UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE GUAYAQUIL

CARRERA DE MECATRÓNICA

DESARROLLO DE UN SISTEMA DE TELEOPERACIÓN DE BRAZO ROBÓTICO MEDIANTE KINECT PARA LA EMULACIÓN DEL MOVIMIENTO DEL BRAZO DE UN OPERADOR

Trabajo de titulación previo a la obtención del
Título de Ingeniero en Mecatrónica

AUTOR: Diego Javier Córdova Legarda
TUTOR: Ing. Tomás Santiago Gavilánez Gamboa

Guayaquil - Ecuador

2024

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Yo, **Diego Javier Córdova Legarda** con documento de identificación Nº **0953627999** manifiesto que:

Soy el autor y responsable del presente trabajo; y, autorizo a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo.

Guayaquil, 04 de marzo del 2024

Atentamente,

Diego Javier Córdova Legarda
0953627999

CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

Yo, **Diego Javier Córdova Legarda** con documento de identificación N° **0953627999** expreso mi voluntad y por medio del presente documento cedo a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que soy autor del Dispositivo Tecnológico: DESARROLLO DE UN SISTEMA DE TELEOPERACIÓN DE BRAZO ROBÓTICO MEDIANTE KINECT PARA LA EMULACIÓN DEL MOVIMIENTO DEL BRAZO DE UN OPERADOR, el cual ha sido desarrollado para optar por el título de: Ingeniero en Mecatrónica, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribo este documento en el momento que haga la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Guayaquil, 04 de marzo del 2024

Atentamente,

Diego Javier Córdova Legarda
0953627999

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, **Tomás Santiago Gavilánez Gamboa**, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: **DESARROLLO DE UN SISTEMA DE TELEOPERACIÓN DE BRAZO ROBÓTICO MEDIANTE KINECT PARA LA EMULACIÓN DEL MOVIMIENTO DEL BRAZO DE UN OPERADOR**, realizado por **Diego Javier Córdova Legarda** con documento de identificación N° **0953627999** obteniendo como resultado final el trabajo de titulación bajo la opción **Dispositivo Tecnológico** que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Guayaquil, 04 de marzo del 2024

Atentamente,

Ing. Tomás Santiago Gavilánez Gamboa, MSc.
1802792646

DEDICATORIA

Dedico este trabajo de titulación a mis padres, Leonardo Javier Córdova Hurtado y Wendy Auxiliadora Legarda Vergara, pilares de mi vida cuyo esfuerzo y sacrificio han sido fundamentales para mi crecimiento personal y profesional. Sus enseñanzas y consejos han alimentado en mí un profundo deseo de superación diaria y me han guiado siempre hacia el camino correcto. Gracias a ustedes, he logrado ser una persona íntegra, dedicada y responsable. Este logro es, en gran medida, el reflejo de su amor y su incansable apoyo.

Diego Javier Córdova Legarda

AGRADECIMIENTO

Agradezco profundamente a Dios, a mi padre y madre, por su inmenso amor y el esfuerzo incansable que han depositado en mí, haciendo posible este logro.

Agradezco a los docentes de la carrera de ingeniería en Mecatrónica, cuya dedicación, conocimiento y pasión por enseñar han sido fundamentales en mi formación académica y profesional.

Asimismo, extiendo mi profunda gratitud a todas las personas que me acompañaron y apoyaron a lo largo de este proceso. Mi sincero agradecimiento al Ing. David Cortez, quien creyó en mi potencial y me apoyó incansablemente, además de brindarme valiosos consejos. al Ing. Edison Llanos por su inmenso apoyo y por compartir conmigo todo su conocimiento, y cuya guía ha sido esencial para mi crecimiento académico. A mi tutor de tesis el Ing. Tomás Gavilanez, por dedicar su tiempo y esfuerzo en mi proceso de formación y desarrollo de mi tesis. Al Ing. Jorge Fariño, por confiar en mí y darme la oportunidad de demostrar mi potencial. Al Ing. Eduardo Cortez, por brindarme su ayuda cuando la necesitaba. Y a mi amiga Magacha, por su constante compañía y aliento, especialmente en esas largas noches de trabajo hasta finalizar la tesis. Gracias por compartir conmigo su valioso tiempo, conocimiento, enseñanzas y por inspirarme a ser mejor cada día.

Diego Javier Córdova Legarda

RESUMEN

En la actualidad, la manipulación de brazos robóticos articulados presenta desafíos significativos debido a la robustez de los movimientos y los retrasos en la comunicación, lo cual puede dificultar la tarea de los operadores en entornos donde la precisión es crucial. Para abordar esta problemática, se ha desarrollado un sistema de teleoperación para un brazo robótico KR10 de la marca KUKA, utilizando el sensor Kinect para replicar los movimientos del brazo del operador con mayor fluidez y tiempos de respuesta mejorados.

En la fase final de pruebas, se verificó que el robot es capaz de seguir la trayectoria determinada por el operador de manera eficaz. Durante esta evaluación, se examinaron meticulosamente el tiempo de respuesta del robot y la precisión de sus movimientos.

La integración entre software y el hardware utilizados resulta en un sistema de teleoperación eficaz que replica los movimientos del brazo del operador. Este resultado se valida mediante pruebas de pick and place realizadas con el robot, lo que permite una ejecución más fluida en los movimientos del brazo robótico.

Palabras claves: Teleoperación, Kinect, RoboDK, Brazo Robótico, KUKA KR10.

ABSTRACT

Currently, the manipulation of articulated robotic arms poses significant challenges due to the robustness of movements and communication delays, which can hinder operators' tasks in environments where precision is crucial. To address this issue, a teleoperation system has been developed for a KUKA KR10 robotic arm, using the Kinect sensor to replicate operator arm movements with increased fluidity and improved response times.

To implement this system, communication is established via a crossover network cable between the computer and the robot controller. C3Bridge is utilized on the KRC4 controller as a bridge to access the KUKA robot. Additionally, RoboDK software serves as an intermediary to transmit Kinect-captured movements and execute them on the actual robot through a program running cyclically on the robot.

The integration of software and hardware results in an effective teleoperation system that replicates operator arm movements. Validation of this system is achieved through pick and place tests conducted with the robot, enabling smoother execution of robotic arm movements.

Keywords: Teleoperation, Kinect, RoboDK, Robotic Arm, KUKA KR10.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

I. Introducción

El objetivo principal de este estudio consiste en el desarrollo de un sistema de teleoperación para un brazo robótico, empleando el sensor Kinect. Este sistema permitirá al usuario controlar el robot mediante los movimientos de su propio brazo, los cuales serán replicados por el brazo robótico. Este proceso se lleva a cabo mediante la integración de diversos softwares, drivers y algoritmos que facilitan la comunicación con el controlador del robot.

El funcionamiento del sistema requiere la implementación de un seguimiento del esqueleto humano utilizando la cámara Kinect, con el fin de capturar las coordenadas y la orientación del brazo derecho, siendo C# el lenguaje de programación elegido para realizar esta tarea. Estos datos, son compartidos mediante un archivo .JSON que servirá para ser procesados y transformados utilizando la API del software RoboDK en Python y así proporcionar el movimiento al brazo robótico en tiempo real.

Además, se debe establecer una comunicación por medio de un cable de red cruzado desde el ordenador hacia el controlador del robot. Tras realizar la configuración correspondiente, se procede a la instalación del programa C3Bridge, encargado de actuar como puente de comunicación entre el ordenador y el brazo robótico. Seguidamente, se ejecuta el programa ”RoboDKsync562” en modo AUT en el SmartPad del robot, lo cual permite recibir los comandos enviados por el script de Python.

La implementación de este sistema de teleoperación conlleva a una mejora significativa en la manipulación del brazo robótico por parte del operador, permitiendo una ejecución más fluida y eficiente de las tareas asignadas. Esta optimización en el control facilita la realización de actividades con una mayor rapidez y un tiempo de respuesta altamente efectivo.

II. PROBLEMA

A nivel global, se observa una diversidad significativa de enfoques para la implementación del control de movimientos en brazos robóticos, los cuales varían en función del entorno de despliegue. Como resultado, no todos estos enfoques exhiben un rendimiento, eficiencia y adaptabilidad uniformes al enfrentar escenarios caracterizados por alteraciones abruptas. Este fenómeno da lugar a notables desventajas que requieren ser abordadas con rigor y precisión [1].

En el contexto de sistemas de teleoperación bilateral, lograr una transparencia perfecta donde el operador sienta control directo sobre el robot remoto se ve desafiado por los retrasos en el canal de comunicación, lo que provoca demoras en las órdenes y dificulta el control fluido. Esto se agrava con sistemas hápticos, esenciales para la emulación de movimientos precisos, pero limitados en eficiencia debido a variables contextuales como el entorno físico, distancias entre componentes y condiciones específicas de las situaciones de acción, lo que influye en la efectividad global del sistema, y complica aún más la superación de los retrasos comunicativos [2].

Una limitación significativa se encuentra en la latencia generada al ingresar una instrucción de programación. Este proceso da lugar a un período de tiempo en el cual hay un retraso entre la emisión de la orden por parte del operador y su ejecución en el robot. Este intervalo temporal introduce una separación entre la acción deseada y su implementación práctica, lo que se traduce en un retardo. La gestión y reducción de este componente temporal son de suma importancia para garantizar un funcionamiento adecuado del sistema. La optimización de la respuesta y coherencia en el comportamiento del robot dependen en gran medida de cómo se maneja y minimiza esta dimensión temporal [3].

Los sensores desempeñan un papel crucial en la operación de los robots, pero enfrentan desafíos en entornos no controlados, como el ruido y las alteraciones en sus señales. Específicamente en sistemas con sensores de distancia u ópticos, que a menudo son autónomos o semiautónomos, la respuesta y el desempeño pueden verse comprometidos en situaciones no ideales. A pesar de ser esencial para la ejecución precisa de acciones programadas, su funcionamiento óptimo depende de la estabilidad del entorno circundante [4].

Aunque el empleo del joystick o 'smartPAD' resulta apropiado para programar rutas automatizadas del robot, su eficacia puede disminuir al ejecutar acciones en tiempo real y variables. En tales casos, estos dispositivos tienden a generar movimientos más rígidos, lo cual varía según las demandas o situaciones en las que se encuentre el operador [5].

En la actualidad, los sistemas de control adoptan diversos enfoques para manipular brazos robóticos articulados, lo que puede generar movimientos robustos, aunque careciendo de naturalidad. Esta situación requiere correcciones, especialmente en entornos donde los operadores deben maniobrar robots industriales de manera específica para lograr resultados exitosos. En este contexto, la implementación de sistemas teleoperados se presenta como una solución propicia, permitiendo una replicación más fluida de los movimientos humanos en los brazos robóticos.

III. OBJETIVOS

III-A. Objetivo general

Desarrollar un sistema de teleoperación de brazo robótico mediante Kinect para la emulación del movimiento del brazo de un operador.

III-B. Objetivos específicos

- Representar en un HMI los puntos representativos del brazo del usuario utilizando Kinect.
- Desarrollar un algoritmo para la teleoperación del robot basado en código abierto.
- Validar el funcionamiento del sistema utilizando una aplicación de recogida y posicionamiento a través de pruebas de campo.

IV. Fundamentos Teóricos

IV-A. Robot Industrial

Un brazo robótico, también conocido como manipulador robótico, es un tipo de robot que se caracteriza por ser programable y poseer funcionalidades análogas a las de un brazo humano tal y como se muestra en la Figura 1. Según su aplicación puede llevar a cabo una variedad de tareas, como soldadura, sujeción, rotación, entre otras. Los mismos pueden operar en modo autónomo o ser controlados manualmente, posibilitando la ejecución de diversos trabajos con una alta precisión [6].

Figura 1. Brazo robot KR10 de la marca KUKA [7].

IV-A1. Partes del robot industrial: Un brazo robótico se compone de articulaciones que actúan como ejes, proporcionando grados de libertad en el movimiento. La cantidad de articulaciones rotativas influye en su capacidad de movimiento, siendo típicas entre cuatro y seis como se muestra en la Figura 2. Además de estas articulaciones, los componentes del brazo robótico incluyen un controlador, una herramienta en el extremo, actuadores, sensores, sistemas de visión, suministro eléctrico y elementos de software [8].

El manipulador es la parte central del robot que realiza las acciones. Está compuesto por ejes móviles conectados en serie, formando una cadena cinemática. Esta cadena define los movimientos y la geometría del robot de manera planificada y controlada [9].

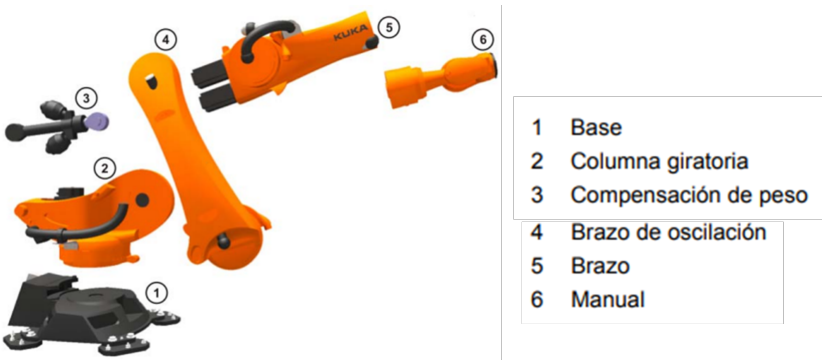


Figura 2. Componentes de la mecánica del robot [9].

La capacidad de movimiento de un robot se determina por sus grados de libertad, los cuales permiten la realización de tareas específicas y complejas. En el ámbito de los brazos robóticos promedio, se considera que contar con seis grados de libertad es ideal. Esta configuración permite una operatividad óptima. Los ejes de estos robots se identifican desde la base hasta la parte superior, siguiendo una secuencia que va desde A1 hasta A6, correspondiente a sus seis articulaciones esenciales, tomando como referencia el brazo robot KUKA con 6 GL, tal como se ilustra en la Figura 3.

Figura 3. Ejes del brazo robot estándar [9].

IV-A2. Tipos de robots industriales: la robótica desempeña un papel crucial, ofreciendo una diversidad de soluciones para mejorar la eficiencia en manufactura y producción. Estos robots se caracterizan por su capacidad para adaptarse a diferentes tareas, desde movimientos complejos y flexibles hasta operaciones de alta precisión y velocidad. Cada diseño responde a retos específicos del entorno industrial, destacando en áreas como flexibilidad, precisión y colaboración con humanos. Sus aplicaciones son tan variadas como sus diseños, entre ellos están:

1. **Robot Articulado**: Similares a un brazo humano, con ejes que se pliegan en diferentes direcciones, ideales para tareas que requieren alta velocidad y flexibilidad, teniendo como ejemplo el brazo robótico de la marca ABB que se muestra en la Figura 4. A menudo requieren controladores dedicados y programación compleja.

Figura 4. Brazo robot articulado 6 GL [10].

2. **Robot Cartesiano**: Caracterizados por su movimiento lineal en tres ejes perpendiculares tal como se puede observar en la Figura 5. Son conocidos por su precisión en aplicaciones de recoger y colocar, carga y descarga.

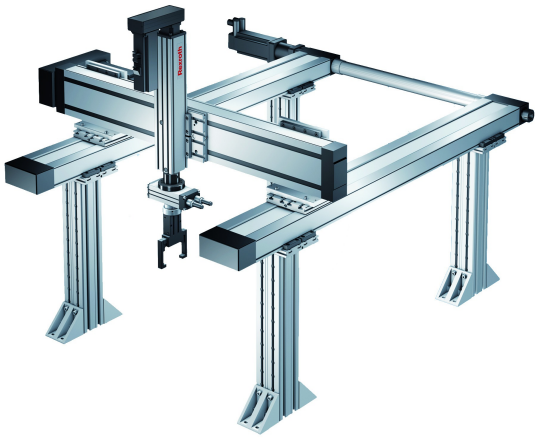


Figura 5. Robot cartesiano 3 GL [11].

3. **Robot Cilíndrico**: Con una articulación rotativa en la base y una prismática, estos robots operan en espacios de trabajo cilíndricos según lo ilustrado en la Figura 6. Son ideales para manipulación y ensamblaje simples.

