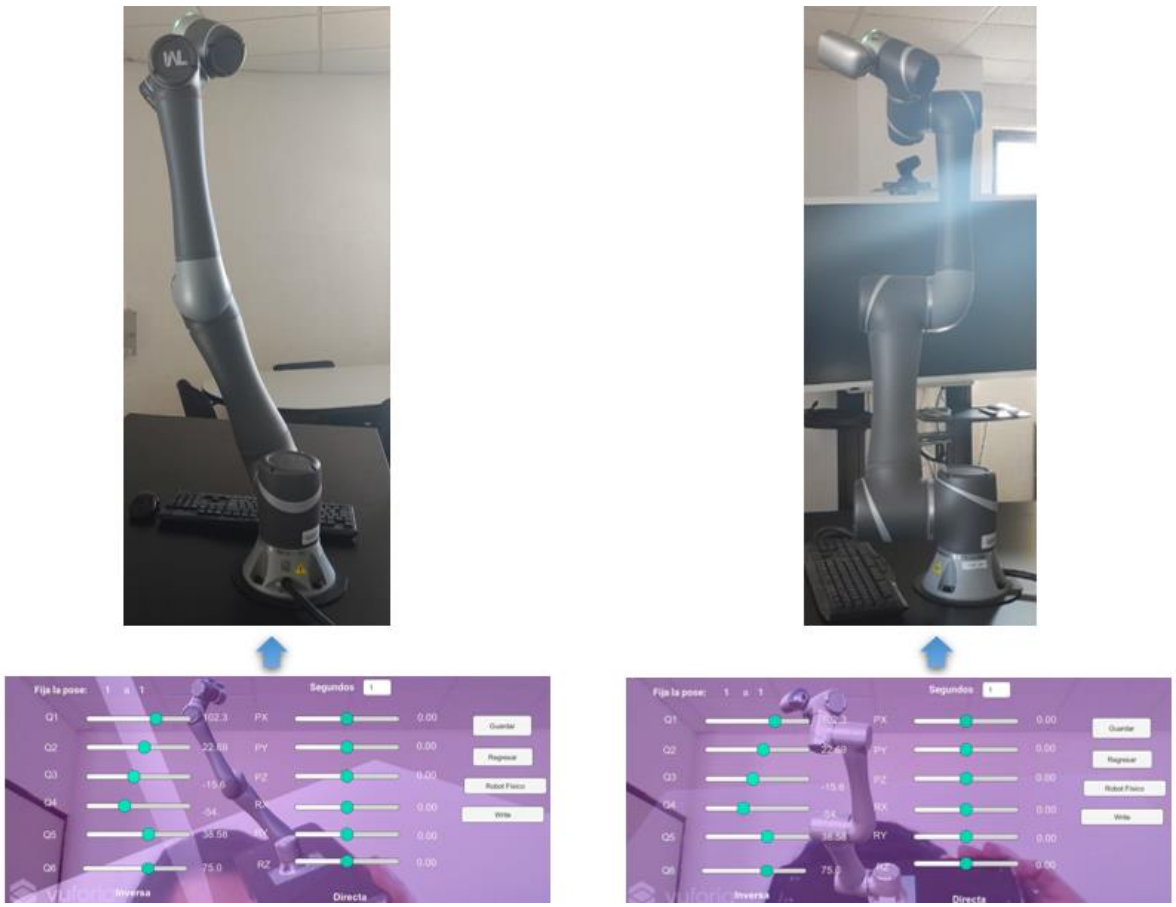

Figura 3.51: Escritura de datos en robot físico a partir de Unity.

3.11 Espacio de operación de cobot

En el laboratorio donde se encuentra instalado el robot TM5-900 es necesario limitar los movimientos de cada joint, ya que el espacio de trabajo está limitado por la mesa donde se encuentra establecido el robot, así como por los componentes electrónicos que se hallan a su alrededor (Figura 3.52). Cabe mencionar que el espacio de trabajo de un robot está definido como el grupo de puntos que pueden ser alcanzados por su efector-final [53].

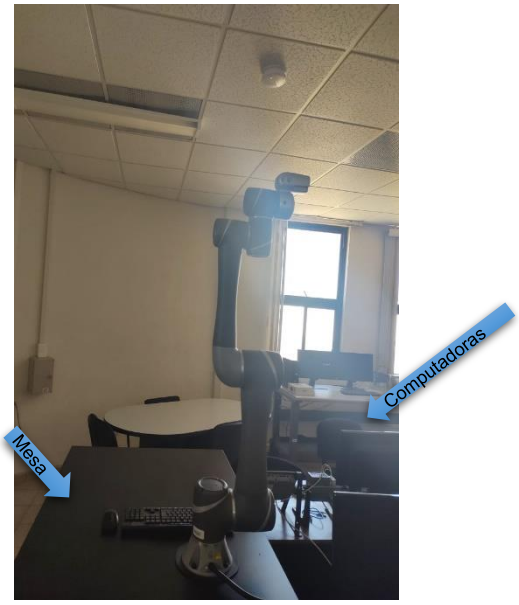

Figura 3.52: Espacio de trabajo en laboratorio.

Los límites que se necesitan establecer, se deben de configurar tanto en Unity como en TMflow®. Para TMflow® existe un módulo (Figura 3.53) en el cual se determinan los puntos con el robot físico, es decir, esta página proporciona la configuración del espacio, el medio es la interfaz del robot virtual y el lado derecho es la interfaz del controlador.

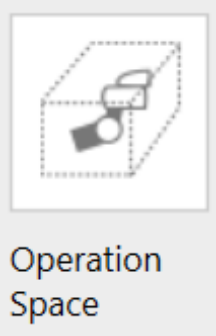

Figura 3.53: Módulo de configuración de espacio de trabajo en TMflow®.

Se puede acceder a la página “Agregar o modificar plano”, haciendo clic en el botón para agregar un nuevo plano o para restablecer las características después de seleccionar las características del plano. En esta página, se creó un plano estableciendo tres puntos, los cuales se muestran en la Figura 3.54 para el eje X y para el eje Y en la Figura 3.55.

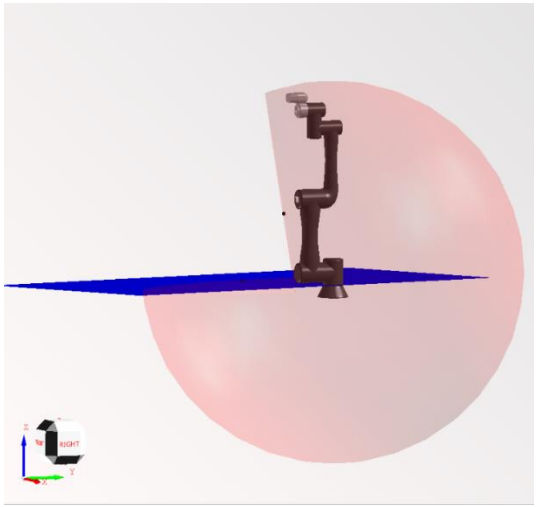

Figura 3.54: Límite de TMflow® en eje X.

El orden de configuración de los tres puntos puede ser aleatorio, y la interfaz virtual del robot mostrará la bola de color correspondiente. Cuando se establecen los tres puntos, aparece un plano virtual azul oscuro, lo que significa que el plano está creado. Es necesario tener en cuenta que cuando se produce el fenómeno de punto común o colinealidad, no se puede crear el plano virtual.

