



UNIVERSIDAD NACIONAL  
AUTÓNOMA DE MÉXICO

TESIS

---

**Percepción artificial y planeación de  
acciones a partir de la cámara RGBD de  
un robot móvil**

---

Que para optar por el grado de:  
Maestra en Ingeniería Eléctrica

*Presenta:*  
Mitzi Anaid Ramírez Estrada

*Directores de Tesis:*  
Dr. Jesús Savage Carmona  
Dr. Marco Antonio Negrete Villanueva

Ciudad de México, 2022-2023

---

---

# Índice general

<b>Resumen</b>	<b>VII</b>
<b>Abstract</b>	<b>IX</b>
<b>Agradecimientos</b>	<b>XI</b>
<b>1. Introducción</b>	<b>3</b>
1.1. Motivación	4
1.2. Planteamiento del problema	5
1.3. Hipótesis	6
1.4. Objetivos	6
1.4.1. General	6
1.4.2. Específicos	6
1.5. Descripción del documento	6
<b>2. Antecedentes</b>	<b>9</b>
2.1. Robots de Servicio en entornos Industriales	9
2.2. Visión Computacional	10
2.2.1. Espacios de color	11
2.2.2. Segmentación de Imágenes	14
2.2.3. Operaciones Morfológicas	15
2.3. Reconocimiento de Objetos	21
2.4. Máquinas de Estado	23
2.5. Sistemas expertos	24
2.6. Manipulación del Robot	25
2.7. Estado del Arte	25
2.7.1. Antecedentes en RoboCup Logistics	26
<b>3. Detección de objetos</b>	<b>29</b>
3.1. Objetos de interés	29
3.2. Caracterización de los objetos	30
3.3. Localización de objetos	33
3.3.1. Nubes de puntos	33

---

<b>4. Planeación de movimientos</b>	<b>35</b>
4.1. Movimiento de cuerpo rígido	36
4.1.1. Posición y Orientación	36
4.2. Transformaciones homogéneas	39
4.3. Descripción cinemática del robot	40
4.4. Manipulación de objetos	41
<b>5. Planeación de acciones</b>	<b>43</b>
5.1. Actividades	43
5.1.1. Navegación	44
5.1.2. Exploración	44
5.1.3. Manipulación	44
5.2. Máquinas de estados	44
5.2.1. Estados	44
5.2.2. Navegación	45
5.2.3. Exploración	45
5.2.4. Manipulación	47
5.2.5. Entradas y salidas	47
5.2.6. Algoritmos de las Máquinas de Estados - Cartas ASM	48
<b>6. Resultados</b>	<b>53</b>
6.1. Herramientas	53
6.1.1. Herramientas de software	53
6.1.2. Herramientas de hardware	57
6.2. Implementación	62
6.2.1. Resultados - Visión	64
6.2.2. Resultados - Manipulación	65
6.2.3. Resultados - Integración	69
<b>7. Conclusiones y Trabajo Futuro</b>	<b>71</b>
7.1. Conclusiones	71
7.1.1. Visión Computacional	71
7.1.2. Manipulación	72
7.2. Trabajo Futuro	72
<b>Referencias</b>	<b>73</b>

---

# Índice de figuras

2.1. Espacio RGB . . . . .	12
2.2. Espacio HSV, representación cónica . . . . .	14
2.3. Unión e Intersección de Conjuntos . . . . .	16
2.4. Complemento . . . . .	17
2.5. Diferencia . . . . .	17
2.6. Reflexión . . . . .	17
2.7. Traslación . . . . .	18
2.8. Operaciones lógicas AND, OR y NOT . . . . .	18
2.9. Operaciones lógicas aplicadas a imagen . . . . .	19
2.10. Operaciones Morfológicas: dilatación y erosión . . . . .	20
2.11. Dilatación aplicada con distinto elemento estructurante . . . . .	21
2.12. Reconocimiento de objetos . . . . .	22
2.13. Detección de Histogramas de Gradientes Orientados (HOG) 2021 . . . . .	26
2.14. Robotino modificado - GRIPS Furbaß y col., 2021 . . . . .	27
3.1. Objetos de interés . . . . .	30
3.2. Ejemplo del espacio de trabajo del robot . . . . .	30
3.3. Efectos de los cambios de iluminación en las piezas . . . . .	31
3.4. Ejemplo de anillo amarillo en: izq) HSV . . . . .	31
3.5. Medias de color a diferentes iluminaciones y en diferentes regiones - base roja . . . . .	32
3.6. Pieza roja - componentes a) H b) S c) V . . . . .	33
3.7. Pieza roja . . . . .	33
3.8. Espacio de trabajo . . . . .	34
4.1. Posición y orientación de un cuerpo rígido . . . . .	37
4.2. Árbol cinemático del robot . . . . .	41
4.3. Descripción cinemática . . . . .	41
5.1. Exploración . . . . .	46
5.2. Diagrama de carta ASM -Navegación . . . . .	49
5.3. Diagrama de carta ASM - Exploración . . . . .	50
5.4. Diagrama de carta ASM - Manipulación . . . . .	51

---

6.1. Robotino 3 Didactic, 2013	Robotino 3 modificado	59
6.2. Kinect versión 1 - Componentes . . . . .		60
6.3. Proyector Infrarrojo y Sensores CMOS 2010 . . . . .		60
6.4. PhantomX Pincher AX-12 (2022) . . . . .		63
6.5. Medias de color . . . . .		64
6.6. OpenCV - espacio HSV - mapa de color (Pai, 2022) . . . . .		65
6.7. Piezas segmentadas . . . . .		66
6.8. Izq) Posición de reposo - Der) posición final . . . . .		67
6.9. Secuencia de graspeo . . . . .		68
6.10. Posición <i>Kinect_hide</i> . . . . .		69

# Resumen

Los robots de servicio integrados en entornos industriales surgen de la necesidad de diversificar las actividades que es posible realizar en una planta de producción y se considera benéfica la integración de visión y la percepción de profundidad en su desarrollo. Una más de las ventajas atribuidas a este tipo de robots es la reducción de riesgos al realizar labores peligrosas o que involucren el contacto con maquinaria que pudiera resultar riesgosa para operadores humanos. Para esto es necesario que el robot pueda operar de la forma más autónoma posible, navegar en el espacio en que se encuentra e interactuar con sus alrededores de forma segura.

En el presente documento se describe el proyecto realizado para integrar un sistema de visión computacional en el comportamiento del un robot de servicio industrial. Para lo anterior, se integró una cámara RGBD en la estructura de un robot de servicio industrial y utilizando técnicas de visión computacional se realizó el procesamiento de los datos obtenidos. Integrando esta información en una secuencia de acciones que permiten al robot manipular un objeto determinado dentro de su espacio de trabajo y transportarlo a otra locación. Los resultados se integraron dentro de los lineamientos establecidos por la competencia *RoboCup Logistics*, donde se busca simular un sistema de producción en serie. Dentro de este entorno es posible hacer pruebas para el desarrollo de robots de servicio para entornos industriales.

---

---



# Abstract

The integration of service robots in industrial environments comes from the urge to diversify the activities that can be made in a production plant. As ground for this project it is considered beneficial to integrate computer vision and depth perception in its behaviour. One of the advantages of these robots is the reduction the risks, by performing dangerous activities or those that involve contact with machines that represent a risk to human operators. To complain with this objective it is necessary that te robot can operate in the most autonomous way possible, navigating through the environment and interact with its surroundings in a safe way.

This document describe a project made to integrate a computer vision system in an industrial service robot's behaviour. In order to achieve it, a RGBD camera was added to the hardware of a mobile robot, this information was processed later using computer vision techniques. The processed information was used during a sequence of actions, organized using Finite State Machines, that dictate the robot's behaviour, with which it is able to manipulate an object in its workspace and transport it to another location. The results were evaluated following the guidelines by the *RoboCup Logistics* competition, where a simulation of a production system takes place. In this setting it is possible to make tests to develop industrial service robots.

---

---

# Agradecimientos

*A mis padres Nohemí y Margarito, que a lo largo de mi vida me han cuidado y acompañado con su guía y su paciencia. Quienes me inculcaron los valores con los que me conduzco día con día. Quienes me han apoyado para cumplir cada una de mis metas. A ustedes, que me demuestran diariamente el incondicional e infinito amor que sienten por mí. Sin ustedes no sería quien soy ni estaría donde estoy, he estado o estaré. No hay palabras suficientes para expresar el amor y agradecimiento que siento.*

*A mis hermanos Mauricio y Uriel, que con su apoyo, consejos, bromas, su compañía y opiniones han acompañado, iluminado y enriquecido mi camino. Por darme seguridad y estar siempre a mi lado. A ustedes, que me cuidan y apapachan todos los días.*

*A Rebeca, Paola, la Sra. Ethel, la Sra. Lucha y el Sr. Ricardo, por hacerme sentir parte de su familia, por recibirme en su hogar, escucharme, motivarme, aconsejarme y regañarme cuando hacía falta.*

*A Lore, Steph y Hugo, por acompañarme desde antes de comenzar el posgrado, por guiarme, aconsejarme y entenderme. Gracias por las risas, las pláticas, las comidas, los cafés, por hacer más llevadero este proceso y por estar siempre dispuestos a ayudar.*

*A Rebeca, Daniel y Joshua, por que gracias a su compañía el laboratorio y las competencias han estado llenos de recuerdos felices, de éxitos, de bromas y risas, de apoyo y trabajo en equipo. Porque cuando me di cuenta ya los consideraba mis amigos y con su presencia hicieron más bonitos los días de trabajo.*

*A David, Fer, Jonathan, Nicole y Aldo, por escucharme y apoyarme desde antes y a lo largo de este proceso, por motivarme, por acompañarme en mis altas y bajas, en mis momentos de duda y por confiar en mí más que yo misma.*

---

*A la Universidad Nacional Autónoma de México, al Posgrado de Ingeniería, al departamento de Procesamiento de Señales, a la Facultad de Ingeniería y al Colegio de Ciencias y Humanidades, porque desde que recibí mi carta de asignación se convirtieron en un segundo hogar durante cada etapa académica y me brindaron las herramientas necesarias para mi formación profesional.*

*Al Doctor Jesús Savage Carmona y al Doctor Marco Antonio Negrete Villanueva, por el inmenso apoyo que me han brindado desde que comencé a involucrarme en el laboratorio y por guiarme en cada paso de este proyecto. Por compartir su conocimiento y experiencia, y por la paciencia que han tenido conmigo.*

*A los miembros del jurado: Dr. Pablo Roberto Pérez Alcázar, M. I. Larry Escobar Salguero y el Dr. Miguel Ángel Padilla Castañeda, por los conocimientos que compartieron durante la maestría y por sus comentarios y consejos.*

*A los demás integrantes del laboratorio de Biorrobótica, por el trabajo conjunto que se ha hecho. Porque toda experiencia conlleva un aprendizaje.*

*Este proyecto se realizó con apoyo de los proyectos:*

- *PAPIIT TA102424, 'Modelos lógico probabilísticos para el desarrollo de robots móviles autónomos'.*
- *PAPIIT IT101524, 'Desarrollo de un sistema de planeación de tareas para la solución de problemas del área de la robótica de servicio'.*
- *PAPIME PE108624, 'Creación de material didáctico para el curso de construcción de robots móviles'.*

*Finalmente, agradezco al CONACYT, por los recursos económicos otorgados para realizar este posgrado.*

---

---

# Capítulo 1

## Introducción

Actualmente uno de los temas que se encuentran en apogeo y con mucho reconocimiento de la aplicación de los avances tecnológicos es la robótica. Constantemente se desarrollan nuevas tecnologías que es posible implementar tanto en el software como en el hardware de estas herramientas, además de la paulatina integración de estos elementos en la vida diaria, en entornos y aplicaciones diversas.

No resulta extraño mencionar que el ambiente donde con más frecuencia se encuentran estos elementos es el industrial, donde desde hace varias décadas es posible encontrar centros de producción masiva en que brazos robóticos y bandas transportadoras se encargan del ensamblaje en serie de diferentes productos. Utilizando dicho modelo, donde se encuentran comunicados y *encadenados* todas las etapas de la producción de los productos, se obtienen fábricas en las que es posible producir una gran cantidad del mismo tipo de producto.

El anterior es uno de los motivos por los cuales se ha encontrado necesaria la introducción de modelos de producción más flexibles, quizás para plantas menos especializadas o para productos que no requieran ser construidos de forma masiva, haciendo posible utilizar las alas industriales de formas variadas para el ensamblaje de productos variados. Así, considerando que las máquinas que se encargan de las etapas de producción no se encuentren físicamente ligadas unas con otras, aunque sí se encarguen de desempeñar tareas específicas dentro de un proceso secuencial, se obtiene el concepto de fábricas inteligentes, correspondientes a la llamada Industria 4.0, a veces mencionada como la 4ta revolución industrial (Kohout y col., 2020).

Navarro Piña menciona que el Instituto Norteamericano de Robótica describe el **robot industrial** como: "*un manipulador multifuncional y programable, diseñado para mover materiales, piezas, herramientas o dispositivos especiales mediante movimientos programados y variables que permiten llevar a cabo diversas tareas* (2021)", siendo estos los antes mencionados brazos armadores que típicamente se ven en las plantas industriales.

Por otro lado, Basco y col., 2018, enfatizan la importancia de la robótica colaborativa, en la cual la flexibilidad y versatilidad de las plantas es imperativa,

---

integrando sistemas que puedan sumarse al sistema de producción mediante el transporte de productos intermedios y finales entre estaciones de trabajo, que sean capaces de coexistir y colaborar con equipos similares a ellos o bien con trabajadores humanos. Siendo este último el tipo de robots con el que se plantea el desarrollo del presente proyecto.

## 1.1. Motivación

El acelerado avance de la tecnología y las necesidades de la sociedad vuelven indispensable la búsqueda de herramientas que permitan mejorar las capacidades de los robots para analizar e interactuar con sus alrededores. Dotar al robot con las herramientas necesarias para adaptarse a su entorno de trabajo es una tarea crucial para muchas aplicaciones, tanto en el entorno industrial, como en el doméstico, en labores de ensamblaje o en tareas de interacción humano-robot (Roveda y col., 2022). Con dicho constante avance tecnológico ha aumentado también el aprovechamiento de los sistemas de visión con los que es posible equipar a los robots, dando lugar al incremento de la autonomía de los mismos y expandiendo la posibilidad de explorar su entorno de trabajo con nuevas y más poderosas herramientas y sensores, mejorando así su desempeño.

Muchas aplicaciones pueden beneficiarse de la visión computacional para mejorar la realización de una actividad, ejemplo de estas pueden ser tareas de ensamblado que requieran alta precisión, colaboración entre robots y humanos, operaciones de robots móviles, entre otras (Roveda y col., 2021). Es por este motivo que el desarrollo de esta investigación busca trabajar con robots de servicio para fábricas inteligentes, cuyo objetivo es realizar tareas útiles para humanos o otros robots en entornos industriales. En este tipo de situaciones es importante que el robot de servicio sea capaz de planear la ruta mediante la cual navegará dentro del entorno, además de qué trayectoria de movimiento seguirá el manipulador, calculando la fuerza a ejercer en cada uno de los actuadores que se encuentren involucrados en dicho movimiento, la orientación desde la que debe aproximarse al objeto que desee manipular, etcétera (Rosenbaum y col., 2006).

Este proyecto se realizó en las instalaciones del laboratorio de Bio-robótica de la Facultad de Ingeniería, bajo la asesoría de los Doctores Jesús Savage Carmona y Marco Antonio Negrete Villanueva; la robótica de servicio ha sido el tema de principal interés entre de los proyectos desarrollados por los colaboradores del laboratorio en años anteriores, por lo que el presente es uno de los primeros trabajos en que se explora la robótica industrial.

Para lo anterior, se consideró posible extrapolar las técnicas y algoritmos explorados previamente en robótica de servicio y evaluar su desempeño en ambientes industriales. Además, aplicando estas necesidades al entorno social en que se desarrolla este proyecto, se considera conveniente permanecer actualizados con los desafíos que se presentan tanto en términos académicos como en los requerimientos de la industria, por lo que proponer el desarrollo y estudio de estos robots y la incursión en este cambio de paradigma sobre la forma en que las

---

fábricas han funcionado tradicionalmente, puede representar una colaboración de valor para el laboratorio de Biorrobótica.

Para cumplir con esta meta, se requiere una modificación de las capacidades y herramientas de los robots para ser utilizados en escenarios industriales e interactuar con los elementos y personas que los componen, es por eso que uno más de los objetivos que se persiguen es otorgar a un robot suficiente autonomía para que realice las tareas que se le asignan requiriendo la menor intervención humana posible. El avance tecnológico previamente mencionado da lugar al uso de nuevos instrumentos que permiten enriquecer la diversidad, la cantidad, la precisión y el tipo de datos que recaba el robot sobre su entorno (Basco y col., 2018). Estos datos en conjunto pueden ser utilizados para mejorar el desempeño del robot.

## 1.2. Planteamiento del problema

La visión computacional, la planeación de acciones y la manipulación de objetos siguen siendo consideradas como problemas abiertos en el campo de la robótica. Evidencia de esto son las diversas competencias que se llevan a cabo año con año, en las cuales se evalúan, entre otras, las herramientas mencionadas. (Sun y col., 2022). Aunado a estas áreas de oportunidad en el desempeño de los robots, es posible integrar dispositivos que permitan recabar más información del ambiente, buscando reducir riesgos y mejorar progresivamente el conocimiento del robot sobre su entorno. En las competencias mencionadas es posible evaluar el desempeño que distintos enfoques tienen para resolver el mismo problema, y abren oportunidades de colaboración entre entidades educativas de diferentes países y permiten la discusión de problemáticas comunes entre estas instituciones.

El procedimiento propuesto se inspira en la competencia *RoboCup Logistics League* que se realiza haciendo uso la plataforma educativa **Robotino** de la empresa **FESTO**, en el cual un equipo de máximo 3 robots debe trabajar de manera colaborativa en la ejecución de diferentes tareas que componen la simulación de un proceso de producción industrial (RoboCup & Mathworks, 2022). Esto requiere compartir información y navegar en un espacio común dentro del cual se encuentran estaciones de trabajo y otros robots que también se encontrarán en movimiento y posiblemente haciendo uso de las estaciones de trabajo.

Esta competencia tiene como objetivo la construcción de varios tipos de *productos*, que consiste en el ensamblado de piezas. Dichos productos se pueden encontrar conformados por distintas combinaciones de cantidad y color de elementos en un orden específico. Este planteamiento hace indispensable poder identificar las características de los elementos que se encuentren en las estaciones de trabajo, y poder clasificarlas por categoría y color, así como determinar la forma óptima en que es conveniente que el robot manipule dichas piezas, por lo que la implementación de algoritmos de visión es también uno de los principales objetivos de esta tarea.

Dado que se trata de la primera incursión en esta competencia, el plantea-



---

miento para este proyecto aborda una sección limitada de la misma, donde se evalúan las habilidades aisladas que es necesario que el robot pueda realizar antes de participar en el proceso de ensamblado completo. Estas habilidades son: **navegación, exploración, y manipulación de objetos.**

### 1.3. Hipótesis

Es posible aumentar el nivel de autonomía de un robot de servicio para fábricas inteligentes al poder planear una secuencia de acciones para realizar movimientos haciendo uso de los sensores disponibles logrando un análisis de su entorno, así como optimizar este análisis al implementar un sistema de visión activa que sea capaz de adaptarse a las condiciones del ambiente, mejorando el desempeño general del robot.

### 1.4. Objetivos

#### 1.4.1. General

Integrar la Visión Computacional como parte del comportamiento de un robot móvil que colabore en actividades de producción industrial además de llevar a cabo una secuencia de acciones que permita llevar a término la cadena de elaboración de un producto final.