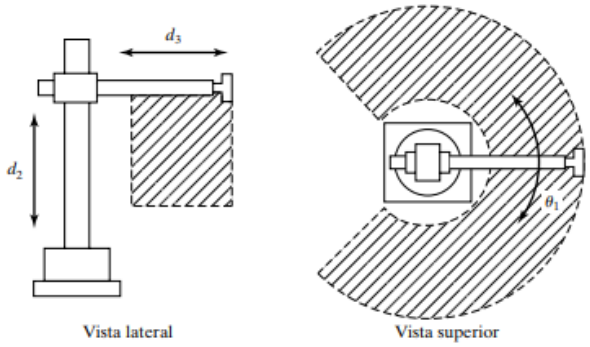


Figura 6. Robot cilíndrico 3 GL [12].

4. **Robot Esférico**: Ofrecen una combinación de articulaciones rotativas y lineales, operando en espacios de trabajo esféricos, con un excelente volumen de trabajo pero un alcance vertical limitado como se evidencia en la Figura 7.

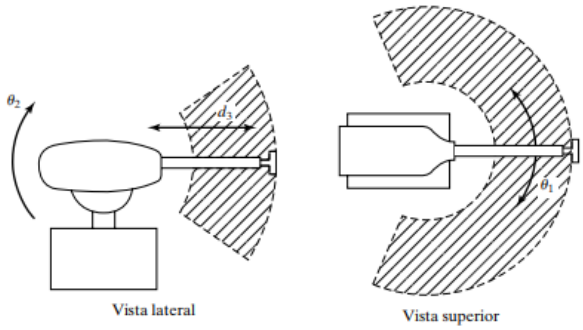


Figura 7. Robot esférico 3 GL [12].

5. **Robot SCARA**: Especializados en movimientos laterales rápidos, utilizados principalmente en aplicaciones de ensamblaje, teniendo como ejemplo el robot SCARA de la marca EPSON que se muestra en la Figura 8. Estos destacan por su velocidad y repetibilidad.

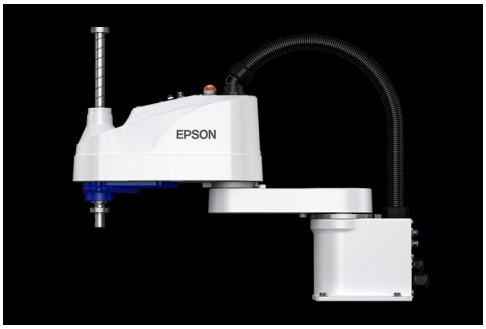


Figura 8. Robot SCARA 3 GL [11].

6. **Robot Delta**: Poseen un diseño de enlaces paralelos tal como se observa en la Figura 9. Son conocidos por su alta velocidad y precisión, ideales en aplicaciones rápidas de recoger y colocar.

Figura 9. Robot Delta 3 GL [13].

7. **Robot Colaborativo**: Estos robots, diseñados para trabajar de manera segura junto a humanos, son versátiles y se integran fácilmente en líneas de producción existentes, teniendo al brazo robótico de la marca UNIVERSAL ROBOT que se puede apreciar en la Figura 10.

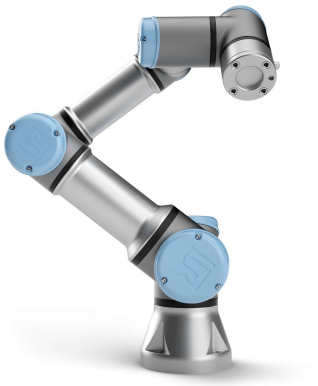


Figura 10. Robot colaborativo 6 GL [14].

Cada uno de estos robots industriales tiene sus ventajas y aplicaciones específicas, siendo seleccionados según las necesidades de cada proceso industrial [15]. A continuación se analizará un aspecto fundamental para la comprensión de un sistema robótico como lo es su programación.

IV-A3. Programación: Los sistemas robóticos de índole industrial desarrollados por la compañía KUKA implementan una codificación de programación exclusiva denominada KRL (KUKA Robot Language). Esto proporciona a los programadores la capacidad de generar secuencias de comandos para el control de movimientos, la administración de la ejecución de programas, la manipulación de entradas y salidas, la creación de subrutinas y la definición de funciones, y diversas funcionalidades adicionales [16].

Para la creación de un programa en el entorno del SmartPad, el proceso inicia con el acceso a la carpeta denominada R1. Una vez dentro, el usuario debe seleccionar la opción de crear un nuevo archivo, al cual se le asignará un nombre específico, marcando el comienzo de la fase de programación. La metodología de programación en KRL destaca por su eficiencia y simplicidad, ya que consiste fundamentalmente en posicionar el robot en las ubicaciones deseadas y guardar estas posiciones de manera secuencial. Este procedimiento facilita notablemente la configuración de las trayectorias requeridas para la ejecución de tareas específicas [9].

En el desarrollo de programas en KRL mediante el uso del SmartPad, es imprescindible familiarizarse con un conjunto de instrucciones específicas. Estas permiten ejecutar movimientos precisos del robot, adaptándose a las necesidades particulares de cada tarea a desempeñar. A continuación, se presentan algunos de los comandos:

- **PTP**: El término “point to point”, por sus siglas en inglés, se refiere a la capacidad de mover el robot de un punto a otro mediante la ruta más eficiente, que no necesariamente corresponde a una línea recta como se evidencia en la Figura 11. Esta funcionalidad es especialmente adecuada para aplicaciones que requieren precisión en tareas como la soldadura de puntos, el transporte de materiales, entre otras actividades.

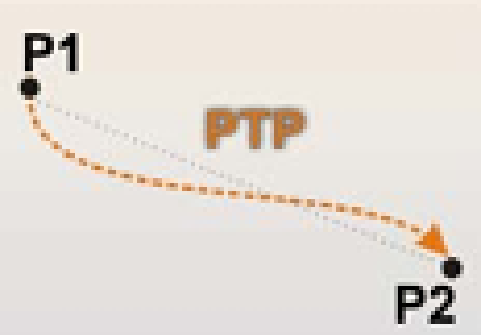


Figura 11. Trayectoria del robot con el comando PTP [9].

- **LIN**: El modo Lineal se refiere al movimiento de la herramienta del robot de un punto a otro, caracterizándose por seguir invariablemente una trayectoria rectilínea como se ilustra en la Figura 12. Esta modalidad es indispensable para aplicaciones que demandan una precisión lineal continua, como en procesos de soldadura a lo largo de trayectorias específicas, aplicaciones de pegado, corte láser, entre otras, garantizando una ejecución uniforme y precisa.

Figura 12. Trayectoria del robot con el comando LIN [9].

- **CIRC**: El comando Circular establece la definición de tres puntos clave: inicio, auxiliar y destino, tal como se ilustra en la Figura 13. Generalmente, se emplea en combinación con el comando LIN y resulta esencial para aplicaciones que requieren el trazado de curvas, circunferencias o la definición de radios específicos.

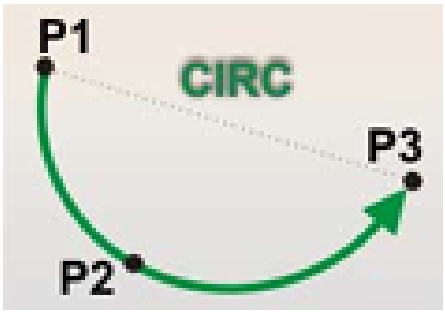


Figura 13. Trayectoria del robot con el comando CIRC [9].

A continuación, en la Figura 14 se presenta la estructura fundamental de un programa elaborado en el entorno del SmartPad. Cada sección posee un propósito específico en el desarrollo y funcionamiento del programa.

```
1  DEF kuka_rocks( )                                    (1)
2  INI                                                  (2)
3  PTP HOME  Vel= 100 % DEFAULT                         (3)
4  PTP P1 Vel=100 % PDAT1 Tool[1] Base[0]
5  PTP P2 Vel=100 % PDAT2 Tool[1] Base[0]
6  PTP P3 Vel=100 % PDAT3 Tool[1] Base[0]
7  OUT 1'' State=TRUE CONT
8  LIN P4 Vel=2 m/s CPDAT1 Tool[1] Base[0]
9  PTP HOME  Vel= 100 % DEFAULT
10 END                                                  (1)
```

Figura 14. Estructura de programación de robot KUKA [9].

1. “DEF nombre del programa()”, se muestra al comienzo cada programa y END, establece la finalización del mismo.

2. INI, contiene parámetros estándar indispensables para ejecutar correctamente el programa y siempre se deberá ejecutar primero.

3. Es la sección donde se puede comenzar a programar con los comandos de movimientos, instrucciones lógicas, entre otras. Siempre comienza y termina con un PTP HOME, el cual es establece la posición incial del robot.

IV-A4. Actuador final: En el ámbito de la robótica industrial y automatización, son mayormente usados los ‘Grippers’ estos son componentes destinados a la sujeción y manipulación de objetos en entornos operativos [17].

Su clasificación esta dada por múltiples diseños, cada uno pensado para facilitar el manejo de objetos con características diversas, incrementando así la eficiencia en aplicaciones industriales y de investigación. Se presentarán los tipos de pinzas más empleados:

1. **Gripper de 2 dedos**: Se encuentran entre las opciones más utilizadas para la sujeción de objetos como lo ilustrado en la Figura 15, adaptándose con facilidad a la geometría de cada pieza.

Figura 15. Gripper estándar eléctrico [18].

2. **Gripper de 3 dedos**: Se emplea para el agarre con una mayor precisión en componentes con geometrías complejas tal como se puede observar en la Figura 16. Garantizando una fijación efectiva incluso en las formas más exigentes.

Figura 16. Gripper adaptable de 3 dedos [19].

3. **SoftGripper de 4 dedos**: Este tipo de “Gripper” se utiliza principalmente para la manipulación delicada de objetos de forma irregular. Está fabricado con materiales elásticos que se pueden inflar, permitiendo así una deformación controlada para adaptarse perfectamente a la geometría del objeto manipulado, su funcionamiento se puede evidenciar en la Figura 17.

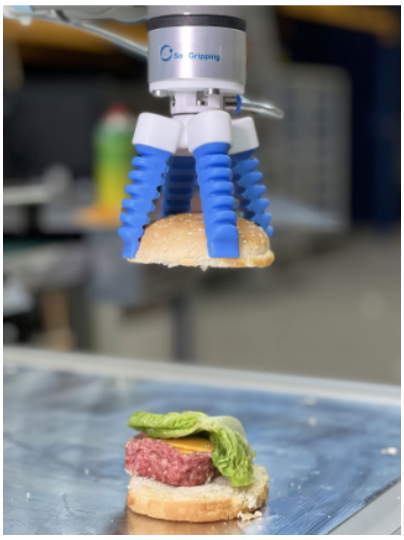


Figura 17. SoftGripper ortientado a la manupulacion de geometrias irregulares [20].

4. **Gripper de vacio**: Estos dispositivos operan independientemente de un suministro de aire, ya que emplean un mecanismo eléctrico para generar el vacío necesario entre el objeto y las ventosas, Como evidencia en la Figura 18. Dicha característica los hace óptimos para ser utilizados en diversas aplicaciones como el empaque, paletizado y almacenamiento, entre otras.

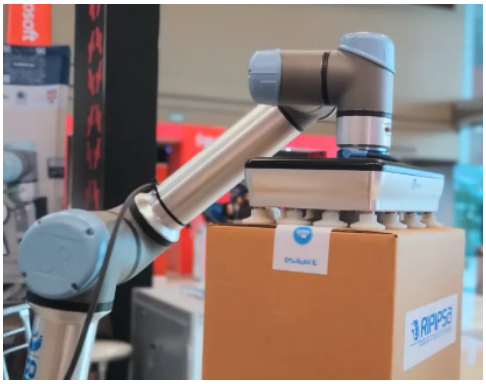


Figura 18. Gripper de vacio modelo VGP20 [20].

IV-A5. Sistemas de accionamiento: Para llevar a cabo el accionamiento del Gripper, la industria robótica recurre principalmente a al uso de tres sistemas, destacando su eficacia y adaptabilidad a diversas aplicaciones [21]. Estos sistemas son:

- **Neumático**: Los sistemas neumáticos son los mas utilizados a en la industria de la robótica, ya que ofrecen rapidez y ligereza para espacios reducidos, se debe integrar el uso de electroválvulas, mangueras y un controlador.

- **Eléctrico**: Este sistema destaca por su alta precisión, control programable y eficiencia energética. Ofrecen un mantenimiento reducido y se integra de manera sencilla a sistemas modernos de automatización.

- **Hidráulico**: Los sistemas de accionamiento hidráulico para Grippers destacan en la automatización industrial por su capacidad para manejar cargas pesadas con alta fuerza y precisión. Aprovechan fluidos a presión para controlar el agarre, estos requieren mantenimiento cuidadoso y una instalación más compleja.

La elección del sistema de accionamiento para un "Gripper" ya sea neumático, hidráulico o servoeléctrico debe basarse en las exigencias específicas de cada aplicación, tomando en consideración aspectos clave como el peso, el espacio disponible y la eficiencia operativa. Los neumáticos ofrecen rapidez y ligereza para espacios reducidos, los hidráulicos brindan robustez para cargas pesadas, y los servoeléctricos destacan por su facilidad de programación y bajo mantenimiento. La selección cuidadosa de estos sistemas optimiza el rendimiento en diversas tareas de automatización [21].

IV-B. Kinect

El Microsoft Kinect presenta una composición que abarca una cámara RGB convencional, un sensor de profundidad que comprende una cámara de infrarrojos (IR) y un proyector, junto a un micrófono y un motor incorporado, estos componentes se muestran en la ilustración 19. Originariamente concebido para el control de juegos en la consola Xbox 360 de Microsoft, el sensor Kinect permite a los jugadores emplear sus manos en lugar de un controlador para manipular personajes o menús en un entorno de juego.

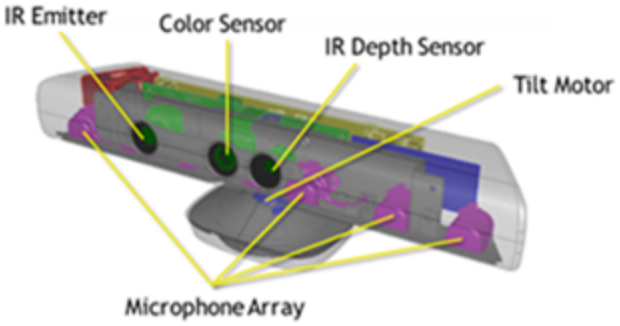


Figura 19. Partes de la cámara Kinect V1 de Xbox 360 [22].

A lo largo del tiempo, Microsoft ha desarrollado diversas generaciones de este hardware, teniendo al Kinect V1 (Xbox 360), V2 (Xbox One) y V3 o más conocida como "Azure". Cada nueva versión ha presentado avances en la eficacia, alcance y capacidad de los sensores, así como en la interpretación de datos en tiempo real. Además de su aplicación en juegos, existen múltiples bibliotecas disponibles que amplían las posibilidades del sensor Kinect hacia campos relacionados con la visión por computadora y el mapeo 3D, robótica, medicina, rastreo humano, entre otros [23].

IV-B1. Modelo v1: El dispositivo Kinect v1, teniendo para este modelo una generación de 20 puntos del esqueleto, ha sido ampliamente empleado para el seguimiento de movimientos y la captura de la estructura corporal en tiempo real, la ilustración de este dispositivo se aprecia en la Figura 20. Utilizando una combinación de tecnologías como sensores infrarrojos y cámaras RGB, el Kinect v1 es capaz de generar mapas tridimensionales de profundidad que permiten detectar la posición y la orientación de las articulaciones del cuerpo humano. Estos datos de profundidad y RGB son procesados mediante algoritmos de visión por computadora, lo que posibilita la creación de modelos esqueléticos virtuales en tiempo real, permitiendo así el seguimiento y la representación precisas de las 20 articulaciones clave del cuerpo [24].

Figura 20. Kinect modelo V1 Xbox 360 [25].

IV-B2. Comparativa entre versiones de Kinect: Para ofrecer una visión detallada de la evolución y las mejoras tecnológicas introducidas en el dispositivo Kinect de Microsoft, se elaboró una comparativa detallada, mostrada en la tabla I. En la misma se desglosa las especificaciones y capacidades distintivas de cada versión, desde el Kinect V1 diseñado para Xbox 360, pasando por el Kinect V2 para Xbox One, hasta llegar al más reciente, el Kinect Azure.

