Escuela Superior Politécnica del Litoral

Facultad de Ingeniería en Mecánica y Ciencias de la Producción

Sistema autónomo de organización usando visión artificial RGB-D y robot colaborativo en banda transportadora.

INGE-3002

Proyecto Integrador

Previo la obtención del Título de:

Ingeniero en Mecatrónica

Presentado por:

Daniela Andrea Espin Ramos
Nahin Alexis Martínez Salazar

Guayaquil – Ecuador
Año: 2025

Dedicatoria

Dedico este logro a mi familia, por ser mi más grande apoyo en mi vida. A mi mamá, Glenda Ramos, por motivarme a ser una mujer luchadora. A mi papá, César Espin, por inspirarme a lograr mis objetivos, aun cuando existieran adversidades. A mis abuelitos, por su amor y apoyo constante. A mi abuelita Rosario; de quien siempre he sentido su compañía a pesar de no estar físicamente. Estoy segura de que está orgullosa de mí. A mi enamorado, por su apoyo y levantarme de mis caídas cuando lo necesité. Por último y de manera muy especial, a mis gatitos, mis fieles compañeros de desveladas que siempre lograron sacarme una sonrisa.

Daniela Espin Ramos.

Dedicatoria

A mi madre, mi padre y mi hermano, quienes son un pilar fundamental en mi vida, son quienes me han dado todo y se han sacrificado para que pueda cumplir mis objetivos.

Al resto de mis familiares, quienes siempre me dan mensajes de ánimo y se muestran fascinados por mis logros sin importar que tan grandes o pequeños estos sean.

A mis amigos, quienes siempre buscan animarme en los momentos más difíciles y con quienes he tenido momentos memorables.

Nahin Martínez Salazar.

Agradecimientos

Ante todo, agradezco a Dios por guiar mis pasos y otorgarme la sabiduría necesaria para culminar esta importante etapa. Al Ph.D. Edwin Valarezo, por ser un mentor fundamental en mi formación; gracias por confiar en mi potencial desde mis inicios y por motivarme a soñar en grande. Asimismo, mi gratitud al Ph.D. Bryan Puruncajas, por su guía constante y su compromiso en impulsar este proyecto hacia la excelencia. A mis amigos, por ser mucho más que compañeros de carrera y transformar la universidad en un lugar lleno de sonrisas y recuerdos memorables que llevaré siempre en mi corazón. A todos aquellos que aportaron a alcanzar este logro, gracias.

Daniela Espin Ramos.

Agradecimientos

Agradezco primero a Dios por brindarme tiempo, salud y sabiduría para poder cumplir con mis objetivos. Agradezco también al Ph.D. Edwin Valarezo por brindar las herramientas y la ayuda necesaria para poder concluir con este proyecto. También, a la gran mayoría de profesores con quienes tuve el placer de trabajar en conjunto, les doy las gracias por sus enseñanzas y el apoyo brindado a lo largo de mi carrera universitaria. Por último, y no menos importante, a mis compañeros y, en especial, a mis amigos que conocí en las aulas de clase, les agradezco por estar siempre presentes en los momentos más difíciles, por brindarme su apoyo desinteresado y por los grandes momentos que vivimos a lo largo de este tiempo.

Nahin Martínez Salazar.

Declaración Expresa

Nosotros Daniela Andrea Espin Ramos y Nahin Alexis Martínez Salazar acordamos y reconocemos que:

La titularidad de los derechos patrimoniales de autor (derechos de autor) del proyecto de graduación corresponderá a los autores, sin perjuicio de lo cual la ESPOL recibe en este acto una licencia gratuita de plazo indefinido para el uso no comercial y comercial de la obra con facultad de sublicenciar, incluyendo la autorización para su divulgación, así como para la creación y uso de obras derivadas. En el caso de usos comerciales se respetará el porcentaje de participación en beneficios que corresponda a favor de los autores.

La titularidad total y exclusiva sobre los derechos patrimoniales de patente de invención, modelo de utilidad, diseño industrial, secreto industrial, software o información no divulgada que corresponda o pueda corresponder respecto de cualquier investigación, desarrollo tecnológico o invención realizada por nosotros durante el desarrollo del proyecto de graduación, pertenecerán de forma total, exclusiva e indivisible a la ESPOL, sin perjuicio del porcentaje que nos corresponda de los beneficios económicos que la ESPOL reciba por la explotación de nuestra innovación, de ser el caso.

En los casos donde la Oficina de Transferencia de Resultados de Investigación (OTRI) de la ESPOL comunique a los autores que existe una innovación potencialmente patentable sobre los resultados del proyecto de graduación, no se realizará publicación o divulgación alguna, sin la autorización expresa y previa de la ESPOL.

Guayaquil, 17 de octubre del 2025.

Daniela Andrea Espin Ramos

Nahin Alexis Martínez Salazar

Evaluadores

Firmado electrónicamente por:
BRYAN JOAO PURUNCAJAS MAZA
Validar únicamente con FirmaEC

Bryan Joao Puruncajas Maza, Ph.D.

Profesor de Materia

Firmado electrónicamente por:
EDWIN GIANNINE VALAREZO ANAZCO
Validar únicamente con FirmaEC

Edwin Giannine Valarezo Añazco, Ph.D.

Tutor de proyecto

Resumen

El presente trabajo aborda el diseño e implementación de un sistema de manipulación autónoma que integra visión artificial RGB-D y robótica colaborativa. El objetivo es desarrollar una solución capaz de detectar, segmentar y clasificar piezas en movimiento sobre una banda transportadora, justificándose en la necesidad industrial de optimizar la eficiencia operativa con mínima intervención humana. Para el desarrollo, se implementó una arquitectura de hardware que combinó procesamiento en una NVIDIA Jetson Nano, una cámara de profundidad Intel RealSense y un brazo robótico, comunicándose vía protocolo MQTT y tópicos de ROS2. El núcleo del sistema consiste en el trabajo conjunto del modelo de segmentación YOLOv5-seg y los algoritmos de cinemática inversa para sincronizar la detección con la actuación del robot en tiempo real. Los resultados experimentales validaron que el modelo de Deep Learning identificó objetos y predijo su posición exitosamente a través de la imagen de la cámara, demostrando una alta capacidad de generalización ante distintas categorías. Se concluye que la arquitectura de control propuesta muestra estabilidad ante variaciones de posición y velocidad, siendo capaz de garantizar una operación autónoma robusta. Para trabajos futuros, se recomienda implementar modos de calibración dinámica y sistemas de estimación de velocidad para perfeccionar el cálculo de trayectorias individuales.

Palabras Clave: Visión artificial, robótica colaborativa, industria 4.0, deep learning, automatización.

Abstract

This project addresses the design and implementation of an autonomous manipulation system that integrates RGB-D computer vision and collaborative robotics. The objective is to develop a solution capable of detecting, segmenting, and classifying moving parts on a conveyor belt, justified by the industrial need to optimize operational efficiency with minimal human intervention. For the implementation, a hardware architecture was developed combining processing on a NVIDIA Jetson Nano, an Intel RealSense depth camera, and a robotic arm, communicating via MQTT protocol and ROS2 topics. The system core integrated the YOLOv5-seg model with the inverse kinematics algorithms to synchronize detection with the robot actuation in real-time. Experimental results validated that the Deep Learning model successfully identified objects and predicted their position using the camera image, demonstrating a high capacity for generalization across different categories. It is concluded that the proposed control architecture exhibits stability against position and speed variations, being able to guarantee robust autonomous operation. For future work, implementing dynamic calibration modes and speed estimation systems is recommended to refine individual trajectory calculations.

Keywords: Computer vision, collaborative robotics, industry 4.0, deep learning, automation.

Índice general

Abreviaturas

2D	Two-Dimensional, bidimensional.
3D	Three-Dimensional, tridimensional.
AGVs	Automated Guided Vehicles, vehículos guiados automáticamente.
BOM	Bill of Materials, lista de materiales.
CAGR	Compound Annual Growth Rate, tasa de crecimiento anual compuesta.
CAPEX	Capital Expenditure, gastos de capital / inversión inicial.
CNC	Computer Numerical Control, control numérico computarizado.
CNEL EP	Corporación Nacional de Electricidad – Empresa Pública.
CPU	Central Processing Unit, unidad central de procesamiento.
CPS	Cyber-Physical Systems, sistemas ciberfísicos.
CUDA	Compute Unified Device Architecture, arquitectura unificada de cómputo de dispositivos.
CV	Computer Vision, visión por computador.
ESPOL	Escuela Superior Politécnica del Litoral.
FP16	Floating Point 16-bit, formato de punto flotante de 16 bits.
FPS	Frames Per Second, fotogramas por segundo.
GFLOPS	Giga Floating Point Operations Per Second, miles de millones de operaciones de punto flotante por segundo.
GPU	Graphics Processing Unit, unidad de procesamiento gráfico.
I4.0	Industria 4.0.
IA	Inteligencia Artificial.
IESS	Instituto Ecuatoriano de Seguridad Social.
IFR	International Federation of Robotics, Federación Internacional de Robótica.
IoT	Internet of Things, internet de las cosas.
IoU	Intersection over Union, intersección sobre unión.
ISO	International Organization for Standardization, Organización Internacional de Normalización.
JSON	JavaScript Object Notation, notación de objetos de JavaScript.
kW	Kilowatt.

kWh	Kilowatt-hora.
LBA2	Laboratorio de Bioinformática y Aprendizaje Autónomo.
LCA	Low-Cost Automation, automatización de bajo costo.
mAP@0.5	Mean Average Precision at 0.5, precisión media promedio con umbral IoU de 0.5.
MMH	Manual Material Handling, manejo manual de materiales.
MQTT	Message Queuing Telemetry Transport, protocolo de comunicación IoT.
NIOSH	National Institute for Occupational Safety and Health, Instituto Nacional para la Seguridad y Salud Ocupacional.
NMS	Non-Maximum Suppression, supresión no-máximos.
OPEX	Operational Expenditure, gastos operativos.
PRI	Periodo de Recuperación de la Inversión.
PYMEs	Pequeñas y Medianas Empresas.
RGB-D	Red, Green, Blue + Depth, color y profundidad.
ROS	Robot Operating System, sistema operativo para robots.
SaaS	Software as a Service, software como servicio.
SBU	Salario Básico Unificado.
TIR	Tasa Interna de Retorno.
TMAR	Tasa Mínima Atractiva de Retorno.
TME	Trastornos Musculoesqueléticos.
USD	United States Dollar, dólar estadounidense.
VAN	Valor Actual Neto.
YOLO	You Only Look Once, solo se mira una vez.

Simbología

(C_x, C_y)	Coordenada del centroide en el plano 2D.
C_x	Posición en el eje x del centroide de la imagen.
C_y	Posición en el eje y del centroide de la imagen.
M_c	Área total de píxeles en la máscara binaria.
M_x	Inercia de píxeles en el eje x de la imagen.
M_y	Inercia de píxeles en el eje y de la imagen.
$^{offset}_{herramienta}$	Compensación de altura debido a la longitud de la herramienta.
P_{cam}	Posición del objeto con respecto a la cámara.
P_{robot}	Posición del objeto con respecto al brazo robótico.
$r_{11}, r_{12} \dots r_{33}$	Componentes de la matriz de rotación y traslación.
R_{3x3}	Matriz de rotación de dimensión 3x3.
$R_x(\emptyset)$	Matriz de rotación en el eje x.
$R_y(\theta)$	Matriz de rotación en el eje y.
$R_z(\varphi)$	Matriz de rotación en el eje z.
T	Matriz de transformación.
T_{robot}^{cam}	Matriz de transformación del robot hacia la cámara.
T_{cam}^{robot}	Matriz de transformación de la cámara hacia el robot.
t_x	Translación en x.
t_y	Translación en y.
t_z	Translación en z.
(x, y)	Posición en el plano 2D.
(x, y, z)	Posición en el espacio 3D.
Z_{final}	Altura final.
Z_{robot}	Altura con respecto al brazo robótico.
$\emptyset$	Ángulo de rotación en el eje x.
θ	Ángulo de rotación en el eje y.
φ	Ángulo de rotación en el eje z.

Índice de figuras

Índice de tablas

Capítulo 1

1. Introducción

Durante décadas, la imagen de la automatización industrial estuvo definida por robots pesados, peligrosos y de alta velocidad; máquinas "ciegas" confinadas a jaulas de seguridad, programadas para ejecutar tareas repetitivas. Los actuales procesos industriales que incluyen robótica rígida, diseñada exclusivamente para la producción en masa, con costos de capital elevados y sin posibilidad de reconfiguraciones, hicieron que una gran cantidad de empresas no sean capaces de integrarlos en sus líneas de producción. Hoy en día surgen sistemas que invitan a las industrias a la modernización de sus procesos gracias a los robots colaborativos, también llamados “cobots”**.** Estos robots han sido diseñados con la idea de ser fáciles de programar y seguros de operar en presencia de un trabajador cerca. Su propósito fundamental no es reemplazar al operario, sino actuar como una herramienta inteligente que amplifica su productividad, encargándose de las tareas repetitivas, poco ergonómicas o que requieren una precisión constante [1].

En la industria moderna, la agilidad y flexibilidad se han convertido en características necesarias y fundamentales para la producción. Los cobots son la respuesta directa a esta demanda, permitiendo a las empresas, incluidas las pequeñas y medianas (PYMEs), adaptar sus líneas de producción a la creciente necesidad de personalización de productos en respuesta a las preferencias de sus clientes en apenas unas pocas horas. Esto hace que el mercado global de cobots experimente un crecimiento explosivo, con una proyección de tasa de crecimiento anual compuesta (CAGR) superior al 32%. Para poner este dato en perspectiva, se espera que el valor de este mercado, que era de aproximadamente 4.5 mil millones de dólares en 2023, se dispare a más de 71 mil millones para 2034 [2].

Esta rápida adopción se vio drásticamente acelerada por la pandemia de COVID-19 que expuso las profundas vulnerabilidades que presentan las cadenas de suministro. Los confinamientos y las normativas de distanciamiento social paralizaron fábricas enteras, demostrando que la dependencia exclusiva de una fuerza laboral presencial es un riesgo operativo

crítico. La robotización, y en particular la automatización flexible, se empieza a considerar como la principal estrategia para construir "resiliencia operativa". Un robot puede operar largas jornadas con supervisión limitada, amplificando la productividad del personal esencial y garantizando la continuidad de los negocios. Como respuesta directa, las instalaciones de nuevos robots industriales alcanzaron un máximo histórico, superando las 542,000 unidades en 2024, más del doble de la cifra registrada una década atrás [3].