#### 1.4.2. Específicos

- Procesar la imagen obtenida por una cámara RGB en la estructura de un robot para identificar objetos en el espacio de trabajo de acuerdo a sus características.
- Utilizar la nube de puntos obtenida de la cámara de profundidad para conocer la posición relativa de los objetos y plataformas de interés y planear una trayectoria del manipulador para tomarlos.
- Identificar los objetos de interés presentes dentro del espacio de trabajo de un robot de servicio para fábricas inteligentes y sujetarlos exitosamente con el manipulador del robot.

### 1.5. Descripción del documento

En este segmento se describe la estructura del documento, buscando dar una semblanza de su contenido, dividido en siete capítulos, compuestos de la siguiente manera:

- En el primer capítulo se encuentra la introducción del proyecto, en ella se establecen la motivación para desarrollar el proyecto, el problema que se busca resolver, la hipótesis y los objetivos que se establecieron.

- 
- En el segundo capítulo se desarrollan los antecedentes teóricos que fueron necesarios para realizar el proyecto, así como las técnicas utilizadas para la implementación.
  - En el tercer capítulo se explica el proceso seguido para la detección de los objetos de interés haciendo uso de visión computacional.
  - En el cuarto capítulo se describen las herramientas utilizadas para la planeación de movimientos del manipulador.
  - En el quinto capítulo se presentan los métodos propuestos para la planeación de acciones del robot.
  - En el sexto capítulo se exponen las herramientas mediante las cuales fue implementado el proyecto y los resultados obtenidos de los experimentos realizados.
  - En el séptimo capítulo se presentan las conclusiones obtenidas de los resultados del proceso y algunas propuestas de trabajo futuro para el proyecto.

---

---

## Capítulo 2

# Antecedentes

Durante este capítulo se describen las bases teóricas que sustentan este proyecto: Visión computacional y procesamiento de imagen y Planeación de movimientos. Se describen herramientas como segmentación por color, operaciones morfológicas, filtrado y detección de bordes, análisis de los movimientos de un manipulador, entre otras.

### 2.1. Robots de Servicio en entornos Industriales

Bajo la opinión de Babel (2022) La Industria 4.0 tiene como objetivo la *interconexión directa e inteligente* de personas, máquinas y productos. De acuerdo con el autor, esta *interconexión inteligente* se refiere en este contexto a:

- Producción flexible.
- Fábricas convertibles.
- Soluciones enfocadas en el consumidor.
- Logística optimizada.
- Red de datos estandarizada.
- Administración de reciclaje de recursos.
- Digitalización de la producción.

El autor también describe que bajo su perspectiva, la llamada Industria 4.0 consiste en la producción industrial en red y que busca optimizar la cadena de valor y mejorar la administración del ciclo de vida de las plantas y fábricas en la red global. La introducción de robots de servicio móviles que realicen actividades de apoyo dentro de las instalaciones de una fábrica pueden contribuir con estos dos últimos objetivos, colaborando en la transformación de las plantas hacia un modelo más flexible y permitiendo diversificar las actividades que se realizan en ellas.

---

Tradicionalmente, la robótica de servicio ha tenido como objetivo ser de ayuda para usuarios que por cualquier motivo se encuentren con dificultades para realizar tareas que sean repetitivas, sucias o incluso peligrosas, teniendo como foco aquellas actividades a realizar dentro de un entorno doméstico. En la actualidad, este tipo de robots aún se encuentran en constante desarrollo y no se cuenta todavía con una introducción real de estas herramientas en la cotidianidad de las ciudades, siendo un poco más comunes en centros de desarrollo como universidades o institutos de investigación. Sin embargo, las habilidades mediante las cuales llevan a cabo las tareas que se menciona, son extrapolables y útiles fuera del ambiente doméstico.

De acuerdo con la Federación Internacional de Robótica, se define un **robot de servicio profesional** como *"aquel que es utilizado en tareas comerciales, y para el cual se requiere cierto grado de autonomía, ya sea parcial o completa (2020)"*. Los robots que se busca tratar en este documento cuentan con un grado alto de autonomía, partiendo de la idea de que, una vez asignada una tarea y con el conocimiento previo que se considera necesario, el robot debe ser capaz de llevar dicha tarea a término, siendo idealmente resistente a cambios en las condiciones del entorno.

## 2.2. Visión Computacional

La visión es una de las herramientas básicas con la que la mayoría de los seres vivos cuenta, es un proceso que realizamos día a día sin que nos represente un gran esfuerzo y a través de ella somos capaces de distinguir formas, profundidades, colores, entre otras características de los objetos de forma automática, realizando complicados cálculos y análisis sin ser conscientes de ello. Es por este motivo que utilizar sistemas para aprovechar la información disponible mediante el análisis computacional de imágenes es una temática que se encuentra con frecuencia entre las investigaciones desde hace décadas. Uno de los ejemplos más representativos de esto es la obra póstuma *Visión*, del científico cognitivo David Marr (1945-1980) publicada en 1982. De acuerdo con Freire, 2007, en dicho trabajo Marr propuso los siguientes tres niveles en los cuales descompone la percepción visual humana:

- **Computacional:** en el cual se determina aquello que se busca estudiar.  
*¿Qué tarea se realiza?*
- **Representación y algoritmo:** donde se plantea la representación que se le dará a las entradas y salidas, además del algoritmo que se utilizará para su procesamiento, es decir, la forma en que el sistema cognitivo realiza la actividad.  
*¿Cómo se lleva a cabo la tarea?*
- **Implementación en hardware,** corresponde al mecanismo físico que se ha de usar para obtener las representaciones y ejecutar el algoritmo seleccionados en el nivel anterior.  
*¿Qué dispositivo puede realizar la tarea? 2007*

---

El autor Richard Szeliski, 2022, menciona que el objetivo de la visión computacional es describir el mundo que vemos en una o más imágenes y reconstruir sus propiedades (forma, iluminación o distribución de colores). Simulando así una forma de imitar el comportamiento de la visión de los seres vivos, mediante máquinas y algoritmos computacionales.

Haciendo una relación con la teoría de Marr, los dos primeros niveles son conceptualmente similares. El tercer nivel, el de la implementación, es el que encuentra el mayor cambio, reemplazando a los agentes biológicos por equipos de cómputo que puedan otorgar la información necesaria para extraer la misma información de las imágenes que obtendría el sistema de visión humano.

Afortunadamente, la visión computacional se encuentra en constante desarrollo, tanto en cuestión de hardware como en relación a los algoritmos que se utilizan, y su uso se encuentra presente en gran variedad de aplicaciones.

El elemento base de este análisis es la **Imagen**, la cual, en palabras de Gonzalez y Woods, 2002, se define como una función bidimensional  $f(i, j)$ , donde  $i$  y  $j$  son coordenadas espaciales de un plano, mientras que la amplitud de  $f$  con cualquier par de coordenadas  $(i, j)$  se conoce como *intensidad* o *nivel de gris* de la imagen en dicho punto. Se considera una **Imagen Digital** siempre y cuando  $i$ ,  $j$  y los valores de la amplitud de  $f$  sean finitos. Los autores destacan también que la imagen digital se compone de un número finito de elementos, cada uno de los cuales tiene una ubicación y valor particulares, comúnmente conocidos como **píxeles**.

### 2.2.1. Espacios de color

Para facilitar el manejo de los datos dentro de las imágenes digitales, se utilizan los llamados *Espacios de color*, que estandarizan las características de los colores, representándolos en un sistema coordinado en el cual cada color es representado por un punto único dentro de dichos espacios (Gonzalez y Woods, 2002). Es decir, dentro de una imagen digital a color, cada píxel de determinado color tiene asociado un punto dentro del respectivo Espacio.

#### Espacio RGB

De acuerdo con Pratt, 2014, las imágenes que se obtienen por medio de cámaras o escáneres de color cuentan con tres sensores principales, cada uno sensible a cierto rango de frecuencias que se asocian a los colores rojo, verde y azul del espectro de luz perceptible por el ojo humano. Estos sensores generan señales de color que son linealmente proporcionales a la cantidad de luz roja, verde o azul detectada, de esta forma se obtiene una terna de valores que resultan en el color mostrado en cada píxel de una imagen digital. Con esta información, se obtiene una representación numérica de las imágenes basada en las proporciones de cada uno de estos colores, formando en espacio de color RGB, *Red*, *Green*, *Blue*. A partir de este planteamiento es posible visualizar el comportamiento de los colores en el espacio tridimensional usando como ejes las proporciones de rojo, verde y azul, donde las coordenadas del vector obtenido con los valores

---

de cada uno de los colores primarios resulta en la representación en el espacio RGB.

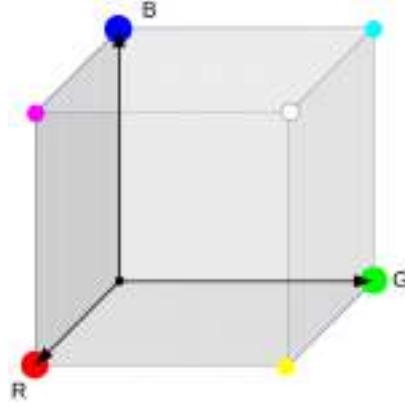


Figura 2.1: Espacio RGB

### Espacio HSV

Szeliski describe el espacio HSV, formado con los componentes Matiz (Hue,  $H$ ), Saturación, (Saturation,  $S$ ), Brillantez o Intensidad (Value  $V$ , Brigttness  $B$  o Intensity,  $I$ ), mencionando que es una transformación no lineal del espacio de color RGB, correspondiente de igual manera a una tercia de valores que describen el color de interés.

Czerwinski y Kania, 2013, detallan que este espacio de color separa la intensidad y el color, mientras que el matiz y la saturación corresponden a la percepción humana, haciendo muy útil esta representación para desarrollo de algoritmos de procesamiento de imagen, aclarando también que el uso de los valores en RGB puede hacer la manipulación de la imagen propensa a deformaciones de la percepción humana del color.

Mencionan también que el espacio RGB es utilizado para almacenamiento, procesamiento, codificación o presentación de las imágenes en televisiones, a diferencia del espacio HSV, cuyas aplicaciones se encuentran más enfocadas a la percepción del color y al uso de la imagen en gráficos de computadora.

Los valores asociados a cada uno de los parámetros del espacio HSV son, como se mencionó anteriormente, transformaciones no lineales de los valores  $R, G, B$  y se pueden obtener mediante las ecuaciones de transformación descritas por Burger y Burge, 2022, en las ecuaciones 2.1 a 2.7:

Siendo:

$$C_{min} = \min(R, G, B), \quad C_{max} = \max(R, G, B), \quad \Delta = C_{max} - C_{min} \quad (2.1)$$

---

Entonces,

$$S = \begin{cases} \frac{\Delta}{C_{max}}, & \text{para } C_{max} > 0. \\ 0, & \text{otros casos,} \end{cases} \quad (2.2)$$

Con el objetivo de mantener los valores de estas transformaciones entre  $[0, 1]$ , se divide entre el valor máximo que será posible que tenga la representación, comúnmente 255.

$$V = \frac{C_{max}}{255}, \quad (2.3)$$

$$R' = \frac{C_{max} - R}{\Delta}, \quad G' = \frac{C_{max} - G}{\Delta}, \quad B' = \frac{C_{max} - B}{\Delta}, \quad (2.4)$$

Posteriormente, dependiendo de cuál de los colores en la representación RGB tuviera el valor máximo, se calcula un valor preliminar del matiz, de la siguiente forma:

$$H' = \begin{cases} B' - G', & \text{cuando } R = C_{max}, \\ R' - B' + 2, & \text{cuando } G = C_{max}, \\ G' - R' + 4, & \text{cuando } B = C_{max}, \end{cases} \quad (2.5)$$

Lo que resulta en una representación dentro del intervalo  $[-1, 5]$  y finalmente se obtiene la normalización dentro del intervalo  $[0, 1]$  como se muestra a continuación.

$$H = \frac{1}{6} \cdot \begin{cases} (H' + 6), & \text{cuando } H' = 0, \\ H', & \text{otros casos,} \end{cases} \quad (2.6)$$

Finalmente, para obtener la representación de la componente H como el correspondiente a un ángulo se multiplica por 360 (Freire, 2007).

$$H^\circ = H \cdot 360 \quad (2.7)$$

Este espacio puede ser gráficamente representado como un prisma circular invertido, donde el radio de la base representa el valor de la saturación (S), La altura del cono representa la intensidad (V), y el matiz (H) se obtiene mediante un ángulo trazado sobre la base del cilindro, comenzando con el color rojo, que se encuentra en 0 grados, el color verde en 120 grados y finalmente el color azul en 240 grados.



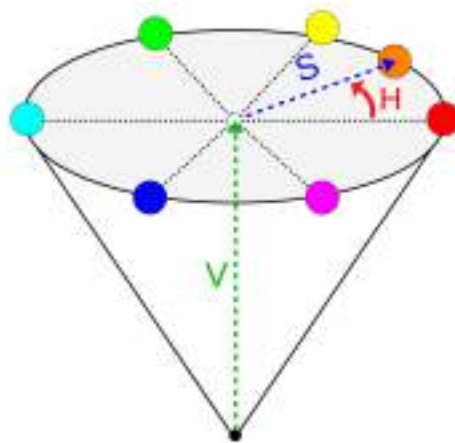


Figura 2.2: Espacio HSV, representación cónica

### 2.2.2. Segmentación de Imágenes

En Gonzalez y Woods, 2002, los autores comentan que una segmentación robusta es un gran avance en el proceso de procesamiento de una imagen en la cual sea necesario identificar objetos. Adicionalmente, Sonka y col., 2008, ahondan en las características de la **segmentación completa** y la **segmentación parcial**. Explican que en la primera, se obtiene como resultado un conjunto de regiones no superpuestas que corresponden totalmente con *objetos del mundo real* en la imagen de entrada. Mientras que en la segmentación parcial las regiones no corresponden directamente con *objetos del mundo real* presentes en la imagen, sino que aíslan características específicas de la imagen, como la brillantez, el color, la textura, etc. Dada la complejidad asociada a la segmentación completa de las imágenes, para obtenerlas es necesario aplicar niveles más altos de procesamiento a comparación de la segmentación parcial.

Se dice que la segmentación completa de una imagen  $R$  es el conjunto finito de regiones  $R_1, \dots, R_S$ , expresado matemáticamente como se muestra en la ecuación 2.8:

$$R = \bigcup_{i=1}^S, \quad R_i \cap R_j = 0, \quad i \neq j \quad (2.8)$$

### Umbralización

La umbralización es una de las herramientas más antiguas para realizar segmentación de imágenes, y tiene como ventaja que no requiere muchos recursos computacionales para realizarla, por lo que es todavía ampliamente utilizada en diferentes aplicaciones. Esta técnica consiste en establecer valores dentro de los cuales se considera aceptable la característica de interés.

---

Este proceso corresponde a la transformación de una imagen de entrada  $f$  a una salida segmentada, una imagen binaria  $g$ , de la siguiente forma:

$$g(i, j) = \begin{cases} 1, & \text{para } f(i, j) \geq T, \\ 0, & \text{para } f(i, j) < T \end{cases} \quad (2.9)$$

Donde  $T$  representa el umbral (Threshold). Después de esta transformación, todos los píxeles dentro de la imagen que se encuentren dentro del/los umbrales definidos tendrán un valor de 1 y a aquellos que no satisfagan la condición se les asigna el valor 0 (o vice versa) (Sonka y col., 2008).

Gonzalez y Woods, 2002, describen el proceso de segmentación de imágenes en el espacio HSV de la siguiente manera:

- Separar la imagen en sus componentes HSV.
- Reconocer las zonas de la imagen (y los valores de los píxeles) donde se encuentran las características de interés.
- Establecer los umbrales de los valores HSV que determinan los píxeles que representan la información de valor.
- Umbralizar: generar una máscara binaria con las regiones donde se encuentran dichos valores.
- Utilizar la máscara y la imagen original para aislar las regiones de interés de la imagen.

### 2.2.3. Operaciones Morfológicas

Los autores explican también que bajo el contexto de procesamiento de imágenes la *morfología matemática* hace alusión a una herramienta para extraer los componentes de la imagen que son útiles para la representación y descripción de la forma de una región. Indica además que el lenguaje que se ha de utilizar para este procesamiento es la teoría de conjuntos y resaltan que los conjuntos en *morfología matemática* representan objetos en una imagen. Debido a la relación mencionada entre el procesamiento morfológico de la imagen y la teoría de conjuntos, se explican en enseguida algunos conceptos iniciales.

#### Teoría de Conjuntos

Sea  $A$  un conjunto perteneciente a  $Z^2$ . Si  $a = (a_1, a_2)$  es un elemento de  $A$ , se representa de la siguiente forma:  $a \in A$ . De forma similar, si  $a$  no es un elemento de  $A$ , se representa como:  $a \notin A$ .

Aquel conjunto en el cual no hay elementos se conoce comúnmente como el conjunto *nulo* o conjunto *vacío* y se utiliza el símbolo  $\emptyset$ .

Un **conjunto** se representa por lo contenido entre dos llaves:  $\{\cdot\}$  y los elementos de los conjuntos que son de nuestro interés en este contexto son los píxeles que componen una imagen digital.

