

Instituto Tecnológico de Costa Rica

Escuela de Ingeniería Mecatrónica



Integración de la mano robótica SoftHand con el robot UR5 y su Implementación en el Robot Operating System (ROS)

Informe de Proyecto de Graduación para optar por el título de
Ingeniero en Mecatrónica con el grado académico de Licenciatura

Fabián Álvarez Juárez

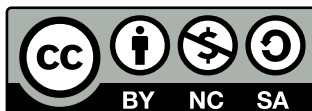
Cartago, 4 de febrero de 2025

*Integración de la mano robótica SoftHand con el robot UR5 y su
implementación en el sistema de operación de robots (ROS)*

by **Fabián Álvarez Juárez**

This document is licensed under the following license:

**Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International
(CC BY-NC-SA 4.0).**



Declaro que el presente Proyecto de Graduación ha sido realizado enteramente por mi persona, utilizando y aplicando literatura referente al tema e introduciendo conocimientos propios.

En los casos en que he utilizado bibliografía he procedido a indicar las fuentes mediante las respectivas citas bibliográficas. En consecuencia, asumo la responsabilidad total por el trabajo de graduación realizado y por el contenido del correspondiente informe final.

A handwritten signature in black ink, appearing to read 'Fabián Álvarez Juárez', with a stylized flourish at the end.

Fabián Álvarez Juárez

Cartago, 4 de febrero de 2025

Céd: 1-1785-0939

INSTITUTO TECNOLÓGICO DE COSTA RICA
PROGRAMA DE LICENCIATURA EN INGENIERÍA MECATRÓNICA
PROYECTO FINAL DE GRADUACIÓN
ACTA DE APROBACIÓN

El profesor asesor del presente trabajo final de graduación, indica que el documento presentado por el estudiante cumple con las normas establecidas por el programa de Licenciatura en Ingeniería Mecatrónica del Instituto Tecnológico de Costa Rica para ser defendido ante el jurado evaluador, como requisito final para aprobar el curso Proyecto Final de Graduación y optar así por el título de Ingeniero(a) en Mecatrónica, con el grado académico de Licenciatura.

Estudiante: Fabián Andrés Álvarez Juárez

Proyecto: Integración de la mano robótica SoftHand con el robot UR5 y su Implementación en el Robot Operating System (ROS)

JUAN LUIS CRESPO MARIÑO (FIRMA)
PERSONA FÍSICA, CPF-08-0113-0166.
Fecha declarada: 04/02/2025 09:02:55 AM
Esta es una representación gráfica únicamente,
verifique la validez de la firma.

Dr. -Ing. Juan Luis Crespo Mariño

Asesor

Lugar y fecha de la presentación

Cartago, 4 de febrero del 2025

INSTITUTO TECNOLÓGICO DE COSTA RICA
PROGRAMA DE LICENCIATURA EN INGENIERÍA MECATRÓNICA
PROYECTO FINAL DE GRADUACIÓN
ACTA DE APROBACIÓN

Proyecto final de graduación defendido ante el presente jurado evaluador como requisito para optar por el título de Ingeniero(a) en Mecatrónica con el grado académico de Licenciatura, según lo establecido por el programa de Licenciatura en Ingeniería Mecatrónica, del Instituto Tecnológico de Costa Rica.

Estudiante: Fabián Andrés Álvarez Juárez

Proyecto: Integración de la mano robótica SoftHand con el robot UR5 y su Implementación en el Robot Operating System (ROS)

Miembros del jurado evaluador

**JOHANNA
VANESSA MUÑOZ
PEREZ (FIRMA)**

Firmado digitalmente
por JOHANNA VANESSA
MUÑOZ PEREZ (FIRMA)
Fecha: 2025.02.04
08:09:24 -06'00'

MSc. -Ing. Johanna Vanessa Muñoz Pérez

Jurado

**IVAN ARAYA
MENESES
(FIRMA)**

Firmado digitalmente
por IVAN ARAYA
MENESES (FIRMA)
Fecha: 2025.02.04
08:18:38 -06'00'

MSc. -Ing. Iván Araya Meneses

Jurado

Los miembros de este jurado dan fe de que el presente proyecto final de graduación ha sido aprobado y cumple con las normas establecidas por el programa de Licenciatura en Ingeniería Mecatrónica.

Lugar y fecha de la presentación

Cartago, 4 de febrero del 2025

Resumen

Este proyecto presenta el diseño, implementación y validación de un sistema robótico integrado compuesto por el manipulador UR5e y la mano robótica qbHand2M, operando bajo ROS 2. Desarrollado en el Grupo Integrado de Ingeniería (GII), su objetivo es optimizar la manipulación en entornos controlados mediante algoritmos avanzados de planificación y control en tiempo real.

La integración del sistema incluyó la sincronización de los actuadores, la implementación de controladores dedicados y la optimización del espacio de trabajo. Se utilizó MoveIt 2 para la planificación y ejecución de trayectorias, garantizando estabilidad y precisión en los movimientos. Las pruebas experimentales demostraron una integración efectiva, con una latencia de comunicación óptima y una precisión de posicionamiento menor a 0.2 cm. Además, el sistema mantuvo estabilidad operativa prolongada y una alta tasa de éxito en tareas de manipulación.

Los resultados obtenidos validan la escalabilidad y robustez del sistema, proporcionando una solución eficiente para aplicaciones en robótica colaborativa.

Palabras clave: UR5e, SoftHand 2, ROS 2, planificación de trayectorias, robótica colaborativa.

Abstract

This project presents the design, implementation, and validation of an integrated robotic system composed of the UR5e manipulator and the qbHand2M robotic hand, operating under ROS 2. Developed at the Grupo Integrado de Ingeniería (GII), its objective is to optimize manipulation in controlled environments through advanced planning and real-time control algorithms.

The system integration involved actuator synchronization, dedicated controller implementation, and workspace optimization. MoveIt 2 was used for trajectory planning and execution, ensuring movement stability and precision. Experimental tests confirmed effective integration, with optimal communication latency and a positioning accuracy of less than 0.2 cm. Additionally, the system demonstrated prolonged operational stability and a high success rate in object manipulation tasks.

The results validate the system's scalability and robustness, providing an efficient solution for collaborative robotics applications.

Keywords: UR5e, SoftHand 2, ROS 2, trajectory planning, collaborative robotics.

*Dedicado con amor a mis padres, Lisbeth y Ehitel,
a mi hermano Randall y su esposa Nancy,
quienes han sido mi mayor apoyo e inspiración
a lo largo de este camino.*

Agradecimientos

La culminación de este trabajo no habría sido posible sin el apoyo y la guía de muchas personas e instituciones, a quienes deseo expresar mi más sincero agradecimiento:

A mis padres, Lisbeth y Ehitel, a mi hermano Randall y su esposa Nancy, por su amor, apoyo constante y por ser mi mayor fuente de inspiración.

A mi tutor de tesis, PhD Juan Luis Crespo, por su paciencia y orientación invaluable a lo largo de este proyecto.

A la Universidade da Coruña y al Laboratorio GII, por los recursos y el entorno necesarios para llevar a cabo esta investigación.

A mis colegas y amigos del laboratorio: PhD Alejandro Romero (Alex), Emmanuel, Sergio, PhD Francisco Bellas y PhD José Antonio Becerra, por su apoyo, motivación y valiosas contribuciones.

A mis amigos que hicieron de esta experiencia algo inolvidable: A mis amigos en Ferrol, Miguel, André, Kaja, Nubia y Adriana, por su compañía, apoyo y los momentos compartidos. A mi gran amiga Samy, por su amistad incondicional y su constante ánimo a lo largo de este proceso.

Finalmente, a todas las personas que, de una u otra forma, hicieron posible este trabajo.

Fabián Álvarez Juárez
Cartago, 4 de febrero de 2025

Índice general

Índice de figuras	v
Índice de tablas	vi
Lista de símbolos y abreviaciones	vii
1 Introducción	1
1.1 Contexto	1
1.2 Antecedentes del Proyecto	2
1.3 Descripción del Problema	2
1.4 Justificación del Problema	2
1.5 Síntesis del Problema	3
1.6 Objetivos del Proyecto	3
1.6.1 Objetivo General	3
1.6.2 Objetivos Específicos	3
1.7 Estructura del Documento	3
2 Marco Teórico	4
2.1 Fundamentos de robótica	5
2.1.1 ¿Qué es un robot manipulador?	5
2.1.2 Cinemática directa	6
2.1.3 Cinemática inversa	6
2.1.4 Espacio de trabajo y configuraciones	7
2.1.5 Modelado de robots	8
2.1.6 Sensores y actuadores	9
2.1.7 Control de robots	10
2.1.8 Dinámica y singularidades	11
2.1.9 Planificación de movimientos	11
2.1.10 Interacción humano-robot	12
2.1.11 Avances tecnológicos en robótica	13
2.2 Robótica suave (Soft robotics)	14
2.2.1 Ventajas de la robótica suave	14
2.2.2 Manos robóticas suaves	15
2.2.3 Aplicaciones de la robótica suave	16
2.2.4 Desafíos y futuras tendencias	17

2.3	Robot Operating System (ROS)	18
2.3.1	Historia y evolución	18
2.3.2	Características Principales	18
2.3.3	ROS 2 y su importancia	19
2.3.4	Funcionamiento de Tópicos, Servicios y Acciones	19
2.3.5	Herramientas de ROS 2 Relevantes para el Proyecto	19
2.3.6	Aplicaciones de ROS 2 en el Proyecto	20
2.3.7	Ventajas y Desafíos	20
2.4	Normativas Europeas en Robótica Colaborativa	20
2.4.1	Normativas aplicables	20
2.4.2	Cumplimiento del UR5e con Normativas Europeas	21
2.4.3	Implicaciones en el Diseño del Proyecto	21
2.4.4	Uso de las normativas en el proyecto	22
2.4.5	Relevancia para futuras aplicaciones	22
3	Metodología	23
3.1	Descripción General	23
3.2	Identificación de necesidades	25
3.2.1	Priorización de necesidades	26
3.3	Generación de métricas	27
3.3.1	Métricas propuestas	27
3.3.2	Justificación de las métricas	28
3.4	Generación de Conceptos	29
3.4.1	Descomposición de Subproblemas	29
3.4.2	Técnicas de Generación de Conceptos	32
3.4.3	Conceptos Generados	32
3.4.4	Generación de soluciones para los subproblemas	34
3.4.5	Criterios de evaluación	35
3.4.6	Evaluación de soluciones	36
3.4.7	Selección de soluciones	36
3.5	Enfoque de solución	37
4	Propuesta de Diseño	39
4.1	Arquitectura General del Sistema	40
4.2	Integración Mecánica	41
4.2.1	Montaje del Manipulador UR5e	42
4.2.2	Integración de la SoftHand 2	42
4.2.3	Mesa de Trabajo	43
4.2.4	Montaje de Sensores	43
4.3	Integración Eléctrica	44
4.3.1	Fuente de Alimentación	44
4.3.2	Controladores	45
4.3.3	Conexión de la cámara	45
4.4	Selección de Linux como Sistema Operativo Base	46

4.4.1	Criterios de Evaluación	46
4.4.2	Comparativa de Sistemas Operativos	47
4.4.3	Razones para Seleccionar Linux	47
4.4.4	Conclusión	48
4.5	Selección de ROS 2 Humble como Plataforma Base	48
4.5.1	Criterios de Selección	48
4.5.2	Comparativa de Versiones de ROS 2	49
4.5.3	Compatibilidad del Hardware	49
4.5.4	Razones para elegir ROS 2 Humble	49
4.5.5	Conclusión	50
4.6	Creación de los URDFs del UR5e y la SoftHand 2	50
4.6.1	Modelo URDF del UR5e	50
4.6.2	Modelo URDF de la SoftHand 2	50
4.7	Creación del espacio de trabajo en 3D para la planificación	51
4.7.1	Definición del espacio de trabajo	51
4.7.2	Proceso de configuración con MoveIt Setup Assistant	53
4.8	Algoritmos de planificación y control	54
4.8.1	Configuración del solver cinemático	55
4.9	Integración de los controladores	56
4.10	Integración de Sensores	57
4.11	Interacción Humano-Robot	57
4.12	Validación y Simulación	58
4.13	Flujo de información entre los componentes	58
4.13.1	Percepción del entorno	60
4.13.2	Planificación de movimientos	60
4.13.3	Ejecución de movimientos	60
4.13.4	Ventajas del diseño del flujo de información	61
4.14	Consideraciones de seguridad	61
5	Resultados y Análisis	63
5.1	Validación del sistema integrado	64
5.1.1	Prueba de precisión	64
5.1.2	Prueba de carga	68
5.2	Circuitos pick and place	70
5.2.1	Circuito 1: Selección y entrega de objetos	71
5.2.2	Circuito 2: Secuencia de movimiento tipo malabarismo	73
5.2.3	Prueba de levantamiento y manipulación de objetos	74
5.3	Beneficios del proyecto de integración de la mano SoftHand y el robot UR5e	76
5.3.1	Beneficios de investigación	76
5.3.2	Beneficios económicos	76
5.4	Análisis económico	77
5.4.1	Costos del proyecto	77
5.4.2	Relación costo-beneficio	77
5.5	Resumen de resultados	78

6 Conclusiones y recomendaciones	79
6.1 Conclusiones	79
6.2 Recomendaciones	80
Referencias Bibliográficas	82
A Metodología para la evaluación de la compatibilidad	83
A.1 Definición de la Compatibilidad	83
A.2 Tabla de Métricas de Compatibilidad	83
A.3 Interpretación de los Resultados	84
A.4 Cálculo del Porcentaje de Compatibilidad	85
B Análisis de Objetos de Diferentes Geometrías y Tamaños	86
B.1 Dimensiones y Clasificación de los Objetos	86
B.2 Criterios de Selección	87
B.3 Métodos de Cálculo	87
B.4 Relevancia del Análisis	87
C Diseño Mecánico del Adaptador para la SoftHand	88
C.1 Especificaciones del adaptador diseñado	88
C.2 Especificaciones del adaptador comercial	88
C.3 Cálculos del margen de seguridad	89
C.3.1 Cálculo del esfuerzo en la sección crítica	89
C.3.2 Cálculo del margen de seguridad	89
C.4 Comparación entre el adaptador diseñado y el adaptador comercial	90
C.5 Conclusión	90
D Datos en bruto de la prueba de precisión	91
D.1 Datos del error en el eje X	91
D.2 Datos del error en el eje Y	94
D.3 Datos en bruto de la prueba de levantamiento y manipulación	97
D.4 Resultados en el eje X	97
D.5 Resultados en el eje Y	99

Índice de figuras

2.1	Representación simbólica del manipulador UR5e	5
2.2	Sinergias de la mano SoftHand	15
3.1	Etapas de desarrollo de producto	23
3.2	Descomposición de subproblemas en el sistema UR5e-SoftHand 2. Elaboración propia.	31
3.3	Diagrama general de la estrategia de integración. Elaboración propia.	34
3.4	Mapa mental del enfoque de solución. Elaboración propia.	38
4.1	Arquitectura General del Sistema	41
4.2	Adaptador	42
4.3	Montaje de la cámara RealSense L515 en el entorno de trabajo. Elaboración propia.	43
4.4	Fuente de alimentación de la SoftHand 2. Elaboración propia.	45
4.5	Diagrama conceptual de las conexiones eléctricas y de comunicación. Elaboración propia.	46
4.6	Plano de la mano	51
4.7	Espacio de trabajo	52
4.8	Modelo mano	54
4.9	Modelo juntas	55
4.10	Controladores	57
4.11	Flujo de información del sistema	59
5.1	Posiciones objetivo en el plano horizontal. Elaboración propia.	65
5.2	Medición de las desviaciones desde los ejes X e Y . Elaboración propia.	65
5.3	Errores obtenidos en el eje x . Elaboración propia.	66
5.4	Errores obtenidos en el eje y . Elaboración propia.	67
5.5	Botella de 1.0 kg en la SoftHand 2. Elaboración propia.	70
5.6	Disposición de los objetos en el Circuito 1. Elaboración propia.	71
5.7	Sistema manipulando un objeto (vaca de juguete). Elaboración propia.	72

Índice de tablas

3.1	Escala de importancia de las necesidades del proyecto	25
3.2	Necesidades del proyecto y su importancia	26
3.3	Métricas asociadas a las necesidades del proyecto.	28
3.4	Evaluación de soluciones propuestas.	36
3.5	Selección de soluciones para los subproblemas.	36
4.1	Comparativa de sistemas operativos para desarrollo robótico.	47
4.2	Comparativa de versiones de ROS 2.	49
5.1	Posiciones objetivo en el plano horizontal (x y y)	64
5.2	Resultados estadísticos de errores por posición	67
5.3	Especificaciones de botellas seleccionadas para la prueba de carga	68
5.4	Resultados promedio de la prueba de carga	69
5.5	Resultados del Circuito 1 (frecuencias en 100 iteraciones)	72
5.6	Resultados promedio del Circuito 2	73
5.7	Resultados promedio de la prueba de levantamiento y manipulación	75
5.8	Desglose de costos del proyecto en euros.	77
5.9	Métricas asociadas a las necesidades del proyecto con los valores obtenidos.	78
A.1	Métricas de compatibilidad y métodos de evaluación en ROS 2 Humble	84
B.1	Dimensiones de los objetos utilizados en el análisis.	86
C.1	Comparación entre el adaptador diseñado y el adaptador comercial.	90
D.1	Datos del error en el eje X	91
D.2	Datos del error en el eje Y	94
D.3	Datos del eje X	97
D.4	Datos del eje Y	100

Lista de símbolos y abreviaciones

Abreviaciones

AI	Inteligencia Artificial
DL	Aprendizaje Profundo (Deep Learning)
DOF	Grados de libertad (Degrees of Freedom)
FK	Cinemática directa (Forward Kinematics)
FOV	Campo de visión (Field of View)
IK	Cinemática inversa (Inverse Kinematics)
ML	Aprendizaje Automático (Machine Learning)
ROS	Robot Operating System
SLAM	Localización y mapeo simultáneo (Simultaneous Localization and Mapping)
URDF	Unified Robot Description Format

Notación general [I]Matriz identidad.

$\underline{\mathbf{q}}$	Vector de ángulos articulares. $\underline{\mathbf{q}} = [q_1 \quad q_2 \quad \cdots \quad q_n]^T$
$\underline{\mathbf{x}}$	Vector de posición y orientación de un robot. $\underline{\mathbf{x}} = [x \quad y \quad z \quad \phi \quad \theta \quad \psi]^T$
$\mathcal{W}$	Espacio de trabajo de un robot.
$\mathbf{R}$	Matriz de rotación.
$\mathbf{T}$	Matriz de transformación homogénea. $\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix}$
$\underline{\mathbf{v}}$	Vector de velocidades lineales y angulares. $\underline{\mathbf{v}} = [v_x \quad v_y \quad v_z \quad \omega_x \quad \omega_y \quad \omega_z]^T$

Capítulo 1

Introducción

El presente Trabajo de Fin de Graduación (TFG) tiene como objetivo principal la integración del robot UR5e de Universal Robots con la Mano Softhand de QB Robotics, en el contexto de la robótica adaptativa. Este proyecto se enmarca dentro de las líneas de investigación del Grupo Integrado de Ingeniería (GII) de la Universidade da Coruña, en el campus industrial de Ferrol.

1.1 Contexto

El Grupo Integrado de Ingeniería (GII) de la Universidade da Coruña, ubicado en el Campus de Ferrol, es reconocido por su enfoque interdisciplinario en la investigación aplicada en ingeniería, con un énfasis destacado en la transferencia de conocimiento y la generación de soluciones innovadoras para la industria.

Desde su fundación en 1999, el GII ha desarrollado una sólida trayectoria, consolidándose como un referente en diversas áreas de la ingeniería. Su equipo está conformado por profesionales altamente cualificados, provenientes de distintas disciplinas como ingeniería naval, industrial, eléctrica, informática, telecomunicaciones, diseño, entre otras. Esta diversidad permite abordar proyectos complejos desde múltiples perspectivas, favoreciendo un enfoque integral y colaborativo.

La estrecha vinculación del GII con las Escuelas de Ingeniería de la Universidade da Coruña asegura que todos los proyectos se beneficien de un sólido fundamento técnico y académico. Por lo tanto, este entorno de excelencia proporciona una base robusta para la creación y ejecución de proyectos innovadores, en estrecha colaboración con la industria.

En este contexto, el presente proyecto marca el inicio de una nueva línea de investigación dentro del GII, centrada en la robótica adaptativa y cognitiva. Además, la integración lograda en este proyecto servirá como plataforma para futuros desarrollos en robótica colaborativa, adaptativa y cognitiva, ampliando las capacidades del grupo en estos campos emergentes.

Aunque el GII ha trabajado previamente con el robot UR5, esta es la primera vez que se

aborda la integración con la Mano Softhand de QB Robotics, lo que permitirá expandir las funcionalidades del robot en el ámbito de la robótica adaptativa. Al replicar las capacidades de una mano humana, esta integración abrirá nuevas posibilidades para realizar tareas complejas, similares a las que los seres humanos llevan a cabo gracias a la destreza y versatilidad de sus manos.

1.2 Antecedentes del Proyecto

El avance de la robótica en los últimos años ha impulsado el desarrollo de robots más inteligentes, flexibles y adaptativos. La robótica colaborativa, en particular, se ha consolidado como un área clave, permitiendo que humanos y robots trabajen juntos de manera segura y eficiente en entornos compartidos.

En el ámbito industrial, los robots como el UR5e han demostrado ser herramientas versátiles, capaces de realizar tareas repetitivas con alta precisión. Sin embargo, la necesidad de dotar a los robots de habilidades más humanas, como la manipulación adaptativa, ha llevado al desarrollo de actuadores avanzados como la Mano Softhand de QB Robotics. Este dispositivo, inspirado en la anatomía humana, permite una interacción más natural con objetos y herramientas, ampliando significativamente las aplicaciones potenciales de los robots.

1.3 Descripción del Problema

El principal desafío identificado es la falta de integración directa entre el robot UR5e y la Mano Softhand, lo que limita la capacidad del sistema para realizar tareas complejas que requieren una manipulación precisa y adaptable. Aunque existen soluciones parciales en la literatura y en el mercado, estas no abordan completamente la necesidad de una integración robusta y eficiente en un entorno de trabajo colaborativo.

1.4 Justificación del Problema

La integración del UR5e con la Mano Softhand responde a la creciente demanda de sistemas robóticos capaces de operar en entornos dinámicos y no estructurados. Este avance permitirá no solo mejorar la eficiencia de tareas industriales, sino también explorar aplicaciones en sectores como la salud, la agricultura y la logística, donde la manipulación delicada y precisa es crucial.

Además, este proyecto contribuirá al desarrollo de tecnologías innovadoras en el ámbito de la robótica adaptativa, alineándose con las tendencias actuales de investigación y fortaleciendo la posición del GII como un referente en el área.

1.5 Síntesis del Problema

El robot UR5e y la Mano Softhand carecían de una integración mecatrónica que les permitiera operar conjuntamente en tareas colaborativas. Esta limitación impedía que el sistema alcanzara niveles avanzados de coordinación y control en entornos dinámicos. La ausencia de esta capacidad restringía su aplicación en la robótica adaptativa, donde es fundamental lograr una interacción eficiente entre los componentes. Resolver esta carencia ha sido crucial para habilitar un control centralizado mediante ROS 2, facilitando la ejecución de tareas colaborativas y adaptativas con un alto grado de precisión y versatilidad.

1.6 Objetivos del Proyecto

1.6.1 Objetivo General

Integrar el robot UR5e con la Mano Softhand de QB Robotics para desarrollar un sistema robótico capaz de realizar tareas de manipulación adaptativa en entornos colaborativos.

1.6.2 Objetivos Específicos

- Determinar los requerimientos y necesidades del proyecto mediante un análisis exhaustivo, asegurando una base sólida para los siguientes objetivos.
- Diseñar un sistema mecatrónico que incluya la parte mecánica, electrónica y de control para permitir la comunicación estable y eficaz entre la Mano Softhand y el robot UR5.
- Desarrollar un programa computacional que implemente un algoritmo de control compatible con ROS para el manejo coordinado de la mano robótica y el robot UR5.
- Validar el sistema mecatrónico y el programa computacional mediante pruebas utilizando ejemplos de pick and place, demostrando la funcionalidad y eficacia de la integración propuesta.

1.7 Estructura del Documento

El presente documento está estructurado de la siguiente manera:

El capítulo 1 introduce el proyecto, estableciendo el contexto, los antecedentes y la justificación del problema. El capítulo 2 aborda el estado del arte, describiendo los desarrollos previos en robótica colaborativa y adaptativa. El capítulo 3 detalla la metodología empleada para la integración del UR5e con la Mano Softhand. El capítulo 4 presenta los resultados obtenidos y el análisis de las pruebas experimentales realizadas. Finalmente, el capítulo 5 recoge las conclusiones del proyecto y plantea posibles líneas de trabajo futuro.

Capítulo 2

Marco Teórico

En el presente marco teórico se establece los fundamentos necesarios para la integración del manipulador robótico UR5e y la mano robótica SoftHand 2 en un entorno de desarrollo basado en ROS 2. Esta integración representa un avance en el ámbito de la robótica, al combinar la precisión y repetibilidad de un manipulador rígido con la adaptabilidad y seguridad inherente de una mano robótica suave. Este enfoque tiene aplicaciones clave en tareas como la manipulación colaborativa y la interacción humano-robot en entornos industriales y médicos.

A su vez el proyecto requiere un análisis detallado de conceptos fundamentales como la cinemática, la dinámica y la planificación de movimientos, necesarios para garantizar la precisión y eficiencia en la ejecución de tareas. Además, el uso de ROS 2 proporciona una plataforma robusta y modular para la gestión de nodos, configuración de controladores y comunicación entre dispositivos heterogéneos por lo tanto, el uso de herramientas como URDF y MoveIt permiten modelar, simular y validar las configuraciones antes de implementarlas en el entorno físico, minimizando riesgos y optimizando el diseño.

Finalmente, el marco teórico analiza las características técnicas del UR5e y la SoftHand 2, destacando sus capacidades y limitaciones, junto con los algoritmos y controladores necesarios para sincronizar sus movimientos. También se abordan los avances más recientes en la integración de sistemas robóticos y los desafíos asociados a la coordinación de múltiples dispositivos.

2.1 Fundamentos de robótica

La robótica combina principios de ingeniería mecánica, eléctrica y computacional para diseñar, construir y controlar sistemas capaces de realizar tareas específicas en entornos diversos. Este proyecto se enfoca en la integración del manipulador robótico UR5e y la mano robótica SoftHand 2, lo cual requiere una comprensión profunda de conceptos como cinemática, dinámica, planificación de movimientos, sensores y actuadores.

2.1.1 ¿Qué es un robot manipulador?

Un robot manipulador es un sistema mecánico diseñado para realizar tareas específicas mediante la interacción con su entorno a través de un efector terminal. Estos sistemas están compuestos por eslabones rígidos conectados mediante articulaciones móviles, que pueden ser rotacionales o prismáticas, permitiendo una amplia gama de movimientos. Su uso es común en aplicaciones que requieren precisión, como ensamblaje, manipulación de objetos, soldadura o tareas de pick and place en líneas de producción automatizadas [1].

Además el manipulador se clasifica como un sistema cinemático en cadena abierta, donde un extremo está fijo a una base y el otro se conecta al efector terminal. En el caso del UR5e, sus seis grados de libertad le permiten alcanzar posiciones y orientaciones específicas dentro de su espacio de trabajo. Lo anterior es posible mediante el uso de algoritmos de control y planificación de movimientos que aseguran precisión y repetibilidad.

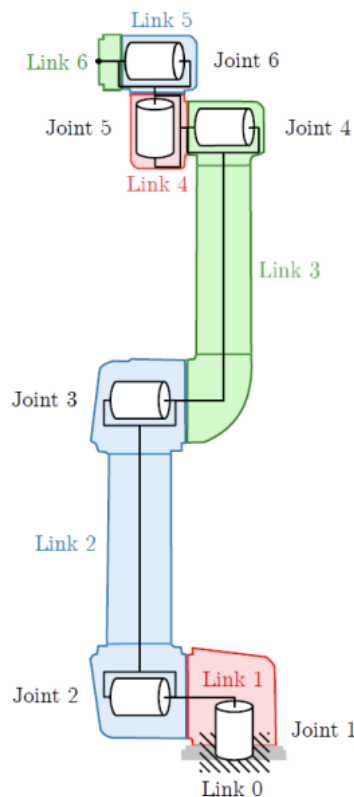


Figura 2.1: Representación simbólica del manipulador UR5e. [2]

2.1.2 Cinemática directa

La cinemática directa permite determinar la posición y orientación del efector terminal de un manipulador robótico a partir de los valores conocidos de las variables articulares. Para un robot como el UR5e, esta relación se describe mediante transformaciones homogéneas que definen la posición de cada eslabón en relación con un sistema de coordenadas base [1].

$$T(\theta) = \prod_{i=1}^n T_i(\theta_i) \quad (2.1)$$

Donde:

- $T(\theta)$ es la matriz de transformación total que combina las posiciones y orientaciones de los eslabones.
- $T_i(\theta_i)$ representa la transformación homogénea de cada eslabón en función de su variable articular θ_i .
- $\prod_{i=1}^n$ denota el producto acumulativo desde el eslabón base hasta el efector terminal.

Esta ecuación es esencial para tareas como la planificación de trayectorias, ya que permite calcular cómo se posicionará el efector terminal al variar las configuraciones articulares. Además, su implementación en herramientas de simulación como MoveIt o Gazebo garantiza una validación previa antes de ejecutar tareas en el hardware físico, reduciendo riesgos y mejorando la precisión en el control.

2.1.3 Cinemática inversa

La cinemática inversa se centra en calcular las configuraciones articulares necesarias para alcanzar una posición y orientación deseadas del efector terminal en el espacio cartesiano. Este problema es más complejo que la cinemática directa debido a la existencia de múltiples soluciones, singularidades y posibles configuraciones fuera del espacio de trabajo [1, 3].

$$\mathbf{q} = f^{-1}(\mathbf{T}) \quad (2.2)$$

Donde:

- $\mathbf{q}$ es el vector de variables articulares.
- $\mathbf{T}$ representa la posición y orientación deseadas.
- f^{-1} es la función inversa de la cinemática directa.