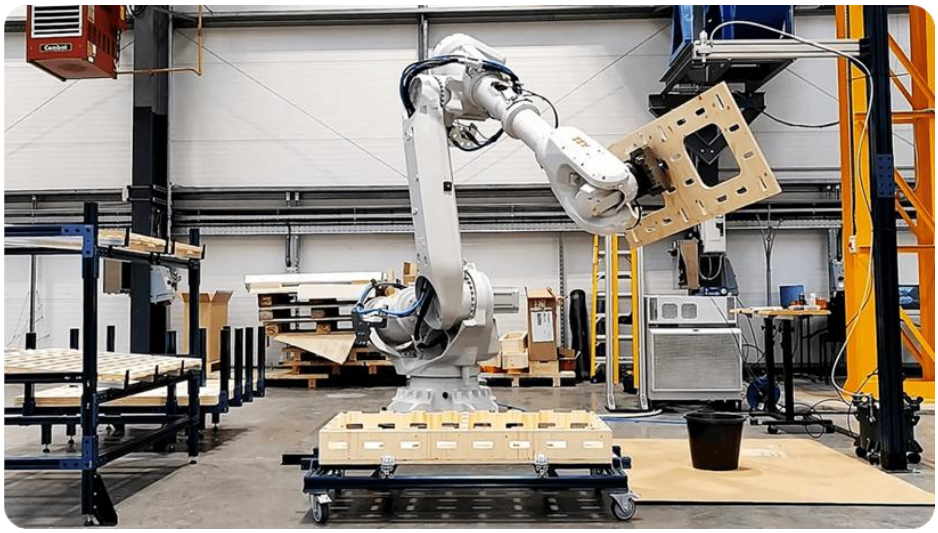
En este escenario, el creciente uso de la robótica colaborativa ha hecho que la automatización flexible trascienda, de modo que pase de ser una simple herramienta de reducción de costos a consolidarse como el pilar central de la competitividad industrial y la manifestación más tangible de la promesa de la Industria 4.0. Esta nueva realidad establece el escenario para sistemas de manufactura más avanzados, en donde la agilidad deja de ser una ventaja y se convierte en un prerrequisito para el crecimiento económico.

1.1 Descripción del problema

Actualmente, el sector industrial global experimenta una transformación digital acelerada, impulsada por la adopción de nuevas tecnologías y procedimientos modernos que la Industria 4.0 ha traído consigo. Esta tendencia ha generado la necesidad de automatizar procesos repetitivos por parte de empresas medianas y grandes, con el objetivo principal de estandarizar la calidad de sus productos y optimizar su eficiencia operativa. En respuesta, muchas industrias optan por implementar soluciones robóticas en sus líneas de producción, como brazos manipuladores, para llevar a cabo operaciones de reposicionamiento y clasificación de objetos [4]. Un ejemplo de estos manipuladores se observa a continuación en la figura 1.

Figura 1

Ejemplo de robot manipulador [5].



Nota. Este robot es utilizado en microfábricas robóticas impulsadas por visión artificial.

Debido a la fuerte competencia existente en el mercado, las empresas han iniciado la búsqueda de soluciones que permitan la mejora de sus procesos y la rapidez con la que estos se realizan, por lo que ya se han implementado sistemas automatizados para satisfacer sus demandas, sin embargo, estos sistemas presentan una desventaja y es que no suelen ser flexibles, es decir que, la línea de producción se diseña únicamente para realizar una tarea con un producto en específico. Los principales problemas que enfrentan este tipo de sistemas son los tiempos de inactividad por reprogramación para cambiar de producto, además del hecho de que operan en "entornos muy controlados", lo que significa que al no estar diseñados para ser escalables ni para permitir la integración de nuevas tecnologías, cualquier cambio en el producto o la adición de un nuevo componente, como un sistema de visión 3D, requiere un complejo y costoso reequipamiento mecánico (*retooling*), en lugar de una simple reconfiguración de software.

Estudios recientes sobre los tiempos de inactividad en procesos de manufactura y operaciones logísticas estiman que, en promedio, una hora de paro representa pérdidas cercanas a $260 000 dólares por hora [6]. En sectores de alta rotación y demanda, como el sector automotriz, el impacto económico de los períodos de inactividad puede incrementarse de manera significativa, alcanzando valores aproximados de $2.3 millones de dólares por hora [7].

A esto se suma otro factor crítico en los procesos de manufactura tradicional: la alta sensibilidad al costo de la mano de obra. En industrias donde la manipulación, el ensamblaje y el empaquetado dependen exclusivamente del personal, los costos laborales no solo representan una fracción significativa de los costos operativos totales, sino que actúan como una barrera directa a la escalabilidad [8]. Esto se debe a que, al incrementar las jornadas laborales, la empresa está obligada a incurrir en pagos de horas extra. Esto provoca que el costo de la producción en tiempo extra se dispare, afectando drásticamente los márgenes de ganancia y haciendo que la expansión sea económicamente inviable.

Aquí es donde un sistema de manufactura robotizado presenta su ventaja competitiva más fuerte. Aunque la inversión de capital inicial es considerablemente alta, su costo operativo es bajo y, lo más importante, es escalable. El costo de extender la jornada de producción con un robot es mínimo, consistiendo principalmente en el consumo de energía y un ligero aumento en el mantenimiento preventivo [9]. Un robot puede operar de forma continua por largas jornadas sin que su eficiencia se vea alterada. Esta ventaja es el principal motivo por lo que estos sistemas son escalables en comparación a las empresas que dependen del trabajo manual, ya que, las empresas automatizadas pueden absorber una mayor demanda de producción con un impacto casi nulo en sus costos variables [10].

El desafío global de modernizar las líneas de producción se manifiesta de forma particular en Ecuador. La industria nacional, si bien es un pilar importante de la economía, se encuentra en una fase inicial de adopción de la Industria 4.0. Un reporte publicado por NextGen en 2024 indica que la industria ecuatoriana “está dando sus primeros pasos” en esta área y la gran mayoría de sus procesos siguen siendo manuales y, por ende, propensos al error humano por agotamiento [11]. Esta brecha tecnológica presenta una oportunidad para desarrollar sistemas que no solo automaticen, sino que lo hagan de manera inteligente y adaptable para realizar distintas tareas con variedad de productos, aplicando correctamente el concepto de un sistema flexible en manufactura [12].

Teniendo en cuenta este contexto, el Laboratorio de Bioinformática y Aprendizaje Autónomo (LBA^2) de la ESPOL busca crear un prototipo de un sistema robotizado que resuelva las variables de interés, como la precisión de la clasificación y la carga de procesamiento. Para ello, se debe utilizar percepción avanzada (cámara RGB-D Intel Real Sense D435i) para la segmentación 3D de objetos, además debe ser fácilmente reprogramable para nuevos productos. A esto se suman consideraciones de hardware, las cuales exigen que el sistema de control sea portable y opere en tiempo real en el microprocesador Jetson Nano, con una comunicación de baja latencia para mantener coordinación entre el módulo de visión y el brazo robótico. El propósito es, por tanto, un sistema escalable que utilice inteligencia artificial para adaptarse a las necesidades del sector industrial local.

1.2 Justificación del problema

Los procesos industriales que realizan actividades de clasificación y manipulación de objetos son conocidos por involucrar tareas monótonas y repetitivas que pueden generar agotamiento y accidentes dado la naturaleza tediosa de las mismas. Para solucionar esto, muchas industrias usan sistemas de automatización pensados para una sola tarea que terminan rindiendo en términos de velocidad, pero que carecen de flexibilidad.

Un ejemplo de un sistema “rígido” en la industria ecuatoriana son las líneas de embotellado de alta velocidad, como las implementadas por empresas de bebidas como Arca Continental en Guayaquil, cuya referencia se observa en la figura 2. Estas líneas pueden procesar hasta 400 botellas por minuto con una eficiencia extraordinaria [13]. Sin embargo, esta eficiencia depende de su rigidez ya que están diseñadas mecánicamente para un solo tipo de envase. Si la producción requiere un cambio a un formato de botella o lata diferente, la línea completa debe detenerse para una reprogramación entera del sistema.

Figura 2

Línea de producción automatizada para un solo tipo de producto [13].

Por otra parte, en la industria plástica ecuatoriana, gran parte de los procesos de control de calidad y clasificación de accesorios de tuberías, como uniones o codos, aún dependen significativamente de la intervención humana debido a su abundante cantidad de producción, como se evidencia en la figura 3. Si bien esto permite una flexibilidad inicial y aprovecha la disponibilidad de mano de obra local, conlleva limitaciones inherentes a la fatiga y subjetividad del operador. Estudios sobre manufactura industrial indican que la inspección manual no es infalible; incluso cuando se revisa el 100% de la producción, las tasas de error en la detección de posibles defectos en el producto final pueden oscilar entre el 20% y el 40% debido a factores humanos y ambientales [14]. Esta brecha en la eficiencia no solo compromete la uniformidad de la producción, sino que también representa un cuello de botella productivo frente a las tecnologías de automatización que, las cuales garantizan una reducción drástica de los desperdicios operativos.

Figura 3

Línea de clasificación de accesorios de tuberías.

Para empresas que están en transición hacia el uso de tecnología en sus procesos, como es el caso de muchas en la industria ecuatoriana, estos sistemas tradicionales representan un elevado costo a la hora de readaptarse a cambios en la producción [11], dando como resultado que muchas organizaciones opten por el trabajo manual para mantener la flexibilidad operativa.

La ventaja de un operario humano es que puede ser instruido verbalmente para que trabaje con un nuevo producto o tarea con una agilidad que los sistemas convencionales no permiten. Sin embargo, esta flexibilidad intrínseca de las personas introduce una gran desventaja: el error. La fatiga y la desconcentración son factores inevitables en tareas monótonas a largo plazo, y sus consecuencias son evidentes. A nivel mundial, se estima que el error humano es responsable de hasta un 80% de las desviaciones en procesos logísticos y de un 23% del tiempo de inactividad en procesos de manufactura [15]. En la práctica, esta merma en la precisión se traduce directamente en errores de clasificación, aumentando los costos por desperdicio de material, necesidad de reprocesamiento e incluso llegando a generar una fractura en la relación de confianza con el cliente cuando el producto no cumple sus expectativas.

Adicionalmente, exponer a los trabajadores a movimientos repetitivos durante jornadas extensas es una causa directa de trastornos musculoesqueléticos (TME) y fatiga ocupacional. Agencias como el Instituto Nacional de Seguridad y Salud Ocupacional (NIOSH) de Estados

Unidos, identifican las tareas de “Manejo Manual de Materiales” (MMH) como una causa principal de problemas musculares [16]. Los trabajadores que realizan estas tareas están sujetos a riesgos por movimientos repetitivos, posturas incómodas y levantamiento forzado, lo cual, al realizarse de manera continua termina conduciendo a un alto índice de lesiones, en su mayoría en la espalda y zona lumbar, e incluso llegando a ser causa de accidentes graves en el ámbito laboral [17]. Como consecuencia, no sólo se afecta al bienestar y salud del trabajador, sino que también se generan costos altos para las empresas por el aumento en el nivel ausentismo y baja productividad por parte del personal.

Desde una perspectiva económica, la dependencia de la mano de obra humana para tareas flexibles presenta una limitación estructural debido a que la producción está directamente relacionada con los horarios establecidos. Si normalmente un turno equivale a 8 horas de trabajo, en caso de que el mercado demande que la producción se duplique, un turno ya no será suficiente. La empresa se ve forzada a contratar un segundo turno de personal, lo cual involucra duplicar la mayoría de costos relacionados a la mano de obra como salarios, beneficios, etc. En contraste, un sistema flexible basado en un robot colaborativo, si bien requiere una inversión inicial alta, puede operar en múltiples turnos con un costo adicional muy bajo, compuesto principalmente por consumo de electricidad y mantenimiento. Por ende, esta inversión inicial, aunque alta, no se compara con el costo recurrente y permanente de contatar un turno adicional.

Por estas razones, surge la necesidad de un prototipo que resuelva este dilema entre flexibilidad y precisión. La industria moderna requiere sistemas que combinen la capacidad de adaptación de un operarador humano con la precisión, velocidad y resistencia de un sistema robótico. El desarrollo de un sistema con visión autónoma basado en inteligencia artificial, como el que se propone, representa un avance tecnológico necesario para impulsar la competitividad, la calidad del producto y la seguridad laboral en la industria ecuatoriana.

1.3 Objetivos

1.3.1 Objetivo general

Diseñar e implementar un sistema de manipulación autónoma que integre visión artificial por medio de una cámara de profundidad RGB-D, un modelo de IA de clasificación y segmentación de objetos y el control de un brazo robótico colaborativo para manipular piezas en movimiento sobre una banda transportadora con la finalidad de crear un prototipo de sistema de manufactura flexible robotizado que pueda utilizarse en la industria.

1.3.2 Objetivos específicos

1. Analizar los requerimientos funcionales del sistema, los recursos computacionales y hardware necesarios para el diseño, integración e implementación de los sistemas de visión RGB-D y control del robot colaborativo probándolo en un proceso de clasificación automatizada.
2. Desarrollar e implementar un algoritmo de visión artificial capaz de detectar, segmentar y clasificar piezas en movimiento en tiempo real utilizando técnicas de procesamiento de imágenes y aprendizaje profundo.
3. Integrar y sincronizar la imagen de video RGB-D, el modelo de IA, la banda transportadora y el robot colaborativo para garantizar la operación segura y eficiente del sistema, aun cuando este esté expuesto a perturbaciones y condiciones dinámicas.
4. Evaluar el desempeño del sistema implementado en términos de precisión de clasificación, tiempo de respuesta y eficiencia operacional, comparando los resultados con métodos tradicionales de manipulación manual o semiautomática usados en sistemas de manufactura.

1.4 Marco teórico

1.4.1 Sistemas de manufactura flexible

Definición y objetivo de un sistema de manufactura flexible

En el ámbito de la automatización industrial, un Sistema de Manufactura Flexible (FMS) se define como un sistema de producción diseñado para adaptarse de manera eficiente y con mínima intervención humana a los cambios en el tipo de producto a fabricar, las cantidades de producción o las secuencias de ensamblaje o manufactura [18]. El objetivo de un FMS es superar la rigidez de la producción en masa tradicional, donde las líneas de ensamblaje son altamente eficientes pero incapaces de manejar variaciones sin importar lo pequeñas que sean; y la ineficiencia de la producción por lotes, que son flexibles pero lentos [19]. Por lo tanto, un FMS está diseñado para ser rentable en entornos de producción de "alta mezcla y bajo volumen" (*High-Mix, Low-Volume*), que es precisamente la demanda del mercado moderno que la Industria 4.0 busca satisfacer.