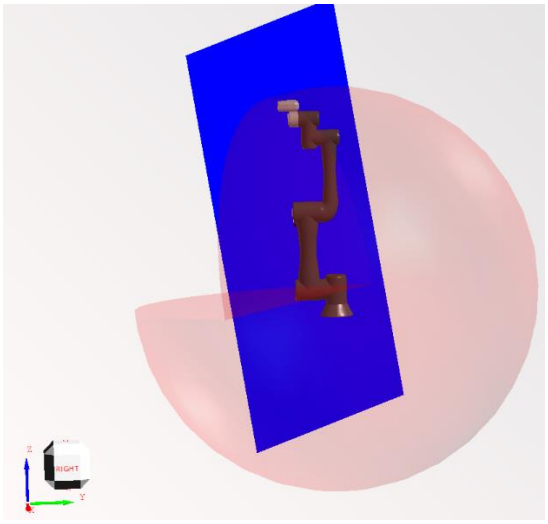

Figura 3.55: Límite de TMflow® en eje Y.

Una vez completada la creación del espacio de operación, se podrá identificar objetos en la pantalla 3D como se muestra en la Figura 3.56. Toda la esfera es el intervalo máximo de movilidad del robot, el espacio reducido está en verde, el espacio de velocidad máxima está en rojo y el espacio de detención es el espacio eliminado del intervalo de movilidad del robot [31].

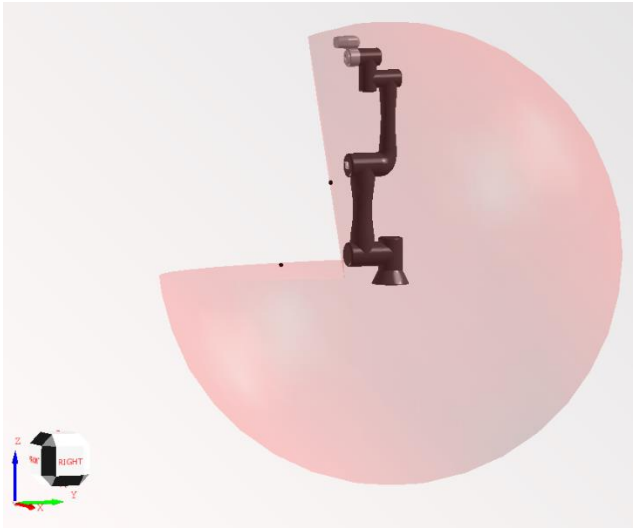

Figura 3.56: Espacio de trabajo establecido en TMflow®.

Para delimitar los planos en Unity, se creó un plano tanto en el eje X (Figura 3.57), como en el eje Y (Figura 3.58).

Se tomó en cuenta la posición del robot físico para poder referenciar ambos planos.

El eje X corresponde a la mesa donde se encuentra posicionado el robot físico.

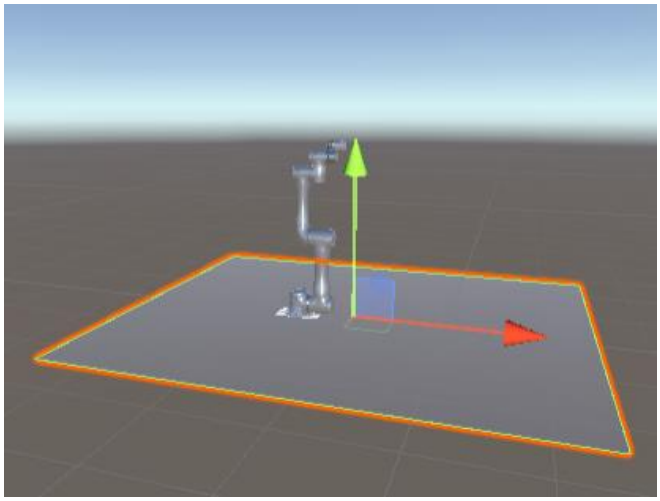

Figura 3.57: Límite de Unity en eje X.

El eje Y limita los movimientos del robot para evitar que se encuentre con los componentes electrónicos que se localizan alrededor del robot físico.

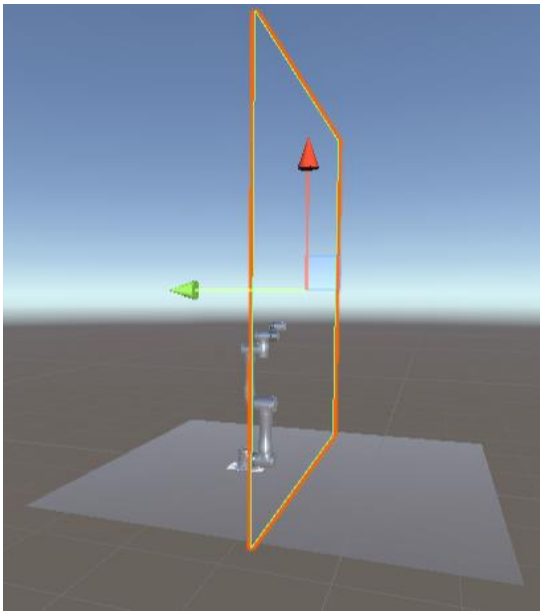

Figura 3.58: Límite de Unity en eje Y.

En la siguiente imagen (Figura 3.59) se muestran los planos X y Y establecidos debidamente para la aplicación de Unity, y de esta manera evitar cualquier accidente con el robot físico desde la simulación.

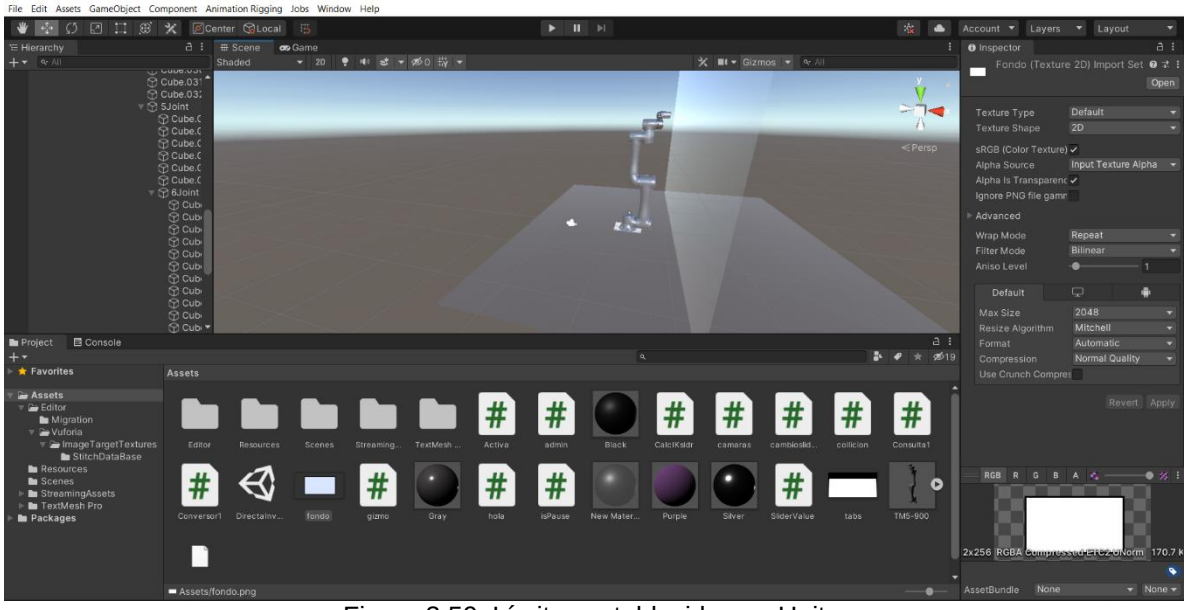

Figura 3.59: Límites establecidos en Unity.

Cuando se ejecuta la aplicación (Figura 3.60), se pueden observar los límites del robot de manera virtual, estos límites seguirán siendo visibles y fieles a la posición del robot físico cuando el usuario manipule al robot virtualmente.