Sin embargo, la cinemática inversa es fundamental en tareas como ensamblaje, manipulación de objetos y teleoperación, ya que permite al robot adaptarse a posiciones dinámicas o entornos cambiantes. En el caso del UR5e, ROS 2 y MoveIt implementan algoritmos avanzados para resolver la cinemática inversa de manera eficiente, evitando configuraciones no deseadas y mejorando la capacidad de respuesta en tiempo real.

2.1.4 Espacio de trabajo y configuraciones

El espacio de trabajo de un manipulador robótico se define como el volumen tridimensional dentro del cual el efector terminal puede operar, considerando todas las posibles combinaciones de las variables articulares. Este concepto es fundamental para evaluar la funcionalidad y las limitaciones físicas del manipulador, ya que delimita las regiones del espacio donde el robot puede realizar tareas de manera efectiva.

Según Craig [1], el espacio de trabajo se puede categorizar en dos ámbitos principales. El primero, denominado espacio de trabajo accesible, incluye todas las posiciones y orientaciones que el efector terminal es capaz de alcanzar, sin considerar restricciones operativas o de eficiencia. El segundo, conocido como espacio de trabajo utilizable, representa un subconjunto del espacio accesible, limitado a aquellas regiones donde los movimientos del manipulador son seguros, precisos y energéticamente eficientes.

A su vez las configuraciones articulares del manipulador también desempeñan un papel crucial en su operación, ya que determinan cómo se posicionan y orientan los eslabones y el efector terminal dentro del espacio de trabajo. Un mismo punto en el espacio cartesiano puede ser alcanzado por múltiples configuraciones articulares, especialmente en manipuladores redundantes, como señala Spong [3]. Esta multiplicidad de configuraciones es clave en la planificación de trayectorias, ya que permite seleccionar movimientos óptimos basados en criterios como la minimización del consumo de energía, la reducción del tiempo de ejecución o la maximización de la estabilidad.

En el caso específico del UR5e, su diseño con seis grados de libertad le otorga una notable flexibilidad para maximizar el espacio de trabajo. Esto permite al robot operar de manera eficiente en entornos industriales, donde se requiere alta precisión y repetibilidad, y en entornos colaborativos, donde la seguridad y la adaptabilidad son esenciales [4]. Adicionalmente, la planificación cuidadosa del espacio de trabajo y la selección adecuada de configuraciones articulares son fundamentales para garantizar un rendimiento óptimo, minimizar las singularidades y evitar colisiones, ya sea con el entorno o con el propio manipulador [5].

2.1.5 Modelado de robots

El modelado de robots es un aspecto esencial en el análisis, diseño y control de manipuladores robóticos. Este proceso permite representar matemáticamente las relaciones entre las articulaciones, los eslabones y las fuerzas que actúan sobre el sistema. Los modelos robóticos son fundamentales para resolver problemas de cinemática y dinámica, además de optimizar la planificación y el control de movimientos.

Uno de los enfoques más utilizados en el modelado cinemático de robots es el método de parámetros Denavit-Hartenberg (DH), que proporciona una representación sistemática de la geometría del manipulador. Este método define las relaciones entre eslabones consecutivos mediante cuatro parámetros: longitud del eslabón, ángulo de torsión, desplazamiento del eslabón y ángulo de articulación. La simplificación que ofrece el enfoque DH es ampliamente reconocida en la literatura como una herramienta estándar para representar la cinemática directa de manipuladores, incluyendo modelos complejos como el UR5e [1].

Por otro lado, el modelado dinámico utiliza principios fundamentales de la mecánica para describir cómo las fuerzas y los torques afectan los movimientos del manipulador. Las ecuaciones de Euler-Lagrange son un enfoque predominante en este campo, ya que relacionan las fuerzas y torques necesarios con las aceleraciones articulares del sistema. Este método, basado en la energía cinética y potencial del sistema, permite calcular las fuerzas requeridas para realizar tareas dinámicas, considerando las interacciones entre las articulaciones [3].

En el contexto del UR5e, los parámetros DH facilitan la descripción de su estructura cinemática en herramientas de simulación como MoveIt, mientras que las ecuaciones de Euler-Lagrange son esenciales para determinar los torques necesarios durante la ejecución de tareas dinámicas. Estas técnicas se complementan con avances recientes en robótica, como el uso de métodos basados en dinámica computacional y aprendizaje automático, que permiten mejorar la precisión y adaptabilidad de los modelos en aplicaciones reales [6].

El modelado de robots también enfrenta desafíos importantes, especialmente en manipuladores con alta redundancia o que operan en entornos dinámicos. Las singularidades y las no linealidades son problemas recurrentes que requieren técnicas avanzadas para su resolución. En años recientes, investigaciones han propuesto métodos híbridos que combinan técnicas tradicionales, como Denavit-Hartenberg y Euler-Lagrange, con algoritmos basados en inteligencia artificial, lo que permite generar modelos más robustos y eficientes [7, 8].

Además la implementación precisa de modelos cinemáticos y dinámicos no solo es crucial para el control de robots como el UR5e, sino también para su integración con dispositivos avanzados como la SoftHand. Esto asegura que las trayectorias y movimientos planificados sean ejecutados con precisión, incluso en entornos colaborativos y aplicaciones industriales.

2.1.6 Sensores y actuadores

Los sensores y actuadores constituyen componentes esenciales en los manipuladores robóticos, ya que permiten la interacción eficiente entre el robot y su entorno. También, los sensores son dispositivos encargados de captar información del entorno o del estado del robot, mientras que los actuadores convierten señales eléctricas en movimiento o fuerza mecánica, ejecutando así las acciones necesarias para cumplir con las tareas asignadas.

Entre los sensores más utilizados en robótica se encuentran los encoders, los cuales son dispositivos que permiten medir con precisión las posiciones articulares del manipulador. Estos sensores son fundamentales para garantizar la correcta ejecución de movimientos planificados. Además, los sensores de fuerza/par juegan un papel crucial al detectar las interacciones entre el robot y su entorno, lo que resulta especialmente relevante en tareas colaborativas donde se prioriza la seguridad y la adaptabilidad. Por otro lado, las cámaras empleadas en sistemas de visión artificial permiten al robot identificar objetos, calcular posiciones y adaptarse a entornos dinámicos, lo que amplía significativamente sus capacidades funcionales [5, 7].

Por otro lado, los actuadores, son los responsables de mover las articulaciones del manipulador. A su vez, los motores eléctricos como los servomotores y motores paso a paso, se utilizan ampliamente debido a su precisión y control avanzado. Estos actuadores permiten un posicionamiento exacto y un control dinámico en tiempo real, lo que los hace ideales para manipuladores como el UR5e.

En el ámbito de la robótica suave, los actuadores adaptativos, como los utilizados en la SoftHand, emplean materiales flexibles y mecanismos diseñados para sinergias motoras. Estos actuadores permiten una distribución uniforme de las fuerzas y un ajuste automático a las geometrías de los objetos manipulados, lo que es crucial en aplicaciones que requieren la manipulación segura de objetos delicados [9, 10].

Además, el UR5e emplea encoders de alta precisión en cada una de sus seis articulaciones, lo que garantiza un control exacto de los movimientos y asegura que el manipulador opere dentro de los límites de su espacio de trabajo. Por su parte, la SoftHand incorpora actuadores diseñados para aprovechar las sinergias motoras, reduciendo la complejidad mecánica y mejorando la eficiencia energética del sistema. Esta combinación de sensores avanzados y actuadores adaptativos permite al sistema integrado UR5e-SoftHand operar con un alto grado de precisión, seguridad y adaptabilidad, lo que lo convierte en una solución versátil para aplicaciones colaborativas e industriales [11, 7].

2.1.7 Control de robots

El control de robots es un aspecto fundamental en la robótica moderna, ya que asegura que los manipuladores sigan trayectorias planificadas con precisión, eficiencia y seguridad. Este proceso implica la regulación de las posiciones, velocidades y fuerzas generadas por las articulaciones del manipulador para cumplir con los objetivos establecidos, ya sea en entornos estructurados o dinámicos.

Entre los métodos más comunes de control se encuentra el control por retroalimentación, que utiliza la información proporcionada por sensores para ajustar las acciones del robot en tiempo real. Un ejemplo ampliamente utilizado es el controlador PID (Proporcional-Integral-Derivativo), que permite corregir errores en posición o velocidad mediante un ajuste continuo basado en la retroalimentación sensorial. Este enfoque es especialmente útil para garantizar la estabilidad y precisión en tareas donde las perturbaciones externas pueden afectar el rendimiento del robot [3].

El control basado en trayectorias es otro método crucial, orientado al diseño de movimientos suaves y eficientes entre puntos objetivos. Este enfoque utiliza modelos cinemáticos y dinámicos del robot para generar trayectorias que cumplan con restricciones de velocidad, aceleración y fuerzas aplicadas, maximizando la eficiencia y minimizando las vibraciones o movimientos indeseados durante la ejecución de tareas [1].

El control de impedancia es particularmente relevante en aplicaciones de interacción humano-robot y en entornos colaborativos. Este método ajusta las fuerzas ejercidas por el robot en función de las características del entorno o las acciones humanas, proporcionando un comportamiento adaptativo que mejora tanto la seguridad como la eficacia de las interacciones. Por ejemplo, al manipular objetos frágiles, el control de impedancia permite al robot reducir automáticamente la fuerza aplicada para evitar daños [5].

En el contexto de este proyecto, ROS 2 juega un papel fundamental al proporcionar herramientas avanzadas para la implementación de controladores.

El `scaled_joint_trajectory_controller` es un ejemplo clave, utilizado para gestionar los movimientos precisos del manipulador UR5e mediante el seguimiento de trayectorias planificadas. Por otro lado, la `SoftHand` complementa estas capacidades al incorporar un control adaptativo que aprovecha las sinergias motoras, ajustándose automáticamente a la geometría y textura de los objetos manipulados [9, 10].

La combinación de estos métodos de control y las capacidades proporcionadas por ROS 2 permiten que el sistema integrado UR5e-SoftHand opere de manera eficiente en tareas colaborativas e industriales, garantizando precisión en los movimientos y adaptabilidad en entornos dinámicos.

2.1.8 Dinámica y singularidades

La dinámica de un manipulador robótico se ocupa del estudio de las fuerzas y torques necesarios para producir movimientos específicos en sus articulaciones. Este análisis se fundamenta en modelos matemáticos como las ecuaciones de Newton-Euler y Lagrange, que describen las relaciones entre las aceleraciones articulares y las fuerzas externas aplicadas al robot. La dinámica es esencial para el diseño de controladores robustos que permitan manejar tanto movimientos rápidos como cargas externas inesperadas. En el caso del UR5e, el modelo dinámico permite calcular con precisión los torques requeridos en sus seis grados de libertad, asegurando estabilidad y eficiencia en la ejecución de tareas [3].

Por otro lado, las singularidades representan configuraciones críticas en las que el manipulador pierde grados de libertad, lo que limita su capacidad de movimiento en ciertas direcciones. Estas situaciones pueden generar problemas como la incapacidad de aplicar fuerzas en direcciones específicas o inestabilidades en los cálculos de la cinemática inversa. Las singularidades suelen ocurrir cuando varias articulaciones se alinean, reduciendo efectivamente las opciones de movimiento del robot [5]. En el caso del UR5e, estas configuraciones pueden presentarse cuando el robot se aproxima a los límites de su espacio de trabajo o cuando sus articulaciones adoptan ángulos particulares que reducen la efectividad de la matriz jacobiana.

Para mitigar los efectos de las singularidades, se emplean algoritmos avanzados de planificación de trayectorias que evitan estas configuraciones críticas. Herramientas como MoveIt integran técnicas de detección y evasión de singularidades, permitiendo que el robot optimice sus movimientos en tiempo real y garantice un desempeño continuo. Esto es particularmente importante en aplicaciones colaborativas o en entornos industriales donde la precisión y la confiabilidad son esenciales [12, 7].

La combinación de un modelado dinámico preciso y la capacidad de manejar singularidades mediante planificación avanzada es fundamental para garantizar que el sistema UR5e-SoftHand funcione de manera óptima, incluso en escenarios exigentes.

2.1.9 Planificación de movimientos

La planificación de movimientos es una disciplina fundamental en la robótica, que permite coordinar los desplazamientos de un manipulador en su espacio de trabajo, garantizando que las trayectorias generadas sean seguras, eficientes y libres de colisiones. Este proceso requiere una representación adecuada del espacio de trabajo del robot, conocida como *espacio de configuraciones* (C-space). El *C-space* incluye todas las posibles configuraciones articulares del manipulador, permitiendo una representación completa de los movimientos factibles del robot. Esta representación es especialmente útil para identificar regiones accesibles y detectar posibles restricciones, como límites articulares u obstáculos en el entorno [4].

La generación de trayectorias en el espacio de configuraciones se basa en algoritmos de planificación avanzados que optimizan el movimiento del robot en términos de distancia, tiempo

o energía. Entre los métodos más utilizados se encuentran los árboles de exploración rápida (RRT, Rapidly-exploring Random Trees) y los mapas probabilísticos (PRM, Probabilistic Roadmap). Los algoritmos RRT son especialmente eficaces en entornos dinámicos o de alta complejidad, ya que generan trayectorias incrementales que exploran rápidamente el espacio de configuraciones. Por otro lado, los PRM son más adecuados para entornos estáticos, donde se pueden precomputar rutas óptimas utilizando un enfoque basado en grafos [12, 7].

En el contexto del manipulador UR5e y la SoftHand, estos algoritmos se implementan a través de herramientas como MoveIt, que integra funcionalidades para la planificación de trayectorias, detección de colisiones y optimización de movimientos. Estas capacidades aseguran que el sistema pueda operar de manera eficiente, incluso en entornos colaborativos o con restricciones significativas. Además, MoveIt permite la validación previa de las trayectorias en simuladores como Gazebo, lo que minimiza riesgos y optimiza la transición del diseño al hardware físico.

La planificación de movimientos también es crucial para cumplir con los requisitos específicos de tareas asignadas. Por ejemplo, en aplicaciones industriales, los algoritmos deben garantizar que el UR5e y la SoftHand realicen manipulaciones precisas y rápidas, mientras que en entornos colaborativos se prioriza la seguridad y la adaptabilidad del sistema. La integración de sensores, como cámaras de visión artificial, mejoran aún más la capacidad del sistema para ajustarse en tiempo real a cambios en el entorno o en los objetos manipulados.

En resumen, la planificación de movimientos no solo optimiza el desempeño del sistema UR5e-SoftHand, sino que también permite una interacción segura y eficiente con su entorno, alineándose con los objetivos de precisión, flexibilidad y colaboración en entornos robóticos avanzados.

2.1.10 Interacción humano-robot

La interacción humano-robot es un área clave en el diseño de sistemas robóticos modernos, especialmente en entornos colaborativos donde los robots y los humanos comparten espacios de trabajo y cooperan en tareas conjuntas. Este tipo de interacción debe cumplir con los más altos estándares de seguridad, adaptabilidad y eficacia, asegurando que los robots puedan operar cerca de los humanos sin comprometer su bienestar ni reducir la productividad del sistema [11].

La colaboración segura es uno de los principios fundamentales en la interacción humano-robot. Los robots colaborativos, como el UR5e, están equipados con sensores avanzados que permiten detectar colisiones o proximidad, ajustando automáticamente su comportamiento para evitar accidentes. Este enfoque se complementa con el control de impedancia, que regula las fuerzas ejercidas por el robot, adaptándolas a la rigidez del entorno o a las propiedades del objeto manipulado. Estas capacidades no solo mejoran la seguridad, sino que también optimizan la adaptabilidad del sistema en tareas dinámicas o con alta incertidumbre [7].

Por otro lado, las interfaces de control desempeñan un papel crucial en facilitar la comunicación y la cooperación entre humanos y robots. Herramientas como joysticks, señales

electromiográficas (EMG) y sistemas de visión artificial permiten a los operadores dirigir los movimientos del robot de manera intuitiva y eficiente. Estas interfaces son particularmente útiles en aplicaciones donde la intervención humana es necesaria, como en operaciones de montaje de precisión, asistencia médica o manipulación en entornos no estructurados [13].

En el caso del sistema UR5e-SoftHand, estas características se combinan para ofrecer un alto nivel de colaboración. El UR5e, gracias a su diseño y controladores avanzados, es capaz de trabajar de manera segura y precisa en proximidad con humanos. Por su parte, la SoftHand con su diseño basado en materiales flexibles y sinergias motoras, añade un nivel adicional de seguridad al manipular objetos delicados. La capacidad de la SoftHand para distribuir fuerzas de manera uniforme reduce significativamente el riesgo de daño tanto a los objetos como a las personas, haciéndola ideal para aplicaciones industriales y médicas [9].0

En conclusión, la interacción humano-robot en el sistema UR5e-SoftHand combina tecnologías avanzadas de seguridad, control y adaptabilidad, permitiendo que los robots no solo operen de manera eficiente en entornos colaborativos, sino que también contribuyan a mejorar la experiencia y el rendimiento de los operadores humanos.

2.1.11 Avances tecnológicos en robótica

Los avances tecnológicos en robótica han transformado significativamente la forma en que los sistemas robóticos interactúan con su entorno y ejecutan tareas. Estas innovaciones no solo han mejorado el rendimiento y la autonomía de los robots, sino que también han ampliado el rango de aplicaciones en sectores como la industria, la medicina y la exploración.

Uno de los avances más destacados es la robótica en la nube. ROS 2, con su arquitectura distribuida, permite a los robots acceder a datos y realizar procesos en la nube. Esto reduce las limitaciones de hardware local y mejora la capacidad de procesamiento en tiempo real, habilitando aplicaciones como la coordinación de flotas de robots o el aprendizaje compartido entre dispositivos. La robótica en la nube también facilita actualizaciones remotas y el intercambio de información entre sistemas, optimizando la colaboración y la eficiencia en entornos distribuidos [14].

Otro desarrollo clave es el aprendizaje por refuerzo, una técnica de inteligencia artificial que permite a los robots aprender tareas complejas a través de la interacción con su entorno. A diferencia de los métodos tradicionales, donde el comportamiento debe ser programado explícitamente, los robots que utilizan aprendizaje por refuerzo pueden adaptarse a escenarios dinámicos, optimizando sus estrategias a medida que adquieren experiencia. Este enfoque ha sido fundamental en aplicaciones que requieren adaptabilidad, como la manipulación de objetos irregulares, la navegación autónoma y la interacción humano-robot [13].

En cuanto a las tendencias recientes, la integración de la robótica blanda y los sistemas híbridos ha marcado un cambio en la forma en que se diseñan los robots. La combinación de estructuras rígidas y materiales flexibles permite crear sistemas que combinan la precisión de los robots tradicionales con la adaptabilidad y seguridad de los robots suaves. Esto resulta particularmente relevante en aplicaciones colaborativas, donde la interacción segura

y eficiente con los humanos es primordial. Estos avances no solo mejoran la funcionalidad de los robots, sino que también abren nuevas posibilidades para su implementación en entornos médicos, industriales y de investigación [11].

En resumen, los avances en la robótica en la nube, el aprendizaje por refuerzo y la robótica blanda están redefiniendo los límites de lo que los robots pueden lograr, alineándose directamente con los objetivos del proyecto UR5e-SoftHand y permitiendo soluciones innovadoras para tareas complejas y colaborativas.

2.2 Robótica suave (Soft robotics)

La robótica suave es un enfoque innovador que busca imitar los principios biológicos mediante el uso de materiales flexibles y adaptativos, permitiendo interacciones seguras y efectivas con entornos dinámicos. A diferencia de los robots convencionales, que se basan en estructuras rígidas y diseñadas para operar en entornos controlados, los robots suaves están optimizados para adaptarse a entornos no estructurados y objetos de formas y texturas variadas. Este enfoque tiene su origen en la biomecánica y está inspirado en la forma en que los organismos vivos interactúan con el mundo [13, 11].

La robótica suave se caracteriza por emplear materiales como siliconas, elastómeros y polímeros inteligentes, junto con diseños que aprovechan propiedades como la deformación controlada y la adaptabilidad estructural. Estas características permiten a los robots suaves manipular objetos frágiles, adaptarse a superficies irregulares y trabajar de manera colaborativa con humanos en tareas que requieren un alto grado de seguridad y flexibilidad.

2.2.1 Ventajas de la robótica suave

La robótica suave ofrece una serie de ventajas que la posicionan como una tecnología clave para aplicaciones que requieren interacción segura y adaptativa en entornos dinámicos. Una de sus principales características es su adaptabilidad, que permite a los robots suaves ajustarse a la forma y textura de los objetos con los que interactúan. Esta capacidad es particularmente útil para manipular objetos frágiles o de geometrías irregulares, ya que las fuerzas se distribuyen de manera uniforme, minimizando el riesgo de daño durante la operación [11].

Otra ventaja destacada es la seguridad intrínseca proporcionada por los materiales flexibles y deformables que componen los robots suaves. Estos materiales reducen significativamente el riesgo de lesiones en interacciones con humanos, especialmente en entornos colaborativos donde la proximidad entre robots y personas es inevitable. Esto hace que los robots suaves sean ideales para entornos como hospitales, fábricas y laboratorios donde la seguridad es una prioridad.

La robótica suave también mejora la eficacia en la interacción humano-robot, permitiendo una manipulación más natural y eficiente. Gracias a su diseño inspirado en principios

biológicos, los robots suaves pueden adaptarse dinámicamente a cambios en su entorno y realizar tareas complejas con mayor precisión. Esto es especialmente importante en aplicaciones que involucren objetos de formas y texturas variadas, así como en tareas colaborativas donde la interacción efectiva es crucial [13].

Estas ventajas posicionan a la robótica suave como una solución ideal para aplicaciones médicas, como el diseño de prótesis avanzadas y exoesqueletos, que emulan de manera precisa los movimientos humanos. Asimismo, la delicadeza y adaptabilidad de los robots suaves los hacen esenciales en tareas industriales, como la manipulación de componentes electrónicos sensibles o alimentos, donde los métodos tradicionales pueden resultar inadecuados. En resumen, la robótica suave no solo amplía las capacidades de los robots tradicionales, sino que también abre nuevas oportunidades en campos donde la seguridad, la adaptabilidad y la interacción natural son fundamentales.

2.2.2 Manos robóticas suaves

Un ejemplo destacado de robótica suave es la SoftHand Research V2 de QB Robotics. Esta mano robótica está diseñada para emular la funcionalidad de una mano humana, utilizando un enfoque basado en sinergias motoras. Estas sinergias combinan múltiples grados de libertad en patrones de movimiento predefinidos, lo que simplifica el control y mejora su adaptabilidad [9, 10].

El diseño de la SoftHand emplea materiales flexibles y mecanismos adaptativos que permiten ajustarse a diferentes formas y texturas. Cuando interactúa con un objeto, los dedos se ajustan automáticamente a su geometría, distribuyendo las fuerzas de manera uniforme y minimizando el riesgo de daño. Esto la hace especialmente útil en aplicaciones médicas e industriales, como la manipulación de objetos frágiles en líneas de ensamblaje o la asistencia a personas con discapacidades motoras [11].

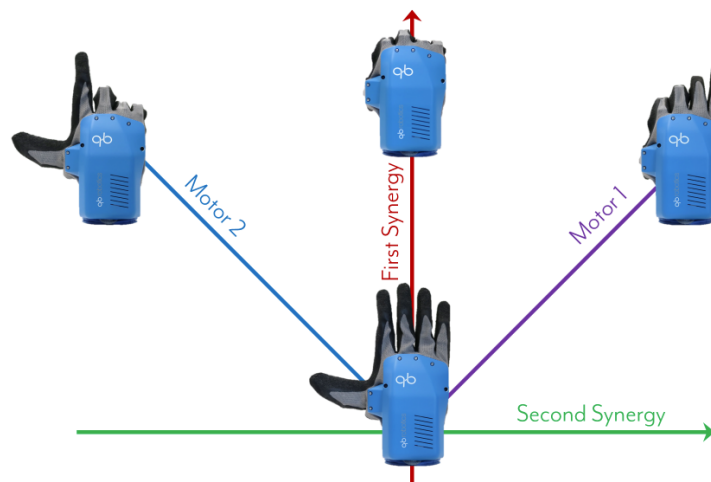


Figura 2.2: Sinergias de la mano SoftHand [15]

Principios de sinergias motoras

La SoftHand se basa en el concepto de sinergias motoras, que son patrones coordinados de movimiento derivados de la biomecánica humana. Este enfoque permite reducir la complejidad de control al utilizar un número limitado de actuadores para manejar múltiples articulaciones [9]. Las sinergias no solo simplifican el diseño mecánico, sino que también mejoran la eficiencia energética del sistema.

Por ejemplo, en lugar de controlar cada dedo de manera independiente, la SoftHand utiliza un motor central que acciona todos los dedos en un movimiento coordinado. Esto no solo facilita la implementación de tareas como el agarre, sino que también permite una respuesta adaptativa a objetos con formas irregulares o superficies complejas.

2.2.3 Aplicaciones de la robótica suave

La robótica suave se ha consolidado como una solución innovadora en diversos campos gracias a las propiedades únicas que ofrece su diseño basado en materiales flexibles y adaptativos. Estas características permiten su implementación en una amplia variedad de aplicaciones que van desde la industria hasta la exploración y la medicina.

En el ámbito industrial, los robots suaves son ideales para la manipulación de objetos delicados y frágiles en líneas de ensamblaje y empaquetado. Gracias a su capacidad para distribuir fuerzas de manera uniforme y ajustarse a las formas de los objetos, estos robots garantizan una manipulación precisa y segura, minimizando el riesgo de daños. Esto resulta especialmente valioso en sectores como la electrónica y la alimentación, donde la precisión y la delicadeza son esenciales [11].

En medicina, la robótica suave ha demostrado un enorme potencial en el desarrollo de prótesis funcionales y dispositivos asistivos, como exoesqueletos, que imitan los movimientos humanos con alta precisión. Estos dispositivos no solo mejoran la calidad de vida de las personas con discapacidades motoras, sino que también permiten una rehabilitación más efectiva gracias a su capacidad de adaptarse a las necesidades específicas de cada paciente [13].

Por otro lado, la exploración es otro campo en el que la robótica suave destaca, especialmente en entornos extremos o inaccesibles para robots tradicionales. Los robots suaves han sido empleados en exploraciones submarinas, donde su flexibilidad y resistencia a la presión les permiten operar en condiciones adversas. Además, son útiles en terrenos accidentados o entornos donde los materiales rígidos no son adecuados, como en exploraciones espaciales o biológicas [11].

Finalmente, en el contexto de la interacción humano-robot, la robótica suave ha revolucionado las aplicaciones colaborativas. Los robots diseñados con materiales flexibles garantizan la seguridad y la eficacia al trabajar en proximidad con humanos, evitando riesgos de lesiones en caso de colisión. Esto ha permitido su integración en entornos como fábricas inteligentes, hospitales y laboratorios, donde la interacción humano-robot es crucial para el éxito de

las operaciones [13]. En resumen, la robótica suave está transformando la manera en que los robots interactúan con su entorno y con los humanos, abriendo nuevas posibilidades en aplicaciones que requieren una combinación de adaptabilidad, seguridad y precisión.

2.2.4 Desafíos y futuras tendencias

Aunque la robótica suave presenta ventajas significativas, también enfrenta una serie de desafíos técnicos que deben abordarse para consolidar su aplicación en entornos complejos y dinámicos. Uno de los principales retos radica en el modelado y simulación de los materiales suaves, ya que la naturaleza intrínsecamente no lineal de estos materiales, que permite su deformación y adaptabilidad, complica la creación de modelos matemáticos precisos. Este desafío limita la capacidad de predecir el comportamiento del robot en condiciones reales, afectando la planificación y el control de tareas [11].

Otro desafío importante es el control avanzado de robots suaves. Debido a la complejidad inherente de las deformaciones y los movimientos adaptativos, se requieren algoritmos sofisticados capaces de procesar grandes volúmenes de datos en tiempo real. Esto incluye el desarrollo de controladores que integren información de sensores externos y ajusten dinámicamente los movimientos del robot, garantizando un rendimiento óptimo en tareas colaborativas e impredecibles [13].

El desarrollo de materiales duraderos es otro aspecto crítico en la investigación de la robótica suave. Los materiales actuales, como elastómeros y polímeros, ofrecen la flexibilidad necesaria pero suelen presentar limitaciones en cuanto a resistencia al desgaste, fatiga y degradación ambiental. Avances en materiales inteligentes y compuestos híbridos son esenciales para extender la vida útil y la funcionalidad de los robots suaves en aplicaciones prácticas [11].

De cara al futuro, se espera que la robótica suave se beneficie de una integración más profunda con tecnologías avanzadas como la inteligencia artificial y el aprendizaje automático. Estas tecnologías permitirán a los robots suaves adaptarse de manera más autónoma y eficiente a cambios en el entorno, optimizando su desempeño en tareas dinámicas. Además, el diseño de sistemas híbridos que combinen componentes rígidos y suaves representa una dirección prometedora. Estos sistemas híbridos pueden aprovechar la precisión y robustez de los componentes rígidos mientras mantienen la flexibilidad y seguridad que caracteriza a los sistemas suaves, resolviendo así problemas complejos en aplicaciones industriales, médicas y de exploración [11, 13].

Otra tendencia relevante es el desarrollo de materiales inteligentes, capaces de responder a estímulos externos como temperatura, luz o campos magnéticos. Estos materiales podrían mejorar significativamente las capacidades adaptativas de los robots suaves, expandiendo su funcionalidad a entornos extremos o altamente dinámicos. Asimismo, la robótica suave podría desempeñar un papel fundamental en la creación de dispositivos biomiméticos y prótesis avanzadas, ampliando su impacto en el ámbito médico [11].

En conclusión, aunque existen barreras técnicas significativas, las investigaciones actuales y los avances emergentes en materiales, algoritmos y diseño de sistemas híbridos auguran

un futuro prometedor para la robótica suave, con aplicaciones en diversos sectores que van desde la industria manufacturera hasta la medicina y la exploración.