Arquitectura de un FMS

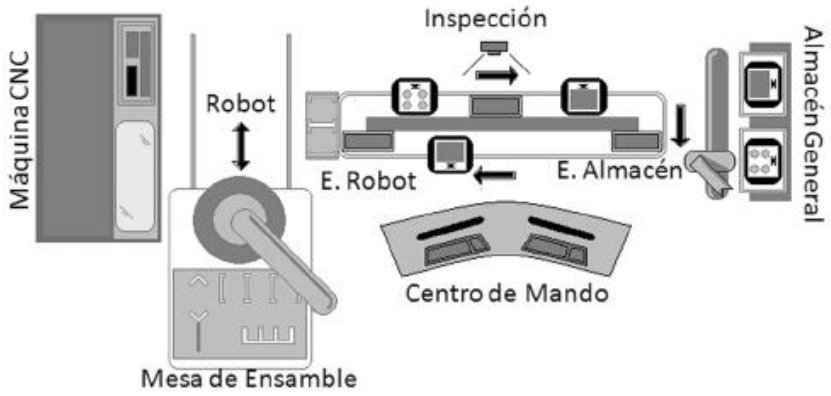
La arquitectura de un FMS se compone en tres subsistemas principales. El primero es el subsistema de procesamiento, que incluye estaciones de trabajo reconfigurables como robots industriales, centros de maquinado (CNC) y estaciones de inspección. El segundo subsistema consiste en el manejo de materiales, buscando conectar físicamente las estaciones de trabajo, usualmente mediante bandas transportadoras o Vehículos Guiados Autónomos (AGVs), para mover piezas y herramientas entre ellas [20].

Sin embargo, el componente que define al FMS es el tercer subsistema, que consiste en llevar a cabo tareas de control computarizado para coordinar todas las operaciones en la línea de producción o manufactura. En el contexto de la Industria 4.0, este control pasa de ser un programa fijo para convertirse en un sistema inteligente y perceptivo, es decir, que sea capaz de identificar posibles eventos que pueden llegar a generar fallas en la línea de producción y, a partir de ello, logre tomar las decisiones correctas para prevenirlo [21]. La figura 4 permite entender a detalle

cómo se compone la arquitectura de un FMS incluyendo los subsistemas previamente mencionados.

Figura 4

Ejemplo de arquitectura básica de un FMS [22].



1.4.2 Industria 4.0

La industria 4.0 y los sistemas ciberfísicos

La Cuarta Revolución Industrial, o Industria 4.0 (I4.0), representa un cambio de paradigma fundamental en la manufactura. A diferencia de la Industria 3.0, que se centraba en la automatización en masa de tareas repetitivas, la I4.0 busca crear sistemas de producción flexibles, inteligentes y conectados [23]. Su objetivo no es solo automatizar, sino crear "fábricas inteligentes" capaces de adaptarse en tiempo real a los requerimientos, tales como la personalización de productos, optimización de recursos o realizar procesos de forma eficiente [24].

Esta transición se fundamenta en la integración de un conjunto de tecnologías clave, entre ellas, el Internet de las Cosas (IoT), la Inteligencia Articial (IA), el *cloud computing* y, como pilar conceptual, los Sistemas Ciberfísicos (CPS) [23].

Tecnologías principales y su interacción

Para que el CPS funcione de manera inteligente depende de la interacción de otras tecnologías mencionadas previamente. Este proceso comienza con el IoT, que actúa como el cerebro principal del CPS al componerse de una red de sensores como cámaras que capturan datos

del entorno físico y del estado de la máquina. Esta captura continua genera un volumen masivo de datos estructurados y no estructurados, conocido como Big Data, los cuales, por sí solos, no tienen valor, razón por la cual deben ser procesados mediante herramientas de Analítica Avanzada (*Big Data Analytics*).

Este proceso permite filtrar el ruido, identificar patrones y extraer información útil en tiempo real, como la forma o posición de un objeto en el espacio. Por último, la implementación de IA y técnicas de aprendizaje profundo utilizan esta información ya procesada para tomar decisiones complejas y autónomas, tales como la clasificación de objetos, la planificación de trayectorias o la predicción de fallos.

Contexto de la I4.0 en Ecuador

En el contexto ecuatoriano, la adopción de tecnologías 4.0 ha sido identificada como un factor clave en el crecimiento del sector industrial [24]. Sin embargo, su implementación enfrenta grandes desafíos, especialmente en las pequeñas y medianas empresas (PYMEs). Estudios recientes sobre las PYMES industriales en Ecuador, particularmente en Guayaquil y Azuay, confirman que el nivel de adopción de la tecnología 4.0 es aún “medio” o “bajo” [25] [26]. Los principales obstáculos identificados son los costos elevados de implementación y mantenimiento, la falta de infraestructura tecnológica adecuada y la complejidad técnica, que requiere personal altamente capacitado [19].

1.4.3 Visión por computador

La Visión por Computador (CV) es una disciplina de la Inteligencia Artificial (IA) cuyo objetivo es replicar y automatizar la capacidad de la visión humana, permitiendo que las computadoras "perciban", interpreten y extraigan información significativa a partir de datos visuales, como imágenes y videos [27]. En el contexto industrial, la CV es el componente sensorial que dota de inteligencia y flexibilidad a los sistemas automatizados.

Para que un sistema robótico opere de forma autónoma en un entorno dinámico, como en conjunto con un robot móvil o una banda transportadora, requiere la habilidad de percibir e

interpretar las distintas variables que se presentan a su alrededor. Esta capacidad es proporcionada por la visión computarizada.

A diferencia de la automatización tradicional, la cual depende de que los objetos estén en posiciones fijas, exactas y predefinidas, los sistemas con visión por computador permiten manejar la incertidumbre, es decir, son capaces de reaccionar de forma óptima ante posibles desajustes o interferencias [28].

1.4.4 Percepción de profundidad 3D

Limitación de la percepción 2D en robótica

Un sistema de manipulación robótica no puede operar eficazmente solo con una cámara de color estándar (RGB), esto debido a que una imagen 2D es fundamentalmente plana, lo que significa que el sistema solo tiene información sobre la forma y el color de un objeto en coordenadas de píxel (u, v). El sistema carece de la información más crítica para la tarea de agarre o sujeción, la cual es la profundidad o distancia, que se mide por medio de la coordenada z. Sin esta tercera dimensión, no es posible saber qué tan lejos debe de extenderse un actuador o brazo robótico para alcanzar un objeto [27].

Las cámaras de profundidad resuelven este problema al capturar la geometría tridimensional del entorno, ya que, estos dispositivos generan un mapa de profundidad, donde el valor de cada píxel ya no representa un color, sino una distancia física. Es esta información tridimensional la que permite al sistema de control robótico calcular las coordenadas x, y, z precisas del objeto, situándolo en el espacio del mundo real.

Principio de funcionamiento: Visión Estéreo Activa

Existen varias tecnologías para la adquisición de profundidad, una de las más robustas y extendidas es la Visión Estéreo Activa. Este método imita la visión binocular humana utilizando dos sensores para calcular la profundidad de los objetos mediante la disparidad, es decir, la diferencia de perspectiva entre las dos vistas.

La visión estéreo se puede clasificar en dos tipos: pasiva y activa. Por su parte, la Visión Estéreo Pasiva es conocida por solo usar luz ambiental, razón por la que falla en superficies sin textura como paredes blancas o piezas de plástico lizas. Por otro lado, está la Visión Estéreo Activa, que es la más popular debido a su principio de funcionamiento. En este tipo de visión, un proyector de infrarrojos (IR) emite constantemente una retícula de puntos de luz invisibles, con la cual se crea una textura artificial en la superficie de los objetos. Dos cámaras IR se encargan de calcular la disparidad que hay entre sus imágenes al momento de captar los rayos IR y de esa forma se determina la distancia de los objetos en relación con la cámara [22].

Cámara Intel RealSense D435i

La cámara Intel RealSense D435i es una implementación comercial de la visión estéreo activa. Este dispositivo integra dos sensores de profundidad IR, un proyector IR y un sensor de color RGB en una sola unidad compacta. Su principal ventaja es la capacidad de generar nubes de puntos 3D densas y precisas a alta velocidad, lo cual es muy efectivo para aplicaciones en tiempo real [23].

El aspecto más importante de este dispositivo es su capacidad para alinear y registrar los datos de la imagen a color 2D con los datos del mapa de profundidad 3D. Esta alineación es el puente técnico que permite al modelo de IA identificar un objeto en la imagen de color (RGB) y, de inmediato, consultar su posición 3D exacta desde el mapa de profundidad para un agarre robótico preciso.

1.4.5 Robótica colaborativa

A diferencia de la robótica industrial tradicional, que opera a altas velocidades en celdas de seguridad aisladas, la robótica colaborativa permite que los trabajadores tengan una interacción más cercana con los principales actuadores robóticos. Un robot colaborativo, o "cobot", es un brazo robótico diseñado explícitamente para interactuar de forma segura y directa con operadores humanos en un espacio de trabajo compartido [29]. Esta seguridad intrínseca de los cobots se logra mediante sus diseños ligeros con formas redondeadas y, lo más importante, equipados con

avanzados sensores de fuerza y torque en las articulaciones con el fin de detectar colisiones inesperadas, inclusive, un leve contacto con un operario, y detenerse instantáneamente [9].

Esta relación humano-robot es fundamental para los objetivos de la Industria 4.0, ya que, elimina la necesidad de barreras físicas y permite una sinergia donde los humanos aportan cognición y adaptabilidad, mientras que los cobots aportan precisión, resistencia y repetitividad [9].

Las características clave que definen a un robot colaborativo incluyen la flexibilidad y la programación sencilla, estos robots pueden ser reconfigurados y reasignados a nuevas tareas con rapidez, a menudo mediante interfaces intuitivas o "programación por demostración". Esto contrasta fuertemente con la reprogramación compleja de los robots industriales tradicionales [30].

A un nivel más avanzado, la Industria 4.0 también requiere una fluida relación robot-robot para crear Sistemas Flexibles de Manufactura (FMS) completamente integrados. Esta interacción, a menudo gestionada como un "sistema multi-agente", es crucial para tareas de ensamblaje complejas o el traspaso de productos entre estaciones. Para gestionar esta complejidad se usan enfoques de control que permitan coordinar las acciones realizadas por los robots. Uno de ellos hace referencia a un control centralizado, donde un único "cerebro" maestro toma todas las decisiones y las asigna a los robots "esclavos". Alternativamente, existe un enfoque descentralizado o distribuido que otorga autonomía a cada robot, permitiéndoles coordinar sus acciones y tareas basándose en comunicación directe entre ellos, lo que ofrece una mayor robustez frente a fallos [31].

Capítulo 2

2. Metodología.

En este capítulo se detalla el proceso realizado para diseñar, comparar y seleccionar la arquitectura del software utilizada para el sistema de manipulación autónoma propuesto. Se debe tener en cuenta que varios de los componentes de hardware fueron requeridos por el cliente, por lo que no se aborda la selección de la mayoría de estos, sino en su integración.

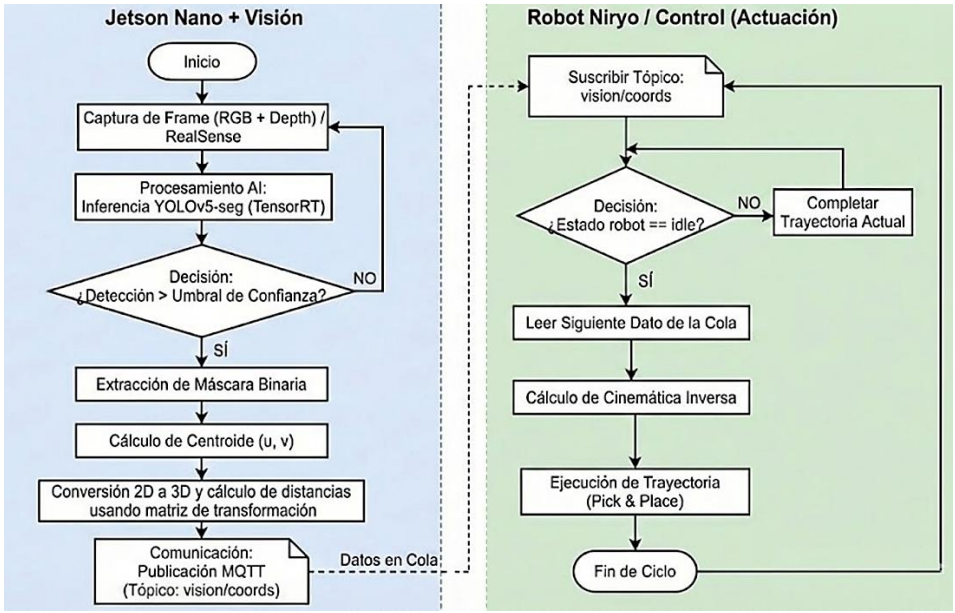
Para lograr la manipulación de las piezas en movimiento sobre la banda transportadora, se implementó una estructura de procesamiento secuencial que integró los datos de video y profundidad, provenientes de la cámara Intel RealSense D435i, con la cinemática del brazo robótico Niryo One. Esta información fue utilizada para la estimación de las posiciones de las piezas, cálculo que es ejecutado por el microprocesador NVIDIA Jetson Nano.

2.1 Metodología del diseño

El proceso de diseño de este sistema se fundamentó principalmente en la metodología iterativa, caracterizada por la constante evaluación y mejora de cada uno de sus módulos. Los resultados de cada fase determinaron si se avanzaba o si se reestructuraba la fase actual o alguna de las anteriores. En la figura 5 se presenta el flujo de trabajo y los criterios de decisión.

Figura 5

Diagrama de flujo de diseño.



2.1.1 Requerimientos de hardware

La mayor parte de los componentes usados en este proyecto fueron dispuestos por el cliente, por lo que usarlos en el prototipo del sistema era un requerimiento por cumplir.