---

Si todos los elementos de  $A$  se encuentran también en el conjunto  $B$ , se dice que el conjunto  $A$  es un **subconjunto** del  $B$  y se escribe de la siguiente forma:

$$A \subseteq B \quad (2.10)$$

La **unión**  $C$  de los conjuntos  $A$  y  $B$  es aquel conjunto al que pertenecen todos los elementos ya sea de  $A$ ,  $B$  o de ambos y se expresa como:

$$C = A \cup B \quad (2.11)$$

La **intersección**  $D$  de dos conjuntos es el conjunto que contiene a los elementos que pertenecen a  $A$  y a  $B$ :

$$D = A \cap B \quad (2.12)$$

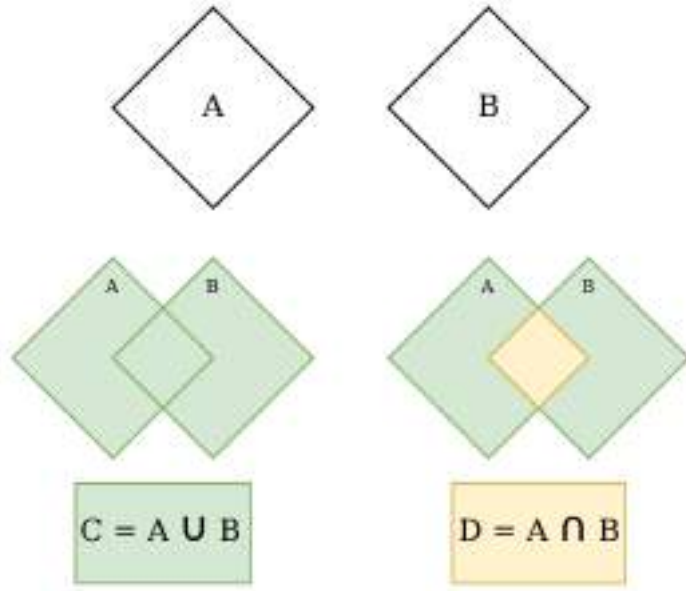


Figura 2.3: Unión e Intersección de Conjuntos

Se dice que dos conjuntos son **mutuamente excluyentes** si no tienen ningún elemento en común,

$$A \cap B = \emptyset \quad (2.13)$$

El **complemento** del conjunto  $A$  es el conjunto de todos aquellos elementos que no pertenecen a  $A$ .

$$A^C = \{\omega \mid \omega \notin A\} \quad (2.14)$$

---

Dado el conjunto Universal  $U$ :

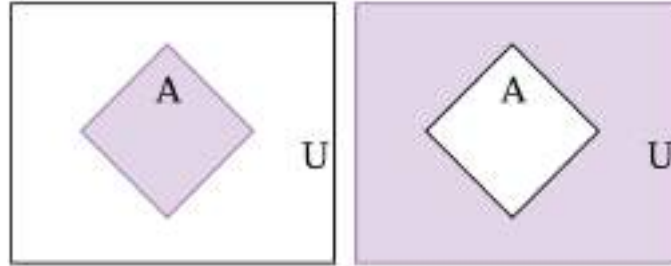


Figura 2.4: Complemento

La **diferencia** entre dos conjuntos  $A$  y  $B$  representada como  $A - B$ , es el conjunto de elementos que pertenecen a  $A$  pero no a  $B$  se define como:

$$A - B = \{\omega \mid \omega \in A, \omega \notin B\} = A \cap B^C \quad (2.15)$$

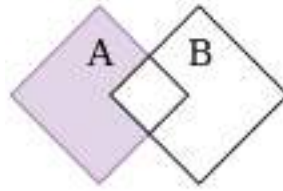


Figura 2.5: Diferencia

La reflexión del conjunto  $B$  se define como:

$$\hat{B} = \{\omega \mid \omega - b, \text{ para } b \in B\} \quad (2.16)$$

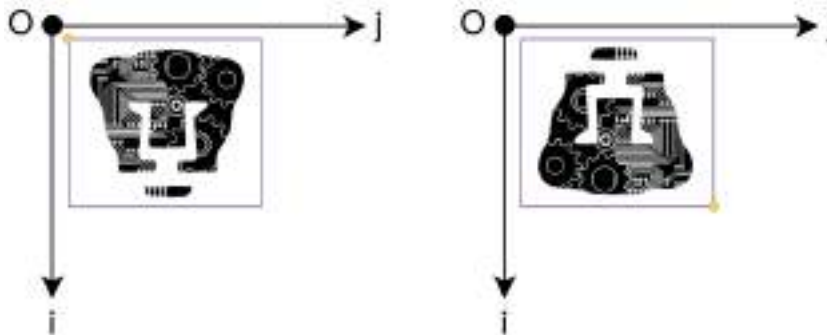


Figura 2.6: Reflexión

La traslación del conjunto  $A$  al punto  $Z = (z_1, z_2)$ , representada como  $(A)_z$ , se define como:

$$(A)_z = \{c \mid c = a + z, \text{ para } a \in A\} \quad (2.17)$$

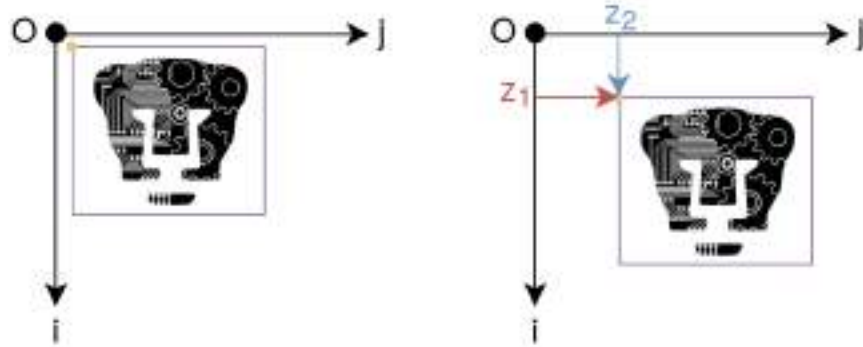


Figura 2.7: Traslación

En las figuras 2.6 y 2.7 el origen de las imágenes originales se encuentra resaltado con un punto amarillo.

### Operaciones Lógicas

Comúnmente el procesamiento morfológico se realiza sobre imágenes en blanco y negro, es decir, imágenes en donde los únicos valores posible para los píxeles son 1 y 0, esto permite que se utilicen operaciones binarias sobre los píxeles de la imagen. Algunas de las operaciones binarias que con más frecuencia se utilizan para el procesamiento de imágenes son las operaciones **AND**, **OR**, y **NOT**. Las dos primeras permiten operar entre dos o más imágenes, como en el caso de aplicar una máscara sobre una imagen, utilizando la operación AND. En la Figura 2.8 se muestra el comportamiento de estas operaciones lógicas.




							
AND			OR			NOT	
X	Y	Z	X	Y	Z	X	Z
0	0	0	0	0	0	0	1
0	1	0	0	1	1	1	0
1	0	0	1	0	1		
1	1	1	1	1	1		

Figura 2.8: Operaciones lógicas AND, OR y NOT

---

Estas operaciones tienen comportamientos similares a los mencionados en teoría de conjuntos, con la limitante de que las operaciones lógicas solo pueden ser utilizadas con valores binarios después de la umbralización de los valores de los píxeles de la imagen, en la Figura 2.9 se observa un ejemplo del comportamiento de estas operaciones en una imagen binaria.

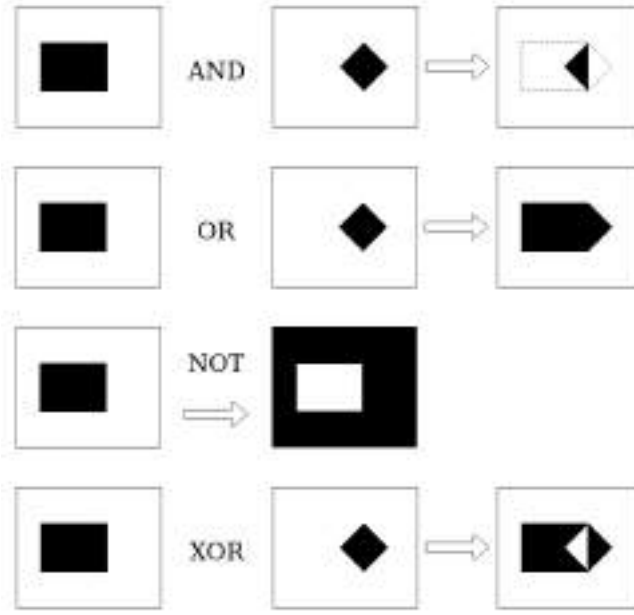


Figura 2.9: Operaciones lógicas aplicadas a imagen

### Dilatación y Erosión

Estas operaciones se consideran básicas dentro el procesamiento morfológico de las imágenes y algunas de las técnicas más avanzadas que se utilizan, se basan en ellas.

#### *Dilatación*

Siendo  $A$  y  $B$  dos conjuntos en  $Z^2$ , la dilatación de  $A$  y  $B$ , expresada como  $A \oplus B$ , se define como:

$$A \oplus B = \{z \mid (\hat{B})_z \cap A \neq \emptyset\} \quad (2.18)$$

Para obtener esta ecuación es necesario obtener la reflexión de  $B$  sobre su origen e invirtiendo su reflexión en  $z$ . La dilatación de  $A$  por  $B$  es el conjunto de todos los desplazamientos de  $z$ , tal que  $B$  y  $A$  se superponen por al menos un elemento. De acuerdo con esta descripción, es posible re-escribir la anterior definición como:

$$A \oplus B = \{z \mid [(\hat{B})_z \cap A] \subseteq A\} \quad (2.19)$$

---

en estas ecuaciones, el conjunto  $B$  es comúnmente mencionado como el *elemento estructurante* de la dilatación y otras operaciones.

Vale la pena mencionar que esta definición de la operación dilatación coincide con el proceso para realizar la convolución, dado que de forma similar se *invierte* uno de los operandos y se recorre sobre el otro elemento de la operación. Es importante recalcar que la convolución es una más de las operaciones que con frecuencia se utilizan para el procesamiento de imágenes, por ejemplo para el filtrado de las imágenes, al hacer la convolución de la forma matricial de la imagen con la matriz que represente el filtro que se busca aplicar.

#### Erosión

Para los conjuntos  $A$  y  $B$ , pertenecientes a  $Z^2$ , la operación erosión, representada como  $A \ominus B$ , se define como:

$$A \ominus B = \{z \mid (B)_z \subseteq A\} \quad (2.20)$$

Es decir, la erosión aplicada a estos conjuntos resulta en el conjunto de todos los puntos  $z$ , tal que  $B$  (traducido en puntos  $z$ ), que son contenidos en  $A$ .

En la figura 2.10 se muestra un ejemplo de los efectos de la operación morfológica, a)imagen original, b)matriz de 3x3 como elemento estructurante, c)operación erosión aplicada a la imagen original, d)operación dilatación aplicada a c).

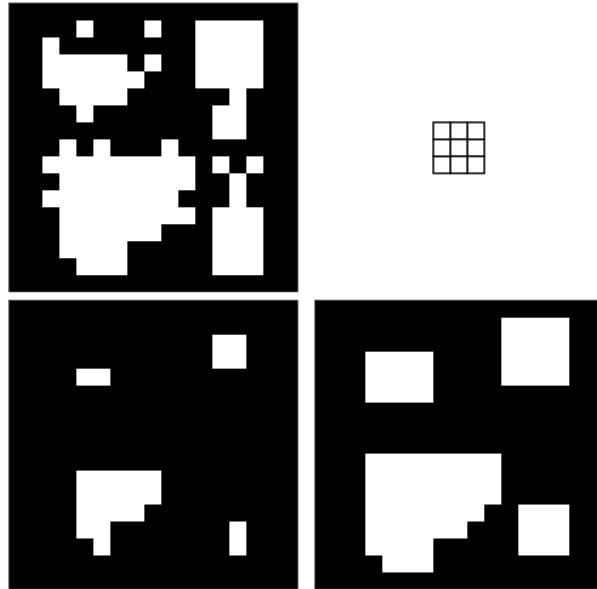


Figura 2.10: Operaciones Morfológicas: dilatación y erosión

Como es posible notar en el ejemplo, estas dos operaciones no son estrictamente inversas entre sí. Además, dado que al realizar la operación erosión se

---

pierde información sobre los detalles de la imagen original es imposible recuperarla aplicando la dilatación. Adicionalmente, elemento estructurante tiene gran influencia en el resultado que se obtendrá, ya que una modificación en sus dimensiones o su forma resulta en una evidente modificación de la imagen final, un ejemplo de esto puede verse en la figura 2.11, utilizando nuevamente 2.10, inciso c) como imagen base.



Figura 2.11: Dilatación aplicada con distinto elemento estructurante

## 2.3. Reconocimiento de Objetos

Una de las partes fundamentales de la visión computacional es la habilidad de reconocer objetos dentro de las imágenes que son procesadas, para lo cual se utilizan algoritmos de reconocimiento de patrones que nos permiten clasificar objetos o regiones dentro de la imagen, haciendo posible entender mejor los elementos que la componen.

Como se ha mencionado anteriormente, para nosotros las tareas asociadas a la visión son actividades que llevamos a cabo de manera natural, sin que requiera un esfuerzo consciente en la mayoría de los casos y, teniendo ya sea un ejemplo visual o bien una descripción sobre las características de nuestro objeto de interés, somos capaces de identificarlo dentro del entorno. Con el objetivo de reconocer los objetos mediante el análisis computacional de las imágenes, es imprescindible conocer las características de dichos objetos, así como de las clases a las que pertenecen.

De acuerdo con Sonka y col., 2008, el diseño de una adecuada representación del conocimiento es la parte más importante de la resolución del problema del entendimiento y se refieren a las descripciones y las características como una representación de *bajo nivel*, aclarando que no pueden ser consideradas en sí mismas como representaciones del conocimiento, aunque sí pueden ser usadas como parte de una estructura de representación más compleja.

Se ha dicho antes que como humanos podemos utilizar descripciones verbales o escritas de la apariencia de un objeto y utilizar esa información para identificarlos, sin embargo, para que una máquina pueda replicar este comportamiento es necesario tener una representación de esta descripción que sea posible procesar matemáticamente. En este contexto las representaciones se hacen de forma



---

numérica, asociando cada característica a una magnitud escalar. Para poder hacer la clasificación de objetos complejos es preferible que la descripción incluya la representación de varias propiedades del objeto en cuestión. Al conjunto de características representadas numéricamente se le conoce como **vectores de características**, que se utilizan como entradas para los algoritmos de reconocimiento. A este vector de características se le conoce también como **patrón**.

En el reconocimiento se les asignan clases a los objetos. En este proceso, el número de clases existentes es frecuentemente conocido desde el planteamiento del problema. Dependiendo del conjunto de las características de la clase que las propiedades del objeto satisfacen, se asignan a una u otra; el elemento encargado de esta la tarea de clasificación es conocido como **clasificador**.

El proceso principal para el reconocimiento de patrones se muestra en la figura 2.12, donde podemos ver que un objeto es la entrada de un bloque de procesamiento del cual se obtiene el **patrón** del objeto, es decir, se construye el **vector de características** al aislar y cuantificar las propiedades del objeto. Este vector es posteriormente ingresado al clasificador y como salida del proceso se obtiene la clase a la que el objeto pertenece.



Figura 2.12: Reconocimiento de objetos

Dependiendo de la aplicación para la cual se busque realizar el reconocimiento serán importantes diferentes características de los objetos. Además, las características de los recursos disponibles también encontrarán cambios. Debido a esto, se han desarrollado muchas y muy variadas técnicas de reconocimiento, utilizando cada vez diferentes características y técnicas de procesamiento de la información. Treiber, 2010, menciona que bajo los siguientes parámetros es posible definir una clasificación de los métodos de reconocimiento de objetos:

- Representación de objetos: basados en la geometría (silueta, forma) o la apariencia (regiones de imagen que pertenecen al objeto).
- Alcance de la información sobre los objetos: características locales (descripción de una parte del objeto) o globales (información sobre el área, perímetro, etc).
- Variación esperada de los objetos: qué tan diferentes pueden ser los objetos dentro de una misma clase.
- Calidad de los datos de la imagen: dependiendo de la aplicación es posible que las imágenes a procesar contengan ruido, oclusiones o menor definición, por lo que requieren más procesamiento.

- 
- Estrategia de comparación: en el proceso de reconocimiento es necesario un paso en el cual se realiza una comparación entre la similitud entre la imagen analizada y la referencia. Dependiendo del algoritmo de comparación usado cambian también los parámetros requeridos.
  - Alcance de la información de los elementos utilizados en la comparación: el autor propone una división en las siguientes tres categorías: valores sin procesar de los píxeles de intensidad, características de bajo nivel (bordes) y alto nivel (líneas o arcos).

Una vez establecidas estas condiciones, el autor lista algunos métodos de reconocimiento de objetos:

- **Métodos globales**  
Buscan modelar y encontrar los objetos únicamente con características globales.
- **Métodos basados en Transformación y Búsqueda**  
La pose del objeto se determina buscando el espacio de transformaciones entre el modelo y la información de la imagen.
- **Métodos basados en Correspondencia Geométrica**  
Utilizan las relaciones geométricas entre varias parte del objeto y establece correspondencias entre el modelo y las características de la imagen.
- **Métodos basados en descriptores**  
Buscan identificar los objetos utilizando descriptores, normalmente de la apariencia de los objetos en regiones locales cerca de los puntos de interés.

El proceso utilizado en este proyecto corresponde a la última categoría de las anteriores, y fue elegido dado que su implementación no requiere muchos recursos computacionales y es suficientemente rápido para la aplicación propuesta. Además, claro, que las características de los objetos de interés lo permiten.

## 2.4. Máquinas de Estado

El término 'máquinas de estado' se refiere a un modelo computacional con el cual se representan procesos de lógica secuencial. Este modelo cuenta con conjuntos de entradas, conjuntos de salidas y el funcionamiento del sistema se compone por estados y funciones de transformación. En este modelo se establece que en cualquier momento de la ejecución es posible estar en uno y solo un estado a la vez, y el salto entre estados se condiciona por medio de las ya mencionadas funciones de transformación, que determinan si se realiza el cambio a otro de los estados.

Czerwinski y Kania, [2013](#), enuncian la definición matemática de la Máquina de Estados Finitos (FSM) utilizando un vector de 5 elementos,  $\{X, Y, S, \delta, \lambda\}$ , donde  $X$  corresponde a un espacio de entradas finitas,  $Y$  al espacio de salidas

---

finitas,  $S$  a un conjunto de estados finitos,  $\delta$  representa la función de transformación y  $\lambda$  la función de salidas. En este caso, la función de transición de una Máquina de estados finitos determina el siguiente estado ( $S^+$ ), y se refiere al mapeo  $\delta : X \times S \rightarrow S$ .

Dentro de las máquinas de estados finitas se encuentran dos grandes clasificaciones, las máquinas Moore y las máquinas Mealy, que reciben su nombre de los investigadores Eduard F. Moore y George H. Mealy, que desarrollaron *la teoría del autómata*. En el planteamiento de la máquina de Moore las salidas dependen únicamente del estado actual de la máquina, expresando este comportamiento en términos de los vectores definidos anteriormente se obtiene:  $\lambda : S \rightarrow Y$ . Por el contrario, en las máquinas Mealy las salidas dependen tanto del estado actual como de las entradas actuales del sistema. Así, la función de salida se puede representar como:  $\lambda : X \times S \rightarrow Y$ .

## 2.5. Sistemas expertos

A mediados del Siglo XX, cuando el concepto de Inteligencia Artificial comenzaba a popularizarse, su aplicación principal se centraba en planeación y solución de problemas. En aquel momento era difícil imaginar que décadas más tarde las aplicaciones más importantes se encontraran en ingeniería del conocimiento y en sistemas expertos. De acuerdo a Giarratano y Riley, 2002, en 1982 el profesor Edward Feigenbaum de la Universidad de Stanford, uno de los pioneros en el desarrollo de tecnología de sistemas expertos, definió un sistema experto como *"un programa computacional que utiliza procedimientos de inferencia para resolver problemas que son suficientemente complicados para requerir conocimiento humano especializado para su solución."*

En 2020, 38 años más tarde de esta definición, Gupta y Nagpal, 2020, refieren que un sistema experto es un programa computacional que emula la capacidad de razonar y el comportamiento de un humano con el conocimiento y experiencia de un experto en un campo específico. También indica que los sistemas expertos son utilizados para resolver problemas complejos utilizando el conocimiento almacenado en una base de datos en forma de reglas.

De acuerdo con Liebowitz, 2019, la adquisición de conocimiento es el proceso de extraer, estructurar y organizar el conocimiento de diferentes fuentes, usualmente humanos expertos en un área, de forma tal que la habilidad de solucionar problemas pueda ser capturada y transformada para que una computadora sea capaz de leerla. De acuerdo con este autor: *"Sin conocimiento explícitamente representado, un sistema experto no es más que un programa de computadora."*

Gupta y Liebowitz señalan también que uno de los pasos intermedios entre la adquisición de conocimiento y la implementación de sistemas expertos es la intervención de un Ingeniero de conocimiento, un individuo que estudia la forma en que los expertos toman decisiones y lo traduce a reglas en términos comprensibles para la máquina. A partir de esto, los autores mencionan que los sistemas expertos son considerados como *Inteligencia Artificial Aplicada* (2020).

---

## 2.6. Manipulación del Robot

Siendo la manipulación de objetos un de los objetivos principales de este proyecto es pertinente explicar la forma en que le es posible al robot tomar los objetos que se espera que transporte. Para esto, es necesario realizar el análisis cinemático del robot, que consiste en el estudio de la relación que guarda el efector final respecto a la estructura que compone el manipulador del robot, es decir el comportamiento que debe tener cada articulación del robot para que el efector final pueda posicionarse en las coordenadas finales deseadas para el comportamiento esperado del robot.

Para realizar este análisis es necesario conocer el número de grados de libertad con los que cuenta el manipulador que se usará, así como el tipo de articulaciones con las que cuenta, los ejes en los que actúa cada una y sus limitaciones de movimiento.

Ceccarelli, 2022, explica que es posible formular numéricamente esta situación para determinar las coordenadas de las articulaciones  $q_i (i = 1, \dots, N)$  como funciones de la configuración del manipulador. Amplía diciendo que la formulación debe considerar aspectos numéricos que están relacionados a la existencia de soluciones y a la multiplicidad de soluciones.