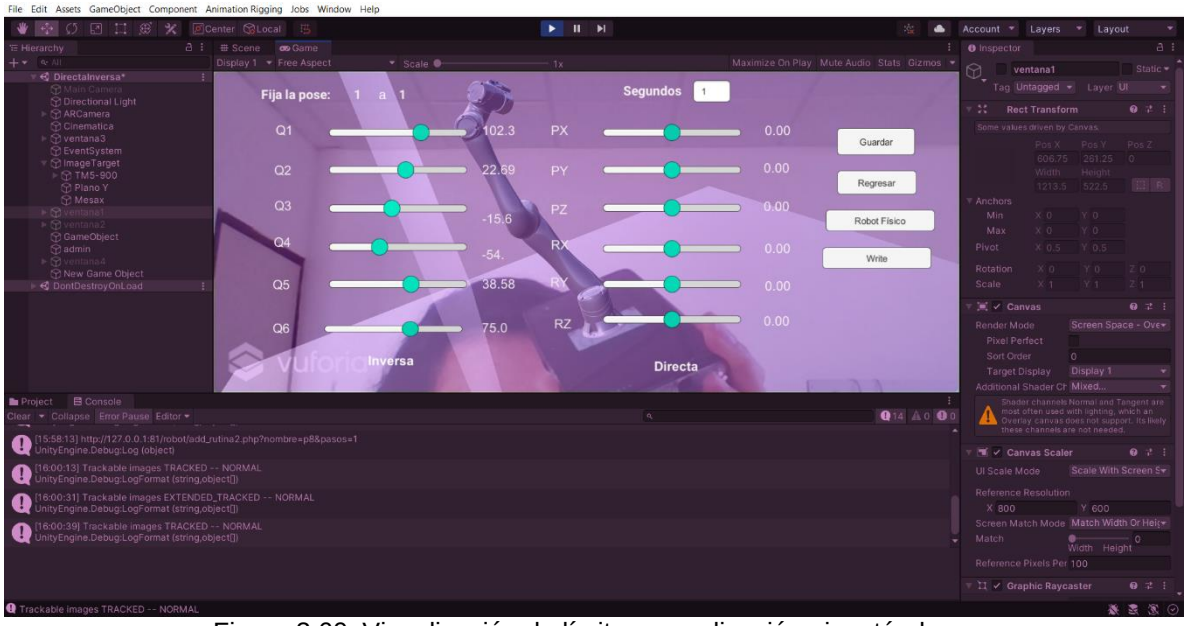

Figura 3.60: Visualización de límites en aplicación ejecutándose.

Si el usuario llega a sobrepasar uno de sus límites, ya sea el plano X o Y, aparecerá una ventana de notificación informando que ha sobrepasado los límites de trabajo del robot (Figura 3.61); por lo cual deberá de regresar con ayuda de slider a una posición segura el grado de libertad que este sobrepasando el límite.

Figura 3.61: Ventana de notificación al pasar los límites de espacio de trabajo en Unity.

4. RESULTADOS

En esta sección se muestra el funcionamiento de la aplicación detalladamente, es decir, se observan los Canvas que se han agregado en Unity para poder generar y visualizar

una rutina del cobot TM5-900. Para lograr visualizar el modelo en 3D mediante realidad aumentada (Figura 4.1), se necesita utilizar el marcador como primer lugar, en este caso se utilizó mediante un celular (dispositivo móvil), si no se llega a ver la imagen se necesitará verificar que el área del marcador no tenga problemas de iluminación y que alcance a visualizar por lo menos un 80% del marcador para poder ver el robot virtual.

Figura 4.1: Funcionamiento de ventana 1 en Unity.

En este caso, se tenía una rutina preconfigurada la cual se denominó “prueba4”, entonces al dar clic en el botón “Ir rutina” automáticamente se redireccionó a la ventana 4, para visualizar la rutina con las entradas de movimientos y tiempo que el usuario seleccionó (Figura 4.2).

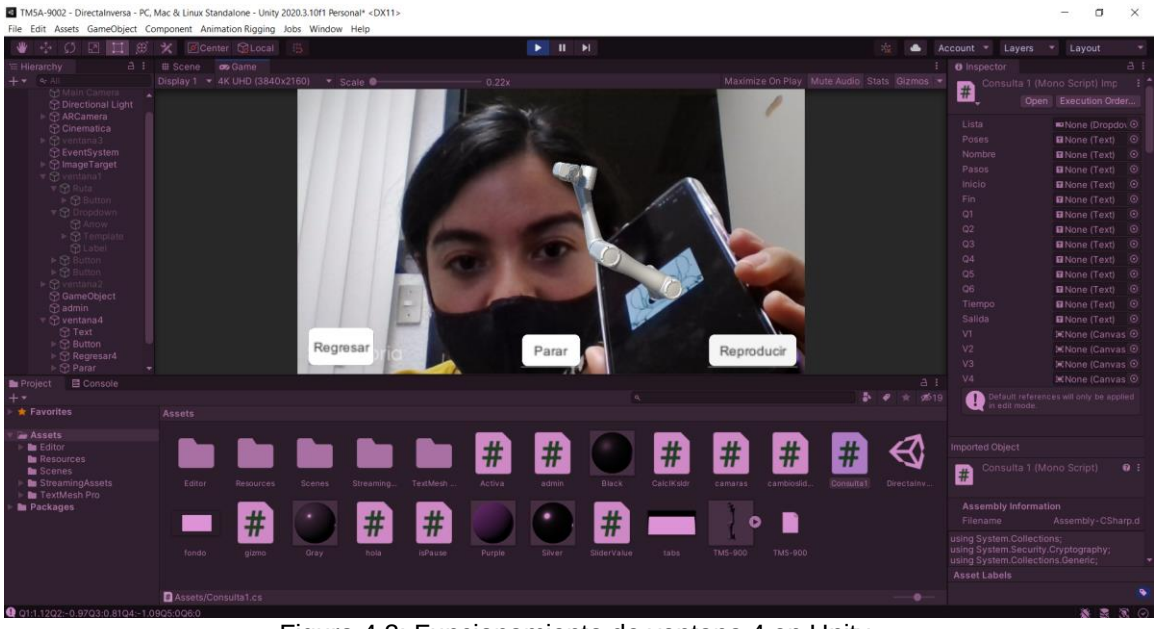

Figura 4.2: Funcionamiento de ventana 4 en Unity.

4.1 Programación de rutina.

En el presente apartado se explicará minuciosamente como agregar una nueva rutina y su ejecución de ésta. Para el manejo de este sistema se tendrá que encender la tableta, para después ejecutar la aplicación. Al dar clic en el botón “Agregar nueva ruta” de la ventana uno, aparecerá la ventana 2. En este caso como nombre de rutina se colocó “rutina 7” y en número de pasos se introdujeron 2 (Figura 4.3). Para pasar a la ventana 3 se le da clic en el botón “Guardar”.

Figura 4.3: Generación de nueva rutina (Ventana 1).

Ya estando en la ventana 3 se deslizó cada Slider de la columna de cinemática inversa, como el usuario pretendía para la pose 1 del cobot, en la Figura 4.4, se muestra su posición. De igual forma se insertó un tiempo de 5 en la casilla de segundos.

Figura 4.4: Generación de pose 1 (Ventana 3).

Para agregar la pose 2 del TechMan® es necesario dar clic en el botón “Guardar” al finalizar la configuración de la posición uno. En este caso se seleccionó la pose que se muestra a continuación (Figura 4.5) con un tiempo de cinco segundos.