Para la unidad central de procesamiento se utilizó la placa de desarrollo NVIDIA Jetson Nano mostrada en la figura 6, permitiendo ejecutar la red neuronal directamente en el equipo, sin depender de conexiones a servidores externos. Esta placa incorpora una GPU basada en la arquitectura Maxwell con 128 núcleos CUDA, capaz de entregar una potencia de cálculo de 472 GFLOPS. Esta capacidad es el mínimo técnico requerido para procesar inferencias de modelos modernos como YOLO en tiempo real.

Figura 6

Microprocesador Jetson Nano [32].

Respecto a la adquisición de imágenes, se trabajó con la cámara de profundidad Intel RealSense D435i, cuya imagen de referencia se muestra en la figura 7. La ventaja de este modelo sobre una cámara web convencional es su tecnología de obturador global (*Global Shutter*) en el sensor de profundidad, la cual es una característica esencial cuando se trabaja con bandas transportadoras, ya que, permite capturar objetos en movimiento sin que sufran deformaciones ni desenfoque. El dispositivo ofrece un rango de operación óptimo que inicia en los 0.3 metros, cubriendo perfectamente la distancia de trabajo hacia la banda, y posee un campo de visión amplio de 87° x 58° que permite visualizar toda el área de clasificación.

Figura 7

Cámara de profundidad RGB-D Intel RealSense [33].

Por su parte, la tarea de manipulación física se realiza mediante el robot Niryo One mostrado en la figura 8. Este es un brazo robótico colaborativo de 6 grados de libertad impulsado por motores paso a paso el cual tiene una capacidad de carga útil de 300 gramos, siendo adecuada para el peso de los accesorios de tubería usados en el proyecto. El robot ofrece un radio de alcance máximo de 440 mm y una repetibilidad de ±1mm, lo cual garantiza la precisión necesaria para interceptar las piezas detectadas por el sistema de visión. Asimismo, su controlador interno basado en Raspberry Pi facilitó la integración en la red local para la recepción de comandos.

Figura 8

Robot colaborativo Niryo One [34].

Por último, se utilizó el kit de banda transportadora (*Conveyor Belt Kit*) de la plataforma de Dobot Magician, esta banda se puede observar en la figura 9. Este módulo posee dimensiones de 700 mm de largo y 215 mm de ancho, ofreciendo un área de trabajo ideal considerando el tamaño de las piezas a utilizar. La banda ofrece velocidades dentro del rango de 0 a 120 mm/s,

esto gracias a que es accionada mediante un motor paso a paso que permite un control preciso de su rapidez. Esta característica es de gran utilidad para realizar pruebas con distintos rangos de velocidades para determinar la precisión y robustez del sistema diseñado.

Figura 9

Módulo de banda transportadora de Dobot Magician [35].

2.1.2 Arquitectura de software

El diseño lógico del sistema se construyó buscando la modularidad y la eficiencia en el uso de los recursos limitados de la placa embebida. Dado que el sistema operativo nativo de la Jetson Nano (JetPack 4.6 basado en Ubuntu 18.04) presenta incompatibilidades con las versiones modernas de ROS, se implementó la solución mediante Docker. Esta herramienta permitió crear un entorno virtual aislado, es decir, un contenedor donde se instaló ROS2 Humble, permitiendo además el acceso directo al hardware de la GPU sin afectar la configuración base del equipo.

Por otro lado, para la gestión del modelo de IA, no se ejecutó directamente en su *framework* original, sino que se utilizó SDK NVIDIA TensorRT, siendo esta una de las herramientas más importantes en el proyecto ya que permitió realizar dos procesos de optimización: la fusión de capas de la red neuronal y la calibración de precisión a FP16 (16 bits). Al reducir la precisión numérica de 32 a 16 bits, se logró aprovechar la arquitectura de los núcleos de la Jetson Nano, reduciendo el consumo de memoria y aumentando significativamente la velocidad de detección (FPS) sin sacrificar la exactitud del modelo.

Finalmente, la comunicación se fundamentó en el protocolo MQTT, lo que permitió a la Jetson Nano publicar mensajes ligeros en formato JSON. Estos incluían únicamente la clase de la pieza

y sus coordenadas cartesianas, asegurando una sincronización óptima entre la detección visual y el movimiento mecánico.

2.1.3 Análisis de alternativas y selección de estrategias

Antes de definir la arquitectura final, se evaluaron distintas configuraciones de software y hardware para garantizar la compatibilidad con el microprocesador Nvidia Jetson Nano, teniendo en cuenta que este dejó de recibir actualizaciones oficiales, limitando el uso directo de versiones recientes de librerías para Deep Learning y ROS2.

A- Selección de arquitectura de control

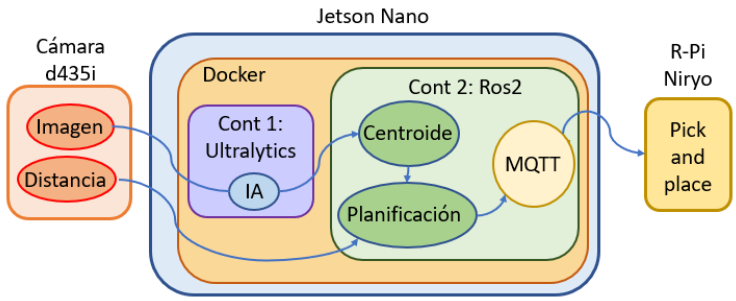
Para enfrentar las dificultades mencionadas previamente, se decidió diseñar y someter a prueba tres arquitecturas con el fin de analizar sus ventajas y desventajas.

Primera arquitectura: Uso de varios contenedores

La primera propuesta se detalla en la figura 10, en ella, se usó una arquitectura con 2 contenedores, uno destinado a usar las dependencias de Python necesarias para ejecutar el modelo YOLOv8-seg mientras que en el segundo se instaló ROS2 para gestionar la lógica de control. El objetivo de esta propuesta fue comunicar ambos contenedores, sin embargo, esta opción presentó una alta complejidad al momento de transmitir los datos.

Figura 10

Esquema de la arquitectura #1.

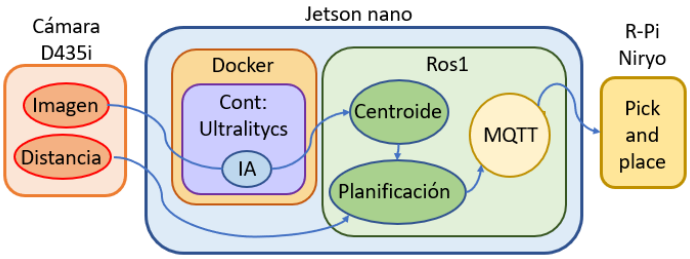


Segunda arquitectura: Comunicación contenedor-host

Como segunda opción se intentó simplificar la topología empleando un único contenedor encargado de manejar el modelo de visión YOLOv8-seg con las librerías de Python necesarias. Además, se implementó ROS (primera versión) dentro del OS nativo del procesador. Esta configuración mostrada en la figura 11, resolvió los problemas de compatibilidad, pero la complejidad de transmisión de datos entre el contenedor y el OS siguió persistiendo.

Figura 11

Esquema de la arquitectura #2.

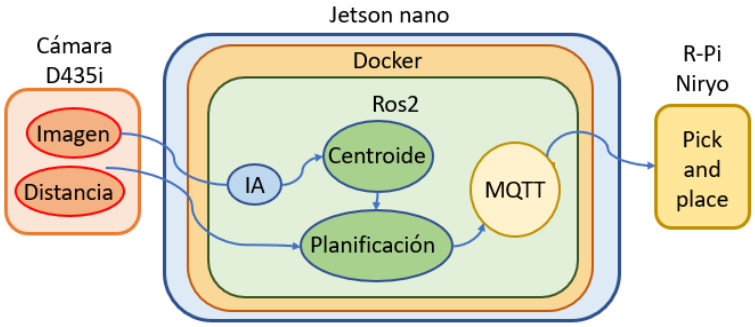


Tercera arquitectura: Contenedor único

Como última opción implementada, se usó un contenedor con ROS2 con el fin de priorizar la facilidad de programación y comunicación entre los distintos nodos del sistema. Se presentó un problema de compatibilidad con las librerías de Python necesarias para el uso de YOLOv8-seg por lo que, para resolver esta dificultad, se empleó el modelo YOLOv5-seg junto con el optimizador TensorRT. La ventaja de esta arquitectura fue su facilidad para comunicar todos los nodos entre sí, sin embargo, la precisión del modelo se vio ligeramente afectada. Su descripción detallada se observa en la figura 12.

Figura 12

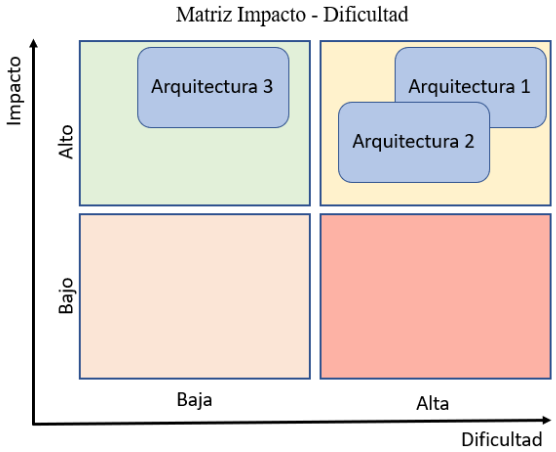
Esquema de la arquitectura #3.



Tras definir las distintas alternativas, se realizó una comparación mediante un análisis de Impacto-Dificultad con el fin de ponderar la complejidad de la implementación de cada una de las opciones disponibles. Gracias a esto se pudo seleccionar de forma objetiva la estructura más idónea para los requerimientos del sistema. En la figura 13 se muestra la comparativa realizada en la matriz

Figura 13

Matriz Impacto – Dificultad.



Como se observa en la matriz, la arquitectura 3 fue la solución más idónea para los requerimientos del proyecto. Esta alternativa fue seleccionada al demostrar un equilibrio entre rendimiento y fiabilidad, a la vez que presenta sencillez en su implementación, facilitando así la integración y el mantenimiento del código.

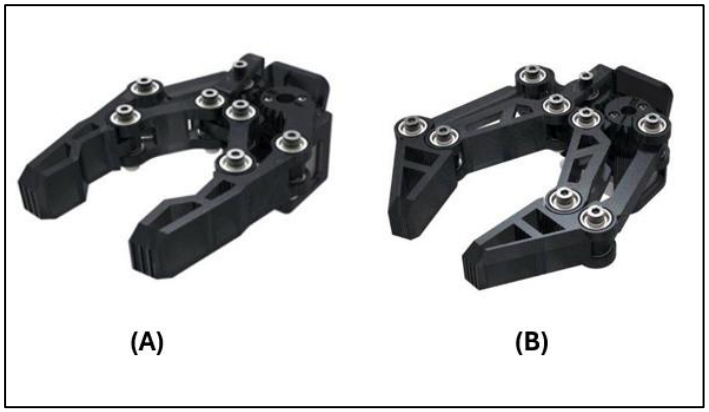
B- Selección del efector final (Gripper)

De manera paralela a la definición de la arquitectura de software, se llevó a cabo la selección del efector final del robot. Si bien la carga útil fue un factor a considerar, el criterio crítico para esta aplicación fue la capacidad de manipular objetos con geometrías curvas como lo son los accesorios de tubería.

Para tomar la decisión, se evaluaron dos alternativas nativas del fabricante: el *Large Gripper* y el *Adaptive Gripper.* A continuación, se observan ambos manipuladores en la figura 14.

Figura 14

Clase de gripper usados en el robot.



Nota. A: Large Gripper. B: Adaptive Gripper [22].

Se utilizó la técnica de criterios ponderados donde se asignaron distintos pesos a las características según su prioridad en el contexto de este proyecto.

Tabla 1

Matriz de decisión ponderada para la selección del efector final

Criterios de Selección	Peso (%)	Opción A: Large Gripper	Opción B: Adaptive Gripper
Adaptabilidad a geometrías curvas	40%	2	5
Rango de apertura	30%	3	5
Facilidad de integración	20 %	5	5
Peso	10 %	5	3
Total	100 %	3.2	4.8

Nota. Tabla realizada tomando en cuenta los requerimientos del sistema.

Como se observa en la Tabla 1, el análisis determinó que el Adaptive Gripper fue la opción idónea. A diferencia del modelo estándar, cuyas pinzas planas paralelas tienden a resbalar sobre superficies cilíndricas, el diseño de dedos flexibles del modelo adaptativo permitió un agarre envolvente que aseguró la estabilidad a la hora de recoger y reubicar las piezas. Además, su mayor rango de apertura proporcionó el margen de seguridad necesario para compensar las pequeñas variaciones de posición inherentes al desplazamiento de la banda transportadora.

2.1.4 Análisis de costos y viabilidad del proyecto

Luego de definir la arquitectura de control y la integración de los componentes físicos, se realizó un análisis económico del sistema con el objetivo dimensionar el costo de implementación del prototipo propuesto. A continuación, la tabla 2 detalla el desglose de costos de los materiales utilizados.

Tabla 2

Desglose de costos del sistema

Componente	Costo
Niryo One	$1 700
Jetson Nano	$200
Cámara Intel RealSense D435I	$350
Banda Dobot Magician	$500
Diseño del sistema de manipulación	$200
Total	$2 950

2.1.4.1 Proyección de inversión para escalamiento industrial (CAPEX)

Para la implementación comercial en un entorno de producción real, se descarta el uso de componentes educativos en favor de hardware certificado bajo normativa ISO 10218-1 (Robots industriales), por ello, se propone una arquitectura de automatización de bajo costo (LCA) que garantiza robustez operativa las veinticuatro horas del día.

La inversión inicial (I_o) requerida para un cliente final se detalla en la tabla 3:

Tabla 3

Inversión inicial industrial (CAPEX)

Rubro	Detalle	Costo Estimado (USD)
Cobot Industrial	Brazo colaborativo 6 ejes, Payload 3-5kg	$7 500
Visión y Cómputo	PC industrial + Cámara + Iluminación	$1 500
Integración mecánica	Gabinete de control, banda, acoples	$1 500
Ingeniería	Montaje, programación, seguridad	$1 000
Total Inversión (I_o)		$11 500

2.1.4.2 Costos operativos (OPEX) y ahorros proyectados

Para determinar la viabilidad financiera, se evalúa el sistema bajo un escenario de máxima exigencia: operación continua de 3 turnos (24 horas). En dicho contexto, se sobreestimaron los costos operativos para garantizar la sostenibilidad del flujo de caja. Los costos anuales de mantener la celda robótica y el sistema se detallan en la tabla 4:

Tabla 4

Resumen de costos operativos (OPEX)

Variable	Descripción de cálculo	Costo Anual (USD)
C_e	0.5 kW × 8760 h × $0.15/kWh	$1 300
C_m	10.5% del CAPEX ($11500)	$1 200
C_s	Soporte SaaS: $100 mensuales	$1 200
Total OPEX anual (C_{op})		$3 700

- **Energía eléctrica (C_e):** Se determinó mediante el producto de la potencia consumida por la celda robótica (0.5 kW) por las horas operativas anuales (8 760 horas). Para el costo unitario, se aplicó una tarifa de seguridad de $0.15/kWh (36% superior al promedio industrial de CNEL EP), resultando en un presupuesto anual de $1 300 de consumo energético.