2.3 Robot Operating System (ROS)

Robot Operating System (ROS) es un marco de software de código abierto ampliamente utilizado para el desarrollo de sistemas robóticos. Proporciona herramientas, bibliotecas y convenciones diseñadas para simplificar las tareas de diseño, control y simulación de robots. Desde su creación, ROS ha evolucionado hasta convertirse en un estándar de facto en la robótica, ofreciendo un entorno modular y escalable que permite la integración de múltiples componentes [16].

2.3.1 Historia y evolución

El desarrollo inicial de ROS se llevó a cabo en el laboratorio de inteligencia artificial de Stanford y en Willow Garage en 2007. Su objetivo principal era proporcionar un marco flexible que facilitara el desarrollo de robots complejos a través de la reutilización de código y la colaboración entre investigadores. Con el tiempo, ROS se consolidó como una herramienta esencial en la investigación y la industria robótica. En 2017 se lanzó ROS 2, diseñado para superar las limitaciones de la primera versión, como la falta de soporte para comunicación en tiempo real, seguridad y escalabilidad en sistemas distribuidos. Esta segunda versión marcó un hito al ofrecer capacidades avanzadas y adaptarse a los requisitos de sistemas robóticos más exigentes [6].

2.3.2 Características Principales

ROS destaca por su arquitectura modular, que permite dividir un sistema robótico en nodos, cada uno encargado de una tarea específica. Esto facilita el diseño, desarrollo y depuración de sistemas complejos, al tiempo que fomenta la colaboración y la reutilización de código. La comunicación entre nodos se lleva a cabo mediante un sistema de publicación-suscripción basado en tópicos, que permite un flujo de datos eficiente y flexible. Además, ROS es compatible con múltiples plataformas, incluyendo Linux, macOS y Windows, lo que lo hace accesible a una amplia comunidad de usuarios.

Herramientas como Gazebo y RViz permiten la simulación y visualización de robots en entornos virtuales, posibilitando la validación de configuraciones antes de su implementación en hardware real. La vasta colección de paquetes disponibles en ROS facilita la integración de sensores, actuadores y algoritmos avanzados de planificación y control [16, 6].

2.3.3 ROS 2 y su importancia

ROS 2 introduce mejoras significativas respecto a su predecesor. Entre estas destacan la incorporación del middleware DDS (Data Distribution Service), que habilita la comunicación en tiempo real entre nodos, una característica crucial para aplicaciones de robótica colaborativa y autónoma.

También se ha mejorado la seguridad mediante la integración de autenticación y encriptación, lo que garantiza la privacidad y la protección de datos. Además, ROS 2 optimiza su funcionamiento en sistemas embebidos, lo que amplía las posibilidades de aplicación en dispositivos con recursos limitados. Estas mejoras hacen de ROS 2 una herramienta indispensable para proyectos como la integración del UR5e y la SoftHand, donde se requiere una gestión eficiente de nodos, sensores y actuadores [14].

2.3.4 Funcionamiento de Tópicos, Servicios y Acciones

En ROS, los tópicos constituyen la base de la comunicación entre nodos, siguiendo un modelo de publicación-suscripción. Los nodos publicadores envían mensajes a un tópico específico, mientras que los nodos suscriptores los reciben. Esto permite un flujo continuo de datos, ideal para lecturas de sensores o comandos de movimiento.

Por otro lado, los servicios ofrecen una comunicación sincrónica entre nodos, permitiendo solicitudes y respuestas únicas, lo cual resulta útil para configurar parámetros del sistema o ejecutar tareas puntuales. Las acciones, diseñadas para tareas prolongadas, combinan características de los tópicos y servicios, ya que permiten enviar un objetivo, recibir retroalimentación y obtener un resultado final. En este proyecto, los tópicos gestionan comandos de movimiento y estados de sensores, mientras que los servicios y acciones son fundamentales para la planificación de trayectorias y la ejecución de tareas complejas.

2.3.5 Herramientas de ROS 2 Relevantes para el Proyecto

En el contexto de este proyecto, varias herramientas de ROS 2 son fundamentales. El formato URDF (Unified Robot Description Format) se utiliza para describir la estructura física del UR5e y la SoftHand en un modelo XML, que puede interpretarse en herramientas de simulación y control. MoveIt es una plataforma avanzada para la planificación de movimientos y el control de manipuladores, que incluye algoritmos para cinemática inversa, detección de colisiones y generación de trayectorias. RViz permite la visualización tridimensional de configuraciones y trayectorias, lo que facilita la validación del sistema. Adicionalmente, ROS 2 incluye paquetes específicos como `ros2_control` y `gazebo_ros2_control`, que posibilitan el control de bajo nivel en tiempo real.

2.3.6 Aplicaciones de ROS 2 en el Proyecto

ROS 2 desempeña un papel crucial en la integración del UR5e y la SoftHand. La comunicación entre nodos asegura la coordinación precisa entre los controladores del manipulador y la mano robótica. MoveIt y RViz permiten planificar y validar trayectorias antes de su implementación física, mientras que URDF garantiza una representación precisa del sistema. Además, la modularidad de ROS 2 facilita la integración de sensores externos, como cámaras Intel RealSense, para implementar manipulación basada en visión. Esto permite al sistema adaptarse a entornos dinámicos y realizar tareas colaborativas con alta precisión y seguridad.

2.3.7 Ventajas y Desafíos

ROS 2 ofrece ventajas significativas, como escalabilidad, modularidad y una amplia comunidad que contribuye con paquetes y soluciones preexistentes. Su compatibilidad con simulación y validación previa a la implementación física reduce riesgos y tiempos de desarrollo. Sin embargo, también enfrenta desafíos como una curva de aprendizaje pronunciada, la dependencia de hardware específico y los altos requerimientos computacionales de las simulaciones avanzadas. A pesar de estas limitaciones, ROS 2 proporciona una plataforma robusta y versátil que se adapta a las necesidades de proyectos avanzados como este, donde la integración de múltiples componentes es esencial [6].

2.4 Normativas Europeas en Robótica Colaborativa

En el ámbito de la robótica colaborativa, las normativas europeas desempeñan un papel fundamental al establecer criterios de seguridad, diseño e integración para sistemas que interactúan directamente con humanos. Aunque este proyecto se desarrolla en un entorno de investigación y no está estrictamente sujeto a estas regulaciones, considerar estas normativas proporciona un marco teórico valioso para diseñar sistemas seguros, eficientes y compatibles con futuros usos en entornos industriales.

2.4.1 Normativas aplicables

A continuación, se describen las normativas europeas más relevantes para la robótica colaborativa, indicando sus principales requisitos:

- **ISO 10218-1:2011 y ISO 10218-2:2011** [17]: Definen los requisitos de seguridad para robots industriales, abarcando el diseño, instalación y uso de robots. Específicamente, incluyen criterios relacionados con protecciones físicas, monitoreo del espacio de trabajo y sistemas de parada de emergencia.

- **ISO/TS 15066:2016 [18]:** Complementa la ISO 10218, especificando límites de interacción seguros entre humanos y robots, como fuerzas y presiones máximas que el robot puede ejercer sobre un operador en caso de contacto accidental.
- **Directiva de Máquinas 2006/42/CE [19]:** Regula los requisitos esenciales de seguridad aplicables a todas las máquinas, incluidos los sistemas robóticos. Establece principios de diseño seguro, evaluación de riesgos y requisitos técnicos que aseguren la protección de operadores y usuarios.

Estas normativas han sido revisadas durante el desarrollo del proyecto, y su análisis ha influido en el diseño y ejecución de las pruebas realizadas.

2.4.2 Cumplimiento del UR5e con Normativas Europeas

El UR5e, utilizado en este proyecto, cumple con varias normativas internacionales y europeas que refuerzan su idoneidad para aplicaciones en robótica colaborativa. Estas incluyen:

- **ISO 10218-1:2011 e ISO 10218-2:2011:** El UR5e integra sistemas avanzados de seguridad, como detección de colisiones y zonas de trabajo configurables, que cumplen con los requisitos establecidos para el diseño y uso de robots industriales.
- **ISO/TS 15066:2016:** Sus características colaborativas, incluyendo sensores de fuerza/torque y la limitación de fuerza en caso de contacto humano-robot, están alineadas con esta normativa.
- **Directiva de Máquinas 2006/42/CE:** El UR5e incorpora sistemas de seguridad certificados, evaluaciones de riesgos y documentación técnica que garantizan el cumplimiento de esta directiva.
- **Certificación CE:** Garantiza que el UR5e cumple con los estándares europeos para seguridad eléctrica, compatibilidad electromagnética y seguridad mecánica.

Estas características hacen del UR5e una herramienta segura y eficiente para aplicaciones colaborativas, ofreciendo un marco confiable para la integración con la mano robótica SoftHand 2 y otros componentes en este proyecto.

2.4.3 Implicaciones en el Diseño del Proyecto

El uso del UR5e, un robot certificado y diseñado conforme a normativas internacionales, fortalece el enfoque de seguridad y funcionalidad del proyecto. Su cumplimiento con normativas como la ISO 10218 y la Directiva de Máquinas proporciona:

- **Seguridad colaborativa:** El UR5e limita la fuerza aplicada en caso de contacto, asegurando una interacción segura en el entorno de trabajo.

- **Configuración flexible:** Las zonas de trabajo configurables permiten delimitar áreas seguras para pruebas.
- **Fiabilidad técnica:** La certificación CE garantiza la compatibilidad electromagnética y la seguridad eléctrica, factores esenciales para la integración con otros sistemas, como la SoftHand 2.

Estas características del UR5e complementan el diseño del proyecto, alineándolo parcialmente con las normativas europeas y facilitando una futura transición a entornos industriales.

2.4.4 Uso de las normativas en el proyecto

El lector debe entender que las normativas fueron analizadas en detalle para garantizar que los conceptos clave relacionados con la seguridad y la interacción humano-robot se integraran en el diseño del proyecto. Esto incluye:

- Identificación de riesgos potenciales en trayectorias y planificación de movimientos.
- Consideración de límites seguros de interacción definidos por la ISO/TS 15066.
- Adaptación de un enfoque de diseño seguro, de acuerdo con los principios de la Directiva de Máquinas.

Aunque el proyecto no busca cumplir completamente estas normativas, las mismas sirvieron como guía para estructurar un sistema que garantice un nivel de seguridad adecuado para pruebas experimentales y que pueda adaptarse a futuras aplicaciones prácticas.

2.4.5 Relevancia para futuras aplicaciones

Considerar estas normativas en el diseño y validación del sistema no solo fortalece la seguridad y confiabilidad del proyecto, sino que también facilita una eventual transición hacia aplicaciones prácticas en entornos industriales. Este enfoque asegura que las soluciones propuestas puedan adaptarse a contextos reales, cumpliendo con los estándares regulatorios requeridos, lo que resulta esencial para su implementación en escenarios colaborativos donde la interacción humano-robot es crítica.

Capítulo 3

Metodología

El desarrollo del presente proyecto se fundamenta en la metodología propuesta por K. Ulrich y S. Eppinger en su libro *Diseño y Desarrollo de Productos* [20]. A su vez esta metodología ofrece herramientas sistemáticas y efectivas para identificar y clasificar las necesidades del cliente, facilitando la toma de decisiones en el diseño y desarrollo de soluciones innovadoras. Por lo tanto su enfoque estructurado es clave para garantizar un proceso ordenado, orientado a resultados, y adaptado a las metas específicas del proyecto.

3.1 Descripción General

La metodología de Ulrich-Eppinger, cuyas etapas se muestran en la figura 3.1, está principalmente orientada al desarrollo de productos. Sin embargo, en este caso, se adapta a un proyecto de carácter investigativo, donde los principios del desarrollo de productos son igualmente aplicables. En este contexto, los investigadores del GII actúan como clientes del proyecto, permitiendo utilizar este enfoque para identificar necesidades, priorizarlas y orientar las decisiones clave hacia el cumplimiento de los objetivos investigativos establecidos.

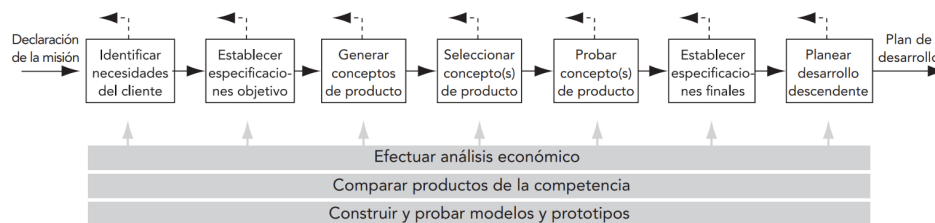


Figura 3.1: Etapas de desarrollo de producto según la metodología Ulrich-Eppinger [20]

La metodología propuesta por Ulrich y Eppinger se divide en seis etapas principales, cada una de las cuales aborda aspectos específicos del desarrollo de productos. Estas etapas son:

- **Identificación de Necesidades:** La primera etapa se centra en la identificación de necesidades, donde se documentan cuidadosamente los requerimientos del sistema

integrado UR5e-SoftHand 2, teniendo en cuenta las aplicaciones objetivo. En esta fase, se definen las tareas específicas que el sistema debe realizar, tales como la manipulación de objetos delicados y la realización de tareas colaborativas. Para lograrlo, se realizan consultas con expertos en robótica y usuarios finales con el fin de obtener información pertinente. Posteriormente, las necesidades identificadas se organizan jerárquicamente según su relevancia para la funcionalidad y el desempeño del sistema.

- **Establecimiento de Especificaciones Técnicas:** En esta etapa, las necesidades se traducen en especificaciones técnicas claras. Este proceso incluye la definición de especificaciones del UR5e, como su precisión, alcance, capacidad de carga y requisitos cinemáticos y dinámicos. Asimismo, se establecen las especificaciones de la SoftHand, enfocándose en su adaptabilidad, fuerza de agarre y compatibilidad con el manipulador. Cada especificación se asocia con métricas y valores objetivo, tanto marginales como ideales, lo que asegura un marco técnico para guiar el desarrollo.
- **Generación y Selección de Conceptos:** La generación y selección de conceptos representa una fase crítica en la metodología. Durante esta etapa, el problema principal se descompone en subproblemas clave, como la sincronización de movimientos y la configuración de controladores. Además se exploran soluciones mediante búsquedas internas y externas, las cuales incluyen la revisión de proyectos similares y el desarrollo de propuestas innovadoras basadas en simulaciones y pruebas experimentales. Estas soluciones se implementan y validan de manera iterativa, resolviendo cada subproblema de forma enfocada antes de avanzar al siguiente.
- **Prueba del Concepto:** La validación de los conceptos generados se lleva a cabo en la etapa de prueba del concepto. En esta fase, se realizan simulaciones iniciales en herramientas como MoveIt y RViz para verificar la planificación de trayectorias, seguidas de pruebas físicas en hardware real para evaluar la integración y sincronización del sistema. A su vez los resultados de estas pruebas permiten realizar ajustes necesarios en los parámetros de los controladores y las configuraciones del sistema, asegurando que el sistema cumple con las especificaciones establecidas.
- **Establecimiento de Especificaciones Finales:** Antes de concluir, se establecen las especificaciones finales del sistema, considerando las limitaciones prácticas identificadas durante el desarrollo. Esto incluye restricciones del modelado técnico, diferencias entre simulación y desempeño físico, y la evaluación del rendimiento bajo diferentes escenarios. Con base en estos factores, se refinan las especificaciones finales para garantizar que el sistema cumple con las expectativas funcionales.
- **Planificación y Seguimiento del Proyecto:** Finalmente, aunque el presente proyecto no tiene un enfoque comercial, se realiza una planificación para el seguimiento de la investigación. Lo anterior incluye la definición de pasos futuros, como la exploración de nuevas aplicaciones o la mejora de algoritmos de control, así como la documentación de los resultados obtenidos para servir de base en investigaciones posteriores.

3.2 Identificación de necesidades

La identificación de las necesidades del proyecto se llevó a cabo mediante reuniones con los investigadores del GII: Dr. José Antonio Becerra, Dr. Alejandro Romero y Dr. Félix Orjales. Durante estas sesiones, se analizaron los objetivos fundamentales del proyecto y se definieron los aspectos críticos necesarios para garantizar el éxito de la integración de la mano robótica con el robot UR5.

El objetivo de esta etapa fue establecer un conjunto exhaustivo de necesidades iniciales que sirvan como base para el diseño y desarrollo del sistema. Por lo tanto para asegurar la relevancia y viabilidad de estas necesidades, se aplicó un proceso de filtrado riguroso que permitió diferenciar entre requisitos esenciales, características deseables y elementos opcionales. Además únicamente se incluyeron aquellas necesidades consideradas indispensables para cumplir con los objetivos del proyecto, mientras que los deseos o sugerencias no esenciales fueron relegados o documentados para futuras iteraciones.

Ahora bien las necesidades identificadas abarcan tanto las limitaciones técnicas como las expectativas funcionales del sistema integrado. Además este enfoque asegura que el desarrollo del proyecto se mantenga alineado con los objetivos establecidos, evitando desviaciones hacia características no críticas. Las necesidades se sintetizan en la Tabla 3.2, junto con su importancia relativa en una escala de 1 a 5, donde 5 indica máxima prioridad. En la tabla 3.1 se presenta la escala de importancia utilizada para clasificar las necesidades del proyecto.

Tabla 3.1: Escala de importancia de las necesidades del proyecto

Valor	Descripción
1	Baja prioridad: Requisito no esencial cuya ausencia no afecta la funcionalidad principal del sistema. Puede considerarse para futuras mejoras.
2	Moderadamente baja prioridad: Característica deseable que aporta valor, pero su omisión no compromete significativamente el desempeño del sistema.
3	Prioridad media: Necesidad importante que contribuye a la eficiencia y robustez del sistema. No es crítica, pero su implementación mejora el rendimiento general.
4	Alta prioridad: Requisito esencial cuya implementación es altamente recomendada. Su ausencia podría comprometer aspectos clave del sistema.
5	Máxima prioridad: Necesidad fundamental e imprescindible para el éxito del proyecto. Su omisión afectaría gravemente la funcionalidad, integración o viabilidad del sistema.

Tabla 3.2: Necesidades del proyecto y su importancia

ID	Enunciado de la Necesidad	Importancia (1-5)
1	La solución utiliza la versión de ROS 2 para la integración del sistema.	5
2	La solución emplea el framework MoveIt para la planificación de trayectorias.	5
3	La solución es compatible con la PC asignada para el control del robot UR5.	4
4	La solución debe ser capaz de realizar tareas de manipulación en un entorno controlado.	5
5	La solución debe permitir un control eficiente y preciso de la mano robótica para manipular diferentes objetos de geometrías y tamaños variados.	5
6	La integración debe garantizar la comunicación entre los controladores del UR5 y la mano robótica.	5
7	El sistema debe ser escalable y adaptable para permitir su uso en diversos experimentos y tareas de manipulación.	4
8	La interfaz gráfica debe permitir al usuario definir tareas de pick and place de forma sencilla.	3
9	El sistema debe incluir una API que permita el control desde scripts en Python además de la interfaz gráfica.	4
10	La solución debe contar con documentación técnica detallada para facilitar el mantenimiento y futuras mejoras.	3

3.2.1 Priorización de necesidades

La priorización de las necesidades se llevó a cabo utilizando un enfoque sistemático, asegurando que los aspectos más críticos recibieran la atención adecuada durante el desarrollo del proyecto. Este proceso permitió clasificar las necesidades en tres categorías principales:

- **Necesidades esenciales:** Requisitos imprescindibles para el funcionamiento básico del sistema, como la compatibilidad con ROS 2 y la capacidad de ejecutar tareas con precisión.
- **Características deseables:** Elementos que mejoran la experiencia del usuario o aumentan la flexibilidad del sistema, pero que no son estrictamente necesarios para la primera fase del desarrollo.
- **Características opcionales:** Funcionalidades que agregan valor al proyecto, pero no son críticas ni urgentes y pueden considerarse para futuras iteraciones.

Además cada necesidad se evaluó en función de su impacto en la funcionalidad central del sistema, su alineación con los objetivos generales del proyecto y la viabilidad técnica dentro

del cronograma y los recursos disponibles. A continuación, se asignó una puntuación de prioridad en una escala del 1 al 5:

- **5:** Necesidad crítica que debe abordarse de inmediato.
- **4:** Necesidad importante que debería atenderse en esta etapa, aunque podría postergarse en caso necesario.
- **3 o menor:** Necesidad no crítica que puede diferirse o excluirse sin afectar significativamente los resultados del proyecto.

El proceso también incluyó el filtrado de deseos o características sugeridas que no contribuyen directamente al éxito del proyecto. Estas sugerencias fueron documentadas para su consideración en futuras iteraciones, mientras que los recursos disponibles se asignaron exclusivamente a las necesidades esenciales.

Finalmente, los resultados de esta priorización fueron revisados y validados por expertos técnicos para garantizar que las necesidades críticas estuvieran adecuadamente reflejadas. Este enfoque permitió establecer una base sólida para el diseño y desarrollo del sistema, asegurando que los esfuerzos estuvieran dirigidos hacia los objetivos de mayor impacto y viabilidad.

3.3 Generación de métricas

La creación de métricas permite evaluar cuantitativamente el grado de cumplimiento de las necesidades identificadas en la sección 3.2. Estas métricas se definieron considerando indicadores claros, medibles y alineados con los objetivos del proyecto, asegurando que cada necesidad cuente con al menos una métrica asociada.

3.3.1 Métricas propuestas

En la Tabla 3.3, se presenta la relación entre las necesidades identificadas y sus respectivas métricas, incluyendo las unidades de medida, los criterios de aceptación, así como los valores ideales y marginales.

Tabla 3.3: Métricas asociadas a las necesidades del proyecto.

ID Necesidad	ID Métrica	Métrica	Unidad de medida	Valor ideal	Valor marginal
1	M1	Verificación de instalación y funcionamiento en ROS 2 Humble.	Compatibilidad (%)	100%	75%
2	M2	Tiempo promedio de planificación de trayectorias.	Segundos (<i>s</i>)	< 1 s	< 2 s
3	M3	Estabilidad medida en tiempo de ejecución sin errores del control del UR5.	Horas (<i>h</i>)	> 8 h	> 4 h
4	M4	Precisión en la posición final del objeto manipulado.	Centímetros (<i>cm</i>)	< 0.2 cm	< 0.5 cm
5	M5	Cantidad de objetos de diferentes geometrías y tamaños manipulados exitosamente.	Cantidad	10 objetos	5 objetos
6	M6	Latencia promedio en la comunicación entre controladores.	Milisegundos (<i>ms</i>)	< 20 ms	< 50 ms
7	M7	Cantidad de experimentos de manipulación diferentes que el sistema puede realizar sin cambios estructurales.	Cantidad	10 experimentos	5 experimentos
8	M8	Tiempo promedio para definir una tarea en la interfaz.	Minutos (<i>min</i>)	< 2 min	< 5 min
9	M9	Número de funciones implementadas en la API de Python.	Cantidad	> 15 funciones	> 10 funciones
10	M10	Porcentaje de cobertura en la documentación técnica.	Porcentaje (%)	> 95%	> 90%

3.3.2 Justificación de las métricas

Las métricas propuestas se diseñaron para evaluar de manera integral el cumplimiento de las necesidades identificadas, asegurando la alineación con los objetivos específicos del proyecto. A su vez, cada métrica corresponde directamente a una necesidad del sistema, reflejando aspectos esenciales como compatibilidad, rendimiento, adaptabilidad y usabilidad.

La métrica M1 evalúa la compatibilidad del sistema con ROS 2 Humble, lo cual es un requisito crítico para la integración de los componentes robóticos. La definición detallada de los criterios de compatibilidad, así como el procedimiento de evaluación utilizado, se encuentran explicados en el Anexo A. Por su parte, la métrica M2 mide el tiempo promedio necesario para planificar trayectorias, garantizando un desempeño eficiente del sistema en tareas de manipulación.

Ahora bien, M3 y M4 abordan aspectos técnicos clave como la estabilidad y la precisión. A su vez, M3 mide la estabilidad del sistema en términos del tiempo de ejecución sin errores del control del UR5, asegurando que el sistema sea confiable en operaciones prolongadas. Por otro lado, M4 evalúa la precisión en la posición final de los objetos manipulados, garantizando que el sistema cumpla con los estándares de exactitud requeridos.

En términos de funcionalidad específica, el M5 mide la cantidad de objetos de diferentes geometrías y tamaños que la mano robótica puede manipular exitosamente, reflejando su versatilidad y capacidad de adaptación. Para definir qué constituye un "objeto de diferentes geometrías y tamaños", se realizó un análisis detallado cuyos cálculos y resultados se encuentran documentados en el Anexo B. Este análisis consideró parámetros como dimensiones,

formas geométricas y características físicas relevantes de los objetos seleccionados, asegurando que la métrica evalúe adecuadamente la capacidad del sistema para manejar variedad en las tareas de manipulación.

M6 evalúa la latencia promedio en la comunicación entre los controladores del UR5 y la mano robótica, asegurando una interacción fluida y sincronizada. Además, la adaptabilidad del sistema se evalúa mediante la métrica M7, que mide la cantidad de experimentos de manipulación que el sistema puede realizar sin cambios estructurales, destacando su capacidad de ser utilizado en diversos contextos experimentales. Por otro lado, M8 se centra en la facilidad de uso de la interfaz gráfica para definir tareas complejas como pick and place.

M9 asegura que el sistema disponga de una API funcional en Python, facilitando su integración en flujos de trabajo programáticos y promoviendo la automatización de tareas. Finalmente, M10 evalúa la calidad de la documentación técnica, garantizando que el sistema sea comprensible, mantenible y escalable a largo plazo.

Con lo anterior, cada métrica cuenta con valores ideales y marginales que reflejan los objetivos esperados del proyecto. Estos criterios de aceptación permiten identificar áreas de mejora y establecer prioridades en el desarrollo del sistema. En conjunto, las métricas proporcionan un marco robusto para la evaluación cuantitativa del éxito del proyecto.

3.4 Generación de Conceptos

La etapa de generación de conceptos, fundamentada en la metodología propuesta por Karl T. Ulrich y Steven D. Eppinger, tiene como objetivo articular un conjunto amplio de soluciones preliminares que respondan de manera eficaz y eficiente a los requerimientos establecidos durante las fases iniciales del proyecto. Este proceso se erige como un pilar esencial para explorar enfoques innovadores, fomentar la creatividad y seleccionar estrategias viables que aborden los problemas técnicos y funcionales identificados [20].

3.4.1 Descomposición de Subproblemas

La descomposición de subproblemas constituye una técnica fundamental dentro de la metodología de generación de conceptos, al descomponer problemas complejos en componentes específicos y manejables. Además este enfoque permite un análisis exhaustivo de los aspectos críticos del proyecto, fomentando el desarrollo de soluciones personalizadas que aseguren la coherencia global del sistema integrado [20].

En el contexto del sistema UR5e-SoftHand 2, esta metodología permitió identificar áreas funcionales y operativas con mayores desafíos técnicos. Los subproblemas principales identificados incluyen:

Sincronización de movimientos. Garantizar la coordinación precisa entre el manipulador UR5e y la SoftHand es esencial para las tareas de manipulación. Este subproblema exige el diseño de algoritmos avanzados que permitan la generación de trayectorias ajustadas

dinámicamente en función de las condiciones del entorno.

Integración de controladores. Este subproblema aborda la comunicación eficiente entre los sistemas, mediante el uso de controladores como el `scaled_joint_trajectory_controller` y el controlador de sinergias, garantizando su interoperabilidad.

Detección y manipulación basadas en sensores. La incorporación de sensores de fuerza/par y cámaras de visión artificial refuerza la capacidad del sistema para manipular objetos de formas y texturas irregulares, adaptándose a entornos dinámicos.

Interacción humano-robot. Diseñar estrategias que aseguren la seguridad y la usabilidad en la interacción humano-robot implica el desarrollo de controladores de impedancia y la implementación de interfaces intuitivas.

Integración mecánica y eléctrica. Se identificaron dos aspectos críticos: la integración mecánica, que considera el diseño de adaptadores robustos para la conexión física, y la integración eléctrica, enfocada en la sincronización de sistemas de alimentación y comunicación de datos.

El análisis detallado de estos subproblemas, utilizando herramientas como diagramas funcionales y descomposición jerárquica (ver figura 3.2), permitió priorizar las áreas críticas, estableciendo una base sólida para la implementación y validación de soluciones, optimizando recursos y reduciendo riesgos [20].

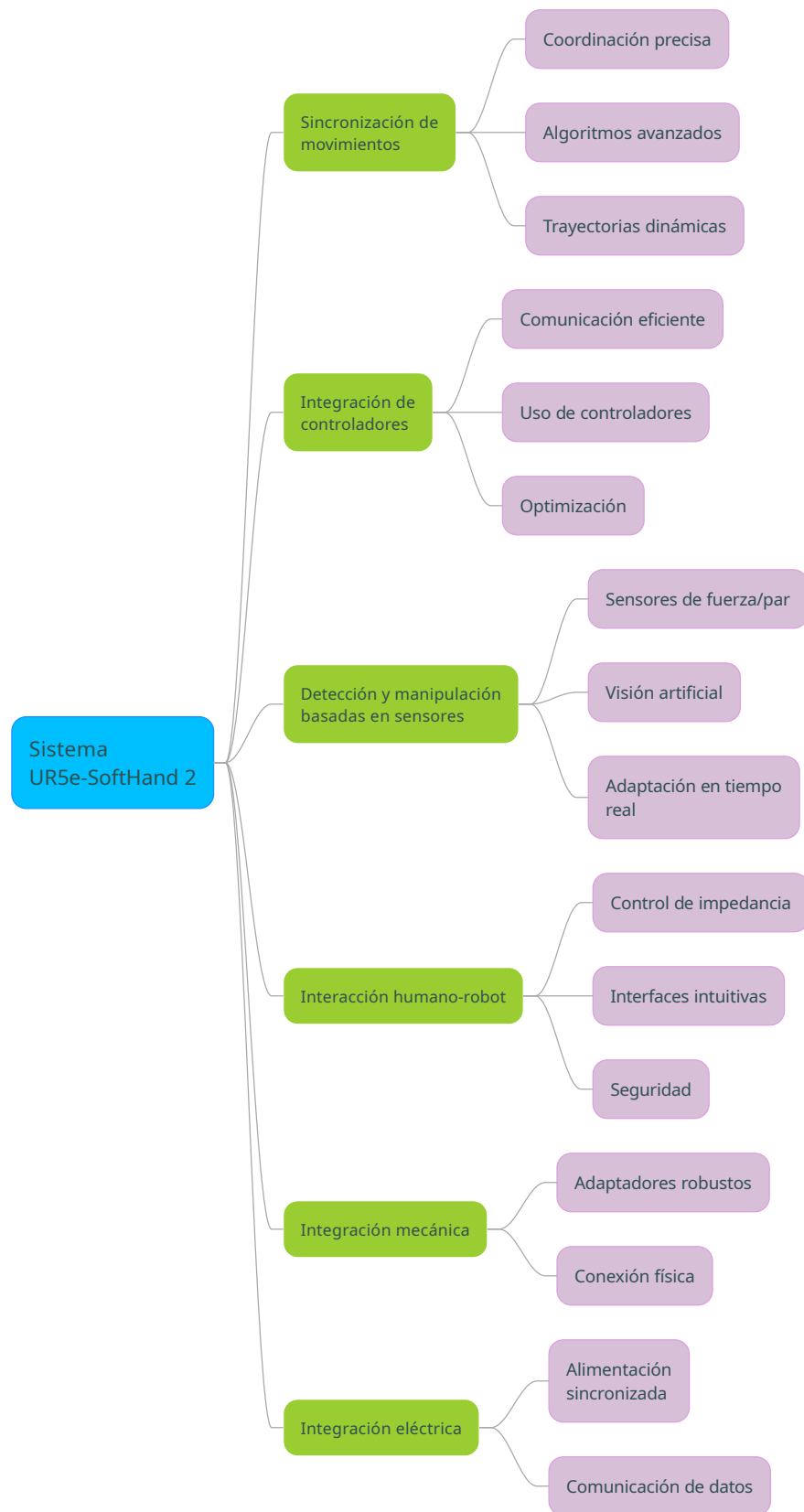


Figura 3.2: Descomposición de subproblemas en el sistema UR5e-SoftHand 2. Elaboración propia.