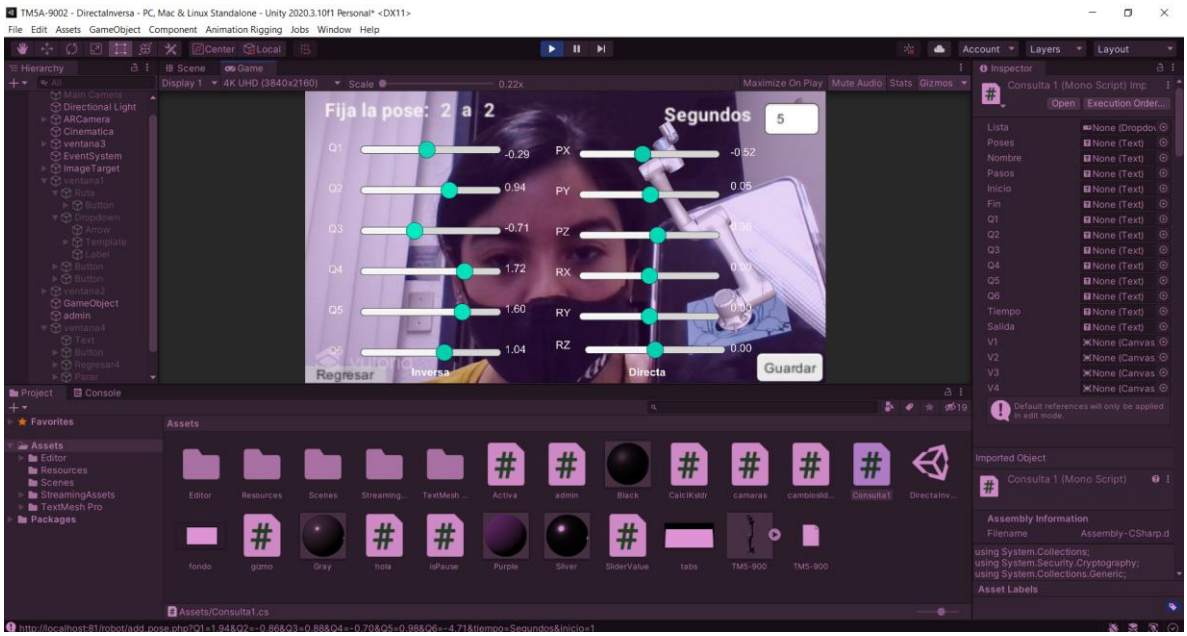

Figura 4.5: Generación de pose 2 (Ventana 3).

Luego de guardar la pose dos, automáticamente se visualiza la ventana uno, donde es necesario dar clic en el botón “Actualizar” (Figura 4.6) para poder observar la rutina recién insertada (“rutina 7”).

Figura 4.6: Actualización de nueva rutina (Ventana 1).

A continuación, en la Figura 4.7 se observa la selección de la rutina utilizada.

Figura 4.7: Selección de nueva rutina (Ventana 1).

Posteriormente se da clic en “Ir Rutina”, y así lograr observar la rutina que se configuró (Figura 4.8).

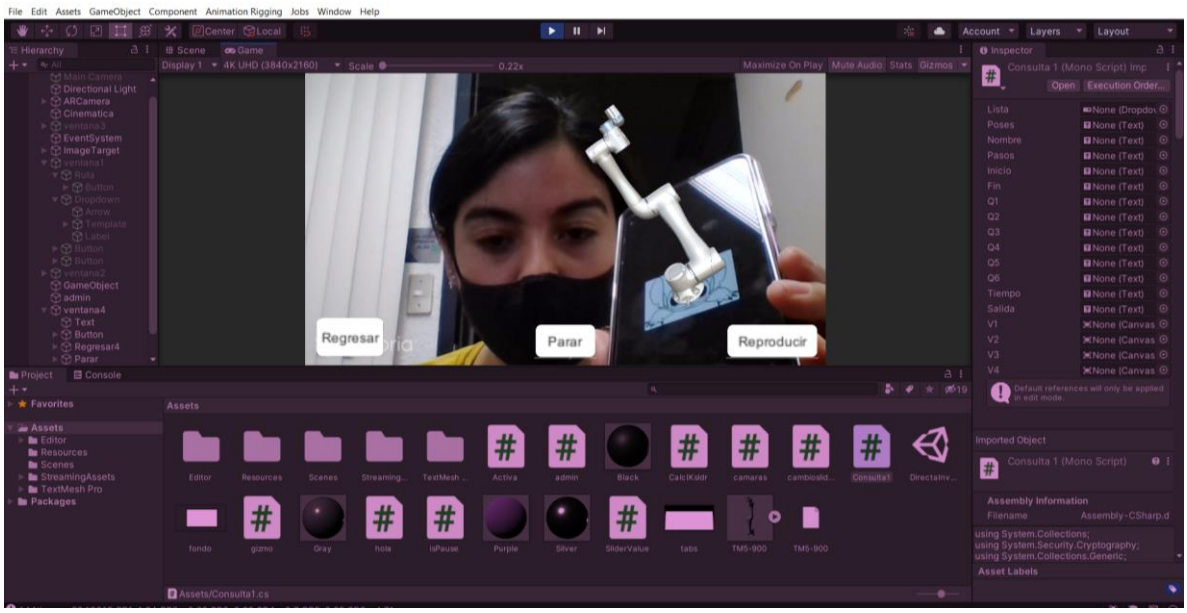

Figura 4.8: Visualización de "rutina 7" (Ventana 4).

4.2 Ejecución de lectura y escritura de datos

El servidor se encarga de dar la respuesta demandada por el cliente. El concepto de cliente hace referencia a un demandante de servicios, este cliente es el robot físico, el cual requiere información proveniente de la red para funcionar [52]. En la Figura 4.9, se evidencia como los grados establecidos en el robot físico, se han leído con éxito en la aplicación realizada en Unity, ya que coinciden los ángulos establecidos en cada joint.

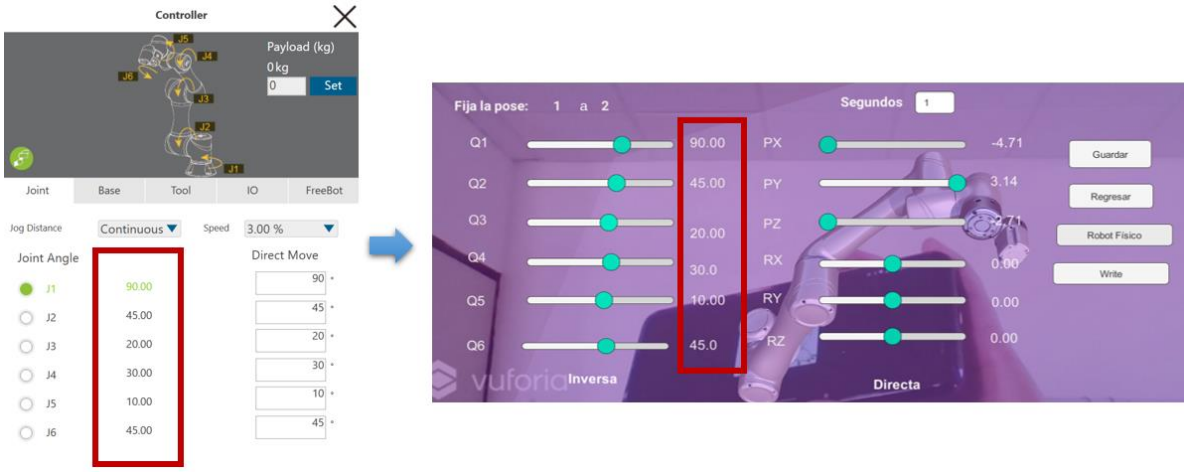

Figura 4.9: Comprobación de datos leídos con el módulo “Controlador”.

En el esquema que se presenta posteriormente (Figura 4.10), podemos ver como la posición tanto del robot físico como virtual corresponden.