- **Mantenimiento industrial (C_m):** Se calculó como el 10.5% de la inversión inicial, un porcentaje que duplica el estándar del 4% sugerido por la Federación Internacional de Robótica (IFR), obteniendo como resultado $1 200 anuales para mantenimiento.
- **Servicio de soporte SaaS (C_s):** Corresponde a una cuota de servicio de $100 mensuales, es decir, $1 200 por año; por ingeniería post-venta que incluye el monitoreo remoto de salud del sistema y re-entrenamiento del modelo de ser necesario.

2.1.4.3 Ahorro en costos laborales

El beneficio económico del proyecto se fundamenta en la sustitución de operación manual de clasificación bajo un régimen de 24 horas. El cálculo del ahorro anual se basa en la normativa laboral ecuatoriana vigente (2025), desglosando las siguientes variables:

- **Costo anual por operario:** Se determinó un costo de $7 925 por trabajador, considerando el Salario Básico Unificado (SBU), la aportación patronal y las provisiones obligatorias de ley.
- **Ahorro anual:** Para mantener la misma disponibilidad del robot, se requiere cubrir tres turnos rotativos. Por tanto, el ahorro total resulta de multiplicar el costo unitario por las 3 plazas laborales liberadas, generando un beneficio de $23 775 anuales sin considerar recargos por horas extras o nocturnas.

2.1.4.4 Indicadores financieros y rentabilidad

El flujo de caja neto anual se obtiene sustrayendo el OPEX ($3 700) del ahorro bruto ($23 775), resultando en un flujo positivo de $20 075. Al proyector este flujo a 5 años con una Tasa de Descuento (TMAR) del 12%, se obtienen los indicadores mostrados en la tabla 5:

Tabla 5

Indicadores de rentabilidad (escenario 24 horas)

Indicador	Resultado
Inversión Inicial	$11 500
Flujo Neto Anual	$20 075
Período de recuperación (PRI)	0.6 años
Valor Actual Neto (VAN)	$60 865
Tasa Interna de Retorno (TIR)	173%

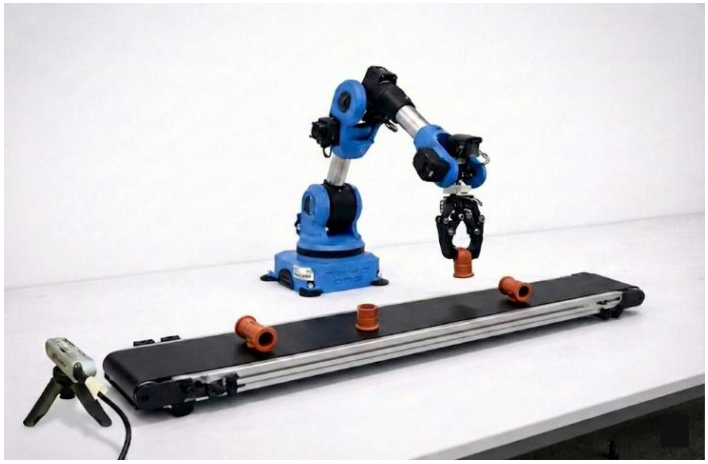
El TIR de 173% puede parecer un valor de rendimiento atípico, pero realmente es producto de cambiar a un esquema de tres turnos, teniendo en cuenta que el costo fijo del activo se diluye ya que el CAPEX no varía si la celda trabaja más o menos horas. Esto permite que el sistema autofinancie su adquisición en menos de un período fiscal, demostrando sostenibilidad y viabilidad.

2.2 Disposición de los componentes

Se estableció la disposición de los componentes del sistema siguiendo un esquema parecido a la de una celda de manufactura, donde la ubicación de cada componente fue determinada a conveniencia para facilitar la ejecución de las pruebas y los procesos del sistema.

Figura 15

Disposición de los elementos para realizar pruebas al sistema de control.



La ubicación de los distintos elementos es la siguiente:

- **Cámara RGB-D:** Fue colocada en un extremo inicial de la banda, es decir, a unos 10 cm del lado donde las piezas comienzan su trayectoria moviéndose, desde la perspectiva de la figura 15, de izquierda a derecha, esto con la finalidad de tener un ángulo de visión completo tanto de la banda como la de los objetos que van sobre ella. Esta configuración facilita la obtención de la posición de las piezas a lo largo de la banda.
- **Manipulador robótico Niryo One:** Se encuentra ubicado a un costado de la banda y cerca del final de esta, más específicamente, a 40 cm desde el inicio y a 20 del final. Su ubicación responde a la necesidad de que el robot debe tener el máximo espacio de trabajo posible y que sea capaz de alcanzar hasta el último sitio en donde las piezas se trasladaran.
- **Banda transportadora:** Para esta configuración, la banda funciona como referencia y los demás componentes se ubican dependiendo de la posición de este elemento.

Comprender la ubicación de los componentes y el flujo de trabajo es un punto importante para la programación de los nodos encargados de la estimación de la posición de las piezas en movimiento con respecto los marcos de referencia de cada uno de los elementos del hardware.

2.3 Implementación del sistema de visión y arquitectura en ROS2

El punto de partida del diseño del sistema de manipulación fue implementar el módulo de visión artificial y a partir de ello se creó la estructura de programación con la cual se incorporó el modelo a la unidad de procesamiento. Luego de establecer la arquitectura se procedió a la creación de los subsistemas encargados de la estimación de posición y comunicación con el controlador del brazo robótico.

2.3.1 Dataset y detalles de entrenamiento del modelo.

En la etapa inicial del desarrollo se implementó un subsistema de visión por computadora que emplea el modelo YOLOv5, el cual es orientado a la clasificación y segmentación de instancias

en tiempo real. Este modelo, como se mencionó anteriormente, fue escogido gracias a su compatibilidad con los programas y dependencias a usar en el sistema de control a implementar.

Para garantizar la adaptabilidad del algoritmo al entorno operativo, se construyó un dataset con 3 732 imágenes. Estos datos fueron particionados para el proceso de aprendizaje siguiendo una proporción de 70/20/10, es decir, 3 330 imágenes fueron destinadas al entrenamiento (*train*), 270 para la validación (*validation*) y 132 reservadas para las pruebas finales (*test*).

Bajo esta estructura de datos, se realizó el entrenamiento del modelo configurando parámetros como el tamaño de lote (*batch size*) de 16, una resolución de entrada de 640x640 píxeles y una duración de 100 épocas para asegurar la convergencia del modelo. La selección de estos hiperparámetros responde a la necesidad de mantener un balance entre la precisión de detección y la velocidad de procesamiento computacional. La resolución de 640x640 se estableció como el punto de equilibrio para preservar información de la imagen necesaria para la extracción de características, sin sobrecargar el procesamiento a realizar.

Esta arquitectura demostró tener los índices de precisión y velocidad de inferencia más altos, además, su compatibilidad con el sistema operativo del microprocesador lo hicieron adecuado para convertirse en la base del sistema de control a implementar.

2.3.2 Arquitectura de nodos en ROS2

Para organizar el funcionamiento del sistema, se dividió el software en cuatro nodos independientes. Cada uno se encargó de una tarea específica y enviaba el resultado al siguiente, permitiendo que el proceso se ejecutara de forma secuencial y ordenada, además de que, con esta estructura se facilitó la detección de errores y mejoró la velocidad de respuesta, ya que cada nodo se dedicó a una única función.

Para la programación se decidió usar dos lenguajes distintos según la necesidad de cada etapa. Los tres primeros nodos se programaron en C++, ya que manejaban imágenes y cálculos matemáticos que requerían la máxima velocidad posible. Mientras que el último nodo se programó

en Python debido a que este lenguaje facilitó la conexión a internet y el envío de mensajes hacia el robot.

Nodo de inferencia: Detección y segmentación.

Este nodo actuó como el cerebro visual del sistema, encargándose de ejecutar la inferencia y segmentación de las piezas mediante el modelo de IA optimizado con TensorRT. Su funcionamiento interno se estructuró en tres etapas secuenciales que permitieron transformar la imagen cruda en datos útiles para el robot.

A. **Gestión de memoria y carga del modelo:** Al arrancar, el software leyó el archivo del motor de inferencia y reservó el espacio necesario en la memoria de la tarjeta gráfica asegurando que el procesamiento de video no congelara el resto del sistema y permitiendo que la CPU y la GPU trabajaran en paralelo de manera eficiente.
B. **Preprocesamiento de la imagen:** Dado que las capturas de la cámara no podían ingresar directamente a la red neuronal, el nodo las redimensionó a un formato de 640x640 píxeles, la cual es la resolución con la que se llevó a cabo los entrenamientos del modelo, y se normalizó sus valores de color.
C. **Fase de inferencia y reconstrucción:** Una vez procesada la imagen en la GPU, el modelo entregó datos crudos que requirieron una decodificación en dos niveles. Por un lado, se utilizó el filtro de Supresión de No-Máximos (NMS) para descartar las detecciones repetidas y definir la ubicación del objeto. Por otro lado, para reconstruir la silueta de las piezas, el sistema creó una máscara binaria que separó la pieza del fondo.

Nodo para el cálculo del centroide.

La función de este nodo fue convertir la información bidimensional en un punto ubicado dentro del espacio tridimensional. Esto se logró fusionando los datos de la máscara binaria, el

mapa de profundidad y las detecciones de la red neuronal para localizar cada pieza en tres dimensiones.

El proceso inició con una sincronización temporal aproximada que alinea los mensajes entrantes de los distintos tópicos para asegurar la coherencia de los datos. Una vez recibidos, el algoritmo calculó el centroide geométrico de las máscaras binarias, del cual se obtiene un píxel que representa el centro de masa de un objeto en 2D, si este píxel carece de datos válidos, es decir, se encuentra fuera de la máscara, se busca el píxel más cercano alrededor de él.

Para la obtención del centroide, este nodo emplea una función encargada de iterar sobre la matriz de las máscaras binarias y obtener un valor ponderado con respecto a cada uno de sus ejes, este proceso es similar al que se realiza para obtener el centro de masa de un objeto.

$$M_c = \sum_x \sum_y I(x,y) \tag{1}$$

$$M_x = \sum_x \sum_y x \cdot I(x,y) \tag{2}$$

$$M_y = \sum_x \sum_y y \cdot I(x,y) \tag{3}$$

En la ecuación 1 se describe el primer paso a realizar, el cual consta de calcular el "área" de la máscara. La función $I(x,y)$ da como resultado 1 si el píxel en la posición (x,y) forma parte de la máscara binaria, caso contrario el resultado es 0. Este proceso se repite para todos los pixeles que conforman la matriz y el resultado es acumulado.

En la ecuación 2 se realiza un proceso similar, el resultado de la función $I(x,y)$ es multiplicado por el valor *x* del píxel con coordenadas (*x,y*), este proceso se repite para todos los pixeles que conforman la matriz de la máscara binaria y el resultado de cada uno de estos

subprocesos son acumulados. Se realiza el mismo procedimiento en la ecuación 3, con la diferencia de que se emplea el valor de *y* del píxel a analizar.

$$C_x = \frac{M_x}{M_c} \tag{4}$$

$$C_y = \frac{M_y}{M_c} \tag{5}$$

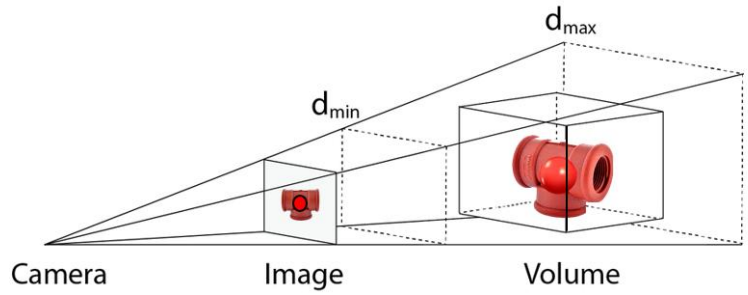
Luego, se promedian cada uno de los valores de M_x y M_y con respecto a M_c, tal como se indica en las ecuaciones 4 y 5, el resultado es la coordenada (C_x, C_y) que indica la posición en el que se encuentra el píxel que representa el centroide de ese objeto.

A continuación, para obtener la clase del objeto, se identifica la posición de la máscara y se la asocia con la caja delimitadora, o *bounding-box*, más cercana detectada por el nodo de inferencia. De esa forma, se combina la información necesaria para análisis posteriores, la forma o máscara del objeto con su etiqueta o clase obtenida a partir del *bounding-box.*

Posteriormente, se extrajo el valor de distancia a partir de la imagen de profundidad obtenida de la cámara. La coordenada o píxel del centroide 2D (*x,y*) previamente calculado es proyectado en una matriz de vectores de posición 3D y se busca un punto que posea las 2 primeras coordenadas idénticas a las del centroide, es decir, un punto que contenga (*x,y,z*), siendo *x* y *y* su posición en la imagen plana y *z* su profundidad. Con este proceso, se obtiene tanto la ubicación del centroide en la imagen de video y la distancia a la que se encuentra ese punto con respecto a la cámara. Este proceso se ejemplifica en la figura 16.

Figura 16

Ejemplo de la proyección del centroide de 2D a 3D [22].



Nodo para la transformación de coordenadas.

Este componente se encargó de homologar los espacios de trabajo del sistema, resolviendo la diferencia de posicionamiento entre la cámara y la base del robot. Para que el manipulador pudiera interactuar con los objetos detectados visualmente, fue indispensable traducir las coordenadas mediante operaciones de álgebra lineal que permitieron unificar ambos marcos de referencia.