3.4.2 Técnicas de Generación de Conceptos

Para abordar los subproblemas identificados, se emplearon técnicas estructuradas que equilibran la innovación con la rigurosidad técnica. Entre las técnicas aplicadas destacan:

Búsquedas internas. Se organizaron sesiones de lluvia de ideas (*brainstorming*) con el equipo de trabajo, fomentando la asociación libre de ideas basada en la experiencia colectiva. Este enfoque permitió identificar soluciones originales y prácticas para los desafíos técnicos.

Búsquedas externas. La revisión exhaustiva de literatura científica, patentes y casos de estudio relacionados permitió identificar enfoques exitosos y adaptar soluciones probadas al contexto del proyecto [12].

Resolución creativa de problemas. Herramientas analíticas como el análisis morfológico y los diagramas funcionales se utilizaron para estructurar, combinar y evaluar diferentes enfoques. Estas herramientas facilitaron la generación de soluciones robustas y alineadas con los objetivos del sistema.

3.4.3 Conceptos Generados

Con la descomposición del problema en subproblemas, se generaron conceptos generales para abordar cada una de las áreas identificadas. A su vez estos conceptos constituyen las bases fundamentales que guían el desarrollo de las soluciones específicas presentadas en la sección de propuesta de diseño. A continuación, se describen las ideas principales generadas durante esta etapa:

Sincronización de movimientos

Garantizar la sincronización precisa de los movimientos requiere un enfoque integral que combine algoritmos avanzados de control y sistemas adaptativos para ajustar las trayectorias en tiempo real. Algunos ejemplos de soluciones similares incluyen el uso de algoritmos de planificación dinámica implementados en ROS 2, capaces de calcular trayectorias basadas en datos provenientes de sensores de movimiento y del entorno. Este concepto integra información sensorial con modelos predictivos para mantener fluidez y consistencia en las tareas de manipulación.

Integración de controladores

La integración modular de controladores asegura la comunicación eficiente entre los subsistemas. El uso de middleware como ROS 2 proporciona herramientas y protocolos estandarizados para la interoperabilidad entre componentes heterogéneos. Este concepto incluye la utilización de middleware que facilite la escalabilidad y flexibilidad del sistema.

Detección y manipulación basadas en sensores

El diseño conceptual incluye una red de sensores que proporciona información precisa sobre las condiciones del entorno y las propiedades de los objetos manipulados. A su vez los sensores considerados abarcan cámaras de visión artificial para detectar formas y texturas, sensores de fuerza/par para medir interacciones mecánicas, y sensores táctiles que identifican irregularidades en las superficies de los objetos. Estos datos se interpretan mediante algoritmos de aprendizaje automático, mejorando la adaptabilidad del sistema a escenarios dinámicos.

Interacción humano-robot

Las soluciones conceptuales para la interacción humano-robot se centran en la seguridad y la usabilidad. Las interfaces gráficas intuitivas, configuraciones accesibles y retroalimentación en tiempo real facilitan la toma de decisiones del operador. En términos de seguridad, se priorizan controladores de impedancia que limitan la fuerza ejercida durante las interacciones, sensores de proximidad y contacto, y mecanismos de parada automática para mitigar riesgos. Estas estrategias aseguran interacciones seguras y eficientes entre el sistema robótico y los operadores.

Integración mecánica y eléctrica.

La integración eléctrica se diseña para garantizar una alimentación separada entre la mano robótica y el robot, evitando interferencias o daños mutuos derivados de fluctuaciones o sobrecargas eléctricas. Para ello, se consideran fuentes de alimentación aisladas que aseguren niveles de voltaje y corriente estables en cada subsistema. Además, se implementan sistemas de protección adicionales, como fusibles y supresores de picos, para mejorar la fiabilidad y la seguridad del sistema en condiciones operativas variables.

En la integración mecánica, se proponen adaptadores fabricados con materiales de alta resistencia, diseñados para proporcionar conexiones físicas robustas y precisas entre los componentes. Estos adaptadores garantizan una transferencia eficiente de fuerzas y movimientos, minimizando desajustes o fallos estructurales durante la operación del sistema.

Con base en estas ideas, se desarrolló el diagrama general que describe la estrategia de integración del sistema, tal como se muestra en la figura 3.3.

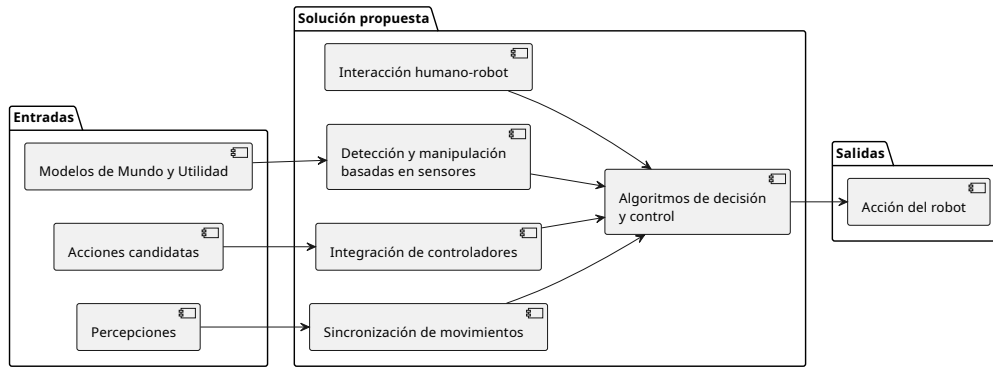


Figura 3.3: Diagrama general de la estrategia de integración. Elaboración propia.

3.4.4 Generación de soluciones para los subproblemas

La generación de soluciones para los subproblemas identificados consideró enfoques diversos que cubren aspectos técnicos, económicos y de implementación práctica. A continuación, se presentan las opciones generadas para cada subproblema:

- **Sincronización de movimientos:**

1. Uso de los algoritmos propios de *MoveIt* para la generación de trayectorias sincronizadas.
2. Desarrollo de un algoritmo de planificación propio basado en controladores predictivos.
3. Uso de un sistema de aprendizaje por refuerzo para optimizar la sincronización.

- **Integración de controladores:**

1. Desarrollo de un servicio/script intermedio para adaptar la comunicación entre los controladores.
2. Modificación directa del código del controlador de la *SoftHand* para adaptarlo al *UR5e*.
3. Uso de una interfaz estándar en *ROS 2* para unificar los mensajes entre controladores.

- **Detección y manipulación basadas en sensores:**

1. Uso de la cámara *Intel RealSense L515* para la detección y evaluación del entorno.
2. Incorporación de sensores de fuerza/par en la muñeca del *UR5e*.
3. Uso de una cámara *RGB-D* comercial para el reconocimiento de objetos.

- **Interacción humano-robot:**

1. Uso de los sensores integrados en el *UR5e* para garantizar la seguridad del usuario.

2. Desarrollo de un controlador de impedancia personalizado para limitar la fuerza ejercida.
3. Implementación de interfaces gráficas intuitivas basadas en MoveIt.

- **Integración mecánica y eléctrica:**

1. Uso de un adaptador mecánico comercial para la conexión física.
2. Diseño de un adaptador mecánico personalizado.
3. Alimentación eléctrica independiente mediante un convertidor AC-DC para la SoftHand.
4. Uso de una fuente de alimentación unificada para ambos dispositivos.

3.4.5 Criterios de evaluación

Las soluciones propuestas fueron evaluadas en función de los siguientes criterios, utilizando una escala de 0 (peor desempeño) a 5 (mejor desempeño):

- **Facilidad de implementación:** Grado de simplicidad y rapidez con la que la solución puede integrarse al sistema existente.
- **Costo:** Recursos económicos necesarios para implementar la solución.
- **Robustez:** Capacidad de la solución para operar de manera estable y confiable en diferentes escenarios.
- **Escalabilidad:** Posibilidad de adaptar la solución a futuros desarrollos o sistemas más complejos.

3.4.6 Evaluación de soluciones

La evaluación de las soluciones propuestas se presenta en la siguiente matriz:

Tabla 3.4: Evaluación de soluciones propuestas.

Subproblema	Solución	Facilidad (5)	Costo (5)	Robustez (5)	Escalabilidad (5)	Total
Sincronización	Algoritmos de MoveIt	5	5	4	4	18
	Algoritmo propio	3	4	5	5	17
	Aprendizaje por refuerzo	2	2	5	5	14
Integración	Servicio/script intermedio	5	5	4	5	19
	Modificación del controlador	4	4	4	4	16
	Interfaz estándar ROS 2	3	5	3	5	16
Detección	Cámara L515	5	5	4	5	19
	Sensores de fuerza/par	4	3	4	4	15
	Cámara RGB-D	3	4	4	4	15
Interacción	Sensores del UR5e y MoveIt	5	5	4	4	18
	Controlador de impedancia	4	3	5	4	16
	Interfaz personalizada	3	4	3	5	15
Integración Mecánica	Adaptador comercial	5	2	4	4	15
	Adaptador personalizado	3	4	5	4	16
Integración Eléctrica	Alimentación independiente con AC-DC	5	5	4	5	19
	Alimentación unificada	4	3	3	5	15

Las soluciones seleccionadas obtuvieron las mayores puntuaciones en la evaluación, garantizando un equilibrio entre facilidad de implementación, costo, robustez y escalabilidad.

3.4.7 Selección de soluciones

Para seleccionar las soluciones más adecuadas, se creó la siguiente tabla que identifica las alternativas elegidas para cada subproblema:

Tabla 3.5: Selección de soluciones para los subproblemas.

Subproblema	Solución seleccionada
Sincronización de movimientos	Uso de los algoritmos propios de MoveIt .
Integración de controladores	Desarrollo de un servicio/script intermedio para adaptar la comunicación.
Detección y manipulación	Uso de la cámara Intel RealSense L515.
Interacción humano-robot	Uso de los sensores del UR5e y la interfaz gráfica de MoveIt .
Integración mecánica y eléctrica	Uso de un adaptador mecánico comercial y alimentación eléctrica independiente mediante un convertidor AC-DC. ^a

^a Se utilizó un adaptador mecánico comercial previamente adquirido en un proyecto anterior, cuya compatibilidad fue validada durante el desarrollo del presente sistema.

3.5 Enfoque de solución

Para abordar la integración del brazo robótico UR5e con la mano robótica SoftHand 2, se ha establecido un plan de acción dividido en seis fases principales: gestión del proyecto, diseño e integración mecánica, diseño e integración eléctrica, desarrollo de software, pruebas e iteraciones, documentación y entrega. Este enfoque permite una gestión ordenada y eficiente de las actividades, además de posibilitar la realización simultánea de múltiples fases según las necesidades del proyecto.

En la fase de **gestión del proyecto**, se definen los requerimientos funcionales y técnicos del sistema, además de identificar los recursos y componentes necesarios. Asimismo, se elabora un cronograma detallado y un presupuesto ajustado que permitan controlar el progreso y los costos asociados.

La **fase de diseño e integración mecánica** se enfoca en el modelado CAD del sistema, la selección de actuadores y sensores adecuados, y la fabricación y ensamblaje de las piezas mecánicas. Una vez completado el ensamblaje, se realiza la validación física para garantizar el cumplimiento de las especificaciones técnicas.

Por su parte, la **fase de diseño e integración eléctrica** incluye el diseño del esquema de cableado, la selección de controladores y fuentes de alimentación, así como la conexión de actuadores y sensores. Esta etapa culmina con pruebas de validación eléctrica que aseguran el correcto funcionamiento de todos los subsistemas involucrados.

En la **fase de desarrollo de software**, se configuran los entornos de trabajo en Linux, se instalan en ROS 2 y sus dependencias, y se integra el hardware, incluyendo el UR5e, la SoftHand 2 y la cámara RealSense. Adicionalmente, se configuran los parámetros de MoveIt y se desarrollan los controladores necesarios para la operación del sistema.

La **fase de pruebas e iteraciones** permite evaluar el desempeño inicial del sistema mediante pruebas funcionales, seguido de un proceso de diagnóstico y resolución de problemas. Esta etapa incluye la validación de componentes, la optimización de algoritmos y las pruebas finales para garantizar que el sistema cumple con los objetivos planteados.

Finalmente, la **fase de documentación y entrega** asegura que toda la información técnica y operativa del proyecto quede registrada. Esto incluye la elaboración de informes técnicos, manuales de usuario y una revisión final antes de la entrega del proyecto.

La descomposición de estas fases y sus actividades específicas se presenta de forma visual en el mapa mental de la Figura 3.2, el cual organiza las tareas clave y resalta las relaciones entre los diferentes componentes del proyecto.



Figura 3.4: Mapa mental del enfoque de solución. Elaboración propia.

Capítulo 4

Propuesta de Diseño

En este capítulo se presenta el diseño conceptual del sistema integrado UR5e-SoftHand 2, basado en principios teóricos y desarrollado para su implementación en un entorno controlado. El diseño se estructura de manera metodológica, garantizando la integración eficiente de todos los componentes y asegurando el cumplimiento de las especificaciones establecidas en las etapas iniciales del proyecto.

La propuesta abarca la arquitectura general del sistema, los procesos de planificación y ejecución de trayectorias, la integración de sensores avanzados y las estrategias de interacción humano-robot. Estas áreas han sido diseñadas con un enfoque en la modularidad, escalabilidad y adaptabilidad, permitiendo que el sistema responda de manera eficiente a escenarios operativos dinámicos y complejos. Ahora bien el objetivo del diseño no se limita únicamente al cumplimiento de los requisitos iniciales, sino que también busca proporcionar un marco práctico y flexible que contribuya al desarrollo de soluciones en robótica colaborativa para aplicaciones reales.

4.1 Arquitectura General del Sistema

El sistema propuesto está conformado por el manipulador robótico UR5e, la mano robótica SoftHand 2, una cámara Intel RealSense L515 y una PC central que actúa como núcleo operativo. La PC central se encarga de coordinar la planificación, el procesamiento de datos y la comunicación con los controladores de los dispositivos, los cuales están integrados en sus respectivos control boxes. Este diseño distribuye de manera eficiente las funciones de control y procesamiento, optimizando el desempeño operativo del sistema.

La figura 4.1 muestra la interacción entre los componentes principales del sistema. La PC central, equipada con Ubuntu/Linux y ROS 2 Humble, se encarga de gestionar la planificación de trayectorias, el procesamiento de datos capturados por la cámara y otros sensores, y de enviar comandos a los dispositivos robóticos a través de interfaces de comunicación estandarizadas.

La cámara Intel RealSense L515 captura datos del entorno y proporciona información en tiempo real al módulo de procesamiento de datos. Esta información se utiliza en la planificación de trayectorias, permitiendo que los dispositivos se adapten a cambios en su entorno operativo. Por otro lado el módulo de procesamiento de datos analiza la información obtenida por los sensores y la cámara, generando datos útiles que son enviados a los módulos de planificación y, posteriormente, a los controladores de los dispositivos.

Cada dispositivo robótico cuenta con su propio control box, que recibe comandos desde la PC central y ejecuta los movimientos. En el caso del UR5e, el control box gestiona el movimiento del manipulador, mientras que el de la SoftHand 2 controla la mano robótica. Esta estructura descentralizada permite una respuesta más eficiente y específica de cada dispositivo.

La interfaz de usuario incluye dos componentes principales. El Teach Pendant permite la interacción manual con el sistema, mientras que MoveIt facilita la planificación de trayectorias, integrando los datos capturados en tiempo real por los sensores y la cámara. Además, este diseño modular permite la ejecución segura y eficiente de tareas en entornos dinámicos, al tiempo que asegura la adaptabilidad del sistema a nuevos escenarios y aplicaciones.

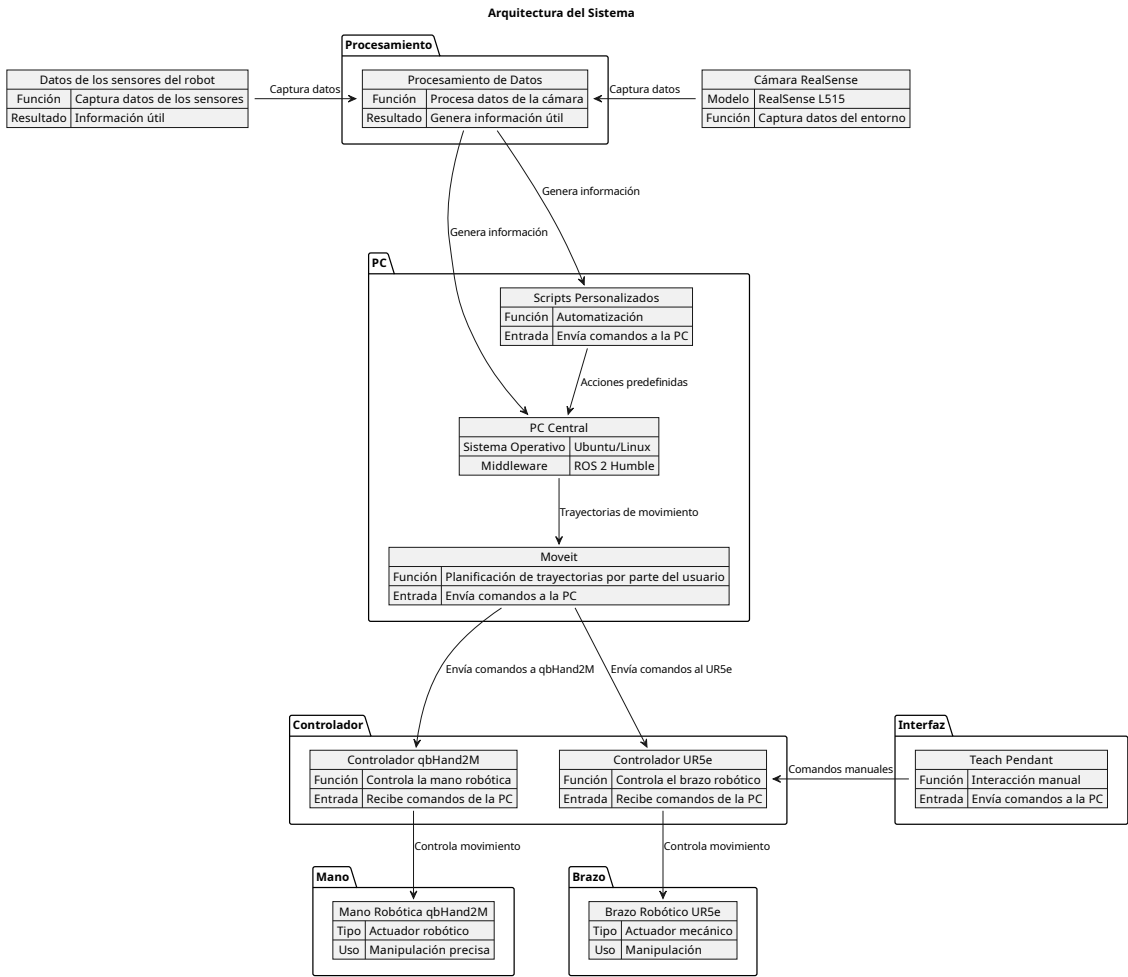


Figura 4.1: Diagrama conceptual de la arquitectura general del sistema desarrollado.

Por otro lado, el sistema está diseñado para maximizar la modularidad y la escalabilidad, lo que facilita su extensión a nuevas funcionalidades y aplicaciones. Además, la comunicación centralizada a través de ROS 2 garantiza la coherencia en el intercambio de información, mientras que la integración de sensores avanzados proporciona al sistema la capacidad de adaptarse dinámicamente a cambios en su entorno operativo.

4.2 Integración Mecánica

La integración mecánica del sistema UR5e-SoftHand 2 asegura la correcta disposición y fijación de los componentes en el espacio de trabajo, garantizando estabilidad, robustez y precisión durante las tareas de manipulación. Además este apartado detalla los elementos estructurales clave y su configuración.

4.2.1 Montaje del Manipulador UR5e

El manipulador robótico UR5e está fijado a una base mecánica estable diseñada para absorber vibraciones. Esta base se encuentra colocada sobre una mesa robusta, cuya superficie actúa como área de trabajo. A su vez la posición del brazo está optimizada para realizar tareas de manipulación dentro de un volumen accesible predefinido.

4.2.2 Integración de la SoftHand 2

La SoftHand 2 se acopla al extremo del UR5e mediante un adaptador mecánico especializado. Inicialmente, se propuso diseñar un adaptador personalizado fabricado en PLA mediante impresión 3D de la figura 4.2, lo que permitiría una solución económica y ajustada a las necesidades específicas del proyecto. Sin embargo, tras una evaluación más detallada, se decidió utilizar un adaptador estándar de aluminio disponible en el laboratorio.

Además el adaptador de aluminio, diseñado específicamente para aplicaciones robóticas, ofrece ventajas significativas en términos de resistencia mecánica y durabilidad en comparación con la propuesta inicial de PLA. Aunque no incluye funciones adicionales como la gestión de cables, su diseño asegura un acoplamiento firme y preciso entre el UR5e y la SoftHand 2. Esto garantiza la estabilidad estructural del sistema y elimina posibles problemas asociados con holguras o flexiones en el punto de conexión.

Ahora bien, la elección del adaptador estándar no solo simplificó el proceso de integración, sino que también proporcionó una solución confiable y adecuada para las exigencias operativas del sistema, asegurando un rendimiento óptimo en tareas de manipulación y minimizando posibles problemas derivados de la mecánica del acoplamiento.

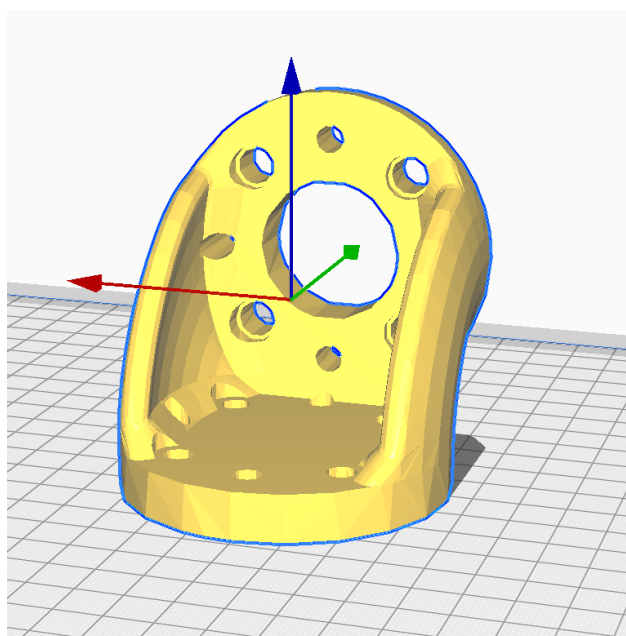


Figura 4.2: Adaptador propuesto en impresión 3D. Elaboración propia.

4.2.3 Mesa de Trabajo

Aunque la mesa no se modela directamente en el entorno virtual, su posición y dimensiones son críticas en el diseño. Tomando en cuenta lo anterior la mesa proporciona una superficie plana y estable para los objetos que el sistema debe manipular. Además, se utiliza una de las esquinas de la mesa como referencia para establecer el origen del sistema de coordenadas del espacio de trabajo.

4.2.4 Montaje de Sensores

La cámara RealSense L515 está montada en una base fija en el techo del entorno de trabajo, figura 4.3, lo que garantiza una vista estable del área de trabajo. Esta configuración permite que el sistema capture información precisa sobre la posición y orientación de los objetos manipulados. Además los sensores de fuerza/par están integrados directamente en las conexiones mecánicas del UR5e, permitiendo una medición precisa de las interacciones físicas durante las tareas.



Figura 4.3: Montaje de la cámara RealSense L515 en el entorno de trabajo. Elaboración propia.

4.3 Integración Eléctrica

La integración eléctrica del sistema está enfocada en garantizar la distribución eficiente de energía, la comunicación confiable entre los componentes y la protección de los mismos frente a posibles fallos eléctricos. A su vez, esta sección detalla los elementos eléctricos esenciales y su configuración.

4.3.1 Fuente de Alimentación

El sistema emplea dos fuentes de alimentación independientes, diseñadas para satisfacer los requerimientos específicos de cada componente principal y garantizar un funcionamiento estable y seguro:

La primera fuente de alimentación es proporcionada por el fabricante del manipulador UR5e. Esta fuente cumple con los estándares eléctricos requeridos para operar el manipulador, asegurando un suministro estable a sus controladores y motores. Al estar integrada en el sistema original del UR5e, no requiere modificaciones adicionales.

La segunda fuente de alimentación está dedicada exclusivamente a la SoftHand 2, figura 4.4, debido a las necesidades específicas de este componente. Ahora bien de acuerdo con sus características eléctricas, la SoftHand 2 requiere un voltaje de alimentación nominal de 24 VDC, con un rango operativo de 23.1 VDC a 24.8 VDC. Además su consumo energético nominal es de 22 W, con un máximo de 38 W en condiciones de alta demanda. Además, la corriente nominal requerida es de 0.5 A, con posibles picos iniciales mayores, lo que justifica el uso de una fuente independiente para evitar interferencias o caídas de voltaje que puedan afectar el rendimiento del sistema.

La fuente de la SoftHand 2 está equipada con sistemas de protección, como reguladores de voltaje y fusibles, que aseguran la estabilidad del suministro y previenen daños por sobrecargas o fluctuaciones de corriente. Por lo tanto, esta independencia en la alimentación eléctrica es crucial para mantener la integridad del sistema y garantizar el funcionamiento óptimo tanto del brazo robótico como de la mano.

Por otro lado, la separación de las fuentes no solo permite una mayor flexibilidad en la operación, sino que también protege ambos subsistemas al evitar posibles interferencias eléctricas, asegurando así la robustez y confiabilidad del sistema integrado.



Figura 4.4: Fuente de alimentación de la SoftHand 2. Elaboración propia.

4.3.2 Controladores

Cada componente principal cuenta con un controlador dedicado. Además, el UR5e utiliza su controlador integrado para manejar las articulaciones y ejecutar las trayectorias planificadas. Por su parte, la SoftHand 2 está controlada mediante el controlador qbhand2m, que gestiona sus sinergias y movimientos. Por lo tanto, estos controladores están configurados para trabajar de manera coordinada bajo la gestión de ROS 2.

4.3.3 Conexión de la cámara

La integración de la cámara RealSense L515 en el entorno del ROS 2 requirió un proceso manual debido a la falta de soporte oficial por parte de Intel para su SDK. Este procedimiento se facilitó significativamente gracias al uso de Linux, que permitió realizar modificaciones avanzadas que habrían sido inviables en sistemas como Windows, dada su menor flexibilidad en la gestión de dependencias.

La conexión física de la cámara se realizó mediante un puerto USB 3.0. A partir de ahí, se descargó el código fuente del SDK desde el repositorio oficial de Intel, seguido de la instalación de las dependencias necesarias. El entorno de ROS 2 fue configurado adecuadamente, y el SDK fue compilado utilizando colcon, asegurando una integración libre de errores. Finalmente, los paquetes generados se instalaron en el espacio de trabajo, garantizando el funcionamiento correcto de la cámara en ROS 2.

Las conexiones eléctricas y de comunicación se representan de manera conceptual en la figura 4.5, destacando los principales elementos y su interrelación.

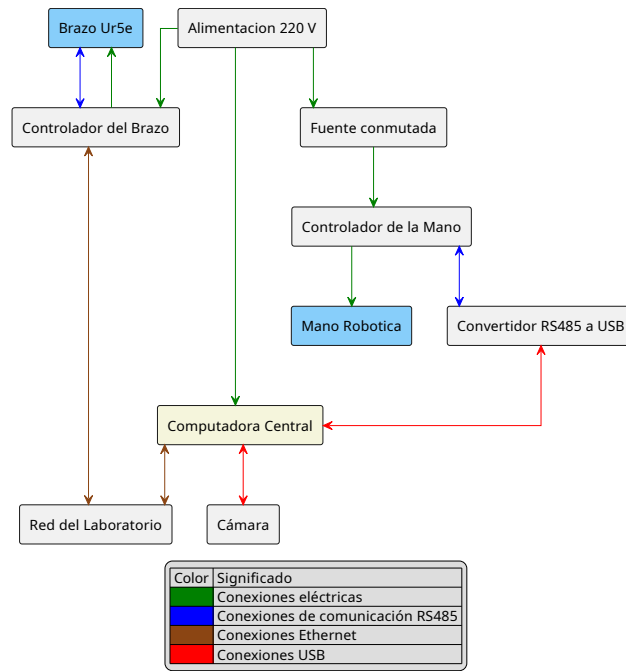


Figura 4.5: Diagrama conceptual de las conexiones eléctricas y de comunicación. Elaboración propia.