Los autores describen que la existencia de soluciones hace referencia a que cuando una configuración dada no se encuentre dentro del rango de movilidad del manipulador, la solución numérica de la cinemática inversa no puede ser obtenida dentro del espacio de trabajo del manipulador. Para lo cual se requiere conocer el rango de movilidad de las articulaciones y los límites del espacio de trabajo antes de unir la solución numérica. En términos matemáticos, el cumplimiento de los límites del espacio de trabajo significa que las soluciones que se obtengan pertenezcan al dominio de los números reales.

De forma similar, es posible encontrar múltiples soluciones debido a la naturaleza *altamente no lineal* de la posición cinemática de los manipuladores, es decir las diferentes configuraciones en las que se pueden posicionar los actuadores para que el efector final llegue a una coordenada determinada dentro del espacio de trabajo.

## 2.7. Estado del Arte

Durante el desarrollo de este proyecto fue notable que dentro del entorno en que se plantea su desarrollo, los equipos que previamente han participado proceden sin utilizar procesamiento de imágenes en gran medida, usando en su mayoría sensores de proximidad y similares. Si bien las piezas que se tiene como objetivo que el robot identifique y sujete cuentan con características similares en forma, el color es una de las diferencias más evidentes tanto entre ellas como en relación al ambiente en que se encuentran de forma regular y se considera una característica de mucha utilidad en detección de objetos. Aunque es cierto que el entorno de la competencia en que se plantea la evaluación del comportamiento de este desarrollo se encuentra bajo condiciones controladas y es evidentemente una

---

simulación de las condiciones de ejecución reales de un ambiente industrial, se sigue considerando conveniente el uso de cámaras y procesamiento de imágenes que permitan explotar la mayor cantidad de recursos disponibles, además de que este enfoque ofrece un aumento en la versatilidad de la información recabada sobre el desempeño de los sistemas.

### 2.7.1. Antecedentes en RoboCup Logistics

Dentro de la documentación disponible en el sitio web oficial de la RoboCup se listan los *Team Description Paper* (TDP), donde se describen las técnicas utilizadas en la competencia por los equipos participantes. En estos documentos se encuentran pocos referentes de cámaras utilizadas en secciones de la competencia fuera de la localización de los códigos ARUCO en las estaciones. Estas cámaras se encuentran típicamente en la parte inferior de los robots. En el TDP presentado por el equipo austriaco GRIPS (Graz Robust and Intelligent Production System) para su participación en la competencia del 2021 Furbaß y col., 2021, el equipo describe que, para la alineación con la banda transportadora, el robot inicia en una posición de referencia cercana a la estación, desde la cual se utiliza el sensor láser y el código ARUCO que se encuentra en sus paredes para realizar la alineación. El equipo reconoce que este primer alineamiento, combinando la información de estas dos fuentes, no es muy preciso. Por lo anterior, utilizan una segunda cámara que se encuentra en una posición más elevada para los movimientos más finos del proceso.

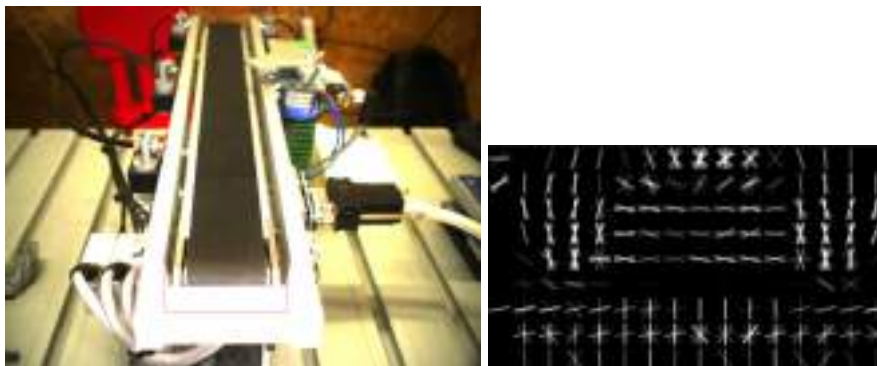


Figura 2.13: Detección de Histogramas de Gradientes Orientados (HOG) 2021

Con este objetivo implementaron un detector de Histogramas de Gradientes Orientados (HOG, por sus siglas en inglés). con el cual detectan la posición de los diferentes elementos de interés dentro del espacio de trabajo. Los autores describen que el HOG identifica regiones de la imagen, en las cuales se encuentran las características más representativas de los elementos ambientales. Para este punto del proceso, las características mencionadas ya eran conocidas por el algoritmo, para lo cual fue necesaria la toma de imágenes de entrenamiento y evaluación. Una vez que se aísla la zona de interés, el HOG genera un conjunto

---

de características. Los autores mencionan que las imágenes de entrenamiento se utilizan para encontrar las características que describen las regiones de interés y las imágenes de evaluación para verificar que dichas características realmente coincidan con los objetos, como es típico en procesos de este tipo.

La descripción del proceso provista por los autores especifica que, una vez que el robot se encuentra en la posición adecuada, la cámara superior toma 4 imágenes con diferentes iluminaciones, logrados con fuentes de luz con las que cuenta el robot. Este esfuerzo se realiza con el objetivo de mitigar los errores debidos a las variaciones de iluminación naturales, y promover una correcta detección de las características de interés aún si las condiciones en las que se encuentra la estación no corresponden con las imágenes con las que fue previamente entrenado el algoritmo. Adicionalmente, para intentar robustecer el procedimiento, los autores utilizaron varios HOG en diferentes objetos, entrenando cada uno con diferentes conjuntos de imágenes, buscando diversificar las características de entrenamiento para su algoritmo de reconocimiento. En la figura 2.14 se observan las modificaciones que el equipo GRIPS llevó a cabo en su robot hasta la edición 2021 de la competencia. Sin embargo, actualmente la implementación ya no cuenta con la cámara elevada y la estructura superior ha sido descartada.



Figura 2.14: Robotino modificado - GRIPS Furbaß y col., [2021](#)

Para realizar la tarea de alineación, el equipo utiliza una combinación de sensores de proximidad que se encuentran debajo de y en el centro de la pinza del manipulador, utilizando también un movimiento que combina la alineación

---

del robot con la máquina y del manipulador con la banda transportadora.

De manera similar, dentro del desarrollo del trabajo para la siguiente participación en la competencia, se planea integrar sensores de proximidad, combinando los avances obtenidos del presente proyecto, con las herramientas vistas durante lo experimentado en la edición 2023. Cabe mencionar que la configuración mecánica de los manipuladores de ambos equipos es y seguirá siendo distinta.

## Capítulo 3

# Detección de objetos

Uno de los objetivos del desarrollo de este proyecto es procesar la imagen capturada por una cámara RGB que forma parte de la estructura de un robot de servicio industrial, cuya tarea es manipular objetos entre estaciones de trabajo, buscando mejorar su desempeño utilizando la información obtenida. En pos de esto, se realizó el siguiente desarrollo.

### 3.1. Objetos de interés

Los objetos que se busca identificar tienen características muy específicas y son fácilmente identificables dentro del espacio de trabajo del robot. Este fue uno de los motivos por los que el procedimiento actual fue propuesto. Se trata de las **bases**, cilindros huecos de colores sólidos (negro o rojo), así como de **anillos** que es posible apilar sobre dichos cilindros y entre ellos, a manera de piezas intermedias, estos pueden ser amarillos, verdes o azules. Tanto las bases como los anillos tienen  $4,5 \times 4,5[cm]$  de diámetro. El objetivo de esta etapa es identificar dichos objetos dentro de las imágenes obtenidas por la cámara del robot y asignarles una coordenada cartesiana asociada al mapa que el robot utiliza como referencia. Estas piezas destacan por su color, dado que los demás elementos en el espacio de trabajo del robot tienen colores predominantemente azules o grises.

Como se puede observar en la figura 3.1 las diferencias de color entre las piezas son claras, por lo que es posible diferenciarlas entre sí utilizando como herramienta la segmentación por color. Además, dentro del espacio de trabajo (ver figura 3.2) del robot, las piezas pueden encontrarse en un área muy reducida dentro de la cual es muy poco probable que se encuentre algún elemento con características de color similares.

Adicionalmente, la región de interés dentro del espacio de trabajo es bastante reducida, dado que el procesamiento de las piezas se lleva dentro de las estaciones de forma independiente al comportamiento del robot, por lo que se cuenta con dos regiones importantes dentro del espacio de trabajo sobre las estaciones,





Figura 3.1: Objetos de interés



Figura 3.2: Ejemplo del espacio de trabajo del robot

que corresponden a la entrada y la salida de la banda transportadora, de donde el robot debe tomar la pieza para ser procesada y, una vez que la estación de trabajo ha concluido su tarea, debe recoger la pieza de la salida de la banda transportadora para llevarla al siguiente paso en la secuencia de ensamblado.

### 3.2. Caracterización de los objetos

El primer paso dentro del procedimiento fue capturar imágenes de los objetos bajo diferentes condiciones de luz. Dada la curvatura de la superficie de los objetos, así como la regiones que en el interior de los cilindros, se obtenían variaciones de color debidas a las sombras, por este motivo, se tomó la media de color no solo con cambios de iluminación sino también considerando diferentes regiones de las piezas.

---










Cambios de Iluminación				
Iluminación Natural				
Iluminación Artificial				
Poca Iluminación				

Figura 3.3: Efectos de los cambios de iluminación en las piezas

Con las imágenes de referencia obtenidas en el espacio de color RGB se realizó la conversión al espacio HSV, en el cual se llevó a cabo la obtención de los vectores correspondientes a las medias de color en este espacio.



Figura 3.4: Ejemplo de anillo amarillo en: izq) HSV y der) RGB

Una vez que se tiene esta información se pasó a la umbralización de las imágenes a evaluar (en este caso el vídeo proporcionado por la cámara KINECT). De esta forma, cada cuadro que compone al vídeo se convierte al espacio HSV y se filtran aquellos píxeles que corresponden con la media de color de las imágenes de referencia.

Para este filtrado fue necesario establecer los umbrales superior e inferior dentro de los cuales se considera que se cumplen las condiciones de la media de color. Estos umbrales permiten asignar una cierta tolerancia para los píxeles que por cambios de iluminación no correspondan exactamente con la media buscada.

Para considerar estos cambios se realizó el proceso de umbralización considerando diferentes medias de color, obteniendo entonces varias máscaras que representaban al objeto dependiendo de las secciones del mismo cuyas medias de

---

	(171.7043650793651, 250.1154761904762, 190.6579365079365, 0.0)
	(165.65058055152394, 254.7634252539913, 254.6455007256894, 0.0)
	(172.40547588005214, 216.39341590612776, 133.07887874837027, 0.0)
	(175.0905902980713, 151.89392168322618, 189.52454704850962, 0.0)

Figura 3.5: Medias de color a diferentes iluminaciones y en diferentes regiones - base roja

color coincidían. Finalmente, se aplicó la operación lógica OR entre las máscaras obtenidas, resultando en una máscara compuesta por todos los píxeles que coincidían con todas las variaciones de iluminación considerados.

En la figura 3.7 se pueden observar las distintas máscaras correspondientes a las medias de color mostradas en la figura 3.5. La última máscara, a la derecha de la imagen, corresponde a la combinación de todas las anteriores. En esta figura se observa el comportamiento del algoritmo con una imagen de la pieza aislada, mientras que en la figura 3.8 se puede apreciar el efecto del algoritmo en una imagen de la pieza dentro del espacio de trabajo estándar que el robot de servicio habría de encontrar en condiciones reales.

En este proceso, además, se agregan las operaciones *erosión* y *dilatación*, la primera utilizada para reducir el ruido obtenido del filtrado por elementos en el espacio de trabajo que estén dentro del rango de color solicitado. Como se puede observar, existen grupos de píxeles en la imagen que desaparecen con la operación erosión al ser demasiado pequeños para considerarse como un elemento de



Figura 3.6: Pieza roja - componentes a) H b) S c) V



Figura 3.7: Pieza roja

interés. Finalmente se aplica la operación dilatación para recuperar los píxeles de los bordes del objeto que se hayan reducido durante la erosión.

### 3.3. Localización de objetos

Una vez obtenida la máscara que contiene todos los píxeles que coinciden con el modelo del objeto de interés se realiza la localización del objeto en el espacio tridimensional. Ya que se ha aislado la pieza en la imagen se obtiene el centroide geométrico del conjunto de píxeles en blanco, representando el centro de la pieza.

#### 3.3.1. Nubes de puntos

Para la obtención de la nube de puntos se utilizó el sensor de profundidad incluido en el dispositivo Kinect, este sensor emite un pulso de luz infrarroja y mide el tiempo que toma el pulso desde que es emitido hasta que regresa al sensor después de haber rebotado en una superficie, con este tiempo el dispositivo calcula la distancia estimada desde el sensor hasta el objeto, de acuerdo con la información del fabricante, el sensor utilizado para este proyecto cuenta con una resolución de  $640 \times 480$  píxeles Davison, 2012. Este conjunto de medidas de profundidad se asocian con su posición respecto al sensor, dando lugar a un *mapa*

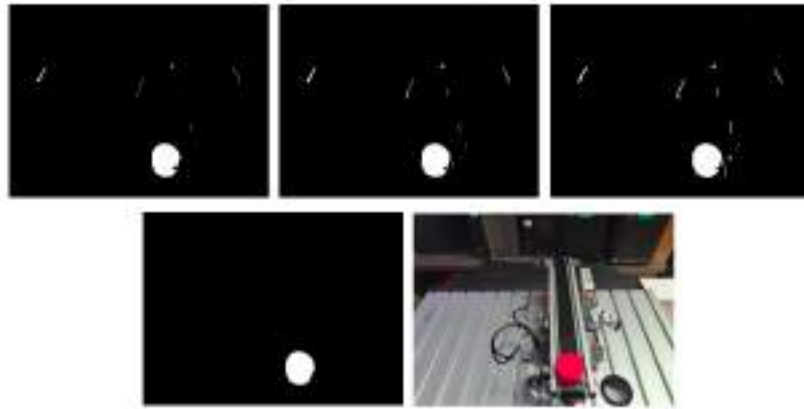


Figura 3.8: Espacio de trabajo

*de profundidades*. Para la estimación de la posición del objeto en el espacio, se realiza la comparación entre la imagen obtenida por la cámara RGB y el mapa de profundidades, asociando el centroide obtenido del objeto de interés con la información de la distancia calculada por el sensor de profundidad, resultando en la posición de la pieza en coordenadas relativas al dispositivo Kinect.

## Capítulo 4

# Planeación de movimientos

Una de las capacidades base que se busca que tenga el robot al deber colaborar con otros equipos mecánicos y estaciones de trabajo donde los elementos cuentan con posiciones específicas es la posibilidad de moverse evitando colisiones con dichos elementos y al mismo tiempo, que el efector final logre obtener las posiciones requeridas para llevar a cabo la actividad que se le solicita. Para esto es necesario realizar el estudio de la construcción mecánica del robot, de tal forma de se calculen las fuerzas que se deben aplicar en cada actuador perteneciente a la estructura del manipulador, mediante los análisis **cinemáticos y dinámicos** del robot.

Esta definición de la posición del robot en función del tiempo se conoce como **trayectoria** (2017). En Robótica, cada punto en el espacio de trabajo corresponde a una configuración única de los actuadores del robot y cada una de estas configuraciones se puede representar como un punto en un espacio conformado por el conjunto de configuraciones posibles de acuerdo con la configuración física del robot. De esta manera, la configuración de un brazo robótico con  $n$  articulaciones se puede representar como una lista de  $n$  posiciones articulares:  $q = (\theta_0, \dots, \theta_n)$ . Siguiendo las condiciones anteriores, los autores definen el *espacio de configuraciones libre* como aquel que contiene todas las configuraciones del robot en las que no colisiona con ningún obstáculo ni excede los límites de sus articulaciones (2017).

Dentro del trabajo realizado se busca que el robot sea capaz de dirigir su efector final a la coordenada en que se estima se encuentre la pieza de interés. Una vez que se ha conectado este objetivo específico con los conceptos teóricos anteriormente mencionados, es posible remarcar el uso de la **Cinemática directa**, la cual se encarga del cálculo de la posición y orientación de un sistema coordinado asociado al efector final del robot desde las coordenadas  $\theta$  de sus articulaciones (Lynch y Park, 2017).

La implementación de este análisis se realizó utilizando la herramienta MoveIt, que es posible utilizar dentro del entorno de ROS. esta herramienta lleva a cabo los cálculos necesarios para obtener los movimientos necesarios en la configuración del manipulador para alcanzar una pose específica.

---

## 4.1. Movimiento de cuerpo rígido

### 4.1.1. Posición y Orientación

De acuerdo con Asada y Slotine, 1986, la estructura formada por un manipulador puede ser modelada por medio de cuerpos rígidos. Adicionalmente, Gross, 2013, define el cuerpo rígido como aquel que no se deforma bajo la influencia de fuerzas, es decir, que las distancias entre diferentes puntos del cuerpo se mantienen constantes.

Los autores también comentan que para describir completamente la localización de un cuerpo rígido dentro del espacio es necesario conocer su posición y su orientación. Para conocer la posición, es necesario que exista un punto de referencia, fijo, y un sistema de coordenadas asociado al mismo, a partir del cual se tomarán las distancias que describen la posición relativa del punto que representa el punto de interés con respecto a la referencia. Convencionalmente, el sistema de referencia utilizado es aquel construido con los vectores unitarios  $(i, j, k)$ , estos vectores dictan la dirección de los ejes  $(X, Y, Z)$  con los cuales se define un espacio tridimensional. De esta forma, se utilizan las distancias sobre estos ejes que definen la relación entre dos puntos del espacio, en este caso, el cuerpo rígido y el origen del sistema.

Además de las distancias, es posible establecer las relaciones angulares que definen la relación entre el origen del sistema y el cuerpo de estudio, siendo posible una inclinación en cualquiera de estos ejes de forma independiente o bien, una combinación de rotaciones en todos ellos.

La posición del cuerpo rígido puede representarse matemáticamente utilizando un vector compuesto por las tres distancias que definen su relación con el sistema de referencia de la siguiente manera:  $P_0 = (x_0, y_0, z_0)$ .

De forma similar, Asada y Slotine, 1986, comentan que para representar la orientación de un cuerpo rígido, se definen tres ejes coordenados esta vez asociados al cuerpo rígido, en la figura 4.1 el origen de este segundo sistema coordenado se encuentra señalado como  $O_1$ , y los ejes como  $(X_1, Y_1, Z_1)$ . Este nuevo sistema se encuentra conectado al cuerpo rígido y se mueve junto a él.