En este sentido, el fundamento matemático detrás de la estructura de este nodo se basó en la aplicación de Matrices de Transformación Homogénea de dimensión 4x4. Estas estructuras permiten encapsular en una sola operación tanto la rotación R, necesaria para alinear la orientación de los ejes cartesianos, como la traslación t, que representa la distancia física entre los orígenes de los dispositivos. La matriz resultante obedece la estructura general que se muestra en la figura 17.

Figura 17

Estructura general de la matriz de transformación usada.

$$T = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Para la construcción de esta matriz, la rotación se gestionó internamente mediante una librería encargada de realizar cálculos con cuaterniones con el fin de evitar caer en singularidades matemáticas. Esta función se encarga de recibir los valores de los ángulos de rotación $\emptyset$, θ, φ, los cuales pertenecen a los ejes *x, y, z* respectivamente, y devuelve los valores de la matriz de R.

$$R_{3x3} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \tag{6}$$

$$R_{3x3} = R_z(\varphi) \cdot R_y(\theta) \cdot R_x(\emptyset) \tag{7}$$

$$R_x(\emptyset) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos(\emptyset) & -sen(\emptyset) \\ 0 & sen(\emptyset) & cos(\emptyset) \end{pmatrix} \tag{8}$$

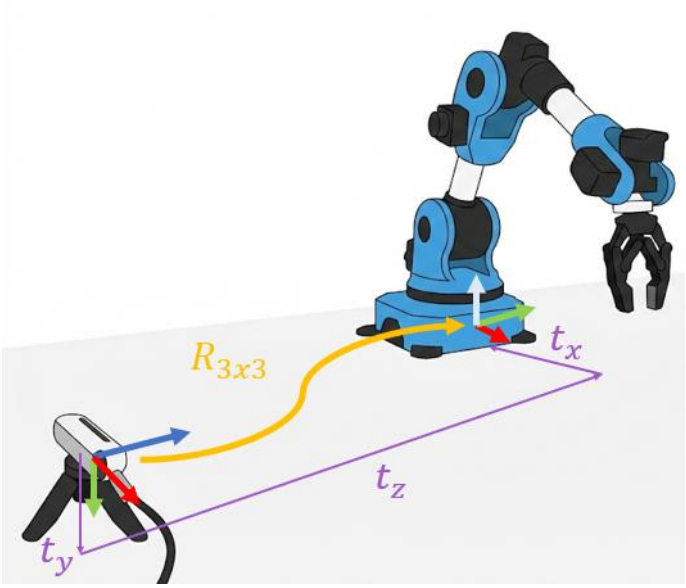
$$R_y(\theta) = \begin{pmatrix} cos(\theta) & 0 & sen(\theta) \\ 0 & 1 & 0 \\ -sen(\theta) & 0 & cos(\theta) \end{pmatrix} \quad (9)$$

$$R_z(\varphi) = \begin{pmatrix} cos(\varphi) & -sen(\varphi) & 0 \\ sen(\varphi) & cos(\varphi) & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (10)$$

El proceso matemático empleado consiste en hallar la matriz de rotación en cada uno de los ejes, tal como se observa en las ecuaciones 8, 9 y 10, y estos resultados son usados para el cálculo de la matriz R, tal como se muestra en la ecuación 7. En conjunto, la descripción del espacio de trabajo con sus respectivos sistemas de referencia y vectores de posición se detallan en la figura 18.

Figura 18

Transformación de coordenadas.



Nota. La matriz de transformación se encarga de cambiar el marco de referencia de la cámara a la del brazo robótico

Para la traslación, se empleó un vector en donde sus valores de t_x, t_y y t_z hacen referencia a la distancia a la que se encuentra los sistemas de referencia a analizar, en este caso, el de la cámara con el del manipulador robótico.

En la configuración específica del sistema, se definieron los ángulos de Euler que describen la orientación física de la cámara respecto al robot, siendo el giro en el eje vertical (Yaw=90°) el parámetro más significativo para alinear ambos sistemas.

Asimismo, un detalle importante de la implementación fue la dirección vectorial de la operación. Inicialmente, las mediciones físicas definieron la posición del robot respecto a la cámara (T_{cam}^{robot}); sin embargo, el control del brazo requería la operación inversa para proyectar los puntos visuales hacia el espacio del manipulador. Por consiguiente, el algoritmo resolvió esto calculando automáticamente la inversión matricial al iniciar el nodo:

$$T_{robot}^{cam} = (T_{cam}^{robot})^{-1} \tag{11}$$

Posteriormente, durante el ciclo de ejecución, cada punto detectado $P_{cam} = [x, y, z, 1]^T$ se multiplicó por la matriz resultante para obtener la coordenada objetivo mediante la relación:

$$P_{robot} = T_{robot}^{cam} \cdot P_{cam} \tag{12}$$

Una vez obtenida la coordenada referenciada a la base del robot, se aplicó una corrección aritmética final en el eje vertical (Z). Dado que el sistema de visión detectaba la superficie superior de la pieza, un movimiento directo a dicha coordenada habría provocado una colisión mecánica. Para evitarlo, se introdujo un parámetro de compensación que ajustó la altura de destino:

$$Z_{final} = Z_{robot} + offset_{herramienta} \tag{13}$$

Nodo para la comunicación.

Este último componente actuó como el puente entre el sistema embebido en el microprocesador y el robot, encargándose de filtrar las detecciones para transmitir únicamente las que eran válidas para la recolección. Su función principal consistió en monitorear la ubicación de los objetos y servir como “disparador” para enviar el mensaje que inicie el movimiento del robot.

El proceso comenzó verificando la posición de la pieza en la banda transportadora; cuando esta ingresaba a la zona de recolección establecida, el sistema habilitaba el envío de datos. Una vez validada la ubicación, la información de las coordenadas y el tipo de objeto se organizó en un

formato JSON ligero. Finalmente, este paquete se envió mediante el protocolo MQTT, asegurando que el robot recibiera una instrucción única gracias a un temporizador de seguridad que evitó mandar órdenes repetidas para una misma pieza.

2.4 Integración y algoritmo de control del manipulador

Tras definir la estructura lógica y de procesamiento, se continuó con la última parte para la integración total del sistema: el control del manipulador. En este paso se realizó la programación del controlador del brazo Niryo One, habilitándolo para establecer la comunicación con la unidad de procesamiento central y a partir de ello realizar las rutinas de manipulación y clasificación de piezas.

2.4.1 Comunicación y recepción de comandos

En el programa incorporado al controlador del brazo robótico, se implementó un cliente del protocolo de comunicación MQTT configurado en modo “Suscriptor”. Este módulo mantuvo una comunicación constante con la Jetson Nano para recibir los paquetes de datos en tiempo real. La información extraída contuvo las coordenadas espaciales de la pieza y la etiqueta de clasificación necesaria para su posterior ordenamiento.

2.4.2 Algoritmo de compensación de trayectoria (Lag compensation)

Luego de recibir las coordenadas y la etiqueta de clasificación del objeto, el sistema enfrentó un desafío de sincronización debido al movimiento continuo de la banda transportadora. Cuando el robot acudiera a la coordenada de ubicación de la pieza, esta ya se habría movido. Este desface aumentaba cuando el manipulador se encontraba ejecutando una tarea previa. Para mitigar este problema, se implementó un algoritmo de compensación cinemática capaz de calcular un *offset* o desplazamiento basado en la velocidad lineal de la banda y el tiempo estimado de arribo del robot, prediciendo el punto de intercepción. De esta manera, el Niryo One iría hacia donde se encontraría la pieza, garantizando su agarre en movimiento.

2.4.3 Ejecución de la clasificación (Pick & Place)

Finalmente, se programaron las rutinas que realizó el brazo robótico para la tarea de *pick-and-place*. Para lograr controlar el movimiento del Niryo One se emplean librerías que se encuentran incorporadas en el microcontrolador del robot. Para indicar las rutinas de movimiento que el brazo robótico debo cumplir, se creó un programa en donde se usó variables que representan los parámetros de posición, los cuales son calculados y enviados desde la Jetson Nano, y se las incluyó en las funciones dedicadas al cálculo de la cinemática inversa del robot.

Estas instrucciones definieron las trayectorias a seguir inmediatamente después de la intercepción del objeto, incorporando las acciones de sujeción, clasificación, liberación de la pieza y el retorno automático a la posición de espera, preparando al robot para la siguiente rutina.

Capítulo 3

3. Resultados y análisis

En el presente capítulo se analizan los resultados de desempeño obtenidos tras la integración final de los módulos de hardware y software del sistema de clasificación autónoma. Con el propósito de evaluar la robustez operativa de la arquitectura propuesta, se realizaron pruebas de estrés bajo condiciones de trabajo dinámicas, simulando un entorno de producción real mediante un flujo continuo y aleatorio de piezas sobre la banda transportadora.

La validación del sistema no se limitó únicamente a registrar el éxito o fracaso de la manipulación física final, sino que abarcó un monitoreo detallado del ciclo completo de control. Se analizó la secuencia operativa, la cual abarca desde la adquisición de las imágenes del entorno, específicamente la información RGB y los mapas de profundidad generados por la cámara, hasta la ejecución mecánica de las trayectorias por parte del brazo robótico.

Con este objetivo, se recopilaron y procesaron indicadores claves de rendimiento que permiten dictaminar la viabilidad técnica de la solución, entendida esta como la capacidad del prototipo para operar con la precisión y estabilidad requeridas en aplicaciones industriales. En este sentido, el análisis presentado a continuación se desglosa en tres puntos importantes que validan la propuesta: en primera instancia, se examina la precisión de la inferencia del modelo de inteligencia artificial en las tareas de etiquetado y segmentación de objetos para asegurar una correcta identificación del entorno; en segunda instancia, se cuantifica la latencia o retardo total del sistema para verificar su capacidad de respuesta en tiempo real ante objetos en movimiento; y finalmente, se evalúa la efectividad del agarre (Pick and Place) bajo condiciones de estrés operativo.

3.1 Análisis del desempeño del modelo de visión artificial

Una vez completado el entrenamiento del modelo neuronal YOLOv5-seg, se procedió a evaluar su capacidad de generalización frente a datos no vistos. Esta etapa es de suma importancia para garantizar que el sistema no solo haya memorizado el conjunto de entrenamiento, sino que sea capaz de identificar correctamente las piezas bajo las variaciones naturales del proceso de manufactura.

Para el análisis cuantitativo de desempeño se utilizaron las métricas obtenidas sobre el conjunto de validación, constituido por 270 imágenes. En este subconjunto, el algoritmo procesó un total de 624 instancias (piezas) distribuidas entre las cuatro clases objetivo (codo, tee, neplo, unión).

3.1.1 Métricas de detección y segmentación

Para efectos de interpretar correctamente la fiabilidad del modelo propuesto, es necesario establecer los criterios técnicos bajo los cuales se juzga la calidad de cada detección. En este estudio, la evaluación se rigió por tres indicadores estándar en la industria, los cuales se definen a continuación:

1. **Intersección sobre Unión (IoU):** Es el parámetro que mide la exactitud geométrica. Matemáticamente, se calcula dividiendo el área de solapamiento entre la predicción y la etiqueta real (Ground Truth) por el área total de su unión.

   Para este análisis se estableció un umbral de IoU ≥ 0.5. Esto significa que cualquier detección que cubra al menos el 50% de la superficie real del objeto se considera un acierto válido, criterio ampliamente aceptado para aplicaciones en tiempo real.
2. **Precisión**: Representa la fiabilidad de las predicciones positivas. Indica la proporción de instancias etiquetadas por el modelo que corresponden efectivamente a la clase predicha. Un valor alto de precisión implica que el sistema minimiza los errores de comisión (falsos positivos).
3. **Sensibilidad (*Recall*):** Evalúa la capacidad del modelo para no perder ninguna pieza. Un recall alto indica que el sistema tiene una tasa mínima de falsos negativos (objetos no detectados).
4. **Precisión media (mAP@0.5):** Representa el promedio de la precisión calculada para todas las clases evaluadas bajo el umbral de IoU establecida, integrando en una sola métrica el equilibrio entre la exactitud de la clasificación y la capacidad de detección.

Tabla 6

Resumen de métricas de desempeño del modelo YOLOv5-seg.

Clase	Precisión	Recall	mAP@0.5
Codo	0.988	0.955	0.989
Neplo	0.996	0.989	0.995
Tee	0.893	0.949	0.976
Unión	0.901	0.974	0.975
Promedio	0.945	0.967	0.984

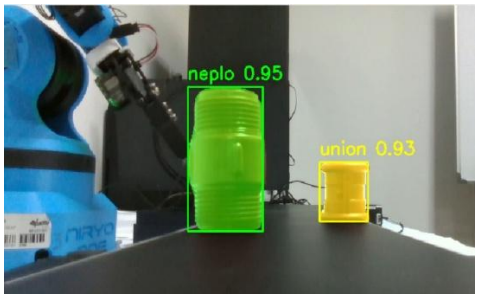
Bajo los parámetros definidos, la Tabla 6 presenta el desempeño alcanzado por el modelo sobre el conjunto de validación. Se observa que el sistema alcanzó una precisión media (mAP@0.5) de 0.984. Este valor evidencia la robustez de la arquitectura propuesta, demostrando que la red neuronal es capaz de generalizar correctamente las características visuales de los accesorios de tubería sin incurrir en sobreajuste (*overfitting*).

Por otro lado, el promedio de la sensibilidad (*recall*) se situó en 0.967. En el contexto de una aplicación industrial de clasificación, esta métrica posee una importancia operativa crítica: un *recall* elevado garantiza que el sistema minimiza los errores de omisión, es decir, asegura que la inmensa mayoría de las piezas que transitan por la banda sean detectadas e identificadas. Esto es preferible a priorizar una precisión pura, ya que dejar pasar una pieza sin clasificar representa un fallo de proceso más costoso que un falso positivo ocasional.

Por su parte, se observa que el promedio de la precisión alcanzó un valor de 0.945, lo cual indica que, de las detecciones realizadas, la tasa de falsos positivos es marginal, confirmando la fiabilidad de las predicciones. Dicha exactitud se demuestra en la figura 19, siendo esta una captura del proceso de detección ocurriendo en tiempo real.

Figura 19

Clasificación de las piezas en tiempo real sobre la banda.