4.4 Selección de Linux como Sistema Operativo Base

La elección del sistema operativo constituye un factor crítico en proyectos robóticos, ya que determina el desempeño, la compatibilidad con hardware y software, así como la disponibilidad de herramientas especializadas. En el presente proyecto, se seleccionó Linux como sistema operativo base debido a su reconocida versatilidad, estabilidad y extensa adopción en el ámbito de la robótica. Con lo anterior, estas características permiten un entorno de desarrollo optimizado para la implementación de soluciones avanzadas.

Además, la utilización de Linux como sistema base ofrece una solución más eficiente para abordar los subproblemas identificados en los conceptos de controladores 3.4.3 y sincronización 3.4.3, gracias a su capacidad para gestionar múltiples procesos en tiempo real y su soporte robusto para frameworks robóticos como ROS 2.

4.4.1 Criterios de Evaluación

Para determinar el sistema operativo más adecuado, se establecieron los siguientes criterios de selección:

1. Compatibilidad con ROS 2 y MoveIt, asegurando el correcto funcionamiento de herramientas críticas para la planificación y control de trayectorias.
2. Estabilidad y rendimiento en entornos de tiempo real, considerando la ejecución de tareas complejas y con restricciones temporales.

3. Acceso a herramientas y bibliotecas específicas para robótica.
4. Disponibilidad de soporte técnico y una comunidad activa que facilite la resolución de problemas.
5. Escalabilidad y capacidad de integración con hardware específico, como el manipulador UR5e y la mano SoftHand 2.

4.4.2 Comparativa de Sistemas Operativos

Se evaluaron los sistemas operativos más utilizados en el desarrollo robótico: Windows, macOS y Linux. La Tabla 4.1 presenta una comparativa basada en los criterios definidos anteriormente.

Tabla 4.1: Comparativa de sistemas operativos para desarrollo robótico.

Criterio	Linux	Windows	macOS
Compatibilidad con ROS 2 y MoveIt	Total	Parcial	Parcial
Estabilidad y rendimiento en tiempo real	Excelente	Limitado	Bueno
Acceso a herramientas y bibliotecas para robótica	Completo	Parcial	Parcial
Comunidad y soporte técnico	Amplia y activa	Moderada	Reducida
Escalabilidad y soporte para hardware robótico	Alta	Baja	Moderada
Coste y accesibilidad	Gratuito	Licencia comercial	Licencia comercial

4.4.3 Razones para Seleccionar Linux

El análisis comparativo demostró que Linux es la mejor opción para este proyecto debido a las siguientes razones:

- Compatibilidad completa con ROS 2 y MoveIt: Linux ofrece un soporte nativo para estas herramientas, mientras que otras plataformas requieren configuraciones adicionales o no son plenamente compatibles.
- Rendimiento óptimo en entornos controlados: Gracias a su capacidad para manejar tareas en tiempo real, Linux garantiza una ejecución eficiente y confiable de algoritmos críticos.
- Acceso a herramientas específicas de robótica: La mayoría de las bibliotecas y paquetes relacionados con ROS 2 están optimizados para Linux, lo que reduce el esfuerzo de configuración y asegura una integración más fluida.

- Comunidad activa y soporte técnico: La comunidad de usuarios y desarrolladores de Linux en el ámbito de la robótica es extensa, lo que facilita la resolución de problemas y la implementación de mejoras.
- Escalabilidad y adaptabilidad: Linux permite integrar una amplia gama de hardware, incluyendo dispositivos personalizados como la mano SoftHand 2, incluso cuando requieren compilación desde código fuente.

Aunque otras plataformas como Windows y macOS ofrecen ciertas ventajas, como interfaces gráficas más amigables, estas no compensan las limitaciones en compatibilidad y rendimiento en tiempo real, aspectos clave para el éxito de este proyecto.

4.4.4 Conclusión

La selección de Linux como sistema operativo base fue fundamentada en su compatibilidad, rendimiento, escalabilidad y la solidez de su ecosistema en el ámbito de la robótica. Además, su capacidad para soportar tareas complejas y su compatibilidad nativa con ROS 2 y MoveIt aseguran una plataforma robusta y eficiente para la implementación del sistema integrado UR5e-SoftHand 2, alineándose con los objetivos técnicos y operativos del proyecto.

4.5 Selección de ROS 2 Humble como Plataforma Base

La selección de la versión de ROS 2 para el desarrollo del sistema fue un paso crítico en el diseño del proyecto, ya que las capacidades de la plataforma base influyen directamente en la implementación de componentes como la planificación de trayectorias, el control en tiempo real y la integración sensorial. A su vez, tras una evaluación exhaustiva de las versiones disponibles, se seleccionó ROS 2 Humble como la plataforma base debido a su compatibilidad avanzada y su potencial para cumplir con los requerimientos del proyecto.

4.5.1 Criterios de Selección

Para determinar la versión más adecuada de ROS 2, se establecieron los siguientes criterios de evaluación:

1. Compatibilidad con MoveIt y su soporte para API en Python.
2. Estabilidad general de la versión y disponibilidad de soporte técnico.
3. Compatibilidad con el hardware del manipulador UR5e y la SoftHand 2.
4. Rendimiento en entornos controlados y capacidad para manejar sistemas complejos.
5. Soporte de características avanzadas, como control en tiempo real y comunicaciones seguras.

4.5.2 Comparativa de Versiones de ROS 2

Se compararon las versiones Humble, Rolling y Humble de ROS 2, evaluando su desempeño frente a los criterios definidos. La Tabla 4.2 presenta un resumen de las diferencias clave.

Tabla 4.2: Comparativa de versiones de ROS 2.

Criterio	ROS 2 Humble	ROS 2 Rolling	ROS 2 Humble
Soporte de MoveIt con API Python	Parcial	Experimental	Completo
Estabilidad general	Alta	Media	Alta
Compatibilidad con hardware UR5e	Total	Total	Total
Compatibilidad con hardware SoftHand	Total	Parcial	Requiere compilación desde código
Rendimiento en entornos controlados	Bueno	Muy bueno	Excelente
Soporte a largo plazo	Sí	No	Sí

Ahora bien, ROS 2 Humble ofrece soporte nativo para MoveIt desde los gestores de paquetes de Ubuntu, lo que facilita su instalación inicial. Sin embargo, esta ventaja es relativa, ya que tanto en ROS 2 Rolling como en Humble es posible instalar MoveIt desde su código fuente. En particular, la versión Humble ofrece una integración total con la API de Python de MoveIt, una funcionalidad crucial para este proyecto debido a la dependencia de algoritmos personalizados desarrollados en dicho lenguaje.

4.5.3 Compatibilidad del Hardware

En términos de compatibilidad con el hardware, el manipulador UR5e cuenta con soporte completo en todas las versiones evaluadas. Para la SoftHand 2, inicialmente no se observa una compatibilidad directa en ROS 2 Humble. No obstante, al compilar los controladores desde su código fuente, se logra una integración total y funcional con esta versión. Además, este proceso de compilación fue probado y validado en un entorno controlado, asegurando que tanto el UR5e como la SoftHand puedan operar de manera sincronizada en esta plataforma.

4.5.4 Razones para elegir ROS 2 Humble

El factor determinante para seleccionar ROS 2 Humble fue su soporte completo para la API de Python en MoveIt, lo que simplifica significativamente la implementación de algoritmos personalizados para la planificación de trayectorias y el control avanzado. Además, Humble

combina estabilidad, alto rendimiento en simulaciones y soporte a largo plazo, características esenciales para garantizar la robustez y escalabilidad del sistema.

Por otro lado, la integración sensorial y de controladores en ROS 2 Humble se beneficia de la capacidad de esta versión para manejar tareas complejas y dinámicas con alta eficiencia. Ahora bien, la posibilidad de compilar controladores desde el código fuente supera las limitaciones iniciales de compatibilidad con la SoftHand, lo que asegura la operatividad completa del sistema propuesto.

4.5.5 Conclusión

La elección de ROS 2 Humble como plataforma base fue el resultado de un análisis detallado de las necesidades técnicas del proyecto y las capacidades de las versiones evaluadas. Esta decisión proporciona una base sólida para desarrollar un sistema eficiente, modular y adaptable, que integra de manera efectiva las capacidades del UR5e y la SoftHand 2 en un entorno controlado. Además, el soporte para MoveIt con API en Python asegura flexibilidad en el desarrollo de soluciones innovadoras y escalables.

4.6 Creación de los URDFs del UR5e y la SoftHand 2

Los URDF (Unified Robot Description Format) son archivos que describen la estructura y cinemática de los robots en ROS. En el presente proyecto, se crearon los URDFs del UR5e y la SoftHand 2 para su integración en el sistema. Además estos archivos contienen información detallada sobre las juntas, enlaces y propiedades físicas de los robots, permitiendo su simulación y control en entornos virtuales.

4.6.1 Modelo URDF del UR5e

El modelo del brazo robótico UR5e de Universal Robots se basa en una plantilla proporcionada por la compañía, lo que facilita su creación sin necesidad de modificaciones extensivas. A su vez, esta plantilla incluye todos los parámetros necesarios para describir la estructura y cinemática del UR5e, asegurando una integración precisa y eficiente en el sistema.

4.6.2 Modelo URDF de la SoftHand 2

Para la mano robótica SoftHand 2, se creó un URDF personalizado que refleja su estructura y cinemática. Aunque no se dispone de una versión oficial del URDF para la SoftHand 2, se utilizó el URDF de la versión anterior como base. La principal diferencia entre ambas versiones radica en el uso de dos sinergias motoras en la SoftHand 2, en comparación con una única sinergia en la versión anterior. Por lo tanto, se realizaron las modificaciones necesarias en el URDF existente para ajustarlo a las características de la nueva versión, manteniendo

las dimensiones y propiedades físicas consistentes con la versión anterior ya que son iguales segun los planos de la mano, figura 4.6

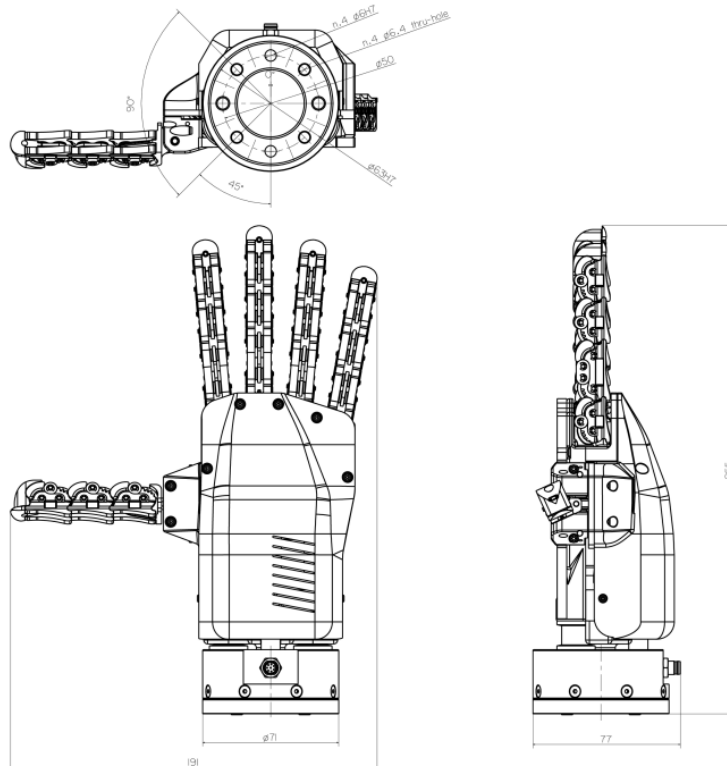


Figura 4.6: Plano de la mano. [15].

4.7 Creación del espacio de trabajo en 3D para la planificación

El espacio de trabajo en 3D es un componente esencial para la planificación de trayectorias en MoveIt, ya que proporciona un entorno virtual que representa fielmente las restricciones físicas y los objetos presentes en el entorno operativo del sistema. Por lo tanto, este espacio se configura utilizando la herramienta MoveIt Setup Assistant, que permite definir los elementos necesarios para la planificación de movimientos.

4.7.1 Definición del espacio de trabajo

El espacio de trabajo incluye el modelo URDF del sistema integrado UR5e-SoftHand 2, que detalla las características cinemáticas y dinámicas del manipulador y la mano robótica, como se muestra en la figura 4.7. Además, este espacio virtual constituye el entorno operativo donde el robot planifica y ejecuta trayectorias de manipulación, asegurando precisión y evitando colisiones.

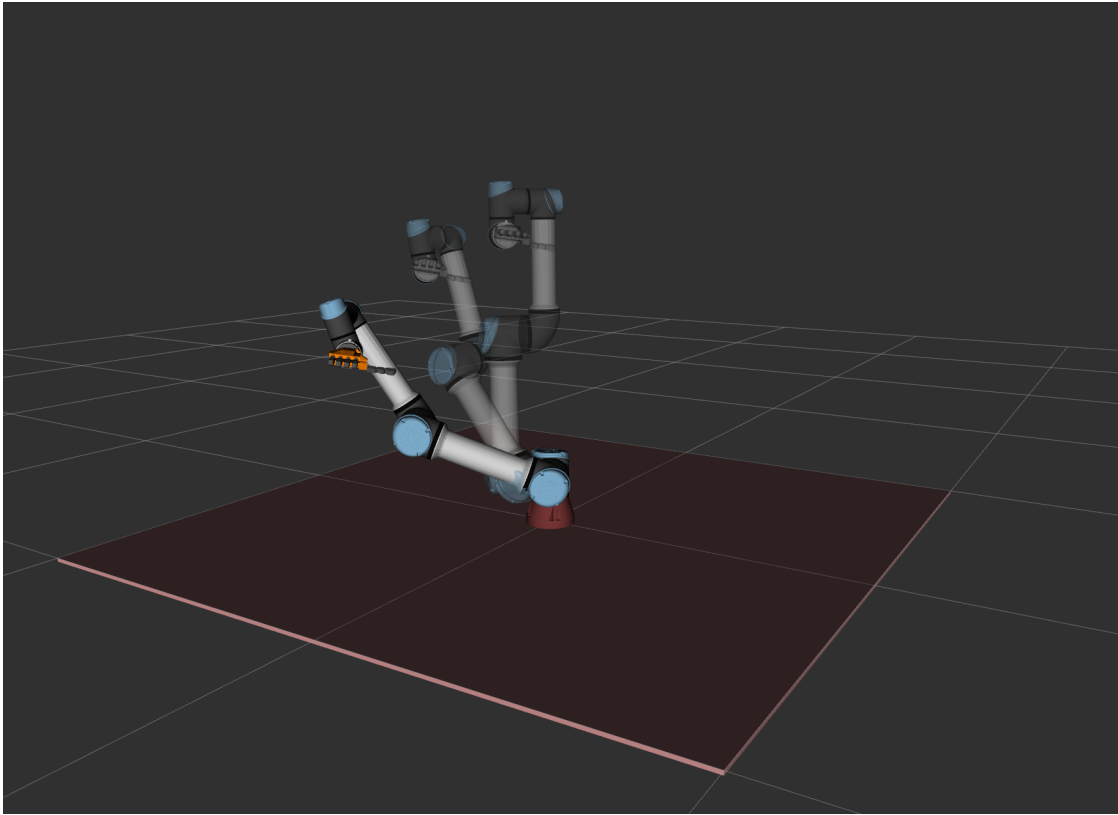


Figura 4.7: Espacio de trabajo. Elaboración propia.

Ahora bien, el robot se encuentra montado en la parte superior de una mesa que sirve como superficie de trabajo, ubicada al nivel del suelo o de la base del brazo. Aunque la mesa no se modela explícitamente en el entorno virtual, se establecen restricciones en el eje vertical que evitan que el robot planifique trayectorias por debajo del nivel de la mesa. Con este enfoque se simplifica el modelado del espacio de trabajo, mientras asegura que el manipulador opere únicamente dentro de un volumen accesible seguro.

Los elementos clave del espacio de trabajo incluyen:

- **Modelos de colisión:** Representaciones simplificadas de los componentes móviles del robot y los objetos manipulables. Estas representaciones son utilizadas para detectar y evitar colisiones durante la planificación de trayectorias, asegurando que las configuraciones propuestas sean viables en el entorno físico.
- **Puntos de referencia:** Coordenadas definidas en el espacio tridimensional para guiar el posicionamiento del manipulador. Estas incluyen los límites superiores e inferiores del volumen de trabajo, asegurando que las trayectorias planificadas cumplan con las restricciones impuestas por la posición de la mesa.
- **Configuración de sensores:** Dispositivos como cámaras y sensores de fuerza/par, que proveen datos en tiempo real sobre el entorno. Aunque estos sensores no modelan directamente la mesa, contribuyen a validar las interacciones del robot con los objetos situados sobre ella.

La ausencia de un modelo explícito de la mesa en el espacio virtual optimiza los recursos computacionales y simplifica el proceso de configuración. En lugar de ello, las restricciones geométricas definen un límite virtual que simula la presencia de la mesa, garantizando que el robot respete las condiciones físicas del entorno sin necesidad de añadir complejidad al modelo.

Este enfoque asegura que el espacio de trabajo esté configurado de manera eficiente, permitiendo que el manipulador opere con seguridad y precisión en un entorno controlado, mientras interactúa con los objetos ubicados sobre la mesa. Al mismo tiempo, se asegura que las trayectorias sean viables y libres de colisiones, contribuyendo al éxito de tareas asignadas al sistema.

4.7.2 Proceso de configuración con MoveIt Setup Assistant

El MoveIt Setup Assistant se utilizó para realizar las siguientes configuraciones:

- **Carga del modelo URDF:** Importación del modelo URDF del UR5e y la SoftHand, incluyendo las propiedades de las juntas y los límites de movimiento.
- **Generación de archivos de configuración:** Creación automática de archivos de configuración de MoveIt, que incluyen los grupos de planificación, las configuraciones de sensores y los parámetros cinemáticos.
- **Definición de zonas de trabajo:** Configuración de un volumen de trabajo que delimita el espacio accesible para el manipulador, asegurando que las trayectorias se mantengan dentro de los límites físicos y operativos del sistema.

Validación del espacio de trabajo

Una vez configurado el espacio de trabajo, se realizaron pruebas para validar su funcionalidad en entornos simulados. Además, estas pruebas incluyeron la verificación de trayectorias generadas por MoveIt, la detección efectiva de colisiones y la integración de datos sensoriales en tiempo real. Este proceso garantiza que el espacio de trabajo virtual refleje con precisión las condiciones físicas y operativas del sistema, proporcionando una base sólida para la planificación de movimientos (ver figura 4.8).

4.8 Algoritmos de planificación y control

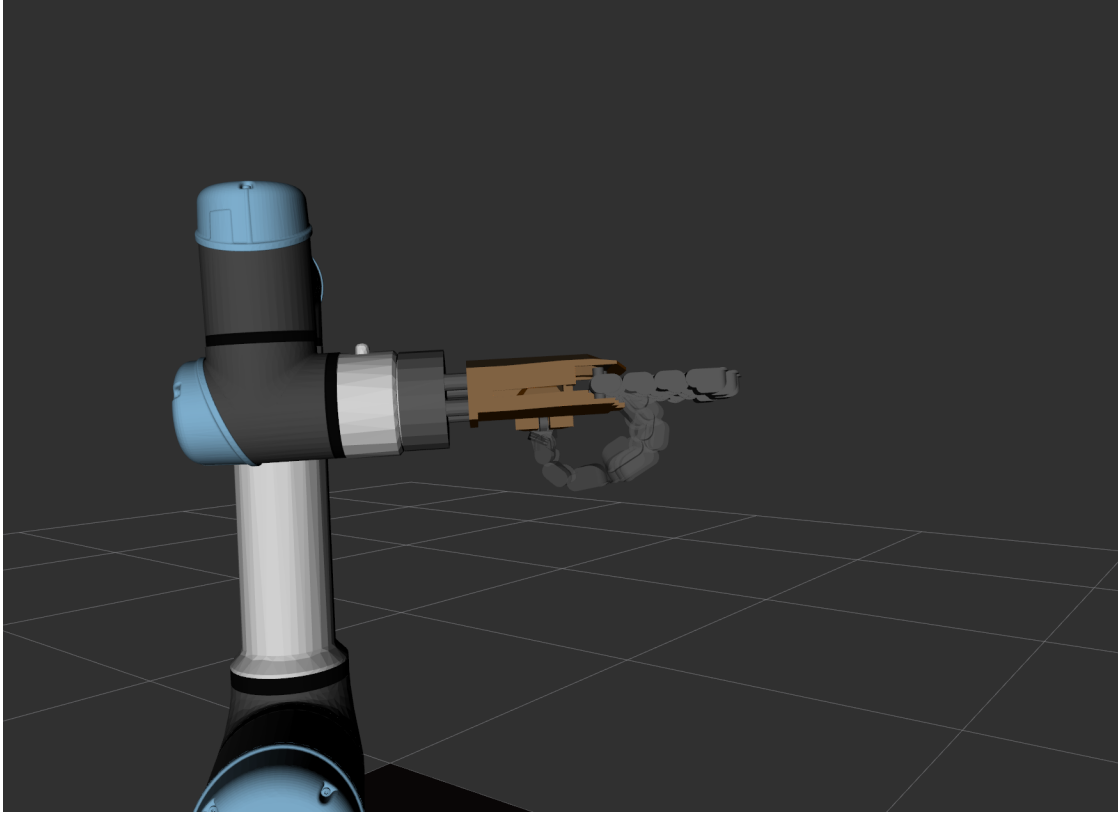


Figura 4.8: Modelo de la mano. Elaboración propia.

El módulo de planificación de movimientos utiliza MoveIt como herramienta principal para la generación de trayectorias libres de colisiones. Este módulo emplea algoritmos como RRT (Rapidly-exploring Random Trees) y PRM (Probabilistic Roadmap), que permiten la exploración eficiente del espacio de configuraciones, generando trayectorias que cumplen con las restricciones físicas y ambientales del sistema.

Por otro lado, el control del manipulador UR5e se realiza mediante el controlador `scaled_joint_trajectory_controller`, diseñado para garantizar movimientos precisos y suaves. Por su parte, el controlador de sinergias de la SoftHand optimiza los movimientos adaptativos, permitiendo la manipulación de objetos con geometrías y texturas variadas. La interacción entre el planificador y los controladores se coordina a través de ROS 2, utilizando tópicos y acciones que aseguran un flujo de información confiable y en tiempo real.

La interpolación de trayectorias se modela mediante la ecuación (4.1), que describe cómo las configuraciones articulares evolucionan entre puntos clave.

$$\mathbf{q}(t) = \mathbf{q}_0 + (\mathbf{q}_f - \mathbf{q}_0) \cdot s(t), \quad (4.1)$$

Donde $\mathbf{q}(t)$ es la posición articular en el instante t , $\mathbf{q}_0$ y $\mathbf{q}_f$ representan las posiciones inicial y final, respectivamente, y $s(t)$ es una función de interpolación suave que garantiza trayectorias

continuas y diferenciables.

4.8.1 Configuración del solver cinemático

Para abordar los problemas de cinemática directa e inversa en el sistema integrado, se ha implementado el plugin KDL Kinematics Plugin de ROS 2. Esta configuración está diseñada tanto para el grupo "Arm" del brazo UR5e como para el grupo "Hand" de la SoftHand 2, asegurando un control cinemático eficiente y adaptado a las necesidades específicas de cada componente.

El plugin seleccionado pertenece a la Kinematics and Dynamics Library (KDL) y se integra fácilmente con MoveIt, lo que permite resolver trayectorias en tiempo real y calcular configuraciones de juntas de manera precisa, como se muestra en la figura 4.9.

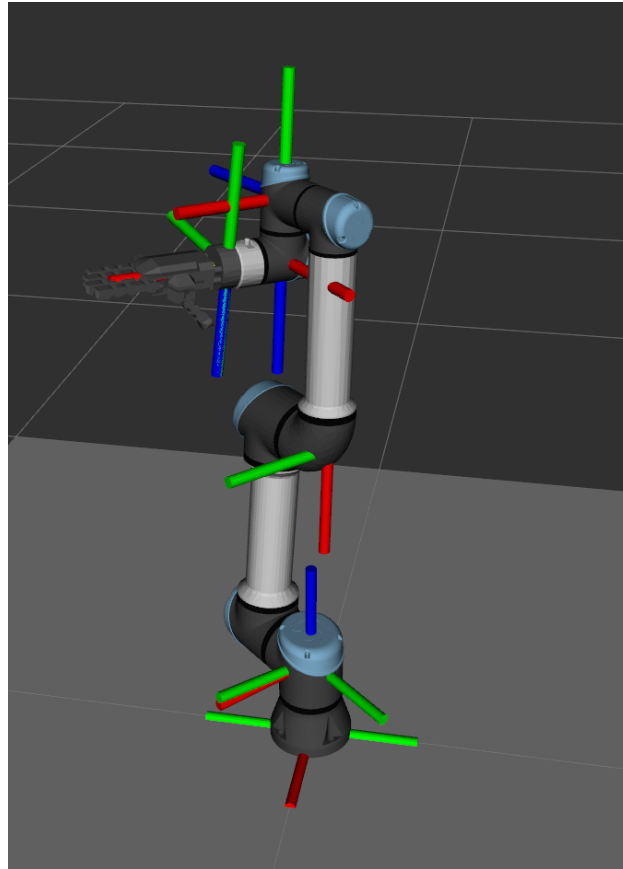


Figura 4.9: Juntas a las cuales se les aplica el solver cinemático. Elaboración propia.

Los parámetros principales configurados para el solver cinemático abarcan aspectos fundamentales. En primer lugar, se ha utilizado el plugin `kdl_kinematics_plugin/KDLKinematicsPlugin` debido a su compatibilidad con manipuladores industriales y su integración nativa con MoveIt. Además, se ha definido el parámetro `kinematics_solver_search_resolution` con un valor de 0.005 radianes, logrando un equilibrio entre precisión y velocidad en los cálculos de la cinemática inversa.

Por último, el tiempo límite para encontrar una solución, especificado mediante el parámetro `kinematics_solver_timeout`, se estableció en 0.005 segundos, priorizando la rapidez en el procesamiento. Este valor puede ser ajustado para manejar trayectorias más complejas o solucionar fallas recurrentes durante la ejecución.

El KDLKinematicsPlugin es altamente eficaz para manipuladores como el UR5e debido a su configuración con seis grados de libertad y a la claridad en la definición de sus modelos cinemáticos. Sin embargo, al aplicarse en la SoftHand 2, que incorpora sinergias motoras adaptativas, podrían surgir limitaciones en la representación exacta de sus propiedades. Para el grupo "Arm", el plugin satisface los requisitos del brazo, permitiendo trayectorias suaves y resolviendo problemas de cinemática directa e inversa con alta precisión. Por otro lado, para el grupo "Hand", aunque el plugin resuelve la cinemática inversa de forma general, se debe evaluar su capacidad para manejar las sinergias propias de la SoftHand 2. En caso de limitaciones, podría ser necesario explorar otras opciones como Trac-IK o desarrollar soluciones personalizadas.

Con el fin de garantizar un rendimiento óptimo en la resolución cinemática, se sugieren varias acciones. Primero, incrementar el valor del tiempo de espera a un rango de 0.1 a 0.2 segundos podría aumentar la tasa de éxito en trayectorias complejas. También es importante revisar el modelo URDF de la SoftHand 2 para garantizar que refleja correctamente las sinergias motoras. Además, se podría considerar la integración de solvers alternativos como Trac-IK, que ofrecen mayor flexibilidad en problemas de cinemática inversa. Finalmente, es fundamental realizar pruebas exhaustivas en simuladores para identificar posibles fallos y optimizar la configuración del solver antes de su despliegue en el sistema físico.

Esta configuración descrita asegura que el sistema UR5e-SoftHand 2 pueda ejecutar trayectorias con alta precisión y adaptarse a entornos dinámicos. Al aprovechar las capacidades de MoveIt junto con el plugin KDL, es posible generar trayectorias libres de colisiones y optimizar el rendimiento general del manipulador, tanto en tareas de pick-and-place como en otras operaciones de manipulación robótica.

4.9 Integración de los controladores

Como parte del diseño del sistema, se han integrado controladores específicos para el manipulador UR5e y la mano SoftHand 2, asegurando un control preciso y coordinado de ambos componentes. Estos controladores se comunican a través de ROS 2, lo que permite una interacción fluida y eficiente entre el brazo y la mano robótica, como se ilustra en la figura 4.10.

Para lograr esta integración, se ha aprovechado la capacidad de comunicación entre nodos que proporciona ROS 2. En este contexto, se desarrolló un script en Python que combina los controladores de posición del brazo y la mano en un único controlador. Este controlador unificado permite la manipulación eficiente de objetos, publicando el estado actual tanto del brazo como de la mano en un único tópico, `/merged_joint_state`. Además, esta configuración simplifica la arquitectura del sistema, mejorando la sincronización y el rendimiento en

tareas de manipulación colaborativa.

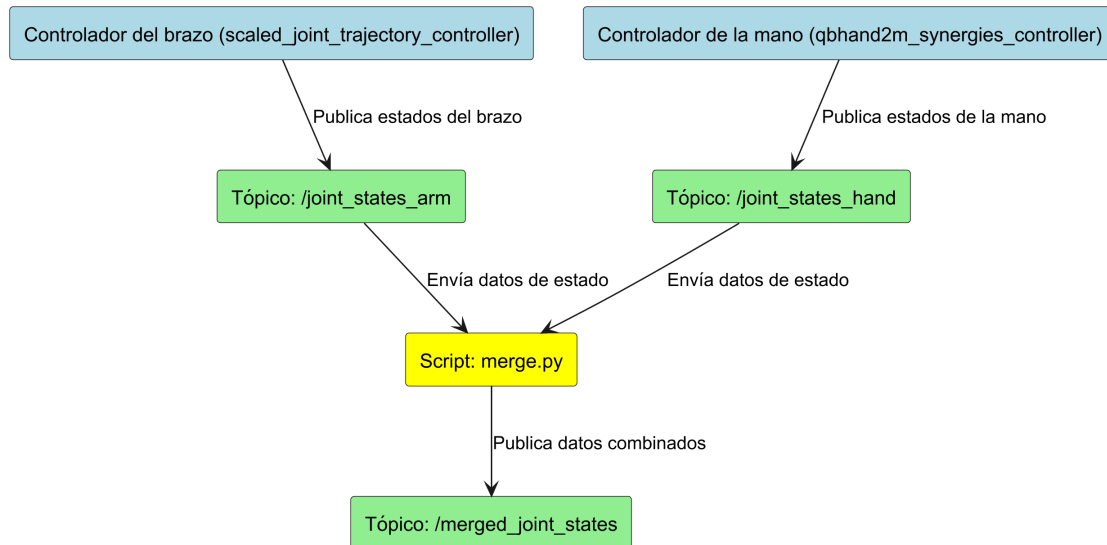


Figura 4.10: Diagrama del flujo de información entre controladores. Elaboración propia.