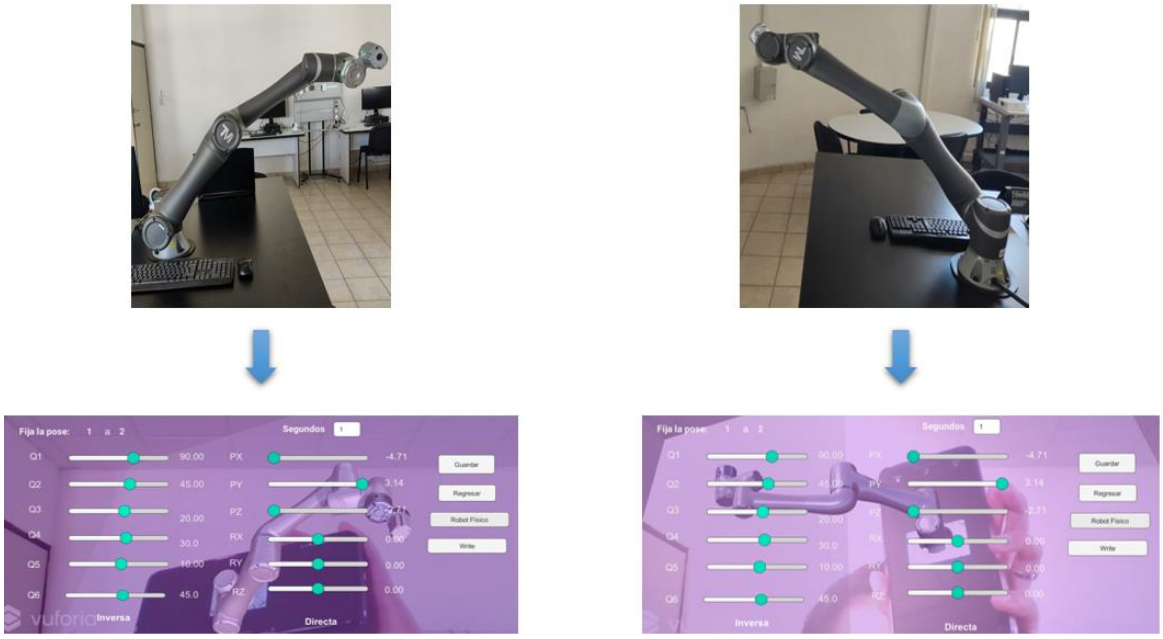

Figura 4.10: Lectura de datos a partir de robot físico.

Al ejecutar el programa, dentro de la consola de Unity, se logran analizar todas las conversiones de bytes que se están ejecutando en tiempo real (Figura 4.11).

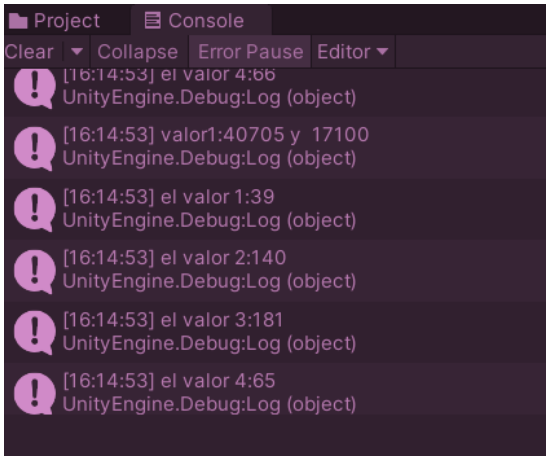

Figura 4.11: Consola de Unity en ejecución.

4.3 PRUEBAS DE FUNCIONAMIENTO

Una vez funcionando la aplicación en el ordenador, se procedió a exportarla a un dispositivo móvil para verificar su correcto funcionamiento. En este caso se utilizó un celular de gama media con sistema operativo Android, la primera pantalla que se observa al ejecutar la aplicación se muestra en la Figura 4.12.

Figura 4.12: Menú inicial de aplicación en Android.

En la cual se selecciona si se desea agregar una nueva rutina, o reproducir alguna otra ya guardada en la base de datos, luego damos clic en “Agregar nueva rutina”, y de esta manera aparece la ventana que muestra la Figura 4.13.

Figura 4.13: Menú para agregar nueva rutina.

En la caja de texto escribimos el nombre de la rutina, así como el número de pasos que tendrá la rutina (Figura 4.14).

Figura 4.14: Menú para guardar la rutina previamente establecida.

Una vez introducidos los datos solicitados, se da clic en guardar (Figura 4.15).

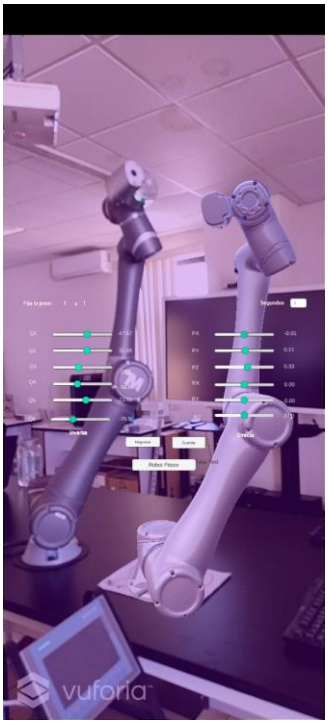

Figura 4.15: Menú para selección la posición del robot y establecer rutina.

Posteriormente deslizamos las sliders para seleccionar el ángulo deseado en cada grado de libertad. De igual manera es posible leer la posición actual del robot físico en lugar de utilizar los sliders, dando clic en “Robot Físico”.

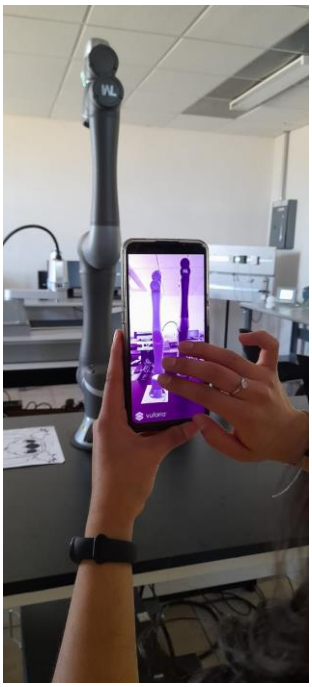

Figura 4.16: Visualización de entorno real al ejecutar aplicación.

Finalmente, en la Figura 4.16 podemos apreciar cómo es que se visualiza de manera real el entorno de ejecución del robot, es decir, en el entorno real solo se visualiza el marcador, y a través de la aplicación se obtiene la vista del robot virtual.
Hay que tener en cuenta que el servidor responda la llamada y que el robot este encendido, si todo es correcto la aplicación verá la imagen de la cámara con la cual tendrá que ver el marcador señalado; al ver el marcador la realidad aumentada se activará y verá la imagen del controlador.

Como resultados medibles se presenta la Tabla 4.1, la cual contiene el tiempo de respuesta de envío de datos de robot virtual a físico, de cada grado de libertad. Este tiempo se obtuvo a través de la consola de Unity y comunicación Modbus, es decir, al ejecutar diversas rutinas, y analizar el tiempo de respuesta que arrojaba dicha consola en cada grado de libertad, siempre es el mismo respectivamente, a pesar de efectuar diferentes posiciones del robot al igual que distintas rutinas.

Tabla 4.1: Tiempo de respuesta de cada grado de libertad.

GDL	**Tiempo de respuesta (segundos)**
1	3.032416 s
2	3.254906 s
3	3.253655 s
4	3.033628 s
5	2.811168 s
6	2.812419 s

Después de analizar los resultados obtenidos, tenemos que el tiempo promedio de respuesta es de 3.033032 segundos, así como el tiempo máximo es de 3.254906 s y el mínimo de 2.811168 s. Los tiempos de respuesta presentados se aproximan a cero, es decir, la aplicación desarrollada funciona a tiempo real, respecto al robot físico, lo que significa que la base de datos siempre va a estar actualizada y sincronizada, de manera que es posible supervisar y monitorizar la información en tiempo real, impulsando la

experiencia colaborativa. Se sabe que es de suma importancia en los procesos de automatización industrial, tener la información de manera inmediata, ya que esto permite rapidez y por consecuencia decisiones apropiadas y acertadas para la línea de producción, que desencadenan en productividad, rendimiento, reducción de costos, disponibilidad y confiabilidad de la empresa.