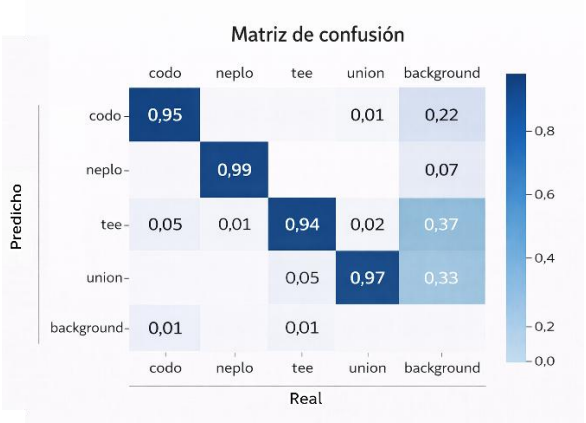
Al desglosar el comportamiento por categorías, la clase “neplo” exhibió el mejor desempeño del conjunto, alcanzando una precisión casi perfecta de 0.996. Este resultado se atribuye a la morfología cilíndrica y alargada de la pieza, la cual ofrece rasgos visuales distintivos que facilitan su discriminación respecto al fondo. Por otro lado, las clases “tee” y “unión” presentaron valores de precisión ligeramente inferiores (0.893 y 0.901 respectivamente), aunque mantuvieron un mAP superior al 97%. Esta leve variación sugiere una ambigüedad menor en la clasificación debida a la similitud de sus proyecciones 2D en ciertas perspectivas, sin que ello comprometa la efectividad global del sistema de detección.

3.1.2 Análisis de la matriz de confusión

Si bien las métricas demuestran un alto desempeño promedio, es imperativo analizar la distribución de los errores de clasificación entre las distintas categorías para identificar posibles debilidades estructurales del modelo. Para este fin, se generó la matriz de confusión normalizada sobre el conjunto de validación, la cual se presenta en la figura 20.

Figura 20

Matriz de confusión normalizada.



La diagonal principal de la matriz exhibe una concentración dominante de predicciones correctas, con tasas de acierto que superan el 94% en todas las categorías evaluadas. Este patrón confirma la estabilidad estructural del clasificador y su capacidad para discriminar eficazmente la gran mayoría de las instancias.

Al profundizar en el desempeño por clase, se destaca la categoría “neplo” como la más robusta del conjunto, alcanzando una precisión de 0.99. Por otro lado, los elementos fuera de la diagonal evidencia un ligero fenómeno de confusión cruzada entre las clases “tee” y “unión”. Este comportamiento varía según la perspectiva de la cámara, ya que, en ciertas rotaciones, la derivación lateral de la “tee” queda oculta bajo el cuerpo de la pieza, generando una silueta idéntica a la de la “unión”. A pesar de esta dificultad óptica, el modelo logró resolver más del 94% de estos casos, demostrando su capacidad para inferir la geometría correcta aún en posiciones desfavorables.

3.2 Generación de coordenadas espaciales y objetivos de agarre

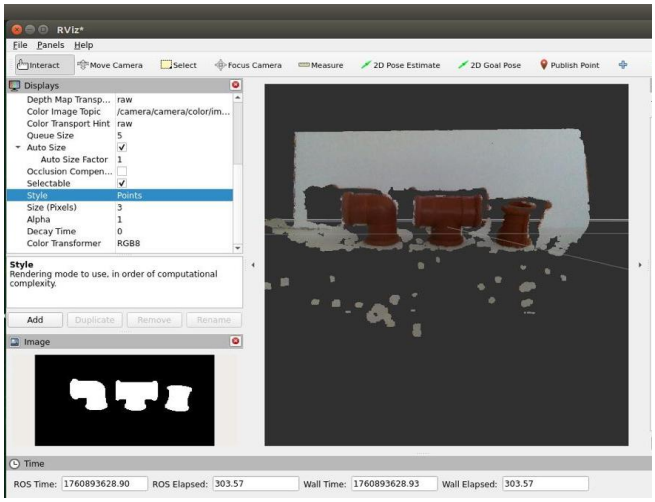
Una vez que el modelo de segmentación ha identificado la clase y la máscara del objeto en el plano bidimensional, el siguiente paso fue proyectar dicha información al espacio 3D para guiar al efector final. En esta etapa se validó la capacidad del sistema para fusionar la información de profundidad con las máscaras de segmentación, generando una nube de puntos filtrada que representa únicamente la geometría de la pieza de interés.

3.2.1 Reconstrucción de la nube de puntos

Utilizando la herramienta de visualización RViz, se verificó la correspondencia espacial entre la imagen RGB y los datos del sensor de profundidad.

Figura 21

Reconstrucción 3D y filtrado de nube de puntos en RViz.



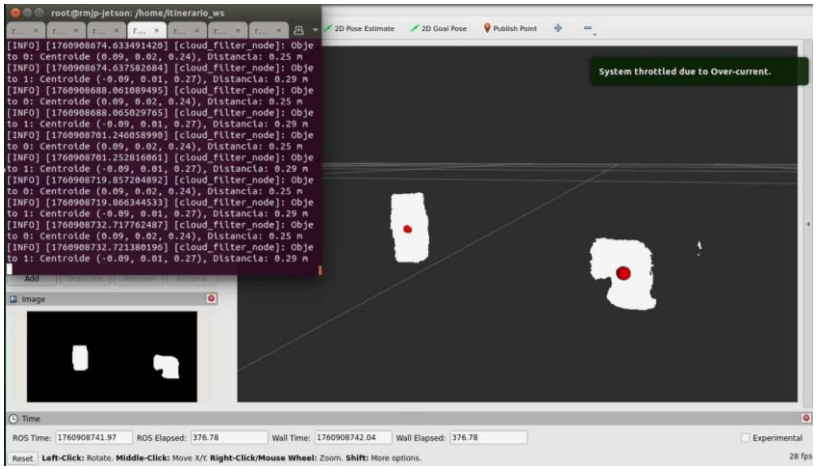
Como se observa en la figura 21, al aplicar la máscara binaria obtenida de la segmentación sobre el mapa de profundidad, el sistema selecciona exclusivamente las coordenadas espaciales que coinciden con la región del objeto, descartando los datos del entorno y de la banda transportadora. Es importante resaltar que este filtrado aísla la geometría de la pieza en el espacio 3D, lo cual permite realizar el cálculo del centroide sobre un conjunto de puntos acotado y específico.

3.2.2 Cálculo de centroides y publicación de coordenadas

A partir de la nube de puntos filtrada, el nodo de procesamiento calcula el centroide geométrico (x, y, z) de cada instancia detectada, obteniendo el vector de posición relativo al sistema de coordenadas de la cámara.

Figura 22

Estimación de centroides y publicación de coordenadas.



La figura 22 presenta la visualización de los datos procesados en tiempo real. Los marcadores rojos superpuestos a las máscaras binarias indican la ubicación espacial calculada para el efector final, mientras que la salida de la terminal expone los valores numéricos de las coordenadas y la distancia euclidiana al objeto. La baja variabilidad observada en estos valores entre fotogramas consecutivos permite destacar la estabilidad del cálculo de localización, proporcionando los vectores de entrada requeridos por el planificador de trayectorias para la resolución del problema de cinemática inversa.

3.3 Latencia y tiempo de respuesta

En una arquitectura de clasificación dinámica, donde los objetos se desplazan continuamente sobre una banda transportadora, el tiempo de procesamiento se convierte en una variable crítica ya que, si la latencia total del sistema supera el tiempo físico que tarda la pieza en atravesar el área de trabajo del robot, la manipulación será inviable.

Para cuantificar esta capacidad de respuesta, se midió el tiempo de inferencia promedio del modelo YOLOv5-seg ejecutándose en la tarjeta embebida NVIDIA Jetson Nano. Las pruebas se realizaron procesando el flujo de video en tiempo real a una resolución de 640x640 píxeles.

Tabla 7

Latencia y frecuencia de procesamiento del sistema

Métrica	Valor
Frecuencia promedio (Hz)	5.00
Tiempo de ciclo promedio (s)	0.200
Desviación estándar (s)	0.154
Latencia Máxima (s)	1.513

La tabla 7 muestra los resultados promedios de procesamiento, obtenidos tras la estabilización del sistema (ventana de muestreo mayor a 100 cuadros). Se puede notar que el sistema mantiene una frecuencia de actualización promedio de 5 Hz, lo cual equivale a un tiempo de ciclo de 0.200 segundos por fotograma. Este rendimiento confirma que la plataforma embebida opera cerca de su límite computacional al ejecutar la segmentación de instancias. Es importante señalar que la desviación estándar es de 0.154 s, con picos ocasionales de latencia que superan el segundo. Esta variabilidad es característico de sistemas operativos no deterministas (como Linux en la Jetson) cuando gestionan cargas intensivas de visión artificial.

A pesar de estas fluctuaciones, el rendimiento promedio de 5 cuadros por segundo resulta funcional para la velocidad establecida de la banda transportadora. Dado que el flujo de piezas es constante y la velocidad lineal es moderada, el sistema de control puede tolerar estos tiempos muertos (200 ms) utilizando el último vector de posición conocido.

3.4 Validación operativa en entorno dinámico

La evaluación culminante del sistema consistió en verificar la integración física de todos los subsistemas mediante una prueba de estrés en condiciones dinámicas. El objetivo de este experimento fue determinar la eficacia real del prototipo para completar el ciclo de manipulación bajo una carga de trabajo continua, validando la sincronización entre la detección visual, la predicción de trayectoria y la actuación mecánica.

3.4.1 Protocolo experimental y resultados

Para esta prueba, se sometió al robot a un flujo aleatorio de 24 piezas distribuidas equitativamente entre las cuatro clases, es decir, 6 para cada tipo. Los objetos fueron colocados sobre la banda en movimiento sin pausas intermedias, obligando al algoritmo de planificación a optimizar los tiempos de retorno a la posición de espera (*Home*) para no perder el siguiente objetivo. Se definió como "éxito" únicamente aquella rutina donde el robot logró interceptar la pieza, sujetarla firmemente, levantarla sin colisiones y depositarla en la zona de clasificación designada.

Tabla 8

Resultados de prueba de estrés

Tipo	Piezas recolectadas	Efectividad
Codo	6/6	100%
Tee	6/6	100%
Neplo	5/6	83%
Unión	4/6	67%
Total	21/24	87.5%

3.4.2 Análisis de eficacia por geometría

En la tabla 8 se muestra que el sistema alcanzó una tasa de éxito global del 87.5%, logrando recolectar y clasificar correctamente 21 de las 24 piezas enviadas. Al revisar este comportamiento por categoría, sobresale el desempeño perfecto en la manipulación de "codos" y "tees". Este resultado resulta importante, dado que estas piezas presentan las geometrías más completas y asimétricas del conjunto. El éxito en estas categorías valida dos decisiones de diseño: la eficacia del cálculo de la orientación (*yaw*) para alinear la muñeca del robot con el eje principal del objeto, y la idoneidad del *gripper* adaptativo, cuyos dedos flexibles demostraron la capacidad de amoldarse a las curvaturas irregulares garantizando una sujeción estable incluso durante la aceleración del brazo.

En contraposición, los fallos registrados (3 de 24) se concentraron en las piezas de menor dimensión. En el caso de las uniones, se obtuvo una efectividad del 67% la cual puede asociarse a limitaciones mecánicas del agarre. Al tratarse de piezas de bajo perfil y superficie lisa, la tolerancia de cierre del *gripper* resultó crítica; en dos ocasiones, la fuerza de cierre provocó el deslizamiento del objeto antes de completar el levantamiento.

De manera similar, el único fallo en la clase “neplo” se debió a una colisión menor del efector final con la parte superior de la pieza, lo que sugiere la necesidad de una calibración milimétrica adicional en la coordenada z para compensar dicha diferencia de altura entre esta pieza y las demás.

Capítulo 4

4. Conclusiones y recomendaciones

En este capítulo se sintetizan los principales hallazgos obtenidos a partir del diseño e implementación de un sistema de clasificación autónomo empleando visión por computadora. Se evalúa el grado de cumplimiento de los objetivos, analizando el desempeño de la integración entre la visión artificial y el control robótico. Asimismo, se presentan recomendaciones orientadas a la optimización, escalabilidad y futura implementación industrial del prototipo, con el fin de maximizar su impacto en entornos productivos reales.

4.1 Conclusiones

- El sistema de visión artificial mostró una alta capacidad de generalización en la segmentación de piezas, logrando diferenciar correctamente objetos de interés con otras geometrías en general, inclusive, logró diferenciar entre objetivos de similares características, pero de categorías distintas, lo cual supera las limitaciones de los sistemas de visión clásicos implementados en el ámbito industrial, los cuales están basados únicamente en patrones de color o formas geométricas rígidas.
- La integración de la cámara de profundidad (RGB-D) permitió que el sistema no solo lograra identificar el objeto de interés, también pudo predecir su posición espacial con gran precisión. Esto concluye que el sistema es capaz de manipular objetos con variaciones de posición y orientación en la banda transportadora sin necesidad de reprogramar el robot o el sistema como tal.
- La arquitectura del sistema de control implementada demostró estabilidad operativa ante variaciones no controladas, como cambios leves en la velocidad de la banda o la orientación aleatoria de las piezas. Este sistema garantiza la operación segura del robot con mínima intervención humana.
- Se logró una sincronización exitosa entre el espacio visual y el espacio de trabajo del robot mediante calibraciones manuales y automáticas de forma precisa. Esto permitió que el

sistema compensara dinámicamente la posición del objeto de interés en movimiento, ajustando la trayectoria del efector final en tiempo real para interceptar la pieza en el momento exacto.

4.2 Recomendaciones

Basado en los hallazgos y las limitaciones identificadas durante la fase de pruebas del prototipo, se presentan una serie de recomendaciones orientadas a la mejora continua del sistema. Estas propuestas tienen como objetivo potenciar la escalabilidad, la robustez, eficiencia y velocidad del sistema ante perturbaciones del entorno y optimizar el rendimiento del procesamiento para facilitar la futura transición de esta solución desde un entorno de laboratorio hacia una implementación en el ámbito industrial.

- Implementar un brazo robótico que sea capaz de operar a mayores velocidades o, inclusive, uno de grado industrial. Esto permitirá probar la arquitectura de programación a condiciones de estrés más rigurosas, aprovechando así la velocidad de procesamiento y comunicación a la que es capaz de llegar el sistema.
- Se recomienda implementar un sistema de calibración dinámica de la cámara mediante el uso de marcadores ubicados en el espacio de trabajo. Esto permitiría al sistema recalcular la matriz de transformación entre la cámara y el robot en tiempo real, de esta forma se garantiza la precisión del agarre incluso si la cámara sufre desplazamientos accidentales o vibraciones durante la operación.
- Como trabajo a futuro, se recomienda incorporar un sistema de estimación de velocidad en tiempo real para los objetos sobre la banda. A diferencia del modelo actual la cual usa una velocidad fija, este módulo permitiría calcular la trayectoria específica de cada pieza, logrando que, a pesar de que existan alguna clase de deslizamiento o empuje, el sistema pueda predecir correctamente la posición en la que se interceptará la pieza.