Para el estudio de la orientación entre los sistemas coordenados establecidos es pertinente mencionar el concepto de las rotaciones básicas. Olguín Díaz, 2019, explica que una simplificación útil para este planteamiento es considerar un movimiento rotatorio sin traslación. Si se considera únicamente la rotación sobre uno de los ejes Cartesianos se conoce como una Rotación Básica. El autor añade en su explicación las matrices asociadas a las rotaciones sobre los ejes. Utilizando estas matrices (mostradas en las ecuaciones 4.1 a 4.3) es posible encontrar la posición de un punto existente en el sistema rotado, mediante la transformación que resulta en sus coordenadas referentes al sistema original.

$$R_{x,\phi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos_\phi & -\sin_\phi \\ 0 & \sin_\phi & \cos_\phi \end{bmatrix} \quad (4.1)$$

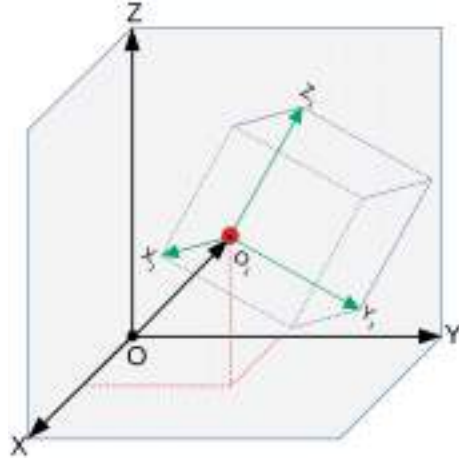


Figura 4.1: Posición y orientación de un cuerpo rígido

$$R_{y,\theta} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \quad (4.2)$$

$$R_{z,\psi} = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (4.3)$$

Como parte del análisis del movimiento del robot comúnmente se encuentran rotaciones que no coinciden con el movimiento sobre solo uno de los ejes, sino que se tratan de movimientos complejos que resultan de combinaciones de otros movimientos. Olguín Díaz, 2019, indica que toda rotación se obtiene con rotaciones básicas sucesivas. Sin embargo, considera importante apuntar que es indispensable conocer el orden en el cual se realizan las rotaciones, dado que rotaciones sobre el sistema original (extrínsecas) dan resultados distintos que las rotaciones realizadas sobre el sistema coordenado asociado al cuerpo rígido (intrínsecas), este planteamiento permite la existencia de diferentes combinaciones de secuencias de rotación que llevan al mismo resultado. Para esto, Olguín menciona el estudio de **Rotaciones Compuestas**, con el cual es posible analizar las interacciones entre las diferentes matrices de rotación y su efecto en el objeto. Dentro de la notación utilizada para este tema es común encontrar lo siguiente:  $R_a^b$  y  $p^{(a)}$ , la primera expresión se refiere a la rotación existente entre el sistema  $b$  usando como referencia al sistema  $a$  y la segunda al punto  $p$  usando como origen al sistema  $a$ . Utilizando esta notación se puede expresar la relación entre el punto  $y$  y las matrices de rotación.



---

Suponiendo la existencia de los sistemas coordenados  $\Sigma_0$ ,  $\Sigma_1$  y  $\Sigma_2$ , la orientación de un punto  $p$  se puede expresar de la siguiente forma:

$$\begin{aligned} p^{(1)} &= R_1^2 p^{(2)} \\ p^{(0)} &= R_0^1 p^{(1)} = R_0^1 R_1^2 p^{(2)} = R_0^2 p^{(2)} \end{aligned} \quad (4.4)$$

En las ecuaciones anteriores se puede observar una de las propiedades del análisis de las matrices de rotación. En este caso, la matriz de rotación compuesta  $R_0^2$  se puede obtener mediante una multiplicación matricial para rotaciones intrínsecas:

$$R_0^2 = R_0^1 R_1^2 \quad (4.5)$$

### ***Ángulos de Euler***

Una vez que se ha establecido como base el análisis de las rotaciones, Olguín indica que: *'toda combinación de rotaciones básicas sobre los ejes principales tiene tres ángulos **independientes** conocidos como los Ángulos de Euler'* (2019), profundiza diciendo que las rotaciones básicas pueden realizarse ya sea todas en el sistema coordenado base (parametrización extrínseca), todas en el sistema coordenado actual (parametrización intrínseca), o bien usando una combinación de las posibilidades anteriores. Esta última opción conlleva procedimientos muy engorrosos, por lo que el autor no recomienda su uso. Olguín indica también que en la práctica hay 24 formas distintas de representar una matriz de rotación utilizando tres ángulos de Euler independientes, conformadas de 12 combinaciones extrínsecas para rotaciones basadas en un sistema coordenado fijo y 12 combinaciones intrínsecas para rotaciones basadas en el sistema coordenado actual.

### ***Cuaterniones***

La representación de la orientación por medio de cuaterniones fue planteada inicialmente por Sir William Rowan Hamilton Hamilton, 1844, y se utiliza en robótica gracias a que no se obtienen singularidades, como sí sucede con la representación con Ángulos de Euler o Ángulos fijos. También se considera imprescindible mencionarlos dado que el middleware ROS, utilizado durante la implementación de este proyecto recurre a ellos para las transformaciones homogéneas que realiza. De acuerdo con Siciliano y Khatib, 2016, un cuaternion  $\epsilon$  se define de la siguiente forma:

$$\epsilon = \epsilon_0 + \epsilon_1 i + \epsilon_2 j + \epsilon_3 k \quad (4.6)$$

En la ecuación anterior, los componentes  $\epsilon_0$ ,  $\epsilon_1$ ,  $\epsilon_2$  y  $\epsilon_3$ , son escalares, conocidos en ocasiones como parámetros de Euler.

---

Por su parte,  $i$ ,  $j$  y  $k$  son operadores y se definen para satisfacer las siguientes reglas de correspondencia:

$$\begin{aligned} ii &= jj = kk = -1 \\ ij &= k, \quad jk = i, \quad ki = j \\ ji &= -k, \quad kj = -i, \quad ik = -j \end{aligned} \quad (4.7)$$

Continuando con la descripción de los editores, mencionan los elementos matemáticos necesarios para el manejo algebraico de los cuaterniones:

**Suma de cuaterniones:** se obtiene al sumar por separado sus respectivos componentes:

$$\begin{aligned} \epsilon_A &= \epsilon_0 + \epsilon_1 i + \epsilon_2 j + \epsilon_3 k \\ \epsilon_B &= \epsilon_4 + \epsilon_5 i + \epsilon_6 j + \epsilon_7 k \\ \epsilon_C &= \epsilon_A + \epsilon_B \\ \epsilon_C &= (\epsilon_0 + \epsilon_4) + (\epsilon_1 + \epsilon_5)i + (\epsilon_2 + \epsilon_6)j + (\epsilon_3 + \epsilon_7)k \end{aligned} \quad (4.8)$$

**Elemento nulo para adición:**

$$0 = 0 + 0i + 0j + 0k \quad (4.9)$$

**Elemento nulo para multiplicación:**

$$I = 1 + 0i + 0j + 0k \quad (4.10)$$

Comúnmente, el elemento  $\epsilon_0$  es conocido como la parte escalas del cuaternion, mientras que los elementos  $(\epsilon_1, \epsilon_2, \epsilon_3)$  se conoce como la parte vectorial.

## 4.2. Transformaciones homogéneas

El uso de Matrices de Transformaciones Homogéneas en el análisis del movimiento, tanto de cuerpos rígidos como su aplicación en robótica, permite el manejo matemático de la orientación y la posición del cuerpo rígido en un espacio tridimensional, estas son matrices de  $4 \times 4$  y cumplen con la siguiente forma:

$$T = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_1 \\ r_{21} & r_{22} & r_{23} & p_2 \\ r_{31} & r_{32} & r_{33} & p_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.11)$$

---

Lynch y Park, 2017, listan a manera de Proposiciones algunas de las propiedades de las matrices de transformación homogéneas, como son:

- La inversa de una matriz de Transformación  $T \in SE(3)$  es también una matriz de transformación y cumple con la siguiente forma:

$$T^{-1} = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} R^T & -R^T p \\ 0 & 1 \end{bmatrix} \quad (4.12)$$

- El producto de dos matrices de transformación es también una matriz de transformación.
- La multiplicación de matrices de transformación cumple con la propiedad asociativa:  $(T_1 T_2) T_3 = T_1 (T_2 T_3)$ , pero no con la propiedad conmutativa:  $T_1 T_2 \neq T_2 T_1$

Siguiendo la descripción de los autores, es útil calcular el valor de  $Rx + p$ , siendo que  $x \in R^3$  y  $(R, p)$  representa la matriz  $T$ . Para hacer compatible el vector  $x$  con las dimensiones de la matriz  $T$ , se concatena un 1 al vector  $x$ , con lo cual es posible realizar el cálculo mediante una multiplicación de matrices 4.13.

$$T \begin{bmatrix} x \\ 1 \end{bmatrix} = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix} = \begin{bmatrix} Rx + p \\ 1 \end{bmatrix} \quad (4.13)$$

De esta operación se obtiene el vector  $\begin{bmatrix} x^T & 1 \end{bmatrix}^T$  se conoce como la representación de  $x$  en coordenadas homogéneas. En el contexto en que no encontramos, estas matrices son comúnmente utilizadas para cambiar los sistemas coordenados que se utilizan como referencia.

De forma similar a las matrices de Rotación mencionadas anteriormente, la multiplicación de estas matrices permite obtener relaciones entre los distintos sistemas de referencia:

$$T_{ab} T_{bc} = T_{a\hat{b}} T_{\hat{b}c} = T_{ac} \quad (4.14)$$

$$T_{ab} v_b = T_{a\hat{b}} v_{\hat{b}} = v_a \quad (4.15)$$

### 4.3. Descripción cinemática del robot

La estructura que compone al robot para la implementación de este proyecto se muestra en la figura 4.2, donde se observan dos perspectivas del robot utilizando la interfaz gráfica Rviz. Para este despliegue, se cuenta con los modelos de los sensores Kinect, Hokuyo, la base del robot y la plataforma. Los elementos adicionales se representan mediante figuras tridimensionales, cuyas medidas corresponden con las de los elementos reales del robot.

Todas las articulaciones del robot con la configuración utilizada para este proyecto son rotacionales.



Figura 4.2: Árbol cinemático del robot

#### 4.4. Manipulación de objetos

Como se menciona en el capítulo anterior, se estima la posición de la pieza de interés utilizando la información obtenida del procesamiento que se realiza a la imagen en conjunto con el análisis de la nube de puntos. Sin embargo, la posición calculada tiene como sistema de referencia aquel asociado a la cámara del sensor Kinect, por lo que es necesario realizar la transformación de esas coordenadas a unas asociadas a la base del manipulador del robot. Con este fin se utilizan las herramientas mencionadas anteriormente. Para esta implementación se utilizó la paquetería **tf** (2019) incluida en la infraestructura del middleware ROS, que facilita estos cálculos en tiempo real, así como su visualización durante el desarrollo y la ejecución. Adicionalmente, el cálculo de la trayectoria final que ha de seguir el manipulador para que el efector final llegue a su destino se realizó por medio de la paquetería MoveIt.

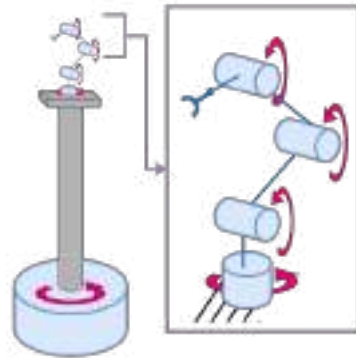


Figura 4.3: Descripción cinemática

---

---

## Capítulo 5

# Planeación de acciones

Un aspecto importante del comportamiento del robot en el contexto que se plantea es que le sea posible desarrollar una serie de actividades que en conjunto lleven a un resultado final. Con este objetivo se desarrollaron varias máquinas de estados en los que la secuencia de acciones, así como las condiciones, variaban dependiendo del objetivo final. Dentro de los escenarios considerados se requiere lograr tres tareas específicas, en las que progresivamente se añadían capacidades al desempeño del robot.

Como se mencionó en secciones anteriores, para el planteamiento de las máquinas de estado se necesita determinar las entradas, salidas, estados, condiciones de transformación y las reglas de correspondencia que permiten el flujo de la información dentro de la secuencia de acciones.

En este caso, las entradas del sistema se refieren a la información que el robot es capaz de recopilar tanto de las condiciones de su ambiente como de su estado interno. Haciendo uso de los sensores con que cuenta el robot puede determinar su posición respecto al mapa, y las posiciones también de los elementos en el entorno con los que debe interactuar. Además, la información para el desarrollo de su actividad es previamente ingresada en una base de conocimiento de la cual el robot debe poder extraer la información y saber en qué parte de la secuencia de acciones se encuentra.

Es deseable que el robot cuente con estrategias que le permitan verificar si el estado en que el robot *crea* que está es correcto y también comprobar que las acciones que se han registrado como realizadas se han completado correctamente.

### 5.1. Actividades

Como parte de las secciones que componen a la competencia *RoboCup Logistics* se creó una categoría en la que se evalúan las diferentes habilidades que es indispensable que el robot tenga: **navegación, exploración y manipulación**. De esta forma se facilita la integración y la evaluación del desempeño del

---

robot en un entorno controlado para cada tarea, a continuación se incluye la descripción de cada una obtenida del reglamento de la competencia (2022).

#### 5.1.1. Navegación

Para la evaluación de la Navegación se solicita que el robot *visite* 12 puntos aleatorios dentro del espacio y permanecer en esa posición por al menos cinco segundos consecutivos, utilizando el menor tiempo posible y evadiendo obstáculos estáticos durante su desplazamiento. Esta etapa podría considerarse la menos compleja dentro de las tres planteadas, pero es una actividad primordial dentro del desempeño del robot, dado que de no ser capaz de realizarla, no sería posible completar las siguientes.

#### 5.1.2. Exploración

Durante la etapa de exploración se busca que el robot lleve a cabo una navegación autónoma, mientras realiza el procesamiento de la imagen capturada por su cámara para el reconocimiento de las estaciones de trabajo. Dicho reconocimiento se hace utilizando los códigos ARUCO que se encuentran en la base de las estaciones de trabajo. Además, el robot debe identificar el tipo de estación, la posición y orientación en que se encuentra cada estación y reportar la información.

#### 5.1.3. Manipulación

En el caso de la manipulación, la posición inicial del robot se encuentra frente a una de las estaciones de trabajo. En la salida de la banda transportadora de la estación se encuentra una pieza que debe ser tomada por el manipulador. El robot debe trasladarse al otro lado de la estación y colocar la pieza en la entrada de la banda transportadora. Este proceso se repite tres veces y dependiendo del nivel de dificultad que elija, se puede trabajar con una o más estaciones en las cuales se debe repetir el mismo procedimiento.

### 5.2. Máquinas de estados

Una vez conocidos los requerimientos establecidos en el reglamento para considerar como exitosas estas tareas, se explicará a continuación cómo se realizó la implementación de las tareas haciendo uso de Máquinas de Estados Finitos.

#### 5.2.1. Estados

Dentro de los Estados que se ejecutan dentro de los algoritmos se lleva a cabo las acciones por medio de las cuales se modifican las entradas y salidas del sistema, haciendo uso de las funciones que provocan cambios en el comportamiento del robot, de forma que se cumplan las condiciones de transición entre estados.

---

### 5.2.2. Navegación

La secuencia de Estados comienza con una etapa en la que se espera que el robot reciba el mensaje del sistema centralizado en el que se le indican las zonas del mapa que deben ser visitadas. Dado que se sabe el número específico de zonas necesarias para considerar la tarea como exitosa, el robot permanece en espera hasta que se completa el mensaje. Debido a que la información de las zonas objetivo se recibe como una cadena de texto, es necesario procesarla y se colocan los nombres en un arreglo desde el cual es más sencillo acceder a la información, además de conocer cuáles de los puntos ya han sido visitados y cuáles aún están pendientes. La condición de transición para este estado depende de la bandera **flag\_zones** mediante la cual se verifica que todas las zonas solicitadas se hayan visitado exitosamente, de lo contrario se sigue intentando, este proceso se repite hasta que se haya completado las zonas o bien se termine el tiempo permitido. Con el arreglo de las zonas que se ha de visitar y sabiendo previamente las posiciones de las zonas en el mapa, se utiliza el algoritmo **Nearest Neighbour**, mediante el cual se calcula una ruta a seguir por el robot, buscando hacer el recorrido en el menor tiempo posible. Cuando se cuenta con la ruta a seguir se utiliza la función **navigate\_to\_location**, con la cual el robot se desplaza hasta el centro de la zona solicitada. Se agrega un tiempo de espera para cumplir con los cinco segundos que se requieren para considerar la visita como exitosa y una vez pasado ese tiempo, se continúa con la ruta. Haciendo uso de la bandera **cont** se verifica el número de zonas visitadas y, en caso de haber completado las 12 zonas, se concluye la ejecución de la máquina de estados.

### 5.2.3. Exploración

En el caso de la etapa de exploración, se consideró establecer una ruta a seguir utilizando puntos del mapa desde los cuales realizar el escaneo en busca de las estaciones de trabajo.

La ruta designada se compone por ocho *Puntos de Inspección Principales* y 11 *Puntos de Inspección Intermedios*. En cada uno de los puntos principales el robot hace una breve pausa y realiza una secuencia de movimientos de base que cubren 90 grados, en los cuáles solo varía la dirección del giro. Por su parte, en los puntos intermedios, se realiza un giro de 360 dando oportunidad a que la cámara encuentre los códigos de las estaciones y obtenga la información sobre ellos, de ser visibles desde su posición, y los reporte. De acuerdo con las reglas de la competencia, el robot debe empezar en un punto específico, junto al borde inferior del mapa, a la izquierda o derecha dependiendo del equipo al que pertenezca. Por motivos de simplicidad, en la Figura 5.1 se muestra el proceso si el robot pertenece al equipo magenta. Una vez que el robot ha salido de la zona designada se establecen como objetivo las coordenadas de la zona que se encuentra en la esquina inferior izquierda del mapa, señalado en la Figura como el *Punto de Inspección Principal 1*. Debido a que el mapa corresponde a un rectángulo de  $14 \times 8$ [m], y a que las zonas de descanso de los robots se encuentran en la arista inferior del mapa, se decidió dividirlo en dos regiones principales,



cada una de  $7 \times 7$ , dentro de las cuales se realiza la trayectoria de exploración. Como se puede observar en la ilustración, en las secciones del recorrido que corresponden a las diagonales de cada mitad del mapa, se definieron 2 *Puntos de Inspección Intermedios*, mientras que en las aristas se utiliza solo uno.

De acuerdo con el reglamento, el posicionamiento de las estaciones en el mapa debe ser simétrico entre cada una de las mitades del mapa, por lo cual teniendo la información sobre las estaciones contenidas en una mitad es posible inferir las de la otra. Al llegar al *Punto de Inspección Principal 4* se verifica si se han encontrado todas las estaciones necesarias. Sin embargo, en caso de que al terminar la ruta de exploración de la mitad izquierda no se hayan localizado todas las estaciones, se continúa con la secuencia en la mitad derecha en busca de la información restante.

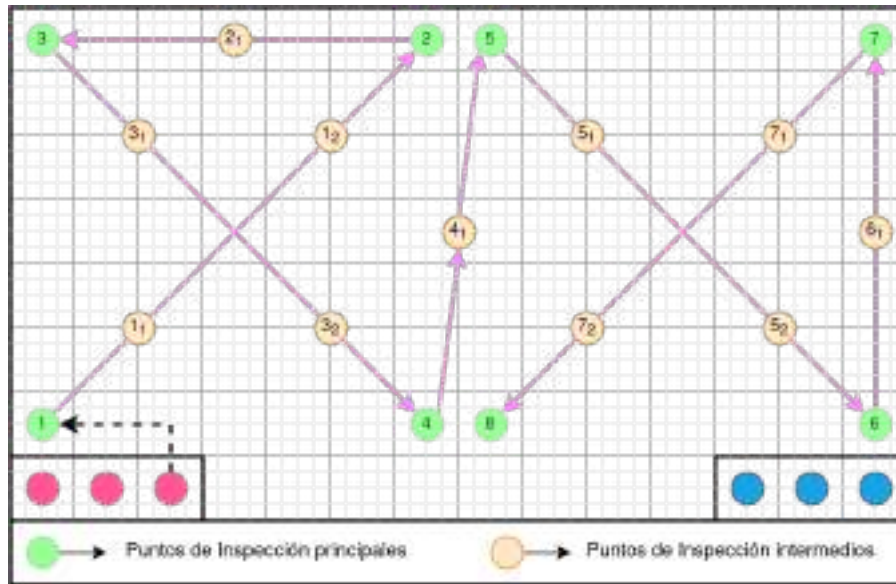


Figura 5.1: Exploración

Dentro de la Figura solo se muestran las rutas a seguir y los puntos de inspección que las componen, pero es importante recordar que las Estaciones de Trabajo que el robot está buscando representan también obstáculos estáticos dentro del entorno y que podrían interferir con la navegación del robot. Se sabe que el sistema de navegación que se está utilizando es un sistema confiable y robusto mediante el cual es posible realizar una navegación con evasión de obstáculos. No obstante, este es uno de los motivos por lo cual se consideró pertinente la definición de los puntos intermedios. Ya que si la navegación se ve interrumpida por algún obstáculo y se excede el tiempo límite definido para dicha ruta, el sistema de navegación descarta esa coordenada y se define como objetivo el siguiente punto de inspección en la ruta. Adicionalmente, se definió que los puntos de inspección intermedios comprendieran perspectivas redundan-

---

tes de algunas zonas, por lo que si los códigos en las paredes de las estaciones no son visibles desde un punto, es posible que lo sea desde otro.