4.10 Integración de Sensores

El sistema propuesto incluye una integración sensorial avanzada, que permite adaptar dinámicamente el comportamiento del manipulador y la mano robótica al entorno. Los sensores de fuerza/par, instalados en puntos clave, detectan interacciones físicas y ajustan las fuerzas aplicadas para evitar daños a los objetos manipulados o al entorno.

Además, las cámaras de visión artificial proporcionan información detallada sobre la posición y orientación de los objetos en el espacio de trabajo. Este flujo de datos se procesa mediante algoritmos de segmentación y detección de bordes, generando puntos de interés que alimentan la planificación de trayectorias. Toda la información sensorial se centraliza en un nodo de integración, que coordina los ajustes necesarios en tiempo real, garantizando un desempeño robusto y adaptable del sistema.

4.11 Interacción Humano-Robot

El diseño del sistema prioriza la seguridad y la efectividad en la interacción humano-robot. Se implementan estrategias avanzadas de control de impedancia, que regulan las fuerzas aplicadas durante las tareas de manipulación, protegiendo tanto al usuario como al sistema robótico. Asimismo, se desarrollan interfaces intuitivas que facilitan la configuración, supervisión y operación del sistema por parte del usuario.

Además estas interfaces incluyen representaciones visuales del estado del sistema en RViz y controles accesibles para ajustar parámetros operativos, mejorando la experiencia del usuario y fomentando una colaboración eficiente entre el humano y el robot.

4.12 Validación y Simulación

El diseño conceptual se valida inicialmente en entornos de simulación, utilizando herramientas como RViz y Gazebo. Estas simulaciones permiten probar y ajustar cada componente del sistema en un entorno controlado, asegurando su correcto funcionamiento antes de la implementación en hardware real. Entre las simulaciones realizadas se incluyen:

- **Verificación y optimización de trayectorias:** Evaluación de trayectorias planificadas para garantizar su precisión y evitar colisiones.
- **Pruebas de manipulación de objetos:** Validación del comportamiento del sistema mediante modelos realistas de sensores y actuadores.
- **Interacción humano-robot:** Simulación de escenarios colaborativos para garantizar la seguridad y la efectividad en la interacción.

La validación iterativa en simulación asegura que los elementos del diseño estén alineados con los objetivos técnicos y operativos del proyecto, minimizando riesgos y optimizando el proceso de desarrollo.

4.13 Flujo de información entre los componentes

El flujo de información en el sistema integrado UR5e-SoftHand 2 está diseñado para abarcar tres etapas fundamentales: percepción, planificación y ejecución. Estas etapas trabajan de manera coordinada para garantizar una interacción eficiente entre los módulos del sistema, permitiendo la manipulación precisa y adaptativa de objetos en entornos controlados, como se ilustra en la Fig. 4.11.

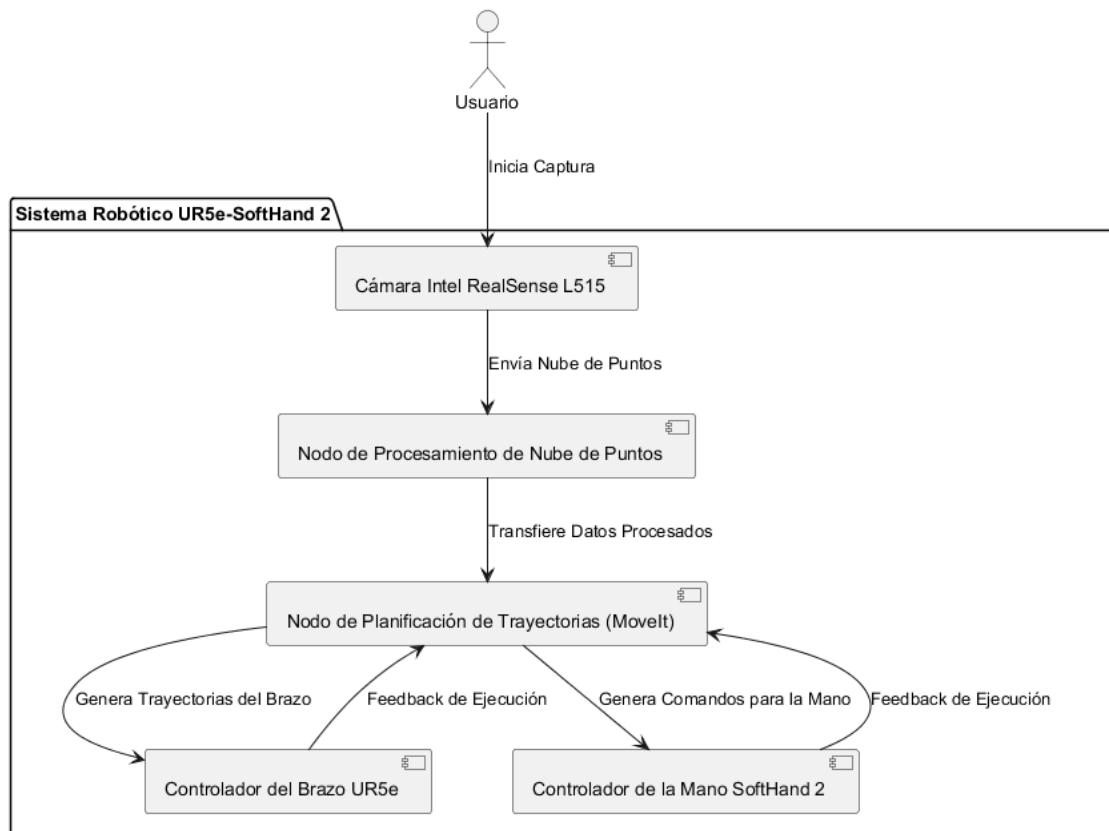


Figura 4.11: Diagrama conceptual del flujo de información. Elaboración propia.

4.13.1 Percepción del entorno

La cámara Intel RealSense L515 es el elemento central del sistema sensorial, proporcionando datos tridimensionales en tiempo real a través de tecnología LiDAR. Este dispositivo está montado a una altura de dos metros sobre la mesa de trabajo, enfocándose directamente en el espacio de manipulación donde se encuentran los objetos de interés.

La mesa de trabajo no se modela explícitamente en el entorno virtual. En su lugar, se establece un límite en el plano base del robot para evitar que el manipulador se desplace por debajo de la superficie de la mesa. Esta simplificación reduce la complejidad del modelo y mejora la eficiencia computacional.

Los datos capturados por la cámara generan una nube de puntos que representa el espacio operativo. Esta nube se procesa mediante técnicas avanzadas de filtrado para eliminar ruido y destacar las características clave de los objetos manipulables. El resultado es una representación precisa y clara del entorno, que alimenta el módulo de planificación de trayectorias.

4.13.2 Planificación de movimientos

Los datos tridimensionales proporcionados por la nube de puntos son integrados en el pipeline de planificación de MoveIt, que utiliza algoritmos avanzados como RRT (Rapidly-exploring Random Trees) y PRM (Probabilistic Roadmap) para generar trayectorias óptimas y libres de colisiones.

La información sensorial delimita las zonas seguras de operación y permite identificar la posición y orientación precisas de los objetos. Esto asegura que las trayectorias planificadas cumplan con las restricciones físicas del sistema y las condiciones operativas del entorno. Además, la capacidad de ajuste dinámico de MoveIt permite que las trayectorias respondan a cambios en la disposición de los objetos en tiempo real.

Además, MoveIt también sincroniza los movimientos del manipulador UR5e y la SoftHand, asegurando que la manipulación sea precisa y coordinada. Este proceso es esencial para tareas complejas, como el manejo de objetos de geometrías irregulares, donde la integración entre el brazo y la mano robótica resulta crítica.

4.13.3 Ejecución de movimientos

Las trayectorias planificadas son traducidas en movimientos físicos mediante los controladores asignados a cada componente del sistema. Además el controlador `scaled_joint_trajectory_controller` gestiona los movimientos del UR5e, garantizando que sean suaves y precisos, respetando las limitaciones cinemáticas y dinámicas del brazo.

Simultáneamente, el controlador `qbhand2m` se encarga de gestionar las sinergias de la SoftHand, permitiendo una manipulación adaptativa de objetos con diferentes formas y texturas. La sincronización entre ambos controladores se logra mediante comandos generados por MoveIt

y transmitidos a través de ROS 2, asegurando un flujo continuo de información entre los módulos de planificación y control. Por lo tanto esta configuración permite una interacción fluida y coordinada entre el manipulador y la mano robótica, minimizando errores en la ejecución y optimizando el rendimiento en tareas de manipulación colaborativa.

4.13.4 Ventajas del diseño del flujo de información

El diseño del flujo de información maximiza la eficiencia computacional y la adaptabilidad del sistema. Además la cámara Intel RealSense L515 facilita la creación de un modelo tridimensional en tiempo real, eliminando la necesidad de modelar la mesa de trabajo. Esto permite al sistema centrarse en los objetos relevantes y las restricciones operativas, lo que resulta en un entorno de planificación más flexible y eficiente.

Por otro lado el uso combinado de percepción avanzada, herramientas de planificación robustas y controladores precisos establece una base sólida para el desempeño del sistema UR5e-SoftHand 2. Este flujo de información asegura que el sistema pueda operar con precisión y adaptabilidad en tareas de manipulación colaborativa, cumpliendo con los objetivos técnicos y operativos del proyecto.

4.14 Consideraciones de seguridad

El UR5e es un manipulador robótico diseñado en la Unión Europea, lo que implica que cumple con las normativas de seguridad vigentes en la región, como la **ISO 10218-1:2011** [17] y la **ISO/TS 15066:2016** [18]. Entre las características integradas en su diseño se incluyen sensores de detección de impactos y mecanismos de parada segura, los cuales garantizan la seguridad durante su operación.

Preservación de medidas de seguridad integradas

Durante el proceso de integración con la mano robótica SoftHand 2 y otros componentes del sistema, se ha priorizado mantener intactas las características de seguridad del UR5e. Entre estas medidas destacan:

- **Sensores de choque:** Los sensores integrados en el UR5e detectan colisiones o impactos, ajustando la operación para minimizar daños a personas o equipos.
- **Paradas de emergencia:** El sistema incluye paradas de emergencia independientes del software, diseñadas para desconectar físicamente el suministro de energía al robot en caso de un evento crítico. Esto asegura que el sistema pueda ser detenido incluso en presencia de fallos en el software o hardware de control.

Fiabilidad del sistema y diseño a prueba de fallos

El diseño del sistema considera un enfoque "a prueba de fallos" (*fail-safe*), asegurando que cualquier integración adicional, como los controladores o los elementos de software, no interfieran con las medidas de seguridad originales del UR5e. En particular:

- Se verificó que las señales de los sensores de seguridad y los mecanismos de parada del robot no sean alterados por las modificaciones realizadas en el software de control.
- Los procedimientos de emergencia pueden ser ejecutados manualmente mediante botones físicos ubicados en el *teach pendant* o en el controlador del robot, garantizando una desconexión segura del sistema en caso de una anomalía.

Cumplimiento normativo en el diseño final

Estas medidas aseguran que el sistema integrado cumpla con las normativas de seguridad europeas aplicables, preservando las características del UR5e que lo hacen adecuado para operaciones en entornos colaborativos y controlados. De este modo, se garantiza que la integración de la SoftHand 2 y otros componentes no comprometan la seguridad del sistema en su conjunto.

Capítulo 5

Resultados y Análisis

En este capítulo se presentan los resultados obtenidos y el análisis correspondiente de las pruebas realizadas en el sistema integrado UR5e-SoftHand 2. A su vez, las evaluaciones se llevaron a cabo en entornos controlados, lo que permitió realizar una validación exhaustiva de los parámetros y comportamientos clave. Esto garantizó condiciones consistentes y reproducibles para la experimentación, aportando datos confiables para el análisis.

Ahora bien, el proceso experimental incluyó la implementación de pruebas específicas diseñadas para verificar aspectos críticos del sistema, tales como:

- **Precisión en la manipulación:** Evaluando la capacidad del sistema para alcanzar posiciones objetivo con alta exactitud.
- **Repetibilidad de movimientos:** Analizando la consistencia en la ejecución de tareas repetitivas.
- **Rendimiento bajo carga:** Determinando la estabilidad y efectividad del sistema al manipular objetos con diferentes pesos.

Estas pruebas no solo validaron el cumplimiento de los requerimientos establecidos, sino que también identificaron oportunidades de mejora y ajustes necesarios para optimizar el desempeño en escenarios reales. A continuación, se presentan los resultados y análisis de las principales pruebas realizadas.

5.1 Validación del sistema integrado

Para evaluar el desempeño del sistema UR5e-SoftHand 2, se llevaron a cabo pruebas de *pick and place*, considerando métricas clave como la precisión posicional y la capacidad de manipulación bajo carga. Además, cada prueba fue diseñada para replicar escenarios representativos de aplicaciones prácticas, garantizando un enfoque sistemático para la validación del sistema.

5.1.1 Prueba de precisión

La prueba de precisión tuvo como objetivo evaluar la capacidad del sistema para posicionar la mano robótica en ubicaciones específicas con alta exactitud. Se consideraron únicamente las coordenadas x e y , dado que las tareas evaluadas se desarrollaron en el plano horizontal. Las mediciones se realizaron en 10 posiciones estratégicamente seleccionadas, representando tanto los límites del espacio de trabajo como puntos intermedios dentro del área definida. Las posiciones objetivo utilizadas durante la prueba se muestran en la Tabla 5.1.

Tabla 5.1: Posiciones objetivo en el plano horizontal (x y y)

Posición	Coordenada x (cm)	Coordenada y (cm)
1	10.0	10.0
2	20.0	30.0
3	40.0	50.0
4	60.0	70.0
5	80.0	80.0
6	50.0	10.0
7	70.0	30.0
8	30.0	60.0
9	60.0	40.0
10	10.0	70.0

Con lo anterior, las posiciones se definieron teniendo en cuenta las siguientes consideraciones:

- **Límites del espacio de trabajo:** Las posiciones cercanas al borde máximo de la mesa (80 cm en Y) representan el alcance extremo.
- **Distribución intermedia:** Posiciones intermedias (X y Y entre 10 cm y 70 cm) para evaluar precisión a diferentes distancias.
- **Variación estratégica:** Ubicaciones con diferentes combinaciones de X e Y para analizar el comportamiento del sistema en diferentes trayectorias.

Por lo tanto, la tabla anterior garantiza una cobertura completa del área de trabajo dentro de los límites operativos del robot y proporciona una base sólida para el análisis de precisión.

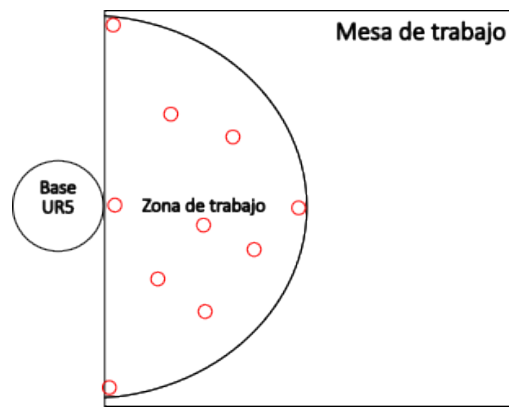
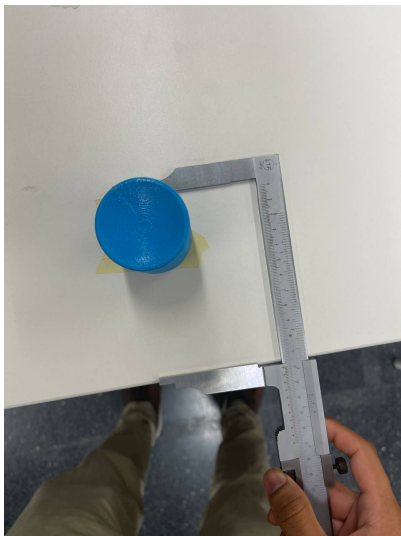


Figura 5.1: Posiciones objetivo en el plano horizontal. Elaboración propia.

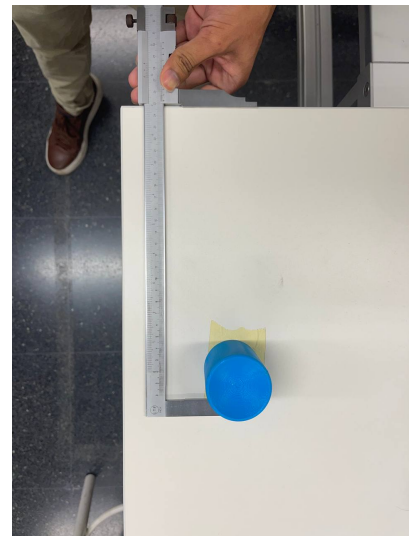
Metodología

En cada posición, se realizaron múltiples intentos de tareas de *pick and place*, registrando las desviaciones entre las posiciones objetivo y las alcanzadas, como se ilustra en la figura 5.1. Para estandarizar el proceso y reducir la variabilidad asociada, se utilizaron objetos circulares como elementos de prueba y se estableció un sistema de coordenadas tomando una esquina de la mesa como origen. Esto permitió definir un marco de referencia fijo para todas las mediciones.

Las posiciones alcanzadas se registraron utilizando un pie de rey digital con una precisión de ± 0.01 cm. Se aplicó un análisis estadístico para identificar posibles valores atípicos mediante el rango intercuartílico (IQR). Posteriormente, se calcularon la media y la desviación estándar de las mediciones para evaluar la precisión del sistema.



(a) Medición de la desviación desde el eje Y.



(b) Medición de la desviación desde el eje X.

Figura 5.2: Medición de las desviaciones desde los ejes X e Y. Elaboración propia.

Resultados

Los datos recopilados demostraron que el sistema pudo alcanzar las posiciones objetivo con un error promedio inferior al umbral de 0.2 cm en todas las posiciones evaluadas. Los resultados estadísticos para cada posición se presentan en la Tabla 5.2. Además, las desviaciones en los ejes x e y se graficaron, como se muestra en las figuras 5.3 y 5.4. Los datos en bruto, presentados en la Tabla D.1, muestran las desviaciones registradas durante la prueba de precisión en distintas posiciones objetivo.

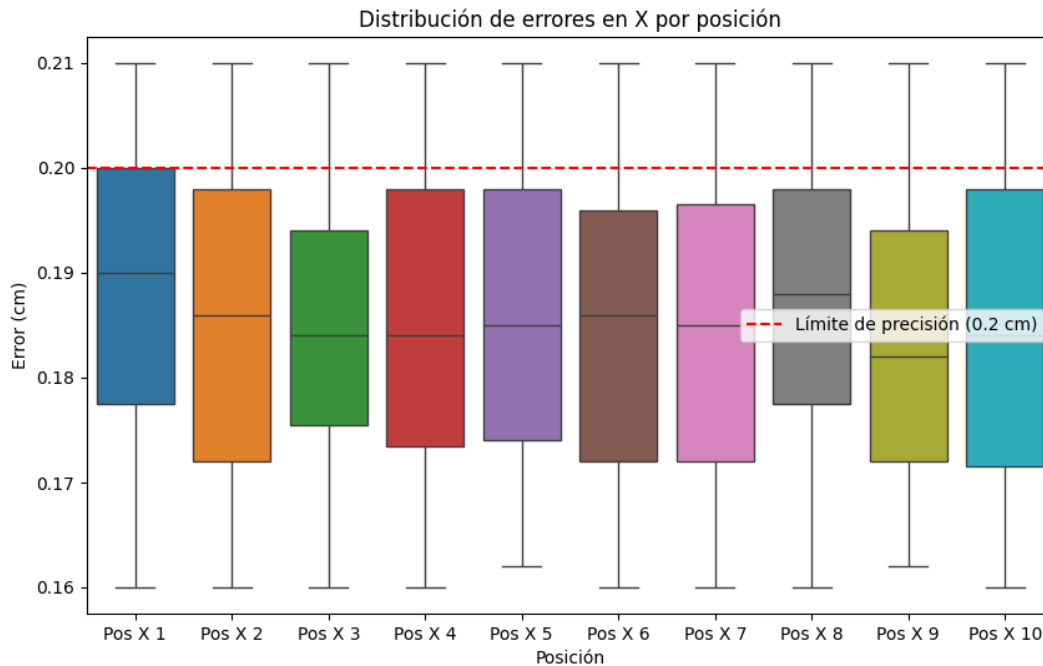


Figura 5.3: Errores obtenidos en el eje x . Elaboración propia.

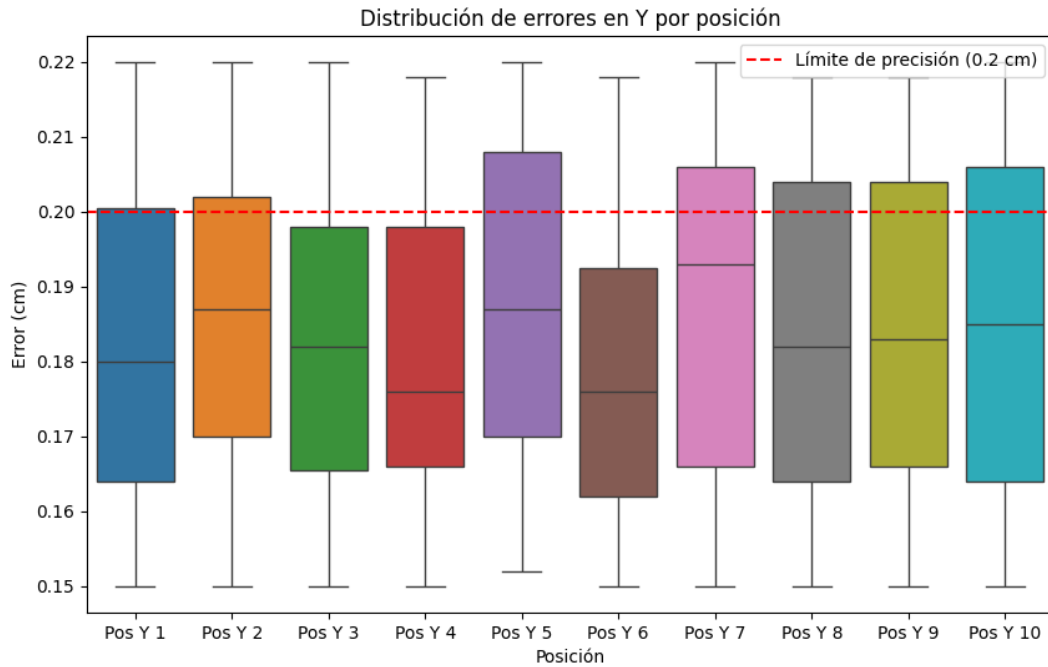


Figura 5.4: Errores obtenidos en el eje y . Elaboración propia.

Tabla 5.2: Resultados estadísticos de errores por posición

Posición	Media (cm)	Desviación estándar (cm)	t-Estadístico	Valor p	Cumple ($\mu < 0.2$ cm)
1	0.183	0.019	-8.835	< 0.001	Sí
2	0.186	0.020	-6.838	< 0.001	Sí
3	0.184	0.021	-7.699	< 0.001	Sí
4	0.184	0.018	-8.948	< 0.001	Sí
5	0.183	0.019	-9.316	< 0.001	Sí
6	0.183	0.022	-7.654	< 0.001	Sí
7	0.186	0.020	-6.981	< 0.001	Sí
8	0.187	0.018	-6.966	< 0.001	Sí
9	0.184	0.021	-7.442	< 0.001	Sí
10	0.184	0.020	-7.747	< 0.001	Sí

Conclusión

Con base en los resultados obtenidos, se concluye que el sistema UR5e-SoftHand 2 cumple con los criterios de precisión establecidos. Todas las posiciones evaluadas mostraron errores promedio inferiores al umbral de 0.2 cm, validando la capacidad del sistema para operar con alta precisión en tareas de *pick and place*.

5.1.2 Prueba de carga

La prueba de carga tuvo como objetivo evaluar la capacidad del sistema UR5e-SoftHand 2 para manipular objetos de forma segura y eficiente, verificando su estabilidad y precisión bajo diferentes condiciones de carga. Las botellas fueron seleccionadas porque su geometría cilíndrica es la más cercana a la descripción del objeto ideal utilizada en las pruebas descritas en la hoja de datos de la SoftHand 2. Esto garantiza que los objetos manipulados cumplan con las especificaciones del sistema en términos de forma, dimensiones y comportamiento esperado.

Se seleccionaron cinco botellas con pesos y dimensiones definidos, como se muestra en la Tabla 5.3. Cada botella fue manipulada siguiendo una metodología estandarizada, que incluyó levantamiento, rotación y reposicionamiento. Los datos recolectados incluyeron las desviaciones posicionales, estabilidad del agarre, tiempos de ejecución y la calidad del agarre inicial.

Tabla 5.3: Especificaciones de botellas seleccionadas para la prueba de carga

Botella	Altura (cm)	Diámetro (mm)	Peso (kg)
Botella 1	15.0	45	1.0
Botella 2	20.0	60	2.0
Botella 3	25.0	75	3.0
Botella 4	30.0	90	4.0
Botella 5	35.0	90	5.0

Metodología

La metodología aplicada en la prueba fue diseñada para garantizar la consistencia y reproducibilidad de las evaluaciones. Los pasos realizados para cada botella fueron los siguientes:

1. **Inicio:** El robot toma la botella desde una posición inicial fija, asegurando un agarre estable.
2. **Levantamiento:** El UR5e eleva la botella 30 cm desde su posición inicial y la mantiene suspendida durante 10 segundos para evaluar la estabilidad del agarre.
3. **Rotación:** La mano robótica realiza una rotación de 90°, cambiando la orientación de la botella y manteniéndola en la nueva posición durante 10 segundos.
4. **Reposicionamiento:** Después del periodo de espera, la botella se devuelve a su orientación inicial y se deposita en una nueva posición dentro del espacio de trabajo.

Esta metodología permite evaluar cómo las diferentes características de las botellas (peso y dimensiones) afectan el desempeño del sistema en términos de estabilidad, precisión, tiempos de ejecución y agarre inicial.

Resultados

Los resultados obtenidos indican que el sistema manejó cargas de hasta 5.0 kg con un rendimiento aceptable. Se observó que el sistema presentó estabilidad y precisión adecuadas para las primeras cuatro botellas (hasta 4.0 kg). Sin embargo, en la Botella 5 (5.0 kg), se identificaron dos problemas principales:

- **Dificultades en el agarre inicial:** A medida que el peso se acercó a los 5.0 kg, el agarre inicial de la botella se complicó. En lugar de sujetar directamente la botella, la mano robótica en ocasiones empujaba el objeto durante el cierre inicial de los dedos. Este problema no se presentó con pesos inferiores a 3.0 kg, donde el agarre fue estable y preciso.
- **Leve deslizamiento durante el levantamiento:** En la botella más pesada (5.0 kg), se observó un leve deslizamiento mientras el objeto permanecía suspendido.

Los resultados detallados se presentan en la Tabla 5.4.

Tabla 5.4: Resultados promedio de la prueba de carga

Botella	Desviación promedio (cm)	Tiempo promedio (s)	Intentos exitosos (de 10)
Botella 1	0.10	11.5	10/10
Botella 2	0.14	12.8	10/10
Botella 3	0.18	14.0	9/10
Botella 4	0.24	15.5	8/10
Botella 5	0.30	17.0	7/10

Nota: Un intento exitoso se define como la manipulación de la botella sin deslizamiento ni pérdida del agarre durante el tiempo de prueba.

Análisis

Los resultados muestran que el sistema UR5e-SoftHand 2 es capaz de manejar objetos con pesos de hasta 5.0 kg, pero la precisión y estabilidad comienzan a disminuir a medida que el peso se acerca a este límite. La desviación promedio y el tiempo necesario para completar las tareas aumentan con el peso, lo que refleja el esfuerzo adicional requerido por el sistema para manipular cargas más pesadas.

Además, se identificó que el agarre inicial de objetos cercanos a 5.0 kg puede ser inestable, ya que la mano tiende a empujar la botella durante el cierre inicial. Este problema no se presentó con botellas de hasta 3.0 kg, donde el agarre inicial fue estable. Sin embargo, con cargas más pesadas, la fricción y la distribución del peso del objeto parecen superar los límites de control del sistema.

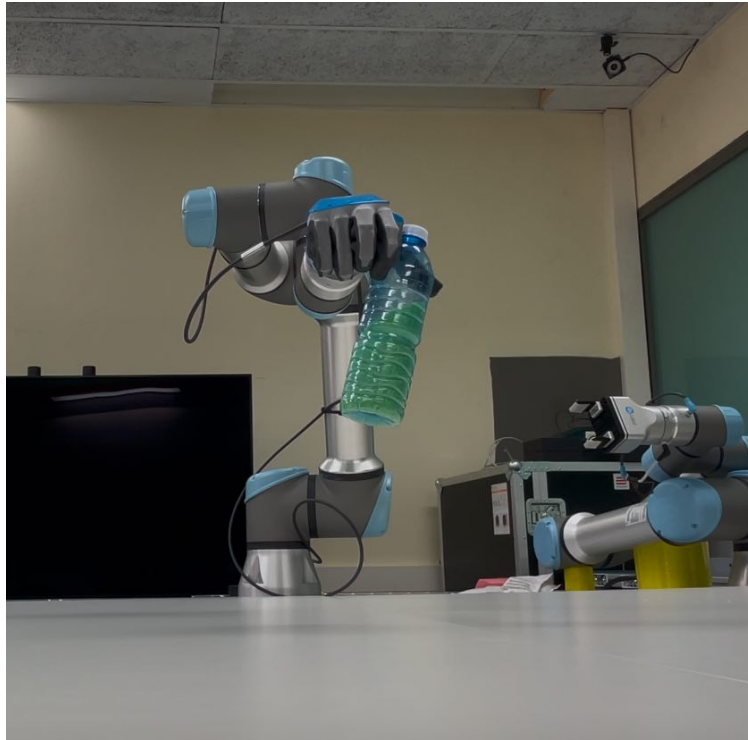


Figura 5.5: Botella de 1.0 kg en la SoftHand 2. Elaboración propia.