- Incorporar un manipulador adicional al sistema para trabajar de manera cooperativa entre los robots. Esta implementación serviría para corroborar la escalabilidad operativa del diseño, con la que se puede diseñar y evaluar distintas estrategias de distribución de carga, aumentando así el rendimiento del sistema.

Para concluir, el presente proyecto valida la factibilidad de la implementación de tecnologías utilizadas habitualmente al entorno investigativo, tal como la segmentación de imágenes y la percepción de profundidad, en aplicaciones industriales. La materialización de este prototipo sienta las bases para la modernización de los procesos de producción, demostrando que la incorporación de tecnologías de vanguardia es un paso viable para la mejora continua del sector industrial.

Referencias

[1] A. Pérez, “Cómo integrar la robótica colaborativa en la industria 4.0,”OBS Business School, 5 Abril 2024. [En línea]. Available: https://www.obsbusiness.school/blog/como-integrar-la-robotica-colaborativa-en-la-industria-40. [Último acceso: 28 Octubre 2025].

[2] S. Zoting, “Collaborative Robots Market Size and Forecast 2025 to 2034,”Precedence Research, 26 Mayo 2025. [En línea]. Available: https://www.precedenceresearch.com/collaborative-robots-market. [Último acceso: 8 Noviembre 2025].

[3] “Global Robot Demand in Factories Doubles Over 10 Years,”International Federation of Robotics (IFR), 25 Septiembre 2025. [En línea]. Available: https://ifr.org/ifr-press-releases/news/global-robot-demand-in-factories-doubles-over-10-years. [Último acceso: 8 Noviembre 2025].

[4] M. Naeem, S. Aslam, M. Suhaib, S. Gul, Z. Murtaza y M. J. Khan, “Design and Implementation of Pick and Place Manipulation System for Industrial Automation,”*2021 International Conference on Artificial Intelligence and Mechatronics Systems (AIMS),* pp. 1-6, 2021.

[5] A. Vina, “Ultralytics,”30 Agosto 2024. [En línea]. Available: https://www.ultralytics.com/es/blog/understanding-the-integration-of-computer-vision-in-robotics. [Último acceso: 4 Noviembre 2025].

[6] S. Stafford, “Teamsense,”25 Septiembre 2025. [En línea]. Available: https://www.teamsense.com/blog/cost-of-downtime-manufacturing. [Último acceso: 27 Octubre 2025].

[7] Siemens, “Siemens AG,”Siemens AG, Erlangen, 2024.

[8] H. Fan, Y. Hu y L. Tang, “Labor costs and the adoption of robots in China,”*Journal of Economic Behavior & Organization,* vol. 186, pp. 608-631, 2021.

[9] “Service Robots See Global Growth Boom,”International Federation of Robot (IFR), 07 Octubre 2025. [En línea]. Available: https://ifr.org/ifr-press-releases/news/collaborative-robots-Market-Grows-by-42-percent. [Último acceso: 4 Noviembre 2025].

[10] “Advantages and disadvantages of robot automation in 2025: A balanced guide,”Standar Bots, 18 Agosto 2025. [En línea]. Available: https://standardbots.com/blog/pros-and-cons-of-robot-automation. [Último acceso: 8 Noviembre 2025].

[11] E. Henriques, “NextGen,”10 Octubre 2024. [En línea]. Available: https://nextgen.ec/blog/automatizacion-industrial/tecnologia-automatizacion-industrial-ecuador/. [Último acceso: 27 Octubre 2025].

[12] A. Hayes, “Investopedia,”5 Octubre 2025. [En línea]. Available: https://www.investopedia.com/terms/f/flexible-manufacturing-system.asp. [Último acceso: 27 Octubre 2025].

[13] Enfoque, “Vistazo,”2 Diciembre 2024. [En línea]. Available: https://www.vistazo.com/enfoque/2024-12-02-guayaquil-nueva-linea-produccion-impulsa-economia-circular-generacion-empleo-AA8406417. [Último acceso: 28 Octubre 2025].

[14] S. Tyrpak, “Inspección Visual Manual,”Global Life Science Service Group, [En línea]. Available: https://blog.pqegroup.com/es-es/cumplimiento-gxp/inspeccion-visual-manual#:~:text=Incluso%20con%20un%20100%25%20de%20inspecciones%2C%20el,oscilan%20entre%20el%2020%25%20y%20el%2040%25.. [Último acceso: 24 enero 2026].

[15] K. Bhalavat, “Plutomen,”16 Septiembre 2024. [En línea]. Available: https://pluto-men.com/human-error-persistent-challenge-manufacturing-operations/. [Último acceso: 27 Octubre 2025].

[16] National Institute for Occupational Safety and Health (NIOSH), “Ergonomic Guidelines for Manual Material Handling,”National Institute for Occupational Safety and Health (NIOSH), Cincinnati, OH, 2007.

[17] A. Colantoni, M. Cecchini, D. Monarca y R. Bedini, “The risk of musculoskeletal disorders due to repetitive movements of upper limbs for workers employed in hazelnut sorting,”*Journal of Agricultural Engineering,* p. 1, 2013.

[18] M. P. Groover, Automation, production systems, and computer-integrated manufacturing, Pearson, 2018.

[19] H. Templemeir, Flexible manufacturing systems: Decision support for design and operation, John Wiley & Sons, 2012.

[20] K. Yoram, “The Global Manufacturing Revolution: Product-Process-Business Integration and Reconfigurable Systems,”John Wiley & Sons, Inc., 2010.

[21] N.-H. Tran, H.-S. Park, Q.-V. Nguyen y T.-D. Hoang, “Development of a Smart Cyber-Physical Manufacturing System in the Industry 4.0 Context,”2019.

[22] J. M. Gamboa y M. J. Ramirez, “Multi-Agent Design To Migrate A Flexible Manufacturing Cell To A Holonic System,”2008.

[23] L. Rojas, S. Jácome y V. Gancino, “Industria 4.0: el reto en la ruta hacia las organizaciones digitales.,”*Estudios de la Gestión: Revista Internacional de Administración,* pp. 123-149, 2020.

[24] O. Alvarez Vásquez y F. R. Arroyo Morocho, “Análisis de la Industria 4.0 como factor diferenciador del Sector Industrial del Ecuador,”*Ciencia Latina Revista Científica Multidisciplinar,* pp. 3314-3324, 2021.

[25] Y. E. Villacís Muguerza, “Análisis de la implementación de tecnologías 4.0 en los porcesos productivos de las empresas industriales de Guayaquil,”2024.

[26] J. M. Avila Lapo, “Impoacto de la industria 4.0 en las Pymes manufactureras del Azuay,”2024.

[27] R. Szeliski, Flexible manufacturing systems: Decision support for design and operation, John Wiley & Sons., 2022.

[28] L. H. Phuong, P. X. Trung,, T. T. Quyen y T. T. T. Mai, “Development of an Intelligence Vision for a Robot System to Pick and Place Objects,”*FME Transactions,* pp. 223-231, 2024.

[29] “¿Qué es la Robótica Colaborativa?,”EDS Robotics, 19 Enero 2022. [En línea]. Available: https://www.edsrobotics.com/blog/robotica-colaborativa-que-es/. [Último acceso: 28 Octubre 2025].

[30] “CADE Cobots,”22 Octubre 2021. [En línea]. Available: https://cadecobots.com/robotica-colaborativa-que-es-ventajas-y-aplicacion/. [Último acceso: 28 octubre 2025].

[31] A. Dahiya, A. Arroyo, K. Dautenhahn y S. Smith, “A Survey of Multi-Agent Human-Robot Interaction Systems,”arXvi, 2002.

[32] “nvidia,”2025. [En línea]. Available: https://www.nvidia.com/es-la/autonomous-machines/embedded-systems/jetson-nano/product-development/. [Último acceso: 14 diciembre 2025].

[33] “Amazon,”2025. [En línea]. Available: https://www.amazon.com/-/es/Intel-RealSense-Cámara-profundidad-82635D435IDK5P/dp/B07MWR2YJB. [Último acceso: 14 diciembre 2025].

[34] “ROHTEK AUTOMATION,”2025. [En línea]. Available: https://rohtekautomation.com/product/niryo-ned-6-axis-robotic-arms/. [Último acceso: 14 diciembre 2025].

[35] “Ubuy,”2025. [En línea]. Available: https://www.ubuy.ec/es/product/OOICZG-dobot-conveyor-belt-kits-the-simplest-production-line?srsltid=AfmBOopOMNgAAWBltuTnOD6HCUDDceD0Vj0WVf4FibdCT-rCiVAwe1UV. [Último acceso: 14 diciembre 2025].

Apéndices

Apéndice A: Especificaciones técnicas del hardware

En esta sección se detallan las características técnicas de los componentes críticos seleccionados para la arquitectura del sistema los cuales son: el microprocesador NVIDIA Jetson Nano, el robot colaborativo Niryo One, la cámara Intel RealSense D435i y la banda transportadora. La información presentada valida la capacidad de cómputo, detección y actuación requerida para el procesamiento de redes neuronales y control de movimiento en tiempo real.

Tabla A.1.

Especificaciones técnicas de la unidad de procesamiento NVIDIA Jetson Nano

Característica	Especificación
GPU	128-core NVIDIA Maxwell
CPU	Quad-core ARM Cortex-A57 MPCore
Memoria	4 GB 64-bit LPDDR4
Conectividad	Gigabit Ethernet, M.2 Key E (Wi-Fi)
Soporte de software	NVIDIA JetPack SDK (Linux, CUDA)

Tabla A.2.

Especificaciones técnicas de la cámara Intel RealSense D435i

Característica	Especificación
Tecnología	Estereoscopía active (Active IR Stereo)
Resolución profundidad	Hasta 1280 x 720 a 90 fps
Resolución RGB	Hasta 1920 x 1080 a 30 fps
Campo de visión	Aprox. 86° x 57°
Rango operativo	0.3 m a 3 m
Interfaz	USB 3.0 Tipo-C

Tabla A.3.

Especificaciones técnicas del co-bot Niryo One

Característica	Especificación
Grados de libertad	6
Carga útil	300 g
Alcance máximo	440 mm
Protocolos	Wi-Fi, Ethernet, MQTT, Modbus TCP

Tabla A.4.

Especificaciones técnicas de la banda transportadora del Dobot Magician Kit

Característica	Especificación
Rango de velocidad	0 a 120 mm/s (Ajustable)
Tipo de actuador	Motor paso a paso
Control de velocidad	Señal PWM/ Control de frecuencia de pulsos
Carga máxima	600 g

Apéndice B: Evidencia de entrenamiento y resultados de visión

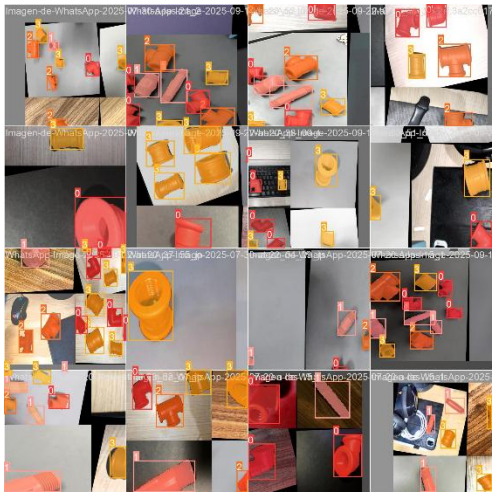
Este anexo documenta el proceso de aprendizaje supervisado del modelo YOLOv5-seg. Se presentan las figuras B.1. y B.2. donde se observan el conjunto de datos y los resultados de la inferencia en el set de validación respectivamente.

Figura B.1.

Mosaico del dataset generado en Roboflow.

Figura B.2.

Resultados de inferencia y segmentación de instancias.



Apéndice C: Sistemas de referencias del robot y la cámara.

Se presenta la disposición de los ejes de coordenadas del robot Niryo One y de la cámara Intel RealSense en las figuras C.1. y C.2. Esta documentación es necesaria para la calibración de ambos sistemas de referencia, con el fin de ejecutar la planificación de trayectorias.

Figura C.1.

Sistema de coordenadas base (X, Y, Z) del Robot Niryo One.

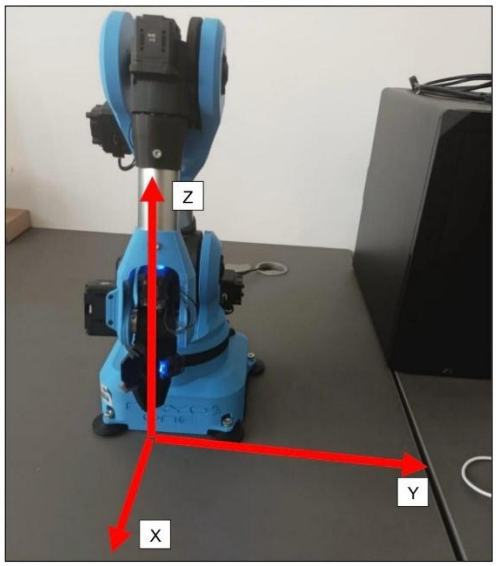
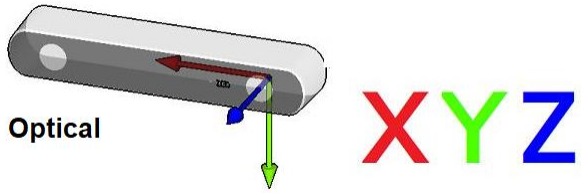


Figura C.2.

Sistema de coordenadas base (X, Y, Z) de la cámara de profundidad.



Apéndice D: Arquitectura de nodos y flujo de datos en ROS2

Esta sección resume la estructura lógica del software implementado sobre el middleware ROS2. El diagrama de computación mostrado en la figura D.1. ilustra la interconexión entre los distintos nodos (procesos independientes) y los tópicos (canales de comunicación) que transportan la información desde la adquisición sensorial hasta la orden de la actuación.

Figura D.1.

Grafo de computación (rqt_graph) del sistema.