Tabla I
TABLA COMPARATIVA DE LAS VERSIONES DE KINECT [26].

Características	Kinect V1	Kinect V2	Kinect Azure
Cámara De Color	640 x 480 30fps	1920 x 1080 30fps	3840 x 2160 30fps
Cámara De Profundidad	320 x 240 30fps	512 x 424 30fps	1024 x 1024 30fps
Cámara Infrarrojo	320 x 240 30fps	512 x 424 30fps	1024 x 1024 15fps
Distancia De Profundidad Máxima	4.5 m	4.5 m	4.5 m
Distancia De Profundidad Mínima	40 cm	50 cm	25 cm
Campo De Visión Horizontal	57 grados	70 grados	120 grados
Campo De Visión Vertical	43 grados	60 grados	120 grados
Motor De Inclinación	SÍ	NO	NO
Puntos De Esqueleto	20 puntos	26 puntos	32 puntos
Esqueletos Rastreables	2 personas	6 personas	6 personas
Conectividad	USB tipo A 2.0	USB tipo A 3.0	USB tipo C
Compatibilidad	Windows 7, 8, 10	Windows 7, 8, 10	Windows 7, 8, 10
Precio Ecuador	30$	50$	599$

IV-B3. Seguimiento de esqueleto: El skeleton tracking, o seguimiento de esqueletos, es un enfoque técnico utilizado en visión por computadora para detectar y rastrear las posiciones y movimientos de las articulaciones y huesos del cuerpo humano, se puede visualizar de mejor forma en la Figura 21. Este proceso implica la extracción y análisis de información tridimensional de imágenes o videos, permitiendo la estimación precisa de la estructura esquelética en tiempo real. Sus aplicaciones versátiles en campos como la realidad aumentada, la interacción humano-computadora, la biomecánica y la animación por ordenador. En robótica, el seguimiento de esqueletos facilita la comunicación y cooperación entre humanos y robots [27].

Figura 21. Representación visual de la estructura del esqueleto [28].

IV-C. Sistema de teleoperación

La teleoperación constituye un método para dirigir y controlar un robot de forma remota, a través de medios electrónicos, ya sea por internet o por redes en especifico. Esto posibilita a un operador ejecutar comandos y acciones sobre el robot desde un lugar a disntacia, lo cual exige una interfaz humano-máquina adecuadamente diseñada. Un sistema de teleoperación, en términos generales, comprende en la capacidad que tiene el operador para observar, guiar y llevar a cabo actividades en entornos distantes o riesgosos sin requerir su presencia física en el sitio de la actividad [29].

IV-C1. Clasificación:

1. **Teleoperación remota**: Es un sistema que da la posibilidad al operador controlar y monitorear dispositivos o robots desde una ubicación remota. Este sistema es eficiente para la realización de tareas en diferentes entornos, esto da lugar a que el operador envíe instrucciones y reciba una retroalimentación en tiempo real para realizar tareas con eficiencia y exactitud [30].

2. **Control bilateral**: Establece una retroalimentación de esfuerzos hacia el operador, combinada se logra mediante la retroalimentación cinestésica, convirtiendo las fuerzas de contacto del esclavo en una fuerza aplicada sobre la mano del operador. Sin embargo, en este enfoque surgen desafíos al sincronizar los movimientos entre el maestro y el esclavo, junto con la transferencia precisa de fuerzas en un entorno con retardos de comunicación [31].

3. **Control coordinado**: Implica sincronización precisa de los movimientos y acciones entre un operador humano y un brazo robótico a distancia, garantizando una comunicación bidireccional efectiva y la reducción de la latencia para lograr la ejecución precisa de tareas [32].

4. **Teleoperación autónoma con control compartido**:
   La teleoperación autónoma con control compartido fusiona la operación dirigida por humanos y la ejecución autónoma, lo que permite una colaboración eficiente entre un operador y un agente autónomo. Esta modalidad ofrece ventajas como una finalización más rápida de las tareas y una mayor precisión. Se resalta un sistema colaborativo de corrección de errores, el cual simplifica la solución de problemas inesperados y mejora la eficiencia temporal de la ejecución. [33].

IV-C2. Aplicaciones: El sistema de teleoperación ha sido ampliamente utilizado en diversos campos, brindando una valiosa ayuda en aplicaciones prácticas. Por lo general, estas aplicaciones combinan una variedad de tecnologías relacionadas con la percepción, el control y el aprendizaje [34]. Entre estas, se pueden destacar las siguientes:

1. **Búsqueda y rescate**:
   El presente sistema incorpora tecnologías de realidad virtual y un robot de diseño centauro destinado a operaciones de rescate. Utiliza Unity3D para generar el entorno virtual y establece una conexión efectiva entre el robot y Unity-Matlab para un manejo remoto preciso. Este método mejora significativamente la interacción entre el operador y el robot, optimizando la realización de tareas en contextos de alto riesgo, donde son esenciales la exactitud y la rapidez de respuesta, un ejemplo de ello se puede apreciar en la Figura 22 [35].

Figura 22. Asistencia con botiquín de primeros auxilios [35].

   Las evaluaciones efectuadas demuestran la capacidad del sistema para manejar objetos de manera delicada y prestar asistencia en emergencias, sin comprometer la seguridad del operador. Esta característica enfatiza la relevancia de la realidad virtual en la robótica aplicada al rescate. La implementación de esta tecnología propone alternativas para la gestión de operaciones críticas de manera segura y establece una base para el avance de los sistemas de teleoperación en contextos de alta peligrosidad.

2. **Seguridad industrial en entornos nucleares**:
   Este sistema está diseñado para la operación remota de robots en entornos de alto riesgo, como es el caso de la manipulación de sustancias nucleares, se emplea una estructura de control unidireccional, la cual no posee retroalimentación táctil, esto hace referencia a la ilustración 23. Este sistema se vale del uso innovador del dispositivo HTC Vive, permitiendo que los usuarios ejecuten comandos de manipulación robótica de forma remota, a través de gestos manuales intuitivos. Este enfoque destaca por su simplicidad operativa y la reducción significativa en la curva de aprendizaje requerida para su manejo eficiente. La incorporación de esta tecnología tiene el objetivo primordial de reforzar los protocolos de seguridad y optimizar los procesos de manejo de materiales radioactivos, facilitando así la interacción humana en la cadena de decisiones críticas [36].

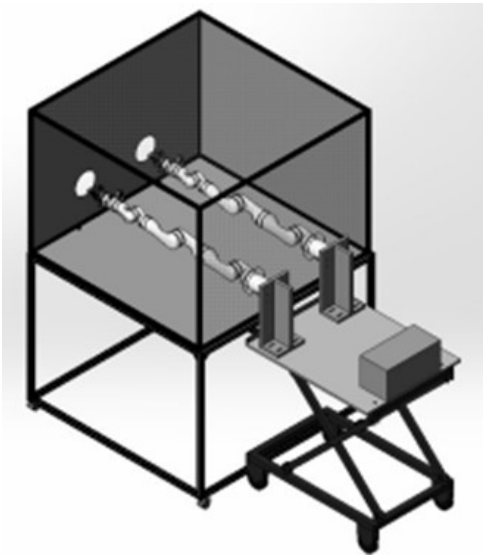


Figura 23. Representación del hardware de un modelo de guanteleta nuclear [36].

Los resultados del proyecto muestran avances significativos en la integración de robótica y tecnologías de inteligencia artificial para operaciones seguras y eficientes dentro de entornos nucleares peligrosos. Mediante la implementación de manipulación robótica automatizada, tecnologías de teleoperación asistiva y sistemas de monitoreo para la detección temprana de fallos, el proyecto contribuye a establecer interfaces más seguras y eficientes para el manejo de materiales nucleares y objetos contaminados.

3. **Fabricación flexible**:
   El sistema de teleoperación se basa en un control colaborativo humano-robot para cirugías mínimamente invasivas teleoperadas, empleando un robot redundante con siete grados de libertad. Este enfoque aprovecha la redundancia del robot para cumplir con la restricción de centro de movimiento remoto (RCM), asegurando una operación precisa y segura, al mismo tiempo que permite una interacción complaciente con el operador, tal y como se aprecia en la Figura 24. La aplicación se enfoca en mejorar la precisión y seguridad en cirugías mínimamente invasivas, reduciendo el impacto físico en el paciente y mejorando la interacción entre el operador y el sistema robótico [37].

Figura 24. Procedimiento quirúrgico teleoperado llevado a cabo por un operador, bajo la supervisión de un médico. [37].

Los resultados obtenidos demuestran una mejora significativa en la precisión de la punta quirúrgica y la adherencia a la restricción de RCM, además de una reducción notable en la fuerza de interacción sobre la pared abdominal del paciente. Esto se logra mediante la integración de un compensador adaptativo que ajusta el comportamiento del robot en tiempo real. Este avance representa un paso significativo en la robótica quirúrgica, ofreciendo un control más intuitivo y efectivo para el operador, y garantizando procedimientos quirúrgicos mínimamente invasivos más seguros y precisos.

IV-D. RoboDK

RoboDK se destaca como una herramienta integral tanto para la simulación como para la programación en entornos en línea y fuera de línea. Esta plataforma permite a sus usuarios diseñar, simular y elaborar secuencias de programación destinadas a una amplia gama de controladores de robots, incluyendo brazos robóticos especializados. Además, facilita la realización de diversas operaciones, abarcando desde tareas de manipulación y posicionamiento "Pick and place" hasta aplicaciones complejas como la impresión 3D mediante el uso de sistemas robóticos [38].

Este software ofrece una API mejor conocida como Interfaz de Programación de Aplicaciones, la cual esta disponible para los lenguajes de programamción de Python, C#, C++, Visual Basic y Matlab. Esta se compone de una variedad de comandos y procedimientos, los cuales están integrados en un lenguaje de programación versátil, el uso de la misma se puede apreciar en la Figura 25. Esta funcionalidad permite programar una amplia gama de robots con un enfoque de programación homogéneo [39].

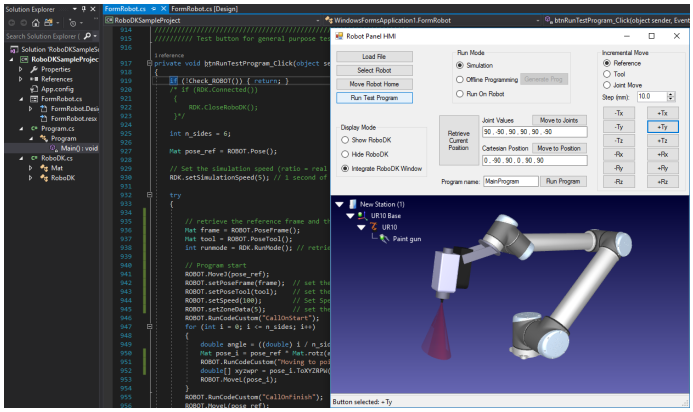


Figura 25. API de RoboDK en C#: Interfaz de Programación de Robots [40].

IV-D1. Aplicaciones: RoboDK se destaca en el ámbito de la robótica por su amplia gama de aplicaciones prácticas, adaptándose a las necesidades de diversos sectores industriales y de investigación. Este software ofrece herramientas avanzadas para la simulación y programación de robots, permitiendo la implementación de procesos automatizados con precisión y eficiencia [39]. Listando las que ofrece la plataforma son:

1. Pick-and-place
2. Mecanizado Robótico
3. Mecanizado con ejes externos
4. Pintura
5. Soldadura por puntos
6. Impresión 3D
7. Cinta transportadora
8. Programa DXF a robot
9. Simulación de visión 2D
10. Dibujando con un robot

V. Marco Metodológico

El objetivo principal de este proyecto es desarrollar un sistema de teleoperación para un brazo robótico, utilizando el sensor Kinect para la captura del movimiento. La implementación de este sistema permitirá replicar los movimientos de un brazo humano en tiempo real, logrando que el operador tenga un mayor control sobre el robot. Para ello se elaboró un diagrama de flujo que se muestra en la Figura 26 en donde se visualiza el proceso de desarrollo del sistema con todas sus etapas.

Figura 26. Diagrama de flujo del sistema de teleoperación, por D. Córdova.

V-A. Skeleton tracking y adquisición de datos

En el contexto de este estudio, se empleó el sensor Kinect V1 de Xbox 360 para llevar a cabo el seguimiento del esqueleto humano. Este dispositivo utiliza una combinación de sensores de profundidad e infrarrojos que permiten capturar con precisión los movimientos del cuerpo humano en un entorno tridimensional. La programación del Kinect se realizó en el lenguaje C#, el cual es ampliamente utilizado por Microsoft en el desarrollo de aplicaciones para este dispositivo. Se eligió este debido a su eficiencia en el procesamiento de datos y su compatibilidad con las bibliotecas y herramientas proporcionadas por Microsoft para el desarrollo de aplicaciones Kinect.

Para habilitar el uso del Kinect, es necesario instalar su SDK, que proporciona acceso a las funciones del dispositivo, así como el Developer Toolkit, que ofrece ejemplos y recursos para el desarrollo de aplicaciones Kinect. Se recomienda utilizar la IDE de Microsoft Visual Studio debido a sus características avanzadas y su integración fluida con el ecosistema de desarrollo de Microsoft.

Para adaptar los ejemplos proporcionados en el Developer Toolkit a las necesidades específicas de este estudio, se descargó y modificó el ejemplo Skeleton Basics D2D. Este ejemplo se utilizó como punto de referencia para desarrollar una función que asigna un sistema de referencia basado en el cuello del sujeto, a partir del cual se miden las distancias a cada una de las articulaciones del brazo derecho en el seguimiento del esqueleto, en la figura 27 se muestran los puntos referenciales mencionados con anterioridad.

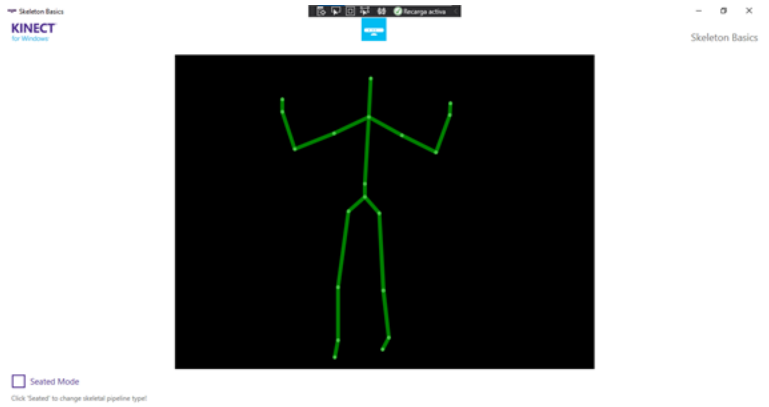


Figura 27. Skeleton Tracking, por D. Córdova.

Se ha incorporado también un sistema de control para el accionamiento del gripper del robot mediante comandos de voz, facilitando la apertura o cierre del efector final. Esta funcionalidad se logra a través del uso del reconocimiento de voz y el micrófono integrado en Kinect. A partir de aquí se extraen los datos de las coordenadas tridimensionales de la mano derecha, y el estado que va a tener el gripper el cual se representa de forma binaria, 1 es para cerrar y 0 es para abrir, estos datos serán almacenados en el archivo .JSON con una estructura definida, así como se detalla en la Tabla II.

Tabla II
TABLA DE DATOS DEL ARCHIVO.

N°	Variable	Datos capturados	Tipo de Dato
1	X	126.6339	Milímetros
2	Y	31.7236	Milímetros
3	Z	12.3486	Milímetros
4	EstadoGripper	1	Binario

V-B. Controlador

Los datos capturados y almacenados temporalmente en formato .JSON serán procesados en Python. Se recomienda descargar e instalar la versión 3.8.10 de Python y utilizar el entorno de desarrollo Visual Studio Code para la escritura y ejecución del código. Además, se debe instalar la API de RoboDK, para lo cual se accede al buscador de extensiones de Visual Studio Code y se instala la extensión correspondiente. Esta API facilitará la interacción con la plataforma de simulación y programación de robots industriales. En el desarrollo del código en Python se hará uso de cuatro librerías fundamentales:

- **json**: Utilizada para la manipulación de datos estructurados en formato .JSON.
- **time**: Facilita la gestión de tiempos y retardos en la ejecución del código.
- **robolink**: Componente esencial de la API de RoboDK, dedicada a la simulación y programación de robots industriales.
- **robodk**: Otra parte fundamental de la API de RoboDK, se utiliza para realizar cálculos matemáticos y otras funciones relacionadas con la manipulación de robots.