CONCLUSIÓN

El generar un gemelo virtual de un robot colaborativo con 6 GDL utilizando como interfaz la realidad aumentada, para lograr su manipulación y poder enseñar las capacidades del cobot, así como realizar rutinas del cobot, cumple con el objetivo del presente trabajo, el cual es el desarrollo de un gemelo virtual de un robot colaborativo, utilizando como interfaz la realidad aumentada, que permite su manipulación y ésta sea capaz de enseñar las capacidades del cobot, para poder producir rutinas del robot TechMan®.

Como se puede observar en el capítulo de resultados, se diseñó el gemelo virtual y se elaboraron los huesos del mismo, introduciéndole el modelado de la cinemática directa e inversa para su control y así poder programar trayectorias de movimiento para el brazo robótico. Por otra parte, se creó la conexión entre el cobot físico y el virtual, creando una aplicación bajo el sistema Android, que integra todo lo realizado en la presente investigación, y de esta manera el usuario pueda interactuar tanto con el robot físico como el virtual de una manera gráfica.

Este proyecto explica la metodología para calcular la cinemática directa e inversa del cobot de 6 GDL, con el fin de ejecutar correctamente las posiciones que se requiere establecer en cada articulación del robot, y así estar en la posición final de forma precisa y óptima. Para el cálculo de la cinemática se utilizaron matrices de transformación homogénea, así como los parámetros de Denavit Hartenberg. Las ecuaciones cinemáticas directas e inversas se codificaron en Unity y se integraron en la representación 3D virtual del cobot, así como se reprodujeron en realidad aumentada.

Para poder operar tanto el robot virtual como el físico, es fundamental comunicarse entre sí, para ello se utilizó el protocolo de comunicación abierto denominado Modbus, por lo

que se describió el proceso de enlace; con lo que se puede concluir que se cumplió con el propósito de poder interactuar con el robot virtual a escala real a través de una aplicación generada en Unity, y así tener un control remoto del TechMan®, para luego ejecutar las rutinas guardadas a través de la base de datos. Se creó un ambiente seguro para manipular el robot colaborativo y evitar colisiones con objetos que se encuentren en su entorno real, estableciendo los límites del espacio de trabajo para el cobot, tanto en Unity como en TMflow®; se muestra el funcionamiento de estos límites y cómo el usuario puede interactuar con ellos.

Con estas actividades realizadas y funcionando correctamente, tenemos como resultado un robot virtual, el cual puede ser manipulado remotamente desde la aplicación móvil y ver gráficamente los movimientos del robot que el usuario quiere generar, y al mismo tiempo hacerlo funcionar en tiempo real, en un espacio de trabajo seguro; esto evitará accidentes tanto con el usuario (riesgos humanos) como con el robot físico, ya que la rutina generada por el usuario se puede ver y reproducir desde diferentes perspectivas y ver claramente cuando colisiona.

El lograr implementar las ecuaciones de cinemática inversa y directa para el control de movimientos del cobot, unido a la creación de su gemelo digital, manipulado mediante la realidad aumentada a través de una interfaz que se ejecuta en dispositivos móviles; da a entender que el aporte del presente estudio es la integración de herramientas tecnológicas, ya que en Unity es donde se unen todos los elementos utilizados para el correcto funcionamiento del sistema realizado.

RECOMENDACIONES

El sistema desarrollado que incorpora la realidad aumentada en un gemelo digital representa un sistema avanzado para la enseñanza y validación de la programación de robots y otras posibles aplicaciones. Una adaptación factible es diseñar una línea de ensamblaje con más de dos robots y hacer que los robots trabajen juntos. La realidad aumentada beneficia la comprobación de programas de robots al proporcionar al usuario una herramienta visual en 3D para una detección más fácil de colisiones y asegurar el espacio de trabajo, de esta forma, los programas de robots validados pueden ser

reutilizados siempre que se mantenga el espacio de trabajo y esto ayuda a reducir el tiempo de programación.

La técnica propuesta también puede extenderse, pues una distinta aplicación posible es diseñar una línea de ensamblaje con más de dos robots y hacer que los robots trabajen en conjunto, es decir, en colaboración con más de un gemelo digital para visualizar el espacio de trabajo y las rutinas gráficamente. Por otro lado, desarrollar una aplicación con la que desde la interfaz del dispositivo móvil se puedan identificar los diferentes elementos que aparecen en el área de trabajo del robot. Cabe mencionar que al implementar la simulación de cobots en tercera dimensión de una línea de ensamble, se puede controlar el robot en su área de trabajo y los elementos que en ella puedan aparecer, es decir, automáticamente se puede mover el robot a la posición del objeto y realizar la maniobra de manipulación deseada sobre el mismo, detectando posibles colisiones y avisando del movimiento en dicho caso. Para toda la funcionalidad comentada anteriormente, la plataforma de desarrollo es adecuada para la educación en ingeniería y la capacitación en robótica.

REFERENCIAS BIBLIOGRÁFICAS

[1] IAT, Realidad Aumentada ¿Qué es? características y tipos, 2020. [En línea]. Available: https://iat.es/tecnologias/realidad-aumentada/#:~:text=La%20realidad%20aumentada%20es%20aquella. [Último acceso: 2020].

[2] UR, «Universal Robots,» 2021. [En línea]. Available: https://www.universal-robots.com/. [Último acceso: 2022].

[3] R. Marín, J. P. Sanz, P. Nebot y R. Wirz, «A Multimodal Interface to Control a Robot Arm via the Web: A Case Study on Remote Programming,» *IEEE,* vol. 52, nº 6, pp. 1506 - 1520, 2005.

[4] J. Guhl, S. Tung Nguyen y J. Krüger, «Concept and architecture for programming industrial robots using augmented reality with mobile devices like microsoft HoloLens,» de *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Limassol, Cyprus, 2018.

[5] Native Robotics, «Boost your sales with AR experience,» Omnifit, 2021. [En línea]. Available: https://native-robotics.com/omnifit.

[6] C. Perez Quintero, S. Li, M. KXJ Pan, W. P. Chan, H. M. Van der Loos y . E. Crof, «Robot Programming Through Augmented Trajectories in Augmented Reality,» de *International Conference on Intelligent Robots and Systems (IROS)*, Madrid, Spain, 2019.

[7] ABB, «RobotStudio® AR,» 20 Agosto 2020. [En línea]. Available: https://new.abb.com/news/detail/66541/ar-smartphone-robot-installations.

[8] S. Ong, A. Yew, N. Thanigaivel y A. Nee, «Augmented reality-assisted robot programming system for industrial,» *Robotics and Computer Integrated Manufacturing,* vol. 61, 2020.

[9] K. Zielinski, K. Walas, J. Heredia y M. Baun Kjærgaard, «A Study of Cobot Practitioners Needs for Augmented Reality Interfaces,» de *IEEE International Conference on Robot & Human Interactive Communication (RO-MAN)*, Vancouver, BC, Canada, 2021.

[10] H. Fang, S. Khim Ong y A. Yeh-Ching Nee, «Robot Programming Using Augmented Reality,» *IEEE,* nº 10939123, 2009.

[11] Lis Data Soutions, «¿Qué es la Industria 4.0?,» 2021. [En línea]. Available: https://www.lisdatasolutions.com/blog/que-es-la-industria-4-0/. [Último acceso: 2022].