#### 5.2.4. Manipulación

Dentro de esta tarea, se aplican los conceptos mencionados en capítulos anteriores referentes al procesamiento de la imagen captada por la cámara RGBD. Es en esta parte de la competencia en que se necesita identificar y localizar las piezas de interés dentro del espacio de trabajo del robot, tomarlas con el manipulador y transportarlas a otra ubicación determinada. Para ello se solicita que el robot inicie el proceso frente a la MPS. Dadas las restricciones del dispositivo Kinect es necesario que el robot se encuentre a cierta distancia mínima del objeto de interés para obtener mediciones confiables del sensor de profundidad, por lo que de la posición inicial el robot debe retroceder un metro más. Es desde esta posición donde se ejecuta una función en la cual se utiliza la imagen RGB y se detectan las líneas contenidas en la Estación, excluyendo aquellas líneas que se sabe no pertenecen a la estación y aquellas que excedan un umbral del ángulo de inclinación, esta discriminación ayuda a disminuir el ruido de líneas detectadas por la cámara. Usando movimientos angulares de la base, se reduce el error entre el ángulo que se detecta del borde de la estación y una referencia horizontal definida empíricamente. Una vez que el robot está alineado se procede a la búsqueda del objeto sobre la superficie de la MPS.

Como se mencionó en capítulos anteriores, para la localización de las piezas se utilizó un algoritmo de segmentación por color de la imagen, una vez localizada, se le asigna una coordenada en el mapa relativa al origen del mismo. Dado que se cuenta ya con la información necesaria de la nube de puntos, y que la distancia que es posible alcanzar con el manipulador es limitada, el robot avanza la distancia que retrocedió anteriormente y se obtiene la transformación homogénea de la coordenada de la pieza con respecto a la posición actual del origen del manipulador del robot. Esta información es enviada al controlador del manipulador para calcular la trayectoria del mismo hacia la pieza. Dentro de la función que realiza esta acción, se posiciona la pinza frente a la coordenada solicitada, un par de centímetros antes, se abre la pinza, se realiza el desplazamiento hacia el frente para tomarla y finalmente se cierra la pinza.

Una vez que la pieza se encuentra en el manipulador, el robot puede retroceder nuevamente y realizar la navegación hasta el punto de entrega (en el extremo opuesto de la MPS). Donde una vez más se utiliza el algoritmo de alineación y procesamiento de imagen para depositar la pieza sobre la banda transportadora. Suponiendo que se haya elegido un nivel de complejidad que involucre el uso de otra MPS, el robot realiza ahora la navegación y repite el proceso.

#### 5.2.5. Entradas y salidas

Un elemento indispensable del comportamiento del robot depende de la información que recibe sobre su estado y el de su entorno.

---

**Localización** - El robot debe *saber* su posición relativa al espacio en que se encuentra, esto, cuando es posible, se logra con un mapa previo del entorno en que se planea que el robot desempeñe sus actividades, y el cambio en su posición dentro del mapa se monitorea y actualiza constantemente mediante una combinación entre la odometría del robot y el algoritmo de localización de Monte Carlo, utilizando las lecturas que recibe del sensor de distancia que se encuentra en la base del robot.

**Navegación** - El robot debe ser capaz de moverse dentro de su entorno, por lo que es necesaria la implementación de un algoritmo que le permita moverse de un punto a otro. Esta navegación se debe realizar sin colisionar con otros objetos en el espacio. En el sistema de navegación utilizado para este proyecto, desarrollado por el Laboratorio de Biorobótica de la Facultad e Ingeniería de la UNAM y probado previamente en la liga @Home (tanto OpenPlatform como D) de la competencia RoboCup, se utilizan las lecturas del sensor de distancia, en conjunto con el algoritmo A\* para realizar el cálculo de una ruta óptima entre dos puntos.

#### 5.2.6. Algoritmos de las Máquinas de Estados - Cartas ASM

A continuación se muestran los diagramas de flujo que representan el funcionamiento de las máquinas de estado utilizadas para cada actividad. Navegación [5.2](#), exploración [5.3](#), manipulación [5.4](#).

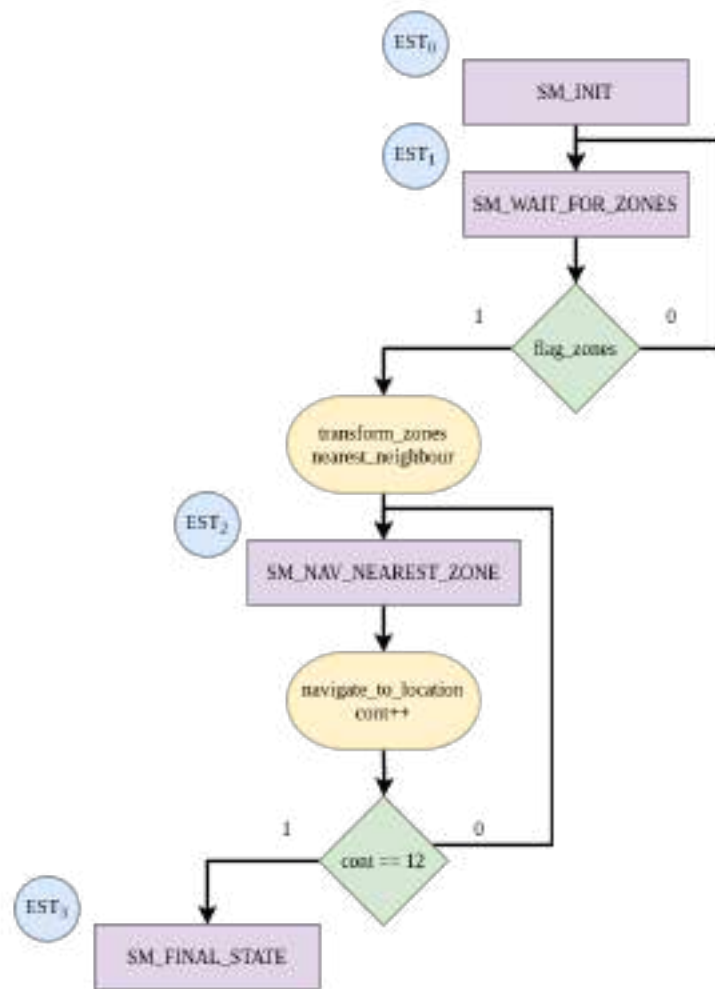


Figura 5.2: Diagrama de carta ASM -Navegación

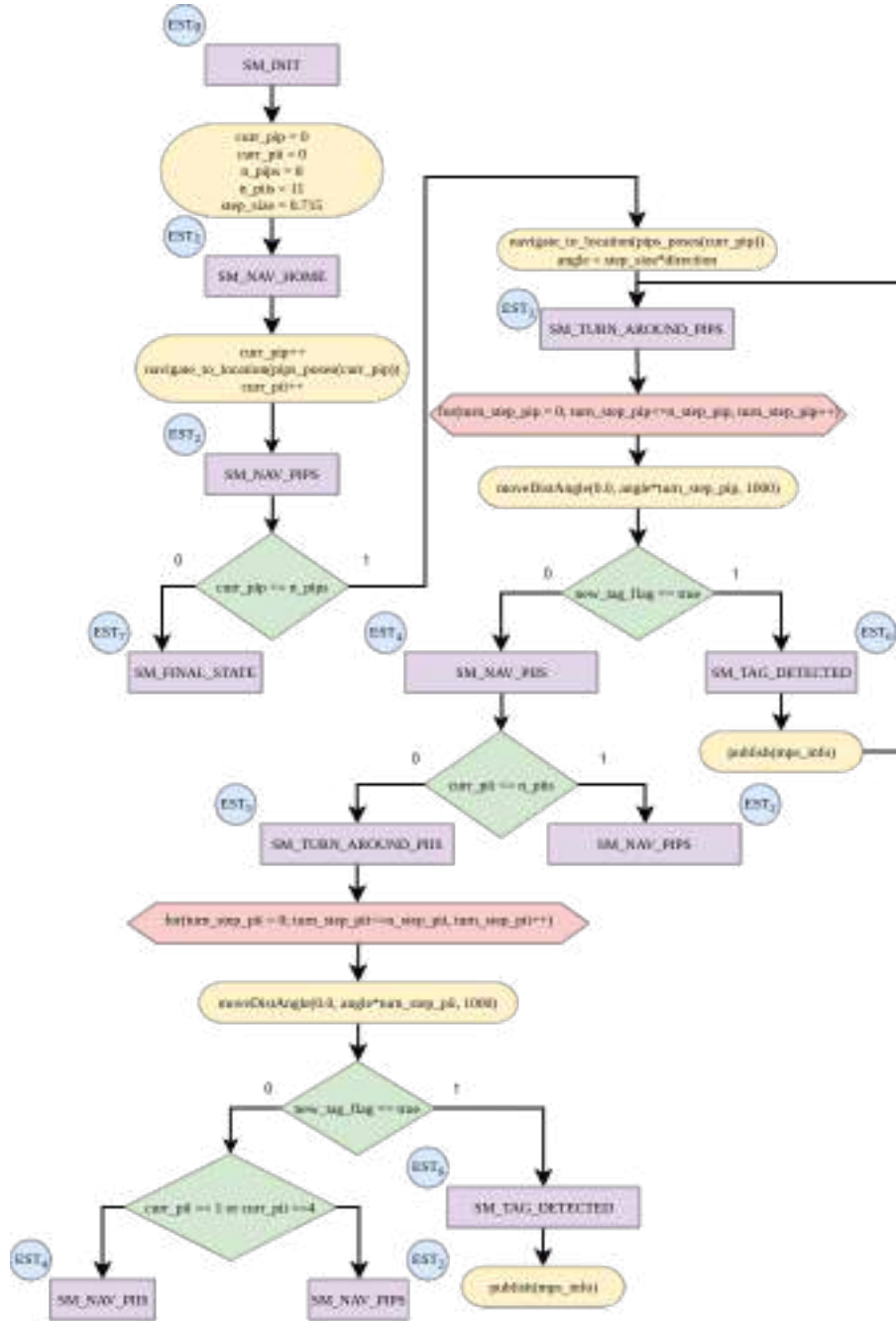


Figura 5.3: Diagrama de carta ASM - Exploración

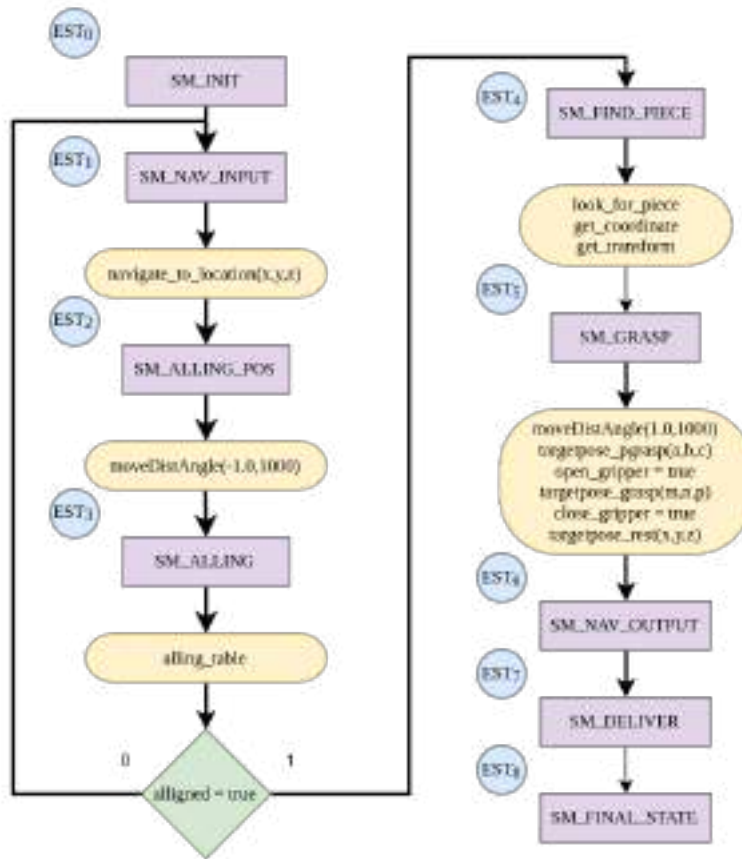


Figura 5.4: Diagrama de carta ASM - Manipulación

---

---

## Capítulo 6

# Resultados

### 6.1. Herramientas

Para lograr los objetivos planteados para el desarrollo de este proyecto se utilizaron varias herramientas de software y de hardware, las cuales hicieron posible la recopilación de los datos necesarios, así como la ejecución de las acciones planteadas como requisitos del proceso planteado. Se ha hecho suficiente hincapié en que el objetivo principal de este proyecto es el aprovechamiento de los datos recabados por una cámara RGBD. En este caso, el dispositivo utilizado para la obtención de la imagen y la nube de puntos es el **Kinect**, un dispositivo distribuido por la empresa **Microsoft**, que en 2010 fue lanzado al mercado como accesorio para la consola de videojuegos XBOX 360. Adicionalmente, como base del robot móvil se utilizó la plataforma de desarrollo **Robotino** perteneciente a la empresa FESTO.

#### 6.1.1. Herramientas de software

##### ROS

En 2018 Edgar Vázquez realiza una breve descripción del sistema ROS, en la cual comenta que el objetivo principal es dar soporte a la reutilización de código en la investigación y desarrollo de robótica, referenciando a Quigley, Morgan y col., [2009](#). Adicionalmente, Vázquez menciona que ROS ofrece las siguientes utilidades:

- Abstracción de hardware.
- Control de dispositivos de bajo nivel.
- Implementación de funcionalidades comunes entre dispositivos.
- Intercambio de mensajes entre procesos.
- Mantenimiento de paquetes.



---

Vázquez añade que durante el desarrollo del comportamiento de un robot es común considerar las funcionalidades a modo de módulos, en que cada uno tiene un objetivo específico. Apunta también que ROS colabora con el transporte de la información entre dichos módulos y menciona como una ventaja la existencia de la comunidad de desarrolladores que trabajan con ROS, haciendo posible el intercambio de información, obteniendo soporte y retroalimentación actualizados.

### Elementos

- **Nodos:** los nodos son los módulos que en conjunto forman la red de trabajo, y se encargan generalmente de las funcionalidades aisladas que se mencionaron anteriormente. Es decir, la activación de los actuadores, el procesamiento de imagen, el control de la interfaz del usuario, cada una de estas actividades se encontraría bajo el cargo de un nodo, independiente de las demás, pero que bajo la estructura de ROS les es posible intercambiar información.
- **Paquetes:** Vázquez comenta que dentro de la estructura de ROS, los paquetes pueden compararse con carpetas, las cuales tienen características estructurales definidas y en las cuales es posible contener a los nodos. Además, dentro del paquete se deben establecer las bibliotecas u otras paqueterías que requiera la correcta ejecución de los nodos, como pueden ser los tipos de datos personalizados, recursos ajenos a ROS o archivos de configuración adicionales.
- **Tópicos:** son el elemento que se utiliza para intercomunicar los nodos. Dentro de la implementación con ROS es posible *publicar* información dentro del entorno por medio de mensajes y haciendo uso de los tópicos, los mensajes pueden ser diversos tipos, dependiendo de los requerimientos de la implementación, adicionalmente, es posible crear nuevos tipos de mensaje de ser necesario (2023).
- **Servicios:** un método distinto a los tópicos mediante el cual también se puede compartir información son los servicios. La diferencia entre estas herramientas es que los servicios necesitan ser solicitados por otro agente, es decir, sigue un modelo petición-respuesta. Los autores comentan que si un proceso necesita realizarse solo en momentos específicos lo recomendable es implementarlo como un servicio. (2018)

### OpenCV

*The Open Source Computer Vision Library*, se trata de una biblioteca de acceso libre a través de la cual es posible utilizar cientos de algoritmos de procesamiento de imágenes y visión computacional.

Dentro de la información que ofrece el sitio web de la biblioteca, se menciona que cuenta con una estructura modular. Algunos de los módulos son:

- 
- *Core functionality*: un módulo para definir estructuras de datos básicas, como arreglos y matrices multidimensionales, así como funciones básicas necesarias para otros módulos.
  - *Image Processing*: Incluye filtros lineales y no lineales, transformaciones geométricas de imágenes, conversión entre espacios de color, obtención de histogramas, etc.
  - *Video Analysis*: Permite estimación de movimiento, sustracción de fondos, y algoritmos de seguimiento de objetos.
  - *Camera Calibration and 3D Reconstruction*: algoritmos geométricos básicos de vistas múltiples, calibración de cámaras sencillas y stereo, estimación de la pose de objetos, elementos de reconstrucción 3D.
  - *2D Features Framework*: detector de características, descriptores y emparejamiento de descriptores.
  - *Object Detection*: detección de objetos e instancias pertenecientes a clases predefinidas.
  - *Video I/O*: interfaz fácil de usar para captura de videos, además de compresores y decompresores de archivos multimedia. (2023)

Arévalo y col., 2002, describen la historia del proyecto OpenCV, mencionando que dio inicio en el año 2000, cuando la empresa Intel y un grupo de investigadores empezaron a trabajar en el desarrollo de una biblioteca de funciones desarrolladas en lenguaje C y especializada en Visión Computacional, con el objetivo de facilitar el uso de estas herramientas para docentes e investigadores, utilizando, además una licencia de Software Libre. Actualmente es posible utilizar esta biblioteca usando diferentes lenguajes de programación y cuenta con integraciones con otros sistemas como ROS y Matlab ®.

## MoveIT

Esta paquetería fue utilizada para la generación de las trayectorias de movimiento que realizaba el manipulador, es decir se realizan dentro de esta paquetería el análisis cinemático y dinámico del manipulador del robot. Para su configuración, MoveIt utiliza elementos dentro de la estructura de ROS para obtener la siguiente información:

- **URDF** (Universal Robot Description Format) - utiliza el parámetro de ROS llamado *robot\_description* para obtener la descripción de la configuración del Robot.
- **SRDF** (Semantic Robot Description Format) - utiliza el parámetro de ROS llamado *robot\_description\_semantic*, esta información se complementa con la recopilada en el URDF, especificando grupos de articulaciones, configuraciones predeterminadas del robot, información adicional sobre

---

colisiones o transformaciones adicionales que sean de interés para el movimiento del robot. (2023)

Dentro de la documentación de la paquetería se sugiere que se generen estos archivos haciendo uso del asistente propio de la misma. **MoveIt configuration** buscará dentro del entorno de ROS otras configuraciones pertinentes, como los límites de las articulaciones, cinemática, planeación de movimientos, percepción, etc. Estos archivos de configuración se encuentran dentro de la carpeta *config* y se generan automáticamente al ejecutar el asistente de configuración de la paquetería. **Robot Interface** MoveIt se comunica con el robot a través de ROS, pudiendo leer el estado actual de las articulaciones, obtener las nubes de puntos u otra información recabada por sensores, al mismo tiempo que se comunica con los controladores del robot («Concepts — MoveIt», 2023).