Se configura un bucle infinito encargado de la lectura constante de los datos desde un archivo .JSON, el cual almacena tanto las coordenadas de interés como el estado asignado al gripper. Dentro de este ciclo, se procede a

la extracción de las coordenadas tridimensionales correspondientes a la mano derecha del esqueleto, así como al estado que adoptará el efector final. El desplazamiento del robot hacia la ubicación deseada se realiza mediante la función 'MoveJ'. Para modificar el estado del gripper, ya sea para abrirlo o cerrarlo, se emplea la función 'setDO'. Con el fin de mantener una sincronización óptima entre las actualizaciones del sistema, se integra un intervalo de espera corto dentro del control del ciclo.

V-C. Comunicación

Para establecer la comunicación entre el controlador del brazo robótico y el computador, se seleccionó el protocolo UDP (User Datagram Protocol) para la transmisión de datos en tiempo real, debido a su baja latencia y la sencillez del protocolo, características que facilitan transmisiones rápidas y eficaces sin requerir confirmaciones de recepción. Para facilitar el intercambio de información entre ambos dispositivos, se optó por el uso de un cable de red cruzado, el cual es idóneo para la conexión entre equipos, tal y como se puede apreciar en la ilustración 28.

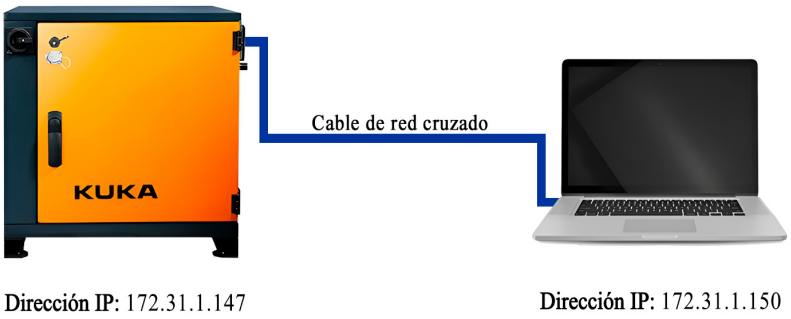


Figura 28. Direcciones IP del controlador y computador, por D. Córdova.

A fin de asegurar una comunicación efectiva dentro de la misma red, se realizaron las adecuadas configuraciones de red, empleando direcciones IP específicas para cada dispositivo. Se implementó el software "C3Bridge Server" como intermediario entre RoboDK y el robot KUKA, estableciendo un "puente" que facilita la integración y el flujo de información entre el software de simulación y el hardware. Para garantizar la correcta ejecución de los comandos de movimiento en el robot, fue necesario ejecutar el programa "RoboDKsync562" en modo "AUT" desde el SmartPad. Este paso es crucial para lograr una sincronización efectiva, asegurando que los comandos enviados sean recibidos y ejecutados adecuadamente en el robot.

V-D. Puesta en marcha

El robot KUKA KR10 muestra una notable capacidad para realizar seguimiento en tiempo real de los movimientos del brazo derecho del operador. Esta característica distintiva del sistema posibilita una interacción fluida y sin precedentes entre el usuario y el robot, optimizando significativamente la ejecución de tareas. Gracias a esta tecnología avanzada, el operador puede controlar el robot de manera remota con una gran facilidad.

Este proceso es parte de un ciclo continuo que se inicia nuevamente con la detección del skeleton tracking realizado en el usuario. Este ciclo continuo asegura una interacción constante y dinámica entre el operador y el robot, permitiendo ajustes en tiempo real y una sincronización en los movimientos, lo que eleva aún más la eficiencia y efectividad de las operaciones robóticas.

Para la ejecución del sistema de teleoperación, es necesario inicializar primero el programa "C3Bridge" en el SmartPad. A continuación, se deben ejecutar los programas desarrollados en C# y Python. Finalmente, se activa el

modo “AUT” en el programa para que este pueda interpretar los datos enviados por el Kinect a través de RoboDK. Los pasos mencionados se ilustran de manera más detallada en el diagrama de flujo de la Figura 29.

Figura 29. Diagráma de puesta en marcha del sistema de teleoperación, por D. Córdova.

VI. Resultados

VI-A. Pruebas de campo

Para asegurar la eficacia del sistema de teleoperación, se implementaron pruebas utilizando un mecanismo de Pick and Place, con el uso de un Gripper flexible, mostrado en la sección de Anexos Figura 46. Con el objetivo de evaluar su precisión, se llevaron a cabo diez ensayos utilizando latas de refrescos como objetos de prueba. Esta metodología permitió cuantificar el número de aciertos logrados, facilitando así una evaluación detallada del desempeño del sistema. Para una representación clara de los resultados, se elaboró una tabla comparativa que destaca tanto los aciertos como los desaciertos observados durante las pruebas. Este enfoque metodológico no solo demuestra la viabilidad del sistema sino que también proporciona información valiosa para futuras optimizaciones.

Tabla III
Resultados del sistema de teleoperación.

N° de pruebas	Resultado	Tiempo de respuesta del robot	Tiempo de prueba	Precisión del movimiento
1	Éxito	0.95s	22s	Alta
2	Éxito	0.94s	30s	Media
3	Fallo	0.96s	40s	Baja
4	Éxito	0.92s	17s	Alta
5	Éxito	0.94s	21s	Alta
6	Éxito	0.92s	22s	Media
7	Éxito	0.93s	18s	Alta
8	Éxito	0.95s	20s	Alta
9	Éxito	0.94s	15s	Alta
10	Éxito	0.93s	17s	Alta

La siguiente muestra corresponde a una de las diez pruebas realizadas utilizando la técnica de Pick and Place. Esta prueba se ilustra de manera más clara en la Figura 30, donde se observa al operador manipulando una lata de refresco. En la Figura 31, se aprecia cómo la lata es colocada en otro lugar, con el propósito de validar el funcionamiento integral del sistema de teleoperación. Este método se ha empleado para evaluar la capacidad del sistema en la manipulación precisa de objetos a distancia, demostrando así su eficacia y fiabilidad en entornos controlados.

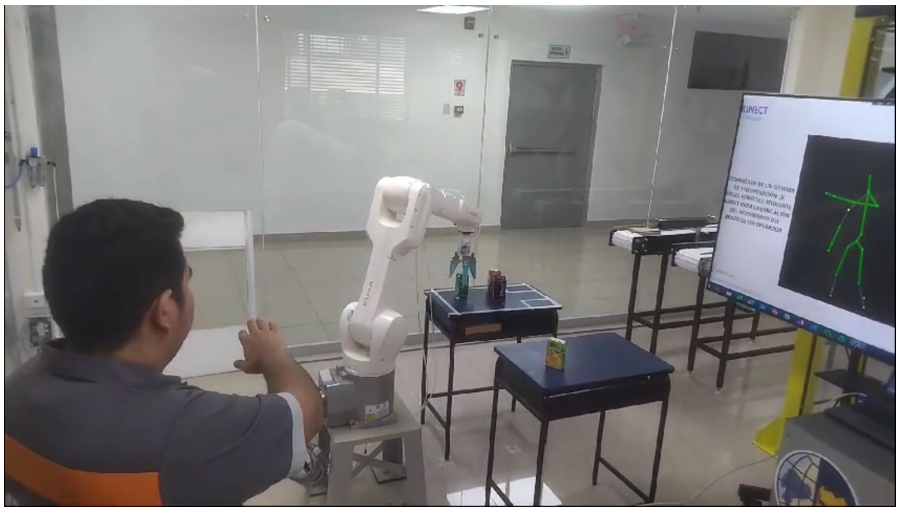


Figura 30. Operador posicionando el gripper para la toma del objeto, por D. Córdova.

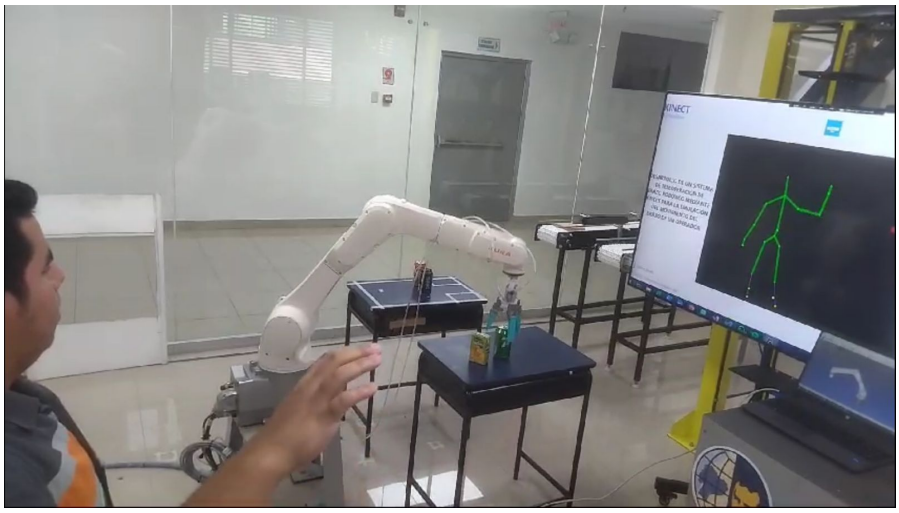


Figura 31. Operador colocando el objeto en otra area, por D. Córdova.

Una de las pruebas diseñadas para evaluar la precisión y destreza del operador consistió en apilar tres latas de refresco una sobre otra, como se muestra en la Figura 32, donde el operador maniobra el gripper para tomar la segunda lata de refresco. En la Figura 33, se observa cómo el operador traslada la segunda lata y la apila sobre la primera de manera correcta y precisa. Finalmente, en la Figura 34, se puede apreciar una tercera lata apilada exitosamente, formando una torre de tres refrescos. Este proceso permite evaluar la destreza y precisión del operador al ejecutar tareas complejas.

Figura 32. Proceso de recolección de la segunda lata, por D. Córdova.

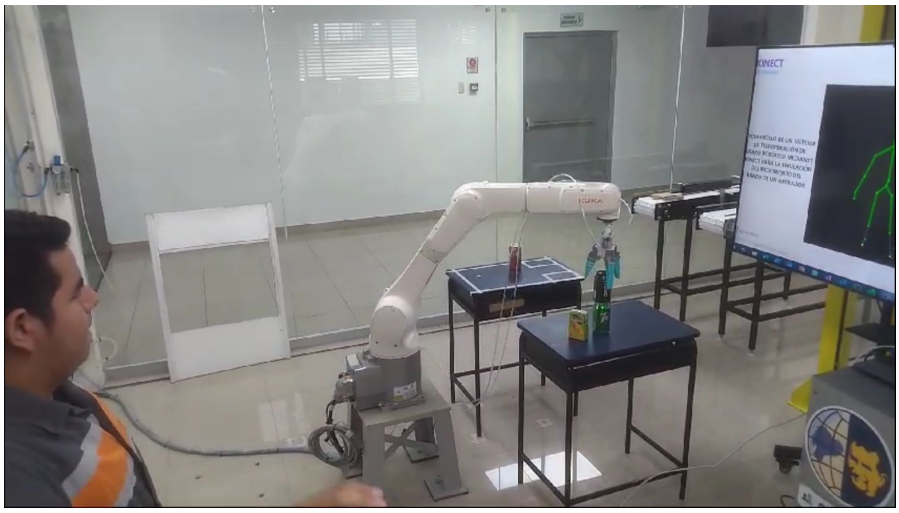


Figura 33. Posicionamiento exitoso de la tercera lata en la pila, por D. Córdova.

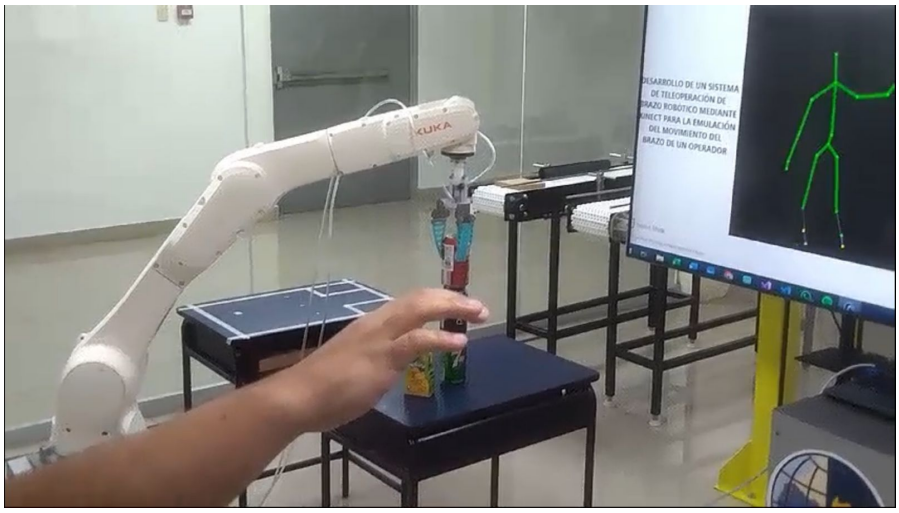


Figura 34. Operador posicionando el robot para la toma del objeto, por D. Córdova.

VI-B. Análisis e interpretación de datos

Las pruebas de campo se llevaron a cabo utilizando el Skeleton Tracking de Kinect, como se muestra en la Figura 35. Los movimientos capturados por el seguimiento de esqueleto sirvieron de base para definir un objetivo, hacia el cual se debía posicionar la mano derecha. Esto permitió registrar la trayectoria de la mano hasta alcanzar la posición deseada, resultando en la obtención de una gráfica que representa la posición euclidiana, tal como se muestra en la Figura 39.

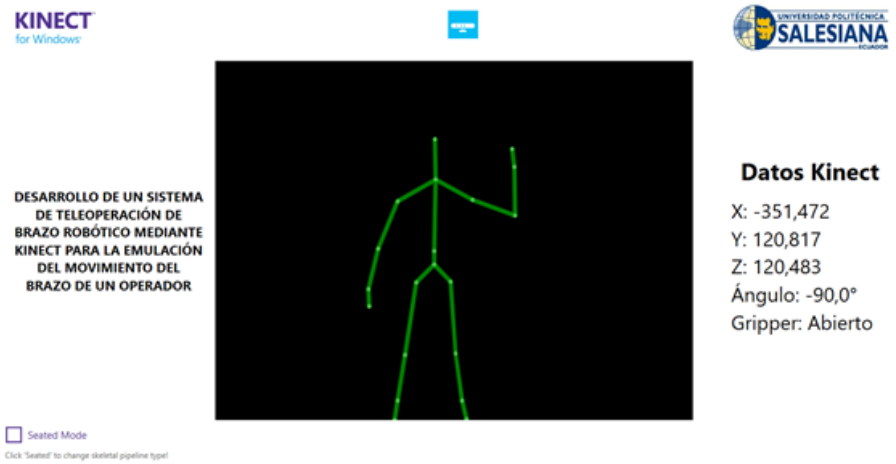


Figura 35. Posición de la mano derecha en un punto establecido, por D. Córdova.

Por consiguiente, el robot procede a reubicarse a la coordenada final establecida por el operador, llevando a cabo un desplazamiento controlado y preciso. Este movimiento se efectúa conforme a un algoritmo predefinido de navegación y posicionamiento, cuyo resultado se puede apreciar gráfica de la Figura 36.

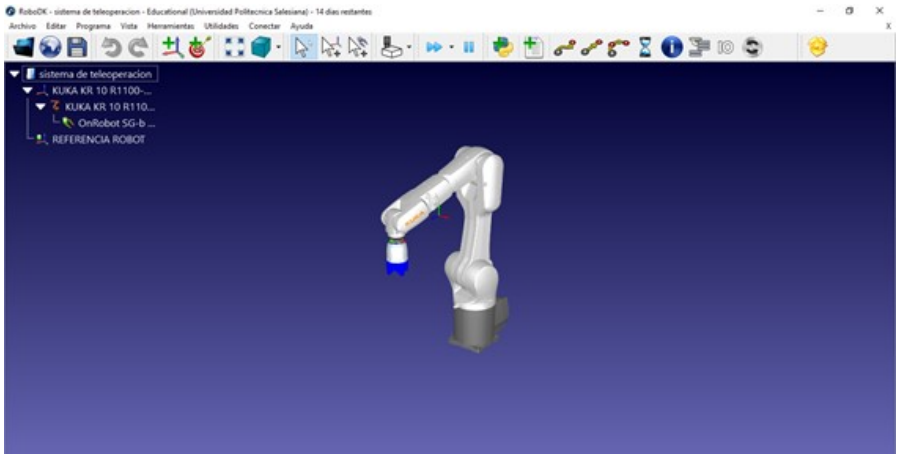


Figura 36. Control de la posición del robot en RoboDK mediante el operador, por D. Córdova.