[12] H. Lasi, P. Fettke, H. G. Kemper, T. Feld y M. Hoffmann , «Industry 4.0,» *Business & Information Systems Engineering,* vol. 6, p. 239–242, 2014.

[13] A. Dujin, C. Geissler y D. Horstkötter, «Industry 4.0 The new industrial revolution How Europe will succeed,» *Berger Strateg. Consult,* pp. 1-24, 2014.

[14] M. Baygin, H. Yetis, M. Karakose y E. Akin, «An effect analysis of industry 4.0 to higher education,» de *International Conference on Information Technology Based Higher Education and Training (ITHET)*, Istanbul, Turkey, 2016.

[15] A. Portillo, «FundationaXCEL a.c.,» Industria 4.0 ¿Qué es?, 2019. [En línea]. Available: https://www.funax.org/fs/blog/mecatronica/62-industria-4-0-que-es. [Último acceso: 2020].

[16] L. Heras Lara, «La realidad aumentada: Una tecnología en espera de usuarios,» *Revista Digital Universitaria. UNAM,* vol. 5, nº 7, 2004.

[17] 3R INDUSTRIA 4.0, *Realidad Aumentada,* 2020.

[18] A. Syberfeldt, O. Danielsson y P. Gustavsson, «Augmented Reality Smart Glasses in the Smart Factory: Product Evaluation Guidelines and Review of Available Products,» *IEEE,* 2017.

[19] P. Croft, «MeriStation,» AS, 2018. [En línea]. Available: https://as.com/meristation/2018/01/10/reportajes/1515567480_172151.html. [Último acceso: 2020].

[20] Ó. Blanco-Novoa, T. M. Fernández-Caramés, P. Fraga-Lamas y M. A. Villar-Montesinos, «ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8281493,» *IEEE,* 2018.

[21] IAT, 2020. [En línea]. Available: https://iat.es/tecnologias/realidad-aumentada/unity/. [Último acceso: 2022].

[22] R. Hussein, «I+I,» El papel de la Inteligencia Artificial en las empresas, 2020. [En línea]. Available: https://www.iti.es/blog/gemelos-digitales-el-futuro-de-la-industria-conectada/. [Último acceso: 2020].

[23] C. Yunqiang , W. Qing , H. Chen, X. Song, H. Tang y T. Mengxiao, «An overview of augmented reality technology,» *IOP Publishing,* 2019.

[24] M. Arzuza, «Slideshare,» Universidad de la costa, 2015. [En línea]. Available: https://www.slideshare.net/masaar/realidad-aumentada-curso/8. [Último acceso: 2020].

[25] C. del Bosque Peón, *Los gemelos digitales en la,* Valladolid: Universidad de Valladolid, 2019.

[26] Esri, «Esri,» Digital Twins Enable Innovation and Savings, 2019. [En línea]. Available: https://www.esri.com/about/newsroom/arcuser/digital-twins-enable-innovation-and-savings/. [Último acceso: 2020].

[27] L. Pedraza, «Sutori,» Linea de tiempo de la Carrera Espacial, [En línea]. Available: https://www.sutori.com/en/story/linea-de-tiempo-de-la-carrera-espacial--yV13ue4AgXJ71CcY1SXANhmD. [Último acceso: 2020].

[28] J. Pelegrí, «Cobot vs Industrial Robot,» Universal Robots, España, 2020.

[29] RIPISA, «La historia de los robots colaborativos,» Universal Robots México, México, 2019.

[30] I. Automation, «Propuesta Técnico-Comercial,» Nuevo León, México, 2020.

[31] Omron, «7M Robot,» 2018. [En línea]. Available: https://assets.omron.eu/downloads/manual/en/v2/i623_collaborative_robots_hardware_installation_manual_en.pdf.

[32] Techman robot, «7M Techman robot,» [En línea]. Available: https://techman-robot.com.mx/. [Último acceso: 2022].

[33] ADR Formación, «Soluciones eLearning,» Curso Online de Solidworks, 2017. [En línea]. Available: https://www.adrformacion.com/knowledge/ingenieria-y-proyectos/_que_es_solidworks_.html.

[34] Universitat politecnica de Catalunya, «BarcelonaTech,» [En línea]. Available: https://vilanovaformulateam.upc.edu/es/shared/patrocinadores/grupo-b/solidworks-logo.png/view.

[35] A. Chirivella González, «Profesional Review,» Blender, 2022. [En línea]. Available: https://www.profesionalreview.com/2022/02/20/blender-que-es-y-para-que-se-utiliza/. [Último acceso: 2022].

[36] ComparaSoftware, «SPA,» 2021. [En línea]. Available: https://www.comparasoftware.com/blender. [Último acceso: 2022].

[37] logotyp, «logotyp.us,» 2020. [En línea]. Available: https://logotyp.us/logo/unity/. [Último acceso: 2022].

[38] Vuforia, «Vuforia Developer Library,» 2021. [En línea]. Available: https://library.vuforia.com/getting-started/getting-started-vuforia-engine-unity. [Último acceso: 2022].

[39] T. Megali, «envatotuts+,» 2017. [En línea]. Available: https://gamedevelopment.tutsplus.com/es/tutorials/vuforia-tips-and-tricks-on-unity--cms-28744. [Último acceso: 2022].

[40] Logopedia, 2020. [En línea]. Available: https://logos.fandom.com/wiki/Android_Studio#2014%E2%80%932019. [Último acceso: 2022].

[41] A. F. Ruiz Olaya, A. Barandica López y F. G. Guerrero Moreno, «Implementación de una Red MODBUS/TCP,» *Ingeniería y Competitividad,* 2004.

[42] Vuforia, «Vuforia: Market-Leading Enterprise AR,» 2020. [En línea]. Available: https://www.ptc.com/en/products/vuforia. [Último acceso: 2020].

[43] M. Ripoll Tarré, «Modelado y animación de un personaje para videojuegos,» 2014. [En línea]. Available: https://upcommons.upc.edu/bitstream/handle/2117/174167/Samper_Marc_memoria_TFG%20%282%29.pdf?sequence=1&isAllowed=y. [Último acceso: 2021].

[44] Disney, *Stitch,* https://clipartmag.com/download-clipart-image#stitch-drawing-ohana-36.jpg.

[45] J. L. Ramírez Arias y A. Rubiano Fonseca, «Modelamiento matemático de la cinemática directa e inversa de un robot manipulador de tres grados de libertad,» *Ingeniería Solidaria,* vol. 8, nº 15, 2012.

[46] J. Villalobos, I. Y. Sanchez y F. Martell, «Statistical comparison of Denavit-Hartenberg based inverse kinematic solutions of the UR5 robotic manipulator,» de *2021 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, Mauritius, Mauritius, 2021.

[47] L. Rodríguez Islas, C. A. Paredes Orta y F. Martell Chávez, «Forward and Inverse Kinematics Control of a Collaborative 6-DOF Robot for an Augmented Reality Educational System,» de *International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, Mauritius, 2021.

[48] A. F. Sánchez Osorio y L. A. Parra Rativa, «Sistema de información WEB par la optimización del proceso de gestión y administración de los laboratorios de informática de la universidad distrital Francisco José Caldas,» *Universidad Distrital Francisco José De Caldas.*

[49] Rossmann-engineering , «SRE-Solutions,» 2017. [En línea]. Available: http://easymodbustcp.net/en/. [Último acceso: 2022].

[50] Universiad de Sonora, «Euler.Mat,» 2020. [En línea]. Available: http://euler.mat.uson.mx/~havillam/ca/Slides/08-Organization.pdf. [Último acceso: 2022].

[51] S. Alvarez, «DesarrolloWeb.com,» Arquitectura cliente-servidor, 30 Agosto 2007. [En línea]. Available: https://desarrolloweb.com/articulos/arquitectura-cliente-servidor.html#:~:text=Normalmente%20el%20servidor%20es%20una,entre%20s%C3%AD%20mediante%20una%20red.. [Último acceso: 2022].