- Joint state - La paquetería escucha los tópicos que publican la información en tiempo real de las articulaciones del robot.
- Transform state - La paquetería utiliza la biblioteca TF para monitorear la información de las transformaciones presentes en el robot y su entorno.
- Controller Interface - La paquetería se comunica con el controlador para el seguimiento de la trayectoria, es necesario un servidor en el robot que realice las acciones solicitadas, ya que esta interfaz solo realiza la solicitud a manera de cliente.

### RobotinoView

Robotino View es un entorno de programación gráfico que es posible utilizar directamente con Robotino y que, de hecho, viene preinstalado con el robot desde la configuración de fábrica. Dentro de este ambiente es posible crear y ejecutar programas de control para el robot. De acuerdo con la información que ofrece FESTO respecto a este entorno, se tienen las siguientes funcionalidades:

- Programas secuenciales son mostrados como Grcfet.
- Control simultáneo de varios Robotino.
- Representación gráfica de componentes gráficos como bloques de función: motores, puertos de entradas y salidas (I/Os), sensores, cámaras, odometría, pinzas, manipuladores, lectura de encoders y salida de voltaje.
- Bloques de función para procesamiento de imágenes: reconocimiento de líneas, búsqueda de rangos de color, reconocimiento de marcadores.
- Bloques de función para navegación: conducción de posición, conductor de ruta, evasión de obstáculos e integración en fábrica.
- Bloques de función para intercambio de datos: UDP, TCP/IP client / server, OPC.

- 
- Descarga y ejecución de programas en RobotinoView, directamente en Robotino.
  - Creación e integración de funciones propias en C++.
  - Interfaz de usuario y manuales en múltiples idiomas. ([2023](#))

### 6.1.2. Herramientas de hardware

#### Robotino FESTO

De acuerdo con la descripción ofrecida por la empresa FESTO en su sitio web, la plataforma educativa Robotino, tiene como objetivo apoyar las labores de investigación y educación mediante un robot móvil al alcance de instituciones educativas, promoviendo el desarrollo en la robótica móvil y robótica de servicio. A continuación una lista de las especificaciones técnicas del robot:

##### Sistema robótico móvil

- Diámetro: 450 mm.
- Altura incluida la carcasa de la unidad de control: 290 mm.
- Peso en vacío sin acumuladores ni torre: 20 kg.
- Carga: máximo 30 kg.

##### Chasis

- Material: Chasis redondo de acero inoxidable.
- Protección anticolidión: banda de protección de goma con sensor de protección anticolidión integrado.
- Sensores: 9 sensores de distancia por infrarrojos con un rango de medición de 4 - 40 cm.
- Sensores: 1 sensor inductivo analógico, 2 sensores ópticos digitales.
- Sensores: 1 giroscopio interno de 3 ejes con sensor de aceleración.

---

### **Actuador**

- Ruedas: 3 ruedas omnidireccionales de 120 mm de diámetro.
- Motores: 3 motores DC, máx. 3600 rpm, con transmisores giratorios y reductor.
- Relación de transmisión: 32:1.

### **Interfaz I/O**

- Entradas/salidas digitales: 8 entradas/salidas digitales con 24 V, protegidas contra sobrecarga y cortocircuito.
- Entradas analógicas: 8 entradas analógicas de 0 V a 10 V a una frecuencia de exploración de 50 Hz.
- Salidas de potencia: dos salidas de relé.

### **Fuente de alimentación de Hardware**

- Posibilidad de alimentación del sistema preparada para una a cuatro baterías de iones de litio de 18 V en paralelo.
- Hasta 10 horas de autonomía con cuatro baterías, una batería permite una autonomía de aproximadamente 2,5 horas.
- 24 V mediante interfaz I/O hasta 2 A.
- 2 conexiones de enchufe de 12 V hasta 2 A.
- Zócalo USB A (5 V).

### **Extensión de montaje**

- Material: Chasis redondo de acero inoxidable.
- Torre de montaje de acero inoxidable con varios dispositivos de montaje y preparada para la guía de cables interna.
- Plataformas de montaje de posicionamiento flexible (120° cada una).

El robot originalmente cuenta con una cámara incluida en la base pero bajo las condiciones de este proyecto no se consideró necesario su uso, en la figura [6.1](#) se observa una imagen de la plataforma Robotino y en [6.1](#) se muestra una imagen del robot utilizado para este desarrollo.

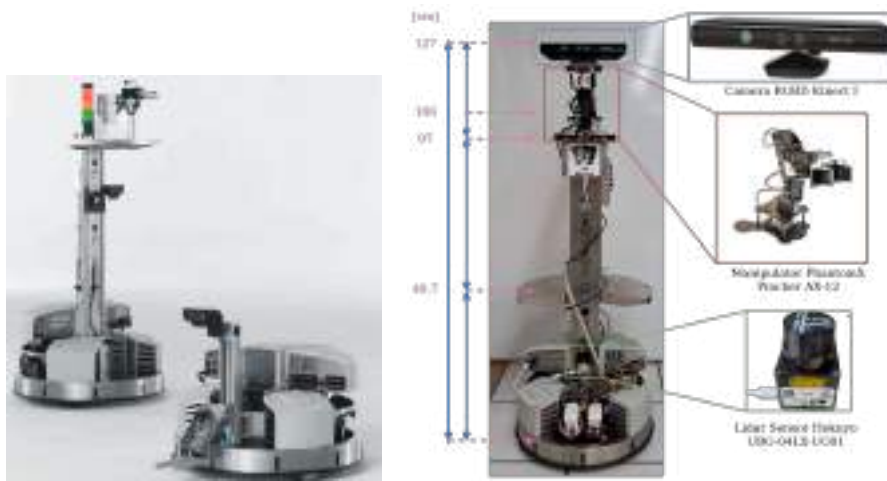


Figura 6.1: Robotino 3 Didactic, 2013

Robotino 3 modificado

### Camara RGBD - Kinect

El autor Vangos Pterneas, 2022, ganador del reconocimiento *Microsoft Most Valuable Professional (2014-2019)*, realiza una descripción del dispositivo Azure Kinect (Microsoft, 2019). Así como un análisis de su impacto en la tecnología actual y las herramientas de hardware que le permiten a este aparato ser tan ampliamente utilizado, dando también una descripción de sus antecesores, los dispositivos *HoloLens*, *Kinect versión 2* y *Kinect versión 1*. El autor narra que en 2009 Microsoft lanzó al mercado el dispositivo llamado *Project Natal*, cambiando el nombre a **Kinect** en 2010 y distribuyéndolo como un accesorio para la consola de videojuegos *XBOX 360*. Este dispositivo es capaz de reconocer las partes del cuerpo de las personas y entender sus voces en tiempo real, creando una interfaz interactiva que permitía a los usuarios utilizar la consola sin necesitar un control físico. Para el proyecto que este documento describe, se utilizó la versión 1 del Kinect, mostrado en la figura 6.2, donde también se observan los componentes del dispositivo.

- Sensor de Profundidad:

Kramer, 2012, especifica que el dispositivo Kinect utiliza una cámara MT9M001 (ver figura 6.3) de la compañía Micron (2004), siendo esta una cámara monocromática con un arreglo de  $1280 \times 1024$  píxeles, aunque estas dimensiones son modificadas antes de la reducción de resolución que se realiza posteriormente.

Por su parte Davison, 2012, describe sus especificaciones, explicando que el sensor de profundidad del Kinect consiste en una fuente de luz infrarroja que proyecta un patrón de puntos, los cuales son leídos después por un sensor infrarrojo CMOS, que detecta los segmentos reflejados del patrón de puntos, además de convertir sus intensidades a distancias.



Figura 6.2: Kinect versión 1 - Componentes

Kramer, 2012, añade que el sensor infrarrojo crea un patrón ruidoso de luz infrarroja estructurada de 830 [nm]. El rango de profundidad del sensor tiene un alcance de 50 [cm] a 3 [m], con una resolución de alrededor de 1 [cm] sobre el eje Z, mientras que la resolución espacial (sobre los ejes X e Y) se encuentra en el orden de los milímetros. Davison, 2012, también explica que cada cuadro generado por el sensor de profundidad cuenta con una resolución de  $(640 \times 480)$  píxeles, siendo capaz de mostrar 11 valores de profundidad de 11 bits, dando como resultado 2048 niveles de sensibilidad.



Figura 6.3: Proyector Infrarrojo y Sensores CMOS 2010

- Cámara de video RGB:

De acuerdo con Kramer, 2012 el dispositivo Kinect contiene una cámara RGB que, según Bob Widenhofer (2010), utiliza un sensor MT9M112 (ver figura 6.3). Kramer y su equipo especifican que la cámara opera a 30 [Hz] enviando imágenes de  $640 \times 512$  píxeles. Sin embargo, es posible utilizar una configuración *High Resolution*, que opera a una frecuencia de 15 cuadros por segundo, con imágenes de  $1280 \times 1024$  píxeles. Los autores mencionan también que la cámara cuenta con balance de blancos

---

<b>Nube de Puntos</b>	
Rango de Profundidad	0.8 [m] a 3.5 [m]
Resolución de la nube de puntos	640 × 480 píxeles
Niveles de profundidad	2048
Frecuencia de salida - Nube de puntos	30 [Hz]
Campo de visión horizontal	58 °
Campo de visión vertical	45 °
Campo de visión diagonal	70 °
Campo de visión horizontal	58 °
<b>Cámara RGB</b>	
Modo <i>Low Resolution</i>	
Frecuencia de publicación	30 [Hz]
Resolución de Imagen	640 × 512
Modo <i>High Resolution</i>	
Frecuencia de publicación	15 [Hz]
Resolución de Imagen	1280 × 1024
<b>Micrófonos (4)</b>	
Resolución	16 bits
Frecuencia de muestreo	16 [KHz]

Cuadro 6.1: Especificaciones - Kinect V1

automático, prevención de parpadeo, saturación de color, referencia de negros y corrección de defectos. Los autores también resaltan que ninguno de los dos dispositivos anteriores es de verdadesa utilizada a menos de que se encuentren correctamente calibrados.

### Manipulador PhantomX Pincher AX-12

El manipulador utilizado para este proyecto es el brazo robótico **PhantomX Pincher AX-12** distribuido por la empresa *Trossen Robotics (Interbotix)*, conocida por poner a disposición del público hardware y software *open source* utilizado en robótica para uso educativo, profesional o de entretenimiento. Los dispositivos de Interbotix se encuentran contruidos con servomotores DYNAMIXEL X-Series, y se pueden obtener en configuraciones de 4, 5 o 6 grados de libertad, y es posible controlarlos mediante el Middleware ROS («Interbotix X-Series Arms — Interbotix X-Series Arms Documentation», 2022).

De acuerdo con la información del fabricante (2022), este dispositivo es un brazo robótico de 4 grados de libertad, diseñado para ser compatible con la plataforma robótica *TurtleBot* de ROS y cuenta con los elementos necesarios para usarlo ya sea como un elemento independiente o bien, como en este caso, integrarlo dentro de otra plataforma robótica, en la figura 6.4 se muestra un ejemplo del manipulador.



---

### Características

- Actuadores Dynamixel AX-12A.
- Base sólida con rodamiento de agujas. Construcción resistente en Delrin/acrílico.
- Controlador robótico Arbotix para procesamiento integrado.
- Pinza paralela personalizable.
- Soportes de montaje para cámaras y sensores.

### Especificaciones

- Peso: 550 [g].
- Alcance Vertical: 35 [cm].
- Alcance Horizontal: 31 [cm].
- Fuerza:
  - 25[cm]/ 40[g].
  - 20[cm]/ 70[g].
  - 15[cm]/ 100[g].
- Fuerza de agarre de la pinza: 500 [g].
- Fuerza de levantamiento de la muñeca: 250[g].

## 6.2. Implementación

Para lograr la integración de todos los elementos anteriormente mencionados, los algoritmos en los que se procesa la imagen se realizan utilizando scripts de python, mientras que las máquinas de estados se programaron en C++.

La evaluación del resultado de esta implementación se dio parcialmente durante la competencia RoboCup 2023, donde se cuenta con las estaciones de trabajo y el set de piezas completas. Durante esta evaluación bajo condiciones reales fue posible observar debilidades en la implementación que fueron cubiertas posteriormente, la evaluación se continuó entonces en las instalaciones del laboratorio de Biorobótica.

Como se ha mencionado anteriormente, se obtuvieron las medias de color de las piezas bajo diferentes condiciones de iluminación. En la figura 6.5 se muestran los colores correspondientes a dichas medias, almacenadas en un archivo yaml desde el cual el segmentador accede una vez habiéndosele indicado qué color de pieza se requiere.



Figura 6.4: PhantomX Pincher AX-12 (2022)

Durante las pruebas realizadas, se observó que existe la posibilidad de confusión entre las piezas amarilla y verde. Esta confusión se asocia a la cercanía de los colores en el mapa de colores correspondiente al espacio HSV usado por OpenCV. Es notable la poca diferencia entre el valor *Hue* de ambas piezas (ver figura 6.6)

De acuerdo a la figura y a los valores obtenidos de las medias, existe un traslape entre las zonas que describen las características de color de ambos colores, mientras que para los demás elementos de interés, los cambios de magnitud son notablemente mayores. Para controlar este factor se utilizó un elemento *delta* dentro de la segmentación, afectando los umbrales de los valores HSV.

De esta manera, al obtener los píxeles de la imagen que pertenecen al objeto de interés, se realiza un promedio de los mismos, mediante el cual se obtiene la coordenada promedio de la posición del objeto respecto a la cámara y realizando transformaciones homogéneas se obtiene su posición respecto a la base del manipulador del robot, luego de lo cual se solicita al planeador de movimientos la trayectoria a realizar para lograr el graspeo.

Como se mencionó anteriormente, el planeador de trayectoria obtiene las posiciones angulares que deben ser alcanzadas por los motores para lograr la posición deseada. De esta forma se establece como objetivo la coordenada correspondiente al centroide de la pieza, el robot abre la pinza y realiza el movimiento de graspeo. Al encontrarse en la posición deseada, se cierra la pinza y se regresa a la posición de descanso del robot, desde la cual le es posible navegar al punto en que debe entregar la pieza nuevamente.

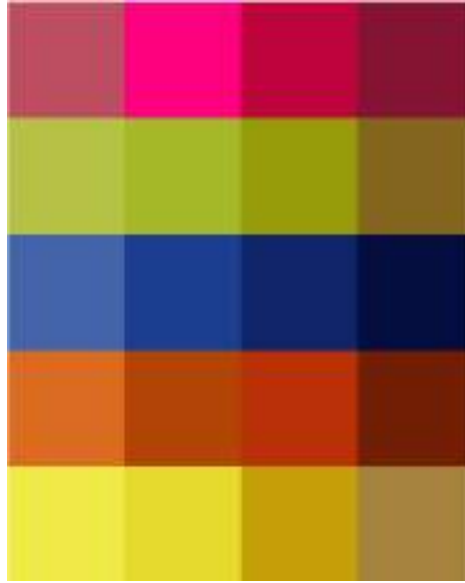


Figura 6.5: Medias de color

Pieza	Experimentos exitosos	Porcentaje de detección
Azul	29	82.85
Amarillo	26	74.28
Naranja	27	77.14
Rojo	30	85.71
Verde	29	82.85

Cuadro 6.2: Resultados - Detección de piezas

### 6.2.1. Resultados - Visión

La evaluación del algoritmo se realizó utilizando el flujo de video obtenido de la cámara Kinect, conectado al robot y mostrando las coordenadas de los objetos buscados. En estas pruebas el objeto de interés se encontraba a una distancia horizontal mínima de 40 [cm] respecto al eje del kinect, esto debido a las limitaciones de la nube de puntos de la cámara. el experimento se realizó en 35 repeticiones, variando las posiciones de las piezas dentro del espacio de trabajo, así como la posición del robot respecto al mismo. Adicionalmente, se colocaba más de un elemento de interés en el rango visual del robot. Después de las modificaciones realizadas al parámetro delta del segmentador, el comportamiento en que se confundían piezas amarillas con piezas verdes no volvió a presentarse.

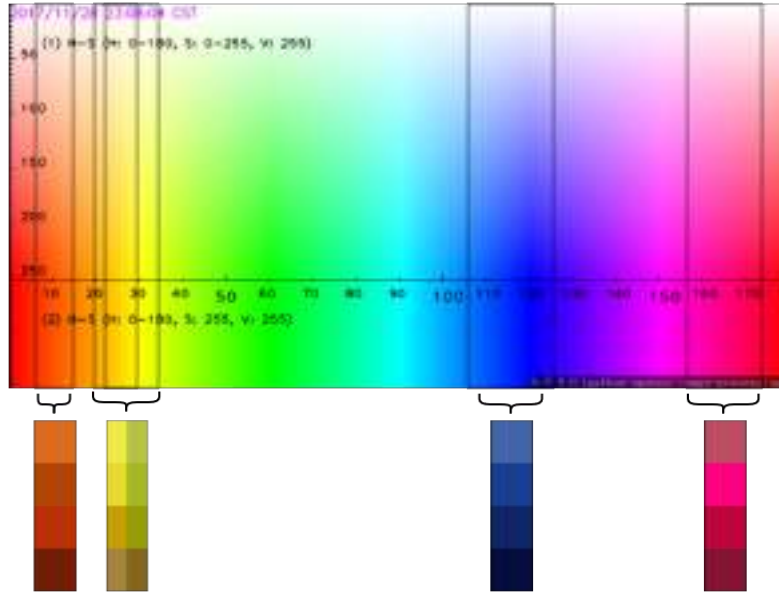


Figura 6.6: OpenCV - espacio HSV - mapa de color (Pai, 2022)

### 6.2.2. Resultados - Manipulación

De acuerdo a lo establecido anteriormente, la planeación de la estrategia de movimientos se lleva a cabo a través de la paquetería MoveIt, para la cual existen ya herramientas para el uso del manipulador, dentro de lo cual se cuenta con la información de la configuración mecánica el robot y sus características físicas, con dicha información se realiza la planeación de las trayectorias. Durante el proceso de prueba se observó que el manipulador puede permanecer en posición extendida cierto periodo de tiempo antes de que el motor en la primera articulación ceda ante el peso de la estructura, por lo que se definió una posición predeterminada de reposo en la cual el mecanismo del manipulador no se ve forzado (ver figura 6.8).

Una vez que se cuenta con las coordenadas que el proceso de visión ha encontrado de las piezas de interés respecto a la imagen, se realiza la transformación homogénea para encontrar las correspondientes teniendo el origen del manipulador como referente. Estas coordenadas son enviadas a MoveIt y, después de planear la trayectoria, se realiza el movimiento. Las estrategias propuestas por MoveIt pueden variar dependiendo de la posición inicial en que se encuentre el manipulador. Sin embargo, dadas las condiciones en las que se encuentran las piezas de interés en el entorno planteado, las coordenadas en las cuales se tendría la mayor variación son  $x$  e  $y$ , ya que la altura es constante respecto a las estaciones de trabajo.