Se llevaron a cabo diversas posturas ejecutadas por el operador para la consecución del seguimiento robótico. Una de estas posturas está ilustrada en la Figura 37, donde se demuestra la capacidad de respuesta del robot ante las distintas configuraciones espaciales adoptadas por el operador. Este proceso no solo pone a prueba la agilidad y la flexibilidad del sistema de seguimiento, sino que también verifica la precisión con la que el robot puede replicar y mantener las poses indicadas

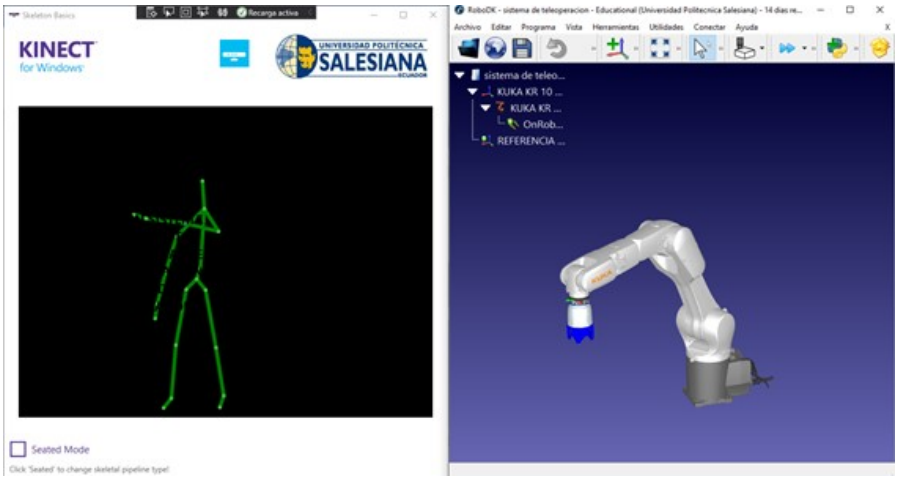


Figura 37. Replica de la posición del robot mediante el control del operador, por D. Córdova.

Para examinar detalladamente la trayectoria y el posicionamiento de la mano derecha en el espacio tridimensional, se procedió a integrar y analizar los datos recabados desde el inicio del movimiento hasta su conclusión en el punto de destino preestablecido. Esta información se ha representado gráficamente para una interpretación más intuitiva y se puede observar en la Figura 38.

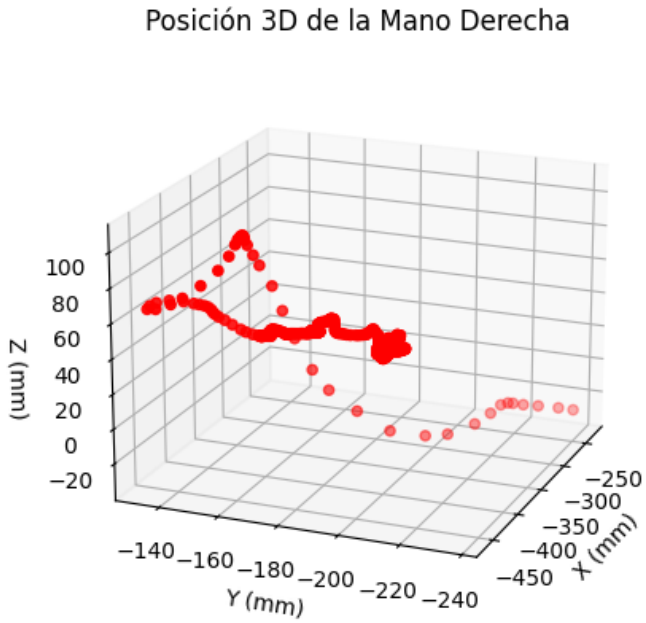


Figura 38. Gráfica tridimensional la trayectoria de la mano derecha, por D. Córdova.

En la imagen proporcionada, observamos un gráfico 3D que nos muestra la trayectoria de la mano derecha en el transcurso de un movimiento específico. Los puntos rojos indican la posición de la mano en distintos instantes de tiempo, mientras que la línea de unión más gruesa denota la trayectoria seguida. Las coordenadas X, Y y Z se presentan en milímetros y revelan el rango de movimiento en cada eje.

Para cuantificar la distancia del sistema, se implementó un protocolo de prueba que requiere el posicionamiento de la mano derecha en un objetivo designado, seguido por su retención en esa ubicación específica. Los datos

recopilados durante esta evaluación se presentan en la Figura 39.

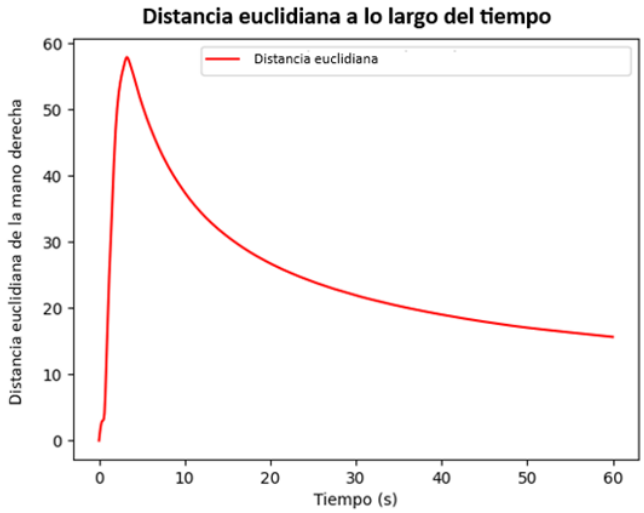


Figura 39. Gráfica de la distancia euclidiana de la mano derecha vs tiempo, por D. Córdova.

La gráfica muestra la distancia euclidiana en función del tiempo. En el gráfico se puede observar que, inicialmente, hay un pico significativo del deplazamiento que corresponde al momento en el que la mano busca y se dirige hacia el objetivo. Posteriormente, la línea muestra una tendencia descendente pronunciada, lo que indica que la posición de la mano se mantiene en el objetivo. Este descenso sugiere que el sistema y el sujeto ganan estabilidad en la retención de la posición a medida que transcurre el tiempo, alcanzando una precisión más constante y reducida al final del periodo registrado, lo cual es un indicador de la efectividad del sistema en la tarea asignada.

VI-C. Procedimiento para puesta en marcha

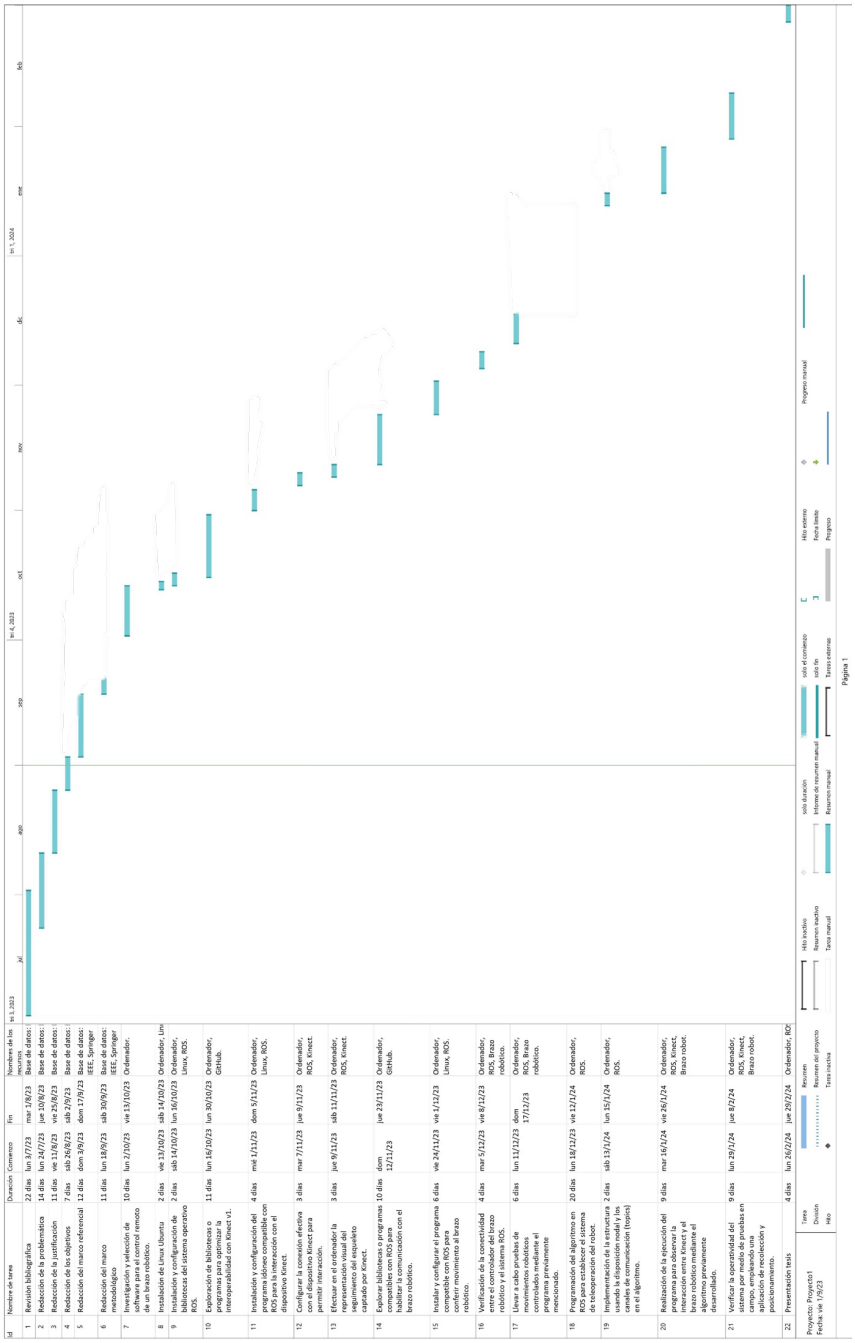
Para garantizar un uso y funcionamiento adecuados del robot, es esencial considerar los siguientes pasos para la activación y utilización óptima del sistema de teleoperación, los cuales se detallan a continuación:

1. Conectar el cable de red cruzado al puerto X66 (KLI) del controlador.
2. Al encender el robot, se debe ejecutar el programa “C3 Bridge Server” para establecer la comunicación.
3. Ejecutar el programa “RoboDKsync562” en modo “AUT” mediante mediante el SmartPad.
4. Iniciar Microsoft Visual Studio y ejecutar el programa diseñado para el seguimiento del esqueleto (skeleton tracking) con el sensor Kinect, lo cual permite generar un archivo en formato “.JSON” con los datos a utilizar.
5. Abrir la estación de RoboDK que contiene el modelo virtual del robot KUKA KR10.
6. Activar la comunicación en tiempo real entre el modelo virtual y el robot físico mediante la ejecución del script de Python.
7. Posicionarse frente a la cámara Kinect facilita el seguimiento de los movimientos por parte del sistema.

VII. CRONOGRAMA

A continuación se muestra el cronograma de trabajo en la tabla IV.

Tabla IV
CRONOGRAMA



VIII. PRESUPUESTO

Tabla V
PRESUPUESTO DE ELEMENTOS.

Nombre del elemento	Ilustración	Descripción	Cantidad	Valor total
Cámara Kinect V1		Cámara con sensor 3D de profundidad	2	40 $
Cable de red cruzado		Longitud de 6 metros	1	5 $
Gripper Neumático		Gripper con sujetador flexible	1	150 $
Acople para manguera neumática		Acople reductor de 1/4 a 1/8 de pulg	2	6 $
Manguera neumática 1/4 pulg		Longitud 2 metros	2	5 $
Manguera neumática 1/8 pulg		Longitud 2 metros	2	5 $
Gastos varios		Transporte hacia la universidad, en elementos y alimentación	1	500 $
		VALOR TOTAL		711 $

IX. Conclusiones

Se concluye que la interfaz gráfica implementada en C# posibilita la visualización de los puntos representativos del brazo derecho del operador y sus trayectorias, como se muestra en la Figura 38. Esa interfaz proporciona una representación visual clara y precisa de los movimientos del brazo.

Se ha desarrollado un sistema de teleoperación que utiliza el sensor Kinect V1 para capturar y transmitir las coordenadas y orientación del brazo derecho del operador a un brazo robótico (ver Figura 37). Esto permite replicar los movimientos con un tiempo de respuesta promedio de 0.9 segundos como se muestra en la Tabla III.

Se ha desarrollado un script en Python que posibilita la conexión al robot mediante la dirección IP y el puerto correspondientes, así como el procesamiento de los datos capturados por Kinect para transformarlos en movimientos del brazo robótico. Este proceso se lleva a cabo mediante el uso de las librerías de RoboDK, esto gestiona la comunicación y el procesamiento de los datos para garantizar una ejecución precisa de los movimientos del brazo robótico de acuerdo con los movimientos del operador humano.

El sistema de teleoperación fue validado exitosamente mediante pruebas de Pick and Place, ver Figura 34, logrando manipular objetos de manera satisfactoria. Estas pruebas demostraron la funcionalidad del sistema en un 90 % como se muestra en la Tabla III.

X. Recomendaciones

Se recomienda posicionar la cámara kinect en un lugar donde halla una adecuada iluminación y a una distancia recomendada de 2 metros para que realice eficientemente el seguimiento del esqueleto y capte correctamente los puntos referenciales del brazo derecho.

Si el controlador KRC4 utiliza Windows 10 IoT, es posible que el puerto 7000 esté ocupado, por lo tanto, se recomienda utilizar el puerto 7001 para evitar conflictos de conexión.

Por defecto, el C3bridge utiliza el puerto 7000 para la comunicación. En este sentido, se sugiere cambiar este puerto al 7001 si el controlador está utilizando Windows 10 IoT, asegurando así una comunicación sin problemas.

Es recomendable desactivar el firewall del computador, ya que puede interferir ocasionalmente con la comunicación entre el controlador del robot.

Para establecer una conexión adecuada, se requiere el uso de un cable de red cruzado.

Asegúrese de conectar el cable de red cruzado al puerto X66 (KLI) del controlador para una comunicación efectiva.

Es recomendable asignar una velocidad baja al robot durante la operación como medida de seguridad, hasta que el operador se familiarice con el control. Esto ayuda a reducir el riesgo de accidentes y permite una respuesta más rápida en caso de emergencia.

Referencias

[1] J. M. Sabater Navarro, «Desarrollo de una interfaz kinestésica paralela y experimentación en control de sistemas hapticos y teleoperados,» Tesis doct., Universidad Miguel Hernández, 2003, pág. 244. DOI: 10.13140/2.1.1064.2888.

[2] M. G. López, M. A. Artega, A. I. Gutiérrez y E. Nuño, «Resultados experimentales del control de un sistema de teleoperación bilateral de robots con retardos variantes en el tiempo,» *Revista Iberoamericana de Automática e Informática industrial*, vol. 19, n.º 1, págs. 96-107, 2021, ISSN: 1697-7920. DOI: 10.4995/riai.2021.14834.

[3] F. Salazar, J. Buele, J. Llugsa-Hinojosa, A. Domínguez, G. Pérez y A. Ibazeta, «Prototype system for the remote operation of a scorbot ER-4u robot in dangerous environments — Sistema prototipo para la teleoperación de un robot scorbot ER-4u en ambientes peligrosos,» *RISTI - Revista Iberica de Sistemas e Tecnologias de Informacao*, vol. 2020, n.º E29, págs. 312-324, 2020.