[52] A. Schiaffarino, «InfraNetWorking,» Modelo cliente servidor, 2019 Marzo 2019. [En línea]. Available: https://blog.infranetworking.com/modelo-cliente-servidor/#:~:text=Servidor%3A%20Un%20servidor%20hace%20referencia,dem%C3%A1s%20agentes%20de%20la%20red..

[53] J. Martínez, «Jau el ingeniero,» 07 Julio 2013. [En línea]. Available: https://jauelingeniero.wordpress.com/2013/07/07/bases-sobre-espacios-de-trabajo-en-robotica-parte-

1/#:~:text=El%20espacio%20de%20trabajo%20de,trabajar%20(simple%20y%20llanamente)..

ANEXO A. ESPECIFICACIONES TÉCNICAS DE COBOT TM5-900 [32].

Model		TM5-700	TM5-900	TM5X-700	TM5X-900	TM5M-700	TM5M-900
Weight		22.1 kg	22.6 kg	21.8 kg	22.3 kg	22.1 kg	22.6 kg
Maximum Payload		6 kg	4 kg	6 kg	4 kg	6 kg	4 kg
Reach		700 mm	900 mm	700 mm	900 mm	700 mm	900 mm
Typical Speed		1.1 m/s	1.4 m/s	1.1 m/s	1.4 m/s	1.1 m/s	1.4 m/s
Joint ranges	J1,J6	+/- 270°	+/- 270°	+/- 360°	+/- 360°	+/- 270°	+/- 270°
	J2,J4,J5	+/- 180°	+/- 180°	+/- 360°	+/- 360°	+/- 180°	+/- 180°
	J3	+/- 155°					
Speed	J1~J3	180°/s					
	J4~J6	225°/s					
Repeatability		+/- 0.05 mm					
Degrees of freedom		6 rotating joints					
I/O ports		Control box			Tool conn.		
	Digital in	16			4		
	Digital out	16			4		
	Analog in	2			1		
	Analog out	1			0		
I/O power supply		24V 1.5A for control box and 24V 1.5A for tool					
IP classification		Robot Arm: IP54 ; Control Box: IP32					
Power consumption		Typical 220 watts					
Temperature		The robot can work in a temperature range of 0-50°C					
Power supply		100-240 VAC, 50-60 Hz				DC22~60VDC	
I/O Interface		3×COM, 1×HDMI, 3×LAN, 4×USB2.0, 2×USB3.0					
Certification		CE, SEMI S2 (optional)					
Robot Vision							
Eye in Hand (Built in)		1.2M/5M pixels, color camera		N/A		1.2M/5M pixels, color camera	
Eye to Hand (Optional)		Support Maximum 2 GigE cameras					

ANEXO B. PARÁMETROS DE SEGURIDAD DE COBOT TM5-900 [32].

			Valor predeterminado	Valor mínimo	Valor máximo	Unidad
Configuración de seguridad de rendimiento	Velocidad de TCP		1.5	0	4.5	m/s
	Fuerza de TCP		150	0	450	N
	Límite de velocidad TCP de la guía manual		1.5	0	4.5	m/s
	Velocidad de la articulación	J1	190	0	190	grados/s
		J2	190	0	190	grados/s
		J3	190	0	190	grados/s
		J4	235	0	235	grados/s
		J5	235	0	235	grados/s
		J6	235	0	235	grados/s
	Par de torsión de la articulación	J1	65	0	170	Nm
		J2	65	0	170	Nm
		J3	65	0	170	Nm
		J4	15	0	45	Nm
		J5	15	0	45	Nm
		J6	15	0	45	Nm
Configuración de seguridad hombre-máquina	Velocidad de TCP		Varía en función de la región del cuerpo seleccionada.	0	1.5	m/s
	Fuerza de TCP			0	450	N
	Límite de velocidad TCP de la guía manual		Los valores predeterminados, mínimo y máximo son los mismos que los de configuración de seguridad de rendimiento.			
	Velocidad de la articulación	J1-J6				
	Par de torsión de la articulación	J1	65	0	170	Nm
		J2	65	0	170	Nm
		J3	65	0	170	Nm
		J4	15	0	45	Nm
		J5	15	0	45	Nm
		J6	15	0	45	Nm
	Reducir tiempo		150	150	800	ms
	Posición de la articulación	J1	-	-270	270	grados
		J2	-	-180	180	grados
		J3	-	-155	155	grados
		J4	-	-180	180	grados
		J5	-	-180	180	grados
		J6	-	-270	270	grados
	Límite cartesiano A/B	X	-	-3000	3000	mm
		Y	-	-3000	3000	mm
		Z	-	-3000	3000	mm

		θz	-	0	359	grados
		R	-	60	3000	mm

ANEXO C. REQUISITOS PARA INSTALAR TMFLOW [32].

Los requisitos mínimos del cliente para instalar TMflow son los siguientes:

Sistema operativo:	Windows 7 SP 1
CPU:	Serie Intel i5
RAM:	4 GB
Espacio de la unidad de disco duro:	2 GB de espacio disponible
Resolución de la pantalla:	1366 por 768
Periféricos:	Puertos USB y puertos Ethernet
Idiomas admitidos:	Inglés, chino simplificado, chino tradicional, japonés, alemán, coreano, vietnamita, español, francés, italiano, danés, holandés, checo, húngaro, rumano, portugués, turco, polaco y tailandés
Requisitos adicionales:	1. 2010Redistributable_bcredist (x64/x86) 2. 2013Redistributable_bcredist (x64/x86) 3. 2015Redistributable_bcredist (x64/x86) 4. .Net Framework 4.52 o superior.

ANEXO D. BIBLIOGRAFÍAS NO REFERENCIADAS

Journals & Books

[1] Q. Chen, S. Zhu and X. Zhang, "Improved Inverse Kinematics Algorithm Using Screw Theory for a Six-DOF Robot Manipulator", *International Journal of Advanced Robotic Systems*, vol. 12, no. 10, p. 140, 2015. Available: 10.5772/60834 [Accessed 2022].

[2] F. De Pace, F. Manuri, A. Sanna and C. Fornaro, "A systematic review of Augmented Reality interfaces for collaborative industrial robots", *Computers & Industrial Engineering*, vol. 149, p. 106806, 2020. Available: 10.1016/j.cie.2020.106806 [Accessed 2022].

ANEXO E. ARTÍCULO CIENTÍFICO GENERADO A PARTIR DE ESTE TRABAJO DE INVESTIGACIÓN

Artículo denominado **“Forward and Inverse Kinematics Control of a Collaborative 6-DOF Robotic for an Augmented Reality Educational System”.**

Se participó en la Conferencia Internacional sobre Ingeniería Eléctrica, Informática, Comunicaciones y Mecatrónica (ICECCME, por sus siglas en inglés), el cual es el evento principal que reúne a profesionales de la industria, académicos e ingenieros de las instituciones relacionadas para intercambiar información e ideas sobre ingeniería eléctrica, informática, comunicaciones y mecatrónica.

El artículo describe el marco matemático y el procedimiento de desarrollo para la implementación de las ecuaciones cinemáticas directas e inversas de un cobot de seis grados de libertad bajo la plataforma Unity, que se utiliza para generar la versión de realidad aumentada del cobot; De la misma forma, la librería Vuforia también se utiliza para crear la aplicación multiplataforma que se puede ejecutar en ordenadores convencionales o en cualquier dispositivo electrónico portátil con sistema Android. La aplicación desarrollada fue validada y verificada para su uso previsto, que es el aprendizaje a distancia y educativo de robótica y tecnologías de realidad virtual y aumentada.

Los artículos presentados en la conferencia ICECCME se publicaron en IEEE Xplore.

Link de publicación: https://ieeexplore.ieee.org/document/9590903.