Para la integración de este comportamiento en la estructura general, se creó

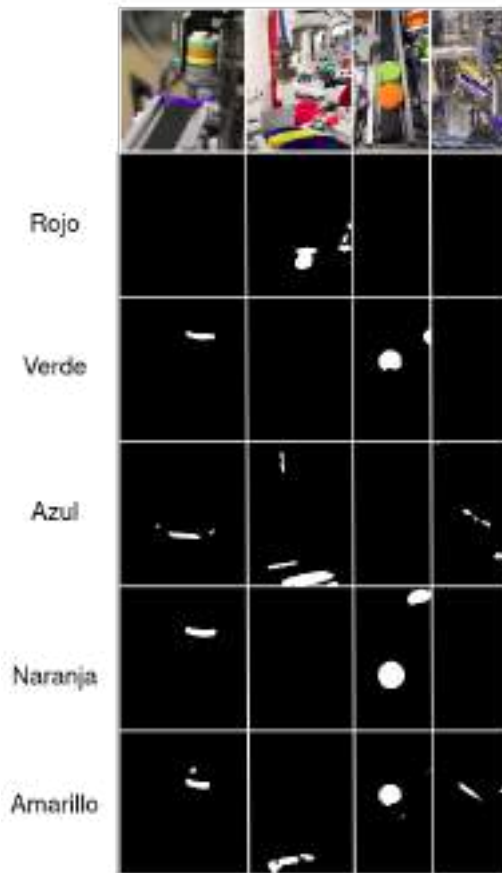


Figura 6.7: Piezas segmentadas

un servicio de ROS, mediante el cual los movimientos del brazo pueden ser accésados únicamente en el momento en que la máquina de estados lo solicita. Al servicio se le ingresa la coordenada de la pieza respecto a la base y una variable de confirmación del movimiento se recibe como resultado. en la figura 6.8 a) se puede observar la visualización que se obtiene del planeador de MoveIt dentro de la herramienta Rviz. En este ejemplo, se observa en color gris la pose original y en color naranja la posición deseada del manipulador, como parte de esta visualización se genera una animación de la trayectoria de movimiento solicitada, antes de ejecutarla.

Una de las ventajas que ofrece el uso de este manipulador se encuentra en la versatilidad de las posiciones que alcanza, pudiendo resisitir cambios en los 3 componentes de las coordenadas sin modificar la posición de la base del robot y utilizando los actuadores disponibles en su estructura, lo que lo haría capaz no solo de manipular los objetos que se encuentren en el centro de la banda trans-

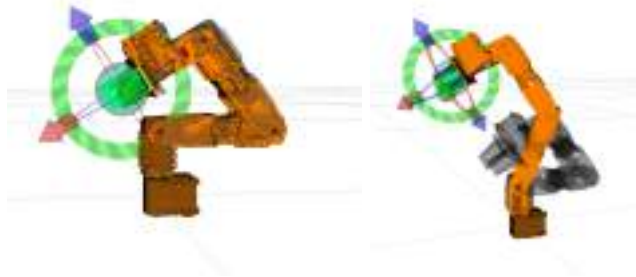


Figura 6.8: Izq) Posición de reposo - Der) posición final

portadora, sino en cualquier otra superficie alcanzable de las estaciones, como pueden ser las plataformas de entrega y las rampas en las cuales se deshechan los elementos que no son de interés para la operación.

La secuencia de movimientos propuesta para la tarea que se describe consiste en cuatro pasos, sin contar el inicio y vuelta a la posición de reposo. Que fueron planteados debido a las limitaciones con las que cuenta el manipulador. Esta limitación consiste en que para ciertas posiciones de  $x$  o  $y$ , el manipulador debe comenzar desde una coordenada mínima  $z = 0,12$ . Adicionalmente, para el accionamiento de la pinza, es necesario bloquear los movimientos del manipulador y enviar una variable booleana, *True* para abrir y *False* para cerrar.

A continuación se describe la secuencia utilizada, que puede ser observada también en la figura 6.7.

### Pasos preestablecidos

#### *Reposo:*

```
arm_srv.request.manipBlocker = false
arm_srv.request.x = 0.025
arm_srv.request.y = 0.0
arm_srv.request.z = 0.10
arm_srv.request.gripperState = false
```

#### *Pinza abierta:*

```
arm_srv.request.manipBlocker = true
arm_srv.request.x = 0.025
arm_srv.request.y = 0.0
arm_srv.request.z = 0.10
arm_srv.request.gripperState = true
```

---

*Inicial:*

```
arm_srv.request.manipBlocker = false
arm_srv.request.x = 0.04
arm_srv.request.y = 0.0
arm_srv.request.z = 0.12
arm_srv.request.gripperState = false
```

*Trayectoria de graspeo:*

```
arm_srv.request.manipBlocker = false
arm_srv.request.x = det_piece.pose.position.x
arm_srv.request.y = det_piece.pose.position.y
arm_srv.request.z = det_piece.pose.position.z
arm_srv.request.gripperState = true
```

*Pinza cerrada*

```
arm_srv.request.manipBlocker = true
arm_srv.request.x = det_piece.pose.position.x
arm_srv.request.y = det_piece.pose.position.y
arm_srv.request.z = det_piece.pose.position.z
arm_srv.request.gripperState = true
```

*Reposo*

```
arm_srv.request.manipBlocker = false
arm_srv.request.x = 0.025
arm_srv.request.y = 0.0
arm_srv.request.z = 0.10
arm_srv.request.gripperState = false
```



Figura 6.9: Secuencia de graspeo

Debido a que en la posición de reposo la pinza obstruye la cámara del dispositivo Kinect, fue necesario determinar una posición del brazo desde la cual

---

no interfiriera en la detección de los objetos. Para lo anterior se especificaron las posiciones angulares que debe tener cada uno de los actuadores del brazo.

#### **Posición *Kinect\_hide***

```
group.setPlanningTime(4.0);  
group.setJointValueTarget('arm_shoulder_pan_joint', 2.40);  
group.setJointValueTarget('arm_shoulder_lift_joint', 1.32);  
group.setJointValueTarget('arm_elbow_flex_joint', 0.31);  
group.setJointValueTarget('arm_wrist_flex_joint', 0.61);  
group.asyncMove();
```



Figura 6.10: Posición *Kinect\_hide*

### **6.2.3. Resultados - Integración**

Al proceder con la integración fue necesario hacer la unión de los modelos del robot y del brazo para poder operarse simultáneamente y que ambas partes se reconozcan una a la otra y puedan interactuar entre sí. Para lo cual se hizo una combinación de los archivos que los describen, una vez que el modelo es correcto y coincide en posiciones y orientaciones con las condiciones reales del robot se procedió a probar los algoritmos que se plantearon anteriormente.

Anteriormente se mencionó de forma breve que al realizar estas pruebas conjuntas, y dado que en la estructura de la máquina de estados se definía que se realizarían secuencialmente los movimientos de base del robot, del manipulador y la ejecución del algoritmo de visión, se obtiene un error debido a la vibración que genera el manipulador en la estructura sobre la cual se encuentra el Kinect.



---

---

## Capítulo 7

# Conclusiones y Trabajo Futuro

### 7.1. Conclusiones

#### 7.1.1. Visión Computacional

El módulo de segmentación de objetos es capaz de localizar entre el 70 y 80 % de las veces los objetos de interés en las pruebas realizadas utilizando las imágenes obtenidas del dispositivo Kinect (640x512 píxeles). Para la obtención de las medias de color de cada pieza fueron utilizadas cuatro imágenes con diferentes iluminaciones y se tomó la muestra de zonas más representativas de cada cambio de iluminación.

El segmentador fue probado tanto en imágenes estáticas obtenidas de videos de ejecuciones de la competencia como con el flujo de video obtenido de la ejecución de los algoritmos con el robot.

Bajo estas condiciones, en 13 de los experimentos, correspondientes al 7 %, la pieza fue correctamente segmentada, pero la coordenada asociada no coorespondía con la posición real del objeto. De acuerdo a esto, el módulo obtiene exitosamente la posición de la pieza en la mayoría de las ocasiones, 93 %, entregando la coordenada asociada para la posterior manipulación de los objetos cuando las pruebas se realizan sin la integración de los movimientos del brazo.

Uno de los motivos que llevaron a la decisión de utilizar segmentación por color como la técnica indicada para este proyecto se basa en lo ligero de la implementación, siendo que en ejecución es imperceptible algún retraso en el sistema cuando se utilizan ya sea el sistema de detección o la localización de los objetos, por lo que no representa una afectación en el tiempo de ejecución del proceso general.

---

A partir de las condiciones mencionadas, se concluye que la implementación de este módulo cumple con los objetivos de "Procesar la imagen obtenida por una cámara RGB en la estructura de un robot para identificar objetos en el espacio de trabajo de acuerdo a sus características." Utilizar la nube de puntos obtenida de la cámara de profundidad para conocer la posición relativa de los objetos y plataformas de interés y planear la trayectoria óptima del manipulador para tomarlos..<sup>a</sup>unque los resultados pueden ser mejorables.

### 7.1.2. Manipulación

El módulo encargado de la manipulación del robot tiene éxito en aproximadamente 70 % de los experimentos, a partir de los cuales se observó la necesidad de integrar adicionalmente un sistema con el cual sea posible la calibración de los movimientos finos del manipulador, debido a que las características del mismo propiciaban en ocasiones el movimiento accidental de la pieza como parte de la trayectoria de manipulación.

En las pruebas realizadas de visión y manipulación combinadas, se encontró que los movimientos que realiza el manipulador producen vibraciones en la estructura del robot. Esta vibración genera un ligero desajuste en la posición del kinect respecto al robot, y por consiguiente se obtiene un error en las mediciones de posición de la cámara, dado que el movimiento del dispositivo no es registrado dentro del modelo del robot y después de varias repeticiones del experimento, lo que comenzó como variaciones imperceptibles se convertía en cambios milimétricos y, dado el tamaño de la pinza del manipulador y de los objetos mismos generaban errores en el graspeo.

Por lo anterior descrito, se considera que el objetivo "Identificar los objetos de interés presentes dentro del espacio de trabajo de un robot de servicio para fábricas inteligentes y sujetarlos exitosamente con el manipulador del robot." se cumplió parcialmente, dado que el objeto es exitosamente identificado, pero la estrategia de movimiento debe incluir movimientos menos abruptos y un posicionamiento más firme del dispositivo Kinect o la cámara que se busque utilizar.

## 7.2. Trabajo Futuro

Como trabajo futuro se propone lo siguiente:

- Visión computacional:
  - Se considera posible la implementación de un sistema que discrimine los objetos de interés integrando más características representativas como pueden ser la silueta o el tamaño del objeto, haciendo un uso más extenso de la nube de puntos disponible.
  - Aumentar la cantidad de muestras tomadas para la obtención de las medias de color de los objetos.

---

■ Manipulación:

- Reducir el número de grados de libertad con que cuenta el manipulador, dado que las capacidades de movilidad del Phantom Pitcher exceden las necesidades actuales y aumentan la complejidad del cálculo de la trayectoria.
- Calcular la cinemática inversa del robot sin el uso de la paquetería MoveIt, lo que facilitaría la personalización del cálculo de trayectoria y la modificación del modelo del manipulador en caso de ser necesario.

---

---

# Bibliografía

- Kohout, P., De Bortoli, M., Ludwiger, J., Ulz, T. & Steinbauer, G. (2020). A multi-robot architecture for the RoboCup Logistics League. *Elektrotech. Inftech.*, 137(6), 291-296. <https://doi.org/10.1007/s00502-020-00826-5>
- Navarro Piña, A. V. (2021). *Robot Industrial: manual de instalación* [OCLC: 1236395081]. Madrid, Paraninfo.
- Basco, A. I., Beliz, G., Coatz, D. & Garnero, P. (2018). *Industria 4.0: Fabricando el Futuro* [Google-Books-ID: geiGDwAAQBAJ]. Inter-American Development Bank.
- Roveda, L., Maroni, M., Mazzuchelli, L., Praolini, L., Shahid, A. A., Bucca, G. & Piga, D. (2022). Robot End-Effector Mounted Camera Pose Optimization in Object Detection-Based Tasks. *Journal of Intelligent & Robotic Systems*, 104(1), 16. <https://doi.org/10.1007/s10846-021-01558-0>
- Roveda, L., Magni, M., Cantoni, M., Piga, D. & Bucca, G. (2021). Human-robot collaboration in sensorless assembly task learning enhanced by uncertainties adaptation via Bayesian Optimization. *Robotics and Autonomous Systems*, 136, 103711. <https://doi.org/https://doi.org/10.1016/j.robot.2020.103711>
- Rosenbaum, D. A., Cohen, R. G., Meulenbroek, R. G. J. & Vaughan, J. (2006). Plans for Grasping Objects. En M. L. Latash & F. Lestienne (Eds.), *Motor Control and Learning* (pp. 9-25). Boston, MA, Springer US. [https://doi.org/10.1007/0-387-28287-4\\_2](https://doi.org/10.1007/0-387-28287-4_2)
- Sun, Y., Falco, J., A. Roa, M. & Calli, B. (2022). Research Challenges and Progress in Robotic Grasping and Manipulation Competitions [Publisher: IEEE]. *IEEE Robotics and Automation Letters*, *Robotics and Automation Letters*, *IEEE*, *IEEE Robot. Autom. Lett.*, 7(2), 874-881. <https://doi.org/10.1109/LRA.2021.3129134>
- RoboCup & Mathworks. (2022). RoboCup ARM Challenge. <https://arm.robocup.org/home>
- Babel, W. (2022). *Industry 4.0, China 2025, IoT: the hype around the world of automation*. Wiesbaden [Heidelberg], Springer.
- IFR. (2020). International Federation of Robotics. Consultado el 12 de agosto de 2022, desde <https://ifr.org/service-robots>
- Freire, P. M. (2007). *La Importancia del Conocimiento. Filosofía y Ciencias Cognitivas*. Netbiblo.

- 
- Szeliski, R. (2022). *Computer Vision: Algorithms and Applications* (Second edition). Cham, Springer Nature Switzerland AG.
- Gonzalez, R. C. & Woods, R. E. (2002). *Digital image processing* (2nd ed). Upper Saddle River, N.J, Prentice Hall.
- Pratt, W. K. (2014). *Introduction to digital image processing* [OCLC: 907445364]. Boca Raton [Florida], CRC Press, Taylor & Francis Group.
- Czerwinski, R. & Kania, D. (2013). Definitions and Basic Properties [Series Title: Lecture Notes in Electrical Engineering]. En *Finite State Machine Logic Synthesis for Complex Programmable Logic Devices* (pp. 9-23). Series Title: Lecture Notes in Electrical Engineering. Berlin, Heidelberg, Springer Berlin Heidelberg. [https://doi.org/10.1007/978-3-642-36166-1\\_2](https://doi.org/10.1007/978-3-642-36166-1_2)
- Burger, W. & Burge, M. J. (2022). *Digital image processing: an algorithmic introduction* (Third edition). Cham, Switzerland, Springer.
- Sonka, M., Hlaváč, V. & Boyle, R. (2008). *Image processing, analysis, and machine vision* (3. ed). Toronto, Thompson Learning.
- Treiber, M. A. (2010). *An Introduction to Object Recognition*. London, Springer. <https://doi.org/10.1007/978-1-84996-235-3>
- Giarratano & Riley. (2002). *Zhuan jia xi tong yuan li yu bian cheng: Third Edition = Expert Systems Principles and Program ming* [OCLC: 952716560]. Beijing, Ji xie gong ye chu ban she.
- Gupta, I. & Nagpal, G. (2020). *Artificial Intelligence and Expert Systems* [Google-Books-ID: 61PdDwAAQBAJ]. Mercury Learning; Information.
- Liebowitz, J. (2019). *The Handbook of Applied Expert Systems* [Google-Books-ID: WzX3DwAAQBAJ]. CRC Press.
- Ceccarelli, M. (2022). *Fundamentals of mechanics of robotic manipulation* (Second edition). Cham, Springer.
- Furbaß, L., Knoflach, L., Kohout, P., De Bortoli, M., Moser, S., Steinbauer-Wagner, G., Masiero, A., Frick, T. & Furhapter, D. (2021). RoboCup Logistics League Graz Robust and Intelligent Production System GRIPS.
- Davison, A. (2012). *Kinect open source programming secrets: hacking the Kinect with OpenNI, NITE, and Java*. New York, McGraw-Hill.
- Lynch, K. M. & Park, F. C. (2017). *Modern robotics: mechanics, planning, and control* [OCLC: ocn983881868]. Cambridge, UK, Cambridge University Press.
- Asada, H. & Slotine, J.-J. E. (1986). *Robot analysis and control*. New York, N.Y, J. Wiley.
- Gross, D. (2013). *Statics* (Second edition). Berlin, Springer.
- Olguín Díaz, E. (2019). *3D motion of rigid bodies: a foundation for robot dynamics analysis*. Cham, Springer.
- Hamilton, W. R. (1844). II. *On quaternions; or on a new system of imaginaries in algebra. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 25(163), <https://doi.org/10.1080/14786444408644923>, 10-13. <https://doi.org/10.1080/14786444408644923>
-

- 
- Siciliano, B. & Khatib, O. (Eds.). (2016). *Springer handbook of robotics* (2nd edition). Berlin Heidelberg, Springer.
- tf - ROS Wiki. (2019). Consultado el 21 de noviembre de 2023, desde <http://wiki.ros.org/tf>
- Technical Committee 2012–2022. (2022). RoboCup Logistics League Rules and Regulations 2022, 48. Consultado el 23 de noviembre de 2022, desde <https://ll.robocup.org/the-rcll-competition/page-3-2/>
- Quigley, Morgan, Gerkey, Brian, Conley, Ken, Faust, Josh, Foote, Tully, Leibs, Jeremy, Wheeler, Rob & Ng, Andrew. (2009). ROS: an open-source Robot Operating System. *ICRA Workshop on Open Source Software*, 3.
- std\_msgs. (2023). Consultado el 13 de octubre de 2023, desde [http://wiki.ros.org/std\\_msgs](http://wiki.ros.org/std_msgs)
- Vázquez Silva, E. d. J. & Negrete, M. (2018). *Desarrollo de un sistema de detección y manipulación de objetos para un robot de servicio* (Tesis doctoral). Universidad Nacional Autónoma de México. Ciudad Universitaria, CDMX, México.
- About - OpenCV. (2023). Consultado el 24 de enero de 2024, desde <https://opencv.org/about/>
- Arévalo, V. M., González, J. & Ambrosio, G. (2002). LA LIBRERÍA DE VISIÓN ARTIFICIAL OPENCV APLICACIÓN A LA DOCENCIA E INVESTIGACIÓN.
- Concepts — MoveIt. (2023). Consultado el 22 de noviembre de 2023, desde <https://moveit.ros.org/documentation/concepts/>
- Festo Didactic InfoPortal. (2023). Consultado el 24 de enero de 2024, desde <https://ip.festo-didactic.com/InfoPortal/Robotino/Software/Programming/EN/RobotinoView.html>
- Didactic, F. (2013). Robotino® - Plataforma de robot móvil para investigación y formación. FESTO Didactic. Consultado el 30 de noviembre de 2023, desde <https://www.lagos.udg.mx/sites/default/files/adjuntos/robotino.pdf>
- Pterneas, V. (2022). *Mastering the Microsoft Kinect: Body Tracking, Object Detection, and the Azure Cloud Services*. Berkeley, CA, Apress. <https://doi.org/10.1007/978-1-4842-8070-6>
- Kramer, J. (Ed.). (2012). *Hacking the Kinect*. New York, Apress.
- Micron. (2004). 1/2-Inch Megapixel CMOS Digital Image Sensor.
- EETimes. (2010). Inside Xbox 360's Kinect controller. Consultado el 31 de enero de 2024, desde <https://www.eetimes.com/inside-xbox-360s-kinect-controller/>
- Interbotix X-Series Arms — Interbotix X-Series Arms Documentation. (2022). Consultado el 9 de octubre de 2023, desde [https://docs.trossenrobotics.com/interbotix\\_xsarms.docs/](https://docs.trossenrobotics.com/interbotix_xsarms.docs/)
- WidowX MKII Robotic Arm - Download Free 3D model by Interbotix - Sketchfab. (2022).
- Robotic Arms. (2022). Consultado el 27 de febrero de 2024, desde <https://www.interbotix.com/Robotic-Arms>
-



---

Pai, A. Y. (2022). AdityaPai2398/Colour-Segmentation-in-OpenCV. Consultado el 26 de febrero de 2024, desde <https://github.com/AdityaPai2398/Colour-Segmentation-in-OpenCV>