[4] S. Ma, T. Tang, H. You, Y. Zhao, X. Ma y J. Wang, «A Robotic Arm System for Automatic Welding of Bars Based on Image Denoising,» *2021 3rd International Conference on Robotics and Computer Vision, ICRCV 2021*, págs. 40-44, 2021. DOI: 10.1109/ICRCV52986.2021.9546976.

[5] S. Hussain, R. P. George, N. Ahmad y R. Jahan, «Machine Learning Methods of Industrial Automation System in Manufacturing and Control Sector using Joystick with and Robotic Technology,» *Proceedings of the 2022 11th International Conference on System Modeling and Advancement in Research Trends, SMART 2022*, págs. 1134-1140, 2022. DOI: 10.1109/SMART55829.2022.10047023.

[6] A. Rahul Gautam, Ankush Gedam, «Review on Development of commercial Robotic Arm,» *Interantional Journal of Scientific Research in Engineering and Management*, vol. 06, n.º 05, págs. 1752-1755, 2017.

[7] KUKA, *Robots industriales de KUKA*, https://www.kuka.com/es-es/productos-servicios/sistemas-de-robot/robot-industrial/kr-agilus, 2023.

[8] Intel, *Brazos robóticos industriales: Cambio en la forma de realizar el trabajo*, https://www.intel.la/content/www/xl/es/r arm.html, 2020.

[9] KUKA Roboter GmbH, «Programación de robots 1,» Augsburg, inf. téc., 2013, pág. 181.

[10] ABB, *Articulated robots*, 2024. dirección: https://new.abb.com/products/robotics/robots/articulated-robots/irb-6620.

[11] AMS, *Robots Cartesianos*, 2022. dirección: https://www.ams-latam.com/es/robots-cartesianos/.

[12] J. J. Craig, «ROBÓTICA,» en *ROBÓTICA*, P. M. G. Rosas, ed., Tercera ed, Naucalpan de Juárez, Edo. de México: Pearson Education, Inc, 2006, cap. capitulo 8, pág. 400, ISBN: 9702607728.

[13] FANUC, *Robots Delta Serie DR-3iB*, 2023. dirección: https://www.fanuc.eu/es/es/robots/p{\'{a}}gina-filtro-robots/delta-robots.

[14] UNIVERSAL ROBOTS, *¿QUÉ ES UN ROBOT COLABORATIVO O COBOT? TODO LO QUE DEBES SABER*, 2020. dirección: https://www.universal-robots.com/es/blog/que-es-un-robot-colaborativo-o-cobot-todo-lo-que-debes-saber/.

[15] Universal Robots, *Tipos de Robots Industriales y Sus Aplicaciones*, https://www.universal-robots.com/mx/blog/tipos-de-robots-industriales-y-sus-aplicaciones/, Consultado en [fecha de consulta], 2023.

[16] I. Košt'ál, «A.NET application searching for data in a log file of the KUKA industrial welding robot,» *Proceedings of the 16th International Conference on Mechatronics, Mechatronika 2014*, págs. 656-661, 2014. DOI: 10.1109/MECHATRONIKA.2014.7018338.

[17] B. Kang y J. Cheong, «Development of Two-Way Self-Adaptive Gripper Using Differential Gear,» *Actuators*, vol. 12, n.º 1, 2023, ISSN: 20760825. DOI: 10.3390/act12010014.

[18] FESTO, *Kits de pinzas para robots de Universal Robots*, 2023.

[19] ROBOTIQ, *Pinza robótica adaptable de 3 dedos*, 2021. dirección: https://robotiq.com/es/productos/pinza-robotica-adaptable-de-3-dedos.

[20] Wegard GmbH, «Bakery SoftGripper,» Hamburg, 2021. dirección: https://soft-gripping.com/solutions/bakery/.

[21] Revista de Robots, *Tipos de gripper y pinzas robóticas*, 2023. dirección: https://revistaderobots.com/sistemas-de-agarre/tipos-de-gripper-y-pinzas-roboticas/.

[22] T. S. Gavilanez, E. A. Gómez, E. Estevez y S. P. Thirumuruganandham, «Dynamic Recognition and Classification of Trajectories in SLRecon Adopted Artificial Intelligence in Kinect,» *Communications in Computer and Information Science*, vol. 1431 CCIS, págs. 84-96, 2021, ISSN: 18650937. DOI: 10.1007/978-3-030-86702-7_8.

[23] N. M. DiFilippo y M. K. Jouaneh, «Characterization of Different Microsoft Kinect Sensor Models,» *IEEE Sensors Journal*, vol. 15, n.º 8, págs. 4554-4564, 2015, ISSN: 1530437X. DOI: 10.1109/JSEN.2015.2422611.

[24] M. W. Rahman y M. L. Gavrilova, «Kinect gait skeletal joint feature-based person identification,» *Proceedings of 2017 IEEE 16th International Conference on Cognitive Informatics and Cognitive Computing, ICCI*CC 2017*, págs. 423-430, 2017. DOI: 10.1109/ICCI-CC.2017.8109783.

[25] Microsoft, *Kinect para Xbox 360*, https://web.archive.org/web/20101121114133/http://www.xbox.com/es-ES/Xbox360/ 2010.

[26] C. Núñez, C. Dennis, A. Larrea y S. Acosta, «Comparación de la tecnología Kinect v1, v2 y Azure centrada en las habilidades motoras de los niños,» *Revista Ibérica de Sistemas e Tecnologias de Informação*, págs. 638-350, 2021.

[27] K. Furuhata, K. Kutsuzawa, D. Owaki y M. Hayashibe, «Systematic Motion Integration with Multiple Depth Cameras Allowing Sensor Movement for Stable Skeleton Tracking,» *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, vol. 2022-July, págs. 1801-1804, 2022, ISSN: 1557170X. DOI: 10.1109/EMBC48229.2022.9870876.

[28] S. Motiian, P. Pergami, K. Guffey, C. A. Mancinelli y G. Doretto, «Automated extraction and validation of children's gait parameters with the Kinect,» *BioMedical Engineering Online*, vol. 14, n.º 1, págs. 1-36, 2015, ISSN: 1475925X. DOI: 10.1186/s12938-015-0102-9.

[29] J. Cui, S. Tosunoglu, R. Roberts, C. Moore y D. W. Repperger, «A review of teleoperation system control,» 2003.

[30] «Teleoperation of Collaborative Robot for Remote Dementia Care in Home Environments,» *IEEE Journal of Translational Engineering in Health and Medicine*, vol. 8, n.º June, 2020. DOI: 10.1109/JTEHM.2020.3002384.

[31] Y. Nagatsu y H. Hashimoto, «Force-based Two-channel Bilateral Control for Position/Velocity Controlled Robots,» *2022 IEEE 17th International Conference on Advanced Motion Control (AMC)*, págs. 181-186, 2022. DOI: 10.1109/amc51637.2022.9729256.

[32] W. Bai, Z. Wang, Q. Cao et al., «Anthropomorphic Dual-Arm Coordinated Control for a Single-Port Surgical Robot Based on Dual-Step Optimization,» *IEEE Transactions on Medical Robotics and Bionics*, vol. 4, n.º 1, págs. 72-84, 2022, ISSN: 25763202. DOI: 10.1109/TMRB.2022.3145673.

[33] M. Shahbazi, S. F. Atashzar y R. V. Patel, «A Systematic Review of Multilateral Teleoperation Systems,» *IEEE Transactions on Haptics*, vol. 11, 2018, ISSN: 23294051. DOI: 10.1109/TOH.2018.2818134.

[34] J. Luo, W. He y C. Yang, «Combined perception , control , and learning for teleoperation : key technologies , applications , and challenges,» vol. 2, págs. 33-43, 2020. DOI: 10.1049/ccs.2020.0005.

[35] C. Cruz, J. Jorge, J. del Cerro y A. Barrientos, «Sistema de realidad virtual para teleoperacion de robots tipo centauro,» *XLIV Jornadas de Automática*, págs. 563-568, 2023. DOI: https://doi.org/10.17979/spudc.9788497498609.563.

[36] O. Tokatli, P. Das, R. Nath et al., «Robot-assisted glovebox teleoperation for nuclear industry,» *Robotics*, vol. 10, n.º 3, 2021, ISSN: 22186581. DOI: 10.3390/robotics10030085.

[37] H. Su, C. Yang, G. Ferrigno y E. De Momi, «Improved human-robot collaborative control of redundant robot for teleoperated minimally invasive surgery,» *IEEE Robotics and Automation Letters*, vol. 4, n.º 2, págs. 1447-1453, 2019, ISSN: 23773766. DOI: 10.1109/LRA.2019.2897145.

[38] I. Salihović, A. Škamo y D. Jokić, «RoboDK to MATLAB Joint Position Transformation,» en *2021 Selected Issues of Electrical Engineering and Electronics (WZEE)*, 2021, págs. 1-6. DOI: 10.1109/WZEE54157.2021.9576924.

[39] T. Al-Geddawy, «A Digital Twin Creation Method for an Opensource Low-cost Changeable Learning Factory,» *Procedia Manufacturing*, vol. 51, págs. 1799-1805, 2020, 30th International Conference on Flexible

Automation and Intelligent Manufacturing (FAIM2021), ISSN: 2351-9789. DOI: 10.1016/j.promfg.2020.10.250. dirección: https://www.sciencedirect.com/science/article/pii/S2351978920321284.

[40] R. Inc, *C# API*, 2024. dirección: https://robodk.com/doc/en/RoboDK-API-API.html.

ANEXO A
PLANO

Diseño de base para gripper flexible:

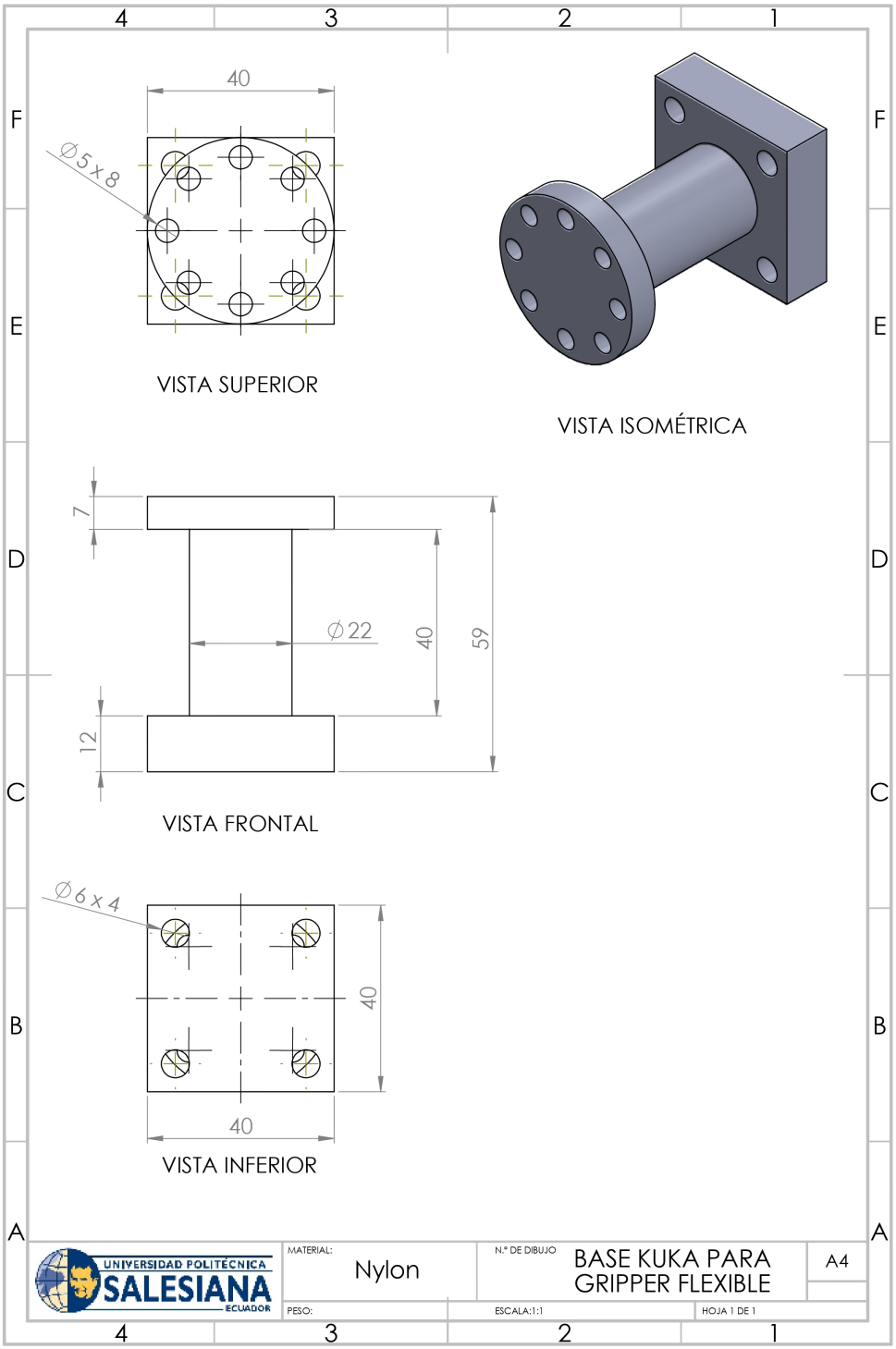


Figura 40. Base del robot KUKA para gripper flexible, por D. Córdova, Solidworks

ANEXO B
FABRICACIÓN

Fabricación de base para gripper flexible:

Figura 41. Proceso de torneado en torno CNC, por D. Córdova.

Figura 42. Programación a pie de máquina en centro de mecanizado CNC, por D. Córdova.

Figura 43. Proceso de planeado en centro de mecanizado CNC, por D. Córdova.

Figura 44. Corte del exceso de nylon, por D. Córdova.

Figura 45. Base del robot KUKA mecanizada, por D. Córdova.

Figura 46. Montaje de gripper con la base fabricada, por D. Córdova.

ANEXO C
CONFIGURACIÓN DE CONEXIÓN ENTRE KUKA KR10 R1100-2 Y EL COMPUTADOR, POR D. CÓRDOVA.

En este documento, se presenta una guía detallada desarrollada específicamente para la configuración del brazo robótico KUKA KR10, con el propósito de establecer una conexión en tiempo real entre el robot y RoboDK. Esta configuración es esencial para la implementación de un sistema de teleoperación, el cual está diseñado para replicar los movimientos del brazo de un operador en el robot.

Se deberá tener ciertas consideraciones que ayudarán para cumplir con exito la configuración del robot, una vez tomadas en cuenta, se deberá establecer una conexión mediante cable de red cruzado con el controlador. Se realizan las respectivas configuraciones e instalaciones tanto en el controlador del robot como en el computador y asi poder realizar una comunicación efectiva. Se realiza la respectiva instalación y configuración de parámetros. El resultado final da un sistema de teleoperación el cual da lugar a poder controlar un robot industrial con tan solo los movimientos del brazo del operador.

El resultado de este procedimiento es la implementación exitosa de un sistema de teleoperación avanzado. Este posibilita el control de un robot industrial mediante la replicación de los movimientos del brazo del operador. La integración efectiva de la tecnología de control con la interfaz de usuario da lugar a una herramienta eficiente y versátil, que amplía significativamente las habilidades humanas dentro de la robótica industrial.

C-1. Consideraciones:

- Si el controlador del KRC4 utiliza Windows 10 IoT, es probable que el puerto 7000 esté ocupado, por lo tanto, se recomienda utilizar el puerto 7001 para evitar conflictos de conexión.
- El C3bridge, de forma predeterminada, utiliza el puerto 7000 para la comunicación. Por lo tanto, se sugiere cambiar este puerto al 7001 si el controlador está utilizando Windows 10 IoT, para garantizar una comunicación sin problemas.
- Se recomienda desactivar el firewall, ya que puede interferir ocasionalmente con la comunicación entre dispositivos.
- Para establecer una conexión adecuada, se requiere el uso de un cable de red cruzado.
- Asegúrese de conectar al puerto X66 (KLI) del controlador para una comunicación efectiva.

C-2. Conexión de hardware:

1. Conectar uno de los extremos del cable de red cruzado al puerto X66 (KLI) del controlador KRC4.

2. El otro extremo conectar al puerto de red del computador.

C-3. Configuración de la comunicación en el controlador KRC4: En esta sección realizaremos la configuración de red donde estableceremos la dirección IP y otros parametros en la interfaz “virtual5 (KLI)” que es la interfaz de red para la comunicación virtual del controlador con el computador. Dentro de la misma se tienen dos configuraciones de las tareas de comunicación del controlador del robot, estas son:

- **Tiempo real tarea de recepción:** En esta configuración se escogue el protocolo de comunicación por el cual se realizará la recepción de los datos en tiempo real.
- **Tarea de recepción:** Es la configuración que determina cómo el robot maneja los datos que recibe.

A continuación, se detallarán los procedimientos requeridos para realizar la configuración correcta de la red.

1. Iniciar sesión como administrador.

2. Se accede a la ruta KUKA → Puesta en servicio → Configuración de red.

3. Se selecciona la opción “Ampliación...”.

4. Se elige la interfaz “Virtual5 (KLI)” y se agregan los siguientes parámetros:

   - **Tipo de dirección:** Dirección IP fija
   - **Dirección IP:** 172.31.1.147 (o cualquier dirección pero que este bajo la misma red)
   - **Máscara de subred:** 255.255.0.0
   - **Standard gateway:** 172.31.1.147 (la misma de la dirección IP)
   - **Servidor DNS:** 0.0.0.0
   - **Interfaz de Windows:** Si

   Las configuraciones del “virtual5 (KLI)” mencionadas se visualizan en la Figura 47, las cuales se muestran en la interfaz del SmartPad con detalle.

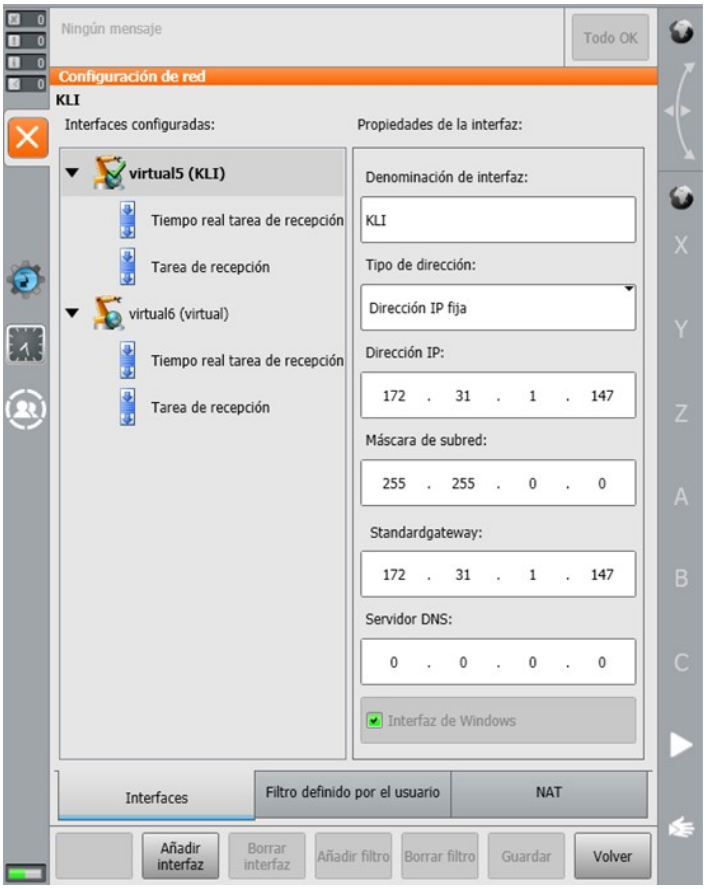


Figura 47. Configuración de la interfaz “Virtual5 (KLI)”, por D. Córdova.

5. Se debe hacer clic en “Agregar interfaz”, ubicado en la parte inferior. Esto generará una nueva interfaz identificada como “Virtual6 (virtual)”.

6. A continuación, se selecciona la interfaz “Virtual6 (virtual)” y se procede a ingresar los siguientes parámetros:

- **Tipo de dirección:** Dirección IP mixta
- **Dirección IP:** 192.168.2.147 (o cualquier dirección pero que este bajo la misma red)
- **Máscara de subred:** 255.255.255.0
- **Standard gateway:** 172.31.1.147 (la misma de la dirección IP)
- **Servidor DNS:** 0.0.0.0
- **Interfaz de Windows:** No

La representación de las configuraciones para “Virtual6 (virtual)” se muestra en la Figura 48, donde se ofrece una visión clara de la interfaz en el SmartPad.

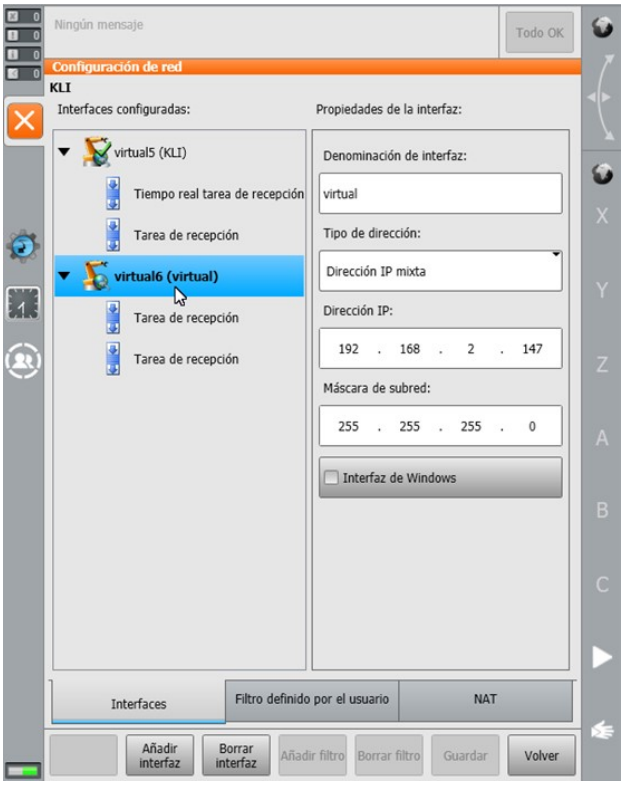


Figura 48. Configuración de la interfaz “Virtual6 (virtual)”, por D. Córdova.

7. Se accede a “Virtual5 (KLI)” y se selecciona “Tiempo real tarea de recepción”. Posteriormente, se elige la configuración siguiente, la cual se puede visualizar en la Figura 49.

   - **Filtro de recepción:** UDP
   - **Filtro de recepción:** Aceptar todo
   - **Filtro de recepción:** EtherNet / Paquetes IP

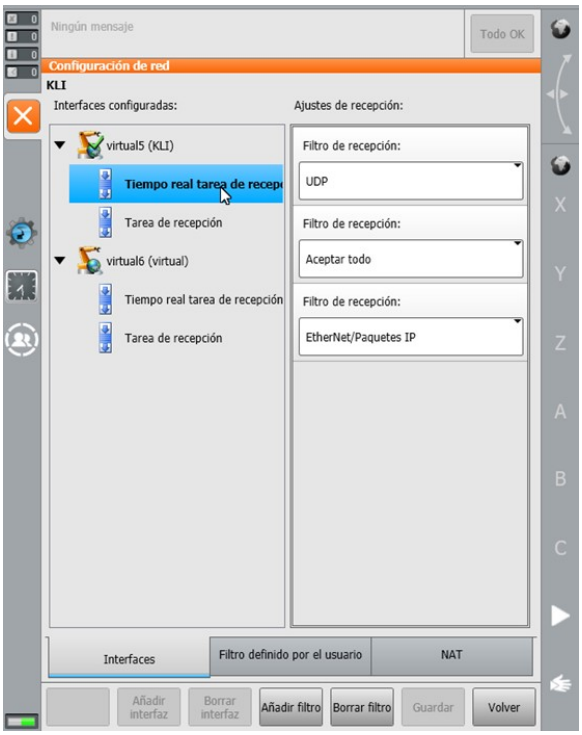


Figura 49. Configuración del "Tiempo real tarea de recepción", por D. Córdova.

8. Se realiza un clic en "Tarea de recepción" y luego se elige "Subred de destino" dentro del "Filtro de recepción".

9. La configuración se guarda y, acto seguido, se procede a salir de "Configuración de red". La secuencia completa de estas acciones está representada en la Figura 50.

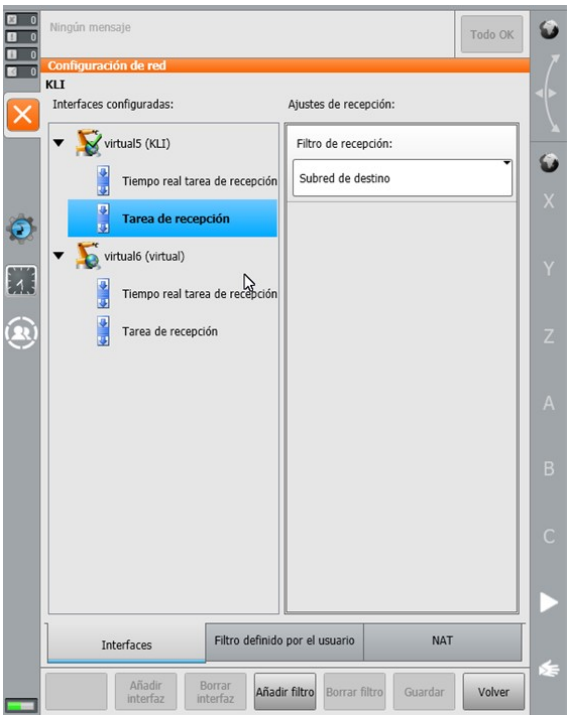


Figura 50. Configuración de "tarea de recepción", por D. Córdova.

10. Se accede a “Virtual6 (virtual)” y se hace clic en la primera “Tarea de recepción”. Posteriormente, se selecciona en “Filtro de recepción” la opción “Subred de destino”. Este proceso se muestra en la Figura 51.

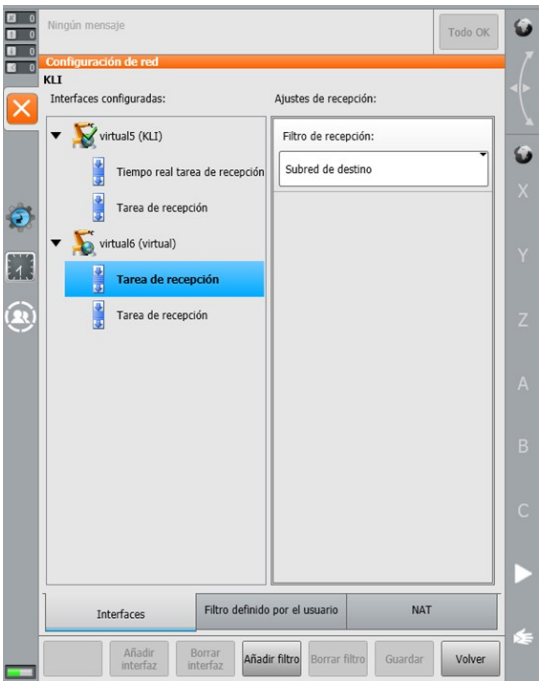


Figura 51. Configuración de “tarea de recepción” de “Virtual6 (Virtual)”, por D. Córdova.

11. A continuación, se realiza un clic en la segunda “Tarea de recepción” y se selecciona la configuración “Subred de destino” en el “Filtro de recepción”, como se muestra en la Figura 52.

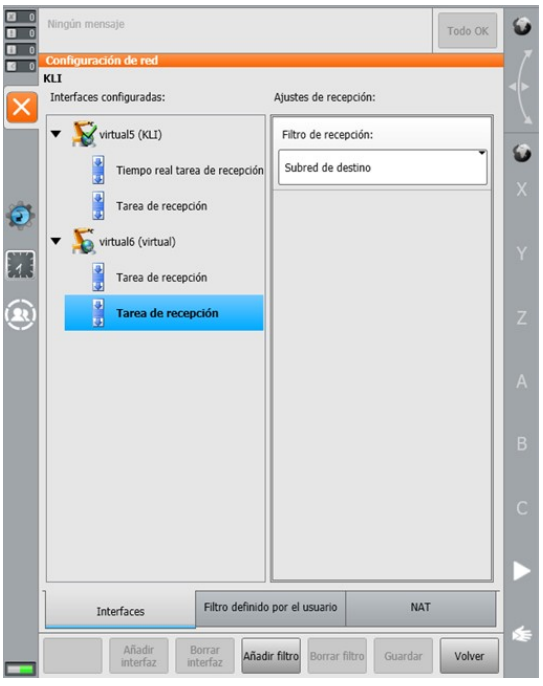


Figura 52. Configuración de la segunda tarea de recepción de Virtual6 (Virtual), por D. Córdova.

C-4. Añadir el puerto 7001:

1. Para configurar el puerto 7001, se accede a Inicio → Puesta en servicio → Configuración de red.

2. Se elige la opción “Ampliación...”.

3. Se selecciona la opción “NAT” ubicada en la parte inferior derecha.

4. Se introduce el valor “7001” en el campo “Número de puerto” y se marcan los protocolos permitidos “TCP/UDP”.

5. Finalmente, se hace clic en “Guardar” para confirmar la configuración. Este procedimiento se ilustra en la imagen 53.

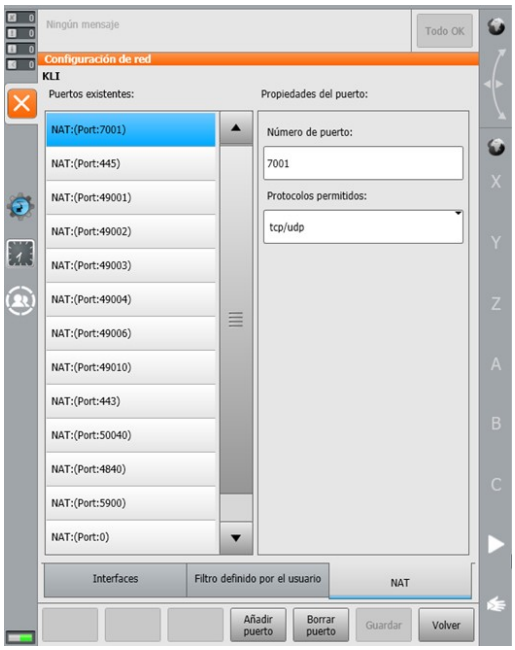


Figura 53. Configuración y creación del puerto 7001, por D. Córdova.

6. Para llevar a cabo un reinicio en frío y permitir que el controlador lea los nuevos cambios realizados: Inicio → Apagar → reiniciar la PC de control.

C-5. Instalación de C3Brige: Para instalar C3bridge en el controlador del brazo robot KUKA, se deben seguir los siguientes pasos:

1. Se descarga el archivo ZIP necesario desde el siguiente enlace: https://robodk.com/files/contents/ en el computador.

2. Se utiliza un Pendrive para transferir la información desde el computador al controlador KRC4.

3. Al abrir el enlace, se navega hacia la sección Robot Drivers → KUKA.

4. Se descomprime el archivo ZIP en un Pendrive.

5. Se accede primero a la interfaz de Windows proporcionada por KUKA.

6. Se inicia sesión como administrador.

7. Se va a Puesta en servicio → Servicio → Minimizar HMI, tal como se muestra en la Figura 54.

Figura 54. Acceso a la interfaz de Windows, por D. Córdova.

8. Se debe insertar el Pendrive y acceder a la carpeta “KUKA-RoboDK-Driver” → “C3 Bridge Server 1.6.1” para copiar y pegar el instalador “c3setup-1.6.1” en el escritorio del controlador del robot. Luego, se procede a dar doble clic en el instalador que ya ha sido pegado en el escritorio para iniciar la instalación.

9. Una vez finalizada la instalación, se encontrará en el escritorio una aplicación llamada “C3 Bridge Server”, tal y como se muestra en la Figura 55.

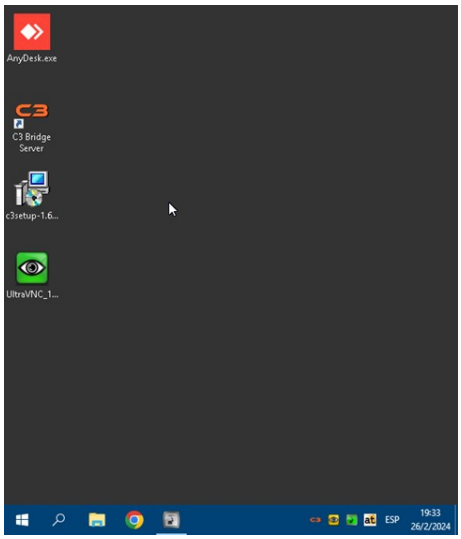


Figura 55. Interfaz de Windows del KRC4, por D. Córdova.

10. Es necesario modificar el puerto predeterminado del C3 Bridge Server; este ajuste debe realizarse antes de ejecutar el programa.

11. Se debe dirigir al buscador de Windows y escribir “Editor de registro” (Registry Editor).

12. Luego, se debe hacer clic para abrir el Editor de registro.

13. A continuación, se accede a la siguiente carpeta: HKEY_CURRENT_USER → Software → C3 Bridge Interface. Esta ruta se visualiza en la Figura 56.

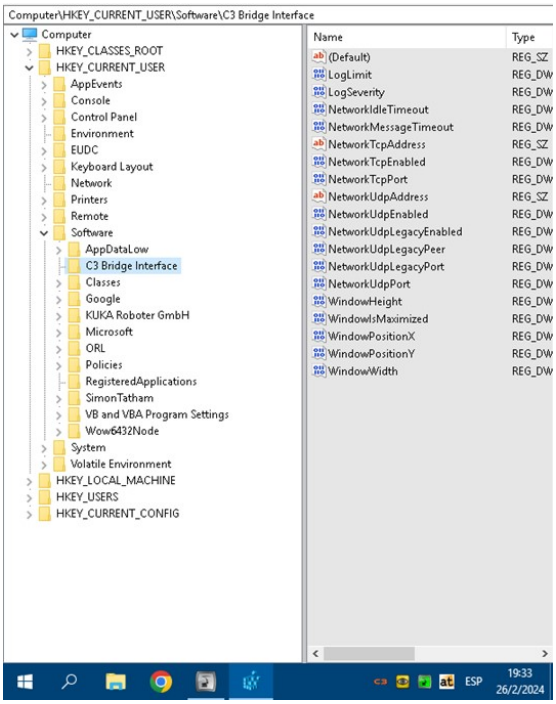


Figura 56. Acceso al “Registy Editor”, por D. Córdova.

14. Se debe hacer doble clic en los registros resaltados mostrados en la Figura 57, y cambiar el valor del puerto a 7001. Luego, se procede a cerrar la ventana.

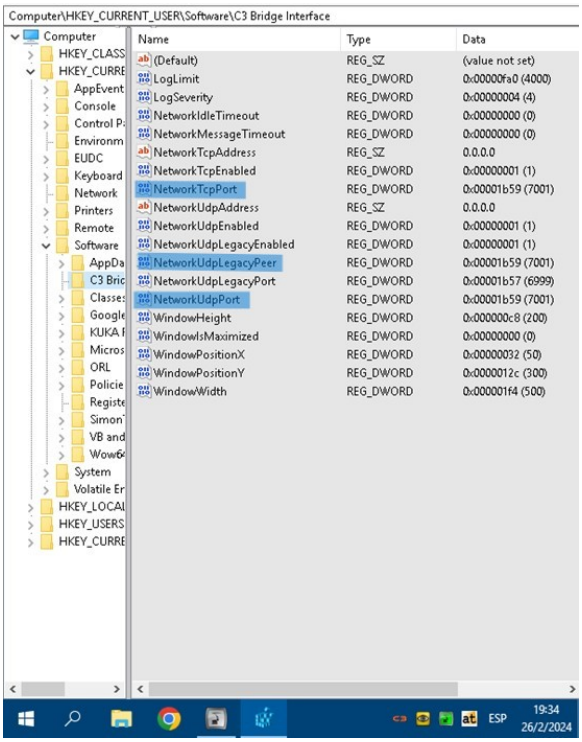


Figura 57. Asignación del puerto 7001, por D. Córdova.

15. Para verificar que el cambio se realizó correctamente, se debe hacer doble clic en “C3 Bridge Server”. Se observará que en la parte inferior izquierda de la barra de tareas aparecerá el logo de “C3 Bridge”, lo que indica que está en ejecución. Luego, se debe hacer clic derecho en el logo y seleccionar “C3 Bridge Interface Server”. El resultado de este procedimiento se puede apreciar en la Figura 58.

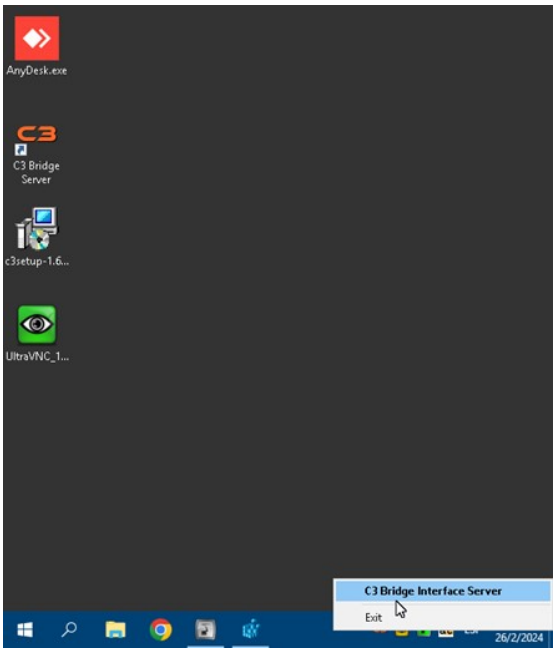


Figura 58. Ejecución del “C3 Bridge Server”, por D. Córdova.

16. Ahora se debería ver un mensaje similar al que se muestra en la Figura 59, pero con la diferencia de que en la imagen mostrada existe una conxión establecida con RoboDK.

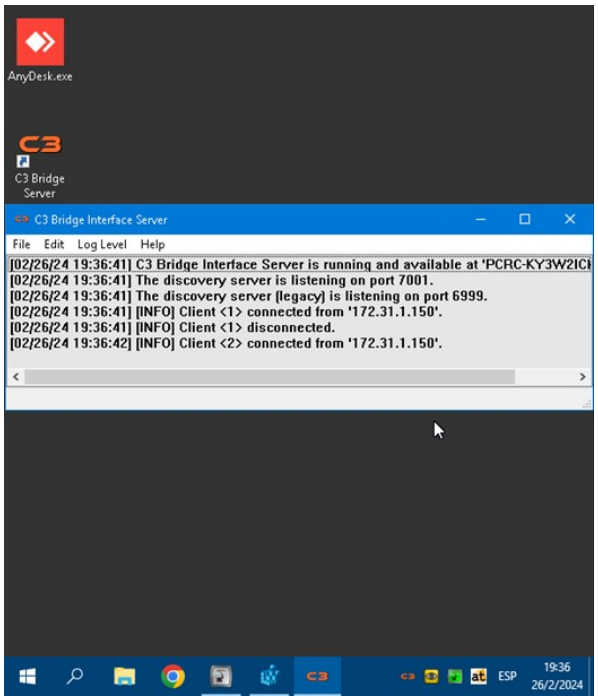


Figura 59. Interfaz del “C3 Bridge Server”, por D. Córdova.

17. Se recomienda iniciar el “C3 Bridge Server” cada vez que el robot se encienda nuevamente.

18. A continuación, se procede a acceder al pendrive para realizar la transferencia del archivo SRC “RoboDKsync562” a la carpeta “R1” del robot.

19. Se accede a KUKA → Pendrive (Nombre del pendrive) → KUKA-RoboDK-Driver → RoboDKsync562. Para llevar a cabo la operación de copiar, se selecciona “Editar” y posteriormente “Copiar”.

20. Luego, se selecciona la carpeta “R1” del robot, seleccionando “Editar” y luego “Pegar”, completando así el proceso de transferencia del archivo, que se puede visualizar en la Figura 60.

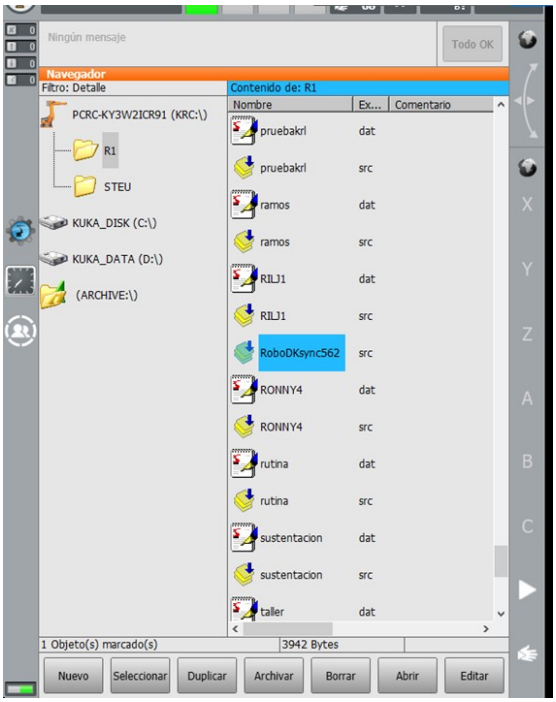


Figura 60. Interfaz de selección de archivos del robot, por D. Córdova.

21. Para agregar las variables necesarias, se accede nuevamente al pendrive y se abre el archivo "$Config" ubicado en la carpeta "KUKA-RoboDK-Driver". Luego, se copia su contenido.

22. Seguidamente, se dirige a la carpeta "STEU" y se abre el archivo "$Config" que se encuentra dentro de ella como se muestra en la Figura 61. Después, se pegan las variables que se han copiado anteriormente.

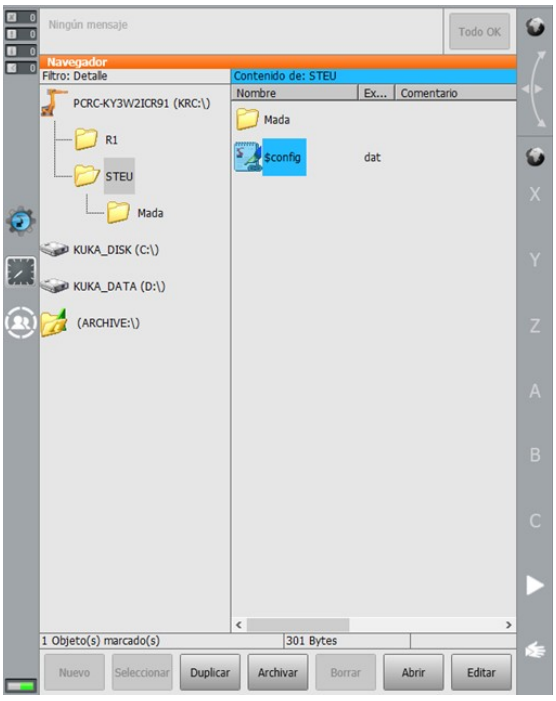


Figura 61. Ruta para selección de archivo "$Config", por D. Córdova.

C-A. Configuración de la comunicación en el computador

1. Se accede al Panel de Control → Centro de redes y recursos compartidos → Cambiar configuración del adaptador.

2. Se hace clic derecho en el adaptador de red que se utilizará (Ethernet) y se selecciona Propiedades.

3. Se da doble clic en "Protocolo de Internet versión 4 (TCP/IPv4)" para abrir una ventana emergente.

4. Se selecciona "Usar la siguiente dirección IP" y se configuran los siguientes parámetros:

   - **Dirección IP:** 172.31.1.150 (o cualquier dirección que este bajo la misma red)
   - **Máscara de subred:** 255.255.0.0
   - **Standard gateway:** 172.31.1.147 (la misma que tiene el robot)

5. Se debe hacer clic en "Opciones avanzadas". Se abrirá una ventana emergente donde se seleccionará "Agregar" en la sección de "Direcciones IP" para añadir una nueva dirección IP con los siguientes parámetros:

   - **Dirección IP:** 192.168.2.150 (o cualquier dirección pero que este bajo la misma red del Virtual6 (virtual))
   - **Máscara de subred:** 255.255.255.0

6. Se guardan los cambios y listo.

C-A1. Verificación de la comunicación:

1. Abrir el símbolo del sistema (cmd) de Windows en el computador.

2. Para verificar la conexión con el puerto X66 (KLI), se deben introducir los comandos correspondientes y presionar la tecla Enter.
   - ping 172.31.1.147

3. Se comprueba la conexión a RSI:

   - ping 192.168.2.147

4. Si, al ejecutar el comando ping hacia la dirección 172.31.1.147, se obtiene como resultado "Respuesta de 172.31.1.147: bytes=32 tiempo=¡1ms TTL=64", entonces se ha configurado exitosamente una conexión de red con el robot KUKA. En caso contrario, se debe verificar que todos los pasos de la guía se hayan configurado correctamente.

C-B. Instalación y configuración de RoboDK

Se debe acceder al buscador y escribir "RoboDK". Luego, seleccionar la primera opción que aparece y hacer clic en "descargar", o acceder directamente mediante el enlace: https://robodk.com/download. Se descargará la versión adecuada según el sistema operativo del ordenador y se ejecutará el archivo descargado para iniciar la instalación. Se seguirán las instrucciones proporcionadas para completar el proceso de instalación.

1. Al abrir RoboDK, se selecciona en la parte superior izquierda el icono del mundo, denominado "Open robot library", lo cual conduce a una extensa biblioteca de robots de diferentes marcas.

2. Dentro del buscador de la librería, se escribe “KUKA KR 10 R1100-2” y se hace clic en “Open”. Posteriormente, se regresa a RoboDK para visualizar el brazo robot seleccionado.

3. Se realiza un clic derecho en el robot y se selecciona “Conectar al robot...”. Esto abrirá un panel en la parte izquierda, donde se deberán configurar los parámetros correspondientes.

   - **Robot IP/COM:** 172.31.1.147
   - **Puerto de robot:** 7001

4. En “More options”, dentro del apartado “Conductor”, se selecciona la opción de “apikuka” y se hace clic en OK.

5. Se hace clic en “Ping”. Si aparece una ventana emergente con el mensaje “EXITOSO”, se hace clic en OK.

6. Para conectar el robot a la PC, se hace clic en “Conectar”. En la barra de estado, el mensaje “Ready” deberá aparecer en verde.

7. Se hace clic derecho en el programa previamente creado y se selecciona “Ejecutar en el robot”.

8. Para ejecutar la simulación en tiempo real, se hace clic derecho en el programa y se selecciona “Ejecutar”. El estado de conexión cambiará a “Working” y el robot se moverá hasta que el programa finalice su ejecución.

9. Se hace clic derecho en el programa y se selecciona “Bucle” para ejecutarlo repetidamente.

10. Si se desea detener el movimiento del robot, simplemente se desmarca la opción “Ejecutar” en el programa.

C-C. Puesta en Marcha del Sistema de Teleoperación en Tiempo Real del brazo robótico usando Kinect

1. Conectar el cable de red cruzado al puerto X66 (KLI) del controlador.

2. Al encender el robot, es imperativo asegurar el correcto inicio del “C3 Bridge Server” para establecer la comunicación.

3. Ejecutar el programa SRC “RoboDKsync562” en modo “AUT” mediante la interfaz SmartPad del controlador.

4. Iniciar Microsoft Visual Studio y ejecutar el programa diseñado para el seguimiento del esqueleto (skeleton tracking) con el sensor Kinect, lo cual permite generar un archivo en formato “.JSON” con las coordenadas y la orientación.

5. Abrir la estación de RoboDK que contiene el modelo virtual del robot KUKA y el script de Python responsable del procesamiento de los datos. Este último realiza también la configuración necesaria para la conexión en tiempo real con el robot físico y su control.

6. Activar la comunicación entre el modelo virtual y el robot físico mediante la ejecución del script de Python.

7. Posicionarse frente a la cámara Kinect facilita el seguimiento de los movimientos por parte del sistema.

Una vez completados estos pasos, se puede apreciar cómo el robot KUKA KR10 realiza un seguimiento de los movimientos del brazo, tanto en su representación física como en el entorno virtual de RoboDK. Esta capacidad única del sistema permite una interacción fluida entre el operador y el robot, facilitando la ejecución de tareas

de manera efectiva. Además, esta integración entre el mundo físico y virtual proporciona una experiencia de teleoperación altamente intuitiva y eficiente, permitiendo al operador controlar el robot de manera remota con una facilidad sin precedentes.