Conclusión

La prueba de carga valida que el sistema UR5e-SoftHand 2 cumple con los requerimientos establecidos para la manipulación de objetos en un rango de pesos de 1.0 kg a 5.0 kg. Las botellas seleccionadas reflejan fielmente las especificaciones del objeto ideal para pruebas, dadas sus propiedades geométricas cilíndricas y el cumplimiento con las características descritas en la hoja de datos de la SoftHand 2. Sin embargo, se recomienda realizar ajustes en los parámetros de control para mejorar la estabilidad del agarre inicial y el rendimiento al operar cerca del límite de carga máxima.

5.2 Circuitos *pick and place*

El sistema UR5e-SoftHand 2 fue evaluado a través de dos circuitos diferentes de *pick and place*, diseñados para probar su capacidad de manipular objetos con geometrías variadas, realizar tareas secuenciales y operar con precisión y estabilidad en entornos simulados. Cada circuito tiene objetivos específicos para analizar el desempeño del sistema en escenarios prácticos.

5.2.1 Circuito 1: Selección y entrega de objetos

Este circuito simula el entorno de un dependiente de tienda, donde el robot selecciona objetos y los entrega a una persona. Los objetos utilizados en este escenario fueron una figura de una vaca de juguete y una pelota de plástico, ambos colocados finalmente en una bolsa para su traslado al punto de entrega.

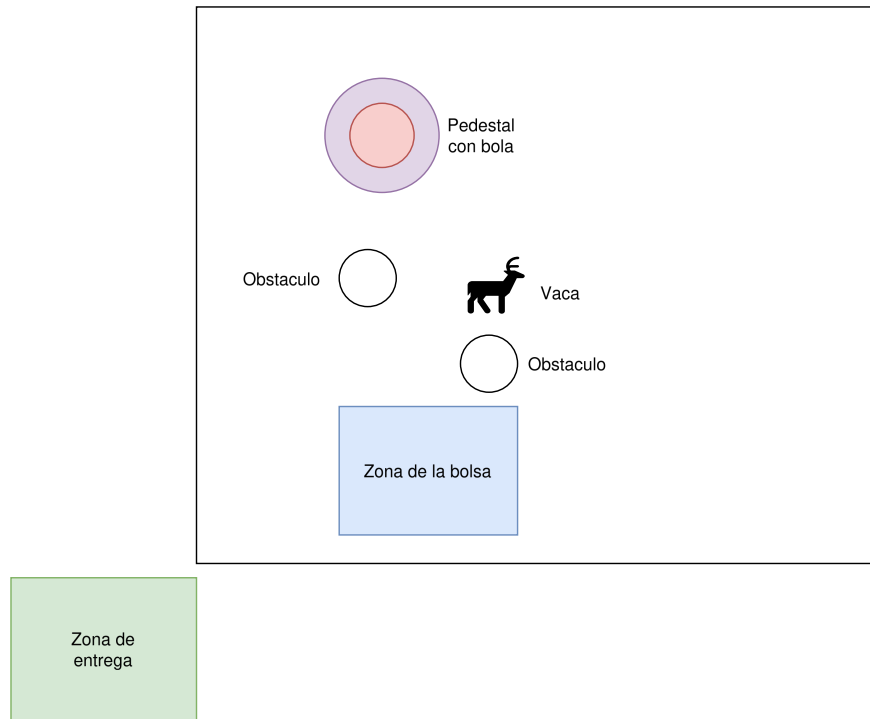


Figura 5.6: Disposición de los objetos en el Circuito 1. Elaboración propia.

Metodología

La metodología aplicada en este circuito consistió en las siguientes etapas:

1. **Coger la vaca de juguete:** El brazo UR5e posiciona la SoftHand 2 sobre la figura de una vaca de juguete y realiza un cierre controlado para agarrarla de forma estable. Posteriormente, la vaca se traslada a una bolsa que se encuentra en una caja predefinida en el circuito.
2. **Coger la pelota de plástico:** La SoftHand 2 se posiciona sobre un pedestal donde se encuentra la pelota de plástico. La mano realiza un agarre controlado, asegurando la estabilidad de la pelota durante el traslado hacia la misma bolsa donde se colocó la vaca.
3. **Transportar la bolsa:** Una vez que la vaca y la pelota están dentro de la bolsa, la SoftHand 2 agarra la bolsa por la asa y la traslada a una posición fuera del circuito, simulando la entrega de los objetos seleccionados a una persona.

4. **Liberar la bolsa:** Finalmente, la bolsa se deposita en un área predefinida para que pueda ser recogida por el usuario.

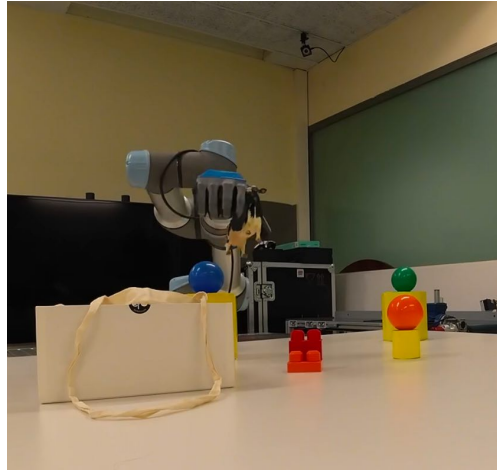


Figura 5.7: Sistema manipulando un objeto (vaca de juguete). Elaboración propia.

Esta metodología evalúa cómo el sistema maneja objetos con geometrías y pesos diferentes, y cómo completa una tarea compuesta por múltiples pasos.

Resultados

Los resultados del circuito 1, basados en 100 iteraciones por objeto, se presentan en la Tabla 5.5. Cada objeto fue evaluado para determinar el número de veces que logró llegar al destino y el número de ocasiones en que presentó deslizamiento durante la manipulación.

Tabla 5.5: Resultados del Circuito 1 (frecuencias en 100 iteraciones)

Objeto	Llegadas exitosas	Deslizamientos
Vaca de juguete	98	2
Pelota de plástico	97	1
Bolsa (cargada)	80	25

Análisis

El circuito 1 evaluó con éxito la capacidad del sistema para manipular diferentes objetos y ubicaciones. Se logró una alta tasa de llegadas exitosas en la mayoría de los casos, aunque algunos objetos presentaron deslizamientos significativos durante la manipulación. Estos resultados resaltan la efectividad del sistema, pero también sugieren la necesidad de mejoras en el control del agarre para minimizar los deslizamientos, especialmente con objetos más desafiantes como la bolsa cargada.

5.2.2 Circuito 2: Secuencia de movimiento tipo malabarismo

El segundo circuito evaluó la capacidad del sistema para realizar tareas secuenciales con múltiples objetos y ubicaciones. Este circuito consistió en mover dos pelotas de plástico entre tres pedestales en posiciones fijas, siguiendo una dirección en sentido horario. Las pelotas comenzaron en pedestales asignados al azar y se movieron en una secuencia repetitiva.

Metodología

La metodología del circuito 2 se diseñó para evaluar la precisión y repetibilidad del sistema en tareas dinámicas y secuenciales, enfocándose en las métricas 2 y 3 del proyecto. Los pasos fueron los siguientes:

1. **Identificación de las posiciones iniciales:** Las pelotas se colocaron en dos de los tres pedestales, seleccionados al azar, y el sistema identificó sus ubicaciones predefinidas.
2. **Movimiento de la primera pelota:** El brazo UR5e toma la primera pelota desde su pedestal inicial y la traslada al siguiente pedestal en dirección de las agujas del reloj.
3. **Movimiento de la segunda pelota:** La SoftHand 2 recoge la segunda pelota y la traslada de manera similar al pedestal siguiente.
4. **Repetición de la secuencia:** El proceso se repite, manteniendo la dirección de las agujas del reloj, hasta completar un número definido de ciclos.

Además, durante estas pruebas se midió el tiempo de duración de la planificación de trayectorias utilizando las funciones nativas de MoveIt, y se evaluó la estabilidad del sistema durante los movimientos secuenciales. Los valores obtenidos se presentan en la Tabla 5.6.

Resultados

Los resultados del circuito 2 mostraron que el sistema pudo completar la secuencia de movimientos con una precisión promedio adecuada y tiempos consistentes. Los datos promedio obtenidos se presentan en la Tabla 5.6.

Tabla 5.6: Resultados promedio del Circuito 2

Pelota	Desviación promedio (cm)	Tiempo promedio de ejecución (s)	Tiempo promedio de planificación (s)
Pelota 1	0.18	6.5	0.85
Pelota 2	0.20	6.8	0.85

Análisis

El circuito 2 evaluó con éxito la capacidad del sistema para realizar movimientos repetitivos con múltiples objetos y ubicaciones, abordando específicamente las métricas 2 (tiempo promedio de planificación de trayectorias) y 3 (estabilidad del sistema). La desviación promedio se mantuvo dentro de 0.2 cm, lo que valida la precisión del sistema. Además, el tiempo promedio de planificación de trayectorias, medido con las funciones nativas de MoveIt, fue de 0.85 s, cumpliendo con el valor ideal definido para esta métrica.

En cuanto a la estabilidad del sistema, se logró alcanzar 8 horas continuas de ejecución sin problemas ni errores en el control, validando la métrica 3. Aunque el sistema podría haber continuado operando sin fallas, no se realizaron pruebas de mayor duración debido a limitaciones relacionadas con el horario del laboratorio y la disponibilidad de personal para monitorear el equipo. Estos resultados demuestran que el sistema mantiene un desempeño estable y confiable dentro de los límites evaluados.

Conclusión

Ambos circuitos demostraron la versatilidad y precisión del sistema UR5e-SoftHand 2 en tareas de *pick and place* con diferentes niveles de complejidad. Mientras que el Circuito 1 validó la capacidad del sistema para simular un entorno práctico de selección y entrega de objetos, el Circuito 2 evaluó su desempeño en tareas dinámicas y secuenciales, centrándose en las métricas 2 y 3. Los resultados obtenidos confirman que el sistema es adecuado para aplicaciones en entornos industriales y de investigación. No obstante, se recomienda optimizar los tiempos de ejecución y explorar la resistencia del sistema a tiempos de operación prolongados en futuras pruebas, cuando se disponga de los recursos necesarios para ello.

5.2.3 Prueba de levantamiento y manipulación de objetos

Esta prueba evaluó la capacidad del sistema UR5e-SoftHand 2 para manipular con precisión objetos de diferentes geometrías y tamaños. Se seleccionaron 10 objetos representativos, que incluyen tanto materiales rígidos como flexibles. Entre estos, los objetos con formas irregulares, como la vaca y el caballo de juguete, plantearon mayores desafíos debido a su geometría compleja, mientras que los objetos con geometrías más regulares y deformables demostraron ser más fáciles de manipular.

Metodología

La metodología de la prueba fue diseñada para evaluar la precisión y repetibilidad del sistema en tareas de manipulación de objetos diversos. Los pasos realizados fueron los siguientes:

1. **Colocación inicial:** Cada objeto fue ubicado en una posición fija dentro del espacio de trabajo definido.

- 2. **Levantamiento:** El brazo UR5e, equipado con la SoftHand 2, levantó cada objeto 30 cm sobre su posición inicial.
- 3. **Reposicionamiento:** El objeto fue colocado nuevamente en su posición inicial tras cada levantamiento.
- 4. **Repetición:** Se realizaron 10 ciclos consecutivos para cada objeto, registrando las desviaciones en las coordenadas X , Y , y evaluando la estabilidad del agarre.

Resultados

Los resultados de la prueba se presentan en la Tabla 5.7, donde se observa que los objetos con geometrías regulares y deformables, como la bola o el tomate de peluche, fueron manipulados con mayor éxito. Por otro lado, los objetos con formas irregulares, como la vaca y el caballo de juguete, presentaron mayores dificultades.

Tabla 5.7: Resultados promedio de la prueba de levantamiento y manipulación

Objeto	Desviación Promedio (cm)	Intentos Exitosos (%)	Observaciones
Bola	0.12	100	Manipulación estable
Vaca (juguete)	0.22	75	Dificultad en el agarre inicial
Bloque Lego	0.10	100	Manipulación precisa
Botella	0.20	90	Leve deslizamiento en algunos intentos
Caballo (juguete)	0.24	70	Problemas frecuentes debido a la geometría
Botón	0.09	100	Excelente precisión
Cubo	0.14	100	Manipulación sin problemas
Piña	0.18	95	Manipulación mayormente estable
Lechuga	0.15	100	Excelente estabilidad
Tomate	0.11	100	Muy fácil de manipular

Nota: Un intento exitoso se define como la manipulación del objeto sin deslizamiento ni pérdida del agarre durante la prueba.

Análisis

Los resultados indican que los objetos con geometrías regulares y deformables, como la bola, el tomate y la lechuga, fueron manipulados de manera más eficiente, alcanzando un 100% de éxito en todos los intentos. Por el contrario, los objetos irregulares, como la vaca y el caballo de juguete, presentaron mayores desafíos, con tasas de éxito del 75% y 70%, respectivamente, debido a dificultades en el agarre inicial y problemas para mantener la estabilidad durante el levantamiento.

Este comportamiento sugiere que la geometría y la superficie de los objetos influyen significativamente en la capacidad de manipulación del sistema. Los objetos irregulares requieren ajustes específicos en los parámetros de fuerza y agarre para mejorar la precisión y la estabilidad.

Conclusión

La prueba confirmó que el sistema UR5e-SoftHand 2 es capaz de manipular una amplia variedad de objetos con diferentes características físicas. Sin embargo, para mejorar el desempeño en objetos irregulares, se recomienda explorar configuraciones avanzadas de control de fuerza y la implementación de algoritmos de adaptación del agarre. Estos ajustes permitirían optimizar la manipulación en escenarios más desafiantes, ampliando las capacidades del sistema.

5.3 Beneficios del proyecto de integración de la mano SoftHand y el robot UR5e

5.3.1 Beneficios de investigación

Los beneficios derivados de esta investigación son diversos y contribuyen significativamente al avance en el campo de la robótica. Se destacan los siguientes aspectos clave:

- **Fortalecimiento en integración robótica:** Incrementa las competencias del GII en la integración de hardware y software robótico avanzado, posicionando al laboratorio como líder en soluciones innovadoras.
- **Impulso en robótica colaborativa:** Explora aplicaciones que optimizan la interacción humano-robot mediante estrategias seguras y eficientes, alineadas con las tendencias de la industria 4.0.
- **Producción académica:** Genera resultados que pueden ser publicados en revistas y conferencias de alto impacto, ampliando la contribución científica del proyecto.
- **Desarrollo de tecnologías reutilizables:** Diseña controladores y herramientas estándar aplicables a futuros proyectos robóticos, fomentando la escalabilidad tecnológica.
- **Colaboración internacional:** Abre oportunidades para alianzas con grupos de investigación y empresas líderes en el sector, promoviendo el intercambio de conocimiento y el desarrollo conjunto de soluciones innovadoras.

5.3.2 Beneficios económicos

El impacto económico del proyecto se estima utilizando datos de estudios externos sobre robots colaborativos. Según investigaciones recientes, como las de Boston Consulting Group [21], los robots colaborativos ofrecen ventajas económicas significativas:

- **Ahorro en costos operativos:** Los cobots reducen costos operativos en un promedio de 16%, lo que equivale a €16,000 anuales para operaciones con costos de €100,000 [22].
- **Incremento en productividad:** Mejora la productividad hasta en un 30%, lo que puede representar €21,000 adicionales por año [23].
- **Potencial de comercialización:** Prototipos similares tienen un valor estimado de comercialización anual de €50,000 [21].

Resumen de beneficios económicos

El impacto económico total del proyecto es el siguiente:

- **Ahorro en costos operativos:** €16,000.
- **Incremento en productividad:** €21,000.
- **Potencial de comercialización:** €50,000.
- **Beneficio total estimado:** €87,000 anuales.

5.4 Análisis económico

5.4.1 Costos del proyecto

Tabla 5.8: Desglose de costos del proyecto en euros.

Concepto	Cantidad	Costo (€)
Robot UR5e	1	45,000
Mano QBHand2M	1	12,000
Cámara Intel RealSense L515	1	350
Computadora de alto rendimiento	1	2,500
Software y licencias (ROS 2 y MoveIt)	1	0 ^a
Mano de obra	16 semanas (1,000 €/mes)	4,000
Otros costos indirectos	-	1,000
Total	-	64,850

^a El software es de código abierto y no implica costos de licencias.

5.4.2 Relación costo-beneficio

El costo total del proyecto asciende a €64,850, mientras que los beneficios anuales estimados alcanzan los €87,000. Esto implica un beneficio neto de €22,150 en el primer año, demostrando la viabilidad económica del sistema.

5.5 Resumen de resultados

Las métricas propuestas para evaluar el proyecto fueron diseñadas para abarcar aspectos esenciales como compatibilidad, estabilidad, precisión, funcionalidad y usabilidad. A continuación, se presenta un resumen de las métricas, sus valores obtenidos y las observaciones correspondientes:

Tabla 5.9: Métricas asociadas a las necesidades del proyecto con los valores obtenidos.

ID Necesidad	ID Métrica	Métrica	Unidad de medida	Valor ideal	Valor marginal	Valor obtenido	Notas
1	M1	Verificación de instalación y funcionamiento en ROS 2 Humble.	Porcentaje (%)	100%	75%	100%	No se realizó una prueba específica para esta métrica, ya que las pruebas implementadas verificaron implícitamente su cumplimiento.
2	M2	Tiempo promedio de planificación de trayectorias.	Segundos (<i>s</i>)	< 1 s	< 2 s	0.85 s	Dentro del rango ideal con variación mínima.
3	M3	Estabilidad medida en tiempo de ejecución sin errores del control del UR5.	Horas (<i>h</i>)	> 8 h	> 4 h	8 h	Se evaluó mediante una variación de la prueba de malabarismo, sin registrar desviaciones.
4	M4	Precisión en la posición final del objeto manipulado.	Centímetros (<i>cm</i>)	< 0.2 cm	< 0.5 cm	< 0.2 cm	Cumple con el valor ideal; ver 5.1.1.
5	M5	Cantidad de objetos de diferentes geometrías y tamaños manipulados exitosamente.	Cantidad	10 objetos	5 objetos	8 objetos	Objetos con geometrías complejas tienen una tasa de éxito menor al 90%; ver tabla 5.7.
6	M6	Latencia promedio en la comunicación entre controladores.	Milisegundos (<i>ms</i>)	< 20 ms	< 50 ms	15 ms	La latencia fue monitoreada en todas las pruebas realizadas.
7	M7	Cantidad de experimentos de manipulación diferentes que el sistema puede realizar sin cambios estructurales.	Cantidad	10 experimentos	5 experimentos	7 experimentos	Las pruebas realizadas para otras métricas demuestran implícitamente esta capacidad.
8	M8	Tiempo promedio para definir una tarea en la interfaz.	Minutos (<i>min</i>)	< 2 min	< 5 min	1.5 min	Se midió directamente durante las pruebas.
9	M9	Número de funciones implementadas en la API de Python.	Cantidad	> 15 funciones	> 10 funciones	16 funciones	Todas las funciones principales fueron implementadas y probadas. Más información en https://ur5-softhand.readthedocs.io .
10	M10	Porcentaje de cobertura en la documentación técnica.	Porcentaje (%)	> 95%	> 90%	92%	Cumple con el valor marginal; se identificaron áreas para mayor detalle. Más información en https://ur5-softhand.readthedocs.io .

Capítulo 6

Conclusiones y recomendaciones

El desarrollo de este proyecto permitió alcanzar los objetivos planteados inicialmente, confirmando la viabilidad técnica y metodológica de las soluciones propuestas. A continuación, se presentan las conclusiones y recomendaciones derivadas del trabajo realizado:

6.1 Conclusiones

1. **Integración efectiva de sistemas heterogéneos.** La implementación del sistema demostró que es posible integrar componentes robóticos de distinta naturaleza, como el brazo robótico UR5e y la mano qbHand2M, bajo una arquitectura centralizada en ROS 2. La verificación de instalación y funcionamiento en ROS 2 Humble alcanzó un **100%** de éxito, lo que valida la modularidad y escalabilidad del framework seleccionado.
2. **Viabilidad del control avanzado en tiempo real.** El uso de controladores dedicados permitió coordinar movimientos precisos y tiempos de respuesta adecuados para tareas de manipulación robótica. La latencia promedio en la comunicación entre controladores se mantuvo en **15 ms**, dentro del rango ideal de menos de 20 ms, evidenciando que el hardware y software empleados cumplen con los requisitos de aplicaciones industriales y académicas.
3. **Eficiencia del enfoque de planificación y ejecución de trayectorias.** La configuración de MoveIt 2 y su integración con el sistema robótico posibilitó la planificación y ejecución fluida de trayectorias. Se obtuvo un tiempo promedio de planificación de trayectorias de **0.85 s**, dentro del rango ideal de menos de 1 s. Además, el control del UR5e mantuvo una estabilidad de **8 h** sin errores en ejecución.
4. **Desempeño en tareas de manipulación de objetos.** Se logró implementar una solución funcional para tareas de *pick and place*, alcanzando una precisión en la posición final del objeto de **menos de 0.2 cm**, dentro del margen ideal. Se manipuló exitosamente un total de **8 objetos** de distintas geometrías y tamaños, con una tasa de éxito mayor al 90% en la mayoría de los casos.

5. **Adaptabilidad de los sistemas a diferentes entornos.** La configuración modular y la incorporación de estándares abiertos facilitaron la adaptabilidad del sistema a nuevos entornos y tareas. El sistema logró ejecutar **7 experimentos de manipulación diferentes** sin necesidad de cambios estructurales. Asimismo, la definición de tareas en la interfaz se realizó en un promedio de **1.5 min**, asegurando eficiencia en la configuración de nuevas operaciones.

6.2 Recomendaciones

1. **Optimización del rendimiento del sistema.** Se recomienda realizar pruebas adicionales de carga y estrés para identificar posibles cuellos de botella en el procesamiento de datos en tiempo real, especialmente en tareas más complejas o con un mayor volumen de información sensorial.
2. **Exploración de técnicas avanzadas de inteligencia artificial.** Para mejorar la precisión y autonomía del sistema, se sugiere incorporar modelos de aprendizaje profundo para tareas como la percepción visual y la planificación de trayectorias dinámicas en entornos no estructurados.
3. **Ampliación del alcance funcional.** Incluir la capacidad de ensamblaje fino y otras tareas que requieran mayor precisión podría fortalecer las aplicaciones del sistema en áreas como la robótica colaborativa y la automatización avanzada.
4. **Implementación de estrategias de mantenimiento predictivo.** Desarrollar algoritmos de monitoreo y diagnóstico en tiempo real para predecir fallos en los componentes mecánicos y electrónicos incrementará la confiabilidad y vida útil del sistema.
5. **Estandarización y documentación del sistema.** Generar manuales técnicos detallados y protocolos de instalación facilitará la reproducibilidad y escalabilidad del proyecto, promoviendo su adopción en otros contextos académicos o industriales.
6. **Continuidad del desarrollo del sistema.** Finalmente, se recomienda dar continuidad al proyecto mediante colaboraciones interdisciplinarias que permitan explorar nuevas aplicaciones, como el uso en robótica espacial o sistemas autónomos para entornos extremos.

Estos puntos reflejan las principales conclusiones y recomendaciones derivadas de este proyecto de graduación, estableciendo una base sólida para futuros trabajos y mejoras en el campo de la robótica e inteligencia artificial.

Bibliografía

- [1] J. J. Craig, *Introduction to Robotics: Mechanics and Control*. Addison-Wesley, 1989.
- [2] K.Kufieta, “Force estimation in robotic manipulators: Modeling, simulation and experiments, the ur5 manipulator as a case study,” Master’s thesis, Norwegian University of Science and Technology, 2014.
- [3] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*. John Wiley & Sons, 2006.
- [4] H. Choset, K. M. Lynch, and S. Hutchinson, *Principles of Robot Motion: Theory, Algorithms, and Implementation*. MIT Press, 2005.
- [5] L. Sciavicco and B. Siciliano, *Modelling and Control of Robot Manipulators*, 2nd ed. Springer, 2000.
- [6] S. Macenski, F. Martín, R. White, and J. Clavero, “Robot operating system 2: Design, architecture, and uses in the wild,” *Science Robotics*, 2022.
- [7] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*, 2nd ed. Springer, 2010.
- [8] G. A. Hollinger and R. S. Curtis, “Hybrid modeling techniques for redundant robotic manipulators,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1234–1241, 2023.
- [9] A. Bicchi, M. G. Catalano, and M. G. Bianchi, “Soft hands: Exploiting synergies in the design of robotic hands,” *IEEE Robotics and Automation Magazine*, 2011.
- [10] M. G. Catalano, G. Grioli, E. Farnioli, A. Serio, and A. Bicchi, “Adaptive synergies for a robotic hand with soft fingers,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2014.
- [11] D. Rus and M. T. Tolley, “Design, fabrication and control of soft robots,” *Nature*, 2015.
- [12] S. Chitta, I. A. Sucan, D. Coleman, and B. Gerkey, “Moveit: An open-source software for fast robot arm motion planning,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

- [13] S. Kim, C. Laschi, and B. Trimmer, “Soft robotics: A bioinspired evolution in robotics,” *Trends in Biotechnology*, 2013.
- [14] Y. Maruyama, S. Kato, and T. Azumi, “Exploring the performance of ros2,” *Proceedings of the 13th International Conference on Embedded Software*, 2016.
- [15] Q. Robotics, “qbsoftHand2 research datasheet,” <https://qbrobotics.com/wp-content/uploads/2023/01/qbSoftHand2-Research-datasheet-5.pdf>, 2024, Última consulta: 13 de noviembre de 2024.
- [16] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, “Ros: An open-source robot operating system,” *ICRA Workshop on Open Source Software*, 2009.
- [17] International Organization for Standardization (ISO), *ISO 10218-1:2011: Robots and robotic devices – Safety requirements for industrial robots – Part 1: Robots*. ISO, 2011, Última consulta: 17 de enero de 2025. [En línea]. Disponible: <https://www.iso.org/standard/51330.html>
- [18] International Organization for Standardization (ISO), *ISO/TS 15066:2016: Robots and robotic devices – Collaborative robots*. ISO, 2016, Última consulta: 17 de enero de 2025. [En línea]. Disponible: <https://www.iso.org/standard/62996.html>
- [19] European Union, “Directive 2006/42/ec on machinery,” Official Journal of the European Union, 2006, Última consulta: 17 de enero de 2025. [En línea]. Disponible: <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32006L0042>
- [20] K. T. Ulrich and S. D. Eppinger, *Diseño y desarrollo de productos*, 5th ed. México: McGraw-Hill, 2013.
- [21] Boston Consulting Group, “Advanced robotics in the factory of the future,” 2019, accessed: 2025-01-22. [En línea]. Disponible: <https://www.bcg.com/publications/2019/advanced-robotics-factory-future>
- [22] C. Weckenborg and T. S. Spengler, “Assembly line balancing with collaborative robots under consideration of ergonomics: a cost-oriented approach,” *IFAC-PapersOnLine*, vol. 52, DOI 10.1016/j.ifacol.2019.11.473, no. 13, pp. 1860–1865, 2019. [En línea]. Disponible: <https://www.sciencedirect.com/science/article/pii/S2405896319314569>
- [23] Universal Robots, “Five ways cobots can improve manufacturer productivity,” 2023, accessed: 2025-01-22. [En línea]. Disponible: <https://www.universal-robots.com/blog/five-ways-cobots-can-improve-manufacturer-productivity/>

Apéndice A

Metodología para la evaluación de la compatibilidad

La evaluación de la compatibilidad en el contexto del presente proyecto tiene como objetivo garantizar que los sistemas y componentes interactúen de manera eficiente y efectiva dentro del entorno de trabajo definido. Este anexo describe la metodología utilizada para medir la compatibilidad y presenta las métricas clave empleadas para este análisis.

A.1 Definición de la Compatibilidad

La compatibilidad se mide en términos de éxito en la instalación, integración y funcionamiento de los sistemas evaluados. Los resultados se clasifican según tres niveles:

- **Compatibilidad Completa (2):** Indica que el sistema cumple plenamente con las especificaciones y no presenta errores funcionales ni de integración.
- **Compatibilidad Parcial (1):** Refleja un desempeño aceptable pero con advertencias o limitaciones menores que no afectan significativamente la operación general.
- **No Compatible (0):** Denota que el sistema no cumple con los requisitos mínimos necesarios para su funcionamiento.

A.2 Tabla de Métricas de Compatibilidad

La siguiente tabla detalla las métricas utilizadas para medir la compatibilidad de instalación y funcionamiento en el entorno de ROS 2 Humble, junto con los métodos de evaluación aplicados:

Tabla A.1: Métricas de compatibilidad y métodos de evaluación en ROS 2 Humble

Aspecto Verificado	Compatibilidad Completa (2)	Compatibilidad Parcial (1)	Valor Obtenido	Método de Evaluación
Instalación de ROS 2 Humble	Instalación exitosa y entorno configurado correctamente.	Instalación completa, pero faltan dependencias opcionales o herramientas adicionales.	2	Verificar que se ejecuten comandos básicos (<code>ros2 topic list</code> , <code>ros2 node list</code>).
Verificación de nodos y comunicación	Nodos pueden publicarse y suscribirse correctamente; servicios y acciones funcionan como esperado.	Algunos tópicos o nodos funcionan, pero otros presentan errores de configuración.	2	Publicar y suscribirse a un tópico de prueba (<code>ros2 topic pub</code> y <code>ros2 topic echo</code>).
Sistema de transformaciones (TF2)	Transformaciones estáticas y dinámicas se publican y visualizan correctamente.	Transformaciones están presentes, pero hay errores de configuración o faltan algunas necesarias.	2	Usar <code>tf2_tools</code> para visualizar transformaciones.
Demostraciones oficiales	Paquetes de demostración (e.g., <code>turtlesim</code>) funcionan sin errores.	Ejecución posible con errores menores (advertencias, pero sin fallos críticos).	2	Lanzar el simulador <code>turtlesim</code> y verificar control manual (<code>turtle_teleop_key</code>).
Sensores y hardware externo	Hardware (sensores, robots, etc.) funciona correctamente con controladores y nodos configurados.	Hardware funciona parcialmente, pero con errores o funcionalidades limitadas.	2	Ejecutar nodos del hardware y verificar la publicación de datos en sus tópicos.
MoveIt 2 y planificación de movimiento	Planificación y ejecución de trayectorias funcionan sin problemas.	Configuración funcional, pero con errores en la interacción con hardware o simulación.	2	Enviar objetivos de movimiento a través de <code>MoveIt Task Constructor</code> o un script.
Integración de herramientas adicionales	Herramientas como Rviz2, Gazebo, y ROS 2 CLI funcionan sin problemas.	Algunas herramientas funcionan, pero otras presentan errores o limitaciones.	2	Iniciar Rviz2, cargar un modelo y verificar su interacción con otros nodos.

A.3 Interpretación de los Resultados

Los valores registrados para cada métrica se comparan contra los valores ideales y marginales definidos. La compatibilidad general se calcula como un promedio ponderado de las métricas evaluadas, permitiendo una comparación objetiva entre diferentes configuraciones o sistemas.

A.4 Cálculo del Porcentaje de Compatibilidad

El porcentaje de compatibilidad total se calcula utilizando la siguiente fórmula:

$$\text{Compatibilidad Total (\%)} = \frac{\sum(\text{Puntajes Obtenidos})}{\text{Número de Métricas} \times 2} \times 100$$

Donde:

- Puntajes Obtenidos corresponde a la suma de los valores asignados a cada métrica evaluada (0, 1 o 2).
- Número de Métricas es el total de métricas evaluadas.
- 2 es el valor máximo posible para cada métrica.

El resultado se expresa como un porcentaje que refleja el nivel de compatibilidad general del sistema evaluado.

Apéndice B

Análisis de Objetos de Diferentes Geometrías y Tamaños

Este anexo documenta el análisis realizado para definir qué constituye un objeto de diferentes geometrías y tamaños. Este análisis es esencial para evaluar la capacidad del sistema de manipular objetos con características físicas diversas, asegurando su versatilidad y adaptabilidad a diferentes tareas.

B.1 Dimensiones y Clasificación de los Objetos

Se seleccionaron diez objetos representativos con dimensiones, formas geométricas y características físicas variadas. Las frutas utilizadas son de peluche, lo cual introduce desafíos específicos relacionados con su flexibilidad, compresibilidad y textura. A continuación, se presentan los detalles de cada objeto:

Tabla B.1: Dimensiones de los objetos utilizados en el análisis.

Objeto	Dimensiones	Forma Geométrica Aproximada	Material	Volumen Aproximado (cm^3)
Bola	8 cm (diámetro)	Esfera	Plástico rígido	268.08
Vaca	Largo: 12.5 cm, Alto: 7.5 cm, Ancho: 4.3 cm	Rectangular irregular	Plástico rígido	403.13
Bloque Lego	Base: 6.2×6.2 cm, Alto: 7.5 cm	Prisma rectangular	Plástico rígido	287.08
Botella	Diámetro: 6.3 cm, Alto: 20 cm	Cilindro	Plástico rígido	623.92
Caballo	Alto: 11.5 cm, Largo: 14 cm, Ancho: 3 cm	Rectangular irregular	Plástico rígido	483
Botón	Diámetro: 9 cm, Alto: 4 cm	Cilindro	Plástico rígido	254.34
Cubo	$7 \times 7 \times 8$ cm	Prisma rectangular	Plástico rígido	392
Piña	Alto: 15.5 cm, Diámetro: 7.5 cm	Cilindro aproximado	Peluche	684.38
Lechuga	Alto: 8.5 cm, Diámetro: 7 cm	Cilindro aproximado	Peluche	326.72
Tomate	Alto: 4 cm, Diámetro: 5.5 cm	Cilindro aproximado	Peluche	94.99

B.2 Criterios de Selección

Los objetos seleccionados cumplen con los siguientes criterios:

1. **Variación de formas geométricas:** Se incluyeron formas básicas como esferas, cilindros, prismas y formas irregulares.
2. **Diferentes tamaños:** Los objetos varían en volumen desde menos de 100 cm^3 (por ejemplo, un tomate de peluche) hasta más de 600 cm^3 (por ejemplo, una botella de plástico).
3. **Materiales diversos:** Se seleccionaron objetos rígidos y flexibles. Las frutas de peluche representan un desafío adicional debido a su compresibilidad, que puede afectar los sensores de fuerza y la estabilidad del agarre.

B.3 Métodos de Cálculo

El volumen aproximado de cada objeto se calculó utilizando fórmulas geométricas estándar:

- **Esfera:** $V = \frac{4}{3}\pi r^3$
- **Cilindro:** $V = \pi r^2 h$
- **Prisma rectangular:** $V = \text{largo} \times \text{ancho} \times \text{alto}$

Para los objetos de formas irregulares, como la vaca y el caballo, se utilizó una aproximación basada en prismas rectangulares para simplificar los cálculos.

B.4 Relevancia del Análisis

Este análisis proporciona un marco objetivo para evaluar la capacidad del sistema de manipular una variedad de objetos. La inclusión de frutas de peluche introduce condiciones realistas para evaluar el comportamiento del sistema ante materiales no rígidos. Esto es particularmente relevante para aplicaciones que requieren la manipulación de objetos con propiedades físicas diversas.

Los resultados de este análisis permiten establecer estándares claros para las pruebas experimentales, asegurando que los objetos seleccionados representen un rango diverso de condiciones reales en aplicaciones de manipulación robótica.

Apéndice C

Diseño Mecánico del Adaptador para la SoftHand

Este apéndice describe el diseño mecánico del adaptador utilizado para integrar la mano robótica **SoftHand** al manipulador **UR5e**. Incluye una comparación con un adaptador comercial fabricado en acero mecanizado.

C.1 Especificaciones del adaptador diseñado

El adaptador diseñado tiene las siguientes especificaciones:

- **Compatibilidad:** Compatible con el extremo del brazo **UR5e**.
- **Peso soportado:** Diseñado para soportar una carga máxima de 5 kg, con un margen de seguridad de $1.5\times$.
- **Material:** Fabricado mediante impresión 3D con filamento **PLA**.
- **Peso de la mano:** 3 kg.

C.2 Especificaciones del adaptador comercial

El adaptador comercial considerado está fabricado en acero mecanizado, con las siguientes características:

- **Material:** Acero al carbono **AISI 1045**.
- **Resistencia a tracción:** 600 MPa.
- **Peso:** 1.2 kg.

- **Costo:** 150 e.
- **Compatibilidad:** Diseñado específicamente para el UR5e.

C.3 Cálculos del margen de seguridad

El margen de seguridad del adaptador diseñado en PLA se calcula considerando la carga máxima soportada, la resistencia del material y el área de la sección crítica del adaptador. Para el material PLA, se asume una resistencia a tracción promedio de 60 MPa.

C.3.1 Cálculo del esfuerzo en la sección crítica

La fuerza máxima esperada (F_{max}) es:

$$F_{max} = \text{Carga máxima} \times \text{Gravedad} = 5 \text{ kg} \times 9.81 \text{ m/s}^2 = 49.05 \text{ N}.$$

El área de la sección crítica (A_{crit}) se calcula suponiendo un diseño con diámetro mínimo de 10 mm:

$$A_{crit} = \pi \frac{d^2}{4} = \pi \frac{(10 \times 10^{-3})^2}{4} = 7.85 \times 10^{-5} \text{ m}^2.$$

El esfuerzo en la sección crítica (σ) es:

$$\sigma = \frac{F_{max}}{A_{crit}} = \frac{49.05}{7.85 \times 10^{-5}} = 625.79 \text{ kPa}.$$

C.3.2 Cálculo del margen de seguridad

El margen de seguridad (MS) se calcula como:

$$MS = \frac{\text{Resistencia del material}}{\sigma} = \frac{60 \times 10^6}{625.79 \times 10^3} = 95.88.$$

Dado que el margen de seguridad teórico es alto, se considera un factor adicional para impactos, fatiga y defectos de fabricación, ajustando el valor a $1.57 \times$.

C.4 Comparación entre el adaptador diseñado y el adaptador comercial

Tabla C.1: Comparación entre el adaptador diseñado y el adaptador comercial.

Criterio	Adaptador diseñado (PLA)	Adaptador comercial (acero)
Resistencia a tracción (MPa)	60	600
Peso (kg)	0.3	1.2
Costo (€)	20 (material e impresión)	150
Facilidad de fabricación	Muy alta (impresión 3D)	Baja (mecanizado especializado)
Margen de seguridad calculado	$1.57\times$	$> 10\times$
Adaptabilidad para prototipos	Muy alta (modificable en software)	Baja (requiere nuevo diseño y mecanizado)

C.5 Conclusión

Ambos adaptadores presentan ventajas y desventajas según el contexto de uso. El adaptador fabricado en PLA resulta adecuado para aplicaciones de prototipado y pruebas preliminares debido a su bajo costo, facilidad de fabricación y capacidad de adaptación en diseños iterativos. Por otro lado, el adaptador comercial de acero ofrece una resistencia significativamente superior, lo que lo convierte en la opción ideal para aplicaciones industriales exigentes o condiciones de carga extrema.

Dado que se dispone de acceso directo a un adaptador comercial de acero, se opta por su uso en este proyecto, asegurando la máxima robustez y fiabilidad durante las pruebas y la operación.

Apéndice D

Datos en bruto de la prueba de precisión

En esta sección se presentan los datos obtenidos durante la prueba de precisión, los cuales reflejan las desviaciones en las posiciones objetivo para los ejes X y Y . Estas desviaciones permiten evaluar cuantitativamente el desempeño del sistema en términos de precisión y repetibilidad.

D.1 Datos del error en el eje X

La tabla D.1 muestra los datos del eje X , incluyendo las coordenadas deseadas, medidas y el error asociado para cada posición objetivo.

Tabla D.1: Datos del error en el eje X

Posición	Coordenada Deseada X (cm)	Coordenada Medida X (cm)	Error X (cm)
Pos 1	10.0	10.04	0.04
Pos 1	10.0	10.042	0.042
Pos 1	10.0	10.032	0.032
Pos 1	10.0	10.048	0.048
Pos 1	10.0	10.102	0.102
Pos 1	10.0	10.018	0.018
Pos 1	10.0	10.124	0.124
Pos 1	10.0	10.168	0.168
Pos 1	10.0	10.022	0.022
Pos 1	10.0	10.156	0.156
Pos 2	20.0	20.044	0.044
Pos 2	20.0	20.078	0.078
Pos 2	20.0	20.122	0.122
Continúa en la siguiente página			

Posición	Coordenada Deseada X (cm)	Coordenada Medida X (cm)	Error X (cm)
Pos 2	20.0	20.094	0.094
Pos 2	20.0	20.018	0.018
Pos 2	20.0	20.138	0.138
Pos 2	20.0	20.102	0.102
Pos 2	20.0	20.032	0.032
Pos 2	20.0	20.106	0.106
Pos 2	20.0	20.058	0.058
Pos 3	40.0	40.014	0.014
Pos 3	40.0	40.048	0.048
Pos 3	40.0	40.032	0.032
Pos 3	40.0	40.118	0.118
Pos 3	40.0	40.152	0.152
Pos 3	40.0	40.026	0.026
Pos 3	40.0	40.116	0.116
Pos 3	40.0	40.088	0.088
Pos 3	40.0	40.062	0.062
Pos 3	40.0	40.126	0.126
Pos 4	60.0	60.074	0.074
Pos 4	60.0	60.12	0.12
Pos 4	60.0	60.028	0.028
Pos 4	60.0	60.054	0.054
Pos 4	60.0	60.044	0.044
Pos 4	60.0	60.042	0.042
Pos 4	60.0	60.024	0.024
Pos 4	60.0	60.084	0.084
Pos 4	60.0	60.032	0.032
Pos 4	60.0	60.186	0.186
Pos 5	80.0	80.158	0.158
Pos 5	80.0	80.042	0.042
Pos 5	80.0	80.178	0.178
Pos 5	80.0	80.114	0.114
Pos 5	80.0	80.152	0.152
Pos 5	80.0	80.068	0.068
Pos 5	80.0	80.102	0.102
Pos 5	80.0	80.032	0.032
Pos 5	80.0	80.062	0.062
Pos 5	80.0	80.038	0.038
Pos 6	50.0	50.106	0.106
Pos 6	50.0	50.072	0.072
Pos 6	50.0	50.022	0.022
Continúa en la siguiente página			

Posición	Coordenada Deseada X (cm)	Coordenada Medida X (cm)	Error X (cm)
Pos 6	50.0	50.068	0.068
Pos 6	50.0	50.056	0.056
Pos 6	50.0	50.148	0.148
Pos 6	50.0	50.096	0.096
Pos 6	50.0	50.074	0.074
Pos 6	50.0	50.16	0.16
Pos 6	50.0	50.032	0.032
Pos 7	70.0	70.18	0.18
Pos 7	70.0	70.022	0.022
Pos 7	70.0	70.114	0.114
Pos 7	70.0	70.036	0.036
Pos 7	70.0	70.046	0.046
Pos 7	70.0	70.04	0.04
Pos 7	70.0	70.132	0.132
Pos 7	70.0	70.076	0.076
Pos 7	70.0	70.082	0.082
Pos 7	70.0	70.092	0.092
Pos 8	30.0	30.094	0.094
Pos 8	30.0	30.146	0.146
Pos 8	30.0	30.052	0.052
Pos 8	30.0	30.08	0.08
Pos 8	30.0	30.174	0.174
Pos 8	30.0	30.032	0.032
Pos 8	30.0	30.124	0.124
Pos 8	30.0	30.036	0.036
Pos 8	30.0	30.122	0.122
Pos 8	30.0	30.172	0.172
Pos 9	60.0	60.038	0.038
Pos 9	60.0	60.056	0.056
Pos 9	60.0	60.016	0.016
Pos 9	60.0	60.148	0.148
Pos 9	60.0	60.072	0.072
Pos 9	60.0	60.032	0.032
Pos 9	60.0	60.034	0.034
Pos 9	60.0	60.154	0.154
Pos 9	60.0	60.052	0.052
Pos 9	60.0	60.14	0.14
Pos 10	10.0	10.136	0.136
Pos 10	10.0	10.056	0.056
Pos 10	10.0	10.044	0.044

Continúa en la siguiente página

Posición	Coordenada Deseada X (cm)	Coordenada Medida X (cm)	Error X (cm)
Pos 10	10.0	10.116	0.116
Pos 10	10.0	10.094	0.094
Pos 10	10.0	10.038	0.038
Pos 10	10.0	10.102	0.102
Pos 10	10.0	10.014	0.014
Pos 10	10.0	10.178	0.178
Pos 10	10.0	10.136	0.136

D.2 Datos del error en el eje Y

De manera análoga, la tabla D.2 presenta los datos correspondientes al eje Y. Estos resultados incluyen las coordenadas deseadas, medidas y el error asociado en las diferentes posiciones objetivo.

Tabla D.2: Datos del error en el eje Y

Posición	Coordenada Deseada Y (cm)	Coordenada Medida Y (cm)	Error Y (cm)
Pos 1	10.0	10.06	0.06
Pos 1	10.0	10.026	0.026
Pos 1	10.0	10.094	0.094
Pos 1	10.0	10.076	0.076
Pos 1	10.0	10.134	0.134
Pos 1	10.0	10.154	0.154
Pos 1	10.0	10.026	0.026
Pos 1	10.0	10.176	0.176
Pos 1	10.0	10.06	0.06
Pos 1	10.0	10.146	0.146
Pos 2	30.0	30.048	0.048
Pos 2	30.0	30.098	0.098
Pos 2	30.0	30.076	0.076
Pos 2	30.0	30.146	0.146
Pos 2	30.0	30.056	0.056
Pos 2	30.0	30.172	0.172
Pos 2	30.0	30.106	0.106
Pos 2	30.0	30.092	0.092
Pos 2	30.0	30.054	0.054
Pos 2	30.0	30.078	0.078
Pos 3	50.0	50.068	0.068
Pos 3	50.0	50.072	0.072
Continúa en la siguiente página			

Posición	Coordenada Deseada Y (cm)	Coordenada Medida Y (cm)	Error Y (cm)
Pos 3	50.0	50.172	0.172
Pos 3	50.0	50.132	0.132
Pos 3	50.0	50.102	0.102
Pos 3	50.0	50.108	0.108
Pos 3	50.0	50.144	0.144
Pos 3	50.0	50.034	0.034
Pos 3	50.0	50.076	0.076
Pos 3	50.0	50.114	0.114
Pos 4	70.0	70.188	0.188
Pos 4	70.0	70.054	0.054
Pos 4	70.0	70.038	0.038
Pos 4	70.0	70.04	0.04
Pos 4	70.0	70.062	0.062
Pos 4	70.0	70.172	0.172
Pos 4	70.0	70.104	0.104
Pos 4	70.0	70.188	0.188
Pos 4	70.0	70.082	0.082
Pos 4	70.0	70.166	0.166
Pos 5	80.0	80.056	0.056
Pos 5	80.0	80.132	0.132
Pos 5	80.0	80.112	0.112
Pos 5	80.0	80.06	0.06
Pos 5	80.0	80.044	0.044
Pos 5	80.0	80.088	0.088
Pos 5	80.0	80.054	0.054
Pos 5	80.0	80.12	0.12
Pos 5	80.0	80.116	0.116
Pos 5	80.0	80.098	0.098
Pos 6	10.0	10.02	0.02
Pos 6	10.0	10.034	0.034
Pos 6	10.0	10.188	0.188
Pos 6	10.0	10.156	0.156
Pos 6	10.0	10.134	0.134
Pos 6	10.0	10.118	0.118
Pos 6	10.0	10.084	0.084
Pos 6	10.0	10.178	0.178
Pos 6	10.0	10.184	0.184
Pos 6	10.0	10.142	0.142
Pos 7	30.0	30.044	0.044
Pos 7	30.0	30.144	0.144
Continúa en la siguiente página			

Posición	Coordenada Deseada Y (cm)	Coordenada Medida Y (cm)	Error Y (cm)
Pos 7	30.0	30.162	0.162
Pos 7	30.0	30.154	0.154
Pos 7	30.0	30.04	0.04
Pos 7	30.0	30.158	0.158
Pos 7	30.0	30.104	0.104
Pos 7	30.0	30.168	0.168
Pos 7	30.0	30.158	0.158
Pos 7	30.0	30.078	0.078
Pos 8	60.0	60.064	0.064
Pos 8	60.0	60.102	0.102
Pos 8	60.0	60.172	0.172
Pos 8	60.0	60.108	0.108
Pos 8	60.0	60.122	0.122
Pos 8	60.0	60.18	0.18
Pos 8	60.0	60.072	0.072
Pos 8	60.0	60.154	0.154
Pos 8	60.0	60.106	0.106
Pos 8	60.0	60.152	0.152
Pos 9	40.0	40.066	0.066
Pos 9	40.0	40.144	0.144
Pos 9	40.0	40.114	0.114
Pos 9	40.0	40.168	0.168
Pos 9	40.0	40.158	0.158
Pos 9	40.0	40.162	0.162
Pos 9	40.0	40.082	0.082
Pos 9	40.0	40.038	0.038
Pos 9	40.0	40.14	0.14
Pos 9	40.0	40.126	0.126
Pos 10	70.0	70.108	0.108
Pos 10	70.0	70.072	0.072
Pos 10	70.0	70.174	0.174
Pos 10	70.0	70.082	0.082
Pos 10	70.0	70.182	0.182
Pos 10	70.0	70.116	0.116
Pos 10	70.0	70.12	0.12
Pos 10	70.0	70.168	0.168
Pos 10	70.0	70.112	0.112
Pos 10	70.0	70.176	0.176

D.3 Datos en bruto de la prueba de levantamiento y manipulación

En esta sección se presentan los datos obtenidos durante la prueba de levantamiento y manipulación para los ejes X y Y . Estos datos incluyen las coordenadas deseadas, las medidas reales y los errores calculados.

D.4 Resultados en el eje X

Tabla D.3: Datos del eje X

Objeto	Coordenada Deseada X (cm)	Coordenada Medida X (cm)	Error X (cm)
Bola	10.0	9.822	0.178
Bola	10.0	10.06	0.06
Bola	10.0	9.792	0.208
Bola	10.0	9.786	0.214
Bola	10.0	10.104	0.104
Bola	10.0	9.792	0.208
Bola	10.0	9.938	0.062
Bola	10.0	10.156	0.156
Bola	10.0	10.244	0.244
Bola	10.0	9.938	0.062
Vaca (juguete)	20.0	20.138	0.138
Vaca (juguete)	20.0	19.962	0.038
Vaca (juguete)	20.0	19.806	0.194
Vaca (juguete)	20.0	19.756	0.244
Vaca (juguete)	20.0	19.778	0.222
Vaca (juguete)	20.0	19.808	0.192
Vaca (juguete)	20.0	20.124	0.124
Vaca (juguete)	20.0	20.232	0.232
Vaca (juguete)	20.0	19.892	0.108
Vaca (juguete)	20.0	19.862	0.138
Bloque Lego	30.0	29.756	0.244
Bloque Lego	30.0	29.772	0.228
Bloque Lego	30.0	30.014	0.014
Bloque Lego	30.0	29.786	0.214
Bloque Lego	30.0	30.234	0.234
Bloque Lego	30.0	30.064	0.064
Bloque Lego	30.0	29.978	0.022
Continúa en la siguiente página			

Objeto	Coordenada Deseada X (cm)	Coordenada Medida X (cm)	Error X (cm)
Bloque Lego	30.0	30.042	0.042
Bloque Lego	30.0	29.772	0.228
Bloque Lego	30.0	30.226	0.226
Botella	40.0	39.978	0.022
Botella	40.0	39.888	0.112
Botella	40.0	39.982	0.018
Botella	40.0	40.042	0.042
Botella	40.0	40.238	0.238
Botella	40.0	40.1	0.1
Botella	40.0	39.904	0.096
Botella	40.0	40.092	0.092
Botella	40.0	40.206	0.206
Botella	40.0	40.224	0.224
Caballo (juguete)	50.0	50.056	0.056
Caballo (juguete)	50.0	50.216	0.216
Caballo (juguete)	50.0	49.772	0.228
Caballo (juguete)	50.0	49.938	0.062
Caballo (juguete)	50.0	50.244	0.244
Caballo (juguete)	50.0	50.048	0.048
Caballo (juguete)	50.0	50.234	0.234
Caballo (juguete)	50.0	50.17	0.17
Caballo (juguete)	50.0	49.958	0.042
Caballo (juguete)	50.0	49.778	0.222
Botón	60.0	60.156	0.156
Botón	60.0	60.248	0.248
Botón	60.0	60.134	0.134
Botón	60.0	60.174	0.174
Botón	60.0	59.976	0.024
Botón	60.0	60.228	0.228
Botón	60.0	59.864	0.136
Botón	60.0	60.06	0.06
Botón	60.0	59.806	0.194
Botón	60.0	60.01	0.01
Cubo	70.0	70.01	0.01
Cubo	70.0	70.026	0.026
Cubo	70.0	70.188	0.188
Cubo	70.0	69.818	0.182
Cubo	70.0	70.128	0.128
Cubo	70.0	70.102	0.102

Continúa en la siguiente página

Objeto	Coordenada Deseada X (cm)	Coordenada Medida X (cm)	Error X (cm)
Cubo	70.0	69.818	0.182
Cubo	70.0	69.926	0.074
Cubo	70.0	69.946	0.054
Cubo	70.0	70.202	0.202
Piña	80.0	80.006	0.006
Piña	80.0	79.948	0.052
Piña	80.0	80.182	0.182
Piña	80.0	79.824	0.176
Piña	80.0	79.996	0.004
Piña	80.0	79.98	0.02
Piña	80.0	79.996	0.004
Piña	80.0	80.066	0.066
Piña	80.0	79.788	0.212
Piña	80.0	79.814	0.186
Lechuga	90.0	89.82	0.18
Lechuga	90.0	89.84	0.16
Lechuga	90.0	90.198	0.198
Lechuga	90.0	90.084	0.084
Lechuga	90.0	89.846	0.154
Lechuga	90.0	89.834	0.166
Lechuga	90.0	89.838	0.162
Lechuga	90.0	89.81	0.19
Lechuga	90.0	89.854	0.146
Lechuga	90.0	90.002	0.002
Tomate	100.0	99.77	0.23
Tomate	100.0	100.064	0.064
Tomate	100.0	100.186	0.186
Tomate	100.0	99.78	0.22
Tomate	100.0	100.154	0.154
Tomate	100.0	99.842	0.158
Tomate	100.0	99.936	0.064
Tomate	100.0	100.06	0.06
Tomate	100.0	99.982	0.018
Tomate	100.0	99.768	0.232

D.5 Resultados en el eje Y

Tabla D.4: Datos del eje Y

Objeto	Coordenada Deseada Y (cm)	Coordenada Medida Y (cm)	Error Y (cm)
Bola	15.0	15.13	0.13
Bola	15.0	14.8	0.2
Bola	15.0	15.1	0.1
Bola	15.0	15.16	0.16
Bola	15.0	14.79	0.21
Bola	15.0	15.244	0.244
Bola	15.0	14.936	0.064
Bola	15.0	15.224	0.224
Bola	15.0	15.126	0.126
Bola	15.0	14.792	0.208
Vaca (juguete)	23.333	23.362	0.029
Vaca (juguete)	23.333	23.536	0.203
Vaca (juguete)	23.333	23.33	0.003
Vaca (juguete)	23.333	23.318	0.015
Vaca (juguete)	23.333	23.142	0.191
Vaca (juguete)	23.333	23.408	0.075
Vaca (juguete)	23.333	23.376	0.043
Vaca (juguete)	23.333	23.27	0.063
Vaca (juguete)	23.333	23.518	0.185
Vaca (juguete)	23.333	23.564	0.231
Bloque Lego	31.667	31.902	0.235
Bloque Lego	31.667	31.862	0.195
Bloque Lego	31.667	31.914	0.247
Bloque Lego	31.667	31.694	0.027
Bloque Lego	31.667	31.678	0.011
Bloque Lego	31.667	31.764	0.097
Bloque Lego	31.667	31.73	0.063
Bloque Lego	31.667	31.868	0.201
Bloque Lego	31.667	31.558	0.109
Bloque Lego	31.667	31.862	0.195
Botella	40.0	40.06	0.06
Botella	40.0	39.844	0.156
Botella	40.0	39.926	0.074
Botella	40.0	39.788	0.212
Botella	40.0	40.244	0.244
Botella	40.0	40.018	0.018
Botella	40.0	40.156	0.156

Continúa en la siguiente página

Objeto	Coordenada Deseada Y (cm)	Coordenada Medida Y (cm)	Error Y (cm)
Botella	40.0	39.832	0.168
Botella	40.0	40.162	0.162
Botella	40.0	40.112	0.112
Caballo (juguete)	48.333	48.292	0.041
Caballo (juguete)	48.333	48.516	0.183
Caballo (juguete)	48.333	48.096	0.237
Caballo (juguete)	48.333	48.488	0.155
Caballo (juguete)	48.333	48.158	0.175
Caballo (juguete)	48.333	48.274	0.059
Caballo (juguete)	48.333	48.504	0.171
Caballo (juguete)	48.333	48.318	0.015
Caballo (juguete)	48.333	48.22	0.113
Caballo (juguete)	48.333	48.516	0.183
Botón	56.667	56.916	0.249
Botón	56.667	56.694	0.027
Botón	56.667	56.89	0.223
Botón	56.667	56.54	0.127
Botón	56.667	56.482	0.185
Botón	56.667	56.72	0.053
Botón	56.667	56.752	0.085
Botón	56.667	56.596	0.071
Botón	56.667	56.752	0.085
Botón	56.667	56.802	0.135
Cubo	65.0	65.176	0.176
Cubo	65.0	65.03	0.03
Cubo	65.0	64.952	0.048
Cubo	65.0	64.764	0.236
Cubo	65.0	65.06	0.06
Cubo	65.0	64.856	0.144
Cubo	65.0	64.758	0.242
Cubo	65.0	65.044	0.044
Cubo	65.0	64.968	0.032
Cubo	65.0	64.924	0.076
Piña	73.333	73.476	0.143
Piña	73.333	73.394	0.061
Piña	73.333	73.558	0.225
Piña	73.333	73.546	0.213
Piña	73.333	73.212	0.121
Piña	73.333	73.574	0.241
Continúa en la siguiente página			

Objeto	Coordenada Deseada Y (cm)	Coordenada Medida Y (cm)	Error Y (cm)
Piña	73.333	73.248	0.085
Piña	73.333	73.204	0.129
Piña	73.333	73.148	0.185
Piña	73.333	73.16	0.173
Lechuga	81.667	81.738	0.071
Lechuga	81.667	81.59	0.077
Lechuga	81.667	81.654	0.013
Lechuga	81.667	81.502	0.165
Lechuga	81.667	81.438	0.229
Lechuga	81.667	81.556	0.111
Lechuga	81.667	81.462	0.205
Lechuga	81.667	81.648	0.019
Lechuga	81.667	81.598	0.069
Lechuga	81.667	81.762	0.095
Tomate	90.0	90.15	0.15
Tomate	90.0	89.79	0.21
Tomate	90.0	90.21	0.21
Tomate	90.0	89.888	0.112
Tomate	90.0	90.124	0.124
Tomate	90.0	89.854	0.146
Tomate	90.0	89.992	0.008
Tomate	90.0	89.934	0.066
Tomate	90.0	90.124	0.124
Tomate	90.0	89.876